Octopus Cloud Service

用户手册

文档版本 01

发布日期 2025-03-21

华为云计算技术有限公司

商标声明


和其他华为商标均为华为技术有限公司的商标。
本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受华为云计算技术有限公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，华为云计算技术有限公司对本文档内容不做任何明示或暗示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

华为云计算技术有限公司

地址： 贵州省贵安新区黔中大道交兴功路华为云数据中心 邮编：550029

网址： https://www.huaweicloud.com/

目 录

1 准备工作

1.1 注册华为账号并开通华为云

在使用华为云服务之前您需要申请华为云账号并进行实名认证。通过此账号，您可以使用所有华为云服务，并且只需为您所使用的服务付费。

如果您已有一个华为云账号，请跳到下一个任务。如果您还没有华为云账号，请参考以下步骤创建。

步骤1 打开**华为云官网**，单击“注册”。

步骤2 根据提示信息完成注册，详细操作请参见**注册华为账号并开通华为云**。

注册成功后，系统会自动跳转至您的个人信息界面。

步骤3 参考**实名认证**完成个人或企业账号实名认证。

----结束

1.2 创建用户并授权使用服务

如果您需要对您所拥有的Octopus进行精细的权限管理，您可以使用**统一身份认证服务**（Identity and Access Management，简称IAM），通过IAM，您可以：

根据企业用户的职能，设置不同的访问权限，以达到用户之间的权限隔离。

如果华为云账号已经能满足您的要求，不需要创建独立的IAM用户，您可以跳过本章节，不影响您使用Octopus服务的其它功能。

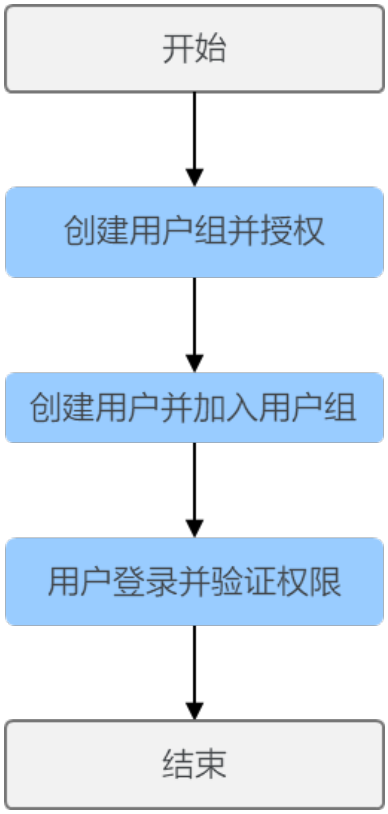
本章节为您介绍对用户授权的方法，操作**流程图**所示。

前提条件

用户组授权之前，请您了解用户组可以添加的Octopus权限，并结合实际需求进行选择，Octopus支持的系统权限，请参见：**Octopus权限管理**。如果您需要对除Octopus之外的其它服务授权，IAM支持服务的所有权限请参见**系统权限**。

示例流程

图 1-1 为用户授权流程图



1. **创建用户组并授权**

   在IAM控制台创建用户组，并授予Octopus服务Octopus FullAccess、BSS Administrator、IAM ReadOnlyAccess等权限。

2. **创建用户并加入用户组**

   在IAM控制台创建用户，并将其加入**1**中创建的用户组。

3. **用户登录**并验证权限

   新创建的用户登录控制台，切换至授权区域，验证权限。

1.3 创建对象存储服务

Octopus云服务使用对象存储服务（Object Storage Service，简称OBS）进行数据存储以及模型的备份和快照，实现安全、高可靠和低成本的存储需求。因此，建议您在使用Octopus云服务之前先创建一个OBS桶，然后在OBS桶中创建文件夹用于存放数据。

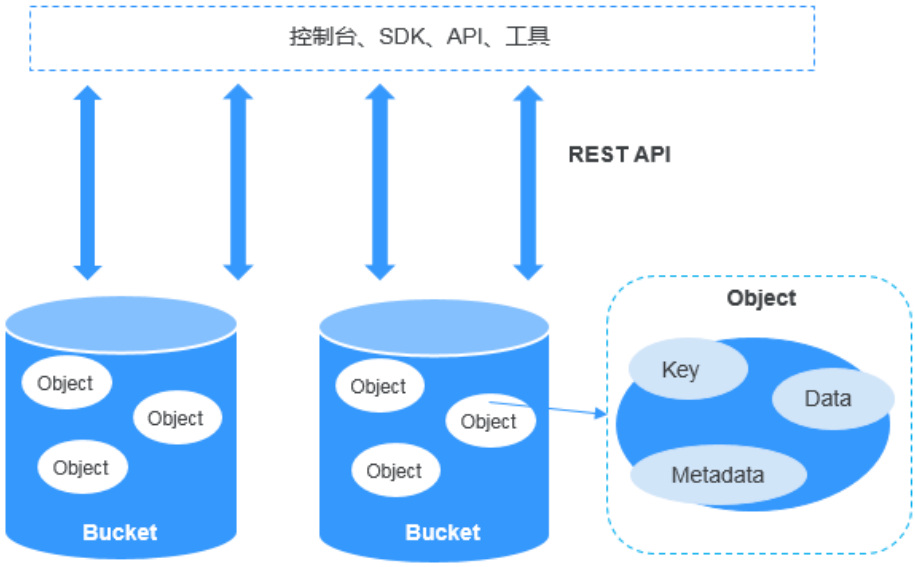
OBS 简介

对象存储服务OBS是一个基于对象的海量存储服务，为客户提供海量、安全、高可靠、低成本的数据存储能力。对象存储服务OBS的基本组成是桶和对象。桶是OBS中存储对象的容器，每个桶都有自己的存储类别、访问权限、所属区域等属性，用户在互联网上通过桶的访问域名来定位桶。对象是OBS中数据存储的基本单位。

对Octopus云服务来说，OBS服务是一个数据存储中心，因为Octopus云服务本身目前没有数据存储的功能。在开发过程中的输入数据、输出数据、中间缓存数据都可以在OBS桶中进行存储、读取。

因此，在使用Octopus云服务之前您需要创建一个OBS桶，然后在OBS桶中创建文件夹用于存放数据。

图 1-2 对象存储服务 OBS



1. 登录**OBS管理控制台**，在桶列表页面右上角单击“创建桶”，创建OBS桶。

   说明

   创建桶的区域需要与Octopus云服务所在的区域一致。

2. 在桶列表页面，单击桶名称，进入该桶的概览页面。
3. 单击左侧导航的“对象”，在对象页面单击“新建文件夹”，创建OBS文件夹。具体请参见**创建文件夹**章节。
4. 复制OBS桶路径

   单击对象文件夹名称，进入桶内，复制对象名称，去掉前面的 “obs:/” ，即为复制OBS桶路径。

图 1-3 复制 OBS 桶路径

1.4 申请公测

介绍用户如何对有公测活动的产品或者服务申请公测。

> **说明**
>
> 当前自动驾驶云服务仅支持专属区域“华北-汽车一”，如果您需要使用自动驾驶云服务，可以参考**提交工单**尝试申请专属区域“华北-汽车一”访问权限。
>
> 自动驾驶云服务在区域“华南-广州”提供了OpenAPI，您也可以通过调用API，访问自动驾驶云服务。详情请参见自动驾驶云服务**API参考**。

操作步骤

步骤1 登录**华为云**首页。

步骤2 单击页面右上角的“控制台”。

步骤3 在控制台页面，选择自动驾驶云服务 Octopus 支持的区域。

步骤4 单击左上角  展开服务列表，在搜索栏中输入“八爪鱼自动驾驶云服务 Octopus”。

步骤5 单击服务名称进入八爪鱼自动驾驶云服务 Octopus管理控制台。

步骤6 单击“立即申请”。

步骤7 根据实际情况设置企业规模、研发人员比例、应用场景、业务当前阶段等信息。

> **说明**
>
> 用户第二次申请公测时，可以继承上一次申请该产品公测活动所填写的必填项信息。

步骤8 选中“同意《 公测试用服务协议 》”。

步骤9 单击“申请公测”，完成公测申请。

等待运营开通公测权限。单击“前往公测管理”，可以查看已经申请的公测产品或者服务。

----结束

1.5 购买开通 Octopus 服务

1.5.1 开通 Octopus 服务概述

自动驾驶云服务（ Octopus ）是由数据资产、数据合规、数据处理、标注服务、训练服务、仿真服务、智驾模型服务、公共配置管理组成。在使用各模块服务之前，需要先购买、开通相应服务。购买、开通服务会根据实际开通情况收取一定的费用，具体收费标准可以参考**计费项**。购买、开通服务成功后，可以在费用中心查询**费用账单**。

前提条件

用户拥有Octopus CommonOperations中的子权限“octopus:serviceOrdering:create”，默认策略中Octopus CommonOperations不允许购买，用户可以选择赋予子用户Octopus FullAccess权限，或者自定义权限策略来购买套餐包。

进入总览

进入总览界面，页面显示Octopus平台的介绍信息，服务购买入口、平台各子服务模块以及资源。

表 1-1 总览页面说明

编号	区域名称	说明
1	平台介绍以及服务购买入口	• 平台介绍：Octopus自动驾驶云服务的介绍。 • 服务购买入口：单击“购买服务”，即可跳转至服务订购界面，可参考**1.5.2 购买服务**。
2	服务模块	显示各服务模块，支持**续费**和**退订**。
3	我的资源	显示购买的扩展资源包，购买扩展资源包可参考**1.5.3 购买扩展资源包**，支持**续费**和**退订**。
4	按需服务	显示合规脱敏服务，支持**1.5.5 开通合规脱敏服务**。
5	模型微调	显示智驾模型微调服务，支持**1.5.6 开通智驾模型微调服务**。
6	我的模型	显示智驾模型服务中的2D预标注、2D图像生成、场景识别功能，支持**1.5.7 开通我的模型和购买套餐包**。

1.5.2 购买服务

使用各服务模块之前，需要先购买服务基础包。Octopus为用户提供了比较优惠的八爪鱼自动驾驶云服务-基础版，让用户可以快速体验Octopus云服务的大部分功能。同时，Octopus为用户提供了资源更丰富、功能更强大、能力更全面的各服务模块基础包。购买服务后，可以在“总览”页面“服务模块”区域**查看服务开通状态**。

表 1-2 支持的云服务基础版

名称	描述
八爪鱼自动驾驶云服务-基础版	支持数据、标注、训练、仿真4个基础服务功能，不含仿真引擎，需单独购买license。
自动驾驶仿真云服务-基础版	支持多种功能操作，用户可通过仿真服务完成在线仿真，仿真场景、创建仿真评测任务等。
自动驾驶数据云服务-基础版	具有路采数据解析能力，内置了50+的场景挖掘规则、具有自动驾驶数据回放功能，具有真实场景转仿真场景能力。
自动驾驶标注云服务-基础版	支持2D/3D目标标注、语义分割、连续帧标注、融合标注，具有满足自动驾驶所需的标注工具，同时支持预标注。
自动驾驶训练云服务-基础版	为自动驾驶研发提供方便易用的训练平台，让用户无需过多关注底层资源，专门聚焦核心算法开发。
自动驾驶合规云服务-基础版	提供数据硬盘递送、数据监管服务。

购买服务

步骤1 登录Octopus服务平台，在左侧菜单栏中单击“总览”。

步骤2 单击“购买服务”。

步骤3 选择需要购买的服务基础版（以购买“八爪鱼自动驾驶云服务-基础版”为例），配置购买时长、购买个数、是否自动续费和仿真器类型。

说明

- 购买了八爪鱼自动驾驶云服务-基础版后，不支持再购买其他云服务基础版。
- 购买了单独云服务模块基础版，不支持再购买八爪鱼自动驾驶云服务-基础版。
- 已购买的服务会在服务后打上“已购买”的标签，不支持到期前的重复购买。
- 合规服务依赖数据服务，在购买合规云服务-基础版前，需先购买数据云服务-基础版。

步骤4 单击“下一步”，购买存储扩容包、通用处理节点和AI处理节点，请详见**购买扩展资源包**。

步骤5 单击“下一步：确认配置”，确认配置后勾选服务声明。

步骤6 单击“去支付”，界面生成支付订单，可根据自身情况选择支付方式，确认付款。

----结束

查看服务开通状态

在“总览”页面“服务模块”查看服务开通状态。购买服务成功后，将在卡片区域左上角显示“生效中”，右上角显示服务到期时间。您还可以根据实际使用情况，对各服务模块进行续费管理，进行**续费和退订**操作。

1.5.3 购买扩展资源包

为了提升服务的计算能力、存储能力、业务拓展能力、行业竞争力等，Octopus还为用户提供了个性化的扩展资源包供用户选择。购买扩展资源包之前，可以先**查看我的资源配额**和**查看资源节点配额**。

表 1-3 云服务扩展资源包列表

名称	资源名称	描述
仿真服务扩容包	规控仿真引擎-在线	支持用户通过图形化界面方式运行仿真引擎，支持自动驾驶决策规划控制算法在线图形化开发调试。
	感知规控仿真引擎-在线	支持用户通过网页/图形化界面方式运行仿真引擎，支持自动驾驶感知决策规划控制算法在线图形化开发调试，支持高精度渲染引擎集成。
	仿真场景编辑器	支持自动驾驶仿真静态路网和动态交通参与物场景编辑。
	规控仿真引擎-并行	支持用户自动驾驶决策规划控制算法大规模测试验证，扩容包允许最大并行仿真数加100个。

名称	资源名称	描述
存储扩容包	自动驾驶数据管理缓存扩容包	当SFS存储剩余容量不足时，购买自动驾驶数据管理缓存扩容包增加SFS存储容量。
	自动驾驶数据管理存储扩容包	当OBS存储剩余容量不足时，购买自动驾驶数据管理缓存扩容包增加OBS存储容量。
通用处理节点	octopus计算型CPU(16u32g)专属实例	需要增加集群节点的数量时，可以购买通用处理节点增加节点实例，业务不会中断。购买成功后**查看资源节点配额**，给新购买的节点打上对应的用途标签即可调度使用。
	octopus计算型CPU(64u128g)专属实例	
AI处理节点	octopus计算型GPU(ant0324g24u96g)专属实例	需要增加集群节点的数量时，可以购买AI处理节点增加节点实例，业务不会中断。购买成功后**查看资源节点配额**，给新购买的节点打上对应的用途标签即可调度使用。 **须知** 该节点资源紧张，无法自助下单购买，需联系客户经理购买。
	octopus.hp.s2	
	octopus计算型NPU(x866sntp24g96u384g)专属实例	

购买扩展资源包

说明

购买扩容包和节点时，需要事前购买相对应的服务。

步骤1 登录Octopus服务平台。

步骤2 在“总览”页签中单击“购买服务”。

步骤3 选择需要购买的扩展资源包的基础配置，然后单击“下一步：资源配置”。

以购买“规控仿真引擎-在线”为例，设置购买时长、购买个数以及是否自动续费和仿真器类型。

步骤4 选择扩展资源包的资源配置，然后单击“下一步：确认配置”。

按需选择存储扩容包、通用处理节点和AI处理节点，以购买“自动驾驶数据管理缓存扩容包”和“octopus计算型CPU(16u32g)专属实例”为例。

步骤5 确认配置后，勾选服务声明，单击“去支付”，界面生成支付订单，可根据自身情况选择支付方式，确认付款。

----结束

查看我的资源配额

在“总览”页面“我的资源”区域查看资源配额情况。

- OBS存储：展示已购买的OBS存储资源。
- SFS存储：展示已购买的SFS存储资源。
- 资源列表：展示购买的所有扩展资源包、状态等信息。
- 续费管理：用户可以根据实际使用情况对资源进行**续费和退订**操作。

查看资源节点配额

左侧菜单栏中单击“运维配置”，在“集群纳管”列表单击cce-user-job，查看纳管详情，查看节点配额使用情况。

1.5.4 续费和退订

购买的服务和扩容包支持续费和退订操作。

续费

如需续费，请在服务和资源右侧的“续费管理”处进行续费操作。您还可以设置到期自动续费（请确认服务是否支持设置到期自动续费）。续费相关操作请参考**续费管理**。

退订

如需退订，请在服务和资源右侧的“续费管理”处进行退订操作。退订相关操作请参考**退订管理**。退订后，系统中的数据将被永久清理，无法恢复，请谨慎操作。

1.5.5 开通合规脱敏服务

在使用数据合规脱敏能力之前，需先开通合规脱敏服务。合规脱敏支持人脸，车牌，点云，GNSS，高程脱敏能力。开通合规脱敏服务后，会根据处理脱敏数据量按需计费，您可以在费用中心查询**费用账单**。

开通服务

步骤1 登录Octopus服务平台，在左侧菜单栏中单击“总览”。

步骤2 单击按需服务区域的合规脱敏产品操作列的“开通服务”。

步骤3 单击“确定”，确认开通合规脱敏服务。

步骤4 开通服务成功后，开通状态列将显示“已开通”。

----结束

1.5.6 开通智驾模型微调服务

在使用智驾模型服务模型微调功能之前，需先开通智驾模型微调服务。开通后，当模型微调状态为“运行中”，会根据创建模型微调任务时选择的资源按需计费。

开通服务

步骤1 登录Octopus服务平台，在左侧菜单栏中单击“总览”。

步骤2 在“模型微调”模块，单击操作栏中的“开通服务”。

步骤3 单击“确定”，确认开通智驾模型微调服务。

步骤4 开通服务成功后，开通状态列将显示“已开通”。

----结束

1.5.7 开通我的模型和购买套餐包

在使用智驾模型服务场景识别、2D图像生成、2D预标注、3D预标注等功能之前，需先开通我的模型。开通后，我的模型是根据API调用次数收取费用，推荐您购买模型套餐包，价格比按需计费模式更优惠。开通服务和购买套餐包之后，您可以在“我的模型”区域**查看开通状态和套餐包使用情况**。

开通我的模型

步骤1 登录Octopus服务平台，在左侧菜单栏中单击“总览”。

步骤2 在“我的模型”模块，单击操作栏中的“开通服务”。

步骤3 选择需要开通服务的用户名，单击操作栏中的“开通服务”。

步骤4 也可选择多个用户名，单击用户名上方的“批量开通”，批量开启服务。

步骤5 取消服务。

单击开通服务的账号后的操作栏中“取消服务”，即可关闭服务。

也可选择已开通服务的多个用户名，单击用户名上方的“批量取消”，批量关闭服务。

----结束

购买套餐包

步骤1 登录Octopus服务平台，在左侧菜单栏中单击“总览”。

步骤2 在“我的模型”模块，单击操作栏中的“购买套餐包”。

步骤3 根据需要选择模型套餐包配置。确认无误后，单击“去支付”，界面生成支付订单，可根据自身情况选择支付方式，确认付款。

----结束

查看开通状态和套餐包使用情况

开通服务后，“开通状态”列表将显示“已开通”。购买套餐包后，在列表中查看当前套餐包使用情况。

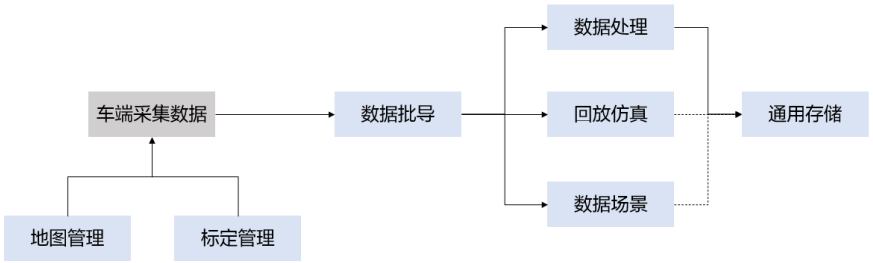
2 数据资产

2.1 数据资产简介

在自动驾驶产品的开发过程中，海量的数据存储和管理是当前自动驾驶平台面临的业务挑战之一。

Octopus平台的数据服务模块提供了海量数据采集、存储以及数据并行处理等功能，供后续服务进行统一使用。数据服务开发流程如下：

图 2-1 数据服务开发流程



- **地图管理**：支持上传高精地图数据，可用于数据回放、仿真场景等功能。
- **标定管理**：支持车辆、车辆传感器标定的配置，方便即时管理车辆以及标定信息。
- **源数据包**：此模块展示导入成功的数据包，支持对源数据包的查看，回放等功能。
- **数据场景**：平台处理完原始采集数据后，平台支持内置和自定义场景挖掘算法，可自动提取对应场景行为的片段，展示在此模块中。
- **数据集**：支持数据集多版本管理和统计。同时支持用户将本地符合平台规范的数据集导入平台，以及将平台上的数据集导出到自有OBS桶中。
- **数据缓存**：提供专用高速文件存储功能，加速训练和评测读取数据集的速度。
- **模型管理**：负责对模型仓库和模型版本进行各种操作，模型仓库可包含多个模型版本，支持上传符合平台规范的模型用于标注或训练任务。
- **通用存储**：支持创建通用存储，用于数据存储。

2.2 数据总览

在数据总览页面可查看平台数据服务的统计数据，帮助用户快速了解各项数据关键指标和业务的健康情况。

进入总览数据页面：

登录Octopus服务平台，在左侧菜单栏中单击“数据资产 > 数据总览”。

图 2-2 数据总览

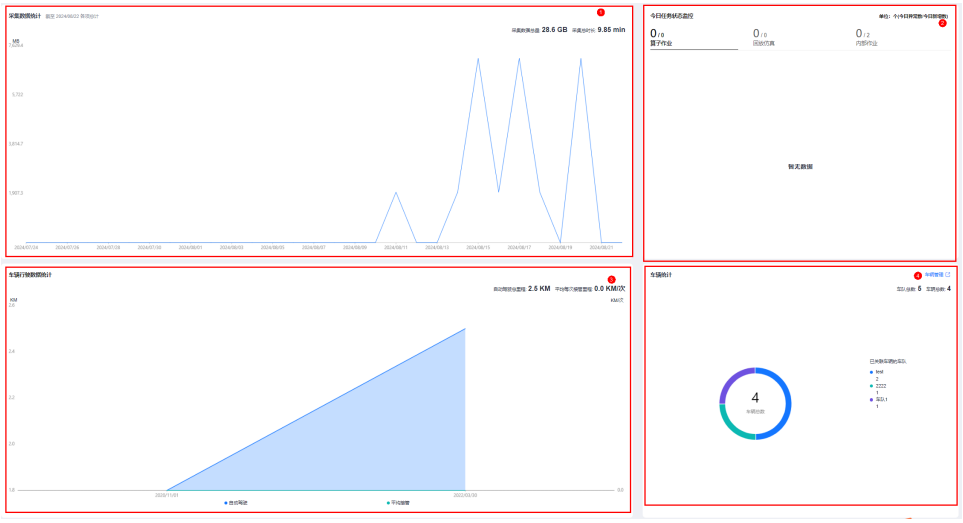


表 2-1 总览页面说明

序号	区域名称	说明
1	采集数据统计	平台数据量的统计信息，并图形化展现历史数据趋势。
2	今日任务状态监控	统计并显示当日算子作业，回放仿真，内部作业的总量及异常数量，以列表形式展示异常任务名称。
3	车辆行驶数据统计	车辆行驶里程统计信息，并图形化展示历史数据趋势： ● 自动驾驶总里程。 ● 平均每次接管里程数（KM/次）。
4	车辆统计	车队和车辆信息统计信息： ● 车队总数、车队名下车辆总数。 ● 车队名称，以及该车队关联的车辆数量。

2.3 地图管理

在地图管理中上传的高精地图数据，可用于数据回放、仿真场景等。平台支持OpenDRIVE格式的地图文件。

上传地图数据包

步骤1 在左侧菜单栏中，选择“数据资产 > 地图管理”。

步骤2 单击“上传”。

- 添加文件：选择本地“.xodr”格式文件，支持上传不超过100MB的文件。
- 名称：默认显示上传的地图文件的名称，可以重新编辑。
- 描述：简要描述地图信息。

步骤3 单击“上传”。

在“地图管理”页面的列表中，可以查看新上传的地图包。

----结束

地图管理相关操作

表 2-2 地图管理相关操作

任务	操作步骤
查看地图包详情	在“地图管理”页面，单击操作栏内的“详情”，用户可查看地图包详情。具体页面参考**地图包详情**。
删除地图包	在“地图管理”页面，单击操作栏内的“删除”，即可删除地图包。
查询地图包	在“地图管理”页面的搜索框输入搜索条件，按回车键即可查询地图包。
更新地图包	在“地图包详情”页面，单击文件后“上传更新”，可对地图文件以及OSGB文件（OSGB：OpenSceneGraphBinary，OSGB文件格式属于3D图文件格式）进行修改。详情请参考**地图包详情**。

地图包详情

在上传地图包后，在地图包列表页，单击操作栏内的“详情”，用户可查看地图包详情。

图 2-3 地图包详情

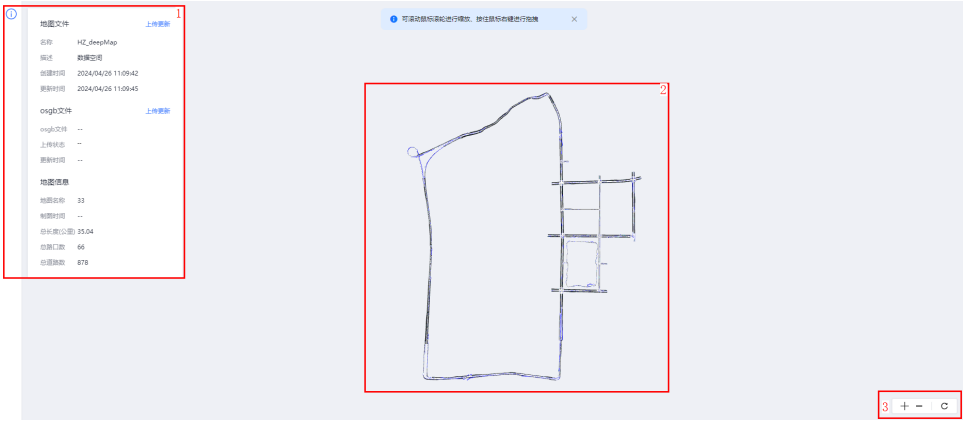


表 2-3 地图包详情说明

序号	区域	说明
1		控制地图包信息显示或者隐藏。 地图包信息包含：地图文件信息、osgb文件信息、地图信息。单击文件后“上传更新”，可对地图文件以及osgb文件进行修改。
2	地图显示区域	• 单击鼠标左键可旋转查看地图轨迹。 • 单击鼠标右键可平移地图轨迹。 • 鼠标滚轮可控制轨迹的大小缩放。 • 鼠标悬停在轨迹任意位置，可查看当前位置的坐标、车道等信息。
3	地图控制区域	单击右下方（放大、缩小、重置）任意选项，可以对地图进行放大、缩小、重置操作。

2.4 标定管理

2.4.1 车辆管理

平台支持配置车队、车辆，以及两者的所属关系。

步骤1 在左侧菜单栏中，选择“数据资产 > 标定管理”。

步骤2 进入“车辆管理”页签创建车队和车辆信息。

步骤3 创建车队。

1. 单击“车队管理”进入车队管理页面。
2. 单击“新建车队”，填写车队名称和描述信息，单击“确认”。
3. 车队创建完成后可以管理车队信息。
   - 查询车队：在搜索输入框中输入搜索条件，按回车键即可查询车队。
   - 编辑车队：单击操作栏内的“编辑”，即可编辑车队信息。
   - 删除车队：单击操作栏内的“删除”，即可删除车队。如需删除车队信息，需先删除该车队关联的车辆。

步骤4 创建车辆。

1. 单击“新建车辆信息”进入新建车辆信息页面。
2. 填写车辆名称和描述信息。
3. 选择车队名称（该车辆所属车队，可新建或直接选择已有车队），单击“创建”。
4. 车辆创建完成后可以管理车辆信息。
   - 查看车辆详情：单击具体的车辆名称，即可查看车辆详情。
   - 查询车辆：在搜索输入框中输入搜索条件，按回车键即可查询车辆。
   - 编辑车辆：单击操作栏内的“编辑”，即可编辑车辆信息。

- 删除车辆：单击操作栏内的“删除”，即可删除车辆。该车辆信息被历史数据批次引用，则无法删除。

----结束

2.4.2 传感器标定

标定数据记录车辆本身以及车辆上传感器的配置信息，一个标定项对应一个传感器标定文件。Octopus标定文件需满足标准，平台会对上传的每个标定文件名、格式和类型做检查。

步骤1 本地编写标定文件。

当前支持车架配置以及传感器标定信息配置：车架配置、相机以及激光雷达。

以创建激光雷达类型传感器标定文件为例详细说明，创建一个名为“pandar_40.yaml”的标定文件，文件格式可参考如下：

```
Pandar40:
  rotation_vel2veh_matrix:
    rows: 3
    cols: 3
    data:
[0.0420798,-0.99903437,0.01262295,0.99872498,0.04241283,0.02739081,-0.02789977,0.01145419,0.9995447
5]
  tanslation_vel2veh_vector:
    rows: 3
    cols: 1
    data: [1.34458068,0.02622486,1.40430678]
  rotation_veh_vel_euler:
    yaw: 0.1
    pitch: 0.1
    roll: 0.1
  translation_veh_vel:
    x: 0.1
    y: 0.1
    z: 0.1
  timeshift_vel_veh: 0.000000000
```

文件开头字段“pandar40”：为传感器命名，长度不能超出64位字符且不能包含特殊字符，命名格式可参考下表，并请结合实际情况自行定义。

表 2-4 标定文件命名参考

标定类型	命名
车架配置	车架配置标定
相机	roof_cam_0 roof_cam_1 roof_cam_2 roof_cam_3 roof_cam_4 ...
激光雷达	lidar

传感器的标定文件编写格式和配置项类似，可参考上述规范。车辆的标定文件配置项与传感器存在差异，参照示例模板，按照实际情况填写即可。

各类型标定文件模板可参考**标定文件模板**部分。

步骤2 在左侧菜单栏中选择“数据资产 > 标定管理”。

步骤3 选择“传感器标定”页签，单击“上传标定项”。

- 格式：当前支持“Octopus”。
- 上传标定文件：根据需要上传车架配置标定，相机标定，激光雷达标定。

  根据提示，将车辆和传感器信息标定文件，添加到对应标定类型。一次可以添加多个，标定文件不得超过平台限制的最大数量。每上传一个标定文件，对应一个标定项。

  说明

  请确保上传标定项中标定名称不重名，且文件内容正确，长度不能超出64位字符且不能包含特殊字符，否则将上传失败。

步骤4 创建标定项。

单击“完成”，Octopus平台将创建和标定文件一一对应的标定项。

----结束

传感器标定相关操作

在“传感器标定”页签，还可以完成以下任务。

表 2-5 传感器标定相关操作

任务	操作步骤
查看标定项详情	单击具体的标定项ID，即可查看标定项详情。
删除标定项	单击操作栏内的“删除”，即可删除标定项。
查询标定项	在搜索输入框中输入搜索条件，按回车键即可查询。

2.4.3 标定文件模板

Vehicle 车辆标定文件模板

标定文件名：“车辆自身参数.yaml”

文件内容示例：

```
# The vehicle config
vehicle:
    # basic
    mass:   #质量
    # Body
    front_track:   #m前轮距
    rear_track:   #m后轮距
    wheelbase:   #m轴距
    height:   #m高度
```

```
    width:  #m宽度
    length:  #m长度
    rear_distance:  #m后轴到车尾距离
    front_distance:  #m后轴到车头距离
    x_CoG:  #m前轴到重心距离，x方向
    y_CoG:  #m中心到重心距离，y方向
    z_CoG:  #m地面到重心距离，z方向
    #  Wheel  and  brake
    wheel_radius:  #m车轮半径
    #  Steering
    min_steering_angle:  #rad车轮最小转向角度，弧度
    max_steering_angle:  #rad车轮最大转向角度，弧度
    min_steering_wheel_angle:  #rad方向盘最小转向角度，弧度
    max_steering_wheel_angle:  #rad方向盘最大转向角度，弧度
    max_steering_wheel_speed:  #rad/s方向盘最大转速
    steering_zero_bias:  #rad初始位置偏移量
    #  Powertrain
    max_speed:  #m/s最大速度
    max_reversing_speed:  #m/s最大倒车速度
    max_deceleration:  #最大减速度
    max_acceleration:  #最大加速度
    rpm_idle:  #r/min引擎空闲状态转速
    rpm_change:  #r/min换挡时引擎转速
    rpm_max:  #  r/min引擎最大转速
    engine_power_max:  #watt引擎最大功率
    #  Transmission
    main_ratio:  #主减速器比率
    gear_ratio_drive:  #驾驶模式传动比率
    gear_ratio_reverse:  #倒车模式传动比率
    #  Coefficient
    lambda:  #等效质量乘数，需大于1
    Cd:  #空气阻力系数
    #  default  lane  params
    lane_width_default:  #车道宽度
    lane_width_margin:  #车道边线宽度
    safe_avoid_distance:  #m安全边线距离
#localizaition
loc:
    yaw_offset:
```

Camera 相机标定文件模板

标定文件名：“roof_cam_999.yaml”

文件内容示例：

```
Roof_Cam_9:
  camera_model: pinhole
  intrinsics:
    rows: 1
    cols: 4
    data:
    - 957.3874
    - 960.8491
    - 957.7912
    - 565.3051
  resolution:
    width: 1020
    height: 1920
  distortion_model: radtan
  distortion_coeffs:
    rows: 1
    cols: 5
    data:
    - -0.3152
    - 0.1068
    - 1.0109e-05
    - 0.00014344
    - -0.0174
```



```
rotation_veh2cam_matrix:
  rows: 3
  cols: 3
  data:
  - -0.9998299042764152
  - 0.017818974176046335
  - 0.004792519744625967
  - -0.0052751312364799794
  - -0.027136554579648777
  - -0.999617350015897
  - -0.01768206109206492
  - -0.9994728825361424
  - 0.027225960978949772
tanslation_veh2cam_matrix:
  rows: 3
  cols: 1
  data:
  - -0.03155699382111174
  - 1.1564462683421137
  - 0.5492014802301336
rotation_veh_cam_euler:
  yaw: 178.97898507859358
  pitch: -0.27459237862937325
  roll: -88.43985827166519
translation_veh_cam:
  x: -0.015740166503007778
  y: 0.580856341427072
  z: 1.1412035414910402
transform_image2veh_matrix:
  rows: 3
  cols: 3
  data: []
transform_veh2image_matrix:
  rows: 3
  cols: 4
  data:
  - -973.6528713841998
  - -939.7369315451515
  - 30.649111295873002
  - 495.5498232950973
  - -15.049594426539091
  - -590.5017166128401
  - -944.1638507975063
  - 1420.2429138739958
  - -0.01768206109206492
  - -0.9994728825361424
  - 0.027225960978949772
  - 0.5492014802301336
timeshift_cam_veh: 0.0
```

Lidar 激光雷达标定文件模板

标定文件名：“lidar_right_999.yaml”

文件内容示例：

```
Pandar40:
  rotation_vel2veh_matrix:
    rows: 3
    cols: 3
    data: [0.999365,0.0349167,0.00707821,-0.034795,0.999256,-0.0166374,-0.00765387,0.0163806,0.999837]
  tanslation_vel2veh_vector:
    rows: 3
    cols: 1
    data: [0.00915069,1.34196,1.38]
  rotation_veh_vel_euler:
    yaw: -1.9940678960476272
    pitch: 0.43853880056948835
    roll: 0.9386082809708912
```



```
translation_veh_vel:
  x: 0.00915069
  y: 1.34196
  z: 1.38
timeshift_vel_veh: 0.000000000
```

2.5 源数据包

2.5.1 数据包格式

2.5.1.1 上传数据格式

在使用Octopus平台收集数据前，请仔细阅读本章节，确保上传数据格式符合平台要求，有助于用户更快速地完成数据收集以及数据格式转换。

上传数据格式：单包上传大小小于200GB。

转换后数据格式：OpenData格式。

使用场景

Octopus平台接收数据包（数据包至少需在一层文件夹内），没有转换OpenData格式时，可用于算子作业输入。

命名规范

用户可将多个数据包存放在同个一级目录下，作为同一批次采集数据上传至Octopus平台。

一级目录的结构可根据业务情况自定义数据包名称，如果数据包为Rosbag格式，示例参考如下：

```
.
├─ octopus
   ├─ Rosbag
      ├─{导入id}
         ├─{数据包名}
            ├─ {数据包名}.bag
            ├─ {数据包名}.yaml     #非必填
            ├─ xxxxx.mp4           #非必填，大小不限制，命名大小不可超过128位，不可有特殊字符。
```

与数据包同名的 yaml 配置文件说明

数据包中必须含有与数据包同名的yaml配置文件主要包括车辆名称、传感器信息和标定ID等信息，详情参考如下：

```
# 华为八爪鱼自动驾驶云服务数据采集说明
project: '项目名称'
module: '感知'
cardrive:
   collect_time: 2020-11-01T08:00:00+08:00     #数据包采集日期，精确到小时即可
   station: '腾飞'                          #选填 数据采集地点名称，站点名称
   car:
      vehicle_name: 'test'              #车辆名称，仅支持在八爪鱼平台创建的车辆
   route: 'shuttlebus_30km'                  #选填 车辆行驶路线
   speed：10km/h                          #选填 车速
   mode: 'auto'                           #选填 路线驾驶意图, auto代表自动驾驶, manual代表人工驾驶采集
   tags: ['主车直行','主车倒车']             #选填 标签，标签个数不超过50个 例：沙尘天，正向设计，驾驶模式
   description: '强风沙天,车辆空载在排土区自动驾驶到接土区前等待长坡道'   #选填 车载情况
```



```
  segments:                          #选填 数据包场景片段
    -
      tags: ['晴天','直行']
      time: 2021-08-27T11:43:07~2021-08-27T11:43:47
data_type: Rosbag                    #必填 数据类型
map_id: MAP1134                      #选填，高精地图ID，字符串类型，配备后才可在回放数据界面展示高精地图信息。
preprocessor:                        #转OpenData算子信息
  id: 10105                          # 算子id
  resource_spec: X86_4Core_8GiB      # 资源规格
```

2.5.1.2 转换后数据格式

Octopus平台支持将上传的Rosbag格式转换为OpenData格式。

数据类型

Octopus平台对数据有以下要求：

- 数据类型：包括各传感器数据、车辆数据、目标推理数据、自车坐标姿态以及标签记录数据等。
- 数据格式：Octopus OpenData格式。其中相机采集数据文件后缀为“.jpg”，激光雷达采集数据文件后缀为“.pcd”，其他采集数据文件后缀为“.pb”（谷歌定义的protobuf格式文件）。

  详情请参考**表2-6**。
- 消息topic具体格式要求请参考“**2.5.1.3 消息topic格式规范**”。
- 接收到的消息topic示例请参考“**2.5.1.4 消息topic格式示例**”。

说明

- 自车相关或每个传感器设备，都对应一个消息topic。
- 采集数据的topic名称支持自定义，包含中英文、数字、“_”“-”，不得超过64个字符。

表 2-6 数据类型和消息 topic 对应关系

分类	数据类型	消息topic（示例）	文件后缀	备注
传感器	相机（**camera**）	camera_front	.jpg	录制车辆路况图像数据。
	激光雷达（**lidar**）	lidar_roof_0	.pcd	以发射激光束探测目标的位置、速度等特征量的雷达系统，探测车辆周围的目标位置，监测移动速度。
	位置数据（**gnss**）	gnss_raw	.pb	通过卫星导航系统，定位车辆位置。
	毫米波雷达（**radar**）	RADAR_FRONT	.pcd	工作在毫米波段探测的雷达，探测车辆周围的目标位置，监测移动速度。
车辆数据	自车坐标和姿态数据（**ego_tf**）	ego_tf	.pb	定位自车所处位置以及当前车辆姿态。

分类	数据类型	消息topic（示例）	文件后缀	备注
	车辆数据（**vehicle**）	vehicle	.pb	车辆底盘信息。
规划推理数据	目标推理数据（**object_array_vision**）	object_array_vision	.pb	感知数据信息。
标签数据	标签记录数据（**tag_record**）	tag_record	.pb	在车端标记驾驶过程中人工和自动驾驶路段以及其他重要信息。
控制数据	控制指令（**control**）	control	.pb	自车的方向盘转角、加速度值等控制数据。
规划路径	规划轨迹（**planning_trajectory**）	planning_trajectory	.pb	自车规划行驶路径。
预测路径	预测跟踪（**predicted_objects**）	predicted_objects	.pb	感知目标的预测路径。
全局规划	全局路径（**routing_path**）	routing_path	.pb	自车全局规划路径。
交通灯	交通灯信息（**traffic_light_info**）	traffic_light_info	.pb	红绿灯。

使用场景

Octopus平台接收到原始数据后，将对数据进行解包、轨迹和接管分析等操作，用于数据总览、数据场景、数据回放、标注服务等模块，请用户结合实际需求，准备好相应模块所需数据。

Octopus平台转换后的OpenData数据服务模块所需数据请见下表：

表 2-7 数据和模块对应关系

类型	消息	数据总览	数据场景	数据回放	标注服务
相机	camera	-	-	√	√
激光雷达	lidar	-	-	√	√
位置数据	gnss	√	-	√	-
自车坐标姿态	ego_tf	-	√	√	-
车辆数据	vehicle	-	√	√	-

类型	消息	数据总览	数据场景	数据回放	标注服务
感知推理	object_array_vision	-	√	√	-
接管及打标签信息	tag_record	-	-	√	-
控制指令	control	-	-	√	-
规划轨迹	planning_trajectory	-	-	√	-
预测跟踪	predicted_objects	-	-	√	-
全局规划	routing_path	-	-	√	-
交通灯	traffic_light_info	-	-	√	-
毫米波雷达	radar	-	-	√	-

采集数据命名规范

用户可将多个数据包存放在同个一级目录下，作为同一批次采集数据上传至Octopus平台。

一级目录的结构可根据业务情况自定义数据包名称，示例参考如下：

```
.
├─ raw-data
├─ package01
├─ OpenData
├─ 2020-04-28-08-00
└─ ...
```

Octopus OpenData 格式数据说明

Octopus OpenData格式数据目录结构可根据实际采集节点种类及数量进行修改，示例参考如下：

```
.
├─ OpenData
    ├─ camera_0
        ├─xxx.jpg
        ├─xxx.jpg
        ├─timestamp.jpg
        ├─...
    ├─ lidar_roof_0
        ├─xxx.pcd
        ├─xxx.pcd
        ├─timestamp.pcd
        ├─...
    ├─ gnss
        ├─episode-1.pb
    ├─other sensor
        ├─episode-1.pb
    ...

    ├─ octopus_data_collection.yaml
```

说明

1. Octopus平台对Octopus OpenData数据包内文件大小限制如下：
   - “.yaml”文件小于10kb。
   - “.jpg”文件小于2MB。
   - “.pcd”文件小于10MB。
   - “.pb”文件小于50MB。
2. MP4命名大小不可超过128位，不可有特殊字符。MP4的时间需要和Rosbag包中的时间匹配。
3. 数据名称仅包含中文、大小写英文、数字、“-”“_”，不超过64位。
4. 数据包名称仅包含中文、大小写英文、数字、“-”“_”，不得出现其他字符，且长度不超过64个字符。

“octopus_data_collection.yaml”配置文件说明

数据包中有“Octopus_data_collection.yaml”配置文件，各类型传感器的名字必须和文件夹名称一致，格式也必须与规范相匹配。

配置文件，主要包括车辆名称、传感器信息和标定ID等信息，详情参考如下：

```
# 华为八爪鱼自动驾驶云服务数据采集说明
cardrive:
    collect_time: 2020-11-01T08:00:00+08:00     #数据包采集日期，精确到小时即可
    station: '腾飞'                          #选填 数据采集地点名称，站点名称
    car:
        vehicle_name: 'test0805'            #车辆名称，仅支持在八爪鱼平台创建的车辆
    route: 'shuttlebus_30km'                #选填 车辆行驶路线
    mode: 'auto'                           #选填 路线驾驶意图, auto代表自动驾驶, manual代表人工驾驶采集
    tags:                                 #选填 数据包对应标签ID
    description: ''                        #选填 数据包描述
data_type: opendata                          #必填 数据包类型，转换后的OpenData数据中包含
ocotopus_data_collection.yaml文件
map_id:''                                  #选填，高精地图ID，字符串类型，配备后才可在回放数据界面展示高精地图信息。
```

2.5.1.3 消息 topic 格式规范

Vehicle

对于车辆自身基本数据录制的消息格式，需遵循一定规范，其中部分字段为必选，其他请根据实际需要自由选取。

表 2-8 vehicle 消息格式规范

格式名称	说明
VehicleInfo	车辆信息

须知

消息格式中部分参数为必选，如使用该数据类型，则不可缺少该参数字段，否则会导致数据上传Octopus平台失败。

```
/**********************************
    content: Octopus 输入数据格式
    version: 0.1
**********************************/
syntax = "proto3";

package Octopusdata;

message VehicleFrame {
    uint64 stamp_secs = 1;                #必选。时间戳，单位：秒
    uint64 stamp_nsecs = 2;               #必选。时间戳，单位：纳秒
    uint32 autonomy_status = 3;             #非必选。自动驾驶状态
    sint32 gear_value = 4;                #必选。只应从枚举常量中赋值
    float vehicle_speed = 5;              #必选。行驶速度，如果齿轮是倒挡，值为负。
    float steering_angle = 6;             #必选。转向，以角度表示。顺时针或向右为正，0为垂直或直角。
    float yaw_rate = 7;                  #Unit: deg/s
    float interior_temperature = 8;         #Unit: Celsius
    float outside_temperature = 9;          #Unit: Celsius
    float brake = 10;                    #必选。刹车制动按压百分比（0代表不按，1代表完全按下）。
        uint64 timestamp = 11;                #必选。时间戳。
    int32 turn_left_light=12;             #必选。左转灯。
    int32 turn_right_light=13;             #必选。右转灯。
    float longitude_acc=14;               #必选。纵向加速度。
    float lateral_acc=15;                #必选。横向加速度。

}

message VehicleInfo {
    repeated VehicleFrame vehicle_info = 1;
}
```

Camera

采集的camera数据通过转换工具可以保存为“.jpg”图片数据。

Lidar

采集的点云数据通过转换工具可以保存为标准的pcd格式数据。

Gnss

对于卫星导航系统数据录制的消息格式，需遵循一定规范，其中部分字段为必选，其他请根据实际需要自由选取。

表 2-9 gnss 消息格式规范

格式名称	说明
GnssPoints	gps点

须知

消息格式中部分字段为必选，如使用该数据类型，则不可缺少该参数字段，否则会导致数据上传Octopus平台失败。

```
/**********************************
    content: Octopus 输入数据格式
    version: 1.0
```

```
**********************************/
syntax = "proto3";

package Octopusdata;

message GnssPoint {
    uint64 stamp_secs = 1;     #必选。时间戳，单位：秒
    uint64 stamp_nsecs = 2;    #必选。时间戳，单位：纳秒
        float latitude = 3;        #必选。纬度
        float longitude = 4;       #必选。经度
        float elevation = 5;       #必选。海拔高度，单位：米
        uint64 timestamp = 6;      #必选。时间戳
}

message GnssPoints {
    repeated GnssPoint gnss_points = 1;
}
```

Radar

雷达采集的点云数据通过转换工具可以保存为标准的pcd格式数据。

Ego_tf

对于自车角度位置数据录制的消息格式，需遵循一定规范，其中部分字段为必选，其他请根据实际需要自由选取。

表 2-10 ego_tf 消息格式规范

格式名称	说明
LocalizationInfo	主车信息

须知

消息格式中部分字段为必选，如使用该数据类型，则不可缺少该参数字段，否则会导致数据上传Octopus平台失败。

```
/**********************************
    content: Octopus 输入数据格式
    version: 1.0
**********************************/
syntax = "proto3";

package Octopusdata;


message LocalizationInfoFrame {
    uint64 timestamp = 1;                #必选。时间戳。
    uint64 stamp_secs = 2;               #必选。时间戳，单位：秒
    uint64 stamp_nsecs = 3;              #必选。时间戳，单位：纳秒
    float pose_position_x = 4;           #必选。自车x轴坐标
        float pose_position_y = 5;           #必选。自车y轴坐标
        float pose_position_z = 6;           #必选。自车z轴坐标
    float pose_orientation_x = 7;        #必选。自车四元数x值
        float pose_orientation_y = 8;        #必选。自车四元数y值
        float pose_orientation_z = 9;        #必选。自车四元数z值
    float pose_orientation_w = 10;       #必选。自车四元数w值
    float pose_orientation_yaw=11;       #必选。朝向角，单位：rad
    float velocity_linear=12;            #必选。速度，单位：m/s
```

```
    float velocity_angular=13;              #必选。角速度，单位：rad/s
    float acceleration_linear=14;           #必选。加速度，单位：m^2/s
    float acceleration_angular=15;          #必选。角加速度，单位：rad^2/s
}

message LocalizationInfo {
    repeated LocalizationInfoFrame localization_info = 1;
}
```

Object_array_vision

对于目标推理数据录制的消息格式，需遵循一定规范，其中部分字段为必选，其他请根据实际需要自由选取。

表 2-11 object_array_vision 消息格式规范

格式名称	说明
TrackedObject	感知目标

须知

消息格式中部分字段为必选，如使用该数据类型，则不可缺少该参数字段，否则会导致数据上传Octopus平台失败。

```
/**********************************
    content: Octopus 输入数据格式
    version: 1.0
**********************************/
syntax = "proto3";

package Octopusdata;


message Object {
    uint64 id = 1;                      #必选。目标推理数据object数组id
    string label = 2;                   #必选。标记物体类型
    float pose_position_x = 3;          #必选。目标物x轴坐标
        float pose_position_y = 4;          #必选。目标物y轴坐标
        float pose_position_z = 5;          #必选。目标物z轴坐标
    float pose_orientation_x = 6;       #必选。目标物四元数x值
        float pose_orientation_y = 7;       #必选。目标物四元数y值
        float pose_orientation_z = 8;       #必选。目标物四元数z值
    float pose_orientation_w = 9;       #必选。目标物四元数w值
    float pose_orientation_yaw = 10;    #必选。朝向角，单位：rad
    float dimensions_x = 11;            #必选。目标物x方向尺寸（长）
    float dimensions_y = 12;            #必选。目标物y方向尺寸（宽）
    float dimensions_z = 13;            #必选。目标物z方向尺寸（高）
    float speed_vector_linear_x = 14;   #必选。目标物x方向速度
        float speed_vector_linear_y = 15;   #必选。目标物y方向速度
        float speed_vector_linear_z = 16;   #必选。目标物z方向速度
        float relative_position_x = 17;     #必选。目标物相对于主车x方向位置
        float relative_position_y = 18;     #必选。目标物相对于主车y方向位置
        float relative_position_z = 19;     #必选。目标物相对于主车z方向位置
}
message TrackedObjectFrame {
    uint64 timestamp = 1;               #必选。时间戳
    uint64 stamp_secs = 2;              #必选。时间戳，单位：秒
    uint64 stamp_nsecs = 3;             #必选。时间戳，单位：纳秒
    repeated Object objects = 4;        #必选。object数组
}
```

```
message TrackedObject {
    repeated TrackedObjectFrame tracked_object = 1;
}
```

Tag_record

对于标签记录数据录制的消息格式，需遵循一定规范，其中部分字段为必选，其他请根据实际需要自由选取。

表 2-12 tag_record 消息格式规范

格式名称	说明
ScenarioSegments	场景片段

```
/**********************************
    content: Octopus 输入数据格式
    version: 1.0
**********************************/
syntax = "proto3";

package Octopusdata;

message ScenarioSegment {
 uint32 scenario_id = 1;                #必选。场景id
 string source = 2;                    #必选。片段的来源
 uint64 start = 3;                     #必选。片段的开始时间（时间戳）
 uint64 end = 4;                       #必选。片段的结束时间（时间戳）
}

message ScenarioSegments {
 repeated ScenarioSegment segments = 1;
}
```

Control

对于控制数据的消息格式，需遵循一定规范，其中部分字段为必选，其他请根据实际需要自由选取。

表 2-13 control 消息格式规范

格式名称	说明
ControlCommand	控制命令

```
/**********************************
    content: Octopus 输入数据格式
    version: 1.0
**********************************/
syntax = "proto3";
package Octopusdata;
message CommandFrame {
    uint64 stamp_secs = 1;
    uint64 stamp_nsecs = 2;
    uint64 timestamp = 3;                    #必选，时间戳
    float acceleration=4;                   #必选，加速度值
    float front_wheel_angle=5;                #必选，方向盘转角
    int32 gear=6;
```

```
}
message ControlCommand {
    repeated CommandFrame command_frame = 1;
}
```

Predicted_objects

对于预测路径数据的消息格式，需遵循一定规范，其中部分字段为必选，其他请根据实际需要自由选取。

表 2-14 predicted_objects 消息格式规范

格式名称	说明
PredictionObstacles	预测障碍物

```
/**************************************
   content: Octopus 输入数据格式
   version: 1.0
**************************************/
syntax = "proto3";
package Octopusdata;
message PathPoint {
   float x = 1;          #必选，预测轨迹点x坐标
   float y = 2;          #必选，预测轨迹点y坐标
   float z = 3;          #必选，预测轨迹点z坐标
   float theta = 4;
   float kappa = 5;
   int32 lane_id= 6;
   float v=7;
   float a=8;
   float relative_time=9;
}
message PredictionTrajectory {
  repeated PathPoint path_point = 1;  #必选，预测轨迹多个点
}
message Obstacle {
   uint64 obstacle_timestamp = 1;
   int32 id=2;               #必选，预测目标的id
   float x = 3;  #非必选，预测目标的x坐标
   float y = 4;  #非必选，预测目标的y坐标
   float z = 5;  #非必选，预测目标的z坐标
   repeated PredictionTrajectory prediction_trajectory = 6;  #必选，预测目标的多条轨迹
}
message PerceptionObstacle {
  uint64 stamp_secs = 1;
  uint64 stamp_nsecs = 2;
  uint64 timestamp = 3;  #必选，预测目标的时间戳
  repeated Obstacle obstacle_info= 4;  #必选，多个目标的预测信息
}
message PredictionObstacles {
  repeated PerceptionObstacle perception_obstacle= 4;  #必选，多条帧数据
}
```

Planning_trajectory

对于规划路径数据的消息格式，需遵循一定规范，其中部分字段为必选，其他请根据实际需要自由选取。

表 2-15 planning_trajectory 消息格式规范

格式名称	说明
PlanTrajectory	规划路径

```
/**************************************
    content: Octopus 输入数据格式
    version: 1.0
**************************************/
syntax = "proto3";
package Octopusdata;
message TrajectoryPoint {
    float x = 1;   #必选，轨迹点x坐标
    float y = 2;   #必选，轨迹点y坐标
    float z = 3;   #必选，轨迹点z坐标
    float theta = 4;
    float kappa = 5;
    int32 lane_id=6;
    float v=7;   #必选，速度
    float a=8;   #必选，加速度
    float relative_time=9;  #必选，相对时间
}
message Trajectory {
    uint64 stamp_secs = 1;
    uint64 stamp_nsecs = 2;
    uint64 timestamp = 3;    #必选，时间戳
    float total_path_length = 4;
    float total_path_time=5;
    int32 gear=6;      #非必选，档位
    int32 trajectory_type=7;
    int32 vehicle_signal=8;
    repeated TrajectoryPoint trajectory_points = 9;  #必选，轨迹
}
message PlanTrajectory {
     repeated Trajectory trajectory_info= 1;
}
```

Routing_path

对于全局规划路径数据的消息格式，需遵循一定规范，其中部分字段为必选，其他请根据实际需要自由选取。

表 2-16 routing_path 消息格式规范

格式名称	说明
RoutingFrames	规划路径

```
/**************************************
    content: Octopus 输入数据格式
    version: 1.0
**************************************/
syntax = "proto3";

message Point{
    float x = 1;
    float y = 2;
    float z = 3;
}
```

```
message Path{
    uint64 id = 1;
    repeated Point path_point = 2;
}

message RoutingPath{
    uint64 timestamp = 1;
    uint64 stamp_secs = 2;
    uint64 stamp_nsecs = 3;
    repeated Path routing_path_info = 4;
}

message RoutingFrames{
    repeated RoutingPath routing_frame = 4;
}
```

Traffic_light_info

对于交通灯数据的消息格式，需遵循一定规范，其中部分字段为必选，其他请根据实际需要自由选取。

表 2-17 traffic_light_info 消息格式规范

格式名称	说明
TrafficLightInfo	交通灯

```
/**********************************
    content: Octopus 输入数据格式
    version: 1.0
**********************************/
syntax = "proto3";

package Octopusdata;

message Light {
    uint64 id = 1;
    uint64 color = 2;
    uint64 state = 3;
    uint64 type = 4;
    float location_x = 5;
    float location_y = 6;
    float location_z = 7;
}

message Lights {
    uint64 timestamp = 1;
    uint64 stamp_secs = 2;
    uint64 stamp_nsecs = 3;
    repeated Light lights = 4;
}

message TrafficLightInfo {
    repeated Lights trafficlight_info = 1;
}
```

2.5.1.4 消息 topic 格式示例

消息topic具体格式要求请参考“**消息topic格式规范**”。接收到的消息topic示例请参考如下示例：

Vehicle

```
vehicle_info {
    stamp_secs: 1604996332
    stamp_nsecs: 847945211
    autonomy_status: 0
    gear_value: 4
    vehicle_speed: 43.93000030517578
    steering_angle: 0.699999988079071
    yaw_rate: 0.0
    interior_temperature: 0.0
    outside_temperature: 0.0
    brake: 0.0
    timestamp: 1604996332847
    turn_left_light: 0
    turn_right_light: 0
    longitude_acc: -0.03125
    lateral_acc: 0.0
}
```

Gnss

```
gnss_points {
    stamp_secs: 1604996332
    stamp_nsecs: 855145358
    latitude: 30.18027687072754
    longitude: 120.21341705322266
    elevation: 7.392796039581299
    timestamp: 1604996332855
}
```

Ego_tf

```
localization_info {
    timestamp: 1604996332855
    stamp_secs: 1604996332
    stamp_nsecs: 855301408
    pose_position_x: 1165.5460205078125
    pose_position_y: -479.2198486328125
    pose_position_z: -1.48505699634552
    pose_orientation_x: 0.003883248195052147
    pose_orientation_y: -0.0031167068518698215
    pose_orientation_z: 0.7017714977264404
    pose_orientation_w: 0.7123847603797913
    pose_orientation_yaw: 1.5557808876037598
    velocity_linear: 12.21684455871582
    velocity_angular: 0.014540454372763634
    acceleration_linear: 0.23571151494979858
    acceleration_angular:  0.0
}
```


Object_array_vision

```
tracked_object {
  timestamp: 1604996332862
  stamp_secs: 1604996332
  stamp_nsecs: 862911489
  objects {
    id: 26175
    label: "Car"
    pose_position_x: 1154.59912109375
    pose_position_y: -496.5350646972656
    pose_position_z: -1.8222997188568115
    pose_orientation_z: 0.714431643486023
    pose_orientation_w: 0.6997052431106567
    pose_orientation_yaw: 1.5916229486465454
    dimensions_x: 4.513162136077881
    dimensions_y: 1.7747581005096436
    dimensions_z: 1.628068208694458
    speed_vector_linear_x: 0.012852923013269901
    speed_vector_linear_y: -9.972732543945312
    relative_position_x: -17.48011016845703
    relative_position_y: 10.685434341430664
    relative_position_z: -0.17673441767692566
  }
  objects {
    id: 26170
    label: "Pedestrian"
    pose_position_x: 1180.902099609375
    pose_position_y: -504.7625732421875
    pose_position_z: -1.3601081371307373
    pose_orientation_z: -0.7057344317436218
    pose_orientation_w: 0.7084764242172241
    pose_orientation_yaw: -1.5669186115264893
    dimensions_x: 0.7922295331954956
    dimensions_y: 0.7891787886619568
    dimensions_z: 1.6868246793746948
    speed_vector_linear_x: 0.13573257625102997
    speed_vector_linear_y: 1.5281875133514404
    relative_position_x: -25.306795120239258
    relative_position_y: -15.737456321716309
    relative_position_z: 0.39350399374961853
  }
  objects {
    id: 26169
    label: "Pedestrian"
    pose_position_x: 1175.647216796875
    pose_position_y: -506.730712890625
    pose_position_z: -1.569373607635498
    pose_orientation_z: 0.6943609118461609
    pose_orientation_w: 0.7196269631385803
    pose_orientation_yaw: 1.5350627899169922
    dimensions_x: 0.8029457330703735
    dimensions_y: 0.7876891493797302
    dimensions_z: 1.6028095483779907
    speed_vector_linear_x: 0.06551000475883484
    speed_vector_linear_y: 0.0022428608499467373
    relative_position_x: -27.355571746826172
    relative_position_y: -10.512933731079102
    relative_position_z: 0.19844147562980652
  }
  objects {
    id: 26168
    label: "Pedestrian"
    pose_position_x: 1173.3189697265625
    pose_position_y: -507.2300109863281
    pose_position_z: -1.6026556491851807
    pose_orientation_z: 0.717462956905365
    pose_orientation_w: 0.6965966820716858
    pose_orientation_yaw: 1.600306749343872
    dimensions_x: 0.7922430038452148
```



```
  dimensions_y: 0.7811086177825928
  dimensions_z: 1.6341478824615479
  speed_vector_linear_x: -0.04817964881658554
  speed_vector_linear_y: -0.21502695977687836
  relative_position_x: -27.89008903503418
  relative_position_y: -8.192517280578613
  relative_position_z: 0.16775710880756378
}
objects {
  id: 26155
  label: "Bus"
  pose_position_x: 1172.106689453125
  pose_position_y: -478.5303039550781
  pose_position_z: -0.48812994360923767
  pose_orientation_z: -0.7203028798103333
  pose_orientation_w: 0.6936596632003784
  pose_orientation_yaw: -1.6084778308868408
  dimensions_x: 11.322981834411621
  dimensions_y: 2.9294095039367676
  dimensions_z: 3.1415622234344482
  speed_vector_linear_x: -0.017722932621836662
  speed_vector_linear_y: 0.1302066147327423
  relative_position_x: 0.7977913022041321
  relative_position_y: -6.548437118530273
  relative_position_z: 0.9966707229614258
}
objects {
  id: 26153
  label: "Bus"
  pose_position_x: 1148.1876220703125
  pose_position_y: -490.8350524902344
  pose_position_z: -0.954763650894165
  pose_orientation_z: 0.6907882690429688
  pose_orientation_w: 0.7230570912361145
  pose_orientation_yaw: 1.5251574516296387
  dimensions_x: 10.779899597167969
  dimensions_y: 2.856076717376709
  dimensions_z: 2.811084508895874
  speed_vector_linear_x: 0.03153659775853157
  speed_vector_linear_y: 0.23439916968345642
  relative_position_x: -11.868709564208984
  relative_position_y: 17.1827335357666
  relative_position_z: 0.6278138756752014
}
objects {
  id: 26141
  label: "Bus"
  pose_position_x: 1171.7779541015625
  pose_position_y: -512.5936889648438
  pose_position_z: -0.9443151354789734
  pose_orientation_z: -0.7186583876609802
  pose_orientation_w: 0.6953632831573486
  pose_orientation_yaw: -1.6037421226501465
  dimensions_x: 10.841312408447266
  dimensions_y: 2.9661808013916016
  dimensions_z: 3.2250704765319824
  speed_vector_linear_x: 0.0513402484357357
  speed_vector_linear_y: 0.006104861851781607
  relative_position_x: -33.26952362060547
  relative_position_y: -6.731308937072754
  relative_position_z: 0.8776476979255676
}
objects {
  id: 26133
  label: "Bus"
  pose_position_x: 1146.657958984375
  pose_position_y: -508.7508239746094
  pose_position_z: -0.883571445941925
  pose_orientation_z: 0.7007946968078613
```



```
    pose_orientation_w: 0.713362991809845
    pose_orientation_yaw: 1.5530219078063965
    dimensions_x: 12.186415672302246
    dimensions_y: 2.824420690536499
    dimensions_z: 3.292656183242798
    speed_vector_linear_x: 0.005901232361793518
    speed_vector_linear_y: 0.013970088213682175
    relative_position_x: -29.803848266601562
    relative_position_y: 18.443498611450195
    relative_position_z: 0.8749525547027588
  }
  objects {
    id: 26120
    label: "Bus"
    pose_position_x: 1170.993408203125
    pose_position_y: -525.5801391601562
    pose_position_z: -1.104852318763733
    pose_orientation_z: -0.7154129147529602
    pose_orientation_w: 0.6987019181251526
    pose_orientation_yaw: -1.5944297313690186
    dimensions_x: 10.749905586242676
    dimensions_y: 2.7170863151550293
    dimensions_z: 3.0421104431152344
    speed_vector_linear_x: 0.016746148467063904
    speed_vector_linear_y: -0.23609620332717896
    relative_position_x: -46.26727294921875
    relative_position_y: -6.141877174377441
    relative_position_z: 0.8449855446815491
  }
}
```

Tag_record

```
segments {
  scenario_id: 100000000
  source: "takeover"
  start: 1617336642300
  end: 1617336652300
}
segments {
  scenario_id: 100000000
  source: "vehicle"
  start: 1617336672300
  end: 1617336692300
}
```

Control

```
stamp_secs: 1617336640
stamp_nsecs: 795364700
timestamp: 1617336640795
acceleration: 0.7146586179733276
front_wheel_angle: 2.9919793605804443
```

Predicted_objects

```
stamp_secs: 1617336640
stamp_nsecs: 971891550
timestamp: 1617336640971
obstacle_info {
  obstacle_timestamp: 1617336640699
  id: 6711
  x: -123.08731842041016
  y: 486.83221435546875
  z: 0.575542688369751
  prediction_trajectory {
    path_point {
      x: -103.26817321777344
```



```
      y: 486.0815734863281
      theta: -0.007839304395020008
      v: 4.405668258666992
      relative_time: 4.5
    }
    path_point {
      x: -102.82765197753906
      y: 486.0737609863281
      theta: -0.00746726430952549
      v: 4.405668258666992
      relative_time: 4.599999904632568
    }
    ......
  }
}
obstacle_info {
  obstacle_timestamp: 1617336640699
  id: 6744
  x: -145.0320587158203
  y: 491.35015869140625
  z: -0.40381166338920593
  prediction_trajectory {
    path_point {
      x: -145.0320587158203
      y: 491.35015869140625
      theta: -2.9442124366760254
      v: 1.0038001537322998
    }
    path_point {
      x: -145.1304931640625
      y: 491.3304748535156
      theta: -2.9442124366760254
      v: 1.0038001537322998
      relative_time: 0.10000000149011612
    }
    ......
  }
}
obstacle_info {
  obstacle_timestamp: 1617336640699
  id: 6760
  x: -138.3047332763672
  y: 489.9286193847656
  z: -0.12651222944259644
}
```

Planning_trajectory

```
stamp_secs: 1617336640
stamp_nsecs: 809739351
timestamp: 1617336640809
trajectory_points {
  x: -151.27487182617188
  y: 486.55096435546875
  theta: 0.0023324606008827686
  kappa: -0.0017824547830969095
}
trajectory_points {
  x: -151.21182250976562
  y: 486.5510559082031
  theta: 0.0022713469807058573
  kappa: -0.0017127590253949165
}
......
```

Routing_path

```
timestamp: 1630057162125
stamp_secs: 1630057162
```



```
stamp_nsecs: 125769156
routing_path_info {
  id: 1
  path_point {
    x: -203.34230041503906
    y: 125.63516998291016
    z: -0.5
  }
  path_point {
    x: -203.34915161132812
    y: 125.72517395019531
    z: -0.5
  }.....}
```

Traffic_light_info

```
timestamp: 1630057508000
stamp_secs: 1630057508
lights {
  id: 1
  color: 1
  location_x: -206.60186767578125
  location_y: 459.9820861816406
  location_z: 3.0
}
lights {
  id: 2
  color: 2
  location_x: -74.1282958984375
  location_y: 484.984619140625
  location_z: 4.0
}
lights {
  id: 3
  color: 3
  location_x: 59.96036911010742
  location_y: 473.6038513183594
  location_z: 5.0
}
```

2.5.2 查看源数据包

原始数据包含车辆采集的各类传感器数据信息，数据导入任务创建完毕后，平台对数据包进行扫描，此模块展示导入成功的数据包。

数据包列表

步骤1 在左侧菜单栏中，选择“数据资产 > 源数据包”。

步骤2 在“数据包”页签，默认显示数据包列表。

步骤3 单击列表右上角  ，可切换至图片视图。

表 2-18 图表视图相关操作

任务	操作步骤
切换视图	单击列表右上角  ，数据包可以由列表和图片两种视图相互切换。

任务	操作步骤
数据日历	1. 单击列表右上角  ，选择年月份，日历中显示代表当日场景片段的色块。 2. 单击色块，在下侧展示采集车辆的采集信息。 3. 鼠标悬停在图片上，单击播放按钮，页面跳转至“回放”页面。
查询数据包	可通过数据包ID、导入任务ID、数据包名称、车辆名称等类型筛选数据包。
查看数据包详情	单击图片下侧的“详情”，可查看数据包详情，具体可参考**数据包详情**。
数据回放	鼠标悬浮在图片上，单击图片，页面自动跳转至“数据回放”模块，可查看采集车辆的路测数据，具体可参考**数据回放**。
删除数据包	单击图片下侧的“删除”，可删除数据包。

----结束

数据包详情

步骤1 在左侧菜单栏中，选择“数据资产 > 源数据包”。

步骤2 选择“数据包”页签，在列表视图时单击指定数据包“操作”栏内的“详情”，进入数据包详情页面。

表 2-19 数据包详情列表

区域	说明
数据包信息	包含数据包名称、状态、描述、视频数据、标签、导入ID等信息。 • 数据回放：单击页面右上角的“进入回放页面”，可回放数据包，具体可参考**数据回放**。 • 场景挖掘：支持数据包进行场景挖掘，单击页面右上角“场景挖掘”，后台会自动进行内置场景挖掘，挖掘后的场景存至数据场景中。具体请参考**场景挖掘**。 • 状态：数据包状态为异常时，在异常状态后显示异常原因。 • 视频数据：单击MP4名称，可在线播放、暂停、支持调节声音，全屏播放，调节播放倍速以及播放方式。当前平台只支持h264编码的MP4可视化播放。 • 标签：单击“编辑”，新增标签或选择已有标签，具体操作请参考**添加标签**。 • 导入ID：单击导入ID，页面跳转至导入任务详情。 • 传输方式：显示数据包输入来源。如对象存储或合规服务。 • 原始数据路径：显示原始数据包存储的路径。 • 数据量：显示数据包数据量的大小。

区域	说明
数据包详情	数据目录、数据类型、标定项ID、数据格式、数据量。
数据处理	可查看此数据包关联的数据处理作业，单击作业ID，可跳转至作业处理界面。
收集信息	车辆名称、标定ID、驾驶模式、采集站点、驾驶路线、采集时间、和总数据量以及各类传感器采集的数据量。

----结束

说明

- 如果数据包异常，一般是由于与数据包同名的yaml文件配置错误，可参照**与数据包同名的yaml配置文件说明**，修改文件内容，再次创建数据收集任务。
- 数据处理算子未运行完毕，不可回放。

2.5.3 场景挖掘

2.5.3.1 场景挖掘

平台支持基于内置场景挖掘算法和场景识别大模型挖掘的能力，对数据包进行场景挖掘。具体操作步骤如下：

步骤1 在左侧菜单栏中，选择“数据资产 > 源数据包”。

步骤2 选择“数据包”页签，执行场景挖掘。

- 勾选需要内置场景挖掘的数据包，单击“场景挖掘”。
- 单击操作栏中的“详情”，进入数据包详情，单击“场景挖掘”。

步骤3 挖掘后的场景可在“数据场景”中查看。

步骤4 本次数据包已经完成场景挖掘时，在数据包详情页再次单击“场景挖掘”时，可选择“重新挖掘”或“跳转至数据场景页面”查看。

说明

当数据包的回放处理未完成时，不支持重新挖掘。

步骤5 查看内置场景挖掘作业信息。

选择“数据处理 > 数据处理”，在“作业总览”页签单击“内部作业”进入内部作业列表页，可查看内置场景的挖掘状态等信息。单击“详情”可以查看内置场景挖掘作业日志。

----结束

2.5.3.2 内置场景挖掘规则

内置场景挖掘算法都是基于规则进行片段挖掘。平台支持的内置场景挖掘规则如下：

道路---道路环境---高速

检验规格:

1. 包含高精地图信息
2. 主车行驶区域road级别type为motorway

道路---道路环境---城市快速路

检验规格:

1. 包含高精地图信息
2. 主车行驶区域road级别type为townExpressway

道路---道路环境---城市主干道

检验规格:

1. 包含高精地图信息
2. 主车行驶区域road级别type为town

道路---道路环境---一般道路

检验规格:

1. 包含高精地图信息
2. 主车行驶区域road级别type为unknown

道路---道路环境---小区内部路

检验规格:

1. 包含高精地图信息
2. 主车行驶区域road级别type为lowSpeed

道路---道路环境---乡村道路

检验规格:

1. 包含高精地图信息
2. 主车行驶区域road级别type为rural

道路---道路环境---停车位

检验规格:

1. 包含高精地图信息
2. 主车行驶区域lane级别type为parking

道路---道路行驶阶段---驶入路口

检验规格:

1. 包含高精地图信息
2. 主车行驶区域road的junction_id由-1变为非-1

道路---道路行驶阶段---驶出路口

检验规格:

1. 包含高精地图信息
2. 主车行驶区域road的junction_id由非-1变为-1

道路---道路行驶阶段---道路中行驶

检验规格:

1. 包含高精地图信息
2. 主车行驶区域road的junction_id非-1

道路---道路行驶阶段---上匝道

检验规格:

1. 包含高精地图信息
2. 主车行驶区域lane级别type为exit

道路---道路行驶阶段---下匝道

检验规格:

1. 包含高精地图信息
2. 主车行驶区域lane级别type为entry

道路---道路行驶阶段---匝道中行驶

检验规格:

1. 包含高精地图信息
2. 主车行驶区域lane级别type为onRamp

道路---道路行驶阶段---匝道分流

检验规格:

1. 包含高精地图信息
2. 主车行驶区域lane级别type为offRamp

道路---道路行驶阶段---匝道合流

检验规格:

1. 包含高精地图信息
2. 主车行驶区域lane级别type为connectingRamp

道路---交通规则---道路限速

检验规格:

1. 包含高精地图信息
2. 主车行驶区域lane级别type中speed值非空

道路---交通规则---红绿灯

检验规格:

1. 包含高精地图信息
2. 主车行驶区域signal非空且type符合OpenDRIVE的红绿灯规范

道路---垂直角度---上坡

检验规格:

1. 过程持续时间超过3秒
2. 全过程主车海拔高度持续上升超过2米

道路---垂直角度---下坡

检验规格:

1. 过程持续时间超过3秒
2. 全过程主车海拔高度持续下降超过2米

主车行为---驾驶方向---直行

检验规格:

1. 主车不处于停车或转弯状态
2. 整个过程持续时间大于10秒

主车行为---驾驶方向---倒车

检验规格:

1. 主车速度<0
2. 整个过程持续时间大于10秒

主车行为---驾驶方向---左转

检验规格:

1. 主车方向盘持续向左打且未停止行驶
2. 整个过程中主车朝向变化大于45°小于160°

主车行为---驾驶方向---右转

检验规格:

1. 主车方向盘持续向右打且未停止行驶
2. 整个过程中主车朝向变化大于45°小于160°

主车行为---驾驶方向---U 型掉头

检验规格：

1. 主车方向盘持续向左打且未停止行驶
2. 整个过程中主车朝向变化大于160°小于200°

主车行为---驾驶方向---主车换道行驶

检验规格：

1. 包含高精地图信息
2. 主车行驶区域在同一road下切换不同的lane

主车行为---加速度---加速

检验规格：

1. 主车的车速每秒加速超过5%
2. 过程持续时间超过3秒
3. 加速前主车的车速高于2米/秒

主车行为---加速度---减速

检验规格：

1. 主车的车速每秒减速超过5%
2. 过程持续时间超过3秒
3. 减速前主车的车速高于5米/秒，主车的车速低于2米/秒时停止检测

主车行为---加速度---急加速

检验规格：

1. 主车的车速每秒加速超过10%
2. 过程持续时间超过3秒
3. 加速前主车的车速高于2米/秒

主车行为---加速度---急减速

检验规格：

1. 主车的车速每秒减速超过10%
2. 过程持续时间超过3秒
3. 减速前主车的车速高于5米/秒，主车的车速低于2米/秒时停止检测

主车行为---加速度---匀速

检验规格：

1. 以15km/h为起点，15km/h以上每10km/h分为一个速度级别
2. 主车的车速处于同一速度级别，持续时间超过10秒

主车行为---驾驶行为---主车跟车

检验规格：

1. 被超车辆与主车横向距离在±5米以内
2. 跟车过程中主车速度高于5米/秒
3. 持续时间超过10秒

主车行为---驾驶行为---主车超车

检验规格：

1. 被超车辆与主车横向距离在±5米以内
2. 被超车辆车速高于5米/秒
3. 超车过程持续少于10秒
4. 超车过程中主车速度高于5米/秒

主车行为---驾驶行为---连续转弯

检验规格：

1. 连续两次或两次以上的主车转弯行为
2. 前一次转弯的结束时间与后一次转弯的开始不超过10秒

主车行为---驾驶行为---紧急制动

检验规格：

主车的车速在1秒内从高于25km/h降低到低于5km/h

他车行为---前车行为---切出

检验规格：

1. 检测在主车前方30米范围内的他车切出行为
2. 主车处于直行状态且车速高于10千米/小时
3. 目标车辆与主车横向距离从1米到远离至2米的过程不超过3秒

他车行为---前车行为---静止

检验规格：

1. 检测在主车前方60米范围内、横向±5米范围内且车速低于1m/s的车辆
2. 持续时间超过3秒且持续过程中始终存在车辆处于上述状态

他车行为---前车行为---前车紧急制动

检验规格：

1. 检测在主车前方30米范围内的他车减速行为
2. 前车的车速在1秒内从高于25km/h降低到低于5km/h

他车行为---旁车行为---切入

检验规格:

1. 检测在主车前方30米范围内的他车切入行为
2. 主车处于直行状态且车速高于10千米/小时
3. 目标车辆与主车横向距离从2米到接近至1米的过程不超过3秒

他车行为---旁车行为---超车

检验规格:

1. 目标车辆与主车横向距离在±5米以内
2. 目标车辆车速高于5米/秒
3. 主车的车速高于5米/秒
4. 超车过程持续少于10秒

他车行为---前车方向---前车直行

检验规格:

1. 目标车在主车前方30米范围内处于直行状态
2. 主车不处于停车或转弯状态
3. 整个过程持续时间大于10秒

他车行为---前车方向---转弯

检验规格:

1. 检测在主车前方60米范围内、横向±5米范围内的车辆
2. 目标车辆每秒持续向相同方向转向
3. 全过程持续时间超过3秒
4. 全过程目标车辆转向幅度超过45°

他车行为---前车加速度---加速

检验规格:

1. 目标车在主车前方30米范围内处于直行状态
2. 目标车的车速每秒加速超过5%
3. 加速持续时间超过3秒

他车行为---前车加速度---减速

检验规格:

1. 检测在主车前方60米范围内、横向±5米范围内的车辆

2. 过程持续时间超过3秒
3. 目标车辆每秒减速超过9%
4. 减速开始前目标车辆速度需高于5米/秒，速度低于2米/秒时停止检测

他车行为---前车加速度---前车均速

检验规格：

1. 以15km/h为起点，15km/h以上每10km/h分为一个速度级别
2. 目标车的车速处于同一速度级别，持续时间超过10秒

对象特征---行人

检验规格：

1. 检测在主车前方60米范围内、横向±5米范围内的目标
2. 目标存在于上述范围内超过3秒

对象特征---机动车类型---卡车、轿车、大客车

检验规格：

1. 检测在主车前方60米范围内、横向±5米范围内的目标
2. 目标存在于上述范围内超过3秒

对象特征---非机动车---自行车

检验规格：

1. 检测在主车前方60米范围内、横向±5米范围内的目标
2. 目标存在于上述范围内超过3秒

时间-白天、夜晚、黎明或傍晚

检验规格：

1. 数据包中提供摄像头数据
2. 摄像头数据中，上述时间规格持续时间超过3秒

天气-晴天、雾天、雨天、雪天、多云、阴天

检验规格：

1. 数据包中提供摄像头数据
2. 摄像头数据中，上述天气规格持续时间超过3秒

补充说明

1. 上述距离范围指物体几何中心距离
2. 有效长度不足10秒的场景片段在输入场景库时会自动在片段前后填充时间使整个片段扩展到10秒

2.5.4 数据回放

数据回放页面可自动播放车辆渲染后的点云图像以及与之同步的视频轨迹，用户选用不同形式进行数据回放，方便排查和定位路测数据存在的问题。

前提要求

3D回放对回放机器配置有以下要求：

1. 回放机器需要GPU硬件。硬件加速的方式：在chrome设置-高级中打开硬件加速 。
2. 机器的参考配置（最低配置）：8核cpu 、UHD620的gpu 、16G内存 、100Mbps带宽。

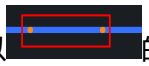
查看数据回放

数据回放页面详细说明如下：

图 2-4 数据回放详情页

表 2-20 数据回放页面详细说明

序号	区域名称	说明
1	主车运行状态数据	自动驾驶和人工接管的相关数据信息。如加速度、速度和方向盘数据，数据随着采集车辆的行驶动态变化。
2	数据列表	• 回放图层参考**回放图层**信息。 • 数据图表参考**数据图表**信息。
3	点云和感知回放区域	采集车辆采集的雷达数据，渲染后得到的点云图像，和随着采集车辆的行驶感知在采集车辆周围出现的其他物体，如其他车辆和行人等。目前可感知的物体类型请见**感知物体类型**。
4	视频回放信息	车辆摄像头采集的数据，经过脱敏处理生成的视频回放信息。支持视频的放大和缩小。

序号	区域名称	说明
5	控制菜单项	 控制数据列表和主车运行状态的显示或者隐藏。  控制视频回放信息的显示或隐藏。
6	视角和2D/3D切换	视角切换：跟车、俯视、驾驶、自由。当在数据回放页面拖动鼠标时，即可切换为自由视角。 2D/3D切换：视频轨迹播放分2D/3D两种播放方式。
7	视频播放控件	控制视频播放暂停回放按钮，支持逐帧播放以及视频播放倍速。 从数据场景进入回放时，在视频进度条上会以  的形式保持数据场景片段的标记，且视频开始的起始点为数据场景片段开始的时间点。
8	创建作业	根据需要，支持回放场景生成其他类型作业的创建。单个数据包的数据片段场景不能超出500个。

回放图层

在回放图层区域，可以选择不同图层，多层次观看数据回放视频。增加主车规划轨迹和感知目标预测轨迹，可以直观检验规控算法和控制模型的有效性及准确性。

- 对比回放源：可选择不同的回放源，进行对比。
- 摄像头：控制车辆摄像头采集的视频数据显示或隐藏。
- 主车定位：控制主车定位信息显示或隐藏。
- 感知目标：包含选项有感知目标ID、物体、类型、朝向、感知框。控制感知目标显示或隐藏。当选择了对比源之后，感知目标会由不同颜色和不同类型的感知框来展示。
- 信号灯：如果在场景片段中包含了信号灯时，回放片段中支持信号灯的展示，控制信号灯信息显示或隐藏。
- 毫米波雷达：控制毫米波雷达图像显示或隐藏。
- 激光雷达：控制3D点云图像显示或隐藏。
- 轨迹：包含有planning_trajectory、localization和predicted_objects。控制主车规划轨迹、主车的车辆行动轨迹和感知目标预测轨迹显示或隐藏。其中不同感知目标拥有不同的预测路径，一个感知目标可以拥有多条预测路径。
- 高精地图：当视频中含有高精地图map_id时，此选项才会显示。显示“路面”，控制高精地图路面显示或者隐藏。

数据图表

在数据回放过程中，数据图表以曲线的形式显示各个数据指标值变化情况。

步骤1 选择“数据图表”页签，单击“添加图表”。

步骤2 选择配置。

步骤3 单击“确认”，左侧数据图表侧以曲线的形式展示数据。

----结束

对比回放源

步骤1 数据回放页面，单击左侧“回放图层”下的“对比回放源”。

步骤2 单击对比项后的“选择”，选择对比项。

步骤3 单击“确定”，感知目标会由不同颜色和不同类型的感知框来展示。

----结束

视频轨迹播放

步骤1 3D回放页面，单击倍速播放选项，选择任意播放速率，视频按选中的速率播放。

步骤2 单击回放图层中的任意图层，控制该图层折叠或者展开，单击任意图层下的任意选项，在回放页面中，显示该选项对应的数据信息，再次单击该图层选项，在回放页面中，隐藏该选项对应的数据信息。

步骤3 生成作业。

单击界面右上角“创建作业”，根据需要创建作业，页面右侧出现作业创建窗口：

1. 数据场景作业。

表 2-21 数据场景作业参数

参数	说明
任务类型	数据场景生成。
开始时间	不得超过视频开始时间。
片段时长	设置片段时长。
场景标签	同一类别标签只可单选，不同类别场景标签支持多选。标签数量不超过50个。可新建标签，操作请参考**标签管理**。

2. 数据抽帧作业。

表 2-22 数据抽帧参数

参数	说明
任务类型	数据抽帧。
开始时间	不得超过视频开始时间。
片段时长	设置片段时长。
抽帧间隔	片段不得超过10秒。

参数	说明
抽帧topic	选择数据包的所有camera和lidar类的topic。
存储目录	选择存储目录。
抽帧结果是否生成数据集	可根据需要，选择抽帧结果是否生成数据集。
名称	简要描述场景，只能包含数字、英文、中文、下划线、中划线。
描述	简要描述数据集，不包含“@#$%^&*< > \”，不超过256个字符。

说明

- 在创建作业期间，支持拖动主进度条查看回放详情，仿真场景生成成功后，该场景会在“仿真服务 > 仿真场景 > 场景”中展示。
- 从数据场景片段进入回放时，在视频进度条上会以[redacted]的形式，保持数据场景片段的标记，在创建作业时，进度条上会以[redacted]的形式表示仿真场景片段，当拖动蓝色片段时长两端靠近场景片段附近的时候，会主动吸附在黄点上，吸附的区间为（-1s，1s）。

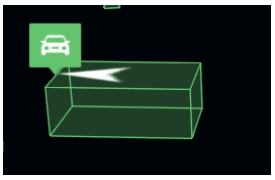
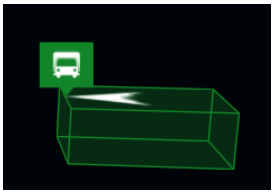
步骤4 单击“确认”，作业状态变为“作业运行中，已开始**秒”，当作业状态为“已完成”时，作业创建成功。

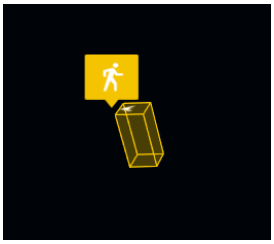
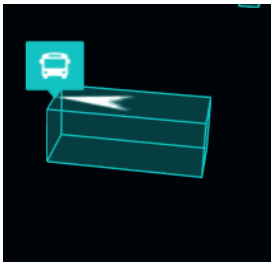
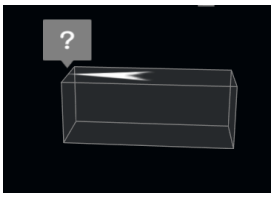
步骤5 创建数据场景作业和抽帧作业成功后，可根据需要单击“跳转到数据场景片段”、“跳转到通用存储”、“跳转到数据集”，平台跳转至相应的界面。

----结束

感知物体类型

表 2-23 感知物体类型

类别	名称	box感知框
轿车	car	
货车	Truck	

类别	名称	box感知框
行人	Pedestrian	
公交车	bus	
自行车	bicycle	
未知障碍物	unknow	

2.5.5 回收站

Octopus平台支持用户删除数据包，执行删除操作的数据包将暂存回收站，等待用户可根据项目情况及时恢复或永久删除。具体操作参考如下：

步骤1 在左侧菜单栏中，选择“数据资产 > 源数据包”。

步骤2 在“数据包”页签，选择指定数据包操作栏内的“更多 > 删除”，根据界面提示删除数据包，删除后的数据包将放入回收站。

步骤3 查看数据包状态。

单击“回收站”，删除后的数据包状态为“待处理”，此状态下的数据包还可进行以下操作：

- 还原：恢复该数据包，单击操作栏中“还原”，删除后的数据包将从回收站中恢复至数据包列表，可继续用于回放、标注等操作。

- 永久删除：该数据包将会被Octopus平台永久删除，单击操作栏中“永久删除”，确认无误后单击“删除”。

须知

永久删除后不可恢复且不能再被使用，请谨慎操作。

----结束

2.6 数据场景

2.6.1 场景展示

Octopus平台处理完原始采集数据后，平台支持内置和自定义场景挖掘算法，可自动提取对应场景行为的片段，展示在数据场景模块中。用户可将其生成单个仿真场景片段，为后续仿真开发做准备。

数据场景依赖以下三个topic：ego_tf（主车定位）、 object_array_vision（目标感知）、vehicle（底盘）。

基于模型的场景挖掘依赖camera（相机）传感器。

自定义场景挖掘算法对topic无特殊要求，由客户算法自行定义。

前提条件

符合以下两种条件中的其中之一，即可在此模块查看数据场景片段。

- 选择自定义挖掘算子（数据标记类型）的作业正常运行完毕（自定义场景片段）。
- 在源数据包详情界面，手动触发“场景挖掘”按钮，并完成数据场景挖掘作业（内置场景片段）。

生成仿真场景

生成仿真场景必须同时依赖以下两个topic：ego_tf(主车定位)、object_array_vision(目标感知)。根据生成路径不同可分为仿真场景库场景和通用存储场景。

仿真场景库场景

步骤1 在左侧菜单栏中单击“数据资产 > 数据场景”。

步骤2 单击数据包ID前的“ ”，可展开数据包下场景片段。

步骤3 生成仿真场景。

- 单个生成仿真场景（片段时长需要大于10秒，小于99秒）：
  鼠标悬停在目标场景片段时间轴上，单击“生成仿真场景”选项。
- 批量生成仿真场景：
  勾选场景片段，单击列表上方的“生成仿真场景”。

步骤4 编辑仿真场景信息。

- 任务类型：仿真场景生成。
- 仿真器：可选择仿真器A和仿真器B。
- 生成路径：可选仿真场景库和通用存储，此处选择仿真场景库。
- 优先级：当前支持S、A、B、C、D。级别顺序为：S > A > B > C > D。
- 场景片段名称：格式为“车辆名称_场景标签_该场景片段ID”，也可以在此处再次选择场景片段。

步骤5 信息编辑结束后，单击“确认”。

步骤6 成功生成仿真场景后，该场景片段在时间轴上会显示“进入仿真场景”。

步骤7 单击“进入仿真场景”，页面自动跳转至“仿真服务 > 场景管理 > 场景”，在此可对其进行运行，编辑和删除等操作。

说明

- 如果该场景片段需要重新生成仿真场景，则需在仿真服务模块删除该场景片段生成的仿真场景。
- 配置了高精地图map_id的数据包，静态场景将使用配置好的高精地图文件显示。未配置map_id的数据包，平台将自动生成一份地图做仿真场景使用，地图参考场景发生的时间戳内的自车轨迹数据。
- 当场景标签关联仿真标签之后，生成的仿真场景推送到仿真场景库，会自带场景标签。

----结束

通用存储场景

步骤1 在左侧菜单栏中单击“数据资产 > 数据场景”。

步骤2 单击数据包ID前的“ ”，可展开数据包下场景片段。

步骤3 生成仿真场景。

- 单个生成仿真场景（片段时长需要大于10秒，小于99秒）：
  鼠标悬停在目标场景片段时间轴上，单击“生成仿真场景”选项。
- 批量生成仿真场景：
  勾选场景片段，单击列表上方的“生成仿真场景”。

步骤4 编辑仿真场景信息。

- 任务类型：仿真场景生成。
- 仿真器：可选择仿真器A和仿真器B。
- 生成路径：可选仿真场景库和通用存储。此处选择通用存储。
- 目录名称：下拉选择目录名称。
- 场景片段名称：格式为“车辆名称_场景标签_该场景片段ID”，也可以在此处再次选择场景片段。

步骤5 信息编辑结束后，单击“确认”。

步骤6 当作业状态为“已完成”时，仿真场景生成成功。

步骤7 单击仿真场景名称，页面自动跳转至“数据资产 > 通用存储 >目录名称（创建作业时选择的）> 数据处理任务”，在此可对其进行查看和删除等操作。详情请参考**通用存储详情**。

说明

- 如果该场景片段需要重新生成仿真场景，则需在仿真服务模块删除该场景片段生成的仿真场景。
- 配置了高精地图map_id的数据包，静态场景将使用配置好的高精地图文件显示。未配置map_id的数据包，平台将自动生成一份地图做仿真场景使用，地图参考场景发生的时间戳内的自车轨迹数据。

----结束

数据场景相关操作

在“数据场景”模块，还可以完成以下操作。

表 2-24 数据场景相关操作

任务	操作步骤
切换视图	单击操作列的“切换视图”，场景片段可以由列表和时间轴两种视图相互切换。
删除场景片段	选择目标场景片段，单击列表上方的“删除”，可删除单个或多个场景片段。
回放场景片段	鼠标悬停在目标场景片段时间轴上，单击带时间段的按钮，页面自动跳转至“数据回放”模块，详情请参考**数据回放**。
筛选场景片段	在搜索输入框中输入搜索条件查询场景片段。

2.6.2 标签管理

标签管理中的标签分为平台内置标签和用户自定义标签。

平台内置标签详情请查看**表1 内置标签列表**。用户还可根据实际需求，自定义设置需要的标签。

当前支持两种方式添加标签：手动在线创建和导入标签。

表 2-25 内置标签列表

场景分类	子类	标签
道路	道路环境	高速
		城市快速路
		城市主干道
		一般道路
		小区内部路
		乡村道路

场景分类	子类	标签
		停车位
	道路行驶阶段	驶入路口
		驶出路口
		道路中行驶
		上匝道
		下匝道
		匝道中行驶
		匝道分流
		匝道合流
	交通规则	道路限速
		红绿灯
	垂直角度	上坡
		下坡
主车行为	驾驶方向	主车直行
		主车倒车
		主车左转
		主车右转
		主车U型掉头
		主车换道行驶
	加速度	主车加速
		主车减速
		主车急加速
		主车急减速
		主车均速
	驾驶行为	主车跟车
		主车超车
		主车连续转弯
		主车紧急制动
	速度	主车低速
		主车中速

场景分类	子类	标签
		主车高速
他车行为	前车行为	前车切出
		前车静止
		前车紧急制动
	旁车行为	旁车切入
		旁车超车
	前车方向	前车直行
		前车转弯
	前车加速度	前车加速
		前车减速
		前车均速
对象特性	行人	行人
	机动车类型	轿车
		卡车
		大客车
	非机动车	自行车
时间	-	白天
	-	夜晚
	-	黎明或傍晚
天气	-	晴天
	-	雾天
	-	雨天
	-	雪天
	-	多云
	-	阴天
-	-	接管

手动添加标签

步骤1 在左侧菜单栏中选择“数据资产 > 数据场景”。

步骤2 单击“标签管理”，单击“新增分类”，输入分类名称，新建分类。

步骤3 单击分类名称后的，输入标签路径名称，用户将新建一个标签路径。

步骤4 单击分类后的，输入标签名称和描述信息。

步骤5 选择仿真标签，单击“选择标签”。

勾选左侧目标标签后，右侧界面会显示左侧勾选的目标标签。

在筛选框中输入内容，查找目标标签。

多项搜索：可根据需要决定是否启用多项搜索，输入多个关键字，中间用“;”隔开，可搜索多个关键字。

步骤6 单击“确认”，界面展示含有目标标签的场景。

步骤7 单击“保存”，保存信息。

----结束

导入标签

步骤1 在左侧菜单栏中选择“数据服务 > 数据场景”。

步骤2 单击“标签管理”，单击“ 导入标签”。

步骤3 下载标签模板。

单击 “下载模板”，下载标签文件模板“template.json”。

步骤4 填写标签信息。

根据模板填写信息，单次只支持导入一个标签文件，但可以在json文件中写入多个标签：

```
[
  {
    "name": "新标签1",
    "description": "用途1"
  },
  {
    "name": "新标签2",
    "description": "用途2"
  }
]
```

步骤5 导入标签文件。

单击 “选择文件”，单击“确定”，将编写好的json文件上传。上传成功后刷新页面，可以在页面上看到新创建的标签。

说明

- 批量导入支持同时导入最多200个标签。
- 当前导入标签只支持在自建标签下导入。

----结束

标签管理相关操作

在“标签管理”页面，还可以完成以下操作。

表 2-26 标签管理相关操作

任务	操作步骤
修改标签	单击标签后  ，修改标签描述和仿真标签。 **说明** • 内置标签不允许修改。 • 当场景标签关联仿真标签之后，生成的仿真场景推送到仿真场景库，会自带场景标签。
删除标签或分类	单击标签或分类后的  ，可删除指定标签或分类。 **说明** 如果标签在使用中时，则该标签不可删除。
查看标签	单击标签后  ，可查看标签详情。
搜索标签	在搜索输入框中输入搜索条件，按回车键即可查询。

2.7 数据集

2.7.1 创建数据集

平台支持用户创建数据集，通过数据提取将原始数据提取合并，进行数据预览。根据数据来源不同，创建的方式不同。

数据集来源分类

数据集来源分如下类型：

- 本地：直接从本地计算机上传数据集。

  说明

  需提前准备文件保存至本地，且需满足如下条件：

  - 仅支持上传扩展名为png、jpg、jpeg、pcd、json、yaml、txt、xml、csv、laz、las、bin、npy、wav、flac、mp3、m4a的文件。
  - 单个文件最大为100MB，文件夹最大为10GB，文件数量最多为10000。
  - 每一级文件或文件夹路径最长不能超过255字符。文件路径最长不能超过896字符。
- 标注：通过现有的标注任务，快速创建新的数据集。
- OBS：支持从对象存储服务（OBS）中导入数据集。

  说明

  需提前将数据集上传至OBS，并获取对应的访问密钥、私有访问密钥和OBS地址。
- 通用存储：可以从八爪鱼通用存储服务导入数据集。
- 数据集：基于现有数据集，创建子集。

创建数据集步骤

步骤1 创建用于数据集增加数据时进行数据筛选和格式转换的镜像。可参考**9.2 制作镜像（数据集）**。

步骤2 在左侧菜单栏中选择“数据资产 > 数据集 ”。

步骤3 在“数据集”页签单击“创建数据集”，填写数据集名称和描述信息。

步骤4 根据不同数据来源配置相关参数，配置完成后勾选“我已阅读并同意《八爪鱼自动驾驶云服务声明》”，单击“创建”。

- 数据来源：本地

表 2-27 数据来源为本地的数据集

参数	说明
数据来源	选择数据来源，此处选择“本地”。
数据类型	选择数据类型，可选择“图片”、“3D点云”、“音频”和“文本”。
数据格式	选择数据格式，可选择“OCTOPUS”、“USER_DEFINE”和自定义格式。
标注状态	根据需要设置标注状态，可设置“未标注”、“已标注”。
自定义镜像	选择已创建的镜像以及镜像版本。
启动脚本路径	选择脚本的启动文件路径，文件路径为在脚本中的相对路径，当前只支持.py类型的启动文件，如/root/dataset.py。
选择文件	选择已准备好的本地文件上传。

- 数据来源：标注

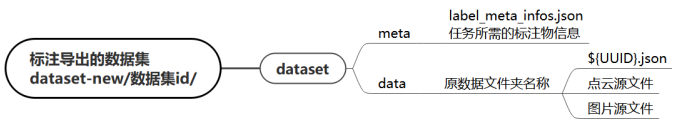
表 2-28 数据来源为标注的数据集

参数	说明
数据来源	选择数据来源，此处选择“标注”。
数据类型	选择数据类型，可选择“图片”、“3D点云”、“音频”和“文本”。
数据格式	选择数据格式，仅支持“OCTOPUS”格式。
标注状态	设置标注状态，仅支持“已标注”状态。
标注项目	选择现有的标注项目。
自定义镜像	选择已创建的镜像以及镜像版本。

参数	说明
启动脚本路径	选择脚本的启动文件路径，文件路径为在脚本中的相对路径，当前只支持.py类型的启动文件，如/root/dataset.py。
批次任务	选择标注项目中的批次任务。
标注	选择批次任务后选择标注。
标注帧	可选择“全部标注帧”、“有效标注帧”、“无效标注帧”。
标注数据集	可选择“全部数据集”、“已标注数据集”。
难例数据集	可选择“全部数据集”、“仅难例数据集”、“非难例数据集”。

- 数据来源：OBS

表 2-29 数据来源为 OBS 的数据集

参数	说明
数据来源	选择数据来源，此处选择“OBS”。
数据类型	选择数据类型，可选择“图片”、“3D点云”、“音频”和“文本”。
数据格式	选择数据格式，可选择“OCTOPUS”、“USER_DEFINE”和自定义格式。
标注状态	根据需要设置标注状态，可设置“未标注”、“已标注”。
自定义镜像	选择已创建的镜像以及镜像版本。
启动脚本路径	选择脚本的启动文件路径，文件路径为在脚本中的相对路径，当前只支持.py类型的启动文件，如/root/dataset.py。
访问密钥	请输入访问密钥（AK）。
私有访问密钥	请输入私有访问密钥（SK）。
源路径	选择OBS桶中的地址，建议选择data、meta的上一级目录。 

说明

- 访问密钥ID（AK）和私有访问密钥（SK），在导入数据时，通过AK识别访问用户的身份，通过SK对请求数据进行签名验证，用于确保请求的机密性、完整性和请求者身份的正确性。
- 输入的访问密钥和私有访问密钥需要具有OBS服务如下权限：

  obs:object:GetObject、obs:object:PutObject、obs:bucket:ListAllMyBuckets、obs:bucket:ListBucket

- 数据来源：通用存储

表 2-30 数据来源为通用存储的数据集

参数	说明
数据来源	选择数据来源，此处选择“通用存储”。
数据类型	选择数据类型，可选择“图片”、“3D点云”、“音频”、“文本”和“视频”。
数据格式	选择数据格式，可选择“OCTOPUS”、“USER_DEFINE”和自定义格式。
标注状态	根据需要设置标注状态，可设置“未标注”、“已标注”。
自定义镜像	选择已创建的镜像以及镜像版本。
启动脚本路径	选择脚本的启动文件路径，文件路径为在脚本中的相对路径，当前只支持.py类型的启动文件，如/root/dataset.py。
存储目录	选择存储目录。
筛选条件	根据需要选择筛选条件。

- 数据来源：数据集

表 2-31 数据来源为数据集的数据集

参数	说明
数据来源	选择数据来源，此处选择“数据集”。
数据集形式	仅支持“数据集子集”。
源数据集	选择源数据集。源数据集仅支持已发布的版本
自定义镜像	选择已创建的镜像以及镜像版本。
启动脚本路径	选择脚本的启动文件路径，文件路径为在脚本中的相对路径，当前只支持.py类型的启动文件，如/root/dataset.py。
标签	根据需要选择标签。
自定义属性	根据需要选择自定义属性。

参数	说明
标注物	根据需要选择合适的标注物。

----结束

2.7.2 数据集管理

数据集建成后，在“数据集”列表，可对数据集进行以下操作。

表 2-32 数据集列表相关操作

任务	操作步骤
查找数据集	在搜索输入框中输入搜索条件，按回车键即可查询。
查看数据集详情	在“数据集”列表，单击目标数据集名称，查看数据详情、数据预览、待发布区、版本管理和子集管理，详情请参考**数据集详情**。 **说明** • 数据集详情页也可以单击 “添加版本” 增量更新数据集。 • 点云数据集因在线渲染点云图像，故会出现短暂“加载中”过程，请用户耐心等待。 • 视频类、压缩文件不支持预览。
删除数据集	选择单个数据集，单击操作栏的“删除”，删除数据集。
添加数据	• 在“数据集”列表，选择操作栏的“添加数据”，填写数据信息，单击“创建”。 • 在数据集详情页，也可以单击 “添加数据” ，为数据集添加数据。 新添加的数据会先进入待发布区。添加数据时，需要设置数据集输出模式及目标路径（仅数据集数据格式不为OCTOPUS时可设置），其他参数可参考**步骤4**： • 模式： – 全量替换：全量替换目标路径下待发布区中的所有文件和文件夹。 – 增量上传：增量添加文件和文件夹到目标路径下。 **说明** • 全量替换：仅与已发布数据进行比较，如果新添加的文件与已发布数据出现重名文件，则当MD5相同时不重复上传文件；MD5不同时，重命名新添加的文件后继续上传文件。 • 增量上传：与已发布数据和待发布区数据进行比较，如果仅与待发布区数据出现重名文件且MD5不同时，替换文件，其他规则与全量替换相同。 • 目标路径：选择数据集存放的OBS目标路径，若不选择则默认存放在数据集根目录下。当数据格式为OCTOPUS格式时，目标路径固定为根目录，不可设置。

任务	操作步骤
导出数据集	单击操作栏的“导出”，即可导出数据集，具体步骤参考**导出任务**。
缓存数据集	当缓存加速状态为“未缓存”、“缓存过期”、“缓存失败”时，选择操作栏中的“更多 > 缓存加速”，可缓存数据集至“数据缓存”模块。

数据集详情

步骤1 在“数据集”列表，单击数据集名称，进入数据集详情界面。

步骤2 查看数据详情。

数据详情页签展示了数据集ID、名称、描述、数据来源、数据集类型等信息。

同时包含“待发布区”、“版本管理”和“子集管理”三个模块。

步骤3 查看数据预览。

数据预览页签支持查看OCTOPUS格式数据集的标注物、标签和自定义属性信息。

文件树可选择路径，单击数据名称查看预览数据。预览点云数据时，支持放大、缩小、旋转点云图片。

步骤4 待发布区。

表 2-33 待发布区相关操作

功能	步骤
添加数据	单击“添加数据”，给数据集添加新数据。详情可参考**添加数据**。
发布版本	单击操作栏中的“发布”，将待发布数据发布为一个正式版本。每个正式版本记录单独的索引文件。
删除数据	选择操作栏中的“更多 > 删除”，删除已添加至待发布区的数据。
日志	选择操作栏中的“更多 > 日志”，界面跳转至日志列表，在日志列表可查看或者下载日志。
导出	选择操作栏中的“更多 > 导出”，导出待发布区域的数据，具体请参考**导出任务**。
上传文件	本地数据集且状态为中断或初始化，单击“上传文件”上传文件。
停止	选择操作栏中的“更多 > 停止”，停止正在运行或停止中的任务。

步骤5 版本管理。

表 2-34 版本管理相关操作

功能	步骤
搜索版本	在搜索输入框中输入搜索条件，按回车键即可查询。
查看日志	单击操作栏的“日志”，界面跳转至日志列表，在日志列表可查看或者下载日志。
导出版本	单击操作栏的“导出”，可选择导出数据集，具体请参考**导出任务**。
删除版本	单击操作栏的“删除”，可删除当前版本。

步骤6 子集管理。

表 2-35 子集管理相关操作

功能	步骤
搜索子集	在搜索输入框中输入搜索条件，按回车键即可查询。
添加子集	单击“添加子集”，填写信息，添加数据集子集。 **说明** 只有满足条件的数据集才能添加子集。 • 有版本且是已发布状态。 • 非内置数据集。
删除子集	单击操作栏的“删除”，可删除子集。

----结束

2.7.3 导出任务

导出数据集

步骤1 单击数据集列表操作栏的“导出”，选择数据目的地。

- 数据目的地：默认选择OBS。
- 访问密钥：请输入访问密钥（AK）。
- 私有访问密钥：请输入私有访问密钥（SK）。
- OBS目录：请指定数据集导出后存放的目录。

说明

- 访问密钥ID（AK）和私有访问密钥（SK），在导入数据时，通过AK识别访问用户的身份，通过SK对请求数据进行签名验证，用于确保请求的机密性、完整性和请求者身份的正确性。
- 输入的访问密钥和私有访问密钥需要具有OBS服务如下权限：
  obs:object:GetObject、obs:object:PutObject、obs:bucket:ListAllMyBuckets、obs:bucket:ListBucket

步骤2 勾选云服务声明，单击“确认”，等待数据集导出。

步骤3 查看导出的数据集。

单击“导出任务”，即可看到导出任务，可对任务进行删除操作，OCTOPUS格式数据集目录结构如下：

- 导出的数据集文件目录结构可参考如下（图片标注类型）

```
├─{数据集名称}_{时间戳}
  ├─ data
    ├─ 帧目录名称1
      ├─图片名称1.jpg      #对应的已标注图片
      ├─帧目录名称1.json   #该标注图片的所有标注信息
    ├─帧目录名称2
      ├─图片名称2.jpg      #对应的已标注图片
      ├─帧目录名称2.json   #该标注图片的所有标注信息
      …
  ├─ meta
    └─ label_meta_infos.json  #该数据集所有标注物信息
```

- 导出的数据集文件目录结构（点云标注类型）

```
├─{数据集名称}_{时间戳}
  ├─ data
    ├─帧目录名称1
      ├─点云名称1.pcd    #对应的已标注点云
      ├─图片名称1.jpg    #该点云图像对应的已标注图片
      ├─帧目录名称1.json   #该标注点云的所有标注信息
    ├─帧目录名称2
      ├─点云名称2.pcd    #对应的已标注点云
      ├─图片名称2.jpg    #该点云图像对应的已标注图片
      ├─帧目录名称2.json   #该标注点云的所有标注信息
      …
  ├─ meta
    └─ label_meta_infos.json #该数据集所有标注物信息
```

----结束

导出任务相关操作

在“导出任务”页签，也可以进行以下操作：

表 2-36 导出任务相关操作

任务	操作步骤
查找导出任务	在搜索输入框中输入搜索条件，按回车键即可查询。
查看数据集详情	单击数据集名称，界面跳转至数据集详情页。
删除导出任务	单击导出任务操作栏中的“删除”，删除导出任务。

2.7.4 OCTOPUS 数据集格式说明

2.7.4.1 图片标注数据集文件说明

OCTOPUS 格式文件基本要求（图片标注）

上传的OCTOPUS格式数据集需包含以下文件。

```
├─ 时间戳1
   ├─时间戳1.jpg    #时间戳1对应的已标注图片
   ├─时间戳1.json   #时间戳1内该标注图片的所有标注信息
├─ 时间戳2
   ├─时间戳2.jpg    #时间戳2对应的已标注图片
   ├─时间戳2.json   #时间戳2内该标注图片的所有标注信息
```

标注数据.json 文件说明

数据集中必含“.json”文件，用于集合该时间戳已标注图片的所有标注数据信息，包括该图片所在的项目id、数据包id、图片上所有标注框信息等。上传数据集前请保证“.json”文件内容正确。“.json”文件编写的参考样例如下：

```
{
   "frame_id" : 1,
   #帧序号
   "batch_task_id" : 922,
   #批次任务ID
   "project_id" : "ca.....",
   #资源域ID
   "label_mode" : "manual",
   #标注类型：auto和manual两种
   "status" : "labeled",
   #标注任务状态：unlabeled、labeled、unconfirmed、confirmed、all五种
   "sample_type" : "IMAGE",
   #样本类型：包含“IMAGE”、“POINT_CLOUD”、“AUDIO”（音频）、“TEXT”（文本）
   "des_order" : "",
   #此份数据对应的原始数据包描述
   "tag_names" : [],
   #标签名称
   "valid" : true,
   #是否有效，包含“true”和“false”两种
   "create_time" : 1683185878405,
   #标注的创建时间
   "difficult" : false,
   #是否难例，包含“true”难例和“false”非难例
   "label_counts" : [{
   #各类标注物的个数统计
         "label_meta_id" : 1848,
         #标注物使用的标签ID
         "label_num" : 1,
         #标注物个数
         "label_meta_name" : "V3D0504",
         #标注物名称
         "label_meta_desc" : "V3D0504",
         #标注物描述
         "label_meta_attr" : "{\"优先级\":\"0,1\"}",
         #标注物额外属性
         "label_meta_shape" : "multiBox",
         #标注物形状，包含“bndbox、line、circle、polygon、points、dashed、cube_3d、multiBox、
polygon_3d_v2、audio、text、line_3d、dash_3d、line_dash_3d、dash_line_3d、double_line_3d、
double_dash_3d”
         "label_meta_color" : "#7ed321",
         #标注物颜色信息
         "level" : 0
      }, {
```



```
        "label_meta_id" : 1849,
        "label_num" : 1,
        "label_meta_name" : "圆0504",
        "label_meta_desc" : "圆0504",
        "label_meta_attr" : "{\"优先级\":\"0,1\"}",
        "label_meta_shape" : "circle",
        "label_meta_color" : "#417505",
        "level" : 0
      }, {
        "label_meta_id" : 1845,
        "label_num" : 3,
        "label_meta_name" : "线0504",
        "label_meta_desc" : "线0504",
        "label_meta_attr" : "{\"优先级\":\"0,1\"}",
        "label_meta_shape" : "line",
        "label_meta_color" : "#f5a623",
        "level" : 0
      }, {
        "label_meta_id" : 1844,
        "label_num" : 1,
        "label_meta_name" : "点0504",
        "label_meta_desc" : "点0504",
        "label_meta_attr" : "{}",
        "label_meta_shape" : "points",
        "label_meta_color" : "#d0021b",
        "level" : 0
      }, {
        "label_meta_id" : 1846,
        "label_num" : 1,
        "label_meta_name" : "框0504",
        "label_meta_desc" : "框0504",
        "label_meta_attr" : "{\"优先级\":\"0,1\"}",
        "label_meta_shape" : "bndbox",
        "label_meta_color" : "#f8e71c",
        "level" : 0
      }, {
        "label_meta_id" : 1847,
        "label_num" : 1,
        "label_meta_name" : "多边形0504",
        "label_meta_desc" : "多边形0504",
        "label_meta_attr" : "{\"优先级\":\"0,1\"}",
        "label_meta_shape" : "polygon",
        "label_meta_color" : "#8b572a",
        "level" : 0
      }
    ],
    "image_meta_info" : {
    #图片信息
      "id" : "c7...de",
      "name" : "hash0-1590980980006.jpg",
      #图片名称
      "source" : "https://octopus-raw-ca.../label-data/task-922/data/hash0-1590980980006/
hash0-1590980980006.jpg",
      #图片源的obs路径url
      "sensor" : "default_camera",
      #传感器类型
      "timestamp" : 1683185878405,
      #时间戳
      "calibration_item_id" : 0,
      #标定项ID
      "size" : {
      #图片尺寸
        "width" : 1920,
        "depth" : 3,
        "height" : 1080
      }
    },
    "label_task_id" : 21376,
    #批次子任务ID
```



```
  "partitionId" : 20220826,
  "label_update_time" : 1683187695480,
  #标注最近更新时间
  "prefix_folder" : "hash0-1590980980006",
  "image_id" : "9f...46",
  #图片id
  "inspection" : 0,
  "labels" : [
    #标注物信息
    {
        "label_meta_id" : 1844,
        #标注物对应的标签ID
        "create_time" : 1683187362541,
        "name" : "点0504",
           #标注物名称
        "shape_type" : "points",
           #标注物形状：点
        "serial_number" : 0,
        #该帧中标注物唯一自增id
        "label_object_id" : 0,
     #标注物合成对象的唯一自增id，如果标注物之间没有合成则与   serial_number保持一致，追踪任务中同一物体在不同帧中此字段相同
        "attribute" : "",
        #标注物属性
        "label_meta_name" : "点0504",
        "points" : {
        #点的坐标信息
           "size" : 1,
           "points" : [{
                 "xpoint" : 1233.4807,
                 "ypoint" : 689.4183
              }
           ]
        }
     }, {
        "label_meta_id" : 1845,
        "create_time" : 1683187374024,
        "line" : {
        #线的坐标信息
           "size" : 4,
           "points" : [{
                 "xpoint" : 901.138,
                 "ypoint" : 553.583
              }, {
                 "xpoint" : 731.36,
                 "ypoint" : 630.367
              }, {
                 "xpoint" : 618.153,
                 "ypoint" : 681.566
              }, {
                 "xpoint" : 360.516,
                 "ypoint" : 798.086
              }
           ]
        },
        "name" : "线0504",
        "shape_type" : "line",
        #标注物形状:线
        "serial_number" : 1,
        "label_object_id" : 1,
        "attribute" : "{\"优先级\":\"1\"}",
        "label_meta_name" : "线0504"
     }, {
        "label_meta_id" : 1846,
        "create_time" : 1683187387330,
        "bndbox" : {
        #矩形框坐标信息
           "ymin" : 545.4334,
           "xmin" : 1158.3188,
```



```
            "ymax" : 705.71844,
            "xmax" : 1436.3274
         },
         "name" : "框0504",
         "shape_type" : "bndbox",
#标注物形状:矩形框
         "serial_number" : 2,
         "label_object_id" : 2,
         "attribute" : "{\"优先级\":\"1\"}",
         "label_meta_name" : "框0504"
      }, {
         "label_meta_id" : 1847,
         "create_time" : 1683187417245,
         "polygon" : {
         #多边形的坐标信息
            "size" : 3,
            "points" : [{
                  "xpoint" : 135.03,
                  "ypoint" : 482.94937
               }, {
                  "xpoint" : 84.318344,
                  "ypoint" : 554.4891
               }, {
                  "xpoint" : 135.03,
                  "ypoint" : 482.94937
               }
            ]
         },
         "name" : "多边形0504",
         "shape_type" : "polygon",
         #标注物形状：多边形
         "serial_number" : 3,
         "label_object_id" : 3,
         "attribute" : "{\"优先级\":\"1\"}",
         "label_meta_name" : "多边形0504"
      }, {
         "label_meta_id" : 1848,
         "create_time" : 1683187426497,
         "multiBox" : {
         #2.5D框的坐标信息
            "size" : 10,
            "points" : [{
                  "xpoint" : 475.06976,
                  "ypoint" : 645.49835
               }, {
                  "xpoint" : 602.3017,
                  "ypoint" : 645.49835
               }, {
                  "xpoint" : 602.3017,
                  "ypoint" : 537.2833
               }, {
                  "xpoint" : 475.06976,
                  "ypoint" : 537.2833
               }, {
                  "xpoint" : 475.06976,
                  "ypoint" : 645.49835
               }, {
                  "xpoint" : 664.7857,
                  "ypoint" : 632.3677
               }, {
                  "xpoint" : 664.7857,
                  "ypoint" : 537.2833
               }, {
                  "xpoint" : 602.3017,
                  "ypoint" : 537.2833
               }, {
                  "xpoint" : 602.3017,
                  "ypoint" : 645.49835
               }, {
```



```
                    "xpoint" : 664.7857,
                    "ypoint" : 632.3677
                }
            ]
        },
        "name" : "V3D0504",
        "shape_type" : "multiBox",
        #标注物形状：2.5D
        "serial_number" : 4,
        "label_object_id" : 4,
        "attribute" : "{\"优先级\":\"1\"}",
        "label_meta_name" : "V3D0504"
    }, {
        "label_meta_id" : 1849,
        "create_time" : 1683187565067,
        "name" : "圆0504",
        "shape_type" : "circle",
        #标注物形状：圆
        "serial_number" : 5,
        "label_object_id" : 5,
        "attribute" : "{\"优先级\":\"1\"}",
        "label_meta_name" : "圆0504",
        "circle" : {
        #圆的坐标信息
            "xcenter" : 795.6399,
            "ycenter" : 554.03625,
            "radius" : 31.255125
        }
    }
  ]
}
```

必须字段样例

数据集可视化

“.json”文件中必须包含label_counts和labels字段信息。

创建标注任务

“.json”文件中必须包含label_counts和labels字段信息。如果需要json文件中已有的标注信息在平台上直接展示，则label_counts里面的标注物名称、描述、形状、额外属性需要和创建任务使用的平台标签信息保持一致。示例如下：

```
{"label_counts" : [
#标注对象类型的个数统计
{
        "label_meta_id" : 1846,
        #标注物ID
        "label_num" : 1,
        #标注物个数
        "label_meta_name" : "框0504",
        #标注物名称
        "label_meta_desc" : "框0504",
        #标注物描述
        "label_meta_attr" : "{\"优先级\":\"0,1\"}",
        #标注物额外属性
        "label_meta_shape" : "bndbox",
        #标注物形状：矩形框
        "label_meta_color" : "#f8e71c"
        #标注物颜色信息
    }
  ],
"labels" : [
{
        "label_meta_id" : 1846,
        "bndbox" : {
            "ymin" : 545.4334,
            "xmin" : 1158.3188,
```



```
            "ymax" : 705.71844,
            "xmax" : 1436.3274
         },
         "name" : "框0504",
         "shape_type" : "bndbox",
"serial_number" : 2,
         #该帧中标注物唯一自增id
         "label_object_id" : 2,
         #标注物合成对象的唯一自增id，如果标注物之间没有合成则与serial_number保持一致，追踪任务中同一物体在不同帧中此字段相同
         "attribute" : "{\"优先级\":\"1\"}",
         "label_meta_name" : "框0504"
      }
   ]
}
```

2.7.4.2 点云标注数据集文件说明

OCTOPUS 格式文件基本要求（点云标注）

上传的OCTOPUS格式数据集需包含以下文件。

```
.
├─ 文件夹1
   ├─ 点云1.pcd          #点云文件
   ├─ 图片1.jpg          #该点云图像对应的已标注图片
   ├─ 点云1.json         #该点云的所有标注信息
├─ 文件夹2
   ├─ 点云2.pcd          #点云文件
   ├─ 图片2.jpg          #该点云图像对应的已标注图片
   ├─ 点云2.json         #该点云的所有标注信息
```

说明

Metadata.xml样例

<?xml version="1.0" encoding="UTF-8"?>

<ModelMetaData>

<SRS>EPSG:32650</SRS>

<SRSOrigin>262808.2870,3375951.2110,20.6160</SRSOrigin>

</ModelMetaData>

标注数据.json 文件说明

数据集中必含“.json”文件，用于集合该时间戳已标注点云的所有标注数据信息，包括该点云所在的项目id、数据包id、图片上所有标注框信息等。上传数据集前请保证“.json”文件内容正确。“.json”文件编写的参考样例如下：

3D立方体框数据：

该文件说明适用于3D目标识别，3D目标追踪，3D联合任务。

```
{
 "frame_id": 11,
 #帧序号
 "batch_task_id": 2284,
 #批次任务ID
 "project_id": "da...0f",
 #资源域ID
 "label_mode": "auto",
 #标注类型：auto和manual两种
 "status": "labeled",
 #标注任务状态：unlabeled、labeled
 "sample_type": "POINT_CLOUD",
```



```
 #样本类型：包含“IMAGE”（图片）、“POINT_CLOUD”（点云）、“AUDIO”（音频）、“TEXT”（文本）
 "des_order": "",
 #此份数据对应的原始数据包描述
 "tag_names": [
  "a_in_premodel",
  "b_in_premodel"
 ],
 #标签名称
 "valid": true,
 #是否有效，包含“true”和“false”两种
 "create_time": 1669339994173,
 #标注的创建时间
 "difficult": false,
 #是否难例，包含“true”难例和“false”非难例
 "label_counts": [
 #各类标注物的个数统计
  {
   "label_meta_id": 5414,
   #标注物使用的标签ID，
   "label_num": 1,
   #标注物个数
   "label_meta_name": "Car",
   #标签名称
   "label_meta_desc": "自采目标识别",
   #标签描述
   "label_meta_attr": "{\"优先级\":\"0,1\"}",
   #标签额外属性
   "label_meta_shape": "cube_3d",
   #标签形状 包含“bndbox、line、circle、polygon、points、dashed、cube_3d、multiBox、polygon_3d_v2、audio、text、line_3d、dash_3d、line_dash_3d、dash_line_3d、double_line_3d、double_dash_3d”
   "label_meta_color": "#d0021b",
   #标签颜色
   "level": 0
  }
 ],
 "point_cloud_meta_info": {
  #3D点云的标注属性信息，包含标定项id、图片名称、传感器类型、图片大小、图片源的obs路径url、时间戳
  "id": "6a275591-53e1-406e-820e-ad077ae1da49",
  "name": "lidar-269-19.pcd",
  #点云名称
  "source":"https://octopus-raw-da.../label-data/task-2284/data/8/lidar-269-19.pcd",
  #点云源的obs路径url
"calibration_item_id": 269,
  #标定项ID
"sensor": "lidar",
  #传感器类型
  "timestamp": 1669339994173,
  "size": 0
 },
 "image_meta_infos": [
  #图片属性信息，包含标定项id、图片名称、传感器类型、图片大小、图片源的obs路径url、时间戳
  {
   "id": "3b...78",
   "name": "camera03-268-19.jpg",
   "source": "https://octopus-raw-da7.../label-data/task-2284/data/8/camera03-268-19.jpg",
   "sensor": "camera03_0",
   #传感器名称
   "timestamp": 1669339994173,
   "calibration_item_id": 268,
   #标定项ID
   "size": {
   #图片尺寸
    "width": 1920,
    "depth": 3,
    "height": 1020
   }
  }
```



```
],
"label_task_id": 1372,
 #批次子任务ID
"partitionId": 20221124,
"label_update_time": 1688457112216,
 #标注最近更新时间
"prefix_folder": "8",
"image_id": "9a...a5",
"inspection": 0,
"labels": [
 #标注物信息
  {
    "label_meta_id": 5414,
    #标注物对应的标签ID
    "create_time": 0,
    "name": "Car",
    #标注物名称
    "shape_type": "cube_3d",
    #标注物形状：3D立体框
    "serial_number": 0,
    "label_object_id": -1,
    "attribute": "",
    #标注物属性
    "label_meta_name": "Car",
    "inspection": {
    #审核属性
      "miss_label_error": false,
       #漏标
      "vehicle_direction_error": false,
      #车头方向错误
      "attribute_error": false,
      #属性错误
      "out_range_label_error": false,
      #未贴合
      "anchor_error": false,
      #锚点错误
      "classification_error": false,
      #类别错误
      "correct_label": true,
      "extra_label_error": false
      #多标
    },
    "cube_3d": {
      "orientation": 1.53980839,
      "alpha": 1.527701791334979,
      "rotation": {
      #3D框旋转角
        "x": 0.0,
        "y": 0.0,
        "z": 1.53980839
      },
      "bndboxs": [
      #3D联合标注中2D框的标注信息
        {
          "label_meta_id": 5414,
          "bndbox": {
            "ymin": 508.54608,
            "xmin": 926.8102,
            "ymax": 727.8024,
            "xmax": 1189.3324
          },
          "label_meta_color": "#d0021b",
          "serial_number": 2,
          "sensor": "camera03_0",
          "attribute": "{}",
          "source": "https://octopus-raw-da.../label-data/task-2284/data/8/camera03-268-19.jpg",
          "label_meta_name": "Car",
          "inspection": {
            "miss_label_error": false,
```



```
            "vehicle_direction_error": false,
            "attribute_error": false,
            "out_range_label_error": false,
            "anchor_error": false,
            "classification_error": false,
            "correct_label": true,
            "extra_label_error": false
          }
        }
      ],
      "serial_number": 2,
      #标注物合成对象的唯一自增id，如果标注物之间没有合成则与serial_number保持一致，追踪任务中同一物体在不同帧中此字段相同
      "location": {
      #3D框中心点坐标
        "x": 0.6671804785728455,
        "y": 15.472203254699707,
        "z": -1.1619998216629028
      },
      "attribute":  "{\"优先级\":\"1\"}",
      "dimensions": {
      #3D框长宽高
        "length": 4.557755470275879,
        "width": 2.0348410606384277,
        "height": 1.4403225183486938
      }
    }
  }
 ],
 "labels_ext": {
   "track_points": [
     {
       "serial_number": 1,
       "points": [

       ]
     }
   ]
 }
}
```

3D语义分割数据:

除label_counts，labels和label_ext外的其余字段说明参考3D立方体框数据中的样例文件。

```
{
 "label_counts": [
   {
     "label_meta_id": 1853,
     "label_num": 1,
     "label_meta_name": "truck",
     "label_meta_desc": "v0504",
     "label_meta_attr": "{\"优先级\":\"0,1\"}",
     "label_meta_shape": "polygon_3d_v2",
     #标签形状
     "label_meta_color": "#f8e71c",
     "level": 0
   },
   {
     "label_meta_id": 1854,
     "label_num": 1,
     "label_meta_name": "car",
     "label_meta_desc": "car0504",
     "label_meta_attr": "{\"优先级\":\"0,1\"}",
     "label_meta_shape": "polygon_3d_v2",
     "label_meta_color": "#7ed321",
     "level": 0
   }
 ],
```



```
"labels": [
#标注物信息
  {
    "label_meta_id": 1854,
    #标注物对应的标签ID
    "create_time": 0,
    "polygon_3d_v2": {
    #该类标注物使用的符号
      "ascii_char": "$"
    },
    "name": "car",
    #标注物名称
    "shape_type": "polygon_3d_v2",
    #标注物形状
    "serial_number": 0,
    "label_object_id": -1,
    "attribute": "",
    #标注物属性
    "label_meta_name": "car",
    "inspection": {
      "miss_label_error": false,
      "vehicle_direction_error": false,
      "attribute_error": false,
      "out_range_label_error": false,
      "anchor_error": false,
      "classification_error": false,
      "correct_label": true,
      "extra_label_error": false
    }
  },
  {
    "label_meta_id": 1853,
    "create_time": 0,
    "polygon_3d_v2": {
      "ascii_char": "#"
    },
    "name": "truck",
    "shape_type": "polygon_3d_v2",
    "serial_number": 0,
    "label_object_id": -1,
    "attribute": "",
    "label_meta_name": "truck",
    "inspection": {
      "miss_label_error": false,
      "vehicle_direction_error": false,
      "attribute_error": false,
      "out_range_label_error": false,
      "anchor_error": false,
      "classification_error": false,
      "correct_label": true,
      "extra_label_error": false
    }
  }
],
"labels_ext": {
  #该帧点云每个点的类别，和labels中polygon_3d_v2->ascii_char对应
  "ascii_string":"    $$$$$          #####"
}
}
```


说明

ascii_string中保存了点云中每个点的类别，上述样例中点云的点数为23，则此字段会有23个字符，每个字符分别代表一个点的类别。字符具体表示的类别可以根据labels- > polygon_3d_v2- > ascii_char和labels- > name找到。

用来表示点类别的ascii码包括：

['!', '"', '#', '$', '%', '&', "'", '(', ')', '*', '+', ',', '-', '.', '/', '0', '1', '2', '3', '4', '5', '6', '7', '8', '9', ':', '。', '<', '=', '>', '?', '@', 'A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M', 'N', 'O', 'P', 'Q', 'R', 'S', 'T', 'U', 'V', 'W', 'X', 'Y', 'Z', '[', '\\', ']', '^', '_', '`', 'a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p', 'q', 'r', 's', 't', 'u', 'v', 'w', 'x', 'y', 'z', '{', '|', '}', '~']，此外空字符（ ' '）表示未标注的类别。

必须字段样例

数据集可视化

3D立方体框类的“.json”文件中必须包含label_counts和labels字段信息。3D语义分割类的“.json”文件中必须包含label_counts，labels和labels_ext字段信息。3D语义分割类的“.json”文件中必须包含label_counts和labels，labels_ext字段有无取决于是否有关系标注信息。

创建标注任务

3D立方体框类的“.json”文件中必须包含label_counts和labels字段信息。3D语义分割类的“.json”文件中必须包含label_counts，labels和labels_ext字段信息。3D地图类的“.json”文件中必须包含label_counts、labels，labels_ext字段有无取决于是否有关系标注信息。

如果使json文件中已有的标注信息在平台上直接展示，label_counts里面的标注物名称、描述、形状、额外属性需要和创建任务使用的平台标签信息保持一致。

3D立方体框示例如下：

```
{"label_counts": [
  #各类标注物的个数统计
      {
        "label_meta_id": 5414,
        #标注物使用的标签ID，
        "label_num": 1,
        #标注物个数
        "label_meta_name": "Car",
        #标签名称
        "label_meta_desc": "自采目标识别",
        #标签描述
        "label_meta_attr": "{\"优先级\":\"0,1\"}",
        #标签额外属性
        "label_meta_shape": "cube_3d",
        #标签形状
         "label_meta_color": "#d0021b"
        #标签颜色
        }
 ],
 "labels": [
 #标注物信息
   {
     "label_meta_id": 5414,
     #标注物对应的标签ID
     "name": "Car",
     #标注物名称
     "shape_type": "cube_3d",
     #标注物形状：3D立体框
     "attribute": "",
     #标注物属性
     "label_meta_name": "Car",
     "serial_number": 0,
```



```
    #该帧中标注物唯一自增id
    "cube_3d": {
      "rotation": {
      #3D框旋转角
        "x": 0.0,
        "y": 0.0,
        "z": 1.53980839
      },
      "serial_number": 2,
      #标注物合成对象的唯一自增id，如果标注物之间没有合成则与serial_number保持一致，追踪任务中同一物体在不同帧中此字段相同
      "location": {
      #3D框中心点坐标
        "x": 0.6671804785728455,
        "y": 15.472203254699707,
        "z": -1.1619998216629028
      },
      "attribute":  "{\"优先级\":\"1\"}",
      "dimensions": {
      #3D框长宽高
        "length": 4.557755470275879,
        "width": 2.0348410606384277,
        "height": 1.4403225183486938
      }
    }
  }
 ]
}
```

3D语义分割示例如下：

```
{
 "label_counts": [
  {
    "label_meta_id": 1853,
    "label_num": 1,
    "label_meta_name": "truck",
    "label_meta_desc": "v0504",
    "label_meta_attr": "{\"优先级\":\"0,1\"}",
    "label_meta_shape": "polygon_3d_v2",
    #标签形状
    "label_meta_color": "#f8e71c"
  },
  {
    "label_meta_id": 1854,
    "label_num": 1,
    "label_meta_name": "car",
    "label_meta_desc": "car0504",
    "label_meta_attr": "{\"优先级\":\"0,1\"}",
    "label_meta_shape": "polygon_3d_v2",
    "label_meta_color": "#7ed321"
  }
 ],

 "labels": [
 #标注物信息
  {
    "label_meta_id": 1854,
    #标注物对应的标签ID
    "polygon_3d_v2": {
    #该类标注物使用的符号
      "ascii_char": "$"
    },
    "name": "car",
    #标注物名称
    "shape_type": "polygon_3d_v2",
    #标注物形状
    "attribute": "",
    #标注物属性
    "label_meta_name": "car"
  },
```



```
  {
    "label_meta_id": 1853,
    "polygon_3d_v2": {
      "ascii_char": "#"
    },
    "name": "truck",
    "shape_type": "polygon_3d_v2",
    "attribute": "",
    "label_meta_name": "truck"
  }
 ],
 "labels_ext": {
 #该帧点云每个点的类别，和labels中polygon_3d_v2->ascii_char对应
  "ascii_string":"   $$$$$          #####"
 }
}
```

2.7.4.3 音频标注数据集文件说明

OCTOPUS 格式文件基本要求（音频标注）

上传的OCTOPUS格式数据集需包含以下文件（以mp3格式为例）。音频文件支持的格式包含：wav、mp3、flac、m4a。

```
.
├─ 文件夹1
    ├─ audio1.mp3        #音频文件
    ├─ audio1.json       #该音频文件的所有标注信息
├─ 文件夹2
    ├─ audio2.mp3        #音频文件
    ├─ audio2.json       #该音频文件的所有标注信息
```

标注数据.json 文件说明

数据集中必含“.json”文件，用于集合该音频文件的所有标注数据信息，包括该音频所在的项目id、数据包id、音频上所有标注信息等。上传数据集前请保证“.json”文件内容正确。“.json”文件编写的参考样例如下：

```
{
 "frame_id": 1,
 #帧序号
 "batch_task_id": 1368,
 #批次任务id
 "project_id": "ee...3d",
 #资源域ID
 "label_mode": "manual",
 #标注类型：auto和manual两种
 "status": "labeled",
 #标注任务状态：unlabeled、labeled、unconfirmed、confirmed、all五种
 "sample_type": "AUDIO",
 #样本类型：包含“IMAGE”,“POINT_CLOUD”,“AUDIO”,“TEXT”
 "des_order": "",
 #此份数据对应的原始数据包描述
 "tag_names": [],
 #标签名称
 "valid": true,
 #是否有效，包含“true”和“false”两种
 "create_time": 1708657733087,
 #标注的创建时间
 "difficult": false,
 #是否难例，包含“true”难例和“false”非难例
 "label_counts": [
 #各类标注物的个数统计
  {
    "label_meta_id": 7900,
    #标注物使用的标签ID
```



```
      "label_num": 1,
      #标注物个数
      "label_meta_name": "奇怪的声音1",
      #标注物名称
      "label_meta_desc": "1233",
      #标注物描述
      "label_meta_attr": "{\"声音1\":\"2333,4444\",\"声音2\":\"334455,1121333\"}",
      #标注物额外属性
      "label_meta_shape": "audio",
      #标注物形状，包含“bndbox、line、circle、polygon、points、dashed、cube_3d、multiBox、
polygon_3d_v2、audio、text”
      "label_meta_color": "#496832",
      #标注物颜色信息
      "level": 0
    },
    {
      "label_meta_id": 7901,
      "label_num": 1,
      "label_meta_name": "噪声",
      "label_meta_desc": "11",
      "label_meta_attr": "{\"text1\":\"111,222,333\",\"额鹅鹅鹅\":\"1111,333\"}",
      "label_meta_shape": "audio",
      "label_meta_color": "#391c1c",
      "level": 0
    }
  ],
  "audio_meta_info": {
   #音频信息
    "id": "49...bd",
    "name": "0000.mp3",
    #音频名称
    "source": "https://octopus-raw-ee.../label-data/task-1368/data/mp3/0000.mp3",
    #音频源的obs路径url
    "duration": "180.0s"
    #音频时长
  },
  "label_task_id": 1691,
   #批次子任务ID
  "partitionId": 20240222,
  "label_update_time": 1708944569975,
   #标注最近更新时间
  "prefix_folder": "mp3",
  "image_id": "88...91",
  "inspection": 0,
  "labels": [
    {
      "label_meta_id": 7900,
       #标注物对应的标签ID
      "create_time": 0,
      "shape_type": "audio",
      #标注物形状
      "serial_number": 1,
      #该帧中标注物唯一自增id
      "label_object_id": -1,
      "attribute": "{\"声音1\":\"2333\"}",
      #标注物属性
      "audio": {
      #音频子段落标注信息
        "xmin": 48.957073,
        #段落开始时间
        "xmax": 80.938614,
        #段落结束时间
        "gender": "MALE",
        #讲话人性别， 允许不存在此字段
        "author": "role1",
        #讲话人角色， 允许不存在此字段
        "text": "aaaaabbb"
        #音频对应的文本
      },
```



```
    "label_meta_name": "奇怪的声音1"
    #标注物名称
   },
   {
    "label_meta_id": 7901,
    "create_time": 0,
    "shape_type": "audio",
    "serial_number": 2,
    "label_object_id": -1,
    "attribute": "{}",
    "audio": {
     "xmin": 126.331764,
     "xmax": 138.0552
    },
    "label_meta_name": "噪声"
   }
  ]
}
```

必须字段样例

数据集可视化

“.json”文件中必须包含label_counts和labels字段信息。

创建标注任务

“.json”文件中必须包含label_counts和labels字段信息。如果需要json文件中已有的标注信息在平台上直接展示，则label_counts里面的标注物名称、描述、形状、额外属性需要和创建任务使用的平台标签信息保持一致。示例如下：

```
{
 "label_counts": [
  #各类标注物的个数统计
  {
   "label_meta_id": 7900,
   #标注物使用的标签ID
   "label_num": 1,
   #标注物个数
   "label_meta_name": "奇怪的声音1",
   #标注物名称
   "label_meta_desc": "1233",
   #标注物描述
   "label_meta_attr": "{\"声音1\":\"2333,4444\",\"声音2\":\"334455,1121333\"}",
   #标注物额外属性
   "label_meta_shape": "audio",
   #标注物形状
   "label_meta_color": "#496832"
   #标注物颜色信息
  }
 ],
 "labels": [
  {
   "label_meta_id": 7900,
   "shape_type": "audio",
   "serial_number": 1,
   "attribute": "{\"声音1\":\"2333\"}",
   "audio": {
    "xmin": 48.957073,
    "xmax": 80.938614,
    "gender": "MALE",
    "author": "role1",
    "text": "aaaaabbb"
   },
   "label_meta_name": "奇怪的声音1"
  }
```



```
 ]
}
```

2.7.4.4 文本标注数据集文件说明

OCTOPUS 格式文件基本要求（文本标注）

上传的OCTOPUS格式数据集需包含以下文件（以txt格式为例）。文本文件支持的格式包含：txt、yaml、xml、csv。

```
.
├─ 文件夹1
   ├─ text1.txt        #文本文件
   ├─ text1.json        #该文本文件的所有标注信息
├─ 文件夹2
   ├─ text2.txt        #文本文件
   ├─ text2.json        #该文本文件的所有标注信息
```

标注数据.json 文件说明

数据集中必含“.json”文件，用于集合该文本文件的所有标注数据信息，包括该文本所在的项目id、数据包id、文本上所有标注信息等。上传数据集前请保证“.json”文件内容正确。“.json”文件编写的参考样例如下：

```
{
  "frame_id": 1,
  #帧序号
  "batch_task_id": 1368,
  #批次任务id
  "project_id": "ee...3d",
  #资源域ID
  "label_mode": "manual",
  #标注类型：auto和manual两种
  "status": "labeled",
  #标注任务状态：unlabeled、labeled、unconfirmed、confirmed、all五种
  "sample_type": "TEXT",
  #样本类型：包含“IMAGE”，“POINT_CLOUD”，“AUDIO”，“TEXT”
  "des_order": "",
  #此份数据对应的原始数据包描述
  "tag_names": [],
  #标签名称
  "valid": true,
  #是否有效，包含“true”和“false”两种
  "create_time": 1708657733087,
  #标注的创建时间
  "difficult": false,
  #是否难例，包含“true”难例和“false”非难例
  "label_counts": [
  #各类标注物的个数统计
    {
      "label_meta_id": 7900,
      #标注物使用的标签ID
      "label_num": 1,
      #标注物个数
      "label_meta_name": "人物",
      #标注物名称
      "label_meta_desc": "1233",
      #标注物描述
      "label_meta_attr": "{\"男\":\"少年,青年\",\"女\":\"少年,青年\"}",
      #标注物额外属性
      "label_meta_shape": "text",
      #标注物形状，包含“bndbox、line、circle、polygon、points、dashed、cube_3d、multiBox、
polygon_3d_v2、audio、text”
      "label_meta_color": "#496832",
      #标注物颜色信息
      "level": 0
```



```
  },
  {
    "label_meta_id": 7901,
    "label_num": 1,
    "label_meta_name": "水果",
    "label_meta_desc": "11",
    "label_meta_attr": "{\"颜色\":\红,黄\"}",
    "label_meta_shape": "text",
    "label_meta_color": "#391c1c",
    "level": 0
  }
 ],
 "text_meta_info": {
  #文本信息
   "id": "49...bd",
   "name": "0000.txt",
   #文本名称
   "source": "https://octopus-raw-ee.../label-data/task-1368/data/txt/0000.txt"
   #音频源的obs路径url
 },
 "label_task_id": 1691,
  #批次子任务ID
 "partitionId": 20240222,
 "label_update_time": 1708944569975,
  #标注最近更新时间
 "prefix_folder": "txt",
 "image_id": "88...91",
 "inspection": 0,
 "labels": [
   {
     "label_meta_id": 7900,
      #标注物对应的标签ID
     "create_time": 0,
     "shape_type": "text",
     #标注物形状
     "serial_number": 1,
     #该帧中标注物唯一自增id
     "label_object_id": -1,
     "attribute": "{\"人物\":\"男\"}",
     #标注物属性
     "text": {
     #文本标注信息
       "start_idx": 1,
       #标注起始偏移量
       "end_idx": 3
       #标注结束偏移量
     },
     "label_meta_name": "人物"
     #标注物名称
   }
 ]
}
```

必须字段样例

数据集可视化

“.json”文件中必须包含label_counts和labels字段信息。

创建标注任务

“.json”文件中必须包含label_counts和labels字段信息。如果需要json文件中已有的标注信息在平台上直接展示，则label_counts里面的标注物名称、描述、形状、额外属性需要和创建任务使用的平台标签信息保持一致。示例如下：

```
{
 "label_counts": [
```



```
 #各类标注物的个数统计
  {
    "label_meta_id": 7900,
    #标注物使用的标签ID
    "label_num": 1,
    #标注物个数
    "label_meta_name": "人物",
    #标注物名称
    "label_meta_desc": "1233",
    #标注物描述
    "label_meta_attr": "{\"男\":\"少年,青年\",\"女\":\"少年,青年\"}",
    #标注物额外属性
    "label_meta_shape": "text",
    #标注物形状，包含“bndbox、line、circle、polygon、points、dashed、cube_3d、multiBox、
polygon_3d_v2、audio、text”
    "label_meta_color": "#496832",
    #标注物颜色信息
    "level": 0
  },
  {
    "label_meta_id": 7901,
    "label_num": 1,
    "label_meta_name": "水果",
    "label_meta_desc": "11",
    "label_meta_attr": "{\"颜色\":\红,黄\"}",
    "label_meta_shape": "text",
    "label_meta_color": "#391c1c",
    "level": 0
  }
 ],
 "labels": [
  {
    "label_meta_id": 7900,
     #标注物对应的标签ID
    "create_time": 0,
    "shape_type": "text",
    #标注物形状
    "serial_number": 1,
    #该帧中标注物唯一自增id
    "label_object_id": -1,
    "attribute": "{\"人物\":\"男\"}",
    #标注物属性
    "text": {
    #文本标注信息
      "start_idx": 1,
      #标注起始偏移量
      "end_idx": 3
      #标注结束偏移量
    },
    "label_meta_name": "人物"
    #标注物名称
  }
 ]

}
```

2.8 数据缓存

用户购买自动驾驶数据管理缓存扩容包后，可使用数据缓存功能提供专用高速文件存储，加速训练和评测读取数据集的速度。需确保待缓存的数据集中的所有数据均已发布成功。

步骤1 确认待缓存数据集中是否存在未发布的数据。

在数据集列表（数据资产 > 数据集）中，单击数据集名称，查看“数据详情”页签的“待发布区”是否存在未发布的数据：

- 是：参考**步骤4**发布数据，然后执行**步骤2**。
- 否：执行**步骤2**。

步骤2 在数据集列表（数据资产 > 数据集）中，单击操作栏中的“缓存加速”。

步骤3 单击“数据缓存”，在数据缓存界面查看缓存的数据。

页面展示缓存的数据集名称，缓存状态，来源类型，数据大小，创建时间和更新时间。需要删除不需要的缓存数据时，可单击“删除”，删除对应的缓存数据。

缓存状态如下：

- 缓存中：数据集数据正在缓存中。
- 缓存成功：数据集缓存成功。
- 缓存过期：数据集缓存成功后，数据发生变更后状态更新为缓存过期，此时需执行**步骤2**更新缓存。
- 缓存失败：数据集缓存失败，可重复执行**步骤2**至数据缓存成功。

----结束

2.9 模型管理

2.9.1 模型仓库

模型仓库介绍

模型管理模块支持以模型仓库的方式对模型进行纳管和归档，每个模型仓库中又可以创建若干个模型版本，方便对同类模型统一管理。

模型包含镜像和文件两部分。其中镜像与模型仓库绑定，也就是说同一个模型仓库的版本共享一个镜像；文件存储在对象存储（OBS）中， 各个版本的文件一般不同。

模型可用于预标注、预审核、训练、评测、编译、推理等服务，下图介绍3种常见作业流程：

图 2-5 新建**空模型仓库**归档**训练作业产物**

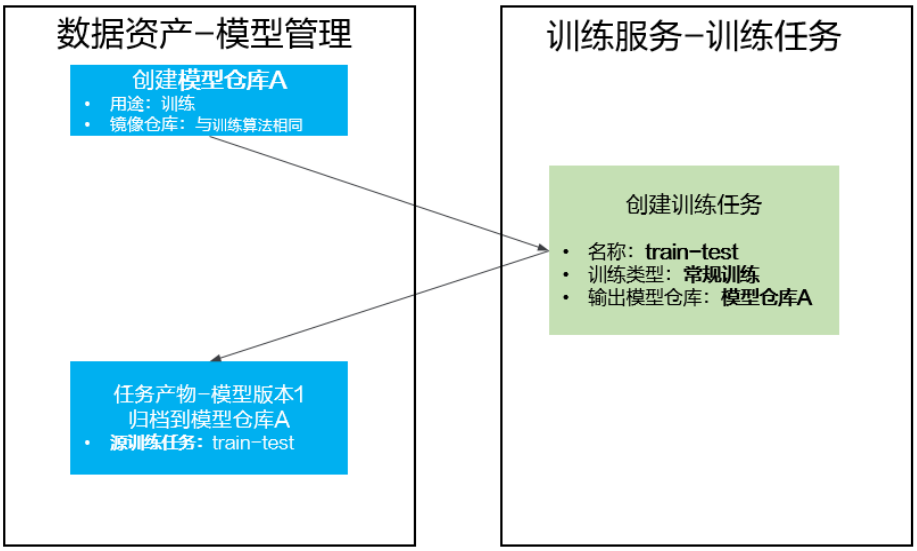


图 2-6 预训练模型上云进行增量训练

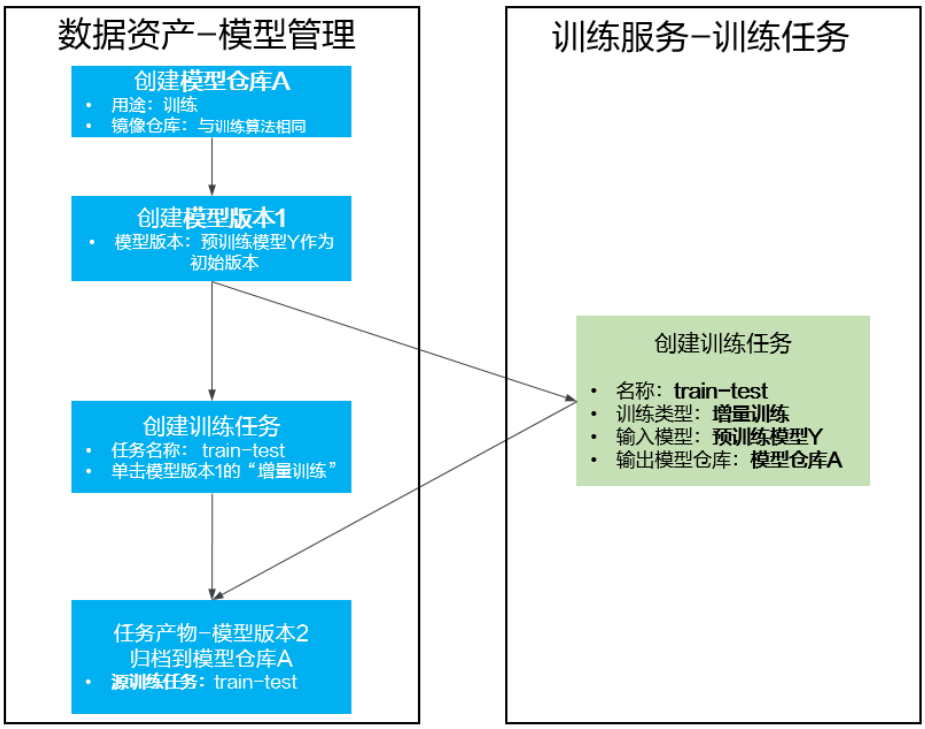
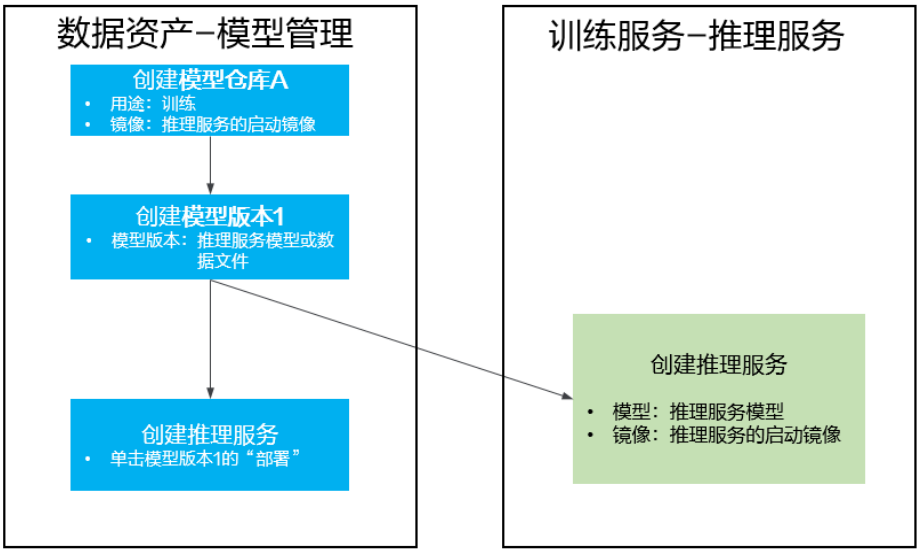


图 2-7 利用云上算力部署模型为推理服务



新建模型仓库

步骤1 登录Octopus服务平台，在左侧菜单栏中选择“数据资产 > 模型管理”。

步骤2 单击“新建模型仓库”，填写模型仓库基本信息。

表 2-37 新建模型仓库

参数	说明
名称	模型仓库名称。不能包含“@^\#$%&*<>'|"/`”，不得超过64个字符。
描述	简要描述模型，不能包含“@^\#$%&*<>'|"/`”，不得超过256个字符。
用途	选择模型仓库用途。 • 训练 • 标注
样本类型	选择样本类型。 • 用途为“训练”时：图片、3D点云 • 用途为“标注”时：图片、3D点云、音频、文本
标注	从下拉列表中选择标注（需提前创建）。
镜像用途	选择镜像用途。 • 用途为“训练”时：训练/评测 • 用途为“标注”时：预标注、预审核
镜像仓库	选择镜像仓库以及版本。
共享级别	选择镜像仓库共享级别。 • 个人：当前操作用户。 • 团队：当前工作空间下被授权的用户可以使用该模型创建评测、编译任务、预标注任务、预审核任务、推理服务和开发环境。不能使用团队模型仓库创建训练任务。

步骤3 单击“创建”，完成模型仓库的创建。

步骤4 查看模型仓库详情。

单击模型仓库名称，进入该模型仓库的详情页面，分为模型仓库详情和版本管理两部分内容。

- 模型仓库详情：

  展示模型ID、名称、描述、用途、来源、样本类型、创建时间、标注等信息。

- 版本管理：

  当仓库用途为训练或标注时，展示该模型所有可用版本的信息，如版本名称、在线编辑、状态、打包状态、创建时间等信息，用户可以在线编辑模型版本、新建模型版本和管理模型版本，详见**在线编辑模型**、**新建版本**、**模型版本管理**。

  说明

  若模型仓库中的版本被推理服务、评测任务等流程使用时则不可以对模型仓库除“描述”外的字段进行编辑。

----结束

上传模型文件新建版本

Octopus支持用户对已有的模型进行版本更新，步骤如下：

须知

数据上传中刷新或关闭浏览器会导致文件上传异常，请谨慎操作！

步骤1 准备模型文件，模型文件具体要求请参见**模型文件说明**。

步骤2 单击模型仓库名称进入详情界面，单击“新建版本”或者在模型列表页单击“新建版本”。

步骤3 根据不同数据来源创建模型版本。

- 数据来源选择“本地”（模型文件存放在本地）。
  a. 根据需要输入模型版本描述，选择关联已有算法。
  b. 在“模型版本”单击添加按钮，选择本地数据。或直接将本地模型文件夹拖入框内。
  c. 单击“上传”，等待数据上传成功。
- 数据来源选择“OBS”（模型文件存放在OBS指定路径）。
  a. 根据需要输入模型版本描述，选择关联已有算法。
  b. 输入访问密钥、私有访问密钥和OBS地址，勾选服务声明信息。
  c. 单击“上传”，等待数据上传成功。

----结束

模型版本管理

在模型仓库详情页还可以完成以下操作。

表 2-38 训练或标注模型版本管理相关操作

操作	操作步骤
增量训练	模型仓库的用途为“训练”，且模型版本状态为“创建成功”的模型版本支持增量训练。 单击操作栏“增量训练”，即可跳转到新建训练任务界面，进行增量训练。
部署模型版本	单击操作栏“部署”，打开部署弹出框，默认选中当前模型版本，部署推理服务。部署任务下发成功后，可以到“训练服务”>“推理服务”，查看部署状态。了解部署推理服务参数信息请查看**新建推理服务**。 **说明** 支持部署的模型要求： • 模型仓库用途为训练 • 模型版本状态为创建成功

操作	操作步骤
导出	选择“操作”列的“更多 > 导出”将指定模型版本导出到OBS指定路径，不支持导出来源为内置的模型版本。
打包	单击操作栏“打包”，提示“模型版本打包中”，待模型打包成功后，当打包状态变为“打包成功”。
下载	当打包状态为“打包成功”时，单击操作栏“下载”，即可将模型下载到本地。
上传	当状态为“初始化”时，可单击版本名称后操作栏内的“上传文件”，选择本地文件上传。
断点续传	当状态为“上传中断”时，可单击版本名称后操作栏内的“断点续传”，选择本地文件重新上传。
删除	单击版本名称后操作栏内的“删除”，删除该版本。版本删除后无法恢复，请谨慎操作。
日志	当数据来源选择“OBS”时或模型版本执行OBS导出时可查看日志。 单击版本名称前的展开图标，在任务的“操作”列单击“日志”，界面跳转至日志列表，在日志列表可查看或者下载日志。
停止	当数据来源选择“OBS”，且任务状态为“导入中”时可手动停止任务。 单击版本名称前的展开图标，在OBS导入任务的“操作”列单击“停止”，任务更新为已停止，版本状态更新为创建失败。 当模型版本执行OBS导出，且任务状态为“导出中”时可手动停止任务。 单击版本名称前的展开图标，在OBS导出任务的“操作”列单击“停止”，任务更新为已停止。

表 2-39 智驾模型版本管理相关操作

操作	操作步骤
模型微调	当模型状态为“创建成功”，可单击操作栏“模型微调”，对模型进行再次调优操作。
删除模型版本	单击版本名称后操作栏内的“删除”，删除该版本。版本删除后无法恢复，请谨慎操作。

查询模型仓库

可按照“来源”、“样本类型”、“用途”、“共享级别”、“镜像仓库ID”和“名称”过滤模型 ，单击搜索，进行查询查找目标模型。

在线编辑模型

平台提供模型编辑器，在模型详情页单击“在线编辑”，进入该模型的在线编辑页面。界面左侧显示的是该模型包内的所有文件，以目录树的形式展示，支持编程语言的渲染，支持MarkDown文件的实时双屏预览。

- 新建文件夹：选中文件夹并单击 ，用户将新建一个该文件夹的子文件夹。选中工程文件并单击 ，将会新建一个新的文件夹，与用户已有的文件夹同级。
- 新建文件：单击，用户可新建文件。
- 修改文件：单击，用户可对文件名称进行修改。
- 下载文件：单击，用户可下载文件。
- 删除文件：单击，用户可删除文件。

  说明

  - 文件（夹）名称不能为空，且只能包含数字、英文、中文、点、下划线和中划线。
  - 删除后不可恢复，请谨慎操作。
- 配置界面：单击 ，按照喜好配置界面基本属性，查看快捷键说明。
- 删除当前模型文件：单击“删除”，删除当前页面的模型文件。删除后不可恢复，请谨慎操作。
- 保存模型文件：单击“保存”，保存当前模型。模型更新完毕请及时保存。

2.9.2 模型文件说明（标注）

模型文件可用于预标注任务、预审核任务。不同任务所需的模型文件要求不同。

2.9.2.1 预标注模型文件

预标注模型完成对数据的推理，并将推理结果按照规定格式放在指定路径中。

模型文件基本要求

自定义模型包通过环境变量获取数据集路径和推理结果存放路径，将每帧数据的推理结果按照规定格式存入规定路径的json文件中。

自定义模型包中必须包含启动文件。除此之外，还可包含一些其他必要的自定义模型文件、自定义脚本、自定义库等。

```
├─ model                  # 模型包根目录，上传模型时进入该目录下（名称可自定义）
    ├─ customer_inference.py     #“模型推理”启动文件（名称可自定义）
    ├─ customer_package          #自定义库（可选，名称可自定义）
        ├─ __init__.py           #自定义库 - >  python库
        ├─ tensor_define.py      #自定义库 - >  python库文件
        └─ ...
    ├─ customer_utils.py         #自定义脚本1（可选，名称可自定义）
    ├─ customer_script2.sh       #自定义脚本2（可选，名称可自定义）
    ├─ sub_directory             #子文件夹
        └─ customer_model.pb     #自定义模型文件（必选，名称可自定义）
    └─ ...                       #其他（可选）
```

- 启动文件

整个自定义模型包的入口文件，该文件在启动容器时被运行。创建预标注任务时，需在“标注脚本”处提供该文件的绝对路径。文件绝对路径为模型仓库中的模型在镜像仓库的镜像中的路径(“/tmp/label/source/model/”)+文件在模型中的相对路径，以上述为例，路径为“/tmp/label/source/model/customer_inference.py”。

- 自定义库

  允许用户使用自定义库，但不推荐使用需要编译的库，以避免与内置库文件冲突。示例中使用Python语言中的package作为自定义库。

- 自定义脚本

  允许自定义除启动文件以外的自定义脚本文件，可根据实际所需编写。

- 自定义模型文件

  自定义模型文件“xxx.pb”，需要通过编写自定义脚本加载并使用。自定义模型文件的存放位置及名称可自定义，可以将相关模型文件保存至子文件夹中，也可以保存至根目录下。

环境变量使用说明

模型推理所需的待标注数据集目录、预标注结果数据目录、标注物文件目录、模型文件目录、预标注日志文件目录均可通过注入镜像的环境变量获取，详情见**镜像制作（标注）**。

预标注结果格式说明

推理完毕后，需要按照规定格式组织预标注结果，并保存在特定路径下的json文件中，路径要求见“模型文件基本要求”。

Json文件内容组织结构如下所示，labels字段中保存了每个预测对象的基本信息。

```
{
    "labels":[]
}
```

其中规模3D大规模点云分割任务还包含“label_ext”字段，具体参考**•3D大规模点云分割:**。

```
{
    "labels":[],
    "labels_ext":{}
}
```

不同类型的任务对象基本信息所需格式不同，具体如下所示：

- **2D目标检测**

  labels中保存每个预测对象的基本信息，每个基本信息格式如下所示。

```
{
'label_meta_id': 168,       #标注对象对应的平台标注物的ID，可从标注物文件（OCTPS_META_PATH）中获取（即id字段）
'bndbox': {                 #矩形框的位置信息
        'xmin': 235,
        'ymin': 123,
        'xmax': 456,
        'ymax': 360
        },
'shape_type': "bndbox",       #标注物的形状，此处表示2D矩形框
'serial_number':5           #当前对象在当前帧的唯一ID， 从1递增
"label_object_id": 5,       #标注物合成对象的唯一自增id,如果标注物之间没有合成则与serial_number保持一致
```

```
"label_meta_name": "Car"       #标注物名称
}
```

- **2D语义分割**

  Labels中保存每个预测对象的基本信息，每个基本信息格式如下所示。

```
{
'label_meta_id': 167,                #标注对象对应的平台标注物的ID，可从标注物文件（OCTPS_META_PATH）中获取（即id字段）
'shape_type': 'polygon',              # 标注物的形状，此处表示多边形
'serial_number': 2,                  #当前对象在当前帧的唯一ID，从1递增
      'polygon':{                    #多边形的具体信息
            'size': 3,               # 多边形点数
            'points':[               # 每个点的位置信息
                  {'xpoint': 123, 'ypoint': 456},
                  {'xpoint': 135, 'ypoint': 467},
                  {'xpoint': 123, 'ypoint': 456}
                  ]
            }
"label_object_id": 2,                #标注物合成对象的唯一自增id,如果标注物之间没有合成则与serial_number保持一致
"label_meta_name": "Car"             #标注物名称
}
```

- **2D车道线**

  Labels中保存每个预测对象的基本信息，每个基本信息格式如下所示。

```
{
'label_meta_id': 288,                #标注对象对应的平台标注物的ID，可从标注物文件（OCTPS_META_PATH）中获取（即id字段）
      'shape_type': 'line',          #标注物的形状，此处表示折线
'serial_number': 35,                 #当前对象在当前帧的唯一ID，从1递增
      'line':{                       #折线的具体信息
            'size': 3,               #折线点数
            'points':[               #每个点的位置信息
                  {'xpoint': 123, 'ypoint': 456},
                  {'xpoint': 135, 'ypoint': 467},
                  {'xpoint': 123, 'ypoint': 456}
                  ]
            }
"label_object_id": 35,               #标注物合成对象的唯一自增id,如果标注物之间没有合成则与serial_number保持一致
"label_meta_name": "Car"             #标注物名称
}
```

- **3D目标检测**

  Labels中保存每个预测对象的基本信息，每个基本信息格式如下所示。

```
{
 'label_meta_id': 174,               #标注对象对应的平台标注物的ID，可从标注物文件（OCTPS_META_PATH）中获取（即id字段）
    'cube_3d': {                     # 3D框的具体信息
      'dimensions': {                #3D框的长宽高
        'width': 1.839784026145935,
        'length': 4.315396785736084,
        'height': 1.55556058883667
      },
      'location': {                  #3D框的位置
        'x': -4.736311912536621,
        'y': -71.40546417236328,
        'z': -1.356909990310669
      },
      'serial_number': 3,            #当前对象在当前帧的唯一ID，从1递增
           'rotation': {"x":0,"y":0,"z":anger},       #3D框的朝向信息
    },
    'shape_type': 'cube_3d',         #标注物形状，此处表示3D框
"label_object_id": 3,                #标注物合成对象的唯一自增id,如果标注物之间没有合成则与serial_number保持一致
"label_meta_name": "Car"             #标注物名称
  }
```


- **3D大规模点云分割**

  Labels中保存每个预测类别的基本信息，每个基本信息格式如下所示。

```
{
"label_meta_id": 1423,                    #标注对象对应的平台标注物的ID，可从标注物文件（OCTPS_META_PATH）中获取（即id字段）
 "polygon_3d_v2": {                       #类别的基础信息
"ascii_char": "\""                        #当前类别对应的ascii码
 },
 "shape_type": "polygon_3d_v2",           #标签的形状，此处表示3D语义分割对应的形状
 "name": "BEV土墙"                        #标注物名称
}
```

  此外，3D大规模点云分割的结果文件中的labels_ext字段示例如下：

```
"labels_ext": {
 "ascii_string":
"#################################################################\\#######"
}
```

  ascii_string中保存了点云中每个点的类别，如果此帧点云的点数为1000，则此字段会有1000个字符，每个字符分别代表一个点的类别。字符具体表示的类别可以根据labels- > polygon_3d_v2- > ascii_char和labels- > name找到。

  用来表示点类别的ascii码包括：

  ['!', '"', '#', '$', '%', '&', "'", '(', ')', '*', '+', ',', '-', '.', '/', '0', '1', '2', '3', '4', '5', '6', '7', '8', '9', ':', '。', '<', '=', '>', '?', '@', 'A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M', 'N', 'O', 'P', 'Q', 'R', 'S', 'T', 'U', 'V', 'W', 'X', 'Y', 'Z', '[', '\\', ']', '^', '_', '`', 'a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p', 'q', 'r', 's', 't', 'u', 'v', 'w', 'x', 'y', 'z', '{', '|', '}', '~']，此外空字符（' '）表示未标注的类别。

2.9.2.2 预审核模型文件

预审核模型完成对已标数据的审核，并将审核结果和审核所用的审核规则按照规定格式放在指定路径中。

模型文件基本要求

自定义模型包通过环境变量获取数据集路径和推理结果存放路径，将每帧数据的审核结果按照规定格式存入规定路径的json文件中。

自定义模型包中必须包含启动文件。除此之外，还可包含一些其他必要的自定义模型文件、自定义脚本、自定义库等。

```
├─ model                  # 模型包根目录，上传模型时进入该目录下（名称可自定义）
    ├─ customer_inspection.py     #“模型推理”启动文件（名称可自定义）
    ├─ customer_package           #自定义库（可选，名称可自定义）
       ├─ __init__.py             #自定义库 - >  python库
       ├─ tensor_define.py        #自定义库 - >  python库文件
       └─ ...
    ├─ customer_utils.py          #自定义脚本1（可选，名称可自定义）
    ├─ customer_script2.sh        #自定义脚本2（可选，名称可自定义）
    ├─ sub_directory              #子文件夹
       └─ customer_model.pb       #自定义模型文件（必选，名称可自定义）
    └─ ...                        #其他（可选）
```

- 启动文件

  整个自定义模型包的入口文件，该文件在启动容器时被运行。创建预审核任务时，需在“标注脚本”处提供该文件的绝对路径。文件绝对路径为模型仓库中的模型在镜像仓库的镜像中的路径(“/tmp/label/source/model/”)+文件在模型中的相对路径，以上述为例，路径为“/tmp/label/source/model/customer_inspection.py”。

- 自定义库

  允许用户使用自定义库，但不推荐使用需要编译的库，以避免与内置库文件冲突。示例中使用Python语言中的package作为自定义库。

- 自定义脚本

  允许自定义除启动文件以外的自定义脚本文件，可根据实际所需编写。

- 自定义模型文件

  自定义模型文件“xxx.pb”，需要通过编写自定义脚本加载并使用。自定义模型文件的存放位置及名称可自定义，可以将相关模型文件保存至子文件夹中，也可以保存至根目录下。

环境变量使用说明

模型推理所需的待审核数据集目录等信息均可通过注入镜像的环境变量获取，详情见**镜像制作（标注）**。

环境变量	描述	获取方式(以python为例)
OCTPS_DATASET_DIR	全量数据集目录	os.getenv('OCTPS_DATASET_DIR')
OCTPS_META_PATH	模型版本关联标注物文件目录	os.getenv('OCTPS_META_PATH')
OCTPS_INSPECTION_ATTRI_DIR	审核属性字段目录	os.getenv('OCTPS_INSPECTION_ATTRI_DIR')
OCTPS_DATASET_INDEX_PATH	待审核的数据帧索引文件目录（用于从全量数据集中筛选出需要审核的数据）	os.getenv('OCTPS_DATASET_INDEX_PATH')
TARGET_RESULT_DIR	预审核结果数据目录	os.getenv('TARGET_RESULT_DIR')
TARGET_RULES_DIR	预审核规则数据目录	os.getenv(' TARGET_RULES_DIR ')
TARGET_LOG_DIR	预审核日志文件目录	os.getenv('TARGET_LOG_DIR')

预审核结果格式说明

审核完毕后，需要按照规定格式组织预标注结果，并保存在特定路径（TARGET_RESULT_DIR）下的json文件中。路径要求见**镜像制作（标注）**。

Json文件内容组织结构如下所示，labels字段中保存每个对象的标注信息、审核模型预测信息（predict_infos）和审核结果信息（inspection）。

```
{
  "labels":[{
    #1. 此对象的标注信息 （直接从源数据labels.json中获取），如果未标注出此对象，则无此部分信息
      ... ...
    #2.此对象的模型预测信息 ，如果模型未预测出此对象，则无此部分信息
    "predict_infos": {
     #形状坐标信息
```

```
      #对象类别名称
      #额外属性信息
    }
    #3.审核结果，如果未审核此对象，则无此部分信息
    "inspection": {
    #字段名称取自OCTPS_INSPECTION_ATTRI_DIR文件
     }
  },
     ... ...
]
}
```

其中3D大规模点云分割任务还包含“labels_ext”和“predict_labels_ext”字段，具体参考**•3D大规模点云分割:**。

```
{
       "labels":[],
       "labels_ext":{ }
       "predict_labels":[]
}
```

以2D目标检测为例，完整json结果文件样例如下：

```
{
  "labels": [
  {
     #1. 此对象的标注信息（直接从源数据labels.json中获取）
    "label_meta_id": 1846,
    "bndbox": {
      "ymin": 545.4334,
      "xmin": 1158.3188,
      "ymax": 705.71844,
      "xmax": 1436.3274
    },
    "name": "框0504",
    "shape_type": "bndbox",
    "serial_number": 2,
    "label_object_id": 2,
    "attribute": "{\"优先级\":\"1\"}",
    "label_meta_name": "框0504",
     #2.此对象的模型预测信息
    "predict_infos": {
      "bndbox": {
        "ymin": 545.4334,
        "xmin": 1158.3188,
        "ymax": 725.71844,
        "xmax": 1456.3274
      },
      "label_meta_name": "框0504",
      "attribute": "{\"优先级\":\"1\"}"
    },
    #3.审核结果
    "inspection": {
      #字段名称取自OCTPS_INSPECTION_ATTRI_DIR文件
      "miss_label_error": false,
      "vehicle_direction_error": false,
      "error_desc": "无效",
      "attribute_error": true,
      "out_range_label_error": true,
      "anchor_error": false,
      "classification_error": false,
      "extra_label_error": false
    }
   }
  ]
}
```

不同类型的标注对象形状基本信息所需格式不同。下面为各类标注对象predict_infos的字段说明：

- **2D目标检测**


```
{"predict_infos": {
        "bndbox": {
          "ymin": 545.4334,
          "xmin": 1158.3188,
          "ymax": 725.71844,
          "xmax": 1456.3274
        },
        "label_meta_name": "框0504",
        "attribute": "{\"优先级\":\"1\"}"
      }
}
```


- **2D语义分割**


```
{"predict_infos": {
        "polygon": {
          "size": 3,
          "points": [
            {
              "xpoint": 135.03,
              "ypoint": 482.94937
            },
            {
              "xpoint": 84.318344,
              "ypoint": 554.4891
            },
            {
              "xpoint": 135.03,
              "ypoint": 482.94937
            }
          ]
        },
        "label_meta_name": "多边形0504",
        "attribute": "{\"优先级\":\"1\"}"
      }
}
```


- **2D车道线**


```
{"predict_infos": {
        "line": {
          "size": 3,
          "points": [
            {
              "xpoint": 901.138,
              "ypoint": 553.583
            },
            {
              "xpoint": 741.36,
              "ypoint": 630.367
            },
            {
              "xpoint": 618.153,
              "ypoint": 681.566
            }
          ]
        },
        "label_meta_name": "线0504",
        "attribute": "{\"优先级\":\"1\"}"
      }
}
```


- **3D目标检测**


```
{
  "predict_infos": {
    "label_meta_name": "Car",
    "cube_3d": {
      "rotation": {
        "x": 0.0,
        "y": 0.0,
        "z": 0.08726646
```



```
    },
    "location": {
      "x": -40.23651584555386,
      "y": 1.2362389665094042,
      "z": -0.8413386615781039
    },
    "attribute": "{}",
    "dimensions": {
      "length": 4.459540762142082,
      "width": 1.4870339632034302,
      "height": 1.4895729290943762
    }
   }
  }
}
```

- **3D大规模点云分割**

```
{"predict_infos": {
        "polygon_3d_v2": {
          "ascii_char": "2"
        },
        "name": "car",
    }
}
```

- **3D大规模点云分割完整样例**

```
{
  "labels": [
    {
      "label_meta_id": 4867,
      "create_time": 0,
      "polygon_3d_v2": {
        "ascii_char": "3"
      },
      "name": "car",
      "shape_type": "polygon_3d_v2",
      "serial_number": 0,
      "label_object_id": -1,
      "attribute": "",
      "label_meta_name": "car",
      "inspection": {
        "miss_label_error": false,
        "vehicle_direction_error": false,
        "error_desc": "",
        "attribute_error": false,
        "out_range_label_error": false,
        "anchor_error": false,
        "classification_error": false,
        "extra_label_error": false
      },
      "predict_infos": {
        "polygon_3d_v2": {
          "ascii_char": "2"
        },
        "name": "car"
      }
    },
    {
      "predict_infos": {
        "polygon_3d_v2": {
          "ascii_char": "4"
        },
        "name": "van"
      }
    }
  ],
  "labels_ext": {
    "ascii_string": "3333333333          3333333333"
  },
  "predict_labels_ext": {
```



```
    "ascii_string": "222222222244444     2222222222"
  }
}
```

labels_ext中保存点云中每个点的标注类别，具体内容说明参考**OCTOPUS数据集格式说明**。predict_labels_ext中保存点云中每个点的模型预测类别。

3D语义分割审核结果可视化说明：针对有审核属性错误的标注对象，展示该标注对象对应位置点的预测类别。

预审核规则格式说明

预审核模型使用的审核规则以字典的格式保存在特定目录下的json文件中（TARGET_RULES_DIR），以便在审核报告中展示。

```
{
"rules": [
        {
           "rule": "",  #规则名称
           "description": "", #规则具体要求
           "inspection_attribute": "" #此规则对应的审核属性,2D包含：多标，漏标，类型错误，未贴合，属性错误。3D包含：多标，漏标，类型错误，未贴合，属性错误，车头方向错误，锚点错误。
        }
... ...
]
}
```

示例：

```
{
    "rules": [
      {
         "rule": "标注框贴合精度",
         "description": "标注框与实际对象误差不超过5个像素",
         "inspection_attribute": "未贴合"
      },
      {
         "rule": "标注框类别",
         "description": "类别标注错误",
         "inspection_attribute": "类型错误"
      }
    ]
  }
```

2.9.3 模型文件说明（训练）

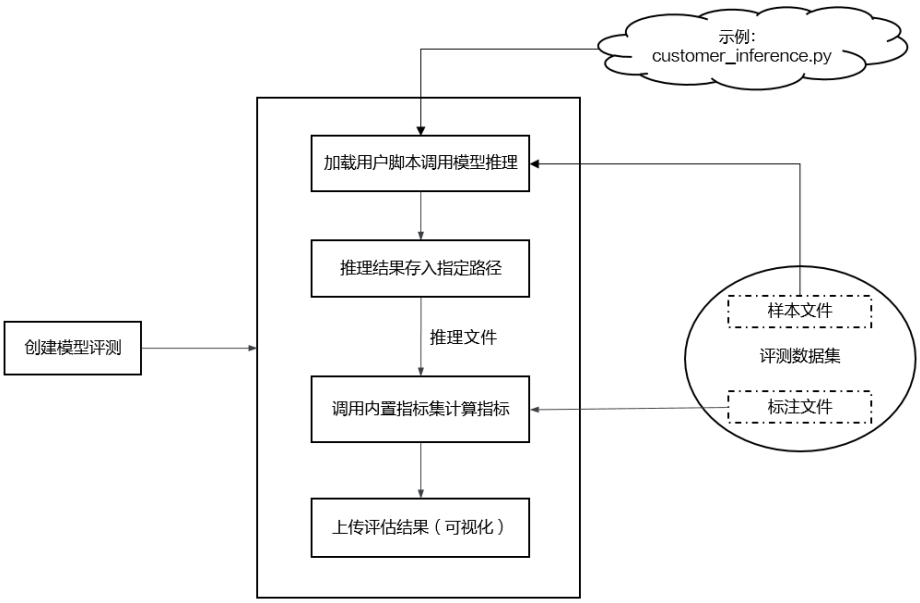
Octopus模型管理模块，支持用户上传模型，并将其用于模型评测、模型编译任务。如果需要将模型用于内置评测模板评测，除模型文件外，需另外包含推理启动文件：

- **customer_inference.py**

  仅当需要使用内置评测指标计算时需要添加推理启动文件，文件名称可自定义，将该文件置于模型目录下。

  为了显示以上文件的使用流程，可参考如下图示：

图 2-8 使用流程



模型文件基本要求

模型文件通常包括模型图和模型权重文件，具体内容视用户算法决定，无其他要求。如果需要使用内置指标计算，则需满足以下条件：

自定义模型包中必须包含推理启动文件以及自定义模型文件，其中推理启动文件需要根据Octopus数据格式对推理结果存储为json进行适配，除此之外还可以包含一些其他必要的自定义脚本、自定义库、自定义算子库等。模型包文件目录结构可参考如下：

```
.
├─ model                          #模型包根目录，上传模型时进入该目录下（名称可自定义）
    ├─ customer_inference.py     #“模型推理”启动文件（如该模型用于内置评测模板评测，需要提供该文件，名称可自定义）
    ├─ customer_package          #自定义库（可选，名称可自定义）
       ├─ __init__.py            #自定义库 - >  python库
       ├─ tensor_define.py       #自定义库 - >  python库文件
       └─ ...
    ├─ customer_utils.py         #自定义脚本1（可选，名称可自定义）
    ├─ customer_script2.sh       #自定义脚本2（可选，名称可自定义）
    ├─ sub_directory             #子文件夹
       ├─ customer_operation.so  #自定义算子库（可选，名称可自定义）
       └─ coustomer_model.pb     #自定义模型文件（必选，名称可自定义）
    └─ ...                       #其他（可选）
```

- 启动文件。

  模型推理脚本，用户接收数据集路径和推理结果存放路径，按照一定要求将每张图片的推理结果存入对应路径json文件中。

- 自定义库。

  允许用户使用自定义库，但不推荐使用需要编译的库，以避免与内置库文件冲突。示例中使用Python语言中的package作为自定义库。

- 自定义脚本。

  允许自定义除启动文件以外的自定义脚本文件，可根据实际所需编写。

- 自定义模型文件。

自定义模型文件“xxx.pb”，需要通过编写自定义脚本加载并使用。自定义模型文件的存放位置及名称可自定义，可以将相关模型文件保存至子文件夹中，也可以保存至根目录下。

2.10 通用存储

2.10.1 通用存储

创建存储目录

步骤1 在左侧菜单栏中，选择“数据资产 > 通用存储”。

步骤2 在“通用存储”页签，单击“创建存储目录”，填写名称和描述信息。

步骤3 单击“确认”。

----结束

通用存储相关操作

在“通用存储”列表，还可以完成以下操作。

表 2-40 通用存储相关操作

任务	操作步骤
查看通用存储	单击通用存储名称，可查看通用存储详情。具体可查看**通用存储详情**。
查询通用存储	在搜索框中输入搜索条件，按回车键即可查询。
删除通用存储	• 选择单个通用存储，单击操作栏的“ 删除”，删除单个通用存储。 • 勾选多个通用存储，单击列表上方的“删除”，可删除多个通用存储。 **说明** 当通用存储中存在数据处理任务时，不可删除通用存储，需先删除完数据处理任务才可删除通用存储。
编辑通用存储	单击操作栏内的“编辑”，可编辑通用存储。

通用存储详情

步骤1 在左侧菜单栏中，选择“数据资产 > 通用存储”。

步骤2 在“通用存储”页签，单击存储目录名称，进入通用存储详情页。

- 存储数据：显示在此通用存储中所有的数据。

  支持按照数据索引和目录树两种方式显示数据。目录树页面仅支持查看数据存储关系和执行导出数据操作，数据索引页面支持如下完整操作：

表 2-41 存储数据相关操作

任务	操作步骤
查看文件详情	单击“文件名称”，显示对应文件的基本信息和文件预览。可编辑文件的“标签”和“自定义属性”。 **说明** – “标签”和“自定义属性”可经过数据处理产生，此外“标签”可在数据服务**标签管理**处创建，“自定义属性”也是在通用存储的**自定义属性**中创建。 – 文件预览支持图片类型和文本类型的数据的显示；支持点云类型的数据的显示、拖拽、三维旋转、缩放等效果；支持视频、音频类型的数据的播放、暂停、跳转、倍速调整、静音。 – 当前平台只支持h264编码的MP4可视化播放，非h264格式可视化播放不支持，只支持播放音频。
查询数据	可选择属性筛选，或输入关键字搜索数据。
导出数据	可选择数据导出至本地或OBS桶中，具体请参考**导出存储数据**。
删除数据	勾选单个或多个数据，单击列表上方的“删除”，可删除单个或多个数据。

文件类型包含以下几种：

表 2-42 文件格式

文件类型	包含文件格式
图片	".bmp"，".jpg"，".jpeg"，".png"，".gif"，".psd"，".tiff"，".wmf"。
点云	".pts"，".las"，".pcd"，".xyz"，".pcap"。
文本	".txt"，".json"，".yaml"，".doc"，".docx"，".csv"，".xlsx"，".xls"，".xml"。
视频	".rmvb"，".rm"，".mp4"，".3gp"，".avi"，".mpg"，".mov"，".wmv"，".mpeg"，".dat"，".asf"。
场景	".xosc"，".xodr"，".osgb"。
音频	".wav"，".flac"，".mp3"，".m4a"。
其他	除以上列表中的其他格式。

- 数据处理任务：显示入库的数据处理任务。

表 2-43 数据处理任务相关操作

任务	操作步骤
查看任务	单击“任务ID”，页面可跳转至任务列表，可查看任务详情。
查看原始数据	单击“原始数据ID”，页面可跳转至数据包列表，可查看数据包详情。
查询数据处理任务	在搜索框中输入任务ID，即可查询。
删除数据处理任务	– 选择单个数据处理任务，单击操作栏的“ 删除”，删除单个数据处理任务。 – 勾选多个数据处理任务，单击列表上方的“删除”，可删除多个数据处理任务。
查看任务的数据详情	单击操作栏内的“数据详情”，可查看此任务包含的数据详情。

- 导出OBS任务：显示数据导出为OBS的任务，包括任务名称，OBS路径，导出状态和创建时间等。

表 2-44 导出 OBS 任务相关操作

任务	操作步骤
查询导出OBS任务	在搜索框中输入搜索条件，即可查询。单击任务名称进入任务详情，在详情里可以查看导出的参数和任务日志。
删除导出OBS任务	– 选择单个导出OBS任务，单击操作栏的“ 删除”，删除单个导出OBS任务。 – 勾选多个导出OBS任务，单击列表上方的“删除”，可删除多个导出OBS任务。

----结束

导出存储数据

步骤1 在左侧菜单栏中，选择“数据资产 > 通用存储”。

步骤2 在“通用存储”页签，单击目录名称，进入通用存储详情页。

步骤3 在“存储数据”页签的“数据索引”中，选择单个或多个文件，单击“导出”。

说明

系统提供目录树页面，将数据按照目录树关系展示，可单击“目录树”，选择单个或多个文件，单击“导出”。

步骤4 单击“下一步”，选择数据目的地。

说明

- 本地导出仅限单个对象，多个对象不允许选择。
- 单个通用存储的对象下载到浏览器的大小不能超过10GB。

- 本地：下载至本地。
- OBS：下载至OBS桶中，需勾选云服务声明。
  - 访问密钥：请输入访问密钥（AK）。
  - 私有访问密钥：请输入私有访问密钥（SK）。
  - OBS目录：请指定数据集导出后存放的目录。

    说明

    - 访问密钥ID（AK）和私有访问密钥（SK），在导入数据时，通过AK识别访问用户的身份，通过SK对请求数据进行签名验证，用于确保请求的机密性、完整性和请求者身份的正确性。
    - 输入的访问密钥和私有访问密钥需要具有OBS服务如下权限：

      obs:object:GetObject

      obs:object:PutObject

      obs:bucket:ListAllMyBuckets

      obs:bucket:ListBucket

步骤5 单击“下一步”，导出完成。

- 当去处选择“本地”时，在本地相关文件夹可查看。
- 当去处选择“OBS”时，在“导出OBS任务”中可查看任务。

----结束

2.10.2 自定义属性

数据处理只能识别带有标签的文件数据，进入通用存储后，用户可以给通用存储中的文件添加key-value映射关系的自定义属性。

创建自定义属性

步骤1 在左侧菜单栏中，单击“数据资产 > 通用存储 ”。

步骤2 选择“自定义属性”页签，单击“创建自定义属性”，填写基本信息。

步骤3 单击“确认”，自定义属性列表可查看新建自定义属性。

----结束

自定义属性相关操作

在“自定义属性”列表，还可以完成以下操作。

表 2-45 自定义属性相关操作

任务	操作步骤
查询自定义属性	可根据“属性名称”或“属性值”查询自定义属性。
删除自定义属性	只允许删除未被通用存储内的数据引用的自定义属性。如果选择的自定义属性被引用，则会提示不允许删除。 • 选择单个自定义属性，单击操作栏的“删除”，删除单个自定义属性。 • 勾选多个自定义属性，单击列表上方的“批量删除”，可批量删除自定义属性。

3 数据合规

3.1 数据合规简介

数据合规服务是自动驾驶云服务为用户提供数据硬盘递送、数据监管服务。

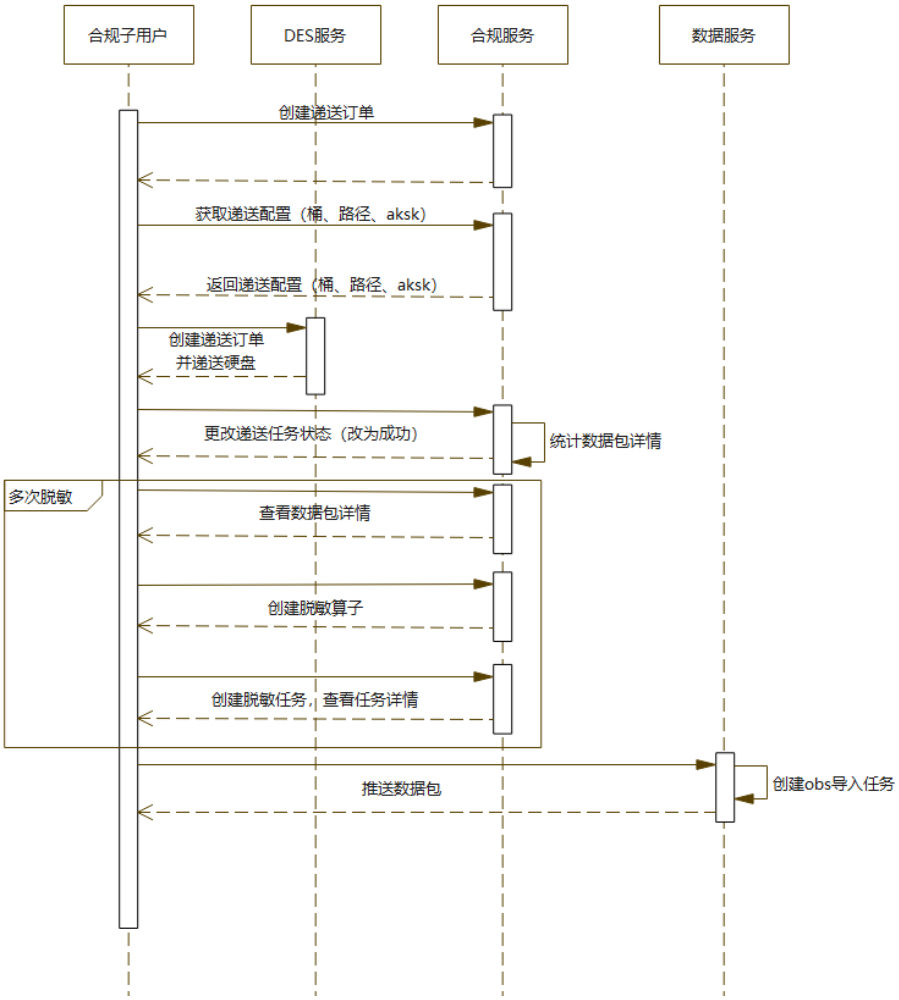
支持对上传合规服务的数据包进行脱敏处理，提供合规脱敏能力，支持将数据包中人脸、车牌、点云脱敏。

支持用户使用合规服务创建脱敏算子、下发脱敏任务。

说明

- 在使用数据合规服务之前，需要先购买合规服务和数据服务。如何**购买服务**?
- 在使用数据合规脱敏能力之前，需要先**开通合规脱敏服务**。

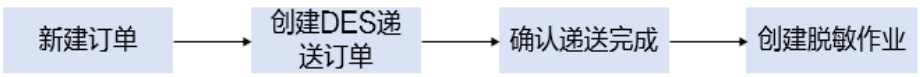
图 3-1 使用合规服务和数据服务时序图



3.2 数据递送

数据递送流程如下：

图 3-2 数据递送流程图



新建订单

步骤1 在左侧菜单栏中，单击“数据合规”。

步骤2 选择“数据递送”页签，单击“新建订单”。

- 数据盘个数：设置数据盘个数，输入值必须在1到12之间。
- 备注：简要描述订单信息。
- 勾选“我已阅读并同意《八爪鱼自动驾驶云服务声明》”。

步骤3 单击“确认”新建一个数据合规订单并查看订单号。

----结束

上传数据包

步骤1 获取传输配置。

在订单的“操作”列单击“获取传输配置”，获取传输配置，包括目标桶、导入目录、访问密钥 (AK)和私有访问密钥 (SK)。

步骤2 准备待脱敏的数据包。

图商采集的数据在上传至八爪鱼平台前，需要对数据包中的文件夹梳理，便于脱敏识别，目录层级需为“根目录/数据存放文件夹”，根目录需为**步骤3**创建的订单号。

上传至八爪鱼平台后数据路径：**data/import-expresses/***根目录/数据存放文件夹*

例如：**data/import-expresses/24/packge-1**

说明

- 24：根目录，即**步骤3**创建的订单号。
- packge-1：需要脱敏的数据存放文件夹。订单号和实际需要脱敏的文件夹之间不需要再加层级。
- 脱敏任务完成后平台存储的原始数据和脱敏后数据路径如下：
  - 原始数据：data/import-expresses/24/packge-1-bak
  - 脱敏后数据：data/import-expresses/24/packge-1

步骤3 上传待脱敏的数据包。

在DES数据快递服务创建递送订单，并完成递送订单。DES订单上传源数据需要包含必要的有效文件，具体可参考**上传数据格式**。

步骤4 数据上传完成后，单击“确认递送完成”，单击“确定”。

----结束

订单相关操作

表 3-1 订单相关操作

任务	操作步骤
确认递送完成	单击订单操作栏中的“确认递送完成”，用户可更改订单的状态。更改状态之前请确认DES快递服务创建递送订单完成。
查看订单详情	选择操作栏中的“更多 > 详情”，查看订单详情。在订单详情页面，选择数据包**创建脱敏作业**，对数据包进行脱敏操作。
编辑订单	单击订单操作栏中的“编辑”，可修改订单的备注信息。
订单推送	订单推送会将数据包推送到数据服务，可以到“ 数据处理 > 数据批导”中查看导入任务进度。
查询订单号	在“数据合规”订单列表页面，在搜索输入框中输入搜索条件，按回车键即可查询订单号。

任务	操作步骤
删除订单	选择操作栏中的“更多 > 删除”可删除订单。

创建脱敏作业

数据包上传成功后，可以选择指定的数据包创建脱敏作业。

步骤1 在订单的“操作”列，选择“更多 > 详情”进入订单详情列表。

步骤2 勾选待脱敏的数据包，单击“创建脱敏作业”，参考如下配置算子信息。

表 3-2 配置脱敏作业算子信息

参数	描述
处理算子	根据需要选择数据脱敏算子。
资源规格	根据需要选择合适的资源规格。
优先级	设定任务的优先级，数值取[-50,50]的整数，数字越大，优先级越高。
环境变量	配置算子的环境变量。最多可以添加10个环境变量。 ● Key：只能由英文、数字、和特殊符号（ ,-_ ）组成，且需要以字母开头 。长度不超过64个字符。 ● Value：只能由英文、数字和特殊符号（ \/,.[]-_ ）组成 。长度不超过64个字符。
合规数据ID	显示待脱敏数据包的ID。

步骤3 单击“创建”，执行数据包脱敏任务。脱敏完成后，数据包状态变为“脱敏完成”。

----结束

3.3 数据脱敏

3.3.1 作业总览

创建作业

步骤1 在左侧菜单栏中，单击“数据合规”。

步骤2 选择“数据脱敏 > 作业总览”页签，单击“创建作业”，填写脱敏作业信息。

表 3-3 脱敏作业

参数	示例	说明
处理算子	选择算子	根据需要选择已创建的算子。
资源规格	选择资源	当前项目中可用的资源规格，资源配置需要平台管理员在集群纳管中创建，支持选择带有GPU的资源规格。
优先级	0	设定任务的优先级，数值取[-50,50]的整数，数字越大，优先级越高。
环境变量	-	配置算子的环境变量。允许添加的环境变量个数不超过10个。 ● Key：只能由英文、数字、和特殊符号（ ,-_ ）组成，且需要以字母开头 。长度不超过64个字符。 ● Value：只能由英文、数字和特殊符号（ \/,.[]-_ ）组成 。长度不超过64个字符。
超时时间	24	作业运行时长超过该时间后，作业会自动停止。 如果该值为空，默认超时时间为24小时，或设置为[1，720]的整数。
选择数据	选择数据包	选择需要操作的数据包中的数据。

步骤3 单击“创建”，即可创建一个脱敏作业。

----结束

脱敏作业更多操作

在“作业总览”页签，还可以完成以下操作。

表 3-4 脱敏作业相关操作

任务	操作步骤
查看作业详情	单击操作栏中的“详情”，可查看作业的详情。 1. 单击作业总览列表，操作栏中的“详情”。 2. 在作业详情页，可在线查看或本地下载作业日志。
重启作业	单击操作栏中的“重启”，可根据需要保持或修改资源规格（默认显示当前资源规格），将目标作业进行重启。 **说明** 仅支持作业状态为“失败”、“停止”、“运行异常”和“上传失败”的作业。

任务	操作步骤
删除作业	• 选择单个作业，单击操作栏的“更多 > 删除”，删除单个作业。 • 勾选多个作业，单击列表上方的“删除”，可删除多个作业。 **说明** 当作业状态为终止态（提交失败、已停止、已完成、运行异常）时，可以删除作业。
停止作业	• 选择单个作业，单击操作栏的“更多 > 停止”，停止单个作业。 • 勾选多个作业，单击列表上方的“停止”，可批量停止作业。 **说明** 当作业状态为过程态（排队中、启动中、运行中）时，可以停止作业。
重新上传作业	单击操作栏中的“更多 > 重新上传”，可重新上传作业。 **说明** 仅支持作业状态为“上传失败”的作业。
搜索作业	• 在时间搜索框中设置搜索时间范围，单击“确认”，即可查询。 • 选择“ID”、“数据包ID”或“算子”，在搜索输入框中输入搜索条件，按回车键即可查询。

3.3.2 作业队列

平台为脱敏作业提供作业队列的功能，用户可在此查看任务队列，同时支持对任务优先级的调整。

说明

其他租户的队列作业不允许操作。

步骤1 在左侧菜单栏中单击“数据合规”。

步骤2 选择“数据脱敏 > 作业队列”页签，可查看作业。

----结束

3.3.3 算子管理

新建算子

步骤1 在左侧菜单栏中单击“数据合规”。

步骤2 选择“数据脱敏 > 算子管理”页签，单击“新建算子”。

步骤3 完成新建算子相关信息。

- 名称：算子名称，不得超过64个字符。支持中英文、数字、“-”、“_”，不支持特殊字符。
- 描述：算子内容、用途等的简要描述，不包含“@#$%^&*< > \”，不得超过255个字符。
- 可见范围：支持私有或团队。
  - 私有：该模板只有创建者可操作，其他用户不可见。
  - 团队：该模板当前工作空间下被授权的用户均可见。
- 运行镜像：可选择镜像仓库中已创建好的镜像。
- 镜像版本：选择镜像版本。
- 启动命令：请输入启动文件地址。
- 数据类型：此处只支持选择“数据脱敏”。对某些敏感信息通过脱敏规则进行数据的变形，实现敏感隐私数据的可靠保护。

步骤4 单击“确认”，即可创建一个脱敏算子。

----结束

脱敏算子相关操作

在“算子管理”页签，还可以完成以下操作。

表 3-5 脱敏作业相关操作

任务	操作步骤
查看算子详情	单击操作栏的“详情”，可查看算子的详情。
编辑算子	单击操作栏的“编辑”，可编辑算子的详情。
删除算子	单击操作栏的“删除”，可删除算子的详情。
搜索算子	选择“算子ID”或“算子名称”，在搜索输入框中输入搜索条件，按回车键即可查询。

3.3.4 算子示例

3.3.4.1 数据脱敏作业

作业输入输出规范

用户完成自定义脱敏算子创建，运行作业容器时Octopus平台向其中注入以下环境变量：

- input_file：待脱敏的文件路径
- raw_dir：抽取的image，gnss，lidar数据存放路径
- desensitized_dir：脱敏后的image，gnss，lidar数据存放路径
- output_dir：脱敏后的文件存放路径

用户根据需要可以自定义环境变量，以rosbag文件为例，可以定义如下环境变量：

- rosbag_version：robag版本
- image_topics：图像数据topic列表
- gnss_topic：gnss数据topic
- lidar_topics：lidar数据topic列表

用户作业容器需要将input_file中的image，gnss，lidar数据抽取到raw_dir，待系统内置算子脱敏完成，并将脱敏后的数据存放到desensitized_dir后，用户算子根据input_file和desensitized_dir中的脱敏数据生成新的数据文件存放到output_dir。

文件结构如下：

```
-raw
--image
---topic0
----timestamp1.jpg
----timestamp2.jpg
---topic1
----timestamp1.jpg
--gnss
---gnss_topic.json
--lidar
---pcd_topic0
----timestamp0.pcd
----timestamp1.pcd
--SUCCESS
-desensitized
--image
---topic0
----timestamp1.jpg
----timestamp2.jpg
---topic1
----timestamp1.jpg
--gnss
---gnss_topic.json
--lidar
---pcd_topic0
----timestamp0.pcd
----timestamp1.pcd
--SUCCESS
-output
--test.bag
--SUCCESS
```

其中，“SUCCESS”文件为标识文件，标识所在阶段的任务结束。

示例代码

下面是rosbag脱敏的算子示例：

```
# mask.py
import json
import logging
import multiprocessing as mp
import os
import shutil
import time
from pathlib import Path
from typing import cast

import av
import numpy as np
import open3d
```



```
from rosbags.highlevel import AnyReader
from rosbags.interfaces import ConnectionExtRosbag1, ConnectionExtRosbag2
from rosbags.rosbag1 import Writer as Writer1
from rosbags.rosbag2 import Writer as Writer2
from rosbags.serde import cdr_to_ros1, serialize_cdr
from rosbags.typesys import get_types_from_msg, register_types
from rosbags.typesys.types import builtin_interfaces__msg__Time as Time
from rosbags.typesys.types import \
    sensor_msgs__msg__CompressedImage as CompressedImage
from rosbags.typesys.types import std_msgs__msg__Header as Header

logging.basicConfig(
    level=logging.INFO,
    format='%(asctime)s - %(levelname)s - %(message)s',
)
LOG = logging.getLogger(__file__)

# Octopus数据服务拉起镜像时灌入的环境变量
# 获取环境变量
input_path = os.getenv('input_path', 'data/hangyan-move.bag.bak')
raw_dir = os.getenv('raw_dir', 'empty_dir/raw')  # 抽取的文件存放目录
desens_dir = os.getenv('desensitized_dir', 'empty_dir/desens')  # 脱敏后的文件存放目录
output_dir = os.getenv('output_dir', 'empty_dir/output')
lidar_process_num = os.getenv('lidar_process_num', 5)  # lidar数据进程数

# 用户自定义环境变量
rosbag_version = os.getenv('rosbag_version', '1')  # rosbag版本，取值为'1'或'2'
image_topics = [
    x.strip(' ')
    for x in os.getenv('image_topics', '/camera_encoded_1').split(',')
]  # 图像数据的topic列表
gnss_topic = os.getenv('gnss_topic',
                       '/inspvax')  # gnss数据的topic，gnss数据只能有一个topic
lidar_topics = [
    x.strip(' ') for x in os.getenv('lidar_topic', '/pandar').split(',')
]  # 点云数据的topic列表

# 注册自定义消息类型
Video_encoded_data_text = Path('msgs/Video_encoded_data.msg').read_text()
NovatelMessageHeader_text = Path('msgs/NovatelMessageHeader.msg').read_text()
NovatelExtendedSolutionStatus_text = Path(
    'msgs/NovatelExtendedSolutionStatus.msg').read_text()
NovatelReceiverStatus_text = Path('msgs/NovatelReceiverStatus.msg').read_text()
Inspvax_text = Path('msgs/Inspvax.msg').read_text()
add_types = {}
add_types.update(
    get_types_from_msg(
        Video_encoded_data_text,
        'kyber_msgs/msg/Video_encoded_data',
    ))
add_types.update(
    get_types_from_msg(
        NovatelMessageHeader_text,
        'novatel_gps_msgs/msg/NovatelMessageHeader',
    ))
add_types.update(
    get_types_from_msg(
        NovatelExtendedSolutionStatus_text,
        'novatel_gps_msgs/msg/NovatelExtendedSolutionStatus',
    ))
add_types.update(
    get_types_from_msg(
        NovatelReceiverStatus_text,
        'novatel_gps_msgs/msg/NovatelReceiverStatus',
    ))
add_types.update(
    get_types_from_msg(
        Inspvax_text,
        'novatel_gps_msgs/msg/Inspvax',
```



```
    ))
register_types(add_types)

# create gnss file
gnss_file_path = Path(raw_dir, 'gnss') / f'{gnss_topic}.json'.strip('/')
Path.mkdir(gnss_file_path.parent, parents=True, exist_ok=True)


def extract_image(input_rosbag):
    '''从原始rosbag中抽取图像数据.'''
    LOG.info('Start extracting image.')
    codec_ctx = av.codec.Codec('hevc', 'r')
    h265_code = codec_ctx.create()
    with AnyReader([Path(input_rosbag)]) as reader:
        for connection, timestamp, data in reader.messages():
            topic = connection.topic
            if topic in image_topics:
                deserialized_data = reader.deserialize(data,
                                                       connection.msgtype)
                try:
                    data = deserialized_data.raw_data
                    packet = av.packet.Packet(data)
                    out = h265_code.decode(packet)
                    img = None
                    for frame in out:
                        if frame.format.name != 'rgb24':
                            frame = frame.reformat(format='rgb24')
                        img = frame.to_image()
                    # 图像存放路径
                    file_name = f'{timestamp}.jpg'
                    f_path = Path(raw_dir, 'image') / topic.strip('/')
                    tmp_path = Path(raw_dir, 'tmp_image') / topic.strip('/')
                    Path.mkdir(tmp_path, parents=True, exist_ok=True)
                    tmp_file = tmp_path / file_name
                    file = f_path / file_name
                    # 当未建立目录时，先基于topic名称建立目录
                    Path.mkdir(file.parent, parents=True, exist_ok=True)
                    img.save(tmp_file)
                    os.chmod(tmp_file, 0o777)
                    os.chmod(file.parent, 0o777)
                    shutil.move(tmp_file, file)
                except Exception as e:
                    LOG.info("%s frame can not trans to jpg, message: %s",
                             timestamp, str(e))
    LOG.info('Finish extracting image.')


def extract_lidar(task_id, task_num, input_rosbag):
    '''从原始rosbag中抽取点云数据.'''
    LOG.info('Start extracting pcd.')
    with AnyReader([Path(input_rosbag)]) as reader:
        for i, (connection, timestamp, data) in enumerate(reader.messages()):
            if i % task_num != task_id:
                continue
            topic = connection.topic
            if topic in lidar_topics:
                deserialized_data = reader.deserialize(data,
                                                       connection.msgtype)
                pcd = open3d.geometry.PointCloud()
                reshaped = deserialized_data.data.reshape(
                    int(len(deserialized_data.data) / 3), 3)
                pcd.points = open3d.utility.Vector3dVector(reshaped)

                file_name = f'{timestamp}.pcd'
                f_path = Path(raw_dir, 'lidar') / topic.strip('/')
                tmp_path = Path(raw_dir, 'tmp_lidar') / topic.strip('/')
                Path.mkdir(tmp_path, parents=True, exist_ok=True)
                tmp_file = tmp_path / file_name
                file = f_path / file_name
```



```
            # 当未建立目录时，先基于topic名称建立目录
            Path.mkdir(file.parent, parents=True, exist_ok=True)
            open3d.io.write_point_cloud(str(tmp_file), pcd)
            os.chmod(tmp_file, 0o777)
            os.chmod(file.parent, 0o777)
            shutil.move(tmp_file, file)
    LOG.info('Finish extracting pcd.')


def extract_gnss(input_rosbag):
    '''从原始rosbag中抽取gnss数据.'''
    LOG.info('Start extracting rosbag.')
    gnss_file = open(gnss_file_path, 'w')
    gnss = dict()
    with AnyReader([Path(input_rosbag)]) as reader:
        for connection, timestamp, data in reader.messages():
            topic = connection.topic
            if topic == gnss_topic:
                deserialized_data = reader.deserialize(data,
                                                       connection.msgtype)
                # 这里以msgytpe为NavSatFix为例
                latitude = deserialized_data.latitude
                longitude = deserialized_data.longitude
                altitude = deserialized_data.altitude
                gnss[timestamp] = {
                    'latitude': latitude,
                    'longitude': longitude,
                    'altitude': altitude
                }
    gnss_file.write(json.dumps(gnss))
    gnss_file.close()
    LOG.info('Finish extracting gnss.')


def _get_masked_image(topic, timestamp):
    '''从脱敏后的图像数据中获取目标图像数据.'''
    file = Path(desens_dir, 'image') / topic.strip('/') / f'{timestamp}.jpg'
    if file.is_file():
        return np.fromfile(file, dtype='uint8')
    else:
        return None


def _get_conn_map(rosbag_version: int, reader, writer):
    '''构建connection的索引.'''
    conn_map = {}
    if rosbag_version == '1':
        for conn in reader.connections:
            if conn.topic in image_topics:
                conn_map[conn.id] = writer.add_connection(
                    '/image',
                    CompressedImage.__msgtype__,
                )
            else:
                ext = cast(ConnectionExtRosbag1, conn.ext)
                conn_map[conn.id] = writer.add_connection(
                    conn.topic,
                    conn.msgtype,
                    conn.msgdef,
                    conn.md5sum,
                    ext.callerid,
                    ext.latching,
                )
    elif rosbag_version == '2':
        for conn in reader.connections:
            if conn.topic in image_topics:
                conn_map[conn.id] = writer.add_connection(
                    '/image',
                    CompressedImage.__msgtype__,
```



```
            )
        else:
            ext = cast(ConnectionExtRosbag2, conn.ext)
            conn_map[conn.id] = writer.add_connection(
                conn.topic,
                conn.msgtype,
                ext.serialization_format,
                ext.offered_qos_profiles,
            )
    return conn_map


def _serialize_data(rosbag_version, data, msgtype):
    '''对数据进行序列化.'''
    if rosbag_version == '1':
        return cdr_to_ros1(serialize_cdr(data, msgtype), msgtype)
    elif rosbag_version == '2':
        return serialize_cdr(data, msgtype)


def generate_rosbag(input_rosbag, output_rosbag):
    '''生成脱敏后rosbag.'''
    LOG.info('Start generating rosbag.')
    gnss_file = open(
        Path(desens_dir, 'gnss') / f'{gnss_topic}.json'.strip('/'), 'r')
    gnss_data = json.load(gnss_file)
    gnss_file.close()
    Writer = Writer1 if rosbag_version == '1' else Writer2
    with AnyReader([Path(input_rosbag)
                    ]) as reader, Writer(Path(output_rosbag)) as writer:
        conn_map = _get_conn_map(rosbag_version, reader, writer)
        for connection, timestamp, data in reader.messages():
            topic = connection.topic
            # 当topic为图像数据的topic时，读取脱敏后图像数据
            if topic in image_topics:
                masked_data = _get_masked_image(topic, timestamp)
                if masked_data is None:  # 没有解析出图像文件时，不要该帧了
                    continue
                deserialized_data = CompressedImage(
                    Header(
                        stamp=Time(
                            sec=int(timestamp // 10**9),
                            nanosec=int(timestamp % 10**9),
                        ),
                        frame_id='0',
                    ),
                    format='jpg',
                    data=masked_data,
                )
                data = _serialize_data(
                    rosbag_version,
                    deserialized_data,
                    CompressedImage.__msgtype__,
                )
            # 当topic为gnss数据时，读取脱敏后gnss数据
            elif topic == gnss_topic:
                deserialized_data = reader.deserialize(data,
                                                       connection.msgtype)
                deserialized_data.latitude = gnss_data.get(
                    str(timestamp)).get('latitude')
                deserialized_data.longitude = gnss_data.get(
                    str(timestamp)).get('longitude')
                deserialized_data.altitude = gnss_data.get(
                    str(timestamp)).get('altitude')
                data = _serialize_data(
                    rosbag_version,
                    deserialized_data,
                    connection.msgtype,
                )
```



```
            # 当topic为点云数据时，读取脱敏后点云数据
            elif topic in lidar_topics:
                deserialized_data = reader.deserialize(
                    data,
                    connection.msgtype,
                )
                file = Path(
                    desens_dir,
                    'lidar',
                ) / topic.strip('/') / f'{timestamp}.pcd'
                point_cloud = open3d.io.read_point_cloud(str(file))
                deserialized_data.data = np.asarray(
                    point_cloud.points).flatten()
            writer.write(conn_map[connection.id], timestamp, data)
    # 生成_SUCCESS文件标识完成数据抽取
    Path(output_dir, '_SUCCESS').touch()
    LOG.info('Finish generating rosbag.')


if __name__ == "__main__":
    LOG.info('Start user operator.')
    process_image = mp.Process(target=extract_image, args=(input_path, ))
    pool_lidar = mp.Pool(processes=lidar_process_num)
    for i in range(lidar_process_num):
        pool_lidar.apply_async(extract_lidar,
                               args=(i, lidar_process_num, input_path))
    process_gnss = mp.Process(target=extract_gnss, args=(input_path, ))
    # 启动子进程
    process_image.start()
    pool_lidar.close()
    process_gnss.start()
    process_image.join()
    pool_lidar.join()
    process_gnss.join()
    LOG.info('Child processes exit.')
    # 生成_SUCCESS文件标识完成数据抽取
    Path(raw_dir, '_SUCCESS').touch()

    # 后面输出的rosbag文件与输入的rosbag文件保持同名
    output_rosbag_file = Path(output_dir, Path(input_path).name)
    # 如果输出文件夹不存在，先创建文件夹
    Path.mkdir(output_rosbag_file.parent, parents=True, exist_ok=True)
    # 检测到脱敏任务结束后，生成新的rosbag文件
    while time.sleep(1) is None:
        if Path(desens_dir).joinpath('_SUCCESS').is_file():
            generate_rosbag(Path(input_path), output_rosbag_file)
            break
```

构建镜像

Dockerfile示例

```
FROM python:3.10
COPY mask.py /home/main/
WORKDIR /home/main/
RUN pip install rosbags requests numpy
USER root
```

构建镜像

```
sudo docker build --no-cache -f Dockerfile -t rosbagmask:0.1 .
```

本地调试

准备一个待处理的rosbag包。


```
sudo docker run -v ${HOME}/data/test.bag:/home/main/test.bag --env input_file=/home/main/test.bag –
env raw_dir=/home/main/raw_dir –env desensitized_dir=/home/main/desensitized_dir –env output_dir=/
home/main/output_dir -it rosbagmask:0.1 /bin/sh -c "python mask.py"
```


4 数据处理

4.1 数据处理简介

数据处理可对自动驾驶过程中采集到的数据进行处理、解析，处理的结果可以用于回访定位问题，并可根据不同功能的算子生成不同的数据处理作业。数据处理操作引导如下：

- **数据批导**：创建数据导入任务，收集采集车辆原始数据。
- **数据处理**：支持根据自定义数据服务算子，对数据包进行处理，创建不同类型的作业。
- **回放仿真**：通过回放仿真算子生成回放仿真作业，支持对比回放功能。

4.2 数据批导

4.2.1 数据批导简介

在使用平台进行自动驾驶开发时，首先需要用户把原始数据上传至Octopus平台，即数据上云。下文介绍了通过对象存储导入数据的方式，帮助用户把车载平台输出的数据上传至Octopus平台上。

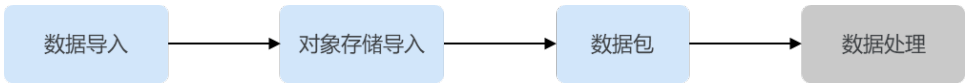
对象存储导入数据：用户需先将数据包上传至对象存储桶（可自定义）中，再导入到Octopus平台。上传速率快，适合上传大数据量（超过2G）的数据包。

说明

- 单个数据包时长均不可超过10分钟。
- 目前MP4文件只支持对象存储导入。

数据导入任务的流程如下：

图 4-1 数据导入任务流程



4.2.2 数据导入

对象存储导入数据

把原始数据上传至平台前，可将数据包上传至对象存储中，从对象存储导入平台。操作步骤如下：

步骤1 在左侧菜单栏中，选择“数据处理 > 数据批导”。

步骤2 在“数据导入”页签单击“导入”，填写具体信息。

选择目标数据包所在的对象存储路径，平台将自动扫描该目录下所有子目录。将此文件夹下的子目录都识别为有效数据包，可导入。上传数据前，请确保原始数据包含必要的有效文件，具体可参考4.2.3.1 上传数据格式。

表 4-1 数据导入参数说明

参数	说明
对象存储访问密钥（AK） 对象存储私有访问密钥（SK）	包含访问密钥ID（AK）和私有访问密钥（SK）两部分，导入数据时，通过AK识别访问用户的身份，通过SK对请求数据进行签名验证，用于确保请求的机密性、完整性和请求者身份的正确性。
对象存储目录	建议对象存储目录选择包含“Rosbag包和与数据包同名yaml”文件夹的上一级目录。
数据包筛选	数据包需包含符合平台要求的传感器数据文件。

说明

- 对象存储访问密钥（AK/SK）需要具有OBS服务如下权限：
  obs:object:GetObject
  obs:object:PutObject
  obs:bucket:ListAllMyBuckets
  obs:bucket:ListBucket

步骤3 数据处理。

可选择数据是否进行OpenData转换，并且选用相对应的数据服务算子（数据回放）以及资源规格。

说明

- 如果yaml和算子中有相同的信息，算子上传后，会覆盖yaml中的信息。
- 没有进行OpenData转换的数据，只能作为算子作业的输入使用。

步骤4 勾选“我已阅读并同意《八爪鱼自动驾驶云服务声明》”。

步骤5 单击“确认”，提交收集任务，界面跳转至数据导入列表。

步骤6 进入任务详情页。

在数据导入列表，单击操作栏中的“详情”，进入到任务详情页。

步骤7 预览数据导入任务信息。

步骤8 查看任务状态。

在任务列表，用户可通过查看任务所处状态了解任务当前进度。

----结束

数据导入相关操作

在“数据导入”列表，还可以完成以下操作。

表 4-2 数据导入相关操作

任务	操作步骤
查看任务详情	1. 单击操作栏的“详情”，可查看任务详情。 2. 在任务详情页，单击数据包名称和算子名称，可跳转至相对应的详情界面。
查询导入任务	在搜索输入框中输入搜索条件，按回车键即可查询。
删除导入任务	单击导入列表后操作栏内的“删除”，可永久性删除导入任务。删除中的导入任务不可删除。
重启导入任务	重启导入状态为“失败”的任务。 ● 单击导入任务操作栏内的“重启”。可重启指定任务。 ● 勾选导入任务，单击列表上方的“重启”，可批量重启指定任务。

4.2.3 数据包格式

4.2.3.1 上传数据格式

在使用Octopus平台收集数据前，请仔细阅读本章节，确保上传数据格式符合平台要求，有助于用户更快速地完成数据收集以及数据格式转换。

上传数据格式：单包上传大小小于200GB。

转换后数据格式：OpenData格式。

使用场景

Octopus平台接收数据包（数据包至少需在一层文件夹内），没有转换OpenData格式时，可用于算子作业输入。

命名规范

用户可将多个数据包存放在同个一级目录下，作为同一批次采集数据上传至Octopus平台。

一级目录的结构可根据业务情况自定义数据包名称，如果数据包为Rosbag格式，示例参考如下：

```
.
├─ octopus
   ├─ Rosbag
      ├─{导入id}
         ├─{数据包名}
            ├─ {数据包名}.bag
            ├─ {数据包名}.yaml     #非必填
            ├─ xxxxx.mp4           #非必填，大小不限制，命名大小不可超过128位，不可有特殊字符。
```

与数据包同名的 yaml 配置文件说明

数据包中必须含有与数据包同名的yaml配置文件主要包括车辆名称、传感器信息和标定ID等信息，详情参考如下：

```
# 华为八爪鱼自动驾驶云服务数据采集说明
project: '项目名称'
module: '感知'
cardrive:
   collect_time: 2020-11-01T08:00:00+08:00     #数据包采集日期，精确到小时即可
   station: '腾飞'                          #选填 数据采集地点名称，站点名称
   car:
      vehicle_name: 'test'              #车辆名称，仅支持在八爪鱼平台创建的车辆
   route: 'shuttlebus_30km'                  #选填 车辆行驶路线
   speed：10km/h                          #选填 车速
   mode: 'auto'                           #选填 路线驾驶意图, auto代表自动驾驶, manual代表人工驾驶采集
   tags: ['主车直行','主车倒车']             #选填 标签，标签个数不超过50个 例：沙尘天，正向设计，驾驶模式
   description: '强风沙天,车辆空载在排土区自动驾驶到接土区前等待长坡道'   #选填 车载情况
   segments:                              #选填 数据包场景片段
      -
        tags: ['晴天','直行']
        time: 2021-08-27T11:43:07~2021-08-27T11:43:47
data_type: Rosbag                         #必填 数据类型
map_id: MAP1134                           #选填，高精地图ID，字符串类型，配备后才可在回放数据界面展示高精地图信息。
preprocessor:                             #转OpenData算子信息
   id: 10105                             # 算子id
   resource_spec: X86_4Core_8GiB              # 资源规格
```

4.2.3.2 转换后数据格式

Octopus平台支持将上传的Rosbag格式转换为OpenData格式。

数据类型

Octopus平台对数据有以下要求：

- 数据类型：包括各传感器数据、车辆数据、目标推理数据、自车坐标姿态以及标签记录数据等。
- 数据格式：Octopus OpenData格式。其中相机采集数据文件后缀为“.jpg”，激光雷达采集数据文件后缀为“.pcd”，其他采集数据文件后缀为“.pb”（谷歌定义的protobuf格式文件）。

  详情请参考**表4-3**。
- 消息topic具体格式要求请参考“**4.2.3.3 消息topic格式规范**”。

- 接收到的消息topic示例请参考“4.2.3.4 消息topic格式示例”。

说明

- 自车相关或每个传感器设备，都对应一个消息topic。
- 采集数据的topic名称支持自定义，包含中英文、数字、“_”“-”，不得超过64个字符。

表 4-3 数据类型和消息 topic 对应关系

分类	数据类型	消息topic（示例）	文件后缀	备注
传感器	相机（camera）	camera_front	.jpg	录制车辆路况图像数据。
	激光雷达（lidar）	lidar_roof_0	.pcd	以发射激光束探测目标的位置、速度等特征量的雷达系统，探测车辆周围的目标位置，监测移动速度。
	位置数据（gnss）	gnss_raw	.pb	通过卫星导航系统，定位车辆位置。
	毫米波雷达（radar）	RADAR_FRONT	.pcd	工作在毫米波段探测的雷达，探测车辆周围的目标位置，监测移动速度。
车辆数据	自车坐标和姿态数据（ego_tf）	ego_tf	.pb	定位自车所处位置以及当前车辆姿态。
	车辆数据（vehicle）	vehicle	.pb	车辆底盘信息。
规划推理数据	目标推理数据（object_array_vision）	object_array_vision	.pb	感知数据信息。
标签数据	标签记录数据（tag_record）	tag_record	.pb	在车端标记驾驶过程中人工和自动驾驶路段以及其他重要信息。
控制数据	控制指令（control）	control	.pb	自车的方向盘转角、加速度值等控制数据。
规划路径	规划轨迹（planning_trajectory）	planning_trajectory	.pb	自车规划行驶路径。
预测路径	预测跟踪（predicted_objects）	predicted_objects	.pb	感知目标的预测路径。
全局规划	全局路径（routing_path）	routing_path	.pb	自车全局规划路径。
交通灯	交通灯信息（traffic_light_info）	traffic_light_info	.pb	红绿灯。

使用场景

Octopus平台接收到原始数据后，将对数据进行解包、轨迹和接管分析等操作，用于数据总览、数据场景、数据回放、标注服务等模块，请用户结合实际需求，准备好相应模块所需数据。

Octopus平台转换后的OpenData数据服务模块所需数据请见下表：

表 4-4 数据和模块对应关系

类型	消息	数据总览	数据场景	数据回放	标注服务
相机	camera	-	-	√	√
激光雷达	lidar	-	-	√	√
位置数据	gnss	√	-	√	-
自车坐标姿态	ego_tf	-	√	√	-
车辆数据	vehicle	-	√	√	-
感知推理	object_array_vision	-	√	√	-
接管及打标签信息	tag_record	-	-	√	-
控制指令	control	-	-	√	-
规划轨迹	planning_trajectory	-	-	√	-
预测跟踪	predicted_objects	-	-	√	-
全局规划	routing_path	-	-	√	-
交通灯	traffic_light_info	-	-	√	-
毫米波雷达	radar	-	-	√	-

采集数据命名规范

用户可将多个数据包存放在同个一级目录下，作为同一批次采集数据上传至Octopus平台。

一级目录的结构可根据业务情况自定义数据包名称，示例参考如下：

```
.
├── raw-data
├── package01
├── OpenData
```

```
├─ 2020-04-28-08-00
└─ ...
```

Octopus OpenData 格式数据说明

Octopus OpenData格式数据目录结构可根据实际采集节点种类及数量进行修改，示例参考如下：

```
.
├─ OpenData
    ├─ camera_0
        ├─xxx.jpg
        ├─xxx.jpg
        ├─timestamp.jpg
        ├─...
    ├─ lidar_roof_0
        ├─xxx.pcd
        ├─xxx.pcd
        ├─timestamp.pcd
        ├─...
    ├─ gnss
        ├─episode-1.pb
    ├─other sensor
        ├─episode-1.pb
    ...

    ├─ octopus_data_collection.yaml
```

说明

1. Octopus平台对Octopus OpenData数据包内文件大小限制如下：
   - “.yaml”文件小于10kb。
   - “.jpg”文件小于2MB。
   - “.pcd”文件小于10MB。
   - “.pb”文件小于50MB。
2. MP4命名大小不可超过128位，不可有特殊字符。MP4的时间需要和Rosbag包中的时间匹配。
3. 数据名称仅包含中文、大小写英文、数字、“-”“_”，不超过64位。
4. 数据包名称仅包含中文、大小写英文、数字、“-”“_”，不得出现其他字符，且长度不超过64个字符。

“octopus_data_collection.yaml”配置文件说明

数据包中有“Octopus_data_collection.yaml”配置文件，各类型传感器的名字必须和文件夹名称一致，格式也必须与规范相匹配。

配置文件，主要包括车辆名称、传感器信息和标定ID等信息，详情参考如下：

```
# 华为八爪鱼自动驾驶云服务数据采集说明
cardrive:
    collect_time: 2020-11-01T08:00:00+08:00     #数据包采集日期，精确到小时即可
    station: '腾飞'                          #选填 数据采集地点名称，站点名称
    car:
        vehicle_name: 'test0805'            #车辆名称，仅支持在八爪鱼平台创建的车辆
    route: 'shuttlebus_30km'                #选填 车辆行驶路线
    mode: 'auto'                          #选填 路线驾驶意图, auto代表自动驾驶, manual代表人工驾驶采集
    tags:                                #选填 数据包对应标签ID
    description: ''                       #选填 数据包描述
data_type: opendata                         #必填 数据包类型，转换后的OpenData数据中包含
ocotopus_data_collection.yaml文件
map_id:''                                 #选填，高精地图ID，字符串类型，配备后才可在回放数据界面展示高精地图信息。
```


4.2.3.3 消息 topic 格式规范

Vehicle

对于车辆自身基本数据录制的消息格式，需遵循一定规范，其中部分字段为必选，其他请根据实际需要自由选取。

表 4-5 vehicle 消息格式规范

格式名称	说明
VehicleInfo	车辆信息

须知

消息格式中部分参数为必选，如使用该数据类型，则不可缺少该参数字段，否则会导致数据上传Octopus平台失败。

```
/**********************************
    content: Octopus 输入数据格式
    version: 0.1
**********************************/
syntax = "proto3";

package Octopusdata;

message VehicleFrame {
    uint64 stamp_secs = 1;                #必选。时间戳，单位：秒
    uint64 stamp_nsecs = 2;               #必选。时间戳，单位：纳秒
    uint32 autonomy_status = 3;             #非必选。自动驾驶状态
    sint32 gear_value = 4;                #必选。只应从枚举常量中赋值
    float vehicle_speed = 5;              #必选。行驶速度，如果齿轮是倒挡，值为负。
    float steering_angle = 6;             #必选。转向，以角度表示。顺时针或向右为正，0为垂直或直角。
    float yaw_rate = 7;                  #Unit: deg/s
    float interior_temperature = 8;         #Unit: Celsius
    float outside_temperature = 9;          #Unit: Celsius
    float brake = 10;                    #必选。刹车制动按压百分比（0代表不按，1代表完全按下）。
        uint64 timestamp = 11;                #必选。时间戳。
    int32 turn_left_light=12;             #必选。左转灯。
    int32 turn_right_light=13;            #必选。右转灯。
    float longitude_acc=14;               #必选。纵向加速度。
    float lateral_acc=15;                 #必选。横向加速度。

}

message VehicleInfo {
    repeated VehicleFrame vehicle_info = 1;
}
```

Camera

采集的camera数据通过转换工具可以保存为“.jpg”图片数据。

Lidar

采集的点云数据通过转换工具可以保存为标准的pcd格式数据。

Gnss

对于卫星导航系统数据录制的消息格式，需遵循一定规范，其中部分字段为必选，其他请根据实际需要自由选取。

表 4-6 gnss 消息格式规范

格式名称	说明
GnssPoints	gps点

须知

消息格式中部分字段为必选，如使用该数据类型，则不可缺少该参数字段，否则会导致数据上传Octopus平台失败。

```
/**************************************
    content: Octopus 输入数据格式
    version: 1.0
**************************************/
syntax = "proto3";

package Octopusdata;

message GnssPoint {
    uint64 stamp_secs = 1;     #必选。时间戳，单位：秒
    uint64 stamp_nsecs = 2;    #必选。时间戳，单位：纳秒
        float latitude = 3;        #必选。纬度
        float longitude = 4;       #必选。经度
        float elevation = 5;       #必选。海拔高度，单位：米
        uint64 timestamp = 6;      #必选。时间戳
}

message GnssPoints {
    repeated GnssPoint gnss_points = 1;
}
```

Radar

雷达采集的点云数据通过转换工具可以保存为标准的pcd格式数据。

Ego_tf

对于自车角度位置数据录制的消息格式，需遵循一定规范，其中部分字段为必选，其他请根据实际需要自由选取。

表 4-7 ego_tf 消息格式规范

格式名称	说明
LocalizationInfo	主车信息

须知

消息格式中部分字段为必选，如使用该数据类型，则不可缺少该参数字段，否则会导致数据上传Octopus平台失败。

```
/**************************************
    content: Octopus 输入数据格式
    version: 1.0
**************************************/
syntax = "proto3";

package Octopusdata;


message LocalizationInfoFrame {
    uint64 timestamp = 1;                #必选。时间戳。
    uint64 stamp_secs = 2;               #必选。时间戳，单位：秒
    uint64 stamp_nsecs = 3;              #必选。时间戳，单位：纳秒
    float pose_position_x = 4;           #必选。自车x轴坐标
        float pose_position_y = 5;           #必选。自车y轴坐标
        float pose_position_z = 6;           #必选。自车z轴坐标
    float pose_orientation_x = 7;        #必选。自车四元数x值
        float pose_orientation_y = 8;        #必选。自车四元数y值
        float pose_orientation_z = 9;        #必选。自车四元数z值
    float pose_orientation_w = 10;       #必选。自车四元数w值
    float pose_orientation_yaw=11;       #必选。朝向角，单位：rad
    float velocity_linear=12;            #必选。速度，单位：m/s
    float velocity_angular=13;           #必选。角速度，单位：rad/s
    float acceleration_linear=14;        #必选。加速度，单位：m^2/s
    float acceleration_angular=15;       #必选。角加速度，单位：rad^2/s
}

message LocalizationInfo {
    repeated LocalizationInfoFrame localization_info = 1;
}
```

Object_array_vision

对于目标推理数据录制的消息格式，需遵循一定规范，其中部分字段为必选，其他请根据实际需要自由选取。

表 4-8 object_array_vision 消息格式规范

格式名称	说明
TrackedObject	感知目标

须知

消息格式中部分字段为必选，如使用该数据类型，则不可缺少该参数字段，否则会导致数据上传Octopus平台失败。

```
/**************************************
    content: Octopus 输入数据格式
    version: 1.0
**************************************/
syntax = "proto3";
```

```
package Octopusdata;


message Object {
    uint64 id = 1;                    #必选。目标推理数据object数组id
    string label = 2;                 #必选。标记物体类型
    float pose_position_x = 3;          #必选。目标物x轴坐标
        float pose_position_y = 4;          #必选。目标物y轴坐标
        float pose_position_z = 5;          #必选。目标物z轴坐标
    float pose_orientation_x = 6;       #必选。目标物四元数x值
        float pose_orientation_y = 7;       #必选。目标物四元数y值
        float pose_orientation_z = 8;       #必选。目标物四元数z值
    float pose_orientation_w = 9;       #必选。目标物四元数w值
    float pose_orientation_yaw = 10;    #必选。朝向角，单位：rad
    float dimensions_x = 11;            #必选。目标物x方向尺寸（长）
    float dimensions_y = 12;            #必选。目标物y方向尺寸（宽）
    float dimensions_z = 13;            #必选。目标物z方向尺寸（高）
    float speed_vector_linear_x = 14;   #必选。目标物x方向速度
        float speed_vector_linear_y = 15;   #必选。目标物y方向速度
        float speed_vector_linear_z = 16;   #必选。目标物z方向速度
        float relative_position_x = 17;     #必选。目标物相对于主车x方向位置
        float relative_position_y = 18;     #必选。目标物相对于主车y方向位置
        float relative_position_z = 19;     #必选。目标物相对于主车z方向位置
}
message TrackedObjectFrame {
    uint64 timestamp = 1;               #必选。时间戳
    uint64 stamp_secs = 2;              #必选。时间戳，单位：秒
    uint64 stamp_nsecs = 3;             #必选。时间戳，单位：纳秒
    repeated Object objects = 4;        #必选。object数组
}
message TrackedObject {
    repeated TrackedObjectFrame tracked_object = 1;
}
```

Tag_record

对于标签记录数据录制的消息格式，需遵循一定规范，其中部分字段为必选，其他请根据实际需要自由选取。

表 4-9 tag_record 消息格式规范

格式名称	说明
ScenarioSegments	场景片段

```
/**********************************
    content: Octopus 输入数据格式
    version: 1.0
**********************************/
syntax = "proto3";

package Octopusdata;

message ScenarioSegment {
 uint32 scenario_id = 1;              #必选。场景id
 string source = 2;                   #必选。片段的来源
 uint64 start = 3;                    #必选。片段的开始时间（时间戳）
 uint64 end = 4;                      #必选。片段的结束时间（时间戳）
}

message ScenarioSegments {
 repeated ScenarioSegment segments = 1;
}
```


Control

对于控制数据的消息格式，需遵循一定规范，其中部分字段为必选，其他请根据实际需要自由选取。

表 4-10 control 消息格式规范

格式名称	说明
ControlCommand	控制命令

```
/**********************************
    content: Octopus 输入数据格式
    version: 1.0
**********************************/
syntax = "proto3";
package Octopusdata;
message CommandFrame {
    uint64 stamp_secs = 1;
    uint64 stamp_nsecs = 2;
    uint64 timestamp = 3;                    #必选，时间戳
    float acceleration=4;                   #必选，加速度值
    float front_wheel_angle=5;                #必选，方向盘转角
    int32 gear=6;
}
message ControlCommand {
     repeated CommandFrame command_frame = 1;
}
```

Predicted_objects

对于预测路径数据的消息格式，需遵循一定规范，其中部分字段为必选，其他请根据实际需要自由选取。

表 4-11 predicted_objects 消息格式规范

格式名称	说明
PredictionObstacles	预测障碍物

```
/**********************************
    content: Octopus 输入数据格式
    version: 1.0
**********************************/
syntax = "proto3";
package Octopusdata;
message PathPoint {
    float x = 1;          #必选，预测轨迹点x坐标
    float y = 2;          #必选，预测轨迹点y坐标
    float z = 3;          #必选，预测轨迹点z坐标
    float theta = 4;
    float kappa = 5;
    int32 lane_id= 6;
    float v=7;
    float a=8;
    float relative_time=9;
}
message PredictionTrajectory {
```



```
  repeated PathPoint path_point = 1;  #必选，预测轨迹多个点
}
message Obstacle {
    uint64 obstacle_timestamp = 1;
    int32 id=2;                #必选，预测目标的id
    float x = 3;  #非必选，预测目标的x坐标
    float y = 4;  #非必选，预测目标的y坐标
    float z = 5;  #非必选，预测目标的z坐标
    repeated PredictionTrajectory prediction_trajectory = 6;  #必选，预测目标的多条轨迹
}
message PerceptionObstacle {
   uint64 stamp_secs = 1;
   uint64 stamp_nsecs = 2;
   uint64 timestamp = 3;  #必选，预测目标的时间戳
   repeated Obstacle obstacle_info= 4;  #必选，多个目标的预测信息
}
message PredictionObstacles {
   repeated PerceptionObstacle perception_obstacle= 4;  #必选，多条帧数据
}
```

Planning_trajectory

对于规划路径数据的消息格式，需遵循一定规范，其中部分字段为必选，其他请根据实际需要自由选取。

表 4-12 planning_trajectory 消息格式规范

格式名称	说明
PlanTrajectory	规划路径

```
/**************************************
    content: Octopus 输入数据格式
    version: 1.0
**************************************/
syntax = "proto3";
package Octopusdata;
message TrajectoryPoint {
    float x = 1;   #必选，轨迹点x坐标
    float y = 2;   #必选，轨迹点y坐标
    float z = 3;   #必选，轨迹点z坐标
    float theta = 4;
    float kappa = 5;
    int32 lane_id=6;
    float v=7;   #必选，速度
    float a=8;   #必选，加速度
    float relative_time=9;  #必选，相对时间
}
message Trajectory {
    uint64 stamp_secs = 1;
    uint64 stamp_nsecs = 2;
    uint64 timestamp = 3;    #必选，时间戳
    float total_path_length = 4;
    float total_path_time=5;
    int32 gear=6;      #非必选，档位
    int32 trajectory_type=7;
    int32 vehicle_signal=8;
    repeated TrajectoryPoint trajectory_points = 9;  #必选，轨迹
}
message PlanTrajectory {
     repeated Trajectory trajectory_info= 1;
}
```


Routing_path

对于全局规划路径数据的消息格式，需遵循一定规范，其中部分字段为必选，其他请根据实际需要自由选取。

表 4-13 routing_path 消息格式规范

格式名称	说明
RoutingFrames	规划路径

```
/**************************************
    content: Octopus 输入数据格式
    version: 1.0
**************************************/
syntax = "proto3";

message Point{
    float x = 1;
    float y = 2;
    float z = 3;
}

message Path{
    uint64 id = 1;
    repeated Point path_point = 2;
}

message RoutingPath{
    uint64 timestamp = 1;
    uint64 stamp_secs = 2;
    uint64 stamp_nsecs = 3;
    repeated Path routing_path_info = 4;
}

message RoutingFrames{
    repeated RoutingPath routing_frame = 4;
}
```

Traffic_light_info

对于交通灯数据的消息格式，需遵循一定规范，其中部分字段为必选，其他请根据实际需要自由选取。

表 4-14 traffic_light_info 消息格式规范

格式名称	说明
TrafficLightInfo	交通灯

```
/**************************************
    content: Octopus 输入数据格式
    version: 1.0
**************************************/
syntax = "proto3";

package Octopusdata;
```

```
message Light {
    uint64 id = 1;
    uint64 color = 2;
    uint64 state = 3;
    uint64 type = 4;
    float location_x = 5;
    float location_y = 6;
    float location_z = 7;
}

message Lights {
    uint64 timestamp = 1;
    uint64 stamp_secs = 2;
    uint64 stamp_nsecs = 3;
    repeated Light lights = 4;
}

message TrafficLightInfo {
    repeated Lights trafficlight_info = 1;
}
```

4.2.3.4 消息 topic 格式示例

消息topic具体格式要求请参考“**消息topic格式规范**”。接收到的消息topic示例请参考如下示例：

- **Vehicle**
- **Gnss**
- **Ego_tf**
- **Object_array_vision**
- **Tag_record**
- **Control**
- **Predicted_objects**
- **Planning_trajectory**
- **Routing_path**
- **Traffic_light_info**

Vehicle

```
vehicle_info {
    stamp_secs: 1604996332
    stamp_nsecs: 847945211
    autonomy_status: 0
    gear_value: 4
    vehicle_speed: 43.93000030517578
    steering_angle: 0.699999988079071
    yaw_rate: 0.0
    interior_temperature: 0.0
    outside_temperature: 0.0
    brake: 0.0
    timestamp: 1604996332847
    turn_left_light: 0
    turn_right_light: 0
    longitude_acc: -0.03125
    lateral_acc: 0.0
}
```

Gnss

```
gnss_points {
    stamp_secs: 1604996332
```



```
  stamp_nsecs: 855145358
  latitude: 30.18027687072754
  longitude: 120.21341705322266
  elevation: 7.392796039581299
  timestamp: 1604996332855
}
```

Ego_tf

```
localization_info {
  timestamp: 1604996332855
  stamp_secs: 1604996332
  stamp_nsecs: 855301408
  pose_position_x: 1165.5460205078125
  pose_position_y: -479.2198486328125
  pose_position_z: -1.48505699634552
  pose_orientation_x: 0.003883248195052147
  pose_orientation_y: -0.0031167068518698215
  pose_orientation_z: 0.7017714977264404
  pose_orientation_w: 0.7123847603797913
  pose_orientation_yaw: 1.5557808876037598
  velocity_linear: 12.21684455871582
  velocity_angular: 0.014540454372763634
  acceleration_linear: 0.23571151494979858
  acceleration_angular： 0.0
}
```

Object_array_vision

```
tracked_object {
  timestamp: 1604996332862
  stamp_secs: 1604996332
  stamp_nsecs: 862911489
  objects {
    id: 26175
    label: "Car"
    pose_position_x: 1154.59912109375
    pose_position_y: -496.5350646972656
    pose_position_z: -1.8222997188568115
    pose_orientation_z: 0.714431643486023
    pose_orientation_w: 0.6997052431106567
    pose_orientation_yaw: 1.5916229486465454
    dimensions_x: 4.513162136077881
    dimensions_y: 1.7747581005096436
    dimensions_z: 1.628068208694458
    speed_vector_linear_x: 0.012852923013269901
    speed_vector_linear_y: -9.972732543945312
    relative_position_x: -17.48011016845703
    relative_position_y: 10.685434341430664
    relative_position_z: -0.17673441767692566
  }
  objects {
    id: 26170
    label: "Pedestrian"
    pose_position_x: 1180.902099609375
    pose_position_y: -504.7625732421875
    pose_position_z: -1.3601081371307373
    pose_orientation_z: -0.7057344317436218
    pose_orientation_w: 0.7084764242172241
    pose_orientation_yaw: -1.5669186115264893
    dimensions_x: 0.7922295331954956
    dimensions_y: 0.7891787886619568
    dimensions_z: 1.6868246793746948
    speed_vector_linear_x: 0.13573257625102997
    speed_vector_linear_y: 1.5281875133514404
    relative_position_x: -25.306795120239258
    relative_position_y: -15.737456321716309
    relative_position_z: 0.39350399374961853
  }
```



```
objects {
  id: 26169
  label: "Pedestrian"
  pose_position_x: 1175.647216796875
  pose_position_y: -506.730712890625
  pose_position_z: -1.569373607635498
  pose_orientation_z: 0.6943609118461609
  pose_orientation_w: 0.7196269631385803
  pose_orientation_yaw: 1.5350627899169922
  dimensions_x: 0.8029457330703735
  dimensions_y: 0.7876891493797302
  dimensions_z: 1.6028095483779907
  speed_vector_linear_x: 0.06551000475883484
  speed_vector_linear_y: 0.0022428608499467373
  relative_position_x: -27.355571746826172
  relative_position_y: -10.512933731079102
  relative_position_z: 0.19844147562980652
}
objects {
  id: 26168
  label: "Pedestrian"
  pose_position_x: 1173.3189697265625
  pose_position_y: -507.2300109863281
  pose_position_z: -1.6026556491851807
  pose_orientation_z: 0.717462956905365
  pose_orientation_w: 0.6965966820716858
  pose_orientation_yaw: 1.600306749343872
  dimensions_x: 0.7922430038452148
  dimensions_y: 0.7811086177825928
  dimensions_z: 1.6341478824615479
  speed_vector_linear_x: -0.04817964881658554
  speed_vector_linear_y: -0.21502695977687836
  relative_position_x: -27.89008903503418
  relative_position_y: -8.192517280578613
  relative_position_z: 0.16775710880756378
}
objects {
  id: 26155
  label: "Bus"
  pose_position_x: 1172.106689453125
  pose_position_y: -478.5303039550781
  pose_position_z: -0.48812994360923767
  pose_orientation_z: -0.7203028798103333
  pose_orientation_w: 0.6936596632003784
  pose_orientation_yaw: -1.6084778308868408
  dimensions_x: 11.322981834411621
  dimensions_y: 2.9294095039367676
  dimensions_z: 3.1415622234344482
  speed_vector_linear_x: -0.017722932621836662
  speed_vector_linear_y: 0.1302066147327423
  relative_position_x: 0.7977913022041321
  relative_position_y: -6.548437118530273
  relative_position_z: 0.9966707229614258
}
objects {
  id: 26153
  label: "Bus"
  pose_position_x: 1148.1876220703125
  pose_position_y: -490.8350524902344
  pose_position_z: -0.954763650894165
  pose_orientation_z: 0.6907882690429688
  pose_orientation_w: 0.7230570912361145
  pose_orientation_yaw: 1.5251574516296387
  dimensions_x: 10.779899597167969
  dimensions_y: 2.856076717376709
  dimensions_z: 2.811084508895874
  speed_vector_linear_x: 0.03153659775853157
  speed_vector_linear_y: 0.23439916968345642
  relative_position_x: -11.868709564208984
```



```
    relative_position_y: 17.1827335357666
    relative_position_z: 0.6278138756752014
  }
  objects {
    id: 26141
    label: "Bus"
    pose_position_x: 1171.7779541015625
    pose_position_y: -512.5936889648438
    pose_position_z: -0.9443151354789734
    pose_orientation_z: -0.7186583876609802
    pose_orientation_w: 0.6953632831573486
    pose_orientation_yaw: -1.6037421226501465
    dimensions_x: 10.841312408447266
    dimensions_y: 2.9661808013916016
    dimensions_z: 3.2250704765319824
    speed_vector_linear_x: 0.0513402484357357
    speed_vector_linear_y: 0.006104861851781607
    relative_position_x: -33.26952362060547
    relative_position_y: -6.731308937072754
    relative_position_z: 0.8776476979255676
  }
  objects {
    id: 26133
    label: "Bus"
    pose_position_x: 1146.657958984375
    pose_position_y: -508.7508239746094
    pose_position_z: -0.883571445941925
    pose_orientation_z: 0.7007946968078613
    pose_orientation_w: 0.713362991809845
    pose_orientation_yaw: 1.5530219078063965
    dimensions_x: 12.186415672302246
    dimensions_y: 2.824420690536499
    dimensions_z: 3.292656183242798
    speed_vector_linear_x: 0.005901232361793518
    speed_vector_linear_y: 0.013970088213682175
    relative_position_x: -29.803848266601562
    relative_position_y: 18.443498611450195
    relative_position_z: 0.8749525547027588
  }
  objects {
    id: 26120
    label: "Bus"
    pose_position_x: 1170.993408203125
    pose_position_y: -525.5801391601562
    pose_position_z: -1.104852318763733
    pose_orientation_z: -0.7154129147529602
    pose_orientation_w: 0.6987019181251526
    pose_orientation_yaw: -1.5944297313690186
    dimensions_x: 10.749905586242676
    dimensions_y: 2.7170863151550293
    dimensions_z: 3.0421104431152344
    speed_vector_linear_x: 0.016746148467063904
    speed_vector_linear_y: -0.23609620332717896
    relative_position_x: -46.26727294921875
    relative_position_y: -6.141877174377441
    relative_position_z: 0.8449855446815491
  }
}
```


Tag_record


```
segments {
  scenario_id: 100000000
  source: "takeover"
  start: 1617336642300
  end: 1617336652300
}
segments {
  scenario_id: 100000000
```



```
  source: "vehicle"
  start: 1617336672300
  end: 1617336692300
}
```

Control

```
stamp_secs: 1617336640
stamp_nsecs: 795364700
timestamp: 1617336640795
acceleration: 0.7146586179733276
front_wheel_angle: 2.9919793605804443
```

Predicted_objects

```
stamp_secs: 1617336640
stamp_nsecs: 971891550
timestamp: 1617336640971
obstacle_info {
  obstacle_timestamp: 1617336640699
  id: 6711
  x: -123.08731842041016
  y: 486.83221435546875
  z: 0.575542688369751
  prediction_trajectory {
    path_point {
      x: -103.26817321777344
      y: 486.0815734863281
      theta: -0.007839304395020008
      v: 4.405668258666992
      relative_time: 4.5
    }
    path_point {
      x: -102.82765197753906
      y: 486.0737609863281
      theta: -0.00746726430952549
      v: 4.405668258666992
      relative_time: 4.599999904632568
    }
    ......
  }
}
obstacle_info {
  obstacle_timestamp: 1617336640699
  id: 6744
  x: -145.0320587158203
  y: 491.35015869140625
  z: -0.40381166338920593
  prediction_trajectory {
    path_point {
      x: -145.0320587158203
      y: 491.35015869140625
      theta: -2.9442124366760254
      v: 1.0038001537322998
    }
    path_point {
      x: -145.1304931640625
      y: 491.3304748535156
      theta: -2.9442124366760254
      v: 1.0038001537322998
      relative_time: 0.10000000149011612
    }
    ......
  }
}
obstacle_info {
  obstacle_timestamp: 1617336640699
  id: 6760
  x: -138.3047332763672
```



```
  y: 489.9286193847656
  z: -0.12651222944259644
}
```

Planning_trajectory

```
stamp_secs: 1617336640
stamp_nsecs: 809739351
timestamp: 1617336640809
trajectory_points {
  x: -151.27487182617188
  y: 486.55096435546875
  theta: 0.0023324606008827686
  kappa: -0.0017824547830969095
}
trajectory_points {
  x: -151.21182250976562
  y: 486.5510559082031
  theta: 0.0022713469807058573
  kappa: -0.0017127590253949165
}
......
```

Routing_path

```
timestamp: 1630057162125
stamp_secs: 1630057162
stamp_nsecs: 125769156
routing_path_info {
  id: 1
  path_point {
    x: -203.34230041503906
    y: 125.63516998291016
    z: -0.5
  }
  path_point {
    x: -203.34915161132812
    y: 125.72517395019531
    z: -0.5
  }......}
```

Traffic_light_info

```
timestamp: 1630057508000
stamp_secs: 1630057508
lights {
  id: 1
  color: 1
  location_x: -206.60186767578125
  location_y: 459.9820861816406
  location_z: 3.0
}
lights {
  id: 2
  color: 2
  location_x: -74.1282958984375
  location_y: 484.984619140625
  location_z: 4.0
}
lights {
  id: 3
  color: 3
  location_x: 59.96036911010742
  location_y: 473.6038513183594
  location_z: 5.0
}
```


4.3 数据处理

4.3.1 作业总览

在作业总览中平台支持创建数据标记，数据图表、数据回放、数据集等作业，数据包选择相对应的算子就可以触发相对应的作业。

说明

1. 数据回放类型的算子作业不需要选择输出仓库。
2. 当选择数据集算子时，数据类型可支持选择通用存储和数据场景。

创建算子作业

步骤1 在左侧菜单栏中，选择“数据处理 > 数据处理”。

步骤2 选择“作业总览”页签，单击“创建作业”，填写算子作业信息。

表 4-15 算子作业

参数	示例	说明
处理算子	选择算子	根据需要选择已创建的算子。
资源规格	选择资源	当前项目中可用的资源规格，资源配置需要平台管理员在集群纳管中创建，支持选择带有GPU的资源规格。
优先级	0	设定任务的优先级，数值取[-50,50]的整数，数字越大，优先级越高。
环境变量	-	配置算子的环境变量。允许添加的环境变量个数不超过10个。 ● Key：只能由英文、数字、和特殊符号（ ,-_ ）组成，且需要以字母开头 。长度不超过64个字符。 ● Value：只能由英文、数字和特殊符号（ \/,.[]-_ ）组成 。长度不超过64个字符。
超时时间	24	作业运行时长超过该时间后，作业会自动停止。 如果该值为空，默认超时时间为24小时，或设置为[1，720]的整数。
数据类型	数据包	选择输入数据类型，通用存储、数据包或数据场景。
选择数据	选择数据包	选择需要操作的数据包中的数据。 当选择的算子的输出类型为“数据集”时，可以选择最多50个数据包合并处理。勾选合并处理后，多个数据包会在同一个作业里面进行处理，不勾选则每个数据包单独创建作业。
输出目录	选择输出目录	选择可输出的通用存储的目录。当选择的算子的输出类型为“数据回放”时不需要设置该项。

步骤3 单击“创建”， 在作业总览页面可查看创建好的作业。

----结束

数据处理作业相关操作

- 在“作业总览”页签，针对算子作业还可以完成以下操作。

表 4-16 作业总览相关操作

任务	操作步骤
作业详情	单击操作栏中的“详情”，可查看作业的详情和日志信息。当作业出现异常时，请参见表4-17分析定位问题。
重启作业	单击操作栏中的“重启”，可根据需要保持或修改资源规格（默认显示当前资源规格），将目标作业进行重启。 **说明** 仅支持作业状态为“失败”、“停止”、“运行异常”和“上传失败”的作业。
停止作业	• 选择单个作业，选择操作栏的“更多 > 停止”，停止单个作业。 • 勾选多个作业，单击列表上方的“停止”，可批量停止作业。 **说明** 当作业状态为过程态（排队中、启动中、运行中）时，可以停止作业。
删除作业	• 选择单个作业，选择操作栏的“更多 > 删除”，删除单个作业。 • 勾选多个作业，单击列表上方的“删除”，可删除多个作业。 **说明** 当作业状态为终止态（提交失败、已停止、已完成、运行异常）时，可以删除作业。
重新上传作业	选择操作栏中的“更多 > 重新上传”，可重新上传作业。 **说明** 仅支持作业状态为“上传失败”的作业。
搜索作业	可根据“作业ID”、“数据ID”、“算子”、“创建时间”、“输出类型”或“状态”搜索作业。

表 4-17 作业异常提示信息和处理方法

数据作业异常提示信息	处理方法
输入文件不存在	请检查输入数据是否存在
创建容器配置失败	请尝试重启作业
镜像拉取失败	请检查镜像是否已上传
无效镜像	请检查镜像是否正确
容器启动异常	请尝试重启作业
创建容器失败	请尝试重启作业

数据作业异常提示信息	处理方法
未选择数据仓库	请重新创建作业并指定数据仓库
内存或磁盘空间不足	请检查资源规格或作业数据量
数据包不存在	请检查数据包是否存在
资源不可用，请检查资源规格	请检查资源规格是否可用
场景片段不存在或源文件不存在	请选择其他场景片段
数据来源与算子类型不匹配	检查算子类型和输入数据类型是否匹配
镜像不存在	请检查镜像是否已上传
pvc不存在	请检查PVC是否已创建
通知导入模块失败	请尝试重新上传或者联系运维人员
导入作业结果超时	请尝试重新上传或者联系运维人员
导入作业结果失败	请尝试重新上传或者联系运维人员
算子不存在	请检查算子是否存在
预处理失败	请尝试重启作业
获取实例资源异常	请检查运维配置是否正确
资源需求超出节点总量	请检查作业资源规格和节点资源总量
标签或自定义属性无效	请检查算子输出的标签或自定义属性是否正确
数据仓库ES操作异常	请尝试重新上传或者联系运维人员
标签或属性数量超过上限	请减少算子输出的标签或自定义属性数量
数据仓库服务重启，导入信息丢失	请尝试重新上传或者联系运维人员
作业运行超时	请优化算子逻辑或者增加作业资源规格
作业运行失败	请查看作业日志或者检查算子逻辑是否正确
内部异常	请查看日志内容或者检查CCE作业是否正常
其他情况	请查看日志内容或者检查CCE作业是否正常

- 在“作业总览”页签，针对内部作业还可以完成以下操作。

表 4-18 作业总览相关操作

任务	操作步骤
作业详情	单击操作栏中的“详情”，可查看作业的详情和作业日志。
删除作业	• 选择单个作业，单击操作栏的“删除”，删除单个作业。 • 勾选多个作业，单击列表上方的“删除”，可删除多个作业。 **说明** 当作业状态为终止态（提交失败、已停止、已完成、运行异常）时，可以删除作业。
搜索作业	可根据“作业ID”、“数据ID”、“作业类型”、“创建时间”或“状态”搜索作业。

4.3.2 作业队列

平台为算子作业、内部作业和脱敏作业提供作业队列的功能，用户可在此查看任务队列，同时支持对任务优先级的调整。

说明

其他租户的队列作业不允许操作。

步骤1 在左侧菜单栏中单击“数据服务 > 数据处理”。

步骤2 选择“作业队列”页签，可根据作业类型（算子作业或内部作业）查看作业。

----结束

数据处理作业队列相关操作

在“作业队列”页签，还可以进行以下操作。

表 4-19 作业队列相关操作

任务	操作步骤
置顶作业	单击操作栏中的“置顶”，即可将作业调整至队列中最高优先级。
置底作业	单击操作栏中的“置底”，即可将作业调整至队列中最低优先级。
上移作业	单击操作栏中的“上移”，即可将作业调整至队列中上一级。
下移作业	单击操作栏中的“下移”，即可将作业调整至队列中下一级。
更新作业优先级	1. 勾选单个或多个作业。 2. 单击“更新优先级”，输入[-50,50]的整数，数字越大，优先级越高。 3. 单击“确定”，优先级更改成功。相同优先级的任务，作业队列更新时间越早，优先级越高。
搜索作业	在搜索输入框中输入“作业ID”，按回车键即可查询。

4.3.3 算子管理

平台支持自定义创建不同类型的算子，用于不同的数据处理作业。

新建算子

步骤1 在左侧菜单栏中，单击“数据处理 > 数据处理”。

步骤2 选择“算子管理”页签，单击“新建算子”。

表 4-20 新建算子参数

参数	说明
名称	算子名称，不得超过64个字符。支持中英文、数字、“-”、“_”，不支持特殊字符。
描述	算子内容、用途等的简要描述，不包含“@#$%^&*< > \”，不得超过256个字符。
可见范围	支持私有或团队。 ● 私有：该模板只有创建者可操作，其他用户不可见。 ● 团队：该模板当前工作空间下被授权的用户均可见。
运行镜像	可选择镜像仓库中已创建好的镜像。
镜像版本	选择镜像版本。
启动命令	镜像的启动命令，具体命令根据镜像启动脚本确定，例如“python3 main.py”。可以参考4.3.4 算子示例中各算子的启动命令。
输出类型	可选择“数据标记”，“数据集”，“数据回放”，“回放仿真”，“数据图表”和“数据脱敏”。 数据标记：用于场景挖掘作业（Rosbag to场景片段）。 数据回放：输出结果是OpenData格式（Rosbag to OpenData）。 数据集：用于数据抽取作业，输出至通用存储（Rosbag to Dataset）。 回放仿真：用于Resim运行，对比回放（感知规划算法）。 数据图表：用于数据回放时，数据图表展示。 具体规范示例请参考“算子示例”。

步骤3 单击“确定”。

在“算子”列表，可查看新创建的算子。

----结束

算子管理相关操作

在“算子管理”页签，还可以完成以下操作。

表 4-21 算子管理相关操作

任务	操作步骤
查看算子详情	单击操作栏的“详情”，可查看算子的详情
编辑算子	单击操作栏的“编辑”，可编辑算子的详情。
删除算子	单击操作栏的“删除”，可删除算子的详情。
搜索算子	选择“算子ID”或“算子名称”，在搜索输入框中输入搜索条件，按回车键即可查询。

4.3.4 算子示例

4.3.4.1 Rosbag 转 OpenData 作业（数据回放）

作业输入输出规范

用户完成自定义Rosbag转OpenData算子创建，运行作业容器时Octopus平台向其中注入以下环境变量：

- rosbag_path: 作为数据源的rosbag存放路径，例如/tmp/data/20220620.bag
- yaml_path: 启动数据收集任务的yaml文件路径，例如/tmp/Octopus_data_collections.yaml
- output_dir: rosbag数据包作业运行结果输出目录，例如/tmp/output
- tmp_dir: 供用户存储作业临时文件的目录，例如/tmp/workspace

用户的作业容器需要解析rosbag，并将转换结果输出到output目录，结果示例如下：

图 4-2 output 目录

```
|--- /tmp/output                                # output_dir
    |--- opendata_to_platform.yaml              # 必需
    |--- _SUCCESS                               # 必需
    |--- gps                                    # sensor 1
        |--- gnss.pb
    |--- camera_01                              # sensor 2
        |--- 1655696184000.jpg
        |--- 1655696184100.jpg
        ...
    |--- camera_02                              # sensor 3
        |--- 1655696184200.jpg
        |--- 1655696184300.jpg
        ...
    |--- pandar                                 # sensor 4
        |--- 1655696184400.pcd
        |--- 1655696184500.pcd
        ...
```

每个传感器提取的数据保存在单独的文件夹，其中camera和lidar传感器提取的样本文件必须以时间戳命名。任务结束标志文件，_SUCEESS或_FAILURE分别代表任务成功或失败。opendata_to_platform.yaml文件以yaml格式记录该OpenData数据包的元数据，格式如下：

图 4-3 文件格式

```
folders:                        # 必填
- folder: string                # 必填 输出文件夹下的目录名，例如：camera_01
  sensor_type: stirng           # 必填 该目录中数据对应的传感器类型
  calibration_item_id: long     # 非必填 传感器对应的标定项id
```

sensor_type字段标识传感器类型，可取以下值：camera、lidar、gnss、vehicle、ego_tf、object_array_vision、traffic_light_matched、tag_record、planning_trajectory、predicted_objects、control、routing_path、localization_visualization。具体定义参考**4.2.3 数据包格式**。

图 4-4 示例 opendata_to_platform.yaml 文件内容

```yaml
folders:
- folder: camera_01
  sensor_type: camera
  calibration_item_id: 1
- folder: camera_02
  sensor_type: camera
  calibration_item_id: 2
- folder: pandar
  sensor_type: lidar
  calibration_item_id: 3
- folder: gps
  sensor_type: gnss
```

示例代码

以下为主程序文件ros2opendata.py中截取的代码片段，分别运行不同的功能，详见注释。

图 4-5 运行前准备

```python
# 导入依赖模块
import os
import rosbag
from argparse import ArgumentParser

bag = rosbag.Bag(os.getenv('rosbag_path'), 'r')  # 通过环境变量rosbag_path获取数据包路径
output_dir = os.getenv('output_dir')  # 通过环境变量output_dir获取输出目录

# 定义入参，例如：通过启动命令指定激光雷达的标定项id
parser = ArgumentParser()
parser.add_argument('--lidar_calibration_id', type=int)
params = parser.parse_args()
```

图 4-6 解析点云消息

```python
# 导入解析点云的依赖
import pcl
from sensor_msgs import point_cloud2 as pc2

os.mkdir(os.path.join(output_dir, 'pandar'))  # 在output_dir创建pandar文件夹
for topic, msg, t in bag.read_messages(topics=['/pandar']):  # 根据topic名筛选点云消息
    # 根据时间戳确定文件名
    file_path = os.path.join(output_dir, 'pandar',
                             '{}.pcd'.format(int(msg.header.stamp.secs * 1000 + msg.header.stamp.nsecs / 1000000)))
    # 读取点云数据
    pc_list = list(pc2.read_points(msg, skip_nans=True, field_names=('x', 'y', 'z', 'intensity')))
    pcd = pcl.PointCloud_PointXYZI()
    pcd.from_list(pc_list)
    # 保存为pcd文件
    pcl.save(pcd, file_path, format='pcd', binary=True)
```

图 4-7 解析 gnss 消息

```
from proto.octopusdata_gnss_pb2 import GnssPoint, GnssPoints  # 导入解析gnss的依赖

os.mkdir(os.path.join(output_dir, 'gps'))  # 在output_dir创建gps文件夹
gnss_list = []
for topic, msg, t in bag.read_messages(topics=['/inspvax']):  # 根据topic名筛选gnss消息
    # 从message读出数据，放入pb字段
    pb_loc_gnss = GnssPoint()
    pb_loc_gnss.stamp_secs = msg.header.stamp.secs
    pb_loc_gnss.stamp_nsecs = msg.header.stamp.nsecs
    pb_loc_gnss.timestamp = int(msg.header.stamp.secs * 1000 + msg.header.stamp.nsecs / 1000000)
    pb_loc_gnss.latitude = msg.latitude
    pb_loc_gnss.longitude = msg.longitude
    pb_loc_gnss.elevation = msg.altitude
    gnss_list.append(pb_loc_gnss)

# 写入pb文件
gn_pb_out = GnssPoints()
gn_pb_out.gnss_points.extend(gnss_list)
with open(os.path.join(output_dir, 'gps', 'gnss.pb'), 'wb') as g:
    g.write(gn_pb_out.SerializeToString())
```

图 4-8 写 opendata_to_platform.yaml 文件

```
import yaml  # 导入yaml模块

collect_yaml = {
    'folders': [
        # 填入激光雷达标定，来自启动命令入参
        {'folder': 'pandar', 'sensor_type': 'lidar', 'calibration_item_id': params.lidar_calibration_id},
        {'folder': 'gps', 'sensor_type': 'gnss'},
    ]
}

# yaml写入输出目录
with open(os.path.join(output_dir, 'opendata_to_platform.yaml'), 'w', encoding='utf-8') as f:
    yaml.safe_dump(collect_yaml, f)
```

构建镜像

Octopus平台依赖算子镜像内的/bin/bash、stdbuf、tee软件，请确保基础镜像内包含上述软件且能通过PATH找到。

1. Dockerfile示例

图 4-9 示例

```
# 基于一个自带ros环境的镜像
FROM ros:noetic
# 把主程序文件拷贝进镜像
COPY ros2opendata.py /home/main
# 把其他需要的库文件放入镜像
COPY utils/* /home/main/libs
# 主程序文件配置可运行权限，安装需要的三方件
RUN chmod u+x /home/main/ros2opendata.py && \
    apt install python3-pip libpcl-dev && \
    pip3 install python-pcl
# 配置PYTHONPATH环境变量，使/home/main/libs内库文件可被引用
ENV PYTHONPATH="${PYTHONPATH}:/home/main/libs"
# 配置运行用户
USER root
```

启动命令：

```
python3 /home/main/ros2opendata.py --lidar_calibration_id 5
```

2. 构建镜像

   运行命令：

   ```
   docker build -f dockerfile -t rosbag2opendata:0.1 .
   ```

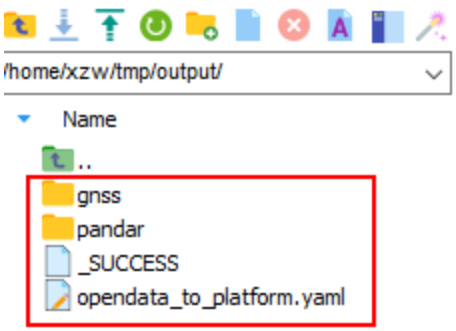
3. 本地调试

   准备一个待处理的rosbag，如~/data/20220620.bag，一个示例Octopus_data_collections文件，如~/data/Octopus_data_collections.yaml运行如下命令（基于上述示例镜像）：

   ```
   docker run -v ${HOME}/data/20220620.bag:/tmp/data/20220620.bag -v ${HOME}/tmp/output:/tmp/output -v ${HOME}/data/Octopus_data_collections.yaml:/tmp/Octopus_data_collections.yaml --env output_dir=/tmp/output --env rosbag_path=/tmp/data/20220620.bag --env yaml_path=/tmp/Octopus_data_collections.yaml --env tmp_dir=/tmp/workspace rosbag2opendata:0.1 /bin/sh -c “/home/main/ros2opendata.py --lidar_calibration_id 5”
   ```

   完成后在${HOME}/tmp/output目录查看运行结果文件：

图 4-10 运行结果



4.3.4.2 场景挖掘作业（数据标记）

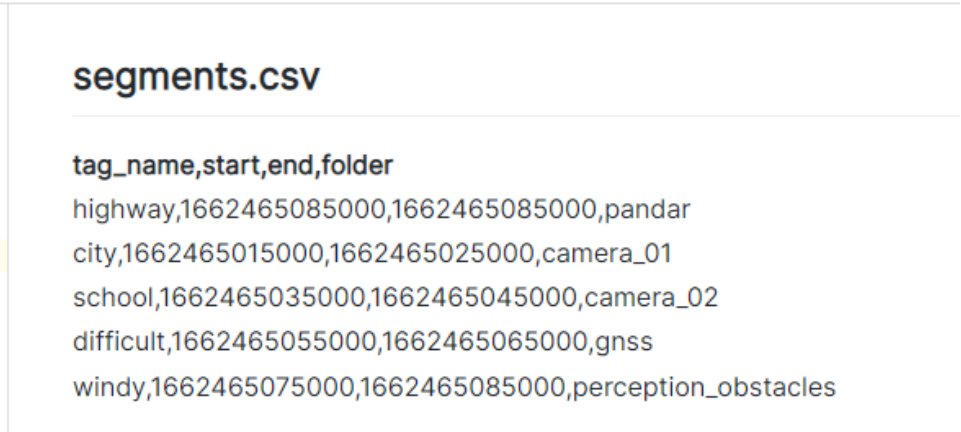
作业输入输出规范

用户完成自定义场景挖掘镜像上传后，在运行作业容器时，Octopus平台会向作业容器中注入以下环境变量：

- rosbag_path: 作为数据源的rosbag存放路径，例如/tmp/data/20220620.bag
- output_dir: 场景挖掘作业运行结果输出目录，例如/tmp/output

用户的作业容器的作用是解析Rosbag，并将场景挖掘结果以csv格式输出到output_dir指定的目录，并以“segments.csv”作为文件名，文件完整路径示例：/tmp/output/segments.csv。

图 4-11 csv 文件内容示例



Csv文件表头固定为“tag_name,start,end”，指定该表的四列数据分别为“标签名”、“开始时间戳”、“结束时间戳”。

- tag_name：标签名对应Octopus平台标签管理模块内的标签，没有预先创建的标签会自动创建。
- start,end：开始和结束时间戳指定该打标片段的时间范围。
- folder：目录名指示OpenData数据包内的特定数据目录，对应为某个传感器。如果需要在相同时间片段上对多个传感器打标，需要为每个传感器输出一行打标信息。

输出结果完成后，作业容器需要在output目录创建一个名为“_SUCCESS”的标识文件，用于通知系统作业已完成。如果作业主动捕获到异常并失败退出，可在output目录创建一个名为“_FAILURE”的标识文件，用于通知系统作业已失败。

运行完成并上传的挖掘结果可以在“数据场景”模块进行检索、查看。选中一个场景片段后可以通过“回放”按键跳转到数据包的对应时间点进行回放，长度在10秒到99秒之间的场景片段可以生成仿真场景。选中片段后的预览样本图根据挖掘片段所对应的传感器类型选择相应的传感器样本来进行展示。

示例代码

代码文件命名为ros_hard_mining.py。

作业输入输出规范示例代码如下图所示：

图 4-12 示例代码

```
import os
import time
import random
from argparse import ArgumentParser
new *
def run_main():
    params = parse_params()
    tags = params.tags.split(',')
    start = int(params.time_range.split(',')[0])
    end = int(params.time_range.split(',')[1])
    sensors = params.sensors.split(',')
    with open(os.path.join(os.getenv('output_dir'), 'segments.csv'), 'w') as f:  从环境变量读取输出文件夹，并创建输出文件segments.csv
        f.write('tag_name,start,end,folder\n')  写入csv header
        for i in range(int(params.number)):
            t1, t2 = random.randint(start, end), random.randint(start, end)
            f.write(f'{random.sample(tags, 1)[0]},{min(t1, t2)},{max(t1, t2)},{random.sample(sensors, 1)[0]}\n')
            print(f'{random.sample(tags, 1)[0]},{min(t1, t2)},{max(t1, t2)},{random.sample(sensors, 1)[0]}')
            time.sleep(2)
    with open(os.path.join(os.getenv('output_dir'), '_SUCCESS'), 'w') as f:
        f.write("")  输出成功标识文件
new *
def parse_params():
    parser = ArgumentParser()
    parser.add_argument('--time_range', default='111111,222222')
    parser.add_argument('--sensors', default='pandar,camera_03encode,localization_localization_info,'
                                             'prediction_prediction_obstacles,holo_ControlCommand,pandar_pic,tagrecord,'
                                             'inspvax,planning_trajectory,holo_VehicleInfoMagotan,'
                                             'perception_perception_obstacles')
    parser.add_argument('--number', default='10')
    parser.add_argument('--tags', default='aaa,bbb,ccc')
    return parser.parse_args()

if __name__ == "__main__":  启动挖掘程序
    run_main()
```

写入难例记录，每行包括标签名、开始时间、结束时间、所属目录名，字段以逗号分隔

同时输出到标准流，可通过作业日志查看

构建镜像

Octopus平台依赖算子镜像内的/bin/bash、stdbuf、tee软件，请确保基础镜像内包含上述软件且能通过PATH找到。

1. Dockerfile示例。

   ```
   FROM ros:noetic
   COPY ros_hard_mining.py /home/main/
   # 算法启动示例:
   # python3 /home/main/ros_hard_mining.py --tags tag1,tag2 --time_range
   1673231275000,1673261275000 --sensors camera_encoded_1,pandar --number 10
   USER root
   ```

2. 构建镜像。

   运行命令：

   ```
   docker build -f Dockerfile -t ros-hard-mining:0.1
   ```

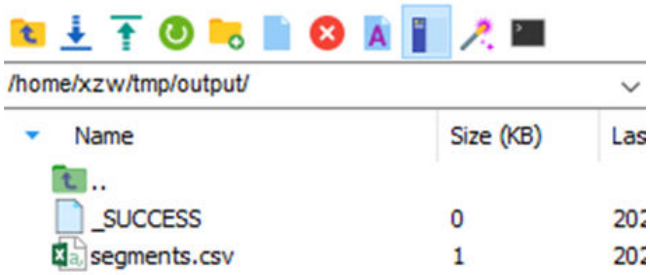
3. 本地调试。

   运行如下命令（基于上述示例镜像）：

   ```
   docker run -v ${HOME}/tmp/output:/tmp/output --env output_dir=/tmp/output --env
   tmp_dir=/tmp/workspace ros-hard-mining:0.1 /bin/sh -c “python3
   /home/main/ros_hard_mining.py --tags tag1,tag2 --time_range 1673231275000,1673261275000 --
   sensors camera_encoded_1,pandar --number 10”
   ```

   完成后在${HOME}/tmp/output目录查看运行结果文件：

图 4-13 运行结果



4.3.4.3 数据提取作业（数据集）

作业输入输出规范

- Input

  平台会以环境变量的形式提供以下参数：

  - rosbag_path： 数据包路径，如果输入数据源是多个，以','分隔。示例：/tmp/DPK0000001/packageName1,/tmp/DPK0000002/packageName2。
  - output_dir：最终输出数据集路径。
  - tmp_dir：供用户存储临时文件的目录。
  - task_content_json：人工打标需要的标签信息，详细参考“人工打标支持”。（如果没有人工打标，不需要此参数）。

- Output

  对于output_dir的格式限定如下：

  - 必须有output_dir/dataset目录，存储数据集文件。数据集文件有格式要求（Octopus、PascalVOC...）。
  - 必须有_SUCCESS或 _FAILURE文件，标志用户镜像任务完成（或失败）。
  - 可选有output_dir/auxiliary目录，用于存储附加文件。对格式没有限制。

  图 4-14 output_dir 的格式

```
|--- output_dir
    |--- _SUCCESS
    |--- dataset
        |--- version-1
            |--- data-info
                |--- 12345678
                    |--- 12345678.pcd
                    |--- 12345678.json
                |--- 12345679
                    |--- 12345679.pcd
                    |--- 12345679.json
                ...
    |--- auxiliary
        |--- video
            |--- video.mp4
        |--- raw-data
            |--- 12345678
            |--- 12345679
            |--- 12345680
        ...
```

- 人工打标支持

  Input

程序必须能接受task_content_json，该参数以环境变量的形式引入镜像。
task_content_json格式为List of Tags。
如果没有tag，为空list。如果只有一个标签，list内只有一个Tag。
如果start与end一致，代表单帧打标。

图 4-15 每个 Tag 格式示例

```
{
  "start": int,       // 起始时间戳，inclusive
  "end": int,         // 结束时间戳，inclusive
  "sensor": string    // 传感器信息
}
```

图 4-16 task_content_json 示例

```
[
  {
    // 单帧打标
    "start": 123123123,
    "end": 123123123,
    "sensor": "camera_05H265"
  },
  {
    // 多帧打标
    "start": 1000000000,
    "end": 9999999999,
    "sensor": "camera_05H265"
  }
]
```

示例代码

作业输入输出规范示例代码如下图所示：

图 4-17 示例 1

```
import os
from argparse import ArgumentParser

import av
import pcl
import rosbag
from sensor_msgs import point_cloud2 as pc2

codec_ctx = av.codec.Codec('hevc', 'r')
h265_code = codec_ctx.create()


def main():
    params = parse_params()

    output_dir = os.getenv('output_dir')
    dataset_dir = os.path.join(output_dir, 'dataset')
    os.makedirs(dataset_dir)

    parse_method = parse_pcd if params.type == 'pcd' else parse_img

    bag = rosbag.Bag(os.getenv('rosbag_path'), 'r')
    for topic, msg, t in bag.read_message(topics=params.topic):
        parse_method(msg, dataset_dir)

    with open(os.path.join(os.getenv('output_dir'), '_SUCCESS'), 'w') as f:
        f.write("")


def parse_pcd(msg, dataset_dir):
    timestamp = int(msg.header.stamp.secs * 1000 + msg.header.stamp.nsecs / 1000000)
    os.makedirs(os.path.join(dataset_dir, str(timestamp)))
    file_path = os.path.join(dataset_dir, str(timestamp), '{}.pcd'.format(timestamp))
    pc_list = list(pc2.read_points(msg, skip_nans=True,
                                   field_names=('x', 'y', 'z', 'intensity')))
    out = pcl.PointCloud_PointXYZI()
    out.from_list(pc_list)
    pcl.save(out, file_path, format='pcd', binary=True)
```

导入需要的类库

解析入参

从环境变量读取输出目录
创建数据集目录

根据算法入参决定使用图片解析函数还是点云解析函数

读取rosbag，解析message

输出成功标识文件

点云解析函数：创建独立文件夹->msg解析成pcd格式->pcd文件存入独立文件夹

图 4-18 示例 2

```
def parse_img(msg, dataset_dir):
    timestamp = int(msg.header.stamp.secs * 1000 + msg.header.stamp.nsecs / 1000000)
    os.makedirs(os.path.join(dataset_dir, str(timestamp)))
    file_path = os.path.join(dataset_dir, str(timestamp), '{}.jpg'.format(timestamp))
    packet = av.packet.Packet(msg.raw_data)
    try:
        out = h265_code.decode(packet)
        for frame in out:
            if frame.format.name != 'rgb24':
                frame = frame.reformat(format='rgb24')
            img = frame.to_image()
            img.save(file_path)
    except Exception as e:
        print("{} frame can not trans to jpg".format(timestamp), e)


def parse_params():
    parser = ArgumentParser()
    parser.add_argument('--topic', default='')
    parser.add_argument('--type', default='pcd')
    return parser.parse_args()


if __name__ == "__main__":
    main()
```

图片解析函数：创建独立文件夹->把msg解析成jpg格式->jpg文件存入独立文件夹

算法入参：topic指定rosbag内的一路topic作为解析对象；type指定输出是点云或图片，可填pcd或img

程序入口

构建镜像

Octopus平台依赖算子镜像内的/bin/bash、stdbuf、tee软件，请确保基础镜像内包含上述软件且能通过PATH找到。

1. Dockerfile示例。

```
FROM ros:noetic
COPY ros_to_dataset.py /home/main/
# 算法启动示例:
# python3 /home/main/ros_to_dataset.py --topic pandar --type pcd
RUN apt install ros-noetic-cv-bridge python3-pcl libpcl-dev
USER root
```

2. 构建镜像。

   运行命令:

```
docker build -f Dockerfile -t ros-to-dataset:0.1
```

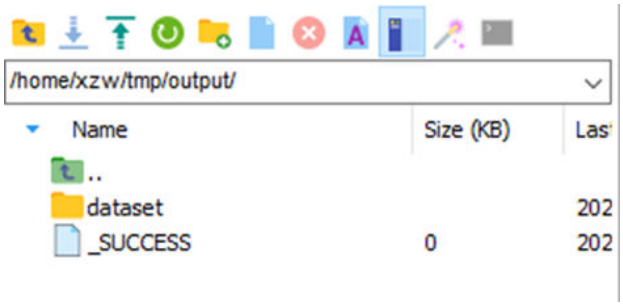
3. 本地调试。

   准备一个待处理的rosbag，如~/data/20220620.bag，运行如下命令（基于上述示例镜像）:

```
docker run -v ${HOME}/data/20220620.bag:/tmp/data/20220620.bag -v
${HOME}/tmp/output:/tmp/output --env output_dir=/tmp/output --env
rosbag_path=/tmp/data/20220620.bag --env tmp_dir=/tmp/workspace ros-to-dataset:0.1 /bin/sh -c
“python3 /home/main/ros_to_dataset.py --topic pandar --type pcd”
```

   完成后在${HOME}/tmp/output目录查看运行结果文件:

图 4-19 运行结果



4.3.4.4 Resim 作业（回放仿真）

作业输入输出规范

运行resim容器时，Octopus平台向容器中注入以下环境变量:

- rosbag_path: 作为输入的rosbag存放路径，例如/tmp/data/20220620.bag
- output_dir: resim作业的运行结果输出的目录，例如/tmp/output
- tmp_dir: 供resim作业存放临时文件的目录，例如/tmp/workspace

用户的resim作业需要输出的文件类似如下结构:

```
|--- /tmp/output                    环境变量output_dir指定的输出目录
  |--- opendata_to_platform.yaml     输出描述文件，详情见下文
  |--- _SUCCESS                      作业完成后输出的标识文件，内容可为空
  |--- planning                      以下各文件夹为resim算法输出内容，按照topic分隔，命名可自定义，与
opendata_to_platform.yaml内一致即可，尽量用常见字符
    |--- planning.pb
  |--- perception
    |--- perception.pb
  |--- prediction
    |--- prediction.pb
  |--- ...
```

opendata_to_platform.yaml内容描述输出的文件夹内的输出内容分别对应哪种数据类型，内容示例如下:

```
folders:
 - folder: planning                  # 与实际输出的目录名一致
   sensor_type: planning_trajectory    # 路径规划类型
- folder: perception                  # 与实际输出的目录名一致
   sensor_type: object_array_vision    # 感知类型
- folder: prediction                  # 与实际输出的目录名一致
   sensor_type: predicted_objects      # 感知物预测类型
```

当前resim作业支持的输出类型为planning_trajectory 、object_array_vision 、predicted_objects 3种，Proto格式定义分别如下：

图 4-20 planning_trajectory 格式定义

```
// planning_trajectory
syntax = "proto3";

package octopusdata;

message TrajectoryPoint {
    float x = 1;
    float y = 2;
    float z = 3;
    float theta = 4;
    float kappa = 5;
    int32 lane_id=6;
    float v=7;
    float a=8;
    float relative_time=9;
}

message Trajectory {
    uint64 stamp_secs = 1;
    uint64 stamp_nsecs = 2;
    uint64 timestamp = 3; //3d
    float total_path_length = 4;
    float total_path_time=5;
    int32 gear=6;
    int32 trajectory_type=7;
    int32 vehicle_signal=8;
    repeated TrajectoryPoint trajectory_points = 9;
}

message PlanTrajectory {
    repeated Trajectory trajectory_info= 1;
}
```

图 4-21 object_array_vision 格式定义

```
// predicted_objects
syntax = "proto3";
package octopusdata;
message PathPoint {
    float x = 1;
    float y = 2;
    float z = 3;
    float theta = 4;
    float kappa = 5;
    int32 lane_id= 6;
    float v=7;
    float a=8;
    float relative_time=9;
}
message PredictionTrajectory {
   repeated PathPoint path_point = 1;
}
message Obstacle {
    uint64 obstacle_timestamp = 1; //3d
    int32 id=2;
    float x = 3;
    float y = 4;
    float z = 5;
    repeated PredictionTrajectory prediction_trajectory = 6;
}
message PerceptionObstacle {
   uint64 stamp_secs = 1;
   uint64 stamp_nsecs = 2;
   uint64 timestamp = 3; //3d
   repeated Obstacle obstacle_info= 4;
}
message PredictionObstacles {
   repeated PerceptionObstacle perception_obstacle= 4;
}
```

图 4-22 predicted_objects 格式定义

```
// object_array_vision
syntax = "proto3";

package octopusdata;

message ObjectPoint {
    uint64 stamp_secs = 1;
    uint64 stamp_nsecs = 2;
    uint64 id = 3;
    uint64 classification = 4;
    float x_position = 5;
    float y_position = 6;
    float z_position = 7;
    float orientation = 8;
    float absolute_velocity_twist_linear_x = 9;
    float absolute_velocity_twist_linear_y = 10;
}

message ObjectList {
    uint64 stamp_secs = 1;
    uint64 stamp_nsecs = 2;
    repeated ObjectPoint object_list = 3;
}

message ObjectVisions {
    repeated ObjectList object_visions = 1;
}
```

构建镜像

Octopus平台依赖算子镜像内的/bin/bash、stdbuf、tee软件，请确保基础镜像内包含上述软件且能通过PATH找到。

1. Dockerfile示例。

图 4-23 Dockerfile 示例

启动命令：

```
bash /home/Octopus/run.sh
```

镜像构建：

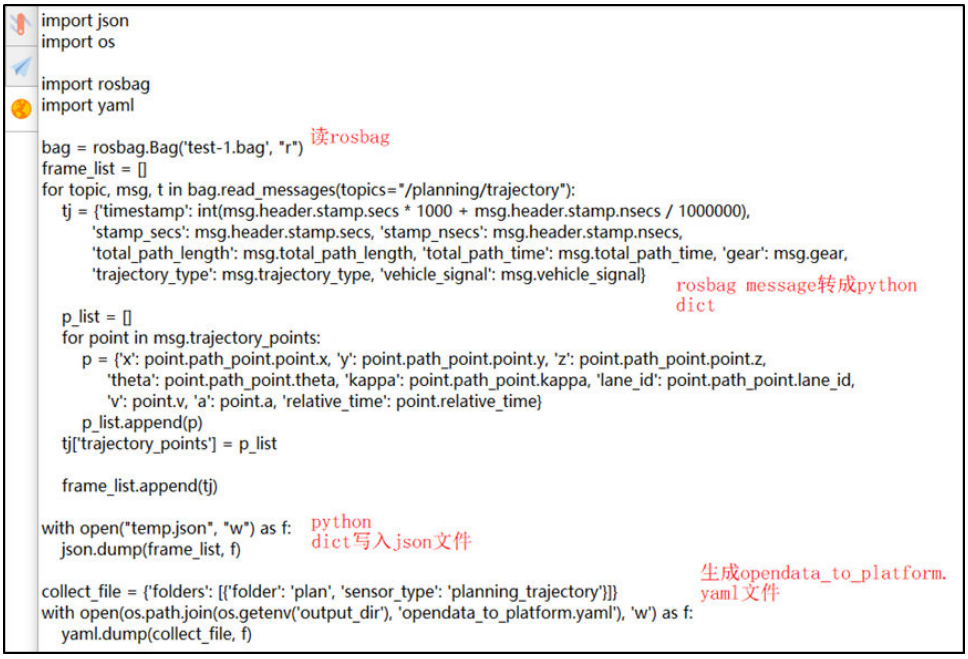
```
docker build -f Dockerfile -t guikong:0.1
```

2. 容器启动脚本示例。

图 4-24 启动脚本示例

3. Resim结果转json示例

图 4-25 Resim 结果转 json 示例



4. Json转pb文件示例

图 4-26 Json 转 pb 文件示例

```
from octopusdata_plantrajectory_pb2 import Trajectory, TrajectoryPoint, PlanTrajectory   导入pb模板

with open("temp.json", "r") as f:   读json文件
    frames = json.load(f)

frame_list = []
for frame in frames:
    tj = Trajectory()
    tj.timestamp = frame['timestamp']
    tj.stamp_secs = frame['stamp_secs']
    tj.stamp_nsecs = frame['stamp_nsecs']
    tj.total_path_length = frame['total_path_length']
    tj.total_path_time = frame['total_path_time']
    tj.gear = frame['gear']
    tj.trajectory_type = frame['trajectory_type']
    tj.vehicle_signal = frame['vehicle_signal']

    p_list = []                                   使用json内容生成pb文件
    for point in frame['trajectory_points']:
        p = TrajectoryPoint()
        p.x = point['x']
        p.y = point['y']
        p.z = point['z']
        p.theta = point['theta']
        p.kappa = point['kappa']
        p.lane_id = point['lane_id']
        p.v = point['v']
        p.a = point['a']
        p.relative_time = point['relative_time']
        p_list.append(p)
    tj.trajectory_points.extend(p_list)

    frame_list.append(tj)

plan_traj_pb_out = PlanTrajectory()
plan_traj_pb_out.trajectory_info.extend(frame_list)

os.makedirs(os.path.join(os.getenv('output_dir'), 'plan'))
# with open("test.pb", "wb") as p:                                    pb文件写入output_dir
with open(os.path.join(os.getenv('output_dir'), 'plan', 'plan.pb'), "wb") as p:
    p.write(plan_traj_pb_out.SerializeToString())

with open(os.path.join(os.getenv('output_dir'), '_SUCCESS'), 'w') as s:   输出_SUCCESS标识文件
    s.write('')
```

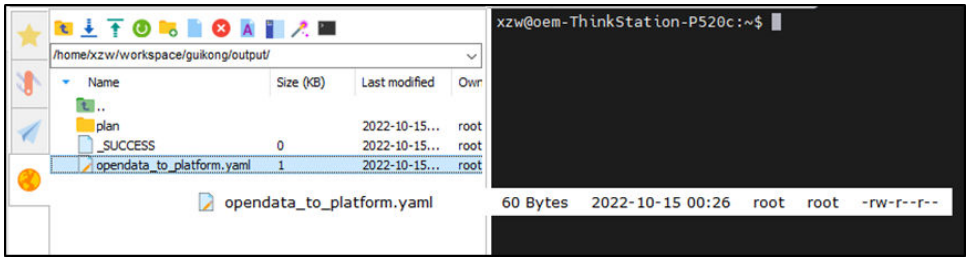
5. 本地调试。

   命令示例：

   ```
   docker run -v ${HOME}/workspace/guikong/2022-09-23-15-26-54_6.bag_pnc.bag:/tmp/input.bag -v ${HOME}/workspace/output:/tmp/output --env rosbag_path=/tmp/input.bag --env output_dir=/tmp/output --entrypoint /bin/sh guikong-demo:0.1 “-c” “bash /home/Octopus/run.sh”
   ```

   运行后查看输出结果：

图 4-27 输出结果



确定算法输出、yaml和_SUCCESS标识文件都存在，且文件所有人可读。

4.4 回放仿真

4.4.1 任务管理

回放仿真功能支持创建回放仿真作业，数据包选择回放仿真算子就可以触发相对应的作业。

创建作业

步骤1 在左侧菜单栏中，选择“数据处理 > 回放仿真”。

步骤2 选择“作业总览”页签，单击“创建作业”，填写回放仿真算子作业信息。

表 4-22 回放仿真算子作业

参数	示例	说明
处理算子	选择算子	选择已创建的输出类型为“回放仿真”的算子。
资源规格	选择资源	当前项目中可用的资源规格，资源配置需要平台管理员在集群纳管中创建，支持选择带有GPU的资源规格。
优先级	0	设定任务的优先级，数值取[-50,50]的整数，数字越大，优先级越高。
环境变量	-	配置算子的环境变量。允许添加的环境变量个数不超过10个。 ● Key：只能由英文、数字、和特殊符号（ ,-_ ）组成，且需要以字母开头 。长度不超过64个字符。 ● Value：只能由英文、数字和特殊符号（ \/,.[]-_ ）组成 。长度不超过64个字符。
超时时间	24	作业运行时长超过该时间后，作业会自动停止。 如果该值为空，默认超时时间为24小时，或设置为[1，720]的整数。
数据类型	数据包	选择输入数据类型，通用存储、数据包或数据场景。
选择数据	选择数据包	选择需要操作的数据包中的数据。
输出目录	选择输出目录	选择可输出的通用存储的目录。

步骤3 单击“创建”， 在任务管理页面可查看创建好的作业。

----结束

回放仿真作业相关操作

在“作业总览”页签，还可以完成以下操作。

表 4-23 作业总览相关操作

任务	操作步骤
任务详情	单击操作栏中的“详情”，可查看作业的详情。当作业出现异常时，请参见表4-17分析定位问题。
作业日志	1. 单击作业总览列表，操作栏中的“详情”。 2. 在作业详情页，可在线查看或本地下载作业日志。
重启作业	单击操作栏中的“重启”，可将目标作业进行重启。 **说明** 仅支持作业状态为“失败”、“停止”、“运行异常”和“上传失败”的作业。
停止作业	• 选择单个作业，单击操作栏的“更多 > 停止”，停止单个作业。 • 勾选多个作业，单击列表上方的“停止”，可批量停止作业。 **说明** 当作业状态为过程态（排队中、启动中、运行中）时，可以停止作业。
删除作业	• 选择单个作业，单击操作栏的“更多 > 删除”，删除单个作业。 • 勾选多个作业，单击列表上方的“删除”，可删除多个作业。 **说明** 当作业状态为终止态（提交失败、已停止、已完成、运行异常）时，可以删除作业。
重新上传作业	单击操作栏中的“更多 > 重新上传”，可重新上传作业。 **说明** 仅支持作业状态为“上传失败”的作业。
搜索作业	选择“ID”、“数据包ID”或“算子”，在搜索输入框中输入搜索条件，按回车键即可查询。
对比回放	单击操作栏中的“对比回放”，界面同时展示回放仿真的任务和数据场景的任务。 **说明** 数据包如果被删除，则无法单击“对比回放”。

4.4.2 作业队列

平台为回放仿真任务提供作业队列的功能，用户可在此查看任务队列，同时支持对任务优先级的调整。

说明

其他租户的队列作业不允许操作。

步骤1 在左侧菜单栏中单击“数据处理 > 回放仿真”。

步骤2 选择“作业队列”页签，查看作业。

----结束

回放仿真作业队列相关操作

在“作业队列”页签，还可以进行以下操作。

表 4-24 作业队列相关操作

任务	操作步骤
置顶作业	单击操作栏中的“置顶”，即可将作业调整至队列中最高优先级。
置底作业	单击操作栏中的“置底”，即可将作业调整至队列中最低优先级。
上移作业	单击操作栏中的“更多 > 上移”，即可将作业调整至队列中上一级。
下移作业	单击操作栏中的“更多 > 下移”，即可将作业调整至队列中下一级。
更新作业优先级	1. 勾选单个或多个作业。 2. 单击“更新优先级”，输入[-50,50]的整数，数字越大，优先级越高。 3. 单击“确定”，优先级更改成功。相同优先级的任务，作业队列更新时间越早，优先级越高。
搜索作业	在搜索输入框中输入“作业ID”，按回车键即可查询。

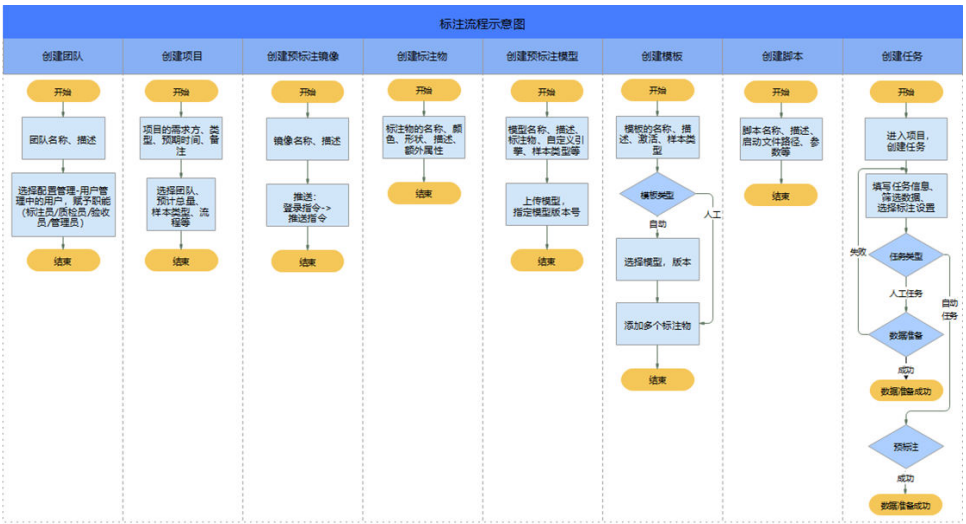
5 标注服务

5.1 标注服务简介

Octopus标注服务为标注团队提供标注平台管理员、团队管理员和标注任务操作人员（标注员/审核员/验收员）三类职能。三类职能对标注平台的操作权限不同，保证标注任务的数据安全。标注服务管理员根据不同标注任务创建标注团队，分配标注任务操作人员不同职能。创建标注项目，并在项目下创建标注模板及标注任务。团队管理员管理团队内部成员，可为团队创建项目和标注任务，供成员认领。标注任务的操作人员是标注任务的实际执行人员，根据标注任务所处流程，有标注员、审核员和验收员三种职能。

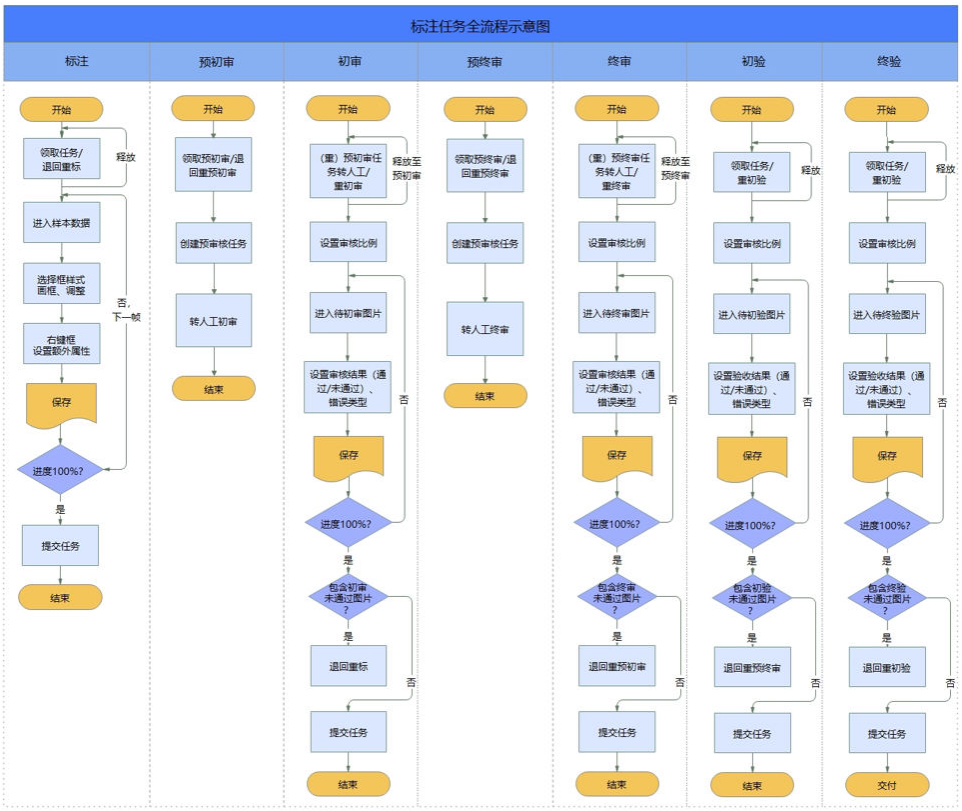
标注服务的开发流程如下：

图 5-1 标注服务开发流程



标注任务创建成功后，由团队中的标注员认领任务，标注任务进入标注流程。标注任务全流程如下，根据项目流程，可对除交付之外的其他流程进行裁剪：

图 5-2 标注任务全流程



- **项目管理**：通过对项目内的任务以及项目内的标注物管理，用户可根据业务需求不同，创建不同类型的项目。
- **团队管理**：为标注团队进行人员职能的分配。未分配到标注项目的团队不可查看该项目信息，确保标注任务的安全性。
- **用户管理**：可为任务提供角色分配，对所有团队下普通用户进行管理。
- **标注管理**：提供项目外的可视化的标注物管理，支持自定义创建多种标注物的形状和颜色，可用于预标注和人工标注指定物体，或自定义算法模型中关联特定标注物。
- **脚本管理**：用户可以根据自身业务的需要创建标注脚本，用于标注任务。
- **模板管理**：提供预标注模板和人工标注模板，用户可根据需求选择。

5.2 项目管理

5.2.1 标注项目

5.2.1.1 创建项目

Octopus标注平台标注任务以项目形式进行管理，平台管理员根据业务的需求方及不同标注任务类型创建项目。项目内包含多个同类型的标注任务，可根据业务需要在项目内追加任务。平台管理员可将标注项目分配给标注团队完成。

项目内包含多个标注任务，在创建项目时需绑定标注团队，指定团队完成项目内的标注任务。

步骤1 在左侧菜单栏中选择“标注服务 > 项目管理”。

步骤2 选择“标注项目”页签，单击“创建项目”，参考下表填写项目信息。

表 5-1 标注项目参数

参数	说明
项目名称	项目名称不支持自定义，由需求方、项目类型、预计完成日期以及备注组成。其中，需求方、项目类型以及预计完成时间必填。
标注团队	负责完成该项目的标注团队。团队需提前创建完毕。
预计总量	预计项目内所有任务的总量，即图片、3D点云、音频文件或文本总帧数。
数据类型	标注任务的数据类型。当前支持图片、3D点云、音频和文本四种类型。不同数据类型支持的文件格式请参见表5-2。
项目任务流程	除交付节点为必选之外，可自由选择其他任务流程节点。 **说明** 数据类型为音频或文本时，项目任务流程不支持选择预初审和预终审。

表 5-2 数据类型支持的文件格式

数据类型	文件格式
图片	".bmp"，".jpg"，".jpeg"，".png"，".gif"，".tiff"。
点云	".pcd"（ascii和binary格式）。
音频	".wmv"，".flac"，".mp3"，".m4v"。
文本	".txt"，".yaml"，".csv"，".xml"。

步骤3 单击“确认”。

在项目列表可查看新创建的项目。

----结束

标注项目相关操作

在“标注项目”页签，还可以进行以下操作。

表 5-3 标注项目相关操作

任务	操作步骤
查找项目	在搜索输入框中输入搜索条件，按回车键即可查询。

任务	操作步骤
编辑项目信息	单击操作栏内的“ 更多 > 编辑”，修改项目信息。 **说明** 该项目下仍有任务，则仅支持修改预计总量。
删除项目	● 选择单个项目，单击操作栏的“更多 > 删除”，删除单个项目。 ● 勾选多个项目，单击列表上方的“删除”，可批量删除项目。 **说明** 删除项目前需删除项目内包含的标注任务。
查看项目详情	单击操作栏内的“ 项目详情”，即可查看项目详情，具体请参考**项目详情**。
添加批次任务	单击操作栏的“添加批次任务”，即可为项目添加批次任务，具体步骤参考**添加批次任务**。

5.2.1.2 项目详情

标注平台支持统计项目的进展情况以及团队成员的标注效率，便于管理员把控项目整体进度，及时掌握成员的标注效率。单击项目名称后操作栏的“项目详情”，查看项目整体进度以及人员标注效率统计。项目详情分为项目概览、关联题库以及人员详情。

说明

数据类型为音频和文本的项目，暂不支持关联题库。项目详情分为项目概览和人员详情两部分。

项目概览

项目概览包括项目信息、项目今日状态以及项目相关数据图表统计，管理员可以快速掌握项目当前进度。

- 项目信息：项目名称、项目标注团队、项目预计结束时间、累计交付通过任务数、累计交付通过帧数以及总时长等信息。
- 今日状态统计：与昨日相比，今日项目中标注任务数量、审核、验收、交付任务数量及变化。绿色箭头表示上升，红色箭头表示下降。

  说明

  - 审核/验收任务：待审核任务数包含待预初审及待预终审，审核中包含预初审，预终审。不统计重预（初终）审任务。
  - 统计数值只显示项目任务流程包含的流程数值，如果流程中不包含，则用“--”代替数值。
- 项目任务流程：展示当前项目的流程。
- 项目相关数据图表统计：为便于管理员快速掌握项目进展情况，平台实时统计标注项目中相关重要数据量变化并将其制作成图表。管理者可通过图表中图形的变化了解当前项目进程。可设置时间范围，查询指定时间段内项目重要指标变化情况。目前为以下五种图表，图标的展示根据项目任务流程有所裁剪：
  - 标注帧数统计：平台统计标注任务从任务下发、任务标注提交、重标注数这几个标注任务常见环节中数量变化，并以帧为单位绘制成图表。

- 项目参与人数：标注员认领任务后，进行标注并保存任务进度，被平台视为参与该项目。同样，审核员认领审核任务进行审核并保存任务进度，被平台视为参与该项目。平台统计参与项目的标注员和审核员人数，并绘制成图表。
- 平均标注效率：标注平台实时统计项目标注速度变化情况，支持按帧数、框数或对象数查看。

  说明

  - 标注工作时长为标注工作花费的总时长（实时统计）。
  - 按帧查看时：标注速度=累计已提交任务的标注工作时长/总帧数。
  - 按对象查看：标注速度=累计已提交任务的标注工作时长/总对象数。
  - 按框查看：标注速度=累计已提交任务的标注工作时长/总框数。
  - 鼠标静止时间大于等于5分钟，静止时间不计入标注工作时长内。
- 单日标注修改帧数：统计当日项目中所有标注员提交的标注任务总帧数。
- 单日重标注任务数：根据项目流程，如果已提交的标注任务不满足下一步流程操作员的通过标准，操作员将其打回给标注员进行重标注，图表统计每天处于重标注状态的任务，同一任务单日多次被退回至重标注，不重复统计。

关联题库

每个标注项目均可关联题库以及考试，课程或考试需在培训考试系统中创建完成，具体可参考**培训考试**。

步骤1 进入项目详情页，单击“关联题库”页签。

步骤2 单击“关联培训”或“关联考试”，选择“课程名称”或“考试名称”以及“职能”。

步骤3 单击“确认”，课程或考试添加成功。

步骤4 编辑题库。

可单击操作栏的“编辑”，编辑职能。

步骤5 解绑题库或考试。

可单击操作栏的“解绑”，解绑题库或考试。

步骤6 查看题库或考试详情。

单击题库或考试名称，可查看题库或考试详情。

----结束

人员详情

标注平台统计承包标注项目的标注团队中所有标注员和审核员的累计工作量情况，如累计工作时长、提交任务数量等。管理员可单击箭头升序或降序排列，查看标注员和审核员的完成情况排序，也可以筛选时间，展示特定时间段内标注员、审核员（初审员/终审员）的任务累计完成情况。标注平台从帧数、标注框或对象三个维度统计累计提交任务数、累计工作时长、累计确认数、平均耗时五个指标，衡量标注员、审核员（初审员/终审员）的任务完成情况。

人员详情模块的展示根据项目任务流程有所裁剪。（标注员统计、初审员统计、终审员统计）

- 累计确认框数：标注员、初审员、终审员提交任务中确认有效框数的数量以及返修环节中新增的有效框数之和。

  说明

  返修后有效框数变更会在重新提交任务时刷新。

- 累计提交任务数：标注员、初审员、终审员认领并提交的任务数量。

  说明

  如果提交的任务，被下一步流程操作员退回时，则此任务不计入累计提交任务数。

- 累计重标注任务数：标注员重标注的任务次数。如果某任务返修多次，则累计重标注任务数叠加，累计提交任务数不变（只有标注员统计重标注任务）。
- 累计工作时长：标注员累计标注总时长、初审员累计初审总时长、终审员累计终审总时长，包含各自返修时长。

  说明

  返修后有效帧变更为无效帧，累计时长会减去无效帧时间。无效帧变更为有效帧，累计时长会加上有效帧时间。

- 平均单框耗时：平均单帧标注/初审/终审的时间，计算方式为：累计工作时长/累计确认有效数。

导出人员标注效率

标注平台支持导出该项目所有标注员，初审员，终审员的标注效率，以Excel形式下载至本地，便于用户查看。单击人员详情页面的导出按钮，即可将当前页面的标注效率导出。

5.2.2 批次任务

5.2.2.1 批次任务列表

一个项目可包含多个标注任务，标注平台支持批量创建任务。

添加批次任务

步骤1 在左侧菜单栏中选择“标注服务 > 项目管理”。

步骤2 选择“标注项目”页签，单击操作栏中的“添加批次任务”。

步骤3 参考如下表格填写任务名称，备注和规范等信息。

表 5-4 创建标注任务

参数	说明
任务名称	为当前项目名称的“项目需求方-项目类型”组成，不可修改。
备注	备注信息可自定义。
规范	可根据需求选择规范，也单击“添加规范”添加新的规范，附上规范便于标注和审核人员在标注和审核过程中依据规范进行标注和审核，提高任务完成质量。

参数	说明
数据集	选择相对应的数据集或标注导入、导出数据集。 ● 数据集中的标注物如果已经存在，则标注物信息以数据集中为准。 ● 数据集中的帧数据是按照帧文件夹名称的字符编码（Unicode）大小进行排序，顺序可在标注界面前后帧体现。
任务类型	任务类型分为“人工标注”或“预标注”。 任务创建需排队，排队状态请在项目管理菜单下的批次任务队列界面进行查看。
标注脚本	选择创建好的标注脚本。 任务类型为“预标注”时存在。
标注物	● 人工标注：标注模板和项目内标注物至少选取一种。当选取标注模板后，标注模板内标注物与项目内有重名标注物时会跳出弹窗，提示用户项目内有重名标注物，如需使用该标注物，单击确认后默认选择项目下标注物展示在下方显示框中展示。 ● 预标注：预标注模板关联模型，最终调用模型关联的镜像仓库，镜像由用户自行维护上传。
资源规格	当前项目中可用的资源规格，资源规格需要平台管理员在运维配置纳管标注任务用途（人工标注或预标注）的作业集群后创建。
优先级	设定任务的优先级，数值取[-50,50]的整数，数字越大，优先级越高。

步骤4 以上信息填写无误后，单击“创建并返回”。如果需批量创建任务，单击“创建不返回”，参照上述步骤继续创建任务。

----结束

批次任务相关操作

单击项目名称，在“批次任务列表”，还可以进行以下操作。

表 5-5 批次任务相关操作

任务	操作步骤
查询批次任务	在批次任务列表搜索框中，可通过任务名称、规范，任务状态和任务类型查询任务。
导出批次任务列表	单击页面的“导出”，可按需选择导出项，可将当前页面的所有标注任务导出至本地，以Excel形式查看。
导出任务	单击操作栏中的“导出数据集”，可创建导出任务。
导出单个批次任务	单击操作栏中的“导出任务列表”项目名称，可按需选择导出项，导出单个批次任务中拆分的任务。

任务	操作步骤
拆分批次任务	单击操作栏中的“拆分”，即可对任务做拆分处理。具体步骤参考**批次任务列表**。
查看批次任务日志	单击操作栏中的“日志”，即可查看批次任务日志。
删除批次任务	选择批次任务单击“删除批次任务”，即可删除批次任务。 **说明** 如果批次任务被拆分成子任务，则需要删除子任务后，才可删除批次任务。

5.2.2.2 任务列表

任务列表展示批次任务列表中所有子任务下的任务，方便用户直观查看所有的任务。

在任务列表还可以进行认领、提交、释放、退回、审核和删除等操作。

在任务列表也可以导出任务。

5.2.2.3 项目内标注物管理

项目内标注物管理展示该项目下创建的标注物，只与该项目关联。

创建标注物

步骤1 在左侧菜单栏中选择“标注服务 > 项目管理”。

步骤2 选择“标注项目”页签，单击项目名称，进入项目内任务列表。

步骤3 选择“标注物管理”页签，单击“创建标注物”，填写相关信息。

表 5-6 创建标注物参数

参数	说明
标注物名称	简洁标识待标注对象特征。
标注物颜色	通过十六进制码指定颜色，建议不同种类标注物颜色做明显区分。
标注物形状	支持点、圆、2.5D、矩形、OBB矩形、实线、虚线、多边形、立方体、音频、文本。具体参数请参考**表5-15**。
标注物描述	描述该标注物的主要作用，如标注对象、适用的标注场景等。
标注物额外属性	描述标注对象可能存在的特殊情况，如遮挡、截断等，支持自定义。 **说明** 标注物的额外属性个数不超过25，每个额外属性的选项个数不超过25。
标注物维度	当标注形状选择“bndbox”和“cube_3d”时，可根据需求设置标注物维度，绘制时即可使用设置好的大小。

----结束

项目内标注物相关操作

在项目内标注物列表，还可以进行以下操作。

表 5-7 项目内标注物相关操作

任务	操作步骤
编辑标注物	单击操作栏内的“编辑”，即可修改标注物信息。
删除标注物	项目内的标注物，无法直接删除，删除项目时，随项目同步删除。
查询标注物	在搜索框内输入标注物名称，可进行模糊查找。
查看标注物详情	单击标注物名称，即可查看标注物详情。

5.2.2.4 规范管理

规范便于标注和审核人员在标注和审核过程中依据规范进行标注和审核，提高任务完成质量。添加规范的方式有以下两种：

- 规范管理中添加规范。
- 添加批次任务时，添加规范。

添加规范

步骤1 在左侧菜单栏中单击“标注服务 > 项目管理”。

步骤2 选择“标注项目”页签，单击项目名称，进入项目内任务列表。

步骤3 选择“规范管理”页签，单击“添加规范”，上传PDF格式规范文件。

步骤4 单击“确认”，上传成功后列表显示规范文件。

----结束

规范管理相关操作

在“规范管理”列表，还可以进行以下操作。

表 5-8 规范管理相关操作

任务	操作步骤
预览规范	单击操作栏中的“预览”，即可在线预览规范。
删除规范	单击操作栏中的“删除”，即可删除规范。但正在被使用的规范不可删除。

任务	操作步骤
查询规范	在搜索框内输入规范的名称，可进行模糊查找。

5.2.2.5 关系管理

项目的数据类型为“文本”，需要创建关系，并对关系进行管理。

创建关系

步骤1 在左侧菜单栏中选择“标注服务 > 项目管理”。

步骤2 选择“标注项目”页签，单击项目名称，进入项目内任务列表。

步骤3 选择“关系管理”页签，单击“创建”，填写创建信息。

- 名称：自定义关系名称。名称只能包含数字、英文、中文、下划线、中划线，输入长度不能超过64个字符。
- 颜色：选择关系的颜色。

步骤4 单击“确认”，添加关系。

----结束

5.2.3 批次子任务

5.2.3.1 批次子任务

批次任务可以拆分为多个子任务，方便多人进行操作。

拆分批次任务

当样本数量庞大时，平台可对批次任务做拆分处理，具体步骤如下：

步骤1 在左侧菜单栏中选择“标注服务 > 项目管理”。

步骤2 选择“标注项目”页签，单击项目名称，进入项目内任务列表。

步骤3 选择“批次任务列表”页签，单击操作栏中的“拆分”。

步骤4 输入子任务样本数量并单击“确认”。子任务样本数量不可大于批次任务的样本总数。

步骤5 单击批次任务名称前的，可查看拆分后的子任务列表。

步骤6 查看子任务详情。单击子任务名称，可查看子任务详情。

----结束

5.2.3.2 统计信息

在批次子任务的“统计信息”页面，可查看该任务详情，项目任务流程以及标注对象统计信息。

- 任务详情：任务名称、数据类型、标注状态、任务包含的标注信息等。
- 项目任务流程：显示任务当前进度情况。
- 标注框数统计：人工/预标注对象数量统计。
- 审核框数量统计：审核结果的统计。
- 标注帧数统计：任务标注帧数的统计。

5.2.3.3 预审核任务

当批次子任务配置并启动了预审核任务时，预审核任务界面才会显示预审核任务详情。

预审核任务相关操作

在“预审核任务”列表，还可以进行以下操作。

表 5-9 预审核任务相关操作

任务	操作步骤
查询预审核任务	在搜索输入框中输入搜索条件，按回车键即可查询任务。
查看模型	单击模型名称，界面跳转至模型详情，可查看模型。
查看任务报告	单击操作栏中的“报告”，可查看或下载任务报告。
查看任务日志	单击操作栏中的“日志”，可查看或下载日志详情。
删除任务	• 选择单个任务，单击操作栏的“删除”，删除单个任务。 • 勾选多个任务，单击列表上方的“批量删除”，可批量删除任务。

5.2.4 所有任务

平台管理员可在所有任务列表页查看所有项目中的子任务。

所有任务相关操作

在“所有任务”列表，还可以进行以下操作。

表 5-10 所有任务相关操作

任务	操作步骤
查询任务	在搜索输入框中输入搜索条件，按回车键即可查询。
查看任务详情	单击任务名称，界面跳转至任务详情，可查看任务详情信息。

任务	操作步骤
查看不同类型和状态的任务	单击数据类型、标注类型、状态和预审核状态过滤条件，查看不同类型和状态的任务。

5.2.5 任务队列

标注批次任务在创建之后，可在此查看任务队列，同时支持对任务优先级的调整。

步骤1 在左侧菜单栏中选择“标注服务 > 项目管理”。

步骤2 选择“任务队列”页签，可根据标注类型（预标注或人工标注）查看标注任务。

----结束

任务队列相关操作

在“任务队列”页签，还可以进行以下操作。

表 5-11 任务队列相关操作

任务	操作步骤
置顶任务	单击操作栏中的“置顶”，即可将任务调整至队列中最高优先级。
置底任务	单击操作栏中的“置底”，即可将任务调整至队列中最低优先级。
上移任务	单击操作栏中的“上移”，即可将任务调整至队列中上一级。
下移任务	单击操作栏中的“下移”，即可将任务调整至队列中下一级。
更新任务优先级	1. 勾选单个或多个批次任务。 2. 单击“更新优先级”，输入[-50,50]的整数，数字越大，优先级越高。 3. 单击“确定”，优先级更改成功。相同优先级的任务，任务队列更新时间越早，优先级越高。
搜索任务	在搜索输入框中输入搜索条件，按回车键即可查询。

5.3 团队用户

5.3.1 团队管理

Octopus标注服务提供团队管理，为不同团队分配不同项目。未分配到标注项目的团队不能查看该标注项目信息，保证标注任务的私密性及安全性。

前提条件

- 用户已开通账号。
- 管理员已添加用户，并分配标注平台完全控制权限。添加用户操作请参考**用户管理**。

创建团队

步骤1 在左侧菜单栏中选择“标注服务 > 团队用户 > 团队管理”。

步骤2 单击“创建团队”，填写团队信息。

- 团队名称：简要标识该团队名称，与其他团队区分，包含中英文、数字、“_”“-”，不得超过64个字符。
- 团队描述：描述该团队补充信息。不能包含“@^\#$%&*<>'|"/”，不得超过255个字符。
- 选择成员职能：下拉框中选择成员为标注员/审核员/验收员/管理员。

  说明

  - 当前工作空间为非default（授权类型为INTERNAL）时，选择成员职能时，只有选择当前工作空间下的授权对象。
  - 一个用户可加入多个团队，但在多个团队中只能承担同一职能。

步骤3 单击“确认”，团队创建完毕。

----结束

团队管理相关操作

在“团队管理”页面，还可以进行以下操作。

表 5-12 团队管理相关操作

任务	操作步骤
查找团队	在搜索输入框中输入搜索条件，按回车键即可查询。
编辑团队	单击操作栏内的“编辑”，修改团队信息。
删除团队	单击操作栏内的“删除”，即可删除该团队。 **说明** 如果团队已与项目绑定，则该团队不可删除。
查看团队详情	单击团队名称，即可查看团队详情。

管理团队成员

平台管理员可根据业务需求对团队中的成员进行管理，可为团队新增人员或删除人员，可为团队中成员重新分配职能。

单击团队名称，可查看该团队内所有成员及职能。

- 查找成员：在搜索框内输入名字进行查找。
- 新增人员：单击团队名称，单击“新增人员”，选择人员并分配职能。
- 删除成员：单击人员名称后“操作”一栏内的“删除”，可删除该成员。删除后不可恢复，请谨慎操作。

说明

- 成员删除后，该成员在此团队关联的所有项目内的标注效率也被删除。
- 该成员名下有未完成任务，不可删除该成员。

5.3.2 用户管理

Octopus标注服务提供用户管理，对所有团队下普通用户进行纳管。

前提条件

- 用户已开通账号并创建子账号。
- 管理员已添加用户，并分配标注平台完全控制权限。添加用户操作请参考**添加用户**。

新增人员

步骤1 在左侧菜单栏中选择“标注服务 > 团队用户 > 用户管理”。

步骤2 单击“新增人员”，填写人员信息。

- 关联用户：下拉框中选择子用户。
- 用户名：简要标识该用户名称，与其他用户区分，包含中英文、数字、“_”“-”，不得超过64个字符。
- 职能：下拉框中选择用户为标注员/审核员/验收员/管理员。

说明

- 当前工作空间为非default（授权类型为INTERNAL）时，只能新增当前工作空间下的授权对象。
- 一个用户只能承担同一职能。

步骤3 单击“确认”，用户创建完毕。

----结束

用户管理相关操作

在用户列表，还可以进行以下操作。

表 5-13 用户管理相关操作

任务	操作步骤
编辑用户职能	单击操作栏内的“编辑”，可针对“用户名”和“职能”进行修正。

任务	操作步骤
删除用户	• 单击操作栏的“删除”，删除单个用户。 • 勾选多个用户，单击列表上方的“删除”，可批量删除用户。
查询用户	在搜索框内输入搜索条件，可进行模糊查找。

5.4 标注管理

5.4.1 标注管理

标注管理主要提供可视化的标注物管理，支持自定义创建多种标注物的形状和颜色，可用于预标注和人工标注指定物体，或自定义算法模型中关联特定标注物。如果在创建标注模板时，没有找到满足当前所需的标注物，则可以通过标注物管理添加新标注物。

说明

不同标注物可依靠标注物名称以及标注物描述区分。

新建标注物

步骤1 在左侧菜单栏中选择“标注服务 > 标注管理 > 标注管理”。

步骤2 单击“创建标注物”，填写相关信息。

表 5-14 创建标注物参数

参数	说明
标注物名称	简洁标识待标注对象特征。
标注物颜色	通过十六进制码指定颜色，建议不同种类标注物颜色做明显区分。
标注物形状	支持点、圆、2.5D、矩形、OBB矩形、实线、虚线、多边形、立方体、音频、文本。具体参数请参考**表5-15**。
标注物描述	描述该标注物的主要作用，如标注对象、适用的标注场景等。
标注物额外属性	描述标注对象可能存在的特殊情况，如遮挡、截断等，支持自定义。 **说明** 标注物的额外属性个数不超过25，每个额外属性的选项个数不超过25。
标注物维度	当标注形状选择“bndbox”和“cube_3d”时，可根据需求设置标注物维度，绘制时即可使用设置好的大小。

表 5-15 标注形状参数说明

形状类型	Json编写示例

矩形	"bndbox":{
	"ymin":506,
	"serialVersionUID":1141184345691781050,
	"xmin":1058.4515,
	"ymax":613.10834,
	"xmax":1110
	}

OBB矩形	"obb": {
	"center_x": 706.1222,
	"center_y": 500.1722,
	"width": 800.1222,
	"height": 503.44,
	"angle": 105.33
	}

实线	"line": {
	"size": 2,
	"points": [{
	"xpoint": 1397.277,
	"ypoint": 669.518
	}, {
	"xpoint": 1514.688,
	"ypoint": 698.297
	}
	]
	}

虚线	"dashed":{
	"yend":837.931,
	"xbegin":318.86,
	"xend":532.573,
	"ybegin":990.066
	}

圆	"circle": {
	"xcenter": 184.83609,
	"ycenter": 281.91388,
	"radius": 129.65723
	}

形状类型	Json编写示例

多边形	"polygon":{
	"size": 10,
	"points": [{
	"xpoint": 1698.5503,
	"ypoint": 497.154
	}, {
	"xpoint": 1732.3251,
	"ypoint": 497.45828
	}, {
	"xpoint": 1796.6951,
	"ypoint": 499.52362
	}, {
	"xpoint": 1858.6555,
	"ypoint": 501.58896
	}, {
	"xpoint": 1920.0,
	"ypoint": 503.05655
	}, {
	"xpoint": 1920.0,
	"ypoint": 524.9086
	}, {
	"xpoint": 1884.8167,
	"ypoint": 525.6847
	}, {
	"xpoint": 1798.072,
	"ypoint": 523.6194
	}, {
	"xpoint": 1699.0431,
	"ypoint": 520.0711
	}, {
	"xpoint": 1698.5503,
	"ypoint": 497.154
	}
	]}

点	"points": {
	"size": 1,
	"points": [{
	"xpoint": 961.81116,
	"ypoint": 863.2868
	}
	]
	}

形状类型	Json编写示例

立方体	"cube_3d": {
	"serial_number": 1,
	"location": {
	"x": 0.4288838867083183,
	"y": 24.40893444102772,
	"z": -0.965
	},
	"alpha": 1.553227360022738,
	"dimensions": {
	"height": 0.9900000095367432,
	"length": 1.600147008895874,
	"width": 1.2214816808700562
	},
	"rotation": {
	"x": 0.0,
	"y": 0.0,
	"z": 6.12610567
	},
	"attribute": "{}"
	}

2.5D	"multiBox": {
	"size": 10,
	"points": [{
	"xpoint": 835.032,
	"ypoint": 591.61725
	}, {
	"xpoint": 1171.9022,
	"ypoint": 591.61725
	}, {
	"xpoint": 1171.9022,
	"ypoint": 441.2934
	}, {
	"xpoint": 835.032,
	"ypoint": 441.2934
	}, {
	"xpoint": 835.032,
	"ypoint": 591.61725
	}, {
	"xpoint": 1374.7489,
	"ypoint": 522.79425
	}, {
	"xpoint": 1374.7489,
	"ypoint": 441.2934
	}, {
	"xpoint": 1171.9022,
	"ypoint": 441.2934
	}, {
	"xpoint": 1171.9022,
	"ypoint": 591.61725
	}, {
	"xpoint": 1374.7489,
	"ypoint": 522.79425
	}
	]
	}

形状类型	Json编写示例

音频	"audio": {
	"sample_type": "IMAGE/POINT_CLOUD/AUDIO/TEXT",
	"audio_meta_info": {
	"duration": "18s",
	"name": "aaa.wav",
	"source": "https://obs_pathXXX/XXX/aaa.wav"
	},
	"labels": [{
	"serial_number": 11,
	"label_meta_id": 125, //允许无
	"label_meta_name": "小明", //允许无
	"shape_type": "audio",
	"audio": {
	"xmin": 0, // 起始时间，单位秒
	"xmax": 10.254, //结束时间，单位秒
	"text": "adheidheoi", //允许空
	"author": "小明", //说话人名称，允许无
	"gender": "FEMALE", //说话人性别，允许无
	},
	"attribute": {}
	}, {
	"serial_number": 12,
	"label_meta_id": 121, //允许无
	"label_meta_name": "小A", //允许无
	"shape_type": "audio",
	"audio": {
	"xmin": 0, // 起始时间，单位秒
	"xmax": 10.254, //结束时间，单位秒
	"text": "adheidheoidsdfsdf", //允许空
	"author": "小A", //说话人名称，允许无
	"gender": "MALE", //说话人性别，允许无
	},
	"attribute": {}
	}
	]
	}

形状类型	Json编写示例

文本	"text": {
	"sample_type": "IMAGE/POINT_CLOUD/AUDIO/TEXT",
	"text_meta_info": {
	"name": "aaa.txt",
	"source": "https://obs_pathXXX/XXX/aaa.txt"
	},
	"labels_ext":{
	"text_attribute":"", //文档属性标注，可为空
	"text_intention":"", //文档意图标注，可为空
	"relations":[//实体关系标注
	{
	"id": 3,
	"type": "从属",
	"color": "#21a735",
	"head_serial_number": 1,
	"tail_serial_number": 2
	},
	{
	"id": 4,
	"type": "同事",
	"color": "#21a735",
	"head_serial_number": 222,
	"tail_serial_number": 34
	}
	]
	},
	"labels": [{
	"serial_number": 11,
	"label_meta_id": 125, //可为空
	"label_meta_name": "水果", //可为空
	"shape_type": "text",
	"text": {
	"start_idx": 2, //起始位置
	"end_idx": 6 //结束位置
	},
	"attribute": {}
	}, {
	"serial_number": 12,
	"label_meta_id": 15,
	"label_meta_name": "人名",
	"shape_type": "text",
	"text": {
	"start_idx": 1476, //起始位置
	"end_idx": 1478 //结束位置
	},
	"attribute": {}
	}
	]
	}

----结束

自定义标注物相关操作

在自定义标注物列表，还可以进行以下操作。

表 5-16 自定义标注物相关操作

任务	操作步骤
编辑标注物	单击操作栏内的“编辑”，即可修改标注物信息。
删除标注物	单击操作栏内的“删除”，即可删除标注物。
查询标注物	在搜索框内输入标注物名称，可进行模糊查找。
查看标注物详情	单击标注物名称，即可查看标注物详情。

5.4.2 脚本管理

Octopus平台提供标注脚本服务，结合模型管理，标注物管理等，更好的服务于标注任务。

创建标注脚本

步骤1 在左侧菜单栏中选择“标注服务 > 标注管理 > 脚本管理”。

步骤2 单击“创建标注脚本”，填写标注脚本信息。

- 名称：包含中英文、数字、“_”“-”，不得超过64个字符。
- 描述：简要描述标注脚本，不包含“@^\#$%&*<>'|"/”，不得超过256个字符。
- **Boot文件路径**

  输入标注脚本启动文件的路径，该路径为启动文件在脚本中的相对路径。
  - 如果启动文件“xxx.py”位于脚本的一级目录下，则路径为“xxx.py”，文本框内只需输入“xxx”。
  - 如果启动文件“xxx.py”位于脚本的二级目录下，则路径为“xxx/xxx.py”，文本框内只需输入“xxx/xxx”。

  说明
  - 用户无需输入启动文件路径的后缀“.py”，平台会自动添加。
  - 当前只支持.py类型的启动文件。
  - 文件路径只能包含数字、英文、下划线、中划线、点、斜杠和空格，且不能超过256字符。
- **参数列表**

  可以自定义boot文件的启动参数，允许添加的参数个数不超过20个。

  说明
  - Key：只能由英文、数字、和特殊符号（,-_）组成，且需要以字母开头。长度不超过64个字符。
  - Value：只能由英文、数字和特殊符号（\/,.[]-_）组成 。长度不超过128个字符。

步骤3 以上信息填写无误后，单击“确认”。

----结束

脚本管理相关操作

在标注脚本列表，还可以进行以下操作。

表 5-17 脚本管理相关操作

任务	操作步骤
编辑标注脚本	单击操作栏内的“编辑”，即可修改标注脚本信息。
删除标注脚本	● 单击操作栏的“删除”，删除单个标注脚本。 ● 勾选多个标注脚本，单击列表上方的“删除”，可批量删除标注脚本。
查询标注脚本	在搜索框内输入搜索条件，可进行模糊查找。
查看标注脚本详情	单击标注脚本名称，即可查看该标注脚本参数。

5.4.3 模板管理

在进行自动驾驶模型训练过程中需要大量有标签的图片或视频数据，因此在模型训练之前需要对处理完的数据进行各类标注，进行场景识别。Octopus提供预标注功能，支持部分预标注模型，能够节省70%的人力成本。也提供人工标注功能，用户可以针对未标注数据在线手动标注或预标注后人工确认难例数据。在创建标注任务前，需创建标注模板。

创建标注模板

创建标注模板是创建标注任务的第一步，标注模板包含该标注任务所需的所有标注类别标签。创建标注模板的步骤请参考如下：

步骤1 在左侧菜单栏中选择“标注服务 > 标注管理 > 模板管理”。

步骤2 单击“创建标注模板”，填写标注模板相关信息。

- 模板名称：支持中英文、数字、“-”、或“_”组成的合法字符串，长度不得超过64字符。
- 模板描述：简要描述模板内容，不包含“@^\#$%&*<>'|"/”，不得超过255个字符。

步骤3 选择标注参数。

- 人工标注
  - 标注类型：人工标注，该标注模板用于人工标注任务。
  - 数据类型：根据模板包含的样本类型选择图片、3D点云、音频或文本。

  - 标注：从下拉框中选择符合用户业务所需的标注物。用户可自定义添加标注物以满足业务所需。
- 预标注
  - 标注类型：预标注，该标注模板用于预标注任务。
  - 数据类型：根据模板包含的样本类型选择图片、3D点云、音频或文本。
  - 模型来源：根据模型的来源来选择。
  - 标注模型：选择模型和版本。
  - 模板标注物：展示模板标注物。
  - 已选标注物：展示已选标注物。
  - 标注：从下拉框中选择符合用户业务所需的标注物。用户可自定义添加标注物以满足业务所需。

    说明

    1. 添加的标注物在预标注任务人工确认阶段，仅用于手动修改标注对象。
    2. 标注类型为预标注时，标注物为所选标注模型内包含的所有标注物标签，支持增加或减少标注物。手动添加的标注物不具备模型预标注功能。

步骤4 以上信息填写无误后，单击“创建并返回”。如果需继续创建任务，单击“创建不返回”，参照上述步骤继续创建任务。

----结束

标注模板相关操作

在标注模板列表，还可以进行以下操作。

表 5-18 标注模板相关操作

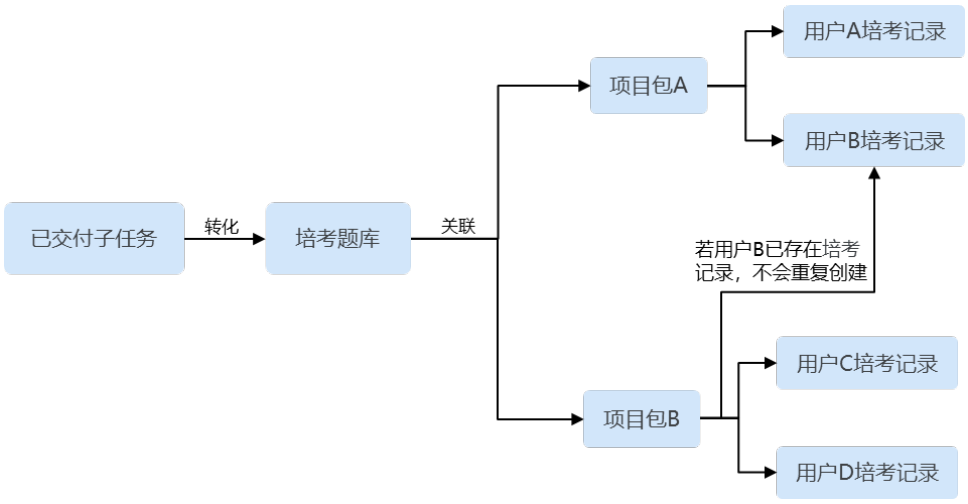
任务	操作步骤
编辑标注模板	单击操作栏内的“编辑”，即可修改标注模板信息。
删除标注模板	● 选择单个标注模板，单击操作栏的“删除”，删除单个标注模板。 ● 勾选多个标注模板，单击列表上方的“删除”，可批量删除标注模板。
查询标注模板	在搜索框内输入搜索条件，可进行模糊查找。
查看标注模板详情	单击标注模板名称，即可查看该模板包含的标注物列表。

5.5 培训考试

5.5.1 培训系统

已交付的子任务，可以转化为培训题库或考试题库，当项目包关联了培训题库或考试题库时，标注员、审核员可通过系统进行培训或考试。

图 5-3 培训考试



培训系统支持题库管理、培训数据统计、正式数据一键转换为培训题目、培训题中可进行结果查看、可和培训员标注结果进行比对。

新增培训题库

步骤1 在左侧菜单栏中选择“标注服务 > 培训考试”。

步骤2 选择“培训系统 > 培训题库”页签，单击“新增”，参考下表填写项目信息；

- 课程名称：可自定义课程名称，只能包含数字、英文、中文、下划线、中划线；
- 课程描述：输入课程描述，不能包含“@^\#$%&*<>'|"/”，不得超过255；
- 标注项目：下拉选择标注项目；
- 标注任务：下拉选择标注任务。

步骤3 单击“确认”，等待培训题库初始化成功后，在培训题库列表可查看详情。

----结束

培训题库相关操作

在培训题库列表，还可以进行以下操作。

任务	操作步骤
编辑培训题库	单击操作栏内的“编辑”，即可修改培训题库信息。
删除培训题库	• 单击操作栏的“删除”，删除单个培训题库； • 勾选多个培训题库，单击列表上方的“删除”，可批量删除培训题库。
查询培训题库	在搜索框内输入搜索条件，可进行模糊查找。
查看培训题库详情	单击培训题库名称，即可查看该培训题库详情。

培训记录

当项目关联了培训题库时，系统会自动在培训系统生成待培训记录。项目管理培训题库操作请参考**关联题库**。

管理员有权限修改培训记录，单击操作栏中“编辑”，可修改培训记录状态，当状态未锁定时，培训状态可根据实际情况发生改变。如果状态为锁定时，培训状态即为修改后的状态。

当考试记录生成后，由对应的标注员或者审核员可登录账号，单击“培训记录”名称，根据提示就可以进行培训。

5.5.2 考试系统

考试系统支持题库管理、考试试卷题目个数配置、正式数据一键转换为考试题目，考试提交可进行系统自动判卷，也可进行人工复核。

新增考试题库

步骤1 在左侧菜单栏中选择“标注服务 > 培训考试”。

步骤2 选择“考试系统 > 考试题库”页签，单击“新增”，参考如下填写项目信息。

- 考试名称：可自定义考试名称，只能包含数字、英文、中文、下划线、中划线；
- 考试描述：输入考试描述，不能包含“@^\#$%&*<>'|"/”，不得超过255；
- 来源：下拉选择标注项目以及标注任务；
- 任务帧数区间：滑动选择任务帧数，范围为“0-10”；
- 考试次数：默认值为5，可自行设定考试次数；
- 考试时长：默认值为60，可自行设定考试时长；
- 及格分：默认值为60，可自行设定及格分数。

步骤3 单击“确认”，等待考试题库初始化成功后，在考试题库列表可查看详情。

----结束

考试题库相关操作

在考试题库列表，还可以进行以下操作。

任务	操作步骤
编辑考试题库	单击操作栏内的“编辑”，即可修改考试题库信息。
删除考试题库	• 单击操作栏的“删除”，删除单个考试题库； • 勾选多个考试题库，单击列表上方的“删除”，可批量删除考试题库。
查询考试题库	在搜索框内输入搜索条件，可进行模糊查找。

考试记录

当项目关联了考试题库时，系统会自动在考试系统生成待考试记录。项目管理考试题库操作请参考**关联题库**。

管理员可查看考试记录，并进行以下操作。

流程	操作
编辑考试记录	在考试记录列表，单击“考试记录”，单击操作栏中的“编辑”，可根据情况编辑考试记录的状态。
复核考试记录	1. 在考试记录列表，单击“考试记录”，单击操作栏中的“复核”。 2. 单击“确认”，单击图片上的“复核”，对分数进行复核。 **说明** 当考试记录的状态为终态（判卷失败、已通过、失败）时，方可进行复核操作。
删除考试记录	在考试记录列表，单击“考试记录”，单击操作栏中的“删除”，可根据情况删除考试记录的状态。 **说明** 当考试记录的状态为终态（判卷失败、已通过、失败）时，方可进行删除操作。 只有一条未通过记录不允许删除。

当考试记录生成后，由对应的标注员或者审核员登录系统，单击“考试记录”名称，根据提示就可以进行考试。

5.6 团队管理员

5.6.1 管理团队

团队管理员可根据业务需求对团队中的成员进行管理，可为团队新增人员或删除人员，可为团队中成员重新分配职能。选择“标注服务 > 团队用户”，单击团队名称，可查看该团队内所有成员及职能。

- 查找成员：在搜索框内输入名字进行查找。
- 新增人员：单击“新增人员”，根据职能选择人员。
- 删除成员：单击人员名称后“操作”栏内的“删除”，可删除该成员。删除后不可恢复，请谨慎操作。

  说明

  - 成员删除后，该成员在此团队内关联的所有项目内的标效也被删除。
  - 如果该成员名下有未完成任务，不可删除该成员。

5.6.2 管理项目

项目需绑定管理员所在团队。

团队管理员管理项目步骤与平台管理员一致。

详细步骤请参考**项目管理**。

5.6.3 其他权限

团队管理员的其他管理权限步骤与平台管理员一致，具体请参考以下：

- 标注物管理请参考**标注管理**。
- 标注脚本管理请参考**脚本管理**。
- 标注模板管理请参考**模板管理**。

说明

- 仅团队管理员可以释放被退回的任务。
- 仅团队管理员可以释放预审核状态的任务。

5.7 标注员

5.7.1 认领标注任务

管理员将任务下发至团队后，团队中的标注员可以认领任务进行标注。 一个标注任务只能由一个标注员完成。同一项目中一个标注员最多认领5个标注任务。

认领标注任务

步骤1 在左侧菜单栏中选择“标注服务 > 项目管理”。

步骤2 选择“标注项目”页签，单击项目名称，查看该项目包含的可认领或已认领未提交的标注批次任务列表。

步骤3 单击未指定标注员的子任务名称后“操作”栏内的“认领”，认领该任务。

步骤4 单击该任务名称进入任务页签。

- 单击任意一张图片，进行人工标注。
- 单击“统计信息”，查看该任务详情，项目任务流程以及标注对象统计信息。
  - 任务详情：任务名称、数据类型、标注状态、任务包含的标注信息等。
  - 项目任务流程：显示任务当前进度情况。
  - 标注框数统计：人工/预标注对象数量统计以及标注数据统计。

步骤5 （可选）查看标注员标注效率。

返回标注项目列表，选择“项目详情 > 人员详情”，阅读**人员详情**查看标注员标注效率。

----结束

提交任务

标注完毕后，标注任务处于“标注中”或“重标注”，标注员可以提交任务进入下一步流程。在任务列表中单击任务名称后“操作”栏内的“提交”，提交任务，根据项目流程，由下一步流程的操作员审核或验收。

说明

提交后无法撤回，请谨慎操作。

释放任务

认领任务后，标注员不能继续标注名下任务时，可将任务释放，返回任务列表中，由其他标注员认领任务。选择任务名称后“操作”栏内的“更多 > 释放”，将任务释放。

说明

任务被退回重新标注时，该任务不能被标注员释放。

查看和下载任务日志

标注任务运行的过程中生成任务日志，平台提供了日志的查看以及下载功能。

单击任务名称，在该任务的详情页面，单击“日志”，可查看该任务日志列表及日志详情，支持下载至本地。

在日志服务页面中的日志列表部分详细展示了该任务包含的日志文件的大小以及最近写入时间。单击文件后的“查看”可查看详细执行过程。

5.7.2 人工标注操作指导

本节主要介绍Octopus标注平台的标注界面操作（以图片标注为例），标注任务的详细操作指导请参考**标注样例**。语音标注任务的详情操作指导请参考**5.10.9 语音标注任务**。文本标注任务的详情操作指导请参考**5.10.10 文本标注任务**。

步骤1 在左侧菜单栏中选择“标注服务 > 项目管理”。

步骤2 选择“标注项目”页签，单击项目名称。

步骤3 选择“批次任务列表”，展开批次任务，单击子任务名称，进入标注子任务详情页面。

步骤4 在待标注图片上单击“人工标注”，进入图片详情页（3D点云任务为单击“操作”列的“标注”）。

步骤5 选取特定形状和标签，对图片中物体进行手动标注。

选择标注形状，标注模板中已包含标注样本中所有标注对象信息，可单击选择对应标注标签。图片标注形状包含：

- 点、圆、切割线。
- 实线、虚线：适用于车道线等直线或折线标注。
- 矩形：适用于人车、红绿灯等矩形框标注。
- obb：适用于2D目标检测OBB类型数据标注。
- 多边形：适用于道路特征提取、语义分割等不规则图形标注。
- 2.5D：适用于2.5D数据标注。

步骤6 根据标注规范，进行标注。

步骤7 修改标注框。

根据标注对象不同，修改方式也有所不同。修改方式可参考以下示例。

- 车道线的修改，可通过在标注界面，单击已经标注的具体标注的线，标注的车道线将处于可选择状态，此时可通过拖拉标注的线中的点修改。
- 人车矩形框的修改，在标注界面，单击选择已经标注的具体标注框，标注框将处于可选择状态，此时可通过拖拉标注框四个顶点修改框体。
- 语义分割多边形标注框的修改，在标注界面，单击待修改的多边形框，标注框处于可选择状态，可通过拖拉多边形框的点修改。
- 点云标注框的修改，双击点云标注框后，在标注界面右侧出现该标注对象的三视图，根据图像判断标注对象的位置和状态，在三视图中拖拉标注框顶点修改。
- 联合标注。联合标注分为点云标注以及2d矩形框标注。2d标注与人车标注修改标注框方式一致。

步骤8 联合标注和点云标注还需添加锚点。单击“添加锚点”（标注类型为联合标注和3D标注，其他标注类型可忽略此步）。

说明

锚点：在Octopus标注平台点云标注类型中，车辆点云图像的标注框顶点与车辆实际顶点重合点为标注框锚点。3D点云图像的车辆标注框锚点可辅助确认车辆实际大小。

步骤9 选择额外属性。无额外属性的标注任务可忽略此步骤。左键标注框，选择属性。还可修改标注标签。

额外属性：属性选项支持自定义。

步骤10 （标注类型为联合标注和3D标注，其他标注类型可忽略此步）单击真实道路图片，展示其放大图。在标注界面左上角有“联合标注”开关。打开“联合标注”开关，表示该标注任务为联合标注。关闭开关表示标注任务为3D标注。

步骤11 标注完成后，勾选标注界面下方“已标注”。如果判断属于难例，还需勾选“难例”。如果判断该图片属于无效图，勾选“无效”。当前数据标注完成后单击“保存任务”。单击“下一张图片”继续标注。

步骤12 在标注过程中，为防止标注丢失，可随时单击“保存任务”。

----结束

5.7.3 标注工具和快捷键说明

标注平台提供多种标注工具和隐藏快捷键，满足不同标注任务所需，提高标注效率。

2D 标注

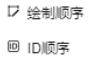
2D标注任务较多，类型多样，故平台开发多种标注工具，辅助标注工作，提高标注人员标注效率。

表 5-19 标注工具说明

一级图标	二级图标	名称	功能	快捷键
	无	选择	拖动标注框。	~

一级图标	二级图标	名称	功能	快捷键
	无	修改	可修改点的位置、线和框的形状，右键可打开属性。	Escape
		从上至下依次为： 点 实线 矩形 obb 多边形 2.5D 虚线 圆形 切割线	分别为点、实线、矩形、obb、多边形、2.5D、虚线、圆。 标注任务创建完毕，该标注任务所使用的所有标注已添加在任务中。单击标注形状，选择标注类别即可进行标注。	参考二级图标中的数字
		从上至下依次为： 复制上帧 对比上帧	复制上帧：将上一帧标注物完全复制到当前帧，适用于当前帧与上一帧标注物位置大致相同。 **说明** 第一帧图片不可用追踪功能。当前帧没有任何标注物。 对比上帧：上一帧和当前帧同屏展示，可进行对比前后帧信息。	复制上帧：F

一级图标	二级图标	名称	功能	快捷键
		从上至下依次为： 添加 列表	在同一标注任务中，会有多张图片中存在相似或一致的待标注对象。此时可利用全局标注工具将标注对象添加到全局标注列表中，只要切换图片应用即可，省去重复标注步骤。 添加： 为标注物添加提示信息后自动加入到列表。 列表： 打开全局标注物列表，可对列表中的标注物进行应用、删除、保留ID等操作。	• 添加：I • 列表：D
	无	比例尺	开启后鼠标指针带比例尺。一格为5px。	shift+b
	无	撤回	撤销上一步。	Ctrl+z
	无	复位	将图片重置到初始位置、大小。	o
	无	悬浮	开启后可通过鼠标悬停标注物查看额外属性、审核属性及描述。	+

一级图标	二级图标	名称	功能	快捷键
		从上至下依次为： 隐藏标注 隐藏标题 隐藏预览图 隐藏标注列表 网格	隐藏标注：开启后隐藏所有标注物。 隐藏标题：开启后隐藏所有标注物的标题。 隐藏预览图：开启后可隐藏界面下方缩略图。 隐藏标注列表：开启后可隐藏界面右侧标注列表。 网格：开启后可全图显示网格，每个小方格的边长为5px。	• 隐藏标注：L • 隐藏标题：T
		从上至下依次为： 绘制顺序 ID顺序	绘制顺序：可调整透明度，区域重叠时依照后绘制在上覆盖原则。 ID顺序：可调整透明度，图层重叠时按照ID较大的覆盖较小的原则，不支持实时刷新。	暂无
	无	上一张	无	Ctrl+Q
	无	下一张	无	Ctrl+E
	无	保存	保存当页信息。	S

2D 人车矩形标注

表 5-20 标注快捷键说明

快捷键	功能
ESC	修改状态。
~	选择移动工具（抓手模式）。
1	绘制点。
2	绘制实线，按住Shift可绘制垂直或水平线。
3	绘制矩形。

快捷键	功能
G	快捷添加分界线和点。
D	全局标注，D启用，选中标注物后i添加。
L	隐藏标注。
shift+B	比例尺。
U、J、H、K	选中边框上下左右微调。
Delete	删除选中目标。
Q、E	上下翻页。
Ctrl+C	复制属性。
Ctrl+V	粘贴属性。
Ctrl+Z	撤回上一步操作。
S	保存当页信息。
>	勾选审核通过并切到下一页。

2.5D 标注

表 5-21 标注快捷键说明

快捷键	功能
`	移动。
Esc	修改。
6	绘制2.5d框。
Q	切换上一张图片。
E	切换下一张图片。
S	保存当页信息。
>	勾选审核通过并切到下一页。
Delete	删除。
左键	拖动图片。
B	快速添加车牌、车轮点。
F	追踪上一帧图片。
U、J、H、K	选中边框上下左右微调。
Ctrl+Z	撤回上一个标注物。

快捷键	功能
Shift+B	比例尺。
D	全局标注物。
L	隐藏标注。
T	隐藏标题。

2D 语义标注

2D语义标注任务较多，类型多样，复用了大部分2D标注工具，在此基础上增加了以下快捷键促进标注速度。请参考下表。

表 5-22 标注快捷键说明

快捷键	功能
空格键	多边形闭合。
X	相交多边形共边。
5（非小键盘）	进入多边形绘制工具。
右键	未闭合状态撤销多边形上一点。
Alt+V/alt+鼠标左键	删除多边形上某点。
Alt+鼠标左键/鼠标滚轮	拖动语义标注的图片。
9（非小键盘）	绘制切割线。
M	绘制切割线后切割。
Z	开启交互式分割。
Ctrl+C	复制属性。
Ctrl+V	粘贴属性。
Delete	删除标注对象。
Shift+b	比例尺。
Ctrl+Z	撤回上一步操作。
S	保存本页标注。
T	隐藏标题。
L	隐藏标注。

2D 车道线标注

表 5-23 标注快捷键说明

快捷键	功能
ESC	修改状态。
2	实线绘制工具。
右键	未闭合状态取消上一点。
Alt+鼠标左键/Alt+V	去除点。
Ctrl+C	复制属性。
Ctrl+V	粘贴属性。
Delete	删除标注对象。
Ctrl+Z	撤回上一步操作。
S	保存当页信息。
左键单击	增加点。
左键双击	增加分段点。

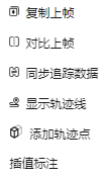
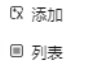
点云标注

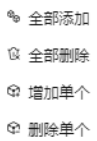
关于点云标注工具的说明，请参考下表。

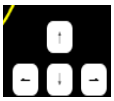
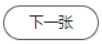
表 5-24 标注工具使用说明

标注工具一级图标	标注工具二级图标	名称	功能	快捷键
	无	选择	单击3D框进入选择状态，可对3D框进行三维拖动。	无
	无	修改	双击3D框进入修改状态，可对3D框的三视图进行调整。	无
		从上至下依次是： 标注 智能缩框标注	标注：在标注类别中选择标注框，进行普通3D框标注。 智能缩框标注：可根据框选范围内点云大小智能缩进标注框。	• 标注：V • 智能缩框标注：B

标注工具一级图标	标注工具二级图标	名称	功能	快捷键
		从上至下依次是: 撤销 恢复	撤销：撤销上一次操作。 恢复：恢复上一次撤销的操作。	• 撤销：Ctrl+Z • 恢复：Ctrl+Y
		从上至下依次是： 旋转 旋转90° 旋转180°	旋转点云图像。长按图标或快捷键点云图像会一直旋转。 按照箭头所示方向分别旋转点云图像90度和180度。	顺转：1或左键图标 逆转：Alt+1或右键图标 顺转90°：2或左键图标 逆转90°：Alt+2或右键图标 旋转180°：3
		从上至下依次是： 记录位置 返回记录位置 复位 点云角度保持原状	通过单击“记录位置”记录当前点云的视角，调整视角后可通过单击“返回记录位置”，回到记录视角。 复位：回到初始点云视角。 点云角度保持原状：开启后，双击打开三视图不会改变点云视角。	• 记录位置：R • 返回记录位置：Shift+R • 复位：O

标注工具一级图标	标注工具二级图标	名称	功能	快捷键
		从上至下依次是： 复制上帧 对比上帧 同步追踪数据 显示轨迹线 添加轨迹点 插值标注	复制上帧：将上一帧标注框完全复制到当前帧。第一帧不可用复制上帧功能。 对比上帧：标注界面分为两部分，左图是上一帧点云，右图是当前帧点云，可对前后两帧信息进行对比。 同步追踪数据：选中标注物后执行该操作可将当前任务中相同ID的所有标注物的大小类型进行覆盖。 显示轨迹线：选择3D框，开启此功能后可将当前任务中与此标注物ID相同的框进行轨迹线的连接。 添加轨迹点：开启后左键添加轨迹点，右键撤销当页轨迹点，并重新规划轨迹线，此ID最后一帧不可操作。 插值标注：当需要在第n帧和第n+k帧中标注完同一个追踪对象时，可以自动计算出此对象在中间k-1帧中的标注框。	复制上帧：F
		从上至下依次为： 添加 列表	添加：选中标注物，添加提示信息后自动进入全局标注物列表。 列表：打开列表可对添加的标注物进行应用、删除、保留ID等操作。	• 添加：I • 列表：D

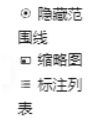
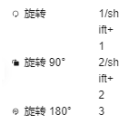
标注工具一级图标	标注工具二级图标	名称	功能	快捷键
	无	假想	点云标注中，可能会出现标注对象点云图像不完整的情况。此时可输入标注框的x、y、z值，（点云距离单位默认为米。）假想标注物的完整图像。或可根据标注物所属类别，单击平台预置类别标注框大小，直接标注。 	暂无
	无	高度过滤	设置高度限制，该高度以下点云无法被框选标注。	暂无
	无	地面点限制	开启后无法对地面点云进行标注。	暂无
	无	删除	选中3D框后单击图标或快捷键可删除框。	Shift+D
	无	长宽互换	可对3D框的xy值进行互换。	T
	无	补框	仅3D联合任务和2D3D关联标注任务点开2D图会显示此工具，当3D框缺少对应2D框时可使用此工具。	暂无
		从上至下依次为： 全部添加 全部删除 增加单个 删除单个	全部添加：为所有3D框添加锚点。 全部删除：删除所有3D框的锚点。 增加单个：选择单个3D框后增加单个锚点。 删除单个：删除单个3D框的锚点。	全部添加：Z 全部删除：X 增加单个：Shift+Z 删除单个：Shift+X

标注工具一级图标	标注工具二级图标	名称	功能	快捷键
		从上至下依次为： 缩略图 标注列表 隐藏范围线	缩略图：开启后隐藏界面下方缩略图。 标注列表：开启后隐藏标注列表。 隐藏范围线：开启后隐藏范围线。	暂无
	无	下降/上升 前移/后移 左移/右移	通过单击按钮或快捷键执行平移操作。	下降：Q 上升：E 前移：W 后移：S 左移：A 右移：D
	无	上旋/下旋 左旋/右旋	每次旋转是0.05π。	上旋：↑ 下旋：↓ 左旋：← 右旋：→
	无	上一张	无	Shift+Q
	无	下一张	无	Shift+E
	无	保存	保存当页信息。	Ctrl+S
	无	审核通过	勾选审核通过保存并进入下一页。	>

3D 语义标注

3D语义标注复杂，类型多样，故平台开发多种标注工具，辅助标注工作，提高标注人员标注效率。

表 5-25 标注工具使用说明

一级图标	二级图标	功能	快捷键
	无	多边形绘制	Z
	无	矩形绘制	X
	无	画笔绘制	C
	无	画笔调整变大小	鼠标滚轮前后滚动
	无	框选绘制	框选：V 确认：Enter 取消：空格键
	无	撤回上一步	Ctrl+Z
		隐藏范围线 缩略图 标注列表	无
		记录当前位置 回到上次位置 复位	记录当前位置：R 回到上次位置：Shift +R 点云图复位：O
	无	显示强度屏	I
		旋转 旋转90° 旋转180°	顺转：1或左键图标 逆转：Shift +1或右键图标 顺转90°：2或左键图标 逆转90°：Shift +2或右键图标 旋转180°：3
	无	下降/上升 前移/后移 左移/右移	下降：Q 上升：E 前移：W 后移：S

一级图标	二级图标	功能	快捷键
	无	上旋/下旋 左旋/右旋	上旋：↑ 下旋：↓ 左旋：← 右旋：→
无	标注物	标注物快捷键，最多8个，按照使用频次排序。	Alt+（1~8）
无	无	点云图复位。	O
无	无	上一张。	Shift+Q
无	无	下一张。	Shift+E
无	无	结束绘制状态。	ESC
无	无	保存当页信息，勾选已标注/审核通过。	Ctrl+S

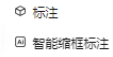
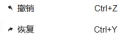
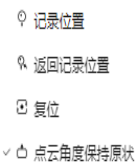
表 5-26 标注快捷键说明

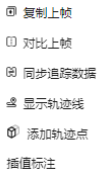
快捷键	功能
1	多边形绘制。
2	矩形绘制。
3	画笔绘制。
4	框选绘制。
Ctrl+Z	撤回上一步。
空格键	取消框选。
Enter	确认框选。
ESC	结束绘制状态。
向后滚动鼠标滚轮	画笔调整变大。
向前滚动鼠标滚轮	画笔调整变小。
O	点云图复位。

2D3D 关联标注

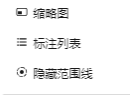
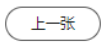
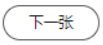
2D3D关联标注复杂，类型多样，故平台开发多种标注工具，辅助标注工作，提高标注人员标注效率。

表 5-27 标注工具说明

一级图标	二级图标	名称	功能	快捷键
	无	选择	单击3D框进入选择状态，可对3D框进行三维拖动。	无
	无	修改	双击3D框进入修改状态，可对3D框的三视图进行调整。	无
		从上至下依次是： 标注 智能缩框标注	标注：在标注类别中选择标注框，进行普通3D框标注。 智能缩框标注：可根据框选范围内点云大小智能缩进标注框	标注：V 智能缩框标注：B
		从上至下依次是: 撤销 恢复	撤销：撤销上一次操作。 恢复：恢复上一次撤销的操作。	• 撤销：Ctrl +Z • 恢复：Ctrl +Y
		从上至下依次是： 旋转 旋转90° 旋转180°	旋转点云图像。长按图标或快捷键点云图像会一直旋转。 按照箭头所示方向分别旋转点云图像90度和180度。	顺转：1或左键图标 逆转：Shift +1或右键图标 顺转90°：2或左键图标 逆转90°：Shift +2或右键图标 旋转180°：3
		从上至下依次是： 记录位置 返回记录位置 复位 点云角度保持原状	通过单击“记录位置”记录当前点云的视角，调整视角后可通过单击‘返回记录位置’，回到记录视角。 复位：回到初始点云视角。 点云角度保持原状：开启后，双击打开三视图不会改变点云视角。	记录位置：R 返回记录位置：Shift+R 复位：O

一级图标	二级图标	名称	功能	快捷键
		从上至下依次是： 复制上帧 对比上帧 同步追踪数据 显示轨迹线 添加轨迹点 插值标注	复制上帧：将上一帧标注框完全复制到当前帧。第一帧不可用复制上帧功能。 对比上帧：标注界面分为两部分，左图是上一帧点云，右图是当前帧点云，可对前后两帧信息进行对比。 同步追踪数据：选中标注物后执行该操作可将当前任务中相同ID的所有标注物的大小类型进行覆盖。 显示轨迹线：选择3D框，开启此功能后可将当前任务中与此标注物ID相同的框进行轨迹线的连接。 添加轨迹点：开启后左键添加轨迹点，右键撤销当页轨迹点，并重新规划轨迹线，此ID最后一帧不可操作。 插值标注：当需要在第n帧和第n+k帧中标注完同一个追踪对象时，可以自动计算出此对象在中间k-1帧中的标注框。	复制上帧：F
		从上至下依次为： 添加 列表	添加：选中标注物，添加提示信息后自动进入全局标注物列表。 列表：打开列表可对添加的标注物进行应用、删除、保留ID等操作。	添加：I 列表：D
	无	2D3D互转开关	开启后绘制2D、3D框后可转换出对应3D、2D框。	无

一级图标	二级图标	名称	功能	快捷键
	无	假想	预设各标注物类别的xyz参考值，选中框后单击对应参考值可直接应用。 	无
	无	高度过滤	设置高度限制，该高度以下点云无法被框选标注。	无
	无	地面点限制	开启后无法对地面点云进行标注。	无
	无	删除	选中3D框后单击图标或快捷键可删除框。	Delete
	无	长宽互换	可对3D框的xy值进行互换。	T
	无	补框	仅3D联合任务和2D3D关联标注任务点开2D缩略图时会显示此工具，当3D框缺少对应2D框时可使用此工具。	G
	无	比例尺	调整比例。	U
	全部添加 全部删除 增加单个 删除单个	从上至下依次为： 全部添加 全部删除 增加单个 删除单个	全部添加：为所有3D框添加锚点。 全部删除：删除所有3D框的锚点。 增加单个：选择单个3D框后增加单个锚点。 删除单个：删除单个3D框的锚点。 注：可通过Shift+左键单击3D框正上方角点修改锚点位置。	全部添加：Z 全部删除：X 增加单个：Shift+Z 删除单个：Shift+X

一级图标	二级图标	名称	功能	快捷键
		从上至下依次为： 缩略图 标注列表 隐藏范围线	缩略图：开启后隐藏界面下方缩略图。 标注列表：开启后隐藏标注列表。 隐藏范围线：开启后隐藏范围线。	无
无	无	标注物	标注物快捷键，最多8个，按照使用频次排序。	Alt+（1~8）
	无	下降/上升 前移/后移 左移/右移	通过单击按钮或快捷键执行平移操作。	下降：Q 上升：E 前移：W 后移：S 左移：A 右移：D
	无	上旋/下旋 左旋/右旋	无	上旋：↑ 下旋：↓ 左旋：← 右旋：→
	无	上一张	无	Shift+Q
	无	下一张	无	Shift +E
	无	保存	保存当页信息，勾选已标注/审核通过。	Ctrl+S

5.8 审核员

5.8.1 认领审核任务

在全流程的情况下，标注任务标注完毕后提交进入审核阶段（预初审-初审-预终审-终审），初审阶段（预初审-初审）提交后进入终审阶段（预终审-终审），团队中的审核员可以根据阶段认领任务进行审核。以下步骤为认领初审任务（预初审-初审）或终审（预终审-终审）任务的操作步骤。

说明

单击“认领”，认领的为审核阶段的第一个流程，如果审核阶段为“预初审-初审-预终审-终审”，则认领的为“预初审”。如果审核阶段为“初审-预终审-终审”，则认领的为“初审”。

步骤1 在左侧菜单栏中选择“标注服务 > 项目管理”。

步骤2 选择“标注项目”页签，单击项目名称。

步骤3 选择“批次任务列表”，单击批次任务前的。

步骤4 单击子任务操作栏中的“认领”，认领该任务。

步骤5 预审核的操作请参考**预审核操作指导**。

步骤6 当标注任务预审核状态为“未启动、已完成、失败”时，单击子任务操作栏中的“人工”，转为人工审核。

步骤7 单击任务名称，单击任意一张图片，进行人工审核。审核的操作请参考**审核操作指导**。

步骤8 查看审核任务统计信息。

单击“统计信息”，查看审核任务的统计信息。

- 任务详情：任务名称、所处状态、标注及审核人员等信息。
- 标注对象统计：人工/预标注任务审核对象类别及数量统计、标注数据统计。

步骤9 查看审核员审核效率。

单击“项目详情 > 人员详情”，阅读**人员详情**查看审核员审核效率。

----结束

提交审核任务

审核完毕后，审核员可提交任务交由下一步流程操作员操作。单击任务名称后“操作”一栏内的“提交”，提交任务。

说明

- 如果提交任务中有标注框存在审核错误，则无法提交。
- 提交后无法撤回，请谨慎操作。

退回任务

当任务不符合通过标准时，当前阶段操作员有权退回至上一阶段任务，单击任务名称后操作栏中的“更多 > 退回”，任务返回至上一阶段操作员名下。

说明

预初审、预终审阶段不可退回。

释放审核任务

审核员因故不能继续进行审核，可将该名下的审核任务释放，返回至项目中的任务列表中，等待其他审核员认领。单击需释放的审核任务名称后“操作”一栏内的“更多 > 释放”，将审核任务释放。

说明

- 任务被退回重新审核时，该任务不能被审核员释放。
- 预初审、预终审阶段不能被审核员释放。
- 标注团队管理员可以释放预初审、预终审阶段的任务。

5.8.2 预审核操作指导

预审核的方式是使用审核模型，生成预审核作业，对标注任务进行审核，可快速得到标注任务的质量指标，通过报告展示预审核结果。

步骤1 在左侧菜单栏中选择“标注服务 > 项目管理”。

步骤2 选择“标注项目”页签，单击项目名称。

步骤3 选择“批次任务列表”，单击批次任务前的。

步骤4 单击“状态”为“待预初审”或“待预终审”的子任务操作栏中的“更多 > 审核”，配置审核配置。

表 5-28 审核配置参数

参数	说明
任务名称	由“批次子任务名称-任务名称”组成，批次子任务名称不可修改，任务名称可自定义。
预审核任务名称	设置预审核任务名称。
审核比例	预审核默认比例为100。
审核模型	选择审核模型和版本。
标注脚本	选择标注脚本。
资源规格	当前项目中可用的资源规格。
优先级	设定任务的优先级，数值取[-50,50]的整数，数字越大，优先级越高。
可一同预审核的任务	选择需要一同审核的标注子任务。

步骤5 配置成功后，单击子任务名称，进入批次子任务详情，选择“预审核任务详情”。

说明

当批次子任务配置并启动了预审核任务时，预审核任务界面才会显示预审核任务详情。

步骤6 查看预审核任务报告

单击操作栏中的“报告”，可查看或下载任务报告。

----结束

预审核任务相关操作

在“预审核任务”列表，还可以进行以下操作。

表 5-29 预审核任务相关操作

任务	操作步骤
查询预审核任务	在搜索输入框中输入搜索条件，按回车键即可查询任务。
查看模型	单击模型名称，界面跳转至模型详情，可查看模型。
查看任务日志	单击操作栏中的“日志”，可查看或下载日志详情。
删除任务	• 选择单个任务，单击操作栏的“删除”，删除单个任务。 • 勾选多个任务，单击列表上方的“批量删除”，可批量删除任务。

5.8.3 审核操作指导

审核是检验标注任务质量的一个手段，审核的方式是从标注任务中抽取一定比例的图片，通过被抽取的图片标注准确率判断该标注任务整体准确率，该标注任务是否通过审核。

步骤1 在左侧菜单栏中选择“标注服务 > 项目管理”。

步骤2 选择“标注项目”页签，单击项目名称。

步骤3 选择“批次任务列表”，单击批次任务前的。

步骤4 单击子任务操作栏中的“更多 > 审核”，配置审核比例。

审核比例：从该标注任务所有样本中抽取一定样本作为审核样本。

步骤5 配置成功后，单击标注任务名称，单击任意一张图片，进行审核。

步骤6 审核图片标注符合规范，整张图片无错误，则该图片审核通过，勾选“审核通过”，并保存。

步骤7 审核图片有超过一处标注不符合规范，则该图审核未通过。根据标注任务类型不同，右键标注对象，勾选不同的审核属性，或者填写详细的审核错误描述。

步骤8 审核完毕后，根据实际情况勾选审核结果，并保存。

----结束

5.9 验收员

5.9.1 认领验收任务

按照全流程计划，标注任务审核完毕后，进入标注任务的下一个阶段依次验收（初验-终验）。验收员依据标注规范和一定验收标准对标注任务能否通过验收进行把关。验收通过，则该标注任务完成。验收不通过，验收员将标注任务退回至绑定的审核员名

下，审核员重新审核该任务，再退回至标注员重新标注。根据验收员认领任务所处流程区分验收员是初验员还是终验员（同一验收员不可同时担任同一任务的初验员和终验员）。

验收员可以认领未绑定验收员的待验收任务，一个验收任务只能由一个验收员认领。以下步骤为认领初验或终验任务的操作步骤。

步骤1 在左侧菜单栏中选择“标注服务 > 项目管理”。

步骤2 选择“标注项目”页签，单击项目名称。

步骤3 选择“批次任务列表”，单击批次任务前的。

步骤4 单击子任务操作栏中的“认领”，认领该任务。

步骤5 单击任务名称，单击任意图片，查看标注和审核结果。

----结束

提交验收任务

验收完毕，且任务通过验收，验收员可提交任务，表示标注任务可进入下一步流程。

退回任务

当任务不符合通过标准时，当前阶段操作员有权退回至上一阶段任务，单击任务名称后操作栏中的“更多 > 退回”，任务返回至上一阶段操作员名下。

释放验收任务

验收员因故不能继续验收时，可将名下的验收任务释放，由其他验收员领取。单击需释放的验收任务名称后“操作”一栏内的“更多 > 释放”，释放该任务。

查看和下载任务日志

标注任务运行的过程中生成任务日志，平台提供了日志的查看以及下载功能。单击任务名称，在该任务的详情页面，单击 " 日志 " ，可查看该任务日志列表及日志详情。支持下载至本地。如果日志较多，可在搜索框中输入关键字，查找指定日志内容。搜索内容为当前已加载内容 ，最多为1M（首次加载时）。如图，在日志服务页面中的日志列表部分详细展示了该任务包含的日志文件的大小以及最近写入时间。单击文件后的 " 查看 " ，该文件的详细执行过程则在日志详情部分展示。也可在日志文件后的 " 操作 " 栏，单击 " 下载 " ，即可将该日志文件下载到本地查看。

5.10 标注样例

Octopus标注平台功能众多，类型复杂。为帮助用户更快适应标注平台功能，熟悉标注平台界面，开展业务。本节以较典型的标注任务为例，详细介绍标注平台对不同标注任务的操作指导，用户可参考操作指导，依据标注规范快速在平台展开标注。

5.10.1 人车类型图片标注任务

人车类型标注任务主要是对真实路采图片中出现的人物、车辆等进行标注。

绘制对象

步骤1 绘制矩形框。

选择矩形图形工具（快捷键3，非小键盘）。

在标注列表中选择需要标注的类别（非必要，也可等标注完成后，右键修改类别）。

单击选择的第一个点，移动鼠标选择需要绘制的第二个，再次单击结束。

步骤2 绘制垂直线。

同选择矩形框工具方式与类别一致选择直线图形工具（快捷键2，非小键盘）。

长按键盘Shift，单击直线开始点，再单击直线结束点，可以绘制成一条垂直线。

同理操作可绘制水平线 。

步骤3 绘制点。

同选择矩形框工具方式与类别一致选择点图形工具（快捷键1，非小键盘）。

左键单击即可绘制点图形 。

----结束

修改标注框

步骤1 修改矩形框。

左键单击矩形框的某条边拖动可修改这个边的位置，左键单击某个角点拖动可以修复角点位置。

左键选中某条边，或者某个角点，可以通过h（←）j（↓）k（→）u（↑）调整像素框1px。

步骤2 修改目标类别。

左键单击目标图形，可进入选择类别的跳出框，即可修改类别。

步骤3 修改额外属性。

左键单击目标图形，如果目标含有额外属性，如果其默认属性错误，单击即可选择属性。

步骤4 修改对象ID。

左键单击目标图形，可以在对象ID栏手动输入数值来修改ID。

----结束

重叠框选择功能

如果出现框完全覆盖无法单击框的情况，可以通过选择标题头位置来选择对应的框。

图 5-4 重叠框选择功能

快捷键使用说明

Octopus平台提供快捷键使用，用户可以利用快捷键快速完成标注，提高标注效率。

表 5-30 人车任务快捷键使用说明

快捷键	功能
ESC	修改状态
shift+B	比例尺
~	选择移动工具（抓手模式）
U、J、H、K	选中边框上下左右微调
Delete	删除选中目标
Q、E	上下翻页
Ctrl+C	复制属性
Ctrl+V	粘贴属性
Ctrl+Z	撤回上一步操作
S	保存本页标注
1	绘制点
2	绘制实线

快捷键	功能
3	绘制矩形
4	绘制obb矩形
G	快捷添加分界线和点
D	全局标注，D启用，选中标注物后i添加
L	隐藏标注
>	勾选审核通过并切到下一页

5.10.2 2.5D 人车图片标注任务

2.5D人车图片标注任务相比于2D人车标注任务，由2D的矩形框转变为2.5D框，可以定位车辆车身的正面与侧面，辅助开发者辨别车辆的行驶方向。

绘制对象

步骤1 单击2.5D图片标注任务，选择一张图片进入人工标注。

步骤2 选择标注工具。

单击左侧工具栏2.5D标注工具。

步骤3 选择标注类别。

标注下拉列表页选择一标注名，进入标注状态。

步骤4 绘制标注框。

单击选择的第一个点，移动鼠标并单击绘制正面框，再次移动鼠标并单击绘制侧面框。

步骤5 绘制点。

单击左侧标注工具栏，选择点（快捷键1，非小键盘）。

左键单击即可绘制点图形。

步骤6 修改标注框。

左键单击2.5D框的某条边拖动可修改这个边的位置，左键单击某个角点拖动可以修复角点位置。

步骤7 修改标注类别。

左键单击目标图形，可进入选择类别的跳出框，即可修改类别。

步骤8 修改额外属性。

左键单击目标图形，如果目标含有额外属性，如果其默认属性错误，单击即可选择属性。

步骤9 修改对象ID。

左键单击目标图形，可以在对象ID栏手动输入数值来修改ID。

----结束

5.10.3 点云标注任务

点云标注任务是指根据标注规范对点云图像中出现的车、人等标注物进行标注。点云距离单位默认为米。

绘制对象

步骤1 单击点云标注任务，单击任意一帧，进入人工标注。

步骤2 选择标注工具。

单击左侧工具栏“智能缩框标注”（快捷键B）。

步骤3 选择标注类别。

标注下拉列表页选择一标注名，进入标注状态。

步骤4 框选标注物。

步骤5 调整三视图。

依据标注规范要求，结合下方真实图片中对应标注物大小，调整点云图像中标注物三视图中标注框。

步骤6 调整2D框（只适用于联合标注）。

1. 单击2D图片。
2. 开启联合按钮。

图 5-5 开启联合按钮

3. 针对点云框对应的2D框进行调整大小。

步骤7 修改标注类别。

左键单击目标图形，可进入选择类别的跳出框，即可修改类别。

步骤8 修改额外属性。

左键单击目标图形，如果目标含有额外属性，单击即可选择属性。

步骤9 修改对象ID。

左键单击目标图形，可以在对象ID栏手动输入数值来修改ID。

----结束

快捷键使用说明

Octopus平台提供快捷键使用，用户可以利用快捷键快速完成标注，提高标注效率。

表 5-31 快捷键使用说明

快捷键	功能
Ctrl+S	保存
Ctrl+Z	撤销
Ctrl+Y	恢复
O	点云图复位
V	默认标注
B	智能标注
X	添加锚点
Shift+X	删除锚点
Shift+D/delete	删除
F	追踪上一帧
T	长宽互换
R	顺时针旋转俯视图
Shift+R	逆时针旋转俯视图
Shift+E	进入下一页
Shift+Q	进入上一页
>	勾选审核通过并切到下一页

5.10.4 点云跟踪标注任务

绘制对象

步骤1 单击点云标注任务，单击任意一帧，进入人工标注。

步骤2 选择标注工具。

单击左侧工具栏“智能缩框标注”（ 快捷键B ）。

步骤3 选择标注类别。

标注下拉列表页选择一标注名，进入标注状态。

步骤4 框选标注物。

步骤5 调整三视图。

依据标注规范要求，结合下方真实图片中对应标注物大小，调整点云图像中标注物三视图中标注框。

步骤6 调整2D框（ 只适用于联合标注 ）。

1. 单击2D图片。
2. 开启联合按钮。
3. 针对点云框对应的2D框进行调整大小。

步骤7 修改标注类别。

左键单击目标图形，可进入选择类别的跳出框，即可修改类别。

步骤8 修改额外属性。

左键单击目标图形，如果目标含有额外属性，单击即可选择属性。

步骤9 修改对象ID。

左键单击目标图形，可以在对象ID栏手动输入数值来修改ID。

步骤10 重复以上标注步骤，直至全部标注完成，切换下一张。

步骤11 打开追踪双屏显示。

步骤12 使用追踪快捷键F。

步骤13 调整点云框三视图，使其能完整包裹点云。

步骤14 右击目标图形，修改对象ID，保证同一辆车在前后帧ID一致。

----结束

连续帧插值工具

当需要在第n帧和第n+k帧中标注完同一个追踪对象时，可以自动计算出此对象在中间k-1帧中的标注框。具体步骤如下：

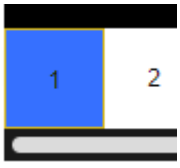
步骤1 选中左侧工具栏“追踪 > 插值标注 > 线性插值/AI插值”，开启插值模式。

插值标注开启中，可通过左侧图标切换或使用快捷键切换插值计算模式（线性插值/AI插值）。

步骤2 在第n帧选中一个追踪对象的标注框。

步骤3 按c键将当前帧设置为关键帧。

图 5-6 设置关键帧



步骤4 向后切至第n+k帧。

步骤5 选中同一个追踪对象（对象id相同）的标注框，如无该对象的标注框，则先使用标注框工具进行标注。

步骤6 按c键将第n+k帧设置为关键帧，此时，平台自动计算中间帧关于此追踪对象的标注框，计算过程中的帧出现计算进度圈。

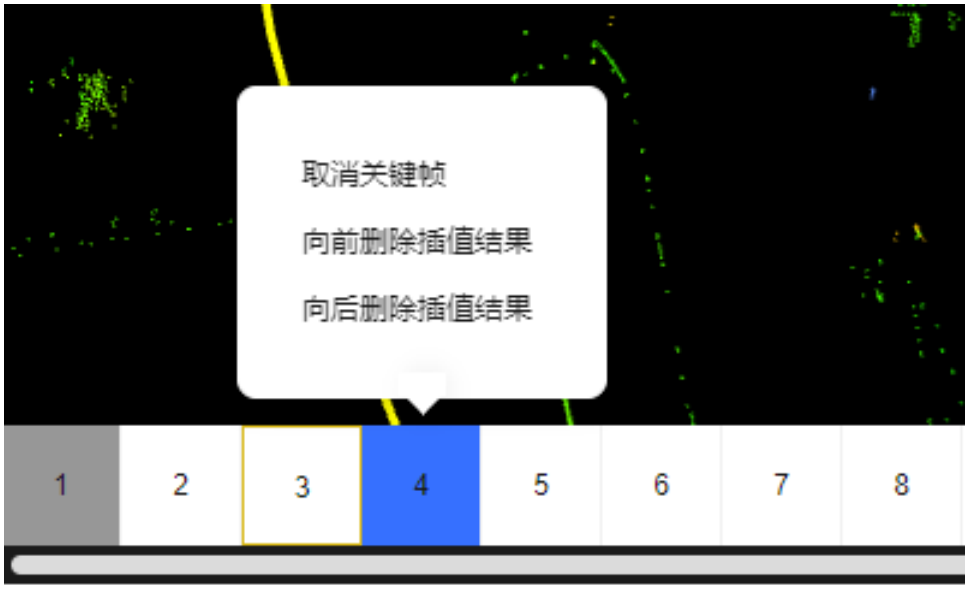
计算结束，进度圈消失，（注：由于线性插值计算较快，基本看不到进度圈）。

如果中间帧的进度圈消失，即可对自动插值结果进行人工修正。

图 5-7 人工修正

步骤7 如果对插值结果或关键帧不满意，可向前或向后一键删除插值结果或取消关键帧。

图 5-8 取消关键帧



步骤8 重复步骤步骤4-步骤6操作，继续向后设置关键帧进行插值计算。

步骤9 单击左侧工具栏“追踪 > 插值标注”， 关闭插值模式。

说明

- 同一个追踪对象，需在最右侧关键帧向由右设置新关键帧，或在最左侧关键帧向左设置新关键帧。
- 当前追踪对象插值结果计算未结束，不可设置其他追踪对象的关键帧。
- 前后关键帧跨度不可超过20。
- 在非无痕模式下使用该工具。

----结束

5.10.5 车道线图片标注任务

车道线图片标注任务是指依据标注规范对真实路采图片中出现的道路中的车道线、斑马线等交通线路进行标注，一般区分实线、虚线，按需求增加颜色、遮挡程度等额外属性。

绘制对象

步骤1 单击车道线标注任务，选择一张图片进入人工标注。

步骤2 绘制对象。

单击左侧工具栏实线（快捷键2，非小键盘），进入绘制折线模式。

步骤3 选择标注名称。

标注下拉列表页选择一标注名，进入标注状态。

步骤4 绘制折线。

通过鼠标左键单击添加线段上的点，鼠标左键双击闭合该线段。未闭合状态可通过alt+z撤销上一个绘制的点。闭合状态下选中线段上的某点可通过“alt+鼠标左键”或“alt+v”删除该点。绘制过程中可通过鼠标左键拖动图片。

步骤5 修改折线。

鼠标左键选中线段上某点可拖动修改点的位置。Ctrl+Z可撤销上一步操作。

步骤6 修改类别。

左键单击目标图形，可进入选择类别的跳出框，即可修改类别。

步骤7 修改额外属性。

左键单击目标图形，如果目标含有额外属性，如果其默认属性错误，单击即可选择属性。

步骤8 修改对象ID。

左键单击目标图形，可以在对象ID栏手动输入数值来修改ID。

----结束

快捷键使用说明

Octopus平台提供快捷键使用，用户可以利用快捷键快速完成标注，提高标注效率。

表 5-32 快捷键使用说明

快捷键	功能
ESC	修改状态
2	实线绘制工具
Alt+z	未闭合状态取消上一点
Alt+鼠标左键	去除点
Delete	删除标注对象
Ctrl+Z	撤回上一步操作
S	保存本页标注
L	隐藏标注
>	勾选审核通过并切到下一页

5.10.6 语义分割图片标注任务

语义分割任务是指根据标注规范将待标注图片中出现的天空、道路、车辆等类标注物进行标注。

绘制对象

步骤1 绘制多边形。

选择左侧工具栏多边形按钮绘制多边形。

步骤2 选择标注。

标注列表页选择符合的标注。

步骤3 绘制标注形状。

绘制标注形状有两种方法：点式绘制和交互式分割工具，用户可根据需要选择绘制方式

1. 点式绘制形状：通过单击目标对象的边缘，点状连线，获得闭合的多边形，操作如下：

   通过鼠标左键单击添加点标注图中对象，未闭合状态可通过“alt+Z”撤销上一步绘制的点，标注过程中可通过“alt+鼠标左键”拖动图片。按住键盘上空格键进行闭合，该对象标注完成。

   说明

   项目类型为语义标注、道路特征提取、可行驶区域，这三类任务是空格键闭合且alt+鼠标左键拖动图片，其余任务使用左键拖动图片和闭合多边形。

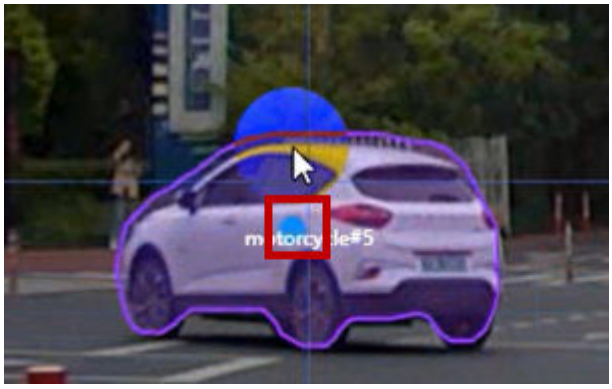
2. 交互式分割工具（仅在“语义标注”项目中使用）：通过在标注对象及附近区域单击正点(目标对象区域的点)或负点（非目标对象区域的点）， 自动得到较优的多边形。具体步骤如下：

   a. 单击快捷键“4，非小键盘”，进入多边形绘制状态。

   b. 单击快捷键“z”键进入目标对象的交互式标注模式， 页面右上角出现“交互式分割已开启”字样。

   c. 在前景区域，通过鼠标左键单击正点（目标对象区域的点），得到预测的多边形。

   图 5-9 单击正点

   d. 在背景区域，通过鼠标左键单击负点（非目标对象区域的点），修正多边形。在多边形区域未包含的前景区域左键单击正点，修正多边形。

图 5-10 单击负点

e. 重复单击正或负点，修正预测的多边形，直到多边形达到较好的效果。

图 5-11 修正预测的多边形

说明

一般情况下，单击3-5个点即可达到较好效果，如果超过10个点还未达到较好效果，则此对象的此次交互式标注基本无效，建议退出。

f. 再次单击快捷键“z”键，可退出目标对象的交互式标注模式， 页面右上角“交互式分割已开启”字样消失。该对象标注完成。

步骤4 切割线工具修改多边形。

单击左侧标注工具栏切割线（快捷键8，非小键盘），切割线空格键闭合，切割线绘制完成通过键盘上“m”键可完成切割。

切割线可用于切割多边形不贴合部分，切割线首尾两点与多边形需有交点。

步骤5 其他常用标注工具。

- 绘制共边对象：通过键盘上“X”键实现两对象共边。

图 5-12 绘制共边对象

图 5-13 绘制共边对象效果

- 全局标注：选中一标注对象，单击左侧工具栏“全局标注物”按钮，或通过键盘快捷键“i”复制该对象到图层中，选中该对象可应用到本图或者下一张图中。该步骤可用于有相同对象的图片，或者有覆盖类型的标注中。

图 5-14 全局标注

- 删除对象：选中一对象，通过键盘上“delete”键可在图中删除该对象。
- 撤回上一步：单击左侧工具栏“撤回”，或通过快捷键“Ctrl+z”撤回上一步。
- 隐藏标注列表和预览图：单击左侧工具栏“隐藏标注列表”、“隐藏预览”，扩大标注区域。
- 调整透明度：单击左侧工具栏“调整透明度”，调整图片。

步骤6 更改标注类别。

多边形闭合后可在该对象上单击鼠标左键，更改该对象类别。

步骤7 修改额外属性。

左键单击目标图形，如果目标含有额外属性，如果其默认属性错误，单击即可选择属性。

步骤8 修改对象ID。

左键单击目标图形，可以在对象ID栏手动输入数值来修改ID。

----结束

快捷键使用说明

Octopus平台提供快捷键使用，用户可以利用快捷键快速完成标注，提高标注效率。

表 5-33 快捷键使用说明

快捷键	功能
空格键	多边形闭合。
X	多边形共边。
4（非小键盘）	进入多边形绘制工具。
Alt+z	未闭合状态撤销多边形点。
Alt+V/alt+鼠标左键	删除多边形上某点。
Alt+鼠标左键/鼠标滚轮	拖动语义多边形图片。
8（非小键盘）	进入切割线模式。

快捷键	功能
M	绘制切割线后切割。
Delete	删除标注对象。
Shift+b	比例尺。
Ctrl+Z	撤回上一步操作。
S	保存本页标注。
D	全局标注模式。
T	隐藏标题。
L	隐藏标注。
>	勾选审核通过并切到下一页。
Z	进入/退出目标对象的交互式分割模式。
交互式模式下鼠标左键	单击正点。
交互式模式下鼠标右键	单击负点。

5.10.7 语义分割点云标注任务

语义分割任务是指根据标注规范将待标注点云图像中出现的天空、道路、车辆等类标注物进行标注。

绘制对象

步骤1 单击大规模3D语义分割任务，单击任意一帧，进入人工标注。

步骤2 单击左侧标注工具栏，选择对应的标注工具。

步骤3 选择对应的标注类别。

步骤4 绘制标注物。

步骤5 修改标注物。

将其他标注物涂成错误标签，应选择该标注所在图层，然后重新选择正确的标注类别绘制。

步骤6 标注车道线。

打开强度屏，辅助标注车道线。

----结束

快捷键使用说明

Octopus平台提供快捷键使用，用户可以利用快捷键快速完成标注，提高标注效率。

表 5-34 快捷键使用说明

快捷键	功能
1	多边形绘制。
2	矩形绘制。
3	画笔绘制。
4	框选绘制。
Ctrl+Z	撤回上一步。
空格键	取消框选。
Enter	确认框选。
ESC	结束绘制状态。
向后滚动鼠标滚轮	画笔调整变大。
向前滚动鼠标滚轮	画笔调整变小。
O	点云图复位。

5.10.8 2D3D 关联标注任务

2D3D关联任务是指根据标注规范将待标注点云图像和图片中出现的天空、道路、车辆等类标注物进行标注，然后自行关联。

绘制对象

步骤1 单击2D3D关联任务，单击任意一帧，进入人工标注。

步骤2 平台依据新建的2D或3D框自动转成对应的3D或2D框。开关状态在同一任务中继承。

步骤3 选择标注类别。标注下拉列表页选择一标注名，进入标注状态。

步骤4 单击左侧标注工具栏，选择对应的标注工具。

- 在图片界面单击左侧工具栏“补框”，绘制2D框，此时右侧标注列表只展示关联对象的标注物。键盘按键“ESC”退出绘制状态，右侧标注列表展示所有的标注物。
- 在点云界面单击左侧工具栏“AI标注”（快捷键b），框选3D，此时右侧标注列表只展示关联对象的标注物。双击空白处退出绘制状态，右侧标注列表展示所有的标注物。

步骤5 双击3D框，打开三视图。

依据标注规范要求，结合下方真实图片中对应标注物大小，调整点云图像中标注物三视图中标注框。

步骤6 修改标注类别。

右键单击目标图形，可进入选择类别的跳出框，即可修改类别。

步骤7 修改额外属性。

右键单击目标图形，如果目标含有额外属性，单击即可选择属性。

步骤8 修改对象ID。

右键单击目标图形，可以在对象ID栏手动输入数值来修改ID，由此将2D框和3D框数据手动关联。

----结束

快捷键使用说明

Octopus平台提供快捷键使用，用户可以利用快捷键快速完成标注，提高标注效率。

表 5-35 快捷键使用说明

快捷键	功能
Ctrl+S	保存。
Ctrl+Z	撤销。
Ctrl+Y	恢复。
O	点云图复位。
Shift+D/delete	删除。
b	进入3D画框绘制状态。
ESC	退出绘制。
R	顺时针旋转俯视图。
Shift+R	逆时针旋转俯视图。
E	进入下一页。
Q	进入上一页。

5.10.9 语音标注任务

语音标注任务是指对单段落语音或多段落进行语音标注，提供单段落和多段落语音标注模板工具，通过管理平台进行标签和属性的配置，支持智能语音交互场景下每一个细分环境的标注，也支持整个交互闭环的整体标注。支持讲话人属性（性别、角色等）配置，支持意图识别、问答标注和设备状态标注能力。当前支持的语音类型包括：wav、flac、mp3和m4a。

绘制对象

步骤1 单击语音标注任务名称，选择任意一个文件进入语音标注模板工具进行人工标注。

步骤2 播放音频。

单击播放按钮  或音频展示区，播放音频。

步骤3 添加标注。

按下鼠标左键在音频展示区域拖动，松开鼠标后，标注列表中将增加一行标注信息。

步骤4 修改标注类型。

单击“标注类型”，选择合适的标注物。

步骤5 完善标注信息。

选择音频的性别和额外属性，填写音频的内容和讲话人角色信息。

- 性别：选择讲话人的性别。
- 内容：描述当前讲话的内容。
- 角色：判断当前讲话人的角色。
- 属性：选择音频的额外属性。

步骤6 修改标注音频。

鼠标悬浮到已标注的音频段落前后，出现方向箭头后，拖动标注框，可以延长或缩短标注音频段落。

步骤7 播放音频段落。

单击标注任务列表操作栏的“播放”，可以播放当前标注的音频段落。

步骤8 删除音频段落。

单击标注任务列表操作栏的“删除”，可以删除当前标注的音频段落。

步骤9 审核音频段落。

单击标注任务列表操作栏的“审核”，可以审核当前标注的音频段落。

步骤10 调整播放倍速。

拖动播放倍速进度条，调整播放倍速。调整范围为(0.1,10]。

步骤11 设置前后静音段。

勾选前后静音段的前面的复选框后，即可在进度条上设置前静音段和后静音段。

步骤12 快进快退播放音频。

单击快进或快退按钮，快速定位音频播放时间点。

步骤13 查看标注物列表和快速搜索标注物。

单击悬浮框中的Open按钮，可以展开标注物列表，查看当前支持的所有标注物；打开Search开关后，可以输入关键字快速搜索标注物。

展开标注物列表，支持过滤搜索。

----结束

5.10.10 文本标注任务

文本标注任务是指单段文本进行标注，支持分词标注、实体标注、实体关系标注、意图标注及文档属性标注。提供文本标注模板工具，且可以通过管理后台进行标签和属性的配置；支持自定义标签属性及默认标签属性模板；支持多种标签展示类型；支持字体大小和行间距设置；提升项目功能完整性和标注服务在文本标注场景下的竞争

力。当前支持文件类型包括：txt、yaml、csv、xml，文件编码仅支持UTF-8。单个文本不超过100M。

绘制对象

步骤1 单击文本标注任务名称，选择任意一个文件进入文本标注模板工具进行人工标注。

步骤2 设置文档属性和意图。

单击“文档属性”，增加文档的属性描述。单击“意图”，增加文档意图描述；

步骤3 添加标注。

在文本展示区域，按下鼠标左键在文本展示区域拖动，松开鼠标后，标注列表中将增加一个标注信息。

步骤4 修改标注类别。

鼠标右键单击已标注的文本，打开标注类型弹窗，修改标注类型。可以为标注文本增加额外属性和审核属性，当标注有问题，可以在审核错误描述中增加错误描述。

步骤5 删除标注。

在标注列表，单击目标标注后面的  ，即可删除当前标注，文本标注模板工具中的标注同步删除。

步骤6 设置关系标注。

文本默认为实体标注，在文本标注模板工具左上角，单击“实体标注”开关，可以切换为“关系标注”。在进行关系标注之前，需要提前在关系管理列表中创建关系类别，详情操作请参考**5.2.2.5 关系管理**。

在“关系标注”状态下，先单击一段已标注文本，再选中另一段已标注文本，即可建立两段标注文本之前的关系。

鼠标右键单击关系类别标签，打开类别弹窗，修改关系类别。

步骤7 删除关系。

在关系列表，单击目标关系后面的  ，即可删除当前关系，文本标注模板工具中的关系同步删除。

----结束

6 训练服务

6.1 训练服务简介

训练服务模块上承接数据服务和标注服务两大模块，为自动驾驶研发提供方便易用的模型训练和评测平台，让用户无需过多关注底层资源，聚焦算法和模型开发。用户可上传符合Octopus平台规范的训练算法，使用成熟的算法创建训练任务生成训练模型。此外，训练服务提供多种模型评测指标，从多维度衡量模型质量。让自动驾驶研发更便捷。训练服务的开发流程如下：

训练服务操作引导如下：

- **算法管理**：负责管理用户上传的符合平台规范的算法。
- **开发环境**：开发环境依托于已购买的集群算力，支持客户在浏览器窗口或线下IDE中进行交互式的算法或模型的开发和调试。
- **训练任务**：用户选择训练算法和训练数据集创建训练任务进行训练。
- **模型评测**：负责管理评测脚本、评测任务和评测对比任务。
- **编译管理**：包含编译任务和编译镜像。训练产生的模型版本，一般不可直接被车载芯片识别，需要经过编译工具，模型编译成车载芯片识别的产物。
- **推理服务**：将模型部署为在线服务进行推理。
- **任务队列**：展示在训练服务创建的所有类型任务，包括任务的名称、类型、资源规格、实例数、优先级、工作空间、创建者、创建时间等信息。

6.2 算法管理

6.2.1 训练算法

平台支持算法创建。用户可通过指定算法的运行镜像和上传符合平台规范的算法文件来完成算法的创建，创建的算法可用于训练任务中。创建训练算法时可根据算法类型单独上传训练文件或将训练文件放置在镜像中，通过选择对应镜像时获取算法文件。

添加算法

步骤1 准备数据。

1. 准备用途为“训练/评测”的镜像和版本，详情可参考**9.1 镜像仓库**。

2. 准备训练算法文件，文件详情可参考**6.2.2 算法文件说明**。

步骤2 在左侧菜单栏中选择“训练服务 > 算法管理”。

步骤3 单击“新建训练算法”，填写算法的名称和描述信息，其他参数参考如下。

- 如果类型选择“仅镜像（包含算法文件）”，参数填写完毕后，勾选服务声明，单击“创建”即可创建算法。
- 如果类型选择“镜像 + 外置算法文件”，参数填写完毕后执行**步骤4**。

表 6-1 新建训练算法

参数	说明
类型	训练算法的类型。 ● 镜像 + 外置算法文件：训练算法文件与镜像独立存在，创建训练算法时需选择镜像，并上传准备好的训练算法文件，训练任务启动时会把文件下载到容器中，并把算法根目录作为工作目录。 ● 仅镜像（包含算法文件）：训练算法文件包含在镜像中，需提前将训练算法文件放入选择的镜像中。
镜像	选择用途为“训练/评测”的镜像和版本，作为训练任务的容器镜像。
共享级别	算法的共享级别。 ● 个人：当前操作用户。 ● 团队：当前工作空间下被授权的用户。
样本类型	使用的样本类型。 ● 图片 ● 3D点云
启动指令	训练算法的启动指令。 ● 对于“镜像 + 外置算法文件”类型的算法，容器默认的工作目录为**算法文件的根目录**。 示例：如果算法文件中有一个名为main.py的训练入口文件，启动指令可以是**python -u main.py**，指令中仅需指定文件的相对路径即可。 ● 对于“仅镜像（包含算法文件）”类型的算法，默认工作目录与**镜像中定义**的保持一致。 启动指令中指定的文件可使用完整的绝对路径，如**python /home/ma-user/main.py**， 也可以使用相对于镜像工作目录的相对路径。
参数列表	可以自定义boot文件的启动参数，需要在算法中定义。单击“新增参数”，填写key和value。
参数命令	参数命令为运行启动脚本的shell命令，由配置的“启动指令”和参数列表中参数及参数值自动生成。
环境变量	通过注入环境变量至容器中，用户可以快速获取业务相关常量。允许添加的环境变量个数不超过20个。单击“新增参数”，填写key和value。

步骤4 单击“下一步”，完成算法的初始化过程。

步骤5 选择训练算法文件。

将本地算法文件夹拖入框中，或单击此处选择本地文件夹，算法管理会将该文件下的所有子文件上传。

步骤6 单击“上传”，等待算法创建成功。

说明

- 上传中刷新或关闭浏览器会导致文件上传中断，请谨慎操作！
- 算法状态处于“创建中”时，如果提交的文件未上传完成，且超过2分钟无上传进程，此时算法状态置为“上传中断”，超过2小时上传中断后，算法状态置为“创建失败”。

----结束

训练算法相关操作

在“算法管理”列表，可对训练算法进行以下操作。

表 6-2 训练算法相关操作

任务	操作步骤
查找算法	选择“算法名称”、“镜像仓库ID”、“状态”、“共享级别”，在搜索输入框中输入搜索条件，按回车键即可查询。
查看算法详情	单击算法名称，查看算法详情。
删除算法	选择操作栏的“更多 > 删除”，删除算法。以下状态不支持删除： ● 算法状态：创建中、删除中。 ● 打包状态：打包中。
编辑算法	单击操作栏的“编辑”，支持修改除“名称”外的所有属性。
上传文件	● 算法状态为“初始化”、“上传中断”、“创建成功”时可上传文件或断点续传： 选择操作栏内的“更多 > 上传文件”，上传本地算法文件。 ● 算法已上传成功后需要追加上传文件时： 选择操作栏内的“更多 > 追加上传”，追加上传本地算法文件。
打包文件	算法状态为“创建成功”且打包状态不为“打包中”的自定义算法可打包文件。 单击操作栏的“打包”，即可打包算法。
下载算法	当算法的打包状态为“打包成功”时，选择操作栏的“更多 > 下载”，即可下载算法。
在线编辑	单击在线编辑栏的“在线编辑”，可在线编辑算法文件，具体请参考**在线编辑算法**。

在线编辑算法

平台提供算法编辑器，在创建成功的算法名称后单击“在线编辑”，或单击算法详情页右上角的“算法编辑”，进入该算法的在线编辑页面。界面左侧显示的是该算法包内的所有算法文件，以目录树的形式展示，支持编程语言的渲染，支持MarkDown文件的实时双屏预览。

- 上传文件：单击 ，用户可以将本地文件夹或者文件上传到算法指定目录。
- 新增文件夹：选中文件夹并单击 ，用户将新建一个该文件夹的子文件夹。选中工程文件并单击 ，将会新建一个新的文件夹，与用户已有的文件夹同级。
- 新增文件：单击，或鼠标放置在文件夹目录上，单击“新增文件”，用户可以新增文件。
- 修改文件：单击，用户可对文件名称进行修改。
- 删除文件：单击，用户可删除文件。

  说明

  - 文件名称不能为空且只能包含字母、中文字符、数字、感叹号、连字符、下划线、句点、星号、单引号和括号，且不能以两个句点开头。
  - 删除后不可恢复，请谨慎操作。
- 配置界面：单击 ，按照喜好配置界面基本属性，查看快捷键说明。
- 删除当前算法文件：单击“删除”，删除当前页面的算法文件。删除后不可恢复，请谨慎操作。
- 保存算法文件：单击“保存”，保存当前算法。算法更新完毕请及时保存。

6.2.2 算法文件说明

上传到Octopus平台的本地算法文件包需要满足Octopus平台要求，本章节介绍算法文件基本要求及相关环境变量说明。

算法文件基本要求

算法文件目录结构可参考如下，需要包括启动文件“xxx.py”（启动文件名可自定义），以及一些必要的训练文件。

- 启动文件（必选）

  算法的启动文件，直接填写相对路径，如 “main.py” 或“tools/main.py”。
- 需要编译的依赖（可选）

  如果使用了第三方的需要编译的算法库，将编译脚本或编译产物或依赖库添加到算法文件根目录下。推荐将通用依赖编译安装操作放在算法绑定的用户自定义镜像。

  不支持动态联网安装依赖，包括但不限于apt/apt-get/pip等命令，建议提前安装在自定义镜像中。

相关参数说明

表 6-3 相关参数说明

名称	环境变量	默认值	备注
数据集目录	DATASET	• CCE: /tmp/data/dataset • ModelArts: /home/ma-user/datasets	数据集在训练任务容器中的存放路径，可自行获取各种数据集信息。
数据集映射	DATASET_MAP	{key1: value1, key2: value2}	以键值对提供数据集名称和容器内路径的变量，其中容器内路径参考数据集目录
数据缓存目录	CACHED_DATASET	• CCE: /tmp/data/cached-dataset • ModelArts: /home/ma-user/cached-datasets	缓存数据集在训练任务容器中的挂载路径。
模型存放目录	RESULT	• CCE: /tmp/result • ModelArts： /home/ma-user/modelarts/result	训练产物的存放路径，产物输出到此路径后，在任务结束时。可在任务详情页的输出模型版本中浏览及执行各种操作。
增量训练模型目录	MODEL	• CCE: /tmp/data/model • ModelArts: /home/ma-user/modelarts/user-job-dir/model	待增量模型版本在训练任务中的存放路径，可自行获取模型文件信息。

如果平台支持多类型资源池，**建议用户使用环境变量适配算法提交任务**，可免去更换默认值的环节。

示例：若任务的“数据输入”中选择了名为“增量数据集1”和“增量数据集2”的两个“数据集”，和名为“全量数据集缓存”的“数据缓存”。

- DATASET_MAP取值：
  ```
  { "增量数据集1": "/home/ma-user/datasets/dataset-0", "增量数据集2": "/home/ma-user/datasets/dataset-1", "全量数据集缓存": "/home/ma-user/cached-datasets/12fdb1cd-6afa-4e63-b031-7484b95df74f/dataset" }
  ```
- 使用环境变量获取以上变量：
  ```
  # 获取数据集根目录
  dataset_dict= json.loads(os.getenv("DATASET_MAP"))
  # 训练任务最多可使用5个数据集，通过迭代方式获取每个数据集路径
  DATASETS_DIR_LIST = [os.path.join(DATASET_DIR), path for path in os.listdir(DATASET_DIR)]
  ```

分布式训练环境变量说明

表 6-4 相关参数说明

名称	环境变量	示例值	备注
节点地址	VC_WORKER_HOSTS	modelarts-job-8f3a5ced-d3d0-486fa236-19a075ec7259-worker0.modelarts-job-8f3a5ced-d3d0-486fa236-19a07	包含多个节点HOST地址，用逗号分隔，一般取第一个作为主节点通信地址。
主节点通信端口	VC_WORKER_PORT	6789	主角点通信端口，用户可从环境变量获取或自设定，推荐使用os.getenv带默认参数的方法，例如port=int(os.getenv("VC_WORKER_PORT", "6789"))。
计算节点数量	MA_NUM_HOSTS	8	表示计算节点数量，8表示当前任务使用8个节点。
单节点GPU数量	MA_NUM_GPUS	8	表示每个节点有8张GPU卡。
节点编号	VC_TASK_INDEX	0	若选择8节点，则节点编号从0~7排列，0号节点称为主节点。

分布式训练启动脚本参考示例如下，其中train.py为原算法启动脚本。

```
# -*- coding: UTF-8 -*-
# Copyright (c) Huawei Technologies Co., Ltd. 2012-2024. All rights reserved.
import argparse
import os
import subprocess
import time
import sys
import torch


print(torch.cuda.is_available())

# 获取分布式训练的相关环境变量
MASTER_HOST = os.getenv("VC_WORKER_HOSTS")
MASTER_ADDR = MASTER_HOST.split(",")[0]
# 通信端口建议自行设定，若使用环境变量需要确保包含默认值，防止None类型触发错误
MASTER_PORT = 6060
NNODES = os.getenv("MA_NUM_HOSTS")
NODE_RANK=os.getenv("VC_TASK_INDEX")
NGPUS_PER_NODE = os.getenv("MA_NUM_GPUS")

if not NGPUS_PER_NODE:
```

```
    NGPUS_PER_NODE = "1"

print("MASTER_HOST: " + MASTER_HOST)
print("MASTER_ADDR: " + MASTER_ADDR)
print("MASTER_PORT: " + MASTER_PORT)
print("NNODES: " + NNODES)
print("NODE_RANK: " + NODE_RANK)
print("NGPUS_PER_NODE: " + NGPUS_PER_NODE)

if int(NNODES) <= 1:
    # 若节点数为1，则为单机训练  若GPU大于1则为多卡训练，否则为单卡训练
    if int(NGPUS_PER_NODE) > 1:
        print("multi-gpus training.")
        command = "python -m torch.distributed.launch"
        command += " --nproc_per_node=" + str(NGPUS_PER_NODE)
        command += " " + os.path.split(os.path.realpath(__file__))[0] + "/Main.py"
    else:
        print("single-gpu training.")
        command = "python -u " + os.path.split(os.path.realpath(__file__))[0] + "/Main.py"
else:
    # 多机分布式训练，使用python torch.distributed.launch或torchrun
    # 用户若聚焦多机分布式训练，仅需关注当前段
    print("multi-machies training.")
    command = "python -u -m torch.distributed.launch"
    command += " --nproc_per_node=" + str(NGPUS_PER_NODE)
    command += " --nnodes=" + str(NNODES)
    command += " --node_rank=" + str(NODE_RANK)
    command += " --master_addr=" + str(MASTER_ADDR)
    command += " --master_port=" + str(MASTER_PORT)
    # 拼接原启动脚本和外部传参
    command += " " + os.path.split(os.path.realpath(__file__))[0] + "/train.py"

user_args = " " + " ".join(sys.argv[1:])
command += user_args

print(command)

# 启动训练脚本并监测退出状态
start = time.time()
process = subprocess.Popen(command, shell=True)
end = time.time()
process.wait()
print("training time:" + str(end - start))

message = process.returncode
print("training message:" + str(message))
sys.exit(message)
```

6.3 开发环境

开发环境依托于已购买的集群算力，支持客户在浏览器窗口或线下IDE中进行交互式的算法或模型的开发和调试。

创建开发环境时可以通过挂载的方式，便捷的访问平台的对象，如算法、模型和数据集。同时支持访问同区域自己的对象存储或外网资源，进行测试数据的导入、开源代码的拉取、三方依赖的安装等操作。极大的提升了算法调试的效率，并保证了和训练任务所使用的的环境和算力的一致性。

创建开发环境前提条件

- 已购买“AI处理节点”，且在“运维配置 > 集群纳管”中已存在类型为“ModelArts”且状态为“可用”的集群。
- 已创建类型为“训练/评测”用途的镜像，详情可参考**制作开发环境镜像**。

创建开发环境

步骤1 在左侧菜单栏中选择“训练服务 > 开发环境”。

步骤2 单击“新建开发环境”，填写基本信息。

表 6-5 新建开发环境

参数	示例	说明
名称	devenv-kkgi9	开发环境名称，包含中英文、数字、“_”“-”，不得超过32个字符。默认自动填充devenv-加五位随机字串。
描述	-	简要描述环境，不包含“@^\#$%&*<>'|"/”，不得超过256个字符。
镜像	-	开发环境启动所使用的容器镜像。可选择各个用途的镜像，包含过去基于开发环境保存的镜像。 请勿对开发环境所使用的镜像进行二次推送，可能会导致开发环境创建或启动时在队列中无法被调度。
资源挂载	-	可选择平台上现有的模型、训练算法和数据集，开发环境启动时以挂载的方式加载到容器中。最多可选择5条。 ● 模型：可选择用途为训练的非空模型仓库。 ● 训练算法：可选创建成功的算法。 ● 数据集：可选择有创建成功版本且非空的数据集。
缓存挂载	-	可选择平台上现有的数据缓存，开发环境启动时以挂载的方式加载到容器中。最多可选择5条。
资源规格	-	选择类型为ModelArts的资源池及其可用规格。
SSH远程接入	-	选择是否开启开发环境的SSH远程接入功能。默认关闭。开启后无法关闭，请谨慎开启。
优先级	0	设置任务的优先级，数值取[-50,50]的整数，数字越大，优先级越高。
是否自动停止	-	设置运行中的开发环境的自动停止的倒计时，可取1~72小时。避免资源浪费。如果不设置，任务将一直运行。

步骤3 单击“确认”。

----结束

管理创建的开发环境

在“开发环境”列表，每条记录包含名称、状态、镜像、资源规格、描述、创建者、创建时间等信息。并可进行以下操作：

表 6-6 开发环境相关操作

任务	操作步骤
搜索开发环境	在搜索输入框中输入一个或多个条件，按回车键即可查询。目前可基于名称和状态进行搜索和筛选。
查看开发环境详情	单击任务名称，进入开发环境详情页，具体见**开发环境详情页介绍**。
自动停止设置	“运行中”的开发环境支持单击状态旁的秒表符号编辑自动停止设置。
打开开发环境	“运行中”的开发环境支持单击“操作”列的“打开”在浏览器中开启Jupyter Notebook页面。
下载私钥	“运行中”且开启了“SSH远程接入”的开发环境可以下载用于SSH连接用的私钥。 **须知** SSH私钥仅可下载一次，请妥善保存。 若私钥丢失，则无法通过SSH连接，但仍可以通过“打开”功能在浏览器中继续使用当前开发环境。
VS Code接入	下载SSH私钥后，单击“VS Code接入”按钮，可跳转到本地VS Code的ModelArts-HuaweiCloud插件，选中下载的私钥完成接入。 **须知** VS Code扩展插件ModelArts-HuaweiCloud首次使用需要等待其自动完成安装，安装成功后会自动建立SSH连接，并在远端完成VS Code Server端的安装。ModelArts-HuaweiCloud插件的具体使用方式请见VS Code插件商店ModelArts-HuaweiCloud介绍。
保存镜像	支持将“运行中”的开发环境保存为快照镜像并归档到八爪鱼镜像仓库中，选择“更多 > 保存镜像”，指定镜像的归档仓库，单击“确认”。保存成功后当前环境正在使用的镜像会切换为刚保存的镜像版本。 ● 保存的镜像中不包含/home/ma-user/work/、/cache和/resource-mounts/等挂载到容器中的目录内容。 ● 建议保存镜像前停止活跃进程和I/O操作。避免因资源占用和冲突等原因导致保存镜像失败。 ● 建议要保存的镜像大小不要超过35GB，最大不能超过50GB，避免因容器引擎的限制导致保存失败。 ● 镜像保存一般需要3-10分钟，届时实例状态处于“快照中”，保存成功后，实例状态会重新回到“运行中”。
编辑开发环境	“排队中”或“已停止”的开发环境支持修改描述、镜像、资源挂载、缓存挂载和资源规格。
停止开发环境	“排队中”、“运行中”、“错误”的开发环境，用户可以单击操作栏的“停止”终止环境。
启动开发环境	“已停止”的开发环境，用户可以单击操作栏的“启动”重新把开发环境加入任务队列。
删除开发环境	选择操作栏的“更多 > 删除”，删除开发环境。

开发环境允许的操作与所处状态约束关系请见下表：

表 6-7 开发环境相关操作与所处状态约束

状态	状态描述	是否占用GPU资源	打开	VSCODE接入	下载私钥	启动	停止	编辑	删除
排队中	开发环境已创建成功，进入任务队列等待系统调度。	-	-	-	-	-	√	√	√
创建中	开发环境已提交到平台，正在创建实例。	√	-	-	-	-	-	-	-
创建失败	开发环境创建失败。	-	-	-	-	-	-	-	√
运行中	开发环境创建成功，处于运行中。	√	√	√	√	-	√	-	√
快照中	开发环境正在保存镜像，仍处于运行中。	√	-	-	-	-	-	-	-
错误	开发环境状态异常。	√	-	-	-	-	√	-	√
启动中	开发环境正在从停止状态中启动。	√	-	-	-	-	-	-	-
启动失败	开发环境启动失败。	-	-	-	-	-	-	-	√
停止中	开发环境正在停止中。	√	-	-	-	-	-	-	-
已停止	开发环境已停止。	-	-	-	-	√	-	√	√

开发环境详情页介绍

在开发环境详情页，提供管理当前开发环境的操作和展示详情信息、拓展资源和事件信息。

- 管理开发环境：支持打开、VS CODE接入、停止/启动、下载私钥、编辑和删除操作。
- 开发环境详情：展示开发环境ID、名称、描述、状态、资源池、资源规格、创建者、镜像、创建时间、更新时间、SSH访问地址、SSH远程接入等信息。
- 拓展资源：展示当前环境中关联的资源挂载和缓存挂载。
  - 展示拓展资源类型、名称、版本、挂载位置（所选资源所对应的OBS存储或SFS存储在开发环境容器中的实际挂载路径）信息。
  - 支持单击资源名称跳转到对应的资源详情。
  - 支持编辑、添加、删除处于“排队中”或“已停止”状态的开发环境的资源挂载和缓存挂载。
- 事件：展示开发环境生命周期中发生的事件，包含名称、事件级别、事件详情和发生时间，并支持按照级别和时间进行筛选和过滤。
- 快照镜像：展示开发环境历史保存过的快照镜像，包含名称、镜像版本、用途、状态、描述、创建时间等信息。
  - 单击镜像名称可跳转至对应的镜像仓库。
  - 单击“删除”可删除该快照记录并同步删除对应的镜像。当快照镜像被当前开发环境所使用时，将无法删除。
  - 当开发环境处于排队中或已停止时，可以单击快照记录后的“切换”按钮切换环境的镜像到历史的某个快照镜像。

6.4 训练任务

Octopus平台为用户提供训练任务管理（支持分布式训练），任务实时日志，产物（模型）管理等多种功能。

创建训练任务

步骤1 在左侧菜单栏中选择“训练服务 > 训练任务”。

步骤2 单击“新建训练任务”，填写基本信息。

- 名称：任务组名称，包含中英文、数字、“_”“-”，不得超过32个字符。
- 描述：简要描述任务，不包含“@^\#$%&*<>'|"/”，不得超过256个字符。
- 资源规格：当前项目中可用的资源规格。
- 计算节点个数：用于运行训练任务的训练节点个数。
- 优先级：设定任务的优先级，数值取[-50,50]的整数，数字越大，优先级越高。

步骤3 选择算法。

- 训练算法：根据业务所需选择算法，自定义算法需提前在“训练服务 > 算法管理”中创建成功。

> **说明**
>
> 当训练算法的“类型”选择为“仅镜像（包含算法文件）”时，且训练任务的资源规格为ModelArts类型时，容器启动使用的默认用户为**ma-user**，需确保该用户拥有调用算法文件和运行依赖的相关权限。

- 参数列表：由算法携带，可修改参数值。
- 环境变量：由算法携带，可修改参数值。

步骤4 选择需要归档的模型仓库。

模型仓库需提前在“数据资产 > 模型管理”中创建成功。

- 常规训练：基于数据集和用户算法训练新模型。
- 增量训练：基于用户导入的模型或已完成训练模型版本（可通过${MODEL}获取该模型版本的文件路径）和新数据集使用选择的算法再次进行训练，生成精度更高的新模型。同常规训练不同的是需要额外选择（最多选择10个模型版本）输入模型和版本,输出模型仓库。

步骤5 选择数据输入

支持从“数据资产”下的“数据缓存”中选择“缓存成功”或“缓存过期”的数据集。数据集/数据缓存最多支持选择10条。

步骤6 选择资源挂载。

如果**步骤2**中的“资源规格”选择的是ModelArts类型的集群，可添加资源挂载，最多支持10条。目前支持挂载的资源只有“数据集”。

步骤7 单击“创建”，在任务列表或分组可查看新建训练任务。

----结束

训练任务相关操作

在“训练任务”列表，可对训练任务进行以下操作：

表 6-8 训练任务相关操作

任务	操作步骤
查找任务	在搜索输入框中输入搜索条件，按回车键即可查询。
查看任务详情	单击任务名称，可在任务详情页查看该任务详情、资源挂载、参数信息、任务日志和资源占用情况。 • 任务详情：任务ID、名称、描述、状态、资源规格等信息。 • 资源挂载：显示任务挂载的资源类型，名称和挂载位置。 • 参数详情：训练算法参数以及环境变量信息。 • 任务日志：任务运行过程中生成的日志信息，详情请查看**训练任务日志查看和下载**。 • 资源占用情况：显示任务占用的CPU、内存、GPU（显存）利用率、占用率等指标百分比折线图，详情请查看**资源占用情况**。

任务	操作步骤
删除任务	● 单击操作栏的“删除”，删除单个任务。 ● 勾选多个任务，单击列表上方的“删除”，可批量删除任务。
重建任务	单击操作栏内的“重建”，输入新任务名称（“任务组名-自定义名称”）和“删除原任务”选项，重建任务时可调整除算法外的所有信息。
停止任务	对于运行中、等待中的任务，用户可以单击操作栏的“停止”终止任务。

训练任务相关操作与任务所处状态约束关系请见下表：

表 6-9 训练任务相关操作与任务所处状态约束

作业状态	作业状态描述	是否占用GPU资源	重建	删除	停止
排队中	训练任务已创建成功，进入任务队列等待系统调度。	-	-	√	√
提交中	训练任务已被系统调度，正在向平台提交。	√	-	-	-
提交失败	训练任务向平台提交失败。	-	√	√	-
等待中	训练任务向平台提交成功，暂未运行。	√	-	√	√
运行中	训练任务向平台提交成功，处于运行中。	√	-	-	√
运行异常	训练任务运行失败。	-	√	√	-
已完成	训练任务运行成功。	-	√	√	-
停止中	单击“停止”后尚未停止。	√	-	-	-
停止失败	单击“停止”后停止失败。	√	-	√	-
已停止	训练任务已停止。	-	√	√	-
删除中	单击“删除”后尚未删除。	√	-	-	-

作业状态	作业状态描述	是否占用GPU资源	重建	删除	停止
删除失败	单击“删除”后删除失败。	仅等待中的任务单击“删除”且删除失败的任务可能会占用	-	√	-

训练任务日志查看和下载

训练任务运行的过程中生成日志，训练任务模块提供了日志的查看以及下载功能，支持用户查看训练任务的运行情况。CCE平台训练任务生成的日志文件有以下四种：

- **train-{id}-{index}.log**：用户实际训练任务的训练日志。
- **train-{id}-{index}-init.log**：Octopus平台提供的前置数据的准备日志。
- **train-{id}-{index}-sidecar.log**：Octopus平台提供的任务流程控制日志，包括日志同步、结果上传。
- **octopus-train-{id}-{index}-supplemental.logs**: Octopus平台任务异常退出或停止产生的错误信息输出日志，运行正常时不产生该日志。

  说明

  {id}为该训练任务ID，{index}为节点编号，例如单节点single-0，多节点distributed-0 distributed-1。

详情页，单击“任务日志”，可查看该训练任务日志详情。支持在线浏览或下载至本地。如果日志较多，用户可在搜索框中输入关键字，查找指定日志内容。

在日志服务页面中的日志列表部分详细展示了该训练任务包含的日志文件的大小以及最新写入时间。单击文件后的“查看”，算法训练的详细执行过程会在日志详情部分展示。用户也可在日志文件后的“操作”栏中，单击“下载”，即可将该日志文件下载到本地查看。

资源占用情况

在任务运行中，资源占用情况模块显示任务占用的CPU、内存、GPU/显存利用率、占用率百分比的折线图。默认显示CPU占用情况折线图。

- 双击任一图例：显示全部资源占用折线图。
- 单击指定图例：只显示该图例折线图。

此模块也可显示多个计算节点运行任务时，资源占用的情况。

- 如果选择2个计算节点运行任务，则可选择查看单个节点资源占用情况。
- 如果1个计算节点上存在多张GPU，则会显示所有GPU占用情况。

  说明

  资源占用情况功能模块，需要用户在制作自定义镜像时安装psutil与pynvml，参考命令如下：

  ```
  pip install psutil pynvml
  ```

  如果未安装psutil与pynvml，则页面无法显示资源使用状况。

6.5 模型评测

6.5.1 评测脚本

在机器学习中，通常需要使用一定的方法和标准，来评测一个模型的预测精确度。自动驾驶领域通常涉及目标检测、语义分割、车道线检测等类别，如识别车辆、行人、可行区域等对象。

Octopus平台提供评测脚本管理功能，支持用户创建、删除、编辑、在线编辑、查询评测脚本等功能。

创建评测脚本

添加评测脚本流程为“初始化评测脚本 > 选择评测脚本文件 > 上传评测脚本文件”。具体操作步骤如下：

步骤1 在左侧菜单栏中选择“训练服务 > 模型评测”。

步骤2 选择“评测脚本”页签，单击“新建评测脚本”，填写脚本名称和描述信息。

步骤3 单击“初始化”，完成脚本的初始化过程。

步骤4 上传文件

将本地脚本文件夹拖入框中，或单击此处选择本地文件夹。文件夹选择完毕后，单击“上传”。

说明

上传中刷新或关闭浏览器会导致文件上传异常，请谨慎操作！

步骤5 查看脚本详情

单击脚本名称，进入该脚本的详情页面。展示脚本ID、名称、描述、状态等信息。

----结束

评测脚本相关操作

在“评测脚本”列表，可对脚本进行以下操作。

表 6-10 评测脚本相关操作

任务	操作步骤
查找脚本	在搜索输入框中输入搜索条件，按回车键即可查询。
查看脚本详情	单击脚本名称，查看脚本详情。
删除脚本	单击操作栏的“更多 > 删除”，删除脚本。 **说明** 在使用中的评测脚本不能被删除。

任务	操作步骤
编辑脚本	单击操作栏内的“编辑”，可修改脚本信息。 **说明** 在使用中的评测脚本不能被编辑。
上传文件	单击脚本名称后操作栏内的“更多 > 上传文件”，上传本地脚本文件。 **说明** 脚本所处状态为“初始化”、“上传中断”的自定义算法可上传文件。
打包文件	单击操作栏的“打包”，即可打包脚本。 **说明** 脚本所处状态为“创建成功”且打包状态不为“打包中”的自定义脚本可打包文件。
下载脚本	当脚本的打包状态为“打包成功”时，单击操作栏的“更多 > 下载”，即可下载脚本。
脚本编辑	单击在线编辑栏的“在线编辑”，可在线编辑脚本，具体请参考**评测脚本在线编辑**。

评测脚本在线编辑

平台提供脚本编辑器，可在线编辑评测文件。单击评测脚本名称后 " 在线编辑 " ，进入该脚本的在线编辑页面。界面左侧显示的是该评测文件包内的所有文件，以目录树的形式展示，支持编程语言的渲染，支持MarkDown文件的实时双屏预览。

- 新增文件夹：选中文件夹并单击 ，用户将新建一个该文件夹的子文件夹。选中工程文件并单击 ，将会新建一个新的文件夹，与用户已有的文件夹同级。
- 新增文件：单击，或单击某个文件夹目录，单击“新增文件”，用户可新增文件。
- 修改文件：单击，用户可对文件名称进行修改。
- 删除文件：单击 ，用户可删除文件。

  说明

  - 文件（夹）名称不能为空，且只能包含数字、英文、中文、点、下划线、中划线。
  - 删除后不可恢复，请谨慎操作。
- 配置界面：单击 ，按照喜好配置界面基本属性，查看快捷键说明。
- 删除当前文件：单击 " 删除 " ，删除当前页面的文件。删除后不可恢复，请谨慎操作。
- 保存文件：单击 " 保存 " ，保存当前脚本。脚本更新完毕请及时保存。

评测脚本使用说明：

1. 模型路径通过环境变量{MODEL}获取，用户在评测过程中需要选择该模型，需要注意的是，以上环境变量仅指明模型文件夹，具体选择模型需用户指定，例如使用os.path.join拼接，或者使用os.listdir查找选择 。
2. 评测数据集通过环境变量{DATASET_MAP}获取：
   ```
   dataset_dict= json.loads(os.getenv("DATASET_MAP"))
   ```
3. 用户产生的评测结果可以存储为任意格式文件，任务完成后可在评测任务详情页的评测结果中查看（支持通用文本文件）或下载结果文件。

   说明

   评测使用以下环境变量：
   - MODEL：模型目录
   - DATASET：数据集目录
   - DATASET_MAP：数据集名称和容器内路径映射关系的JSON字串
   - EVAL_RESULT：用户自定义评测结果存放目录

6.5.2 评测任务

6.5.2.1 创建任务

Octopus平台提供对评测任务的管理，包括创建、删除、停止评测任务的操作。在评测任务页面， 实时显示多条评测任务的状态、任务创建时间等信息。评测任务创建完毕后，触发任务。评测任务结束后会生成评测任务结果文件，详细展示模型的各项评测指标得分。

创建评测任务

评测任务与评测参数和数据集有关。具体步骤如下：

步骤1 在左侧菜单栏中选择“训练服务 > 模型评测”。

步骤2 选择“评测任务”页签，单击“新建评测任务”，填写如下信息。

表 6-11 新建评测任务

参数	描述
名称	任务组名称，包含中英文、数字、“_”“-”，不得超过32个字符。
描述	简要描述任务，不包含“@^\#$%&*<>'|"/”，不得超过256个字符。
类别	分“内置”或“用户”，然后选择对应类别： • 内置：支持“分类”、“2D目标检测”、“3D目标检测”、“2D目标追踪”、“3D目标追踪”“2D语义分割”、“3D语义分割”、“车道线检测”八个类别。 • 用户：不允许选择内置评测项，仅可使用自定义评测。
资源规格	选择当前项目中可用的资源用途为“模型评测”的资源规格，可参考**10.1.2 资源管理**创建资源规格。
优先级	设定任务的优先级，数值取[-50,50]的整数，数字越大，优先级越高。

参数	描述
关联已有算法	可选择关联已有的算法。
模型	选择模型仓库和对应的模型版本，如果需要将模型对应的原始算法也挂载到用户评测容器中，可选择“关联已有算法”，系统默认选择模型创建时的关联算法，用户也可手动更改选择其他算法文件。选择该项后，会自动添加算法路径环境变量${ALGORITHM}至容器中，并在“评测启动指令”文本框内输入环境变量提示信息。
数据集	选择普通数据集或缓存数据集。
评测配置	设置评测配置，详情请参考**步骤3**。

步骤3 选择自定义评测和内置指标评测（至少选择一项）。

- **自定义评测：**
  - 创建自定义BadCase判别：当类别选择“2D、3D目标检测”时，可选择创建BadCase判别；自定义和内置的BadCase判别只能选择其中一种。
  - 评测脚本：可选，如果用户有其他自定义评测脚本，可勾选该项将选择的脚本路径挂载至容器内部，用户可使用${SCRIPT}进行引用。
  - 评测启动指令：用户指定自定义评测的启动命令，根据用户选择注入以下环境变量：

表 6-12 环境变量

类型	启动命令	默认值
模型	${MODEL}	/tmp/data/model
算法	${ALGORITHM}	/tmp/data/algorithm
数据集	${DATASET}	/tmp/data/dataset/dataset-0
自定义评测脚本	${SCRIPT}	/tmp/data/script
评测结果路径	${EVAL_RESULT}	/tmp/result/eval
坏例判别路径	${BADCASE_RESULT}	/tmp/badcase ${BADCASE_RESULT}/annotations 用户提供坏例的标准Octopus标注数据。 ${BADCASE_RESULT}/records 用户保存坏例记录文件位置。

示例命令：


```
python ${SCRIPT}/eval.py --image ${DATASET} --output ${EVAL_RESULT} --model ${MODEL}/best.pt
```

- **内置指标评测：**
  - 创建内置BadCase判别：当类别选择“2D、3D目标检测”时，可选择创建BadCase判别；自定义和内置的BadCase判别只能选择其中一种。
  - 推理启动命令：用户使用内置指标评测时需将数据推理结果存储在特定目录下，相关环境变量包括如下：

    表 6-13 环境变量

类型	启动命令	默认值
模型	${MODEL}	/tmp/data/model
算法	${ALGORITHM}	/tmp/data/algorithm
数据集	${DATASET}	/tmp/data/dataset/dataset-0
推理结果路径	${INFER_RESULT}	/tmp/result/infer
坏例判别路径	${BADCASE_RESULT}	/tmp/badcase ${BADCASE_RESULT}/annotations 用户提供坏例的标准Octopus标注数据。 ${BADCASE_RESULT}/records 用户保存坏例记录文件位置。

    示例命令：

    ```
    python ${MODEL}/detect.py --image ${DATASET} --output ${INFER_RESULT} --model ${MODEL}/best.pt
    ```

  - 评测参数项：根据用户选择的模型类别，动态生成评测所需的参数项，用户可参考前端提示填写。
  - 评测指标项：根据用户选择的模型类别，动态生成的平台支持的内置指标项，用户单击选中对应指标项则可加入到待评测指标中。

步骤4 单击“创建”，在评测任务页面显示新创建的任务信息。

----结束

评测任务相关操作

在“评测任务”列表，可对任务进行以下操作。

表 6-14 评测任务相关操作

任务	操作步骤
查找任务	在搜索输入框中输入搜索条件，按回车键即可查询。

任务	操作步骤
查看任务详情	单击任务名称，可在任务详情页查看该任务详情、参数信息、评测结果、任务日志和资源在占用情况。 ● 任务详情：任务ID、名称、描述、状态、资源类型等信息。 ● 任务视图：根据创建任务时选择不同的评测，显示不同的任务视图。 ● 评测结果：单击任务视图，可显示相对应的评测任务结果，详情请参考**评测结果**。 ● 任务日志：单击任务视图，可显示相对应的任务运行过程中生成的日志信息，详情请查看**评测任务日志查看与下载**。 ● 资源占用情况：单击任务视图，可显示相对应任务占用的CPU、内存、GPU显存利用率、占用率百分比的折线图，详情请查看**资源占用情况**。
删除任务	● 单击操作栏的“删除”，删除单个任务。 ● 勾选多个任务，单击列表上方的“删除”，可批量删除任务。
重建任务	单击操作栏内的“重建”，输入新任务名称（以“任务组名-自定义名称”的形式）和是否删除原任务选项，同时可重新选择需要修改的参数。
停止任务	单击操作栏的“停止”，停止评测任务。
继续任务	单击操作栏中的“继续”，继续执行任务。
对比任务	勾选2-4个任务（要求类别相同，且均为已完成状态），单击“对比”创建对比任务，具体步骤请参考**评测对比**。

评测任务相关操作与任务所处状态约束关系请见下表。

表 6-15 作业状态与操作约束关系

作业状态	重建	删除	停止	继续
排队中	-	√	√	-
等待中	-	-	√	-
提交中	-	-	-	-
提交失败	√	√	-	-
运行中	-	-	√	-
运行异常	√	√	-	√
已完成	√	√	-	-
停止中	-	-	-	-
停止失败	-	√	-	-

作业状态	重建	删除	停止	继续
已停止	√	√	-	√
删除中	-	-	-	-
删除失败	-	√	-	-

6.5.2.2 评测结果

评测任务完成后，可单击评测任务名称查看模型评测各项得分。任务详情页面展示了评测任务详情信息以及评测模型结果信息两部分内容。

任务详情

任务详情页面，详细展示了任务名称、类别、资源规格等基本信息。

任务视图

可查看在当前视图下的评测结果、任务日志和资源占用情况。

坏例判别

仅当用户选择“BadCase”判别后展示，展示badcase的基本信息。运行过程中、坏例判别失败（未获取到坏例结果）时，该视图各项指标显示为"--"。

图 6-1 坏例判别

坏例判别详情：单击“坏例判别”可查看badcase详情。左侧显示统计信息，右侧支持标注数据和预测数据在图像上的对比可视化显示。

保存数据集：单击右上角“保存为数据集”，可将存在坏例判别的数据保存到新数据集中。数据集各属性沿用原始评测数据集，例如“数据类型”、“标注状态”、“数据格式”等。

评测结果

显示模型的评测结果，例如“模型评测结果字典.json”和“模型评测报告.pdf”，可在线查看或下载至本地查看。

用户输出评测结果，如果需创建PDF可视化报告需满足Octopus平台的相关约定，具体如下：

1. 文件约定：用户将待创建PDF的评测结果写入到原始json文件中，文件内容如下：

```
{
  "kind": "rep",
  "docVersion": "Octopus/v1",
   "metadata": {},
   "spec": {
      "overview": [],
      "content": [],
    }
}
```

说明

- kind：表示该文件标识符，当其存在且值为rep时会自动创建PDF，否则将该结果文件直接上传到评测结果文件夹中。
- docVersion：表示八爪鱼评测可视化报告的文档版本。
- metadata：表示文档的基本元信息。
- spec：表示规格，即文档包含的内容。其中overview中保留总览信息，将字段对象渲染为无边线表格概况信息。content渲染为评测结果内容主体。

2. 内容约定：目前支持的元素包括以下：

a. **段落**

```
{
  "kind": "Paragraph",
  "spec": "Test Paragraph!"
}
```

说明

- kind：唯一类型标识。Paragraph标识为段落，用户输出正文或标题文字。
- spec：表示段落内容。

b. **表格**

```
{
   "kind": "Table",
   "spec": {
      "name": "2D Detection Metrics",
      "fields": ["classes", "precision", "recall", "error", "leak", "F-score", "iou cost", "c-distance",
"mPA"],
      "data": [
         ["car", 0.8, 0.8, 0.2, 0.2, 0.8, 0.3, 1.2, 0.8],
         ["pedestrian", 0.8, 0.8, 0.2, 0.2, 0.8, 0.3, 1.2, 0.8]
      ]
   }
}
```

说明

- kind：唯一类型标识，Table标识为表格，多用于输出指标项结果。
- spec：数据主体。
  - name：表格名称。
  - fields：表头元素。
  - data：表主体，每个列表表示一行数据。

c. **柱状图**

```
{
   "kind": "BarChart",
   "spec": {
      "name": "precision-compare",
      "legends": ["min_iou=0.5", "min_iou=0.6"],
      "axes": ["classes", "precision"],
      "labels": ["car", "pedestrian"],
      "data": [
```



```
            [0.8, 0.8],
            [0.7, 0.7]
        ]
    }
}
```

说明

- kind：唯一类型标识，BarChart标识为柱状图，多用于指标结果对比。
- spec：数据主体。
  - name：柱状图名称。
  - legends：图例，列表形式，与data对应。
  - axes：坐标轴设定，x轴与y轴名称。
  - labels：标签列表。
  - data：柱状图数据主体，每个列表表示一个图例单元数据。

d. **折线图**

```
{
    "kind": "LineChart",
    "spec": {
        "name": "car-PR compare",
        "legends": ["min_iou=0.5", "min_iou=0.6"],
        "axes": ["precision", "recall"],
        "data": [
            [[0.2, 0.4, 0.6, 0.8, 1.0], [0.8, 0.8, 0.6, 0.6, 0.4]],
            [[0.2, 0.4, 0.6, 0.8, 1.0], [0.7, 0.7, 0.6, 0.6, 0.5]]
        ]
    }
}
```

说明

- kind：唯一类型标识，LineChart标识为折线图，多用于趋势对比，如PR图。
- spec：数据主体。
  - name：折线图名称。
  - legends：图例，列表形式，与data对应。
  - axes：坐标轴设定，x轴与y轴名称。
  - data：折线图数据主体，每个列表表示一个图例数据（包括x/y双轴）。

e. **饼图**

```
{
    "kind": "PieChart",
    "spec": {
        "name": "BadCase in 2D Detection",
        "data": [30, 50, 80],
        "labels": ["Car", "Truck", "Pedestrian"]
    }
}
```

说明

- kind：唯一类型标识，PieChart标识为饼图。
- spec：数据主体。
  - name：饼图名称。
  - labels：饼图标签与数据data对应。
  - data：饼图数据列表，与标签labels对应。

3. 完整示例。

a. 原始数据“01.json”。

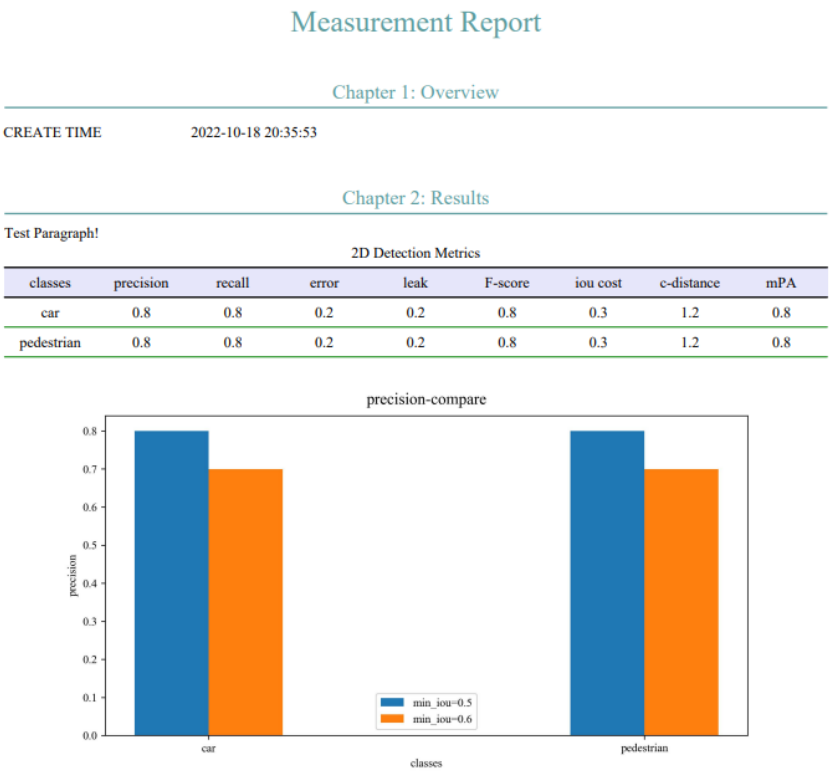

```
{
 "kind": "rep",
 "docVersion": "Octopus/v1",
 "metadata": {
  "name": "模型评测报告",
  "kind": "D2",
  "datetime": "2023-03-16 21:00:00"
 },
 "spec": {
  "overview": [
   {
    "name": "数据集",
    "value": "Octopus/v1"
   }
  ],
  "content": [
   {
    "kind": "Paragraph",
    "spec": "Test Paragraph!"
   },
   {
    "kind": "Table",
    "spec": {
     "name": "2D Detection Metrics",
     "fields": [
      "classes",
      "precision",
      "recall",
      "error",
      "leak",
      "F-score",
      "iou cost",
      "c-distance",
      "mPA"
     ],
     "data": [
      ["car", 0.8, 0.8, 0.2, 0.2, 0.8, 0.3, 1.2, 0.8],
      ["pedestrian", 0.8, 0.8, 0.2, 0.2, 0.8, 0.3, 1.2, 0.8]
     ]
    }
   },
   {
    "kind": "BarChart",
    "spec": {
     "name": "precision-compare",
     "legends": [
      "min_iou=0.5",
      "min_iou=0.6"
     ],
     "axes": [
      "classes",
      "precision"
     ],
     "labels": [
      "car",
      "pedestrian"
     ],
     "data": [
      [0.8, 0.8],
      [0.7, 0.7]
     ]
    }
   },
   {
    "kind": "LineChart",
    "spec": {
     "name": "car-PR compare",
     "legends": [
      "min_iou=0.5",
```



```
        "min_iou=0.6"
       ],
       "axes": [
        "precision",
        "recall"
       ],
       "data": [
        [
         [0.2, 0.4, 0.6, 0.8, 1.0],
         [0.8, 0.8, 0.6, 0.6, 0.4]
        ],
        [
         [0.2, 0.4, 0.6, 0.8, 1.0],
         [0.7, 0.7, 0.6, 0.6, 0.5]
        ]
       ]
      }
     }
    ]
   }
}
```


b. 根据原始数据“01.json”自动创建“01.pdf”。

图 6-2 英文 pdf



Measurement Report

Chapter 1: Overview

CREATE TIME 2022-10-18 20:35:53

Chapter 2: Results

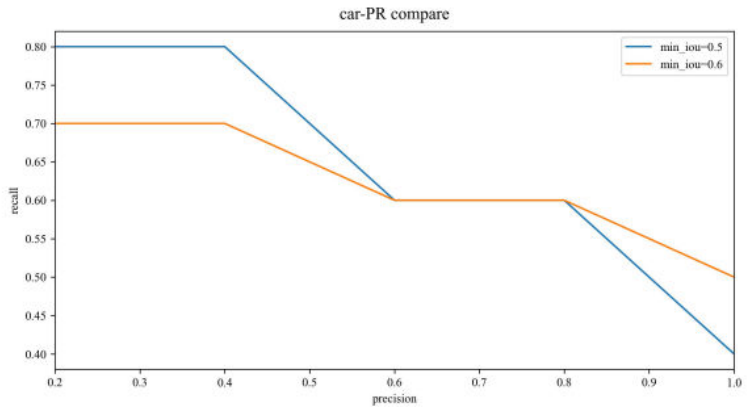
Test Paragraph!

2D Detection Metrics

classes	precision	recall	error	leak	F-score	iou cost	c-distance	mPA
car	0.8	0.8	0.2	0.2	0.8	0.3	1.2	0.8
pedestrian	0.8	0.8	0.2	0.2	0.8	0.3	1.2	0.8

图 6-3 中文 pdf

图 6-4 car-PR compare



评测任务日志查看与下载

评测任务运行的过程中生成日志，训练服务提供日志的查看以及下载功能，支持查看评测任务的运行情况。

生成的日志文件共有四种形式：

- **evaluate-xx-{id}.log**：用户实际评测任务的训练日志。
- **evaluate-xx-{id}-init.log**：Octopus平台提供的前置数据的准备日志。
- **evaluate-xx-{id}-sidecar.log**：Octopus平台提供的任务流程控制日志，包括日志同步、结果上传。

- **octopus-evaluate-xx-{id}-supplemental.logs**: Octopus平台任务异常退出或停止产生的错误信息输出日志，运行正常时不产生该日志。

  说明

  {id}为该训练任务ID，{index}为节点编号，例如单节点single-0，多节点distributed-0 distributed-1。

在评测任务的详情页面，可单击“任务日志”查看任务在运行过程中生成的所有日志。如果日志较多，可在搜索框中输入关键字，查找指定日志内容。

在日志服务页面中的日志列表部分详细展示了该评测任务包含的所有文件的大小以及最新写入时间。单击文件后的“查看”，该文件的详细执行过程则在日志详情部分展示。也可在日志文件后的“操作”栏中，单击“下载”，即可将该日志文件下载到本地查看。

资源占用情况

在任务运行中，资源占用情况模块显示任务占用的CPU、内存、GPU显存利用率、占用率百分比的折线图。默认显示CPU占用情况折线图。

- 双击任一图例：显示全部资源占用折线图。
- 单击指定图例：只显示该图例折线图。

说明

资源占用情况功能模块，需要用户在制作自定义镜像时安装psutil与pynvml，参考命令如下：

```
pip install psutil pynvml
```

如果未安装psutil与pynvml，则页面无法显示资源使用状况。

6.5.3 评测对比

创建评测对比任务

平台支持创建2-4个评测任务结果对比。

步骤1 在左侧菜单栏中单击“训练服务 > 模型评测”。

步骤2 选择“评测对比”页签，单击“新建评测对比”，填写基本信息。

- 名称：对比任务名称，只能包含数字、英文、中文、下划线、中划线。
- 描述：简要描述任务，不包含“@^\#$%&*<>'|"/”，不得超过256个字符。

步骤3 选择对比来源与模式。

- 对比来源：可选择“已有评测任务”或“新建评估任务”。
- 对比模式：可选择“文本对比”或“报告对比”。
  - 文本对比：仅支持对比多个（2个及以上）自定义评测任务产生的文本文件。
  - 报告对比：对于存在多个（2个及以上）内置评测任务时将自动触发内置评测的报告对比，如果同时存在多个自定义评测任务对比文件（符合要对比文件要求）也将触发评测报告对比，此时将产生2份报告对比文件。

步骤4 选择任务。

- 当对比来源为“已有评测任务”时，下拉框选择任务和对比文件。如果符合自定义评测结果对比条件（含自定义评测结果的任务数量不少于2），对于每个任务，用户须选择自定义待对比文件。

- 当对比来源为“新建评测任务”时，新建评测任务，具体步骤可参考**创建评测任务**。

步骤5 单击“添加”，在评测对比任务页面显示新创建的对比任务信息。

步骤6 也可以通过在评测任务列表勾选2-4个任务（要求类别相同，且均为已完成状态）单击“对比”按钮的形式进行新建评测对比任务。

----结束

评测对比相关操作

在“评测对比”列表，可对任务进行以下操作。

表 6-16 评测对比相关操作

任务	操作步骤
查找对比任务	在搜索输入框中输入搜索条件，按回车键即可查询。
查看对比任务详情	单击对比任务名称，可在对比任务详情页查看该评测对比详情、报告对比、任务日志。 • 评测对比详情：任务ID、名称、描述、状态等信息。 • 文本对比：单击“文本对比”，查看或下载评测对比任务对比文本。 • 报告对比：单击“报告对比”，查看或下载评测对比任务在运行过程中生成的报告。 • 任务日志：单击“任务日志”，查看或下载评测对比任务在运行过程中生成的所有日志。
删除对比任务	• 单击操作栏的“删除”，删除单个对比任务。 • 勾选多个任务，单击列表上方的“删除”，可批量删除对比任务。
重建任务	单击操作栏内的“重建”，输入新对比任务名称（以“任务组名-自定义名称”的形式），同时可重新选择需要修改的参数。
停止任务	单击对比任务后的“停止”，停止评测对比任务。

文本格式约定

- 文本对比：支持TXT、JSON文件格式。
- 报告对比：用户自定义评测结果如果需要使用报告对比功能，需满足Octopus格式要求，仅支持JSON文件格式，并且需要满足以下格式要求。

表 6-17 报告对比格式要求

名称	说明	示例
kind	文件标识符，e-res表示评测结果可进行评测对比	e-res

名称	说明	示例
spec	评测结果内容JSON格式支持简单指标结果、带类别复杂指标结果、折线图结果	{“accuracy”: 0.21, “fp”: 1.0, “fn”: 1.0}

a. 简单指标结果。

简单键值对，键表示指标名称，值表示该指标对应的评测结果数值。

```
{
   "accuracy": 0.21,
   "fn": 1.0,
   "fp": 1.0,
}
```

b. 带类别复杂指标结果。

```
{
   "ap": [{"Car": 0.2, "Truck": 0.4}, 0.3],
   "precision": [{"Car": 0.8, "Truck": 0.7}, 0.75],
   "recall": [{"Car": 0.8, "Truck": 0.9}, 0.85],
}
```

说明

键值对，键表示指标名称，值为字典或列表。当值为列表时，第一项为各类别对应的该指标评测结果字典，第二项为所有类别的指标均值。当值为字典时，仅包含上述第一项。

c. 折线图类指标结果。

```
{
   "pr_curve": {
   "Car": [
      [0.0, 0.2, 0.4, 0.6, 0.8, 1.0],
      [1.0, 1.0, 0.8, 0.8, 0.6, 0.4]
   ]
   }
}
```

说明

键值对，键表示指标名称，值为字典或列表。当值为列表时，第一项为各类别对应的该指标评测结果字典，第二项为所有类别的指标均值。当值为字典时，仅包含上述第一项。

以上各种类型可混合使用，参考完整示例如下：

```
{
   "kind": "e-res",
    "spec": {
     "accuracy": 0.21,
     "fn": 1.0,
     "fp": 1.0,
     "ap": [{"Car": 0.2, "Truck": 0.4}, 0.3],
     "precision": [{"Car": 0.8, "Truck": 0.7}, 0.75],
     "recall": [{"Car": 0.8, "Truck": 0.9}, 0.85],
     "pr_curve": {
     "Car": [
        [0.0, 0.2, 0.4, 0.6, 0.8, 1.0],
        [1.0, 1.0, 0.8, 0.8, 0.6, 0.4]
     ]
     }
     }
}
```

6.5.4 模型数据集支持

6.5.4.1 目标检测 2D

Octopus

- **目录**

  标注文件目录结构

```
+--- 1599625710056
|   +--- 1599625710056.jpg
|   +--- 1599625710056.json
+--- 1599625740054
|   +--- 1599625740054.jpg
|   +--- 1599625740054.json
```

  推理文件目录结构

```
+--- 1599625710056
|   +--- 1599625710056.json
+--- 1599625740054
|   +--- 1599625740054.json
```

- **示例标注文件**

  2D目标检测-Octopus标注.json

```
{
  "labels": [
    {
      "label_meta_name": "Car",
      "bndbox": {
        "ymin": 503,
        "xmax": 980,
        "ymax": 524,
        "xmin": 956
      }
    },
    {
      "label_meta_name": "Car",
      "bndbox": {
        "ymin": 508,
        "xmax": 931,
        "ymax": 531,
        "xmin": 904
      }
    }
  ]
}
```

- **示例推理文件**

  2D目标检测-Octopus推理.json

```
{
  "labels": [
    {
      "label_meta_name": "Car",
      "score": 0.89,
      "bndbox": {
        "ymin": 500,
        "xmax": 970,
        "ymax": 520,
        "xmin": 950
      }
    },
    {
      "label_meta_name": "Car",
```



```
    "score": 0.82,
    "bndbox": {
      "ymin": 508,
      "xmax": 930,
      "ymax": 520,
      "xmin": 900
    }
   }
  ]
}
```

KITTI

- **目录**

  标注文件目录结构

```
+--- image_2
|   +--- 000000.png
|   +--- 000001.png
|   +--- 000002.png
+--- label_2
|   +--- 000000.txt
|   +--- 000001.txt
|   +--- 000002.txt
```

  推理文件目录结构

```
+--- 000000.txt
+--- 000001.txt
+--- 000002.txt
```

- **示例标注/推理文件**

  2D目标检测-KITTI.txt

```
Car 0.00 0 -1.57 956 503 980 524 2.61 2.77 5.92 3.39 54.15 -0.01 -1.55 0.
Car 0.00 0 -1.57 904 508 931 531 2.36 1.41 11.72 -3.22 40.29 1.14 -1.57 0.
Car 0.00 0 -1.57 1118 514 1149 538 1.138 2.723 10.583 -0.636 40.702 0.747 -1.57 0.
Car 0.00 0 -1.57 1018 497 1038 520 1.32 1.14 3.78 2.43 37.73 -1.08 -1.57 0.
Car 0.00 0 -1.57 1048 508 1078 536 1.61 0.75 5.70 -3.11 28.20 0.63 -1.59 0.
Car 0.00 0 -1.57 1175 517 1235 557 1.75 1.96 3.75 -7.57 12.66 -0.67 -1.54 0.
Car 0.00 0 -1.57 756 517 802 552 2.61 2.77 5.92 3.39 54.15 -0.01 -1.55 0.
Car 0.00 0 -1.57 304 533 446 612 2.36 1.41 11.72 -3.22 40.29 1.14 -1.57 0.
Car 0.00 0 -1.57 0 491 310 766 1.138 2.723 10.583 -0.636 40.702 0.747 -1.57 0.
Truck 0.00 0 -1.57 510 442 697 601 1.32 1.14 3.78 2.43 37.73 -1.08 -1.57 0.
Car 0.00 0 -1.57 879 516 917 546 1.61 0.75 5.70 -3.11 28.20 0.63 -1.59 0.
Car 0.00 0 -1.57 986 506 1056 577 1.75 1.96 3.75 -7.57 12.66 -0.67 -1.54 0.
Car 0.00 0 -1.57 1072 494 1124 552 1.75 1.96 3.75 -7.57 12.66 -0.67 -1.54 0.
```

VOC

- **目录**

  标注文件目录结构

```
+--- Annotations
|   +--- 000000000.xml
|   +--- 000000001.xml
+--- JPEGImages
|   +--- 000000000.jpg
|   +--- 000000001.jpg
```

  推理文件目录结构

```
+--- 000000000.xml
+--- 000000001.xml
```

- **示例标注/推理文件，推理含score字段**

  2D目标检测-VOC.xml

```
<?xml version="1.0" encoding="utf-8"?>
<annotation>
```



```
    <size>
        <depth>3</depth>
        <width>1920</width>
        <height>1020</height>
    </size>
    <label_counts>
        <label_num>1</label_num>
        <label_meta_id>373</label_meta_id>
    </label_counts>
    <label_counts>
        <label_num>12</label_num>
        <label_meta_id>372</label_meta_id>
    </label_counts>
    <segmented>0</segmented>
    <folder>dataset-1064version-1</folder>
    <label_task_id>14224</label_task_id>
    <origin_name>1598929950099.jpg</origin_name>
    <object>
        <label_meta_id>372</label_meta_id>
        <occluded>0</occluded>
        <bndbox>
            <ymin>503</ymin>
            <xmax>980</xmax>
            <ymax>524</ymax>
            <xmin>956</xmin>
        </bndbox>
        <pose>Unspecified</pose>
        <attribute>{"遮挡":"0","截断":"0"}</attribute>
        <truncated>0</truncated>
        <difficult>1</difficult>
        <name>Car</name>
        <serial_number>16</serial_number>
        <shape_type>bndbox</shape_type>
    </object>
    <filename>hash0-1598929950099.xml</filename>
</annotation>
```

6.5.4.2 目标检测 3D

Octopus

- **目录**

  标注文件目录结构

```
+--- 1611801018801
|   +--- 1611801018801.json
|   +--- 1611801018801.pcd
+--- 1611801024401
|   +--- 1611801024401.json
|   +--- 1611801024401.pcd
```

  推理文件目录结构

```
+--- 1611801018801
|   +--- 1611801018801.json
+--- 1611801024401
|   +--- 1611801024401.json
```

- **示例标注文件**

  3D目标检测-Octopus标注.json

```
{
  "labels": [
    {
      "cube_3d": {
        "location": {
          "z": -0.010374438202406334,
          "y": 54.14756705529089,
```



```
        "x": 3.386641177177716
       },
       "serial_number": 0,
       "dimensions": {
        "height": 2.6135754585266113,
        "width": 2.7737574577331543,
        "length": 5.917389392852783
       },
       "rotation": {"x": 0, "y": 0, "z": 4.729842272904634}
      },
      "serial_number": 0,
      "shape_type": "cube_3d",
      "label_meta_name": "厢式货车",
      "label_meta_id": 598
     },
     {
      "cube_3d": {
       "location": {
        "z": 1.1376991421138332,
        "y": 40.2855870175903,
        "x": -3.224063568250358
       },
       "serial_number": 1,
       "dimensions": {
        "height": 2.3640241622924805,
        "width": 1.4068853855133057,
        "length": 11.719782829284668
       },
       "rotation": {"x": 0, "y": 0, "z": 4.71238898038469}
      },
      "serial_number": 0,
      "shape_type": "cube_3d",
      "label_meta_name": "大型车",
      "label_meta_id": 493
     }
    ]
}
```


- **示例推理文件**

  3D目标检测-Octopus推理.json


```
{
 "labels": [
  {
   "label_meta_name": "厢式货车",
   "cube_3d": {
    "location": {
     "x": 3.38,
     "y": 54.14,
     "z": -0.010
    },
    "dimensions": {
     "length": 5.91,
     "width": 2.77,
     "height": 2.61
    },
    "rotation": {
     "x": 0,
     "y": 0,
     "z": 4.72
    }
   },
   "score": 0.89
  },
  {
   "label_meta_name": "厢式货车",
   "cube_3d": {
    "location": {
     "z": 0.07,
     "y": -12.98,
```



```
        "x": 3.84
      },
      "dimensions": {
        "height": 3.23,
        "width": 2.64,
        "length": 6.50
      },
      "rotation": {
        "x": 0,
        "y": 0,
        "z": 4.71
      }
    },
    "score": 0.82
  }
 ]
}
```

KITTI

- **目录**

  标注文件目录结构

```
+--- label_2
|   +--- 000000.txt
|   +--- 000001.txt
|   +--- 000002.txt
+--- velodyne
|   +--- 000000.bin
|   +--- 000001.bin
|   +--- 000002.bin
```

  推理文件目录结构

```
+--- 000000.txt
+--- 000001.txt
+--- 000002.txt
```

- **示例标注/推理文件**

  3D目标检测-KITTI.txt

```
厢式货车 0.00 0 -1.57 0. 0. 0. 0. 2.61 2.77 5.92 3.39 54.15 -0.01 -1.55 0.
大型车 0.00 0 -1.57 0. 0. 0. 0. 2.36 1.41 11.72 -3.22 40.29 1.14 -1.57 0.
大型车 0.00 0 -1.57 0. 0. 0. 0. 1.138 2.723 10.583 -0.636 40.702 0.747 -1.57 0.
小型车 0.00 0 -1.57 0. 0. 0. 0. 1.32 1.14 3.78 2.43 37.73 -1.08 -1.57 0.
厢式货车 0.00 0 -1.57 0. 0. 0. 0. 1.61 0.75 5.70 -3.11 28.20 0.63 -1.59 0.
小型车 0.00 0 -1.57 0. 0. 0. 0. 1.75 1.96 3.75 -7.57 12.66 -0.67 -1.54 0.
```

6.5.4.3 目标追踪 2D

Octopus

- **目录**

  标注文件目录结构

```
+--- 1599625710056
|   +--- 1599625710056.jpg
|   +--- 1599625710056.json
+--- 1599625740054
|   +--- 1599625740054.jpg
|   +--- 1599625740054.json
```

  推理文件目录结构

```
+--- 1599625710056
|   +--- 1599625710056.json
+--- 1599625740054
|   +--- 1599625740054.json
```


- **示例标注文件**

  2D目标追踪-Octopus标注.json

```
{
"frame_id": 0,
  "labels": [
    {
      "label_meta_name": "Car",
"label_object_id": 1,
      "bndbox": {
        "ymin": 503,
        "xmax": 980,
        "ymax": 524,
        "xmin": 956
      }
    },
    {
      "label_meta_name": "Car",
"label_object_id": 2,
      "bndbox": {
        "ymin": 508,
        "xmax": 931,
        "ymax": 531,
        "xmin": 904
      }
    }
  ]
}
```

- **示例推理文件**

  2D目标追踪-Octopus推理.json

```
{
"frame_id": 0,
  "labels": [
    {
      "label_meta_name": "Car",
"label_object_id": 1,
      "score": 0.89,
      "bndbox": {
        "ymin": 500,
        "xmax": 970,
        "ymax": 520,
        "xmin": 950
      }
    },
    {
      "label_meta_name": "Car",
"label_object_id": 2,
     "score": 0.82,
      "bndbox": {
        "ymin": 508,
        "xmax": 930,
        "ymax": 520,
        "xmin": 900
      }
    }
  ]
}
```

6.5.4.4 目标追踪 3D

Octopus

- **目录**

  标注文件目录结构


```
+--- 1611801018801
|   +--- 1611801018801.json
|   +--- 1611801018801.pcd
+--- 1611801024401
|   +--- 1611801024401.json
|   +--- 1611801024401.pcd
```

推理文件目录结构

```
+--- 1611801018801
|   +--- 1611801018801.json
+--- 1611801024401
|   +--- 1611801024401.json
```

- **示例标注文件**

  3D目标追踪-Octopus标注.json

```
{
"frame_id": 0,
  "labels": [
    {
      "cube_3d": {
        "location": {
          "z": -0.010374438202406334,
          "y": 54.14756705529089,
          "x": 3.386641177177716
        },
        "serial_number": 0,
        "dimensions": {
          "height": 2.6135754585266113,
          "width": 2.7737574577331543,
          "length": 5.917389392852783
        },
        "rotation": {"x": 0, "y": 0, "z": 4.729842272904634}
      },
      "serial_number": 0,
      "shape_type": "cube_3d",
      "label_meta_name": "厢式货车",
      "label_meta_id": 598
    },
    {
      "cube_3d": {
        "location": {
          "z": 1.1376991421138332,
          "y": 40.2855870175903,
          "x": -3.224063568250358
        },
        "serial_number": 1,
        "dimensions": {
          "height": 2.3640241622924805,
          "width": 1.4068853855133057,
          "length": 11.719782829284668
        },
        "rotation": {"x": 0, "y": 0, "z": 4.71238898038469}
      },
      "serial_number": 0,
      "shape_type": "cube_3d",
      "label_meta_name": "大型车",
      "label_meta_id": 493
    }
  ]
}
```

- **示例推理文件**

  3D目标检测-Octopus推理.json

```
{
  "frame_id": 0,
"labels": [
    {
      "label_meta_name": "厢式货车",
```



```
    "cube_3d": {
"serial_number": 1,
      "location": {
        "x": 3.38,
        "y": 54.14,
        "z": -0.010
      },
      "dimensions": {
        "length": 5.91,
        "width": 2.77,
        "height": 2.61
      },
      "rotation": {
        "x": 0,
        "y": 0,
        "z": 4.72
      }
    },
    "score": 0.89
  },
  {
    "label_meta_name": "厢式货车",
    "cube_3d": {
  "serial_number": 1,
      "location": {
        "z": 0.07,
        "y": -12.98,
        "x": 3.84
      },
      "dimensions": {
        "height": 3.23,
        "width": 2.64,
        "length": 6.50
      },
      "rotation": {
        "x": 0,
        "y": 0,
        "z": 4.71
      }
    },
    "score": 0.82
  }
 ]
}
```


6.5.4.5 语义分割 2D

Octopus

- **目录**

  标注文件目录结构

  ```
  +--- 1599625710056
  |   +--- 1599625710056.jpg
  |   +--- 1599625710056.json
  +--- 1599625740054
  |   +--- 1599625740054.jpg
  |   +--- 1599625740054.json
  ```

  推理文件目录结构

  ```
  +--- 1599625710056
  |   +--- 1599625710056.json
  +--- 1599625740054
  |   +--- 1599625740054.json
  ```

- **示例标注/推理文件**

  2D语义分割-Octopus.json

```
{
 "image_meta_info": {
  "size": {
   "height": 1024,
   "width": 2048
  }
 },
 "labels":
  [
   {
    "shape_type": "polygon",
    "label_meta_name": "road",
    "serial_number": 1,
    "label_meta_id": 0,
    "polygon": {"size": 3, "points": [{"xpoint": 1706, "ypoint": 1023}, {"xpoint": 1709, "ypoint": 1023},
     {"xpoint": 1709, "ypoint": 1020}]}
   },
   {
    "shape_type": "polygon",
    "label_meta_name": "bicycle",
    "serial_number": 25,
    "label_meta_id": 18,
    "polygon": {"size": 20, "points": [{"xpoint": 1006, "ypoint": 476}, {"xpoint": 1002, "ypoint": 476},
     {"xpoint": 997, "ypoint": 479}, {"xpoint": 994, "ypoint": 483}, {"xpoint": 994, "ypoint": 491},
     {"xpoint": 990, "ypoint": 497}, {"xpoint": 989, "ypoint": 500}, {"xpoint": 988, "ypoint": 502},
     {"xpoint": 979, "ypoint": 503}, {"xpoint": 971, "ypoint": 507}, {"xpoint": 971, "ypoint": 524},
     {"xpoint": 979, "ypoint": 533}, {"xpoint": 992, "ypoint": 534}, {"xpoint": 1004, "ypoint": 530},
     {"xpoint": 1009, "ypoint": 525}, {"xpoint": 1009, "ypoint": 514}, {"xpoint": 1007, "ypoint": 497},
     {"xpoint": 1007, "ypoint": 490}, {"xpoint": 1011, "ypoint": 483}, {"xpoint": 1008, "ypoint": 478}]}
   }
  ]
}
```


Palette

- **目录**

  标注文件目录结构

  ```
  +--- images
  |   +--- 000000.png
  |   +--- 000001.png
  |   +--- 000002.png
  +--- labels
  |   +--- 000000.png
  |   +--- 000001.png
  |   +--- 000002.png
  ```

  推理文件目录结构

  ```
  +--- 000000.png
  +--- 000001.png
  +--- 000002.png
  ```

- **示例标注/推理文件**

  2D语义分割-palette.png

6.5.4.6 语义分割 3D

Octopus

- **目录**

  标注文件目录结构

  ```
  +--- 1599625710056
  |   +--- 1599625710056.pcd
  |   +--- 1599625710056.json
  +--- 1599625740054
  ```



```
|   +--- 1599625740054.pcd
|   +--- 1599625740054.json
```

推理文件目录结构

```
+--- 1599625710056
|   +--- 1599625710056.json
+--- 1599625740054
|   +--- 1599625740054.json
```

- **示例标注/推理文件**

  3D语义分割-Octopus.json

```
{
  "labels": [
    {
      "label_meta_id": 1,
      "polygon_3d_v2": {
        "ascii_char": "\""
      },
      "shape_type": "polygon_3d_v2",
      "serial_number": 0,
      "label_object_id": -1,
      "name": "car"
    },
    {
      "label_meta_id": 19,
      "polygon_3d_v2": {
        "ascii_char": "4"
      },
      "shape_type": "polygon_3d_v2",
      "serial_number": 0,
      "label_object_id": -1,
      "name": "traffic-sign"
    }
  ],
  "labels_ext": {
    "ascii_string":
"00000000000000000000000000000000000000000000000000004440444400011100
00000000000000000000000000000000000000,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,"
  }
}
```

sementicKITTI

- **目录**

  标注文件目录结构

```
+--- labels
|   +--- 000000.label
|   +--- 000001.label
+--- velodyne
|   +--- 000000.bin
|   +--- 000001.bin
```

推理文件目录结构

```
+--- 000000.label
+--- 000001.label
```

- **示例标注/推理文件**

  3D语义分割-semanticKITTI.label

```
>>> pre = np.fromfile("000000.label", np.uint32)
>>> pre
array([ 0,  0,  0, ..., 48, 48, 48], dtype=uint32)
>>> np.unique(pre)
array([      0,       1,      30,      40,      44,      48,
            50,      51,      52,      70,      71,      72,
            80,      81,      99,   65788,   65790,  131324,
```

```
       131326,   196860,   196862,   589854,   983070,  1572875,
      1638411, 12713994, 12779530, 12845066, 12976138], dtype=uint32)
>>> post = pre & 0xFFFF
>>> post
array([ 0,  0,  0, ..., 48, 48, 48], dtype=uint32)
>>> np.unique(post)
array([  0,   1,  10,  11,  30,  40,  44,  48,  50,  51,  52,  70,  71,
        72,  80,  81,  99, 252, 254], dtype=uint32)
```

6.5.4.7 车道线检测

Octopus

- **目录**

  标注文件目录结构

```
+--- 1628568066600
|   +--- 1628568066600.jpg
|   +--- 1628568066600.json
+--- 1628654064999
|   +--- 1628654064999.jpg
|   +--- 1628654064999.json
```

  推理文件目录结构

```
+--- 1628568066600
|   +--- 1628568066600.json
+--- 1628654064999
|   +--- 1628654064999.json
```

- **示例标注/推理文件**

  车道线检测-Octopus.json

```
{
  "sample_type": "IMAGE",
  "inspection": 2,
  "label_task_id": "24352",
  "label_mode": "manual",
  "inspect_user_ids": [
    "083e18d81600f39a1f48c018646a5697",
    "08b1c9011900257e1fe6c018591d129c",
    "08b1c8a0a5000f491f9dc01820f7478d",
    "098a09be7000251c1f30c018d8df7ac8",
    "0987b0c0b200f3141fb5c0183e769a32",
    "0a36a3849b000f5a1fccc01810ad1ffa"
  ],
  "range_score": -1.0,
  "tags": [
    57,
    41
  ],
  "labels": [
    {
      "label_meta_name": "line_dotted",
      "line": {
        "points": [
          {
            "xpoint": 1656.0662,
            "ypoint": 1080.0
          },
          {
            "xpoint": 1221.8954,
            "ypoint": 745.44366
          },
          {
            "xpoint": 1072.3619,
            "ypoint": 637.4144
          },
          {
```



```
            "xpoint": 1007.8971,
            "ypoint": 595.73413
          },
          {
            "xpoint": 948.4722,
            "ypoint": 561.2173
          }
        ],
        "size": 5
      },
      "shape_type": "line",
      "label_object_id": 0,
      "label_meta_id": 42,
      "serial_number": 0
    }
  ],
  "calibration_id": 15,
  "status": "labeled",
  "label_update_time": 1610332882763,
  "des_order": "sid-20200424000000",
  "label_counts": [
    {
      "label_num": 1,
      "label_meta_shape": "line",
      "label_meta_attr": "",
      "label_meta_desc": "",
      "label_meta_color": "#03A9F4",
      "label_meta_id": 42,
      "label_meta_label_meta_name": "line_dotted"
    },
    {
      "label_num": 7,
      "label_meta_shape": "line",
      "label_meta_attr": "",
      "label_meta_desc": "",
      "label_meta_color": "#F44336",
      "label_meta_id": 43,
      "label_meta_label_meta_name": "line_solid"
    }
  ],
  "label_user_ids": [
    "083e18d81600f39a1f48c018646a5697",
    "08b1c9011900257e1fe6c018591d129c",
    "08b1c8a0a5000f491f9dc01820f7478d",
    "098a09be7000251c1f30c018d8df7ac8",
    "0987b0c0b200f3141fb5c0183e769a32",
    "0a36a3849b000f5a1fccc01810ad1ffa",
    "0a36a3dd63800f541f71c01871a320c9",
    "0a36a48bcd80f5941f5ac0188cc19c86",
    "0a3927a1c000f2cc1f35c018ed7610b1",
    "0aaaa554870026381f3bc01863338c4e"
  ],
  "plate_no": "cid-MDC",
  "project_id": "08229abd258026572f24c01899c784ae",
  "image_meta_info": {
    "sensor": "cam2",
    "source": "https://Octopus-raw-08229abd258026572f24c01899c784ae.obs.cn-
north-4.myhuaweicloud.com/raw-data/relabel/task-24352/202101041057/IMAGE/
1587697451903/1587697451903.jpg",
    "label_meta_name": "1587697451903.jpg",
    "calibration_item_id": 1,
    "size": {
      "height": 1080,
      "width": 1920,
      "depth": 3
    },
    "timestamp": 1587697451903
  },
  "create_time": 1587867585667,
```



```
"difficult": false,
"valid": true}
```

Tusimple

- **目录**

```
test_label.json
```

- **示例标注文件**

车道线检测-Tusimple.json


```
{"lanes": [[-2, -2, -2, -2, -2, -2, -2, -2, -2, -2, 648, 636, 626, 615, 605, 595, 585, 575, 565, 554, 545, 536, 526, 517, 508, 498, 489, 480, 470, 461, 452, 442, 433, 424, 414, 405, 396, 386, 377, 368, 359, 349, 340, 331, 321, 312, 303, 293, 284, 275, 265, 256, 247, 237, 228, 219], [-2, -2, -2, -2, -2, -2, -2, -2, -2, -2, 681, 692, 704, 716, 728, 741, 754, 768, 781, 794, 807, 820, 834, 847, 860, 873, 886, 900, 913, 926, 939, 952, 966, 979, 992, 1005, 1018, 1032, 1045, 1058, 1071, 1084, 1098, 1111, 1124, 1137, 1150, 1164, 1177, 1190, 1203, 1216, 1230, 1243, 1256, 1269], [-2, -2, -2, -2, -2, -2, -2, -2, -2, -2, 713, 746, 778, 811, 845, 880, 916, 951, 986, 1022, 1057, 1092, 1128, 1163, 1198, 1234, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2], [-2, -2, -2, -2, -2, -2, -2, -2, -2, -2, 754, 806, 858, 909, 961, 1013, 1064, 1114, 1164, 1213, 1263, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2]], "h_samples": [160, 170, 180, 190, 200, 210, 220, 230, 240, 250, 260, 270, 280, 290, 300, 310, 320, 330, 340, 350, 360, 370, 380, 390, 400, 410, 420, 430, 440, 450, 460, 470, 480, 490, 500, 510, 520, 530, 540, 550, 560, 570, 580, 590, 600, 610, 620, 630, 640, 650, 660, 670, 680, 690, 700, 710], "raw_file": "clips/0530/1492626760788443246_0/20.jpg"}
{"lanes": [[-2, -2, -2, -2, -2, -2, -2, -2, -2, -2, 642, 634, 626, 618, 610, 602, 594, 586, 578, 570, 562, 554, 546, 538, 530, 522, 514, 506, 498, 490, 482, 474, 466, 458, 450, 442, 434, 427, 419, 411, 403, 395, 387, 379, 371, 363, 355, 347, 339, 331, 323, 315, 307, 299, 291, 283], [-2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, 709, 723, 738, 753, 767, 782, 797, 811, 826, 841, 855, 870, 885, 899, 914, 929, 943, 958, 973, 988, 1002, 1017, 1032, 1046, 1061, 1076, 1090, 1105, 1120, 1134, 1149, 1164, 1178, 1193, 1208, 1222, 1237, 1252, 1267, -2, -2, -2, -2, -2, -2], [-2, -2, -2, -2, -2, -2, -2, -2, -2, -2, 748, 784, 820, 856, 892, 928, 964, 1000, 1036, 1072, 1108, 1144, 1180, 1216, 1252, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2]], "h_samples": [160, 170, 180, 190, 200, 210, 220, 230, 240, 250, 260, 270, 280, 290, 300, 310, 320, 330, 340, 350, 360, 370, 380, 390, 400, 410, 420, 430, 440, 450, 460, 470, 480, 490, 500, 510, 520, 530, 540, 550, 560, 570, 580, 590, 600, 610, 620, 630, 640, 650, 660, 670, 680, 690, 700, 710], "raw_file": "clips/0530/1492628545240318854_0/20.jpg"}
```


6.5.4.8 分类

Octopus

- **目录**

标注文件目录结构

```
+--- 1628568066600
|   +--- 1628568066600.jpg
|   +--- 1628568066600.json
+--- 1628654064999
|   +--- 1628654064999.jpg
|   +--- 1628654064999.json
```

推理文件目录结构

```
+--- 1628568066600
|   +--- 1628568066600.json
+--- 1628654064999
|   +--- 1628654064999.json
```

- **示例标注/推理文件**

分类-Octopus.json

```
{
  "label_meta_name": "Car",
  # 推理文件包含得分
  "score": 0.85
}
```

6.6 编译管理

6.6.1 编译任务

训练产生的模型版本，不可直接被车载芯片识别，需要经过编译工具，将训练产生的模型编译成车载芯片识别的模型。

创建编译任务

步骤1 在左侧菜单栏中选择“训练服务 > 编译管理”。

步骤2 选择“编译任务”页签，单击“新建编译任务”，填写如下信息。

表 6-18 新建编译任务

参数	描述
名称	设置编译任务名称，只能包含数字、英文、中文、下划线、中划线，不得超过32个字符。
描述	简要描述任务信息。不得包含“@^\#$%&*<>'|"/”，不得超过256个字符。
资源规格	选择当前项目中可用的资源用途为“模型编译”的资源规格，可参考**10.1.2 资源管理**创建资源规格。
优先级	设定任务的优先级，数值取[-50,50]的整数，数字越大，优先级越高。
编译镜像	选择编译镜像，需提前创建编译镜像。
参数列表	由编译镜像携带，参数名不支持修改，参数值支持修改。
环境变量	由编译镜像携带，参数名不支持修改，参数值支持修改。
模型	选择待编译模型和版本，模型可以为训练任务产生的模型版本或者通过本地模型文件上传产生的模型版本。

步骤3 以上信息填写无误，单击“创建”。

----结束

编译任务相关操作

在“编译任务”列表，可对任务进行以下操作。

表 6-19 编译任务相关操作

任务	操作步骤
查找任务	在搜索输入框中输入搜索条件，按回车键即可查询。

任务	操作步骤
查看任务详情	单击任务名称，可在任务详情页查看该任务详情、参数详情、编译版本、任务日志和资源在占用情况。 ● 任务详情：任务ID、名称、描述、状态、资源类型等信息。 ● 参数详情：训练算法参数以及环境参数信息。 ● 编译版本：同一个源模型使用不同芯片编译，生成的结果为该模型的不同版本。 ● 任务日志：任务运行过程中生成的日志信息，详情请查看**编译任务日志查看下载**。 ● 资源占用情况：显示任务占用的CPU、内存、GPU及显存占用率百分比的折线图，详情请查看**资源占用情况**。
删除任务	● 单击操作栏的“删除”，删除单个任务。 ● 勾选多个任务，单击列表上方的“删除”，可批量删除任务。
重建任务	单击操作栏内的“重建”，输入新任务名称（以“任务组名-自定义名称”的形式）和是否删除原任务选项，同时可重新选择需要修改的参数。
停止任务	单击该条任务后的“停止”，对停止编译任务。

编译任务相关操作与任务所处状态约束关系请见下表：

表 6-20 操作与状态约束关系

作业状态	重建	删除	停止
排队中	-	√	√
等待中	-	-	√
提交中	-	-	-
提交失败	√	√	-
运行中	-	-	√
运行异常	√	√	-
已完成	√	√	-
停止中	-	-	-
停止失败	-	√	-
已停止	√	√	-
删除中	-	-	-
删除失败	-	√	-

编译任务日志查看下载

编译任务运行过程中生成日志，编译任务模块提供了日志的查看以及下载功能，支持用户查看编译任务的运行情况。编译任务生成的日志文件有以下四种：

- **compile-{id}.log**：用户实际训练任务的训练日志。
- **compile-{id}-init.log**：Octopus平台提供的前置数据的准备日志。
- **compile-{id}-sidecar.log**：Octopus平台提供的任务流程控制日志，包括日志同步、结果上传。
- **octopus-compile-{id}-supplemental.logs**: Octopus平台任务异常退出或停止产生的错误信息输出日志，运行正常时不产生该日志。

在编译任务的详情页面，单击“任务日志”，可查看该编译任务日志详情。支持下载至本地。如果日志较多，用户可在搜索框中输入关键字，查找指定日志内容。

在日志服务页面中的日志列表部分详细展示了该编译任务包含的日志文件的大小以及最新写入时间。单击文件后的“查看”，该文件的详细执行过程则在日志详情部分展示。用户也可在日志文件后的“操作”栏中，单击“下载”，即可将该日志文件下载到本地查看。

资源占用情况

在任务运行中，资源占用情况模块显示任务占用的CPU、内存百分比的折线图。默认显示CPU占用情况折线图。

- 双击任一图例：显示全部资源占用折线图。
- 单击指定图例：只显示该图例折线图。

此模块也可显示多个计算节点运行任务时，资源占用的情况。

- 如果选择2个计算节点运行任务，则可选择查看单个节点资源占用情况。

说明

- 资源占用情况功能模块，需要用户在制作自定义镜像时安装psutil与pynvml，参考命令：**pip install psutil pynvml**
- 如果未安装psutil与pynvml，则页面无法显示资源使用状况。

6.6.2 编译镜像

编译镜像可以将训练模型转换为特定芯片支持的可识别的文件，编译镜像模块支持对编译镜像的增加、查询、删除以及编辑。

创建编译镜像

步骤1 在左侧菜单栏中选择“训练服务 > 编译管理”。

步骤2 选择“编译镜像”页签，单击“新建编译镜像”，填写如下信息。

表 6-21 新建编译镜像

参数	描述
名称	设置编译镜像名称，可包含中英文、数字、“_”“-”，不得超过64个字符。
描述	简要描述，不包含“@^\#$%&*<>'|"/”，不得超过256个字符。
芯片名称	设置芯片名称，可包含中英文、数字、“_”“-”，不得超过64个字符。
镜像	选择镜像仓库和版本。
Boot文件路径	Boot文件路径为在编译镜像中python脚本的绝对路径，不含.py尾缀，比如/home/service/base_compiler_core/customer_service。
参数列表	可以自定义boot文件的启动参数，需要在评测脚本中定义。允许添加的参数个数不超过20个。 ● key：只能由英文、数字、和特殊符号（ ,-_ ）组成，且需要以字母开头 。长度不超过64个字符。 ● value：只能由英文、数字和特殊符号（ \/,.[]-_ ）组成 。长度不超过512个字符。
参数命令	参数命令为运行启动脚本的shell命令，由配置的Boot文件路径和参数列表中参数及参数值组成，填写后会自动生成命令。 **说明** ● 待编译模型挂载路径为${MODEL}=/tmp/data/model。 ● 编译模型结果路径写入${RESULT}=/tmp/result。 ● 用户可使用环境变量获取或直接写入绝对路径。
环境变量	通过注入环境变量至容器中，用户可以快速获取业务相关常量。允许添加的环境变量个数不超过20个。 ● key：只能由英文、数字、和特殊符号（ ,-_ ）组成，且需要以字母开头 。长度不超过64个字符。 ● value：只能由英文、数字和特殊符号（ \/,.[]-_ ）组成 。长度不超过512个字符。

步骤3 以上信息填写无误，单击“创建”。

----结束

编译镜像相关操作

在“编译镜像”列表，可对镜像进行以下操作。

表 6-22 编译镜像相关操作

任务	操作步骤
查询镜像	在搜索输入框中输入搜索条件，按回车键即可查询。

任务	操作步骤
查看镜像详情	单击镜像名称，查看镜像详情。 • 镜像详情：镜像ID、名称、描述、状态、芯片名称、引擎名称、创建时间、更新时间、Boot文件路径等。 • 参数详情：参数列表和环境变量的参数详情。
删除镜像	单击操作栏的“删除”，删除镜像。
编辑镜像	单击操作栏的“编辑”，可修改镜像信息。

6.7 推理服务

新建推理服务

步骤1 在左侧菜单栏中单击“训练服务 > 推理服务”。

步骤2 选择“推理服务”页签，单击“新建推理服务”，填写基本信息。

- 名称：输入推理服务名称，只能包含数字、英文、中文、下划线、中划线，不得超过64个字符。
- 描述：简要描述任务信息。不得包含“@^\#$%&*<>'|"/”，不得超过256个字符。
- 模型：请选择推理服务使用的模型仓库和版本。容器中模型文件默认存放位置：${OCTOPUS_MODEL}，运行镜像将使用当前模型仓库绑定的镜像。用于推理服务的镜像要求参见**“镜像仓库 > 镜像制作（训练）> ModelArts集群”**章节中的推理服务。

  说明

  用于推理服务的模型，应避免在根目录下存在名为config.json的文件，否则可能导致被系统生成的同名配置文件覆盖。
- 资源挂载：请选择可用的资源，目前仅支持挂载模型版本类型的资源，上限为10。
- 资源规格：请选择可用的资源规格，当前仅支持ModelArts类型的资源池，仅支持单卡和8卡两种规格。
- 实例数：设置推理服务实例数，范围1~100，请根据当前集群可用算力设置合适的值。设置为1，表示在当前资源池使用指定的资源规格启动1个实例部署服务。
- 优先级：设定在任务队列中的优先级，数值取[-50, 50]的整数，数字越大，优先级越高。
- 环境变量：通过注入环境变量至容器中，用户可以快速获取业务相关常量。
  - Key：只能由英文、数字、和特殊符号（ ,-_ ）组成，且需要以字母开头 。长度不超过64个字符。
  - Value：只能由英文、数字和特殊符号（ \/,.[]-_ ）组成 。长度不超过512个字符。
- 是否自动停止：可选值为[1, 24] 之间的整数。开启自动停止后，在线服务部署成功后将在您所指定的时间后自动停止以节约算力。后续可以随时打开或关闭该设置。

步骤3 单击“确认”，下发新建推理服务任务。

----结束

推理服务相关操作

在“推理服务”列表，可对任务进行以下操作。

表 6-23 推理服务相关操作

任务	操作步骤
编辑服务	单击操作栏中的“编辑”，打开编辑推理服务弹出框，修改推理服务配置后，单击“确认”。不支持修改服务的名称和模型仓库，当修改了模型版本、资源规格、实例数和环境变量后，根据编辑前的状态，推理服务会重新构建或重新排队，期间服务将不可用。
启动服务	单击操作栏中的“启动”，可以启动服务，在启动服务时，支持重新设置“是否自动停止”配置和自动停止时间。 状态为部署失败或已停止的推理服务才能进行启动操作。
停止服务	单击操作栏中的“停止”，可以停止服务，推理服务停止后再启动将重新构建或排队，请谨慎操作。 状态为构建失败、部署失败、已停止或停止中的推理服务不能进行停止操作。
删除服务	单击操作栏中的“删除”，可以删除服务，推理服务删除后无法恢复，请谨慎操作。
查询服务	在搜索输入框中输入搜索条件，按回车键即可查询。
查看服务详情	单击服务名称，可在服务详情页查看推理服务详情、监控、事件、日志等信息。 ● 推理服务详情：展示ID、状态、模型及版本、镜像、环境变量、资源池、资源规格、实例数、公网地址等信息。其中，公网地址将在服务部署成功后展示。属于私密信息，请勿轻易泄露。 ● 监控：展示AI应用调用次数统计和实时资源统计信息。 ● 事件：展示当前服务使用过程中的关键操作，比如服务构建进度、部署进度、部署异常的原因、服务被启动、停止、更新的时间点等。 ● 日志：展示当前服务的日志信息。支持查询日志，包含最近5分钟、最近30分钟、最近1小时和自定义时间段（自定义时间段您可以选择开始时间和结束时间）。支持输入关键字搜索服务日志。

推理服务各状态允许的操作如下:

表 6-24 推理服务各状态允许的操作

状态	编辑	启动	停止	删除
导入中	-	-	√	√

状态	编辑	启动	停止	删除
构建中	-	-	√	√
构建失败	√	-	-	√
排队中	√	-	√	√
部署中	-	-	√	√
运行中	√	-	√	√
部署失败	√	√	-	√
告警	√	-	√	√
停止中	-	-	-	√
已停止	√	√	-	√

服务监控

- 页面上支持展示当前推理服务的调用总次数和失败次数。
- 支持展示CPU、内存、GPU、显存四种资源的实时占用情况，便于及时调整服务所需资源规格，避免造成资源不足或浪费。

接口访问和调用

1. 获取用户Token。

   调用推理服务接口时首先需要获取IAM子用户Token作为凭据，具体参见**获取IAM用户Token（使用密码）**。请求示例如下：

   请求URI：

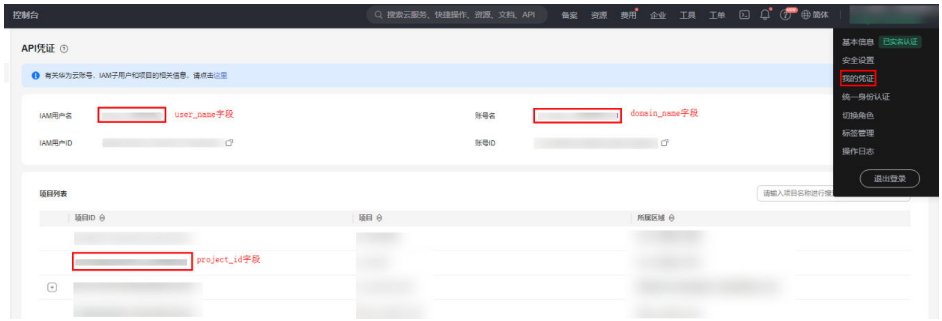
   ```
   POST https://iam.myhuaweicloud.com/v3/auth/tokens?nocatalog=true
   ```

   请求体：

   ```
   {
       "auth": {
           "identity": {
               "methods": [
                   "password"
               ],
               "password": {
                   "user": {
                       "name": "user_name",
                       "password": "user_password",
                       "domain": {
                           "name": "domain_name"
                       }
                   }
               }
           },
           "scope": {
               "project": {
                   "id": "project_id"
               }
           }
       }
   }
   ```

其中，user_name、domain_name、project_id字段可以从“我的凭证”中获取。

图 6-5 获取参数值



请求成功（状态码201）后，从响应的Header中拿到x-subject-token的值即为Token的值。

获取的Token的有效期为24小时。建议进行缓存，避免频繁调用。

2. 获取访问地址。

   访问地址为 {公网地址} /{自定义镜像中的API地址}，例如：https://xxx/v1/infers/xxx/2d-scenario-identification。

   公网地址可以从推理服务列表或者服务详情中获取。

图 6-6 列表获取公网地址

图 6-7 详情获取公网地址

3. 调用推理服务。

   请携带Token调用推理服务。

6.8 任务队列

任务队列页面展示在训练服务创建的所有类型任务，包括任务的名称、类型、资源规格、实例数、优先级、工作空间、创建者、创建时间等信息。同时支持跳转至指定任务详情页和修改任务的优先级。

训练服务的各类型任务在提交创建之后，都将进入对应集群的任务队列等待调度器统一调度。调度器默认按照先调度优先级高的任务，同优先级的任务按照进入队列的时间，先进先出进行调度，支持在界面手动调整调度顺序。最终创建失败的任务不展示在任务列队页面。

步骤1 在左侧菜单栏中单击“训练服务 > 任务队列”。

步骤2 选择集群名称，可查看对应集群内的所有任务信息。

步骤3 管理该集群内的任务。

- 查看指定任务详情：单击任务名称进入该任务的详情页面。
- 修改任务调度顺序：
  - 置顶任务：将该任务置于队列首位。单击“操作”列的“置顶”，再单击“确定”。
  - 置底任务：将该任务置于队列末位。单击“操作”列的“置底”，再单击“确定”。
  - 上移任务：将该任务在队列中的位置前移一位，选择“操作”列的“更多 > 上移”，再单击“确定”。
  - 下移任务：将该任务在队列中的位置后移一位，选择“操作”列的“更多 > 下移”，再单击“确定”。
  - 更新优先级：勾选任务，单击“更新优先级”，输入[-50,50]的整数，数字越大，优先级越高。单击“确定”，优先级更改成功。

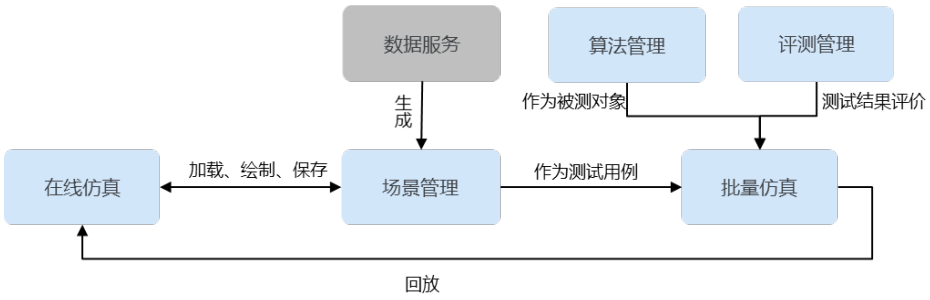
----结束

7 仿真服务

7.1 仿真服务简介

Octopus仿真服务支持多种功能操作。包括用户在云上以类似操作远程桌面方式操作图形化界面的仿真软件的在线仿真服务，基于OpenSCENARIO等标准格式的仿真场景管理。泛化大量仿真场景，规控算法工程管理，多场景并行高速运行的批量仿真服务。用户可通过仿真服务完成在线仿真、仿真场景，创建仿真评测任务等。仿真服务开发流程如下。

图 7-1 仿真服务开发流程



仿真服务操作引导如下：

在线仿真：提供类似本地开发的体验，支持客户直接使用线上仿真器开发，并无痛对接云端的场景管理：包含加载、绘制、保存等操作。

算法管理：用于对接客户的上云算法，并支持算法的版本级管理，并可自动化触发关联的批量算法。

评测管理：支持内置评测配置和自定义评测镜像，对仿真任务中的算法展开评测。

场景管理：包含场景、场景库、逻辑场景、逻辑场景库、测试用例、测试套件等。支持页面上传、泛化、在线仿真编辑等场景录入方式，支持地图、场景的在线预览，并支持场景标签等功能。

任务管理：选择仿真算法和仿真场景创建仿真任务，从行车安全、驾驶行为、乘员舒适性等角度衡量仿真算法控制效果，支持可视化仿真结果。

7.2 在线仿真

使用规范

在线仿真服务集成了一整套完整的仿真工具链，开箱即用。

说明

1. 在使用新的普通用户登录在线仿真时，需修改在线仿真系统配置：
   - 关闭桌面锁屏。
   - 调整屏幕分辨率（默认分辨率不高，可调至适合的分辨率）。
   - 将默认的浏览器更改为Chrome或Firefox。
2. 在线仿真根据用户名称生成系统进程，Linux中不合法的用户名称登录在线仿真会出现异常，合法的用户名称限制如下：
   - 只能包含小写字母、数字、下划线、中划线 。不得超过31个字符。
   - 只能以字母和下划线开头。不可包括连续的下划线和中划线。不可包含linux的关键字或保留字。
3. 目前在线仿真为独占式，同一台在线仿真器同一时间只允许一个用户使用，待用户释放后，才可供其他用户使用。

7.2.1 仿真器

进入、占用、释放仿真器

Octopus平台在线仿真模块为用户提供了在线仿真器。单击  进入在线仿真器，用户进入机器后，机器显示红色按钮  占用，用户退出在线仿真界面不会自动释放机器，会保持占用状态。机器图片的左上角会出现释放按钮  ，只有占用中的用户可以释放。

图 7-2 仿真器

加载场景

Octopus平台支持用户在仿真器中加载场景库中的场景，在线编辑、运行并回放。

步骤1 单击  进入在线仿真页面，单击页面右上方的“加载场景”。

步骤2 选择加载场景方式。

- 选择需要加载的场景、泛化场景或测试用例。
- 如果场景数量多，用户也可根据场景的标签进行过滤，并选择需要加载的场景。
- 在搜索框中输入搜索内容，单击搜索，并在筛选出的场景、测试用例或泛化场景列表中勾选中需要加载的场景。

  多项搜索：可根据需要决定是否启用多项搜索，输入多个关键字，中间用“;”隔开，可搜索多个关键字。

步骤3 单击“确认”，仿真场景加载成功。

步骤4 查看加载场景过程文件。

使用仿真器在线加载场景后，会在“/home/{user}/workspace/Data/Project/Current/Scenarios/”目录下出现加载过的场景文件，打开指定日期的场景文件夹。单击打开该场景文件夹下的“.xml”文件，即可查看下载下来的“osgb”以及“xodr”文件所在路径。

图 7-3 文件所在路径

----结束

保存场景

Octopus平台支持用户在仿真器上利用ScenarioEditor和RoadDesigner模块，共同完成对道路交通场景、车辆传感器等自定义设计，并保存在场景库中，操作步骤如下：

步骤1 单击  进入在线仿真页面，选取场景保存路径。

单击“保存场景”，可以根据场景名称关键字检索，或根据时间周期（最多半年以内）筛选最近加载或修改过的文件列表， 选取场景保存的具体路径。

步骤2 填写场景基本信息。

填写场景的名称、描述信息，选取优先级、场景标签，并勾选协议。

- 场景名称：只能包含数字、英文、中文、下划线、中划线、点，且不支持以点结尾，不得超过256个字符。
- 优先级：当前支持S、A、B、C、D。级别顺序为：S > A > B > C > D。

- 场景描述：不得包含“@#$%^&*< > \”字符，不得超过255个字符。
- 添加场景标签，标签数量不超过50个。
  a. 直接选择：单击“添加标签”，从场景标签中直接选择标签，也可新建标签。
     多项搜索：可根据需要决定是否启用多项搜索，输入多个关键字，中间用“;”隔开，可搜索多个关键字。
  b. Json文件导入：单击“选择json文件”，可选择本地的json文件，直接导入标签。
- 勾选“我已阅读并同意《八爪鱼自动驾驶云服务使用声明》”。

步骤3 单击确认，页面提示“保存场景成功”，表示用户的场景已经保存成功。

----结束

加载算法

加载算法的主要目的是在进行联合仿真时，接入外部控制算法实现闭环控制。通过引入外部的算法，可以提高仿真的精度和复杂度，同时由于算法的分离，可以加快开发进度。

步骤1 单击  进入在线仿真页面，单击页面右上方的“加载算法”。

步骤2 在弹出框中选择算法镜像和版本，单击“确认”即可运行算法。

说明

需提前准备相关可用算法，算法运行失败不会再次运行算法，创建算法可参考**算法列表相关操作**。

算法加载成功后，可以展开界面右侧子窗口，查看仿真器相关信息和算法信息。可单击算法信息中的“停止”按钮来停止使用该算法。也可以再次单击“加载算法”选择其他的算法。

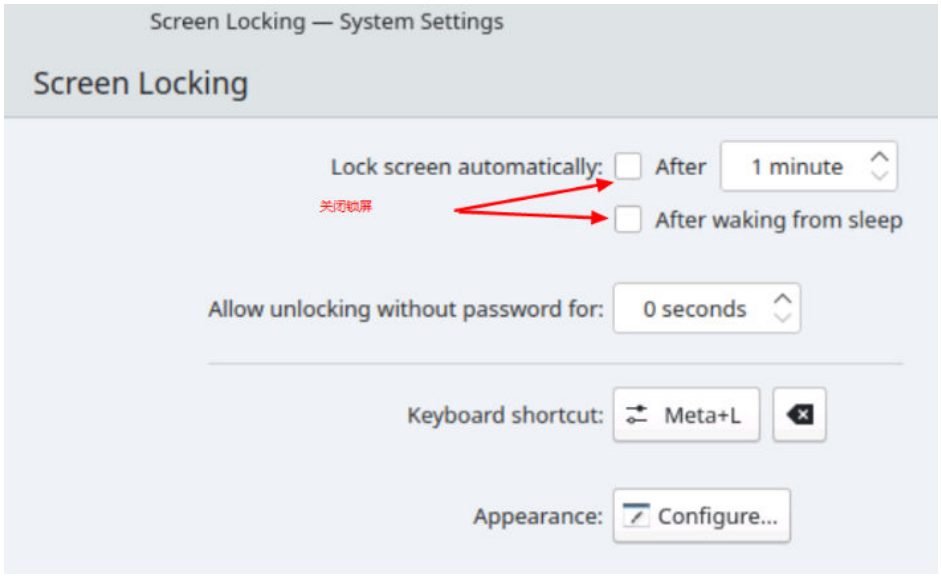
----结束

7.2.2 在线仿真配置

使用在线仿真器，需要用户进行一些单独配置。

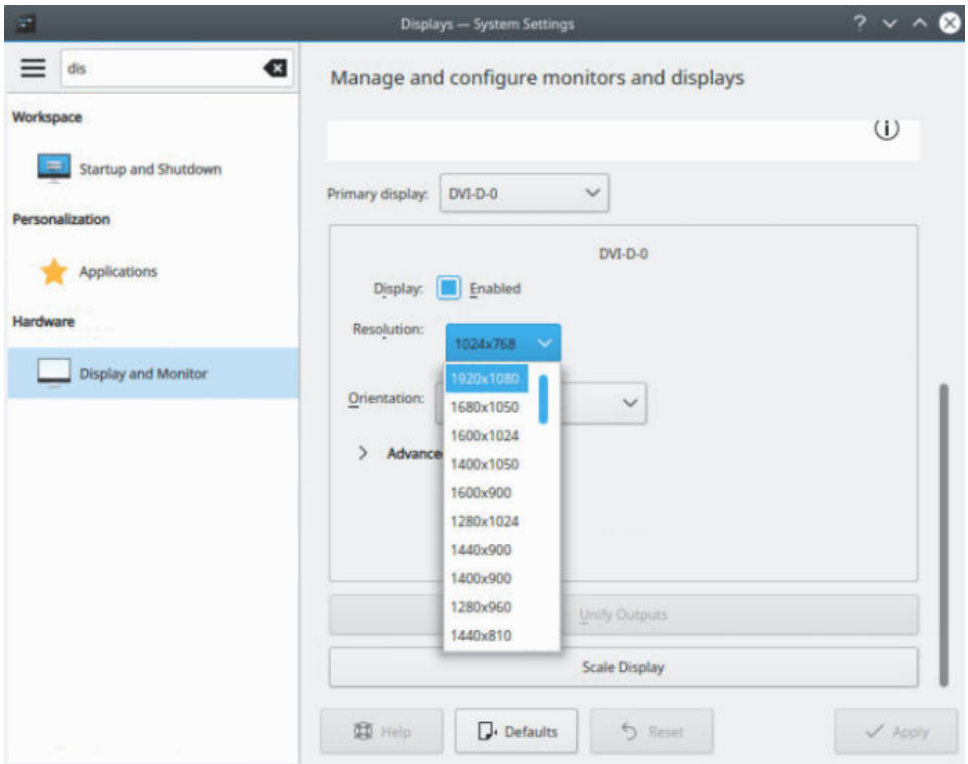
步骤1 关闭锁屏。

图 7-4 关闭锁屏



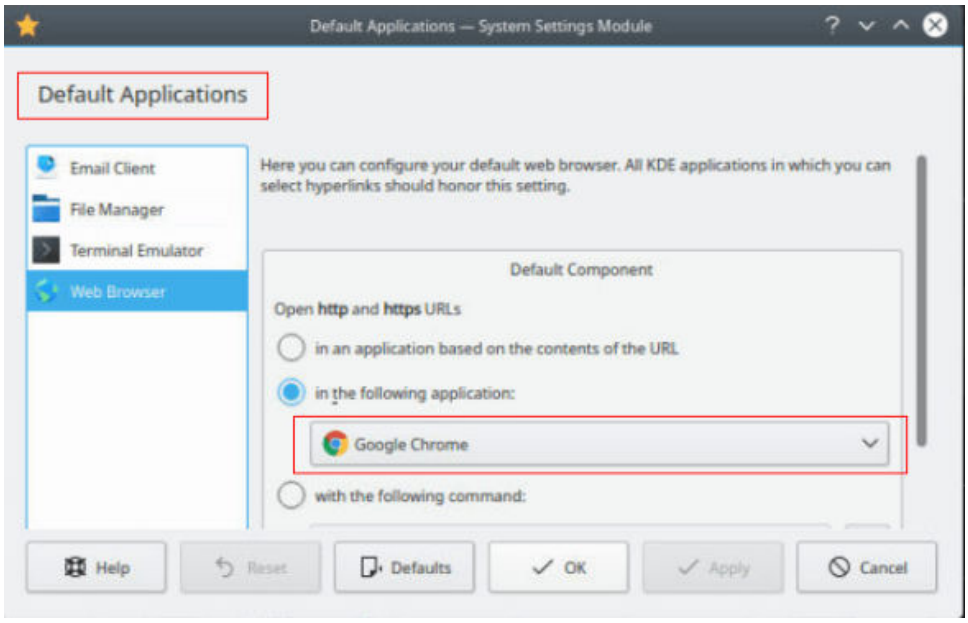
步骤2 调整屏幕分辨率。

图 7-5 调整屏幕分辨率



步骤3 修改默认浏览器（改成chrome或firefox，使用KDE自带的Konqueror目前有兼容问题）。

图 7-6 修改默认浏览器



----结束

7.3 算法管理

7.3.1 算法创建

步骤1 在左侧菜单栏中单击“仿真服务 > 算法管理”。

步骤2 单击“新建算法”，填写算法基本信息。

- 算法类型：固定为“容器镜像”。
- 算法名称：只能包含数字、英文、中文、下划线、中划线，不得超过64个字符。
- 算法描述：不得包含“@#$%^&*< > \”特殊字符，不得超过255个字符。
- 镜像选择：下拉选择仿真算法镜像。

说明

如果镜像仓库中的算法镜像已被其他仿真算法引用，需要在镜像仓库中重新创建算法镜像。

步骤3 配置运行环境。

- 运行命令：输入运行算法的命令，具体命令根据仿真算法镜像启动脚本确定。示例命令如：bash start.sh，python main.py等。

说明

运行命令需满足以下条件：

- 不能为空。
- 必须是满足ASCII码的字符串。
- 不能包含特殊字符\@#$%^&*<>。
- 不能超过255个字符。

- 关键字：根据需要填写算法启动关键字。
- CPU：输入CPU核数。
- 内存：输入内存大小。
- **说明**

  如果需要使用关键字功能，请确保算法程序可以在前台（stdout）打印该日志。建议使用日志库实现输出，如果使用printf等调试打印，可能结果会无效。

步骤4 以上信息填写完成后，单击“创建”新建算法。

步骤5 查看算法详情。

算法新建后，在“算法列表”可以查看新建的算法，单击指定“算法名称”，可以查看算法的基本信息和算法详情。

----结束

算法列表相关操作

表 7-1 算法列表相关操作

任务	操作步骤
搜索算法	在搜索框中输入关键字搜索相关算法。支持通过算法名称和算法ID搜索。
查看算法详情	单击算法名称，即可查看算法详情页。
编辑算法	单击操作栏的“编辑”，即可编辑算法信息。
删除算法	单击操作栏的“删除”，即可删除算法信息。 **说明** 不可删除被任务占用的算法。

7.3.2 算法详情

基本详情

单击指定算法名称，可以查看算法的基本信息、算法详情、任务配置以及镜像版本等信息。

任务配置

当创建任务配置时，如果关联了算法配置，则在算法详情页，会展示此算法关联的批量仿真的任务配置信息，在此模块还可进行以下操作。

表 7-2 任务配置相关操作

任务	操作步骤
搜索任务配置	筛选“任务配置ID”和“任务配置名称”和“是否开启自动化触发”，在搜索框中输入关键字搜索相关任务配置。
查看任务配置详情	单击任务配置名称，页面跳转至任务配置详情页。
开启自动化触发	可开启或者关闭自动触发功能，开启后，系统会自动触发“任务管理”中关联的仿真任务。

镜像管理

步骤1 选择“仿真服务 > 算法管理”，单击指定算法名称。

步骤2 在算法详情页，单击“创建镜像”，输入镜像版本。

步骤3 单击“确认”，镜像列表可查看创建完成的镜像。

步骤4 删除镜像。

单击指定镜像名称后“操作”栏内的“删除”。删除该镜像。删除后不可恢复，请谨慎操作。

----结束

7.4 评测管理

7.4.1 内置评测配置

创建内置评测配置

步骤1 在左侧菜单栏中选择“仿真服务 > 评测管理”。

步骤2 单击“新建评测”，填写基本信息。

- 评测名称：只能包含数字、英文、中文、下划线、中划线，不得超过64个字符。
- 评测描述：不得包含“@#$%^&*< > \”特殊字符，不得超过255个字符。
- 评测类型：分“内置评测配置”和“自定义评测镜像”，选择“内置评测配置”。
- 评测模式：选择“实时评测”或“延时评测”。详情请参考**实时评测和延时评测介绍**。

步骤3 以上信息填写无误后，单击“确认”。

----结束

查看内置评测配置详情

评测类型为内置评测配置时，项目详情包含评测基本信息和自定义评测配置两部分。

- 基本信息：评测名称、评测模式、更新时间等信息。
- 自定义评测配置：该项目包含内置评测指标等信息，其中碰撞为默认，支持添加和删除评测指标。

编辑评测指标

评测类型为内置评测配置时，可为评测添加或删除评测指标，便于管理。

步骤1 单击评测名称，在评测项目详情页，自定义评测配置部分，单击“编辑”。

步骤2 单击“添加评测指标”，选择需要添加的内置评测的指标，单击“确认”。

步骤3 设置指标的阈值和重要度。

- 阈值设置：单击“阈值设置”列中的编辑按钮，可设置对应阈值。部分指标支持设置阈值，请以界面展示为准。
- 重要度：在“重要度”列选择主要或次要。

步骤4 设置评分方案。

可选主要次要log函数、主要次要均匀权重、全部均匀权重。具体介绍请查看**评测分数计算介绍**。

步骤5 删除评测指标。

单击评测指标“操作”列内的“删除”，删除该评测指标。

说明

被任务使用的评测项目和镜像不能删除。

步骤6 以上信息选择无误后，单击“保存”，评测指标编辑完成。

----结束

7.4.2 自定义评测镜像

创建自定义评测镜像

步骤1 在左侧菜单栏中选择“仿真服务 > 评测管理”。

步骤2 单击“新建评测”，填写基本信息。

- 评测名称：只能包含数字、英文、中文、下划线、中划线，不得超过64个字符。
- 评测描述：不得包含“@#$%^&*< > \”特殊字符，不得超过255个字符。
- 评测类型：分“内置评测配置”和“自定义评测镜像”，选择“自定义评测镜像”。
- 评测镜像：选择在镜像仓库中创建好的评测镜像。
- 评测模式：选择“实时评测”或“延时评测”。详情请参考**实时评测和延时评测介绍**。

步骤3 以上信息填写无误后，单击“确认”。

----结束

查看自定义评测镜像详情

评测类型为自定义评测镜像时，项目详情包含评测基本信息和评测镜像两部分。

- 基本信息：评测名称、评测名称、更新时间等信息。
- 评测镜像：该项目包含的镜像版本信息。

创建镜像

评测类型为自定义评测时，可为评测创建多个镜像版本，便于管理。

步骤1 单击评测名称，在评测项目详情页，单击“创建镜像”，填写基本信息。

- 镜像版本：选择镜像版本。
- 运行命令：输入镜像的运行命令，具体命令根据镜像启动脚本确定。示例命令如：bash start.sh，python main.py等。

  说明

  运行命令需满足以下条件：
  - 不能为空。
  - 必须是满足ASCII码的字符串。
  - 不能包含特殊字符\@#$%^&*<>。
  - 不能超过255个字符。

步骤2 以上信息填写无误后，单击“确认”。

步骤3 编辑镜像。

单击镜像版本“操作”列的“编辑”，即可修改镜像运行命令。

步骤4 删除镜像。

单击镜像版本“操作”列的“删除”，删除该镜像版本。删除后不可恢复，请谨慎操作。

----结束

7.4.3 内置评测指标说明

7.4.3.1 内置评测指标简介

评测算法从驾驶安全性，合规性，智能性，舒适性维度对自动驾驶系统进行全面评价。评测指标的pass/fail标准比较复杂，需要对一些评测函数的细节进行介绍。

point_type：是一个PointType的枚举类型，表示该子类指标发生特殊状态（一般是指发生异常）时的时刻点用哪种形式存储起来。目前Octopus使用的PointType共有以下4种类型：

表 7-3 PointType 类型

类型	说明
POINT_TYPE_POINT	表示该子类指标的异常时间点是离散的时间点形式，在任何时刻都可能发生异常。

类型	说明
POINT_TYPE_REGION	表示该子类指标的异常时间点是区间形式，一旦在某个时刻开始发生异常，则在随后一段时间内都会处于异常状态。
POINT_TYPE_ALL	表示该类指标的异常时间点是布尔形式的，从仿真开始到当前时刻的状态要么是完全通过，要么全过程都是异常的，统计类型的指标需要以这种形式表示。
POINT_TYPE_NORMAL	该类型与其他类型相反，如果该类型的点存在，则表示对应的子类指标是通过的，Octopus用该类型保存主车到达终点的时间值。

7.4.3.2 安全性评测指标

碰撞（Collision）检测

碰撞检测的目的是判断主车是否与其它交通参与物发生碰撞。

在进行碰撞检测时，根据与主车碰撞的物体类型的不同对碰撞类型进行细分。

具体分为：

- 车的车碰撞检测
  - 追尾检测
  - 被追尾检测
  - 正面对碰检测
  - 垂直角度碰撞检测
  - 斜角侧碰检测
- 车人碰撞检测
- 自行车碰撞检测
- 摩托车碰撞检测
- 静态障碍物检测
- 道路周边设施碰撞检测
  - 碰撞电线杆检测
  - 碰撞房屋检测
  - 碰撞树木检测
  - 碰撞绿化植被检测
  - 碰撞交通标志检测
  - 碰撞路边栅栏检测
- 未知类型物体碰撞检测

在每个类型的碰撞检测上，会同时进行两种检测。分别为：

- 是否碰撞检测
- 是否减速响应检测
- 是否转向响应检测

其中是否碰撞检测判断该种碰撞是否发生，在碰撞发生的基础上，进一步地判断主车是否有提前响应的动作。

当主车有提前减速或者转向避让，但只是没能及时刹住，本设计认为这种情况比完全没有采取任何措施避免碰撞的表现要好。

是否响应的判断是基于碰撞发生时，主车是否制动减速或者转向，发生了制动减速的标准是减速度大于$3m/s^2$，发生了转向的标准是横摆角速度大于$0.02rad/s$，则主车进行了避免碰撞的响应措施。

另外，对于车的车碰撞，本设计根据碰撞方位进行了细分。

当主车和发生碰撞的副车的夹角在$(0^\circ,15^\circ)$或者$(345^\circ,360^\circ)$内，并且主车位于副车后方，则认为发生追尾碰撞。

当主车和发生碰撞的副车的夹角在$(0^\circ,15^\circ)$或者$(345^\circ,360^\circ)$内，并且副车位于主车后方，则认为发生被追尾碰撞。

当主车与副车的碰撞夹角在$(165^\circ,195^\circ)$内时，则认为发生正面对碰。

当主车与副车的碰撞夹角在$(75^\circ,105^\circ)$或者$(255^\circ,285^\circ)$内时，则认为发生垂直角度碰撞。

当主车与副车的碰撞角度在$[15^\circ,75^\circ]$或$[105^\circ,165^\circ]$或$[195^\circ,255^\circ]$或$[285^\circ,345^\circ]$内时，则认为发生斜角侧碰。

该指标关联的内置可视化时间序列数据为：暂无。

该指标的异常时间点记录类型为：POINT_TYPE_POINT。

碰撞时间（Time to Collision）检测

碰撞时间检测的目的是判断主车在行驶中与其他交通车的碰撞时间是否过小。

碰撞时间是指主车与引导车的相对距离除以主车与引导车的相对速度。

即使主车未发生碰撞，当碰撞时间过小时，发生碰撞的风险太大，这样也是不合理的。

当碰撞时间小于某一阈值（该阈值可用户自定义，本设计默认取1.5s），则判定碰撞时间检测不通过。

该指标关联的内置可视化时间序列数据为：暂无。

该指标的异常时间点记录类型为：POINT_TYPE_POINT。

车头时距（Time Headway）检测

车头时距检测的目的是判断主车行驶过程中与其他交通车的车头时距是否台小。

车头时距是主车与引导车的相对距离除以主车的速度。

即使主车未发生碰撞， 当车头时距过小时（该阈值可用户自定义，本设计默认取2s）， 发生碰撞的风险太大， 这样也是不合理的。

车头时距和碰撞时间两者都是描述碰撞风险大小的。

车头时距适合判断主车和引导车速度都很高， 但相对速度比较小的情况。

碰撞时间适合主车和引导车相对速度比较大的情况。

该指标关联的内置可视化时间序列数据为：暂无。

该指标的异常时间点记录类型为：POINT_TYPE_POINT。

在路（On Road）检测

在路检测的目的是判断主车是否在可行驶的道路上驾驶。

根据OSI中车道类型定义，当主车行驶的道路类型为osi3.Lane.classification.type.TYPE_NONDRIVING，则认为主车在路检测不通过。

该指标关联的内置可视化时间序列数据为：暂无。

该指标的异常时间点记录类型为：POINT_TYPE_POINT。

车道保持（Lane Keeping）检测

车道保持检测的目的是判断主车在行驶过程中能否很好地沿车道中心线行驶。

车道保持检测分为两个指标：

- 偏移车道中心线距离检测
- 偏移车道中心线横摆角检测

偏移车道中心线距离检测是指主车的质心相对于车道中心线的垂直距离，当该偏移距离大于某一阈值时（本设计取0.3m，该阈值可以用户自定义），则偏移车道中心线距离检测不通过。

偏移车道中心线横摆角检测是指主车行驶时速度方向与车道中心线的夹角，当该夹角大于某一阈值时（本设计取0.05rad，该阈值可以用户自定义），则偏移车道中心线横摆角检测不通过。

车道保持检测需要排除主车进行了换道操作，对于换道期间进行偏移车道中心线距离检测和偏移车道中心线横摆角检测，将会出现假阳性的结果。

当主车所在的road id保持不变，在某一时刻，其lane id发生变化，在该时刻的前后一定时间内（本设计取2s）发生换道。

车头横摆角偏离检测关联的内置可视化时间序列数据为：relativeYaw。横向偏移距离检测关联的内置可视化时间序列数据为：lateralOffset。

该指标的异常时间点记录类型为：POINT_TYPE_POINT。

7.4.3.3 合规性评测指标

压实线（Onto Solid line）检测

压实线检测的目的是判断主车行驶过程中是否压到实线。

当主车与距离最近的车道线的小于主车宽度的一半时，并且该车道线的类型为OSI定义的osi3.LaneBoundary.classification.type.TYPE*SOLID*LINE，则认为主车的轮胎已经压到实线。

该指标关联的内置可视化时间序列数据为：暂无。

该指标的异常时间点记录类型为：POINT_TYPE_POINT。

逆行（Reverse Direction Driving）检测

逆行检测的目的是判断主车行驶过程中是否按车道规定的方向行驶。

根据OPNENDRIVE中对车道的lane id的定义， 沿着道路的reference line的前进方向，reference line右侧的lane id由0逐渐递减，左侧的lane id由0逐渐递增。

当主车前进方向与道路的方向相同时， 使标记值flag=1， 当主车与道路前进的方向相反时， 使标记值flag=-1。

当flag与主车所在的lane id的乘积大于0时， 则可以判定主车发生逆向行驶。

该指标关联的内置可视化时间序列数据为：暂无。

该指标的异常时间点记录类型为：POINT_TYPE_POINT。

限速（Speeding）检测

限速检测的目的是判断主车的车速是否超过道路默认限速。

本设计采用最大默认限速120km/h。

该阈值可通过前端进行自定义配置。

限速标志牌前限速（Speed Limit Sign）检测

限速标志牌前限速检测的目的是判断主车在行驶过程中遇到限速标志牌时， 速度是否符合要求。

限速标志牌分为最高限速和最低限速两种。 最高限速是指主车速度不能高于对应的限速数值， 并且不能低于最高限速的75%。

最低限速是指主车速度不能低于对应的限速数值。

当主车距离限速标志牌在道路方向的距离小于某一阈值（本设计取车辆最前端超过限速标志）， 并且主车所在车道是限速标志牌的有效范围， 当主车速度高于最高限速标志数值或低于最低限速标志数值时， 限速标志牌限速检测不通过。

该指标关联的内置可视化时间序列数据为：暂无。

该指标的异常时间点记录类型为：POINT_TYPE_POINT。

指示标志牌前行为（Mandatory Sign）检测

指示标志牌前行为检测的目的是判断主车在这些指示标志牌前的行为是否合理，本设计考虑的指示标志牌有：

- 左转指示牌
- 右转指示牌
- 直行指示牌
- 左转直行指示牌
- 右转直行指示牌
- 左转右转指示牌

- 靠左行驶指示牌
- 靠右行驶指示牌

当主车前端超过左转指示牌， 并且主车不存在左转行为， 则左转指示牌前行为检测不通过。

当主车前端超过右转指示牌， 并且主车不存在右转行为， 则右转指示牌前行为检测不通过。

当主车前端超过直行指示牌， 并且主车不存在直行行为， 则直行指示牌前行为检测不通过。

当主车前端超过左转直行指示牌， 并且主车不存在左转直行行为， 则左转直行指示牌前行为检测不通过。

当主车前端超过右转直行指示牌， 并且主车不存在右转直行行为， 则右转直行指示牌前行为检测不通过。

当主车前端超过左转右转指示牌， 并且主车不存在左转右转行为， 则左转右转指示牌前行为检测不通过。

该指标关联的内置可视化时间序列数据为：暂无。

该指标的异常时间点记录类型为：POINT_TYPE_REGION。

警告标志牌前行为（Warning Sign）检测

警告类交通标志前行为检测的目的是判断主车在各种警告类标志前行为是否合理，主要包括两个方面的检测：

- 在警告类标志前车速是否太大
- 在警告类标志前是否有明显的加速行为

本设计认为当主车的车速大于$30\mathrm{km/h}$或者加速度大于$1\mathrm{m/s^2}$时，警告类标志前行为检测不通过。

该指标关联的内置可视化时间序列数据为：暂无。

该指标的异常时间点记录类型为：POINT_TYPE_REGION。

禁止标志牌前行为（Prohibited Sign）检测

禁止标志牌前行为检测的目的是判断主车在这些禁止类标志牌前的行为是否合理。

本设计考虑评测的禁止标志牌有：

- 禁止机动车标志牌
- 禁止各种车辆标志牌
- 禁止驶入标志牌
- 限制宽度标志牌
- 限制高度标志牌
- 限制重量标志牌

在禁止机动车标志牌，禁止各种车辆标志牌，禁止驶入标志牌等标志前， 当检测到主车的车头越过标志牌并且标志牌的对主车起作用时， 该类标志前的行为检测不通过。

对于限制高度、限制宽度、限制高度等禁止标志牌， 在满足上述触发条件的情况下，并且主车的对应度量值（如高度、宽度、重量）大于标志牌禁止的数值， 则主车在该类标志牌前的行为检测不通过。

该指标关联的内置可视化时间序列数据为：暂无。

该指标的异常时间点记录类型为：POINT_TYPE_REGION。

7.4.3.4 智能性评测指标

换道（Lane Change）检测

换道检测的目的是判断主车在换道过程中的换道持续时间以及换道时的侧向加速度是否合理。

换道是指当主车所在的road id保持不变， 在某一时刻，其lane id发生变化， 在该时刻的前后一段时间内主车处于换道过程。

对于判定换道时的侧向加速度是否合理， 本设计考虑换道时刻的前后2s的时间段内的侧向加速度是否太大， 对于侧向加速度大于一定阈值（可用户自定义，本设计取2m/s2）， 则换道时的侧向加速度检测不通过。

对于判断换道持续时间是否合理， 本设计以主车相对车道中心线的偏离横摆角作为换道开始和结束的判定标记， 在换道时刻之前的第一个偏离横摆角小于0.03rad的时间点为换道开始点， 在换道时刻之后第一个偏离横摆角小于0.03rad的时间点为换道结束点。

换道开始点到换道结束点的时间长度作为换道持续时间， 当换道持续时间小于某一阈值（可用户自定义，本设计取1.5s）时， 或换道时间大于某一阈值（可用户自定义，本设计取6s）时， 换道持续时间检测不通过。

该指标关联的内置可视化时间序列数据为：暂无。

该指标的异常时间点记录类型为：POINT_TYPE_POINT。

到达终点（Reach Destination）检测

到达终点检测的目的是判定主车是否到达场景文件中指定的全局路径规划的终点。

当主车的车辆坐标系原点进入终点为半径R（本设计取R为2m）范围内时， 则判定主车到达了终点。

在没有设置终点时， proto协议会把目标点默认初始化(0,0,0)， 因此本设计认为当终点为(0,0,0)时， 是没有终点的情况， 故不允许场景设计者在设计场景终点时将终点设置为(0,0,0)。

该指标关联的内置可视化时间序列数据为：暂无。

该指标的异常时间点记录类型为：POINT_TYPE_NORMAL。

红灯前行为（Run Red Light）检测

红灯前行为检测的目的是判断主车在遇到红灯时能否在停止线前停车， 并且与停止线的距离保持在合理的范围。

判断能否在停止线前停车是指当主车前端超出停止线后， 主车速度大于零时， 则主车没能在停止线前停车。 这要排除主车在非箭头红绿灯右转的情况。

判断主车停车后距离停止线是否合理时， 如果主车在距离停止线[2,20]范围内发生停车行为， 则停车后与停止线的距离不合理。

如果主车在停止线[0.1, 2]m范围内发生停车行为，判断停止距离合理。

绿灯通行（Drive Through Green Light）检测

绿灯通行检测的目的是判断主车在接近十字路口后， 如果是绿灯， 主车是否直接通行而没有停止。 另外，当交通灯由红灯变为绿灯后， 主车重新启动的时间是否太大。

本设计认为在绿灯状态下， 如果前方没有行人和引导车的情况下， 主车在停止线前20m范围内发生停车行为， 则绿灯前直接通行不通过。

当交通灯由红灯变为绿灯后， 如果主车重新启动的时间大于一定阈值（本设计取3s）， 则绿灯后重新启动时间太大。

该指标关联的内置可视化时间序列数据为：暂无。

该指标的异常时间点记录类型为：POINT_TYPE_POINT。

礼让行人（Polite To Pedestrian）检测

礼让行人检测的目的是判断当行人横穿马路时， 主车是否有礼让行为。

具体的礼让行为包括在行人横穿马路过程中， 进行停车已经停车距离要合适， 并且当行人离开车道后， 主车重新启动时间要合适。

其中停车距离允许用户自定义，本设计取主车前端距离行人最小距离min不低于1m，距离行人最大距离max不高于5m。

是否礼让通过判断车辆前端纵向距离行人低于min/2，速度是否大于0时，如果速度大于0，则没有进行礼让。

重新启动时间同样允许用户自定义，本设计去的默认最长重新启动时间为3s。

当主车行为不满足对应的条件时，则行人横穿马路礼让检测不通过。

该指标关联的内置可视化时间序列数据为：暂无。

该指标的异常时间点记录类型为：POINT_TYPE_POINT。

应对对向车辆占道（Encroaching vehicle）检测

在双向车道路上， 会存在主车与对向车辆存在横向冲突的情况， 应对对向车辆冲突行驶检测的目的是判断主车在这种情况下， 能否进行适当的转向和减速避让， 从而保证安全性。

其中主车需要进行避让的前提条件是：

当主车前端与对向行驶的车道纵向距离一定范围内（本设计取10m）， 并且主车与对向车辆的横向距离小于两者一半车宽的和。

当满足该条件后， 如果主车没有进行转向避让和减速避让， 则对应的检测不通过。

减速避让检测不通过是指主车的加速度大于0。

其中转向避让检测考虑到中国是靠右行驶， 在设计该类测试场景时， 对向车会是在主车的左侧。 转向避让检测不通过是指主车没有向右边转向。

该指标关联的内置可视化时间序列数据为：暂无。

该指标的异常时间点记录类型为：POINT_TYPE_REGION。

跟车起停（Stop and Go）检测

跟车起停检测的目的是判断主车跟随前车停车后能否在前车启动后重新启动。

当主车跟随前车制动停止后， 前车重新启动后， 主车重新启动的时间要合适， 该时间允许用户自定义， 本设计默认取3s。

当重新启动时间大于指定阈值时， 则跟车起停检测不通过。

该指标关联的内置可视化时间序列数据为：暂无。

该指标的异常时间点记录类型为：POINT_TYPE_REGION。

通行速率（Efficiency）检测

通行速率用于评价主车在场景中从起点到终点的效率，主车越快到达终点，则通行速率越高。

本设计取通行速率的默认阈值为0m/s，即如果主车平均速度小于等于0，则该指标不通过。

通行速率指标可有效避免主车一直不动，其他评测指标均通过，导致得分却很高的情况发生。

该指标关联的内置可视化时间序列数据为：speedX。

该指标的异常时间点记录类型为：POINT_TYPE_ALL。

预警系统激活（Warning）检测

预警系统激活用于评价算法是否激活以下五项预警功能:

- 盲区预警
- 前方碰撞预警
- 车道偏离预警
- 泊车碰撞预警
- 后方横向车流预警

当算法pb中检测到预警项且状态为STATE_ACTIVE，则视该预警为激活态，否则为非激活态；当预警状态从非激活态转变为激活态，视为激活一次；

有些场景本身不需要激活预警：例如当一个场景中主车未泊车时，不需要激活泊车碰撞预警，此时如果激活预警反而说明主车算法出错；也有一些场景需要特定次数的激活预警：例如当一个场景中主车驾驶过程中会碰到n个盲区，此时必须正好激活n次才能证明主车算法通过；因此支持让用户设置各项子指标是否需要预警和期望的预警次数；

默认期望的预警次数为-1，此时只要该预警功能激活至少一次，则评测项通过；当设置期望的预警次数为正数或0时（0代表期望预警功能不被激活），只有当预警功能激活次数和期望预警次数相同时，评测项才通过；

该指标仅对有算法pb的场景有效。当算法pb中未设置预警项，或预警项状态皆为STATE_UNKNOWN时，该指标也视为无效；

该指标关联的内置可视化时间序列数据为：暂无。

该指标的异常时间点记录类型为：POINT_TYPE_ALL。

控制辅助系统激活（Control）检测

控制辅助系统激活用于评价算法是否按照预期激活以下十三项功能:

- 自动紧急制动
- 自动紧急转向
- 倒车自动紧急制动
- 自适应巡航控制
- 车道保持辅助
- 自动驾驶辅助
- 自动泊车辅助
- 远程泊车辅助
- 拖车辅助
- 城市驾驶辅助
- 高速自动驾驶辅助
- 巡航控制
- 限速控制

其实现逻辑与预警系统激活（Warning）检测一致。

信息辅助系统激活（Information）检测

信息辅助系统激活用于评价算法是否按照预期激活以下六项功能:

- 倒车摄像头
- 环视摄像头
- 自动远光灯
- 驾驶员状态监测系统
- 抬头显示系统
- 夜视辅助系统

其实现逻辑与预警系统激活（Warning）检测、控制辅助系统激活（Control）检测一致。

7.4.3.5 舒适性评测指标

减速度（Deceleration）检测

减速度检测的目的是:

判断主车在整个行驶过程中制动减速度是否超过对应的舒适性阈值。

本设计的减速度的默认阈值为3m/s^2。

加速度变化率（Jerk）检测

加速度变化率是加速度对时间的导数。

加速度变化率也被称为冲击度，冲击度反映了驾驶员的瞬态冲击体验。

纵向、侧向冲击度的阈值按德国冲击度标准取$15m/s^3$。

平稳起步(Gentle Start)检测

汽车起步时加速度太大会给人带来不舒适的感受。平稳起步检测的目的是判断自动驾驶车辆起步过程中加速度是否过大。

起步过程的判定是指车辆当前速度为0，并在0.5s后速度大于$1m/s$，这个0.5s内的时间段为起步过程。

起步过程中如果加速度大于一定阈值（本设计取$1m/s^2$），则判定起步加速度过大，起步不平稳。

该指标关联的内置可视化时间序列数据为：speedX，accX。

该指标的异常时间点记录类型为：POINT_TYPE_POINT。

乘员舒适性（Driving Comfort）检测

乘员舒适性检测关注的是自动驾驶车辆行驶过程中，驾驶员感受到的舒适程度。

舒适程度通常可以利用整个行驶过程中的速度方差来进行客观反映，而变异系数是可以对不同速度区间舒适程度进行比较。

变异系数的公式如下所示。

$$cv = \frac{\sigma}{\mu}$$

cv表示变异系数，σ表示标准差，μ表示均值。

本设计当主车速度的变异系数大于0.15时，判定乘员舒适性检测不通过。

该指标关联的内置可视化时间序列数据为：speedX。

该指标的异常时间点记录类型为：POINT_TYPE_ALL。

平顺性（Ride Comfort）检测

平顺性检测通常指汽车的垂向平顺性。平顺性用加速度均方根值来衡量。

加速度均方根值计算公式如下所示。

$$M_{rms} = \sqrt{\frac{\sum_{i=0}^{n} x_i^2}{n}}$$

M_{rms}表示变量x的均方根值，x_i表示第i个x值，n表示x值的个数。

汽车的垂向平顺性是由悬架系统决定的，自动驾驶算法对垂向平顺性几乎没有影响，其影响的是车辆的纵向和侧向平顺性。

因此，本设计平顺性检测从纵向平顺性和侧向平顺性进行考量。

平顺性检测考虑的是整个仿真时间段的加速度均方根值。当纵向或侧向加速度均方根值大于$0.5m/s^2$，则认为对应的纵向/侧向平顺性检测不通过。

纵向平顺性关联的内置可视化时间序列数据为：accX。横向平顺性关联的内置可视化时间序列数据为：accY。

该指标的异常时间点记录类型为：POINT_TYPE_ALL。

急转向（Steering）检测

侧向加速度过大会对车辆的侧倾稳定性和乘员体验造成不良影响，急转向检测的目的是判断主车在行驶过程中，侧向加速度是否过大。

侧向加速度的阈值设置为2.3 m/s^2。

该指标关联的内置可视化时间序列数据为：accY。

该指标的异常时间点记录类型为：POINT_TYPE_POINT。

蛇行（Snake Driving）检测

自动驾驶车辆在行驶过程中，当车道的曲率发生较大变化时，可能会出现横向控制效果不佳导致的长时间车辆横向振荡。

蛇行检测的目的是判断车辆是否出现横向振荡，利用车辆的横向加速度的正负变化来判断蛇行是否发生。

正值大于$1\mathrm{m/s}^2$和负值小于$-1\mathrm{m/s}^2$的比例都大于该时间段的10%时，则判断此时间段发生蛇行。

在及少数的连续S型弯道情况下，可能会出现假阳性结果，这会在评测报告中进行体现。

该指标关联的内置可视化时间序列数据为：accY。

该指标的异常时间点记录类型为：POINT_TYPE_REGION。

急刹（Emergency Braking）检测

自动驾驶车辆急刹有两个典型阈值：ACC（Adaptive Cruise Control）的最大减速度，和AEB（Autonomous Emergency Braking）的最大减速度。

急刹检测的目的是判断主车在行驶过程中是否达到ACC和AEB的最大减速度。

- ACC的最大减速度通常为$-3\mathrm{m/s}^2$。
- AEB的最大减速度通常为$-6\mathrm{m/s}^2$。

该两项子指标关联的内置可视化时间序列数据均为：accX。

该指标的异常时间点记录类型为：POINT_TYPE_POINT。

7.4.4 评测分数计算介绍

7.4.4.1 评分方案介绍

Octopus评测指标共有30多项大类指标，当规控算法未通过某些评测指标后，评测分数应能反映算法的性能表现。

本设计根据指标的重要程度将其分为三大类：

- 主要指标（以下简称A类）。
- 次要指标（以下简称B类）。
- 未定义重要度指标（以下简称C类）。

本设计提供如下三种内置的评分方案：

- AB类log函数评分。
- AB类均匀权重评分。
- C类均匀权重评分。

接下来对三种评分方案进行详细介绍。

7.4.4.2 AB 类 log 函数评分方案

在该评分方案中，A类指标必须通过，如果该类指标有一项没有通过，则得分直接不及格（低于60分）。

在有A类指标参与评测时，B类指标即使有几项没有通过，整个得分也不会不及格。

B类指标不通过数超过一定比例时，对应的分数要小于80分。

C类指标不参与评分。

在没有A类指标参与评测时，B类指标评分权重更大，这样只要有几项B类指标不通过，则对应的评分会小于60分。

AB 类 log 函数评分原则（Principle）

1. A类，错一项就不能及格（低于60）。
2. B类，再怎么错都不会不及格（除非评分项不包括A类）。
3. 全错则是0分，全对则是100分。
4. 如果评分项是空集，则是0分。
5. C类， 不参与评分

AB 类 log 函数评测分数计算实现（Equation）

本设计的评测分数旨在反映自动驾驶的安全性，因此计算过程中的评测分值分布为：

- A类：60分
- B类：40分

具体实现公式为：

$$score(a_0,a_1,b_0,b_1) = \begin{cases} 100 - f_A(a_0,a_1,60) - f_B(b_0,b_1,40) & a_0 > 0, b_0 > 0 \\ 100 - f_A(a_0,a_1,100) & a_0 > 0, b_0 = 0 \\ 100 - f_B(b_0,b_1,100) & a_0 = 0, b_0 > 0 \\ 0 & a_0 = 0, b_0 = 0 \end{cases}$$

其中：

- a_0: A类指标参与评测的总数目。
- a_1: A类指标未通过的数目。
- b_0: B类指标参与评测的总数目。
- b_1: B类指标未通过的数目。

另外，A类的扣分函数f_A是：

$$f_A(a_0,a_1,s) = \begin{cases} (s - 40.01) \cdot \log_{a_0} a_1 + 40.01 & a_0 > 1, a_1 > 0 \\ s & a_0 = 1, a_1 = 1 \\ 0 & a_0 \geq 1, a_1 = 0 \end{cases}$$

B类的扣分函数f_B是：

$$f_B(b_0,b_1,s) = s \cdot \log_{b_0 + 1}(b_1 + 1)$$

7.4.4.3 AB 类均匀权重（Average）评分方案

该方案同样分为AB两类指标，其中A类总分为60分，B类总分为40分，A类指标按均匀权重扣分，B类指标同样按均匀权重扣分。

AB 类均匀权重评分原则（Principle）

1. A类60分，各A类指标得分权重相同。
2. B类40分，各B类指标得分权重相同。
3. 全错则是0分，全对则是100分。
4. 如果评分项是空集，则是0分。
5. C类， 不参与评分。

AB 类均匀权重评测分数计算实现（Equation）

本设计的评测分数旨在反映自动驾驶的安全性，因此计算过程中的评测分值分布为：

- A类：60分
- B类：40分

具体实现公式为：

$$score(a_0,a_1,b_0,b_1) = \begin{cases} 60 \cdot (1 - a_1/a_0) + 40 \cdot (1 - b_1/b_0) & a_0 > 0, b_0 > 0 \\ 100 \cdot (1 - a_1/a_0) & a_0 > 0, b_0 = 0 \\ 100 \cdot (1 - b_1/b_0) & a_0 = 0, b_0 > 0 \\ 0 & a_0 = 0, b_0 = 0 \end{cases}$$

其中：

- a_0: A类指标参与评测的总数目。
- a_1: A类指标未通过的数目。
- b_0: B类指标参与评测的总数目。
- b_1: B类指标未通过的数目。

7.4.4.4 C 类均匀权重评分（Average）方案

当用户选择该评分方案时，就不需要设置评测指标的重要度，各个指标按均匀权重进行扣分。

C 类均匀权重评分原则（Principle）

各指标得分权重相同。

C 类均匀权重评测分数计算实现（Equation）

此方案下总分为100分，在计算得分时不考虑指标重要度，只关注失败指标个数占总指标的比例。

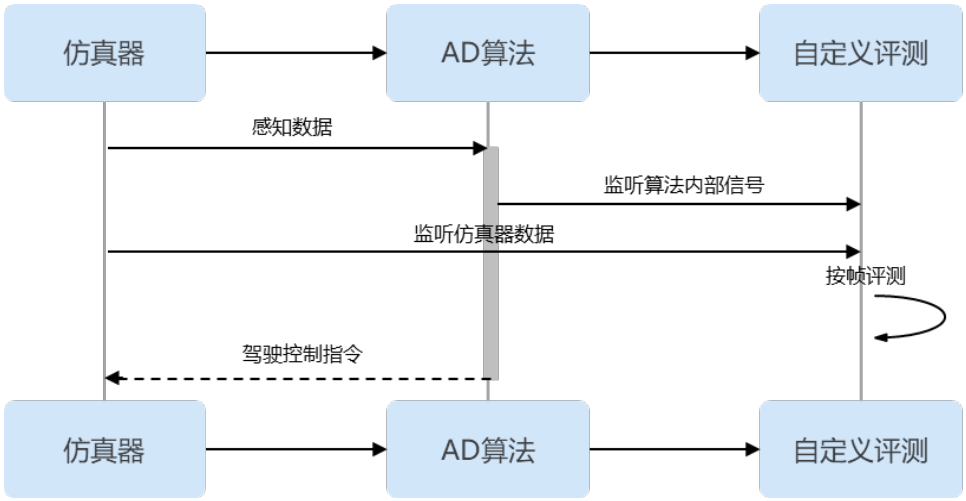
具体实现公式为：

$$score(c_0, c_1) = \begin{cases} 100 \cdot (1 - (c_1)/c_0) & c_0 > 0 \\ 0 & c_0 = 0 \end{cases}$$

7.4.5 实时评测和延时评测介绍

实时评测

图 7-7 实时评测



实时评测的基本架构如上图所示，实时评测算法从仿真器和AD算法按帧接收数据，每接收一帧数据，就调用一次评测函数，在最后仿真结束时将评测结果写成评测pb文件。

实时评测的实现包括如下几个步骤：

步骤1 代码内实现与仿真器的通信，实时接收仿真器的帧数据，也可同时接收仿真器和AD算法的数据。

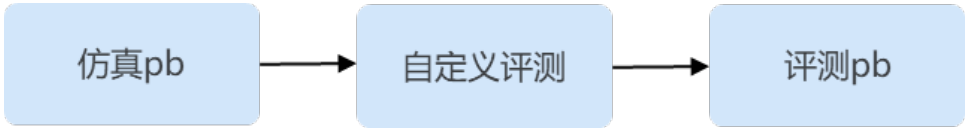
步骤2 处理每帧数据，不断更新评测结果。

步骤3 仿真结束时，将最后一帧的评测结果作为最终的评测结果，通过EVA_PATH环境变量获取评测pb路径，经评测结果写入到评测pb文件中。

----结束

延时评测

图 7-8 延时评测



如上图所示，延时评测以仿真pb文件作为输入，进行评测逻辑处理后，将评测结果写成评测pb。

其中仿真pb是通过八爪鱼提供的sim_osi.proto进行序列化和反序列化，评测pb是通过八爪鱼提供的eva.proto进行序列化和反序列化的。

延时评测算法的实现有如下几个步骤：

步骤1 在代码内通过SIM_OSI_PATH环境变量获取仿真pb路径，通过EVA_PATH环境变量获取评测pb路径。

通过文件Open的方式打开仿真pb路径，读取字节流，利用sim_osi.proto中的SimData反序列化仿真pb中的内容。该步骤会得到一个SimData的内存对象，用户通过访问对象中的字段即可获取自己关注的数据。

步骤2 SimData中包含仿真器输出的整个仿真过程数据，用户处理根据自身评测逻辑处理所有帧数据。

步骤3 用户自定义的评测指标包含通过，不通过等结果，将该结果写入到eva.proto中的Evaluation类中，然后通过文件Open的形式打开评测pb路径，将评测结果写成评测pb文件。

步骤4 写成评测pb文件后，延时评测镜像的工作就完成了，仿真平台的控制程序在运行自定义评测容器时会主动将评测pb文件上传到对象存储中，前端通过下载该评测pb文件进行解析，可以将自定义评测结果和内置评测结果一样完全兼容地进行展示。

----结束

评测算法代码开发完成后，将代码构建成算法镜像上传到仿真平台评测管理模块即可被仿真任务使用。在制作评测算法镜像的Dockerfile中，建议将评测代码编译成的二进制文件COPY到系统的/usr/bin目录下，便于在前端界面填写评测镜像的运行命令时直接填写该二进制文件的名称即可。在镜像中新建一个shell脚本来运行评测代码也是可以接受的方案。

7.5 场景管理

7.5.1 场景管理分类设计使用逻辑

场景管理提供所有仿真场景、测试用例和泛化场景的管理功能，用户可上传符合平台规范的自定义场景，也可将场景下载至本地开发。Octopus平台自研场景标签分类体系，从多维度深层次科学分类场景。仿真场景库可自建仿真场景库，集合相同场景格式的不同条件仿真场景，检验在特定条件下仿真算法控制质量。

仿真服务场景管理分为三大类型：

- 场景和场景库。
- 逻辑场景和逻辑场景库。
- 测试用例和测试套件。

场景和场景库

其中片段式场景仿真是自动驾驶系统测试的重要手段，当前业内对于片段式场景普遍遵循ASAM主导的OpenX系列标准。OpenSCENARIO对动态驾驶环境进行了描述，交通参与物之间通过其他物体的状态变化作为触发条件，进而改变自身的状态。

通过OpenX场景可对算法与环境的动态交互能力进行测试，场景库的目的则是将一批有相同测试目的的场景进行汇总，如想测试Acc算法的应对切入功能，可将多个应对切入的测试场景归到一个场景库，进而在创建仿真任务时可直接选择该场景库进行仿真。

逻辑场景和逻辑场景库

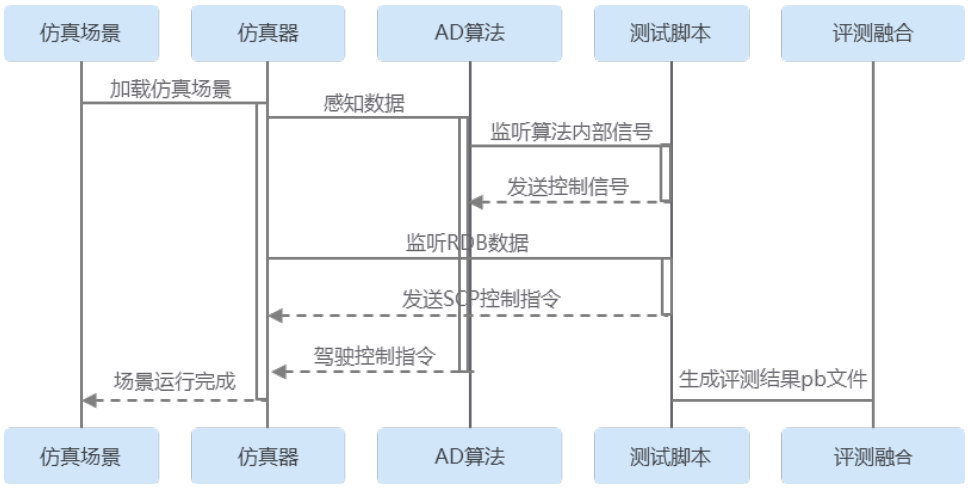
逻辑场景是基于状态空间变量对功能场景的进一步详细描述，每个逻辑场景都有场景参数，比如前车的车速及其加速度，自车与前车距离等参数，这些参数都有一定的取值范围，根据这些参数可以派生出任意数量的具体场景。

逻辑场景库是不同逻辑场景的数据集合，以树状结构的形式表现出来，便于对逻辑场景进行统一的、有效的组织、管理和应用，比如当用户想系统管理和方便查看超车的逻辑场景，可以将所有超车相关的逻辑场景加入一个场景库中。

测试用例和测试套件

如上所述，OpenSCENARIO场景能描绘动态环境的，但无法根据主车内部动力学状态、自动驾驶算法状态作为触发条件来驱动各个交通参与物变化，因此为了实现更加精细的测试控制，需要额外提供一个测试脚本实现与仿真器中的交通参与物和算法内部数据的交互。

图 7-9 测试用例和测试套件



如上图所述，测试脚本能同时仿真器数据运行时RDB数据和AD算法的内部数据，如通过RDB判断主车与前车距离小于20m，可发送某个控制信号给算法改变esp状态，也可以通过SCP指令控制场景中的副车改变运动姿态。

测试脚本很大程度上弥补了单纯场景仿真的不足，能够实现以算法内部信号为触发条件，改变仿真场景中交通参与物的状体和算法内部状态。

因此，在场景的基础上，添加与该场景相匹配的测试脚本，就形成了一个测试用例。也就是说，测试用例是一个场景和测试脚本的集合。

同样，测试套件是将测试目的相同的测试用例归到一起，方便创建仿真任务时直接选择。

7.5.2 场景库管理

仿真场景库模块提供场景库管理功能，支持对场景库的增删改查功能。自建场景库将场景格式相同的多个仿真场景集合，便于用户进行开发。

创建场景库

步骤1 在左侧菜单栏中单击“仿真服务 > 场景管理”。

步骤2 选择“场景库”页签，单击“场景库分类”后的“新增一级分类”，输入场景库分类名称，用户将新建一个场景库分类。

步骤3 单击场景库文件夹后的，用户将新建一个场景库。

- 场景库分类：当前场景库分类名称。
- 场景库名称：只能包含数字、英文、中文、下划线、中划线，不得超过64个字符。
- 仿真器：支持仿真器B。
- 描述（非必填）：不能包含“@#$%^&*< > \”，不得超过255个字符。

步骤4 添加场景。

单击新建的场景库名称，在场景列表单击“添加场景”。

- 可勾选场景前勾选框选择场景。

- 可筛选历史场景库中已有的场景到当前的场景库中。
- 如果场景数量较多，可搜索场景或基于场景标签筛选场景后选择。

步骤5 单击“确定”，提示“添加场景成功”，在场景库列表页面展示添加的场景。

----结束

场景库相关操作

在“场景库”页签，可对场景库进行以下操作。

表 7-4 场景库相关操作

任务	操作步骤
查看场景库信息	单击左侧场景库名称，查看右侧该场景库信息以及场景库包含场景信息。 • 场景库信息：场景库名称、创建人、仿真器、描述、关联仿真任务数和创建时间等信息。 • 场景列表：该场景库中包含的所有场景。场景的具体操作请参考**场景管理**。
修改场景库信息/场景库分类信息	单击场景库名称或场景库分类后的，修改场景库或场景库分类的信息。
删除场景库/场景库分类	单击场景库名称或场景库分类后的，删除指定场景库或场景库分类。
查询场景	根据“场景名称”或“创建人”，输入搜索条件，查询场景列表中的场景。

7.5.3 场景管理

仿真场景模块支持对单个仿真场景的增删改查操作。用户可根据场景类型，依据平台提示，上传符合要求的场景文件。场景创建完毕后，用户可选择在线仿真机器加载场景，通过仿真器内置算法检验场景质量。

创建场景

仿真场景支持用户上传符合仿真器场景规范的自定义场景。添加场景的步骤可参考如下：

步骤1 在左侧菜单栏中选择“仿真服务 > 场景管理”。

步骤2 选择“场景”页签，单击“创建场景”，填写基本信息。

- 场景名称：只能包含数字、英文、中文、下划线、中划线、点，且不支持以点结尾，不得超过256个字符。
- 仿真器：支持仿真器B。
- 优先级：当前支持S、A、B、C、D。级别顺序为：S > A > B > C > D，默认级别是D。

- 场景描述：简要描述场景，不包含“@#$%^&*< > \”，不超过255个字符。

步骤3 上传文件。

仿真器B为动态场景描绘文件。上传场景文件需包含以下类型。

- 场景文件版本：选择对应的场景文件版本。
- 动态场景文件：动态场景描绘文件（文件格式为“.xosc”）。
- 地图文件：开放格式和实际上的公路网络在驾驶模拟应用程序中的描述标准（文件格式为“.xodr”）。
- 3d模型文件：文件格式为“.osgb”。

步骤4 添加场景标签，标签数量不超过50个。

- 直接选择：单击“添加标签”，从场景标签中直接选择标签，也可新建标签。
  - 多项搜索：可根据需要决定是否启用多项搜索，输入多个关键字，中间用“;”隔开，可搜索多个关键字。
  - 推荐标签：平台可解析场景文件中的内容推荐标签，目前场景格式支持OpenSCENARIO系列（推荐标签开关显示的前提是文件上传）。

    自动推荐标签的依据为：OpenSCENARIO元素到OpenLABEL标签体系元素的映射，主要为OpenSCENARIO的Environment和Entity字段到OpenLABEL的Weather和RoadUser标签树的映射。

    推荐标签有以下约束：

    - 不支持使用参数引用ParameterDeclaration的OpenSCENARIO场景文件。
    - 不支持根据场景文件中引用的外置文件Catalogs的内容进行标签推荐。
    - 推荐标签的范围为OpenLABEL体系的VehicleBus，VehicleCar，VehicleCycle，VehicleMotorcycle，VehicleTrailer，VehicleTruck，VehicleVan，HumanWheelchairUser，RoadUserAnimal，HumanPedestrian，WeatherWind，WeatherRain，WeatherSnow，MotionStop，Traffic。
- Json文件导入：单击“选择json文件”，可选择本地的json文件，直接导入标签。

步骤5 勾选“我已阅读并同意《八爪鱼自动驾驶云服务使用声明》”。

步骤6 单击“创建”。

----结束

场景相关操作

在“场景”页签，可对场景进行以下操作。

表 7-5 场景相关操作

任务	操作步骤
查看场景信息	单击场景名称，查看该场景信息以及场景库包含场景信息。 ● 场景详情：场景名称、场景格式等信息。 ● 文件名称：该场景包含的所有场景文件信息。必须包含开放格式和实际上的公路网络在驾驶模拟应用程序中的描述标准（文件格式为“.xodr”）。可单击操作栏中的“下载”或“替换文件”，将文件下载本地或替换场景文件。 ● 场景预览：根据场景文件的不同情况，场景预览会以不同的方式呈现。详见**场景预览**。 ● 导出标签：单击场景标签下的“导出json文件”，即可导出标签文件至本地。
查询场景	选择“场景名称”、“场景ID”或“创建人”，在搜索输入框中输入搜索条件，按回车键即可查询。也可按照 " 标签筛选 " 查询场景具体可参考**标签筛选**。
修改场景信息	单击场景名称后操作栏内的“编辑”，修改场景信息。
删除场景	● 单击场景名称后操作栏内的“更多 > 删除”，删除该场景。 ● 勾选多个场景前勾选框，单击场景列表上方的“删除”，可批量删除场景。 **说明** ● 删除后该场景将不再应用于仿真场景库中，请谨慎操作。 ● 绑定仿真任务的仿真场景不允许删除，否则会导致仿真任务运行失败。删除场景需先删除与之相关的仿真任务。
历史结果对比	1. 单击场景名称后操作栏内的“更多 > 历史结果对比”。 2. 选择需要对比的任务名称，单击“确认”。 3. 页面跳转至“任务对比”页面，即可查看任务对比结果。
已关联用例	单击操作栏中的“更多 > 已关联用例”，可查看场景的已关联的用例。

场景预览

场景预览当前有两种呈现方式：动态场景预览和地图场景预览。

- 动态场景预览：场景文件中存在完整的地图文件和动态场景文件，且动态场景文件为.xosc格式时显示。
- 地图场景预览：场景文件缺失或部分缺失，动态场景文件为.xml格式，场景文件解析失败或其他不支持动态场景预览的情况时显示。

动态场景预览

图 7-10 动态场景预览页面

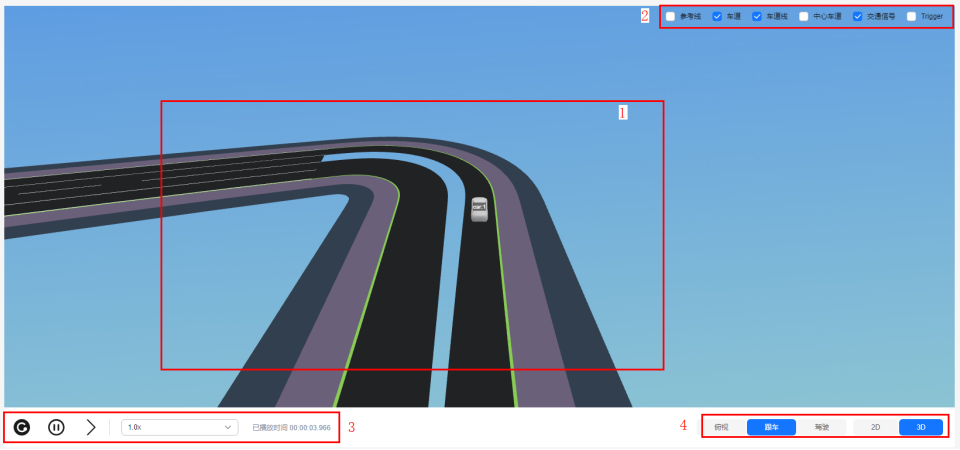


表 7-6 动态场景预览页面详细说明

序号	区域名称	说明
1	动态场景预览区域	车辆的行驶轨迹，随着主车的行驶感知在主车辆周围出现的其他物体，如其他车辆、行人和交通信号等。目前可感知的物体类型请见**感知物体类型**。鼠标移动至道路时，道路会变红。遇到信号灯，车辆会按照红绿灯指示行驶。
2	交通参与物状态	可根据需要选择显示参与物状态。当前支持的参与物有参考线、车道、车道线、中心车道、交通信号、Trigger。 默认勾选Trigger，当出现  时，单击图标，地图中会出现trigger的详细信息。 交通信号包含信号灯和交通标识牌。
3	视频播放控件	控制视频播放暂停回放按钮，支持逐帧和倍速播放。地图文件超过1MB不支持高倍速预览。
4	场景切换	视角切换：跟车、俯视、驾驶，自由。当在3D回放页面拖动鼠标时，即可切换为自由视角。 2D/3D切换：可单击“2D/3D”，可切换2D，3D场景。

地图场景预览

图 7-11 地图场景预览页面

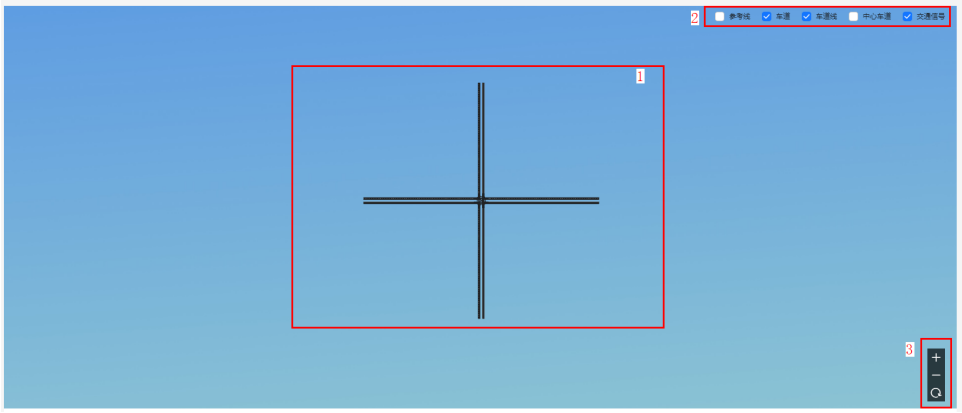


表 7-7 地图场景预览页面详细说明

序号	区域名称	说明
1	地图场景预览区域	鼠标上下滚动，可以放大缩小图片，鼠标左键可旋转图片，右键可拖动图片。
2	交通参与物状态	可根据需要选择显示参与物状态。当前支持的参与物有参考线、车道、车道线、中心车道、信号灯。
3	场景刷新	单击右下方加号、减号和刷新按钮，放大、缩小、还原地图。

7.5.4 逻辑场景库管理

逻辑场景库模块提供逻辑场景库管理功能，支持对逻辑场景库的增删改查功能。

创建逻辑场景库

步骤1 在左侧菜单栏中单击“仿真服务 > 场景管理”。

步骤2 选择“逻辑场景库”页签，单击“逻辑场景库分类”后的“新增一级分类”，输入逻辑场景库分类名称，用户将新建一个逻辑场景库分类，目前逻辑场景库分类层级支持四层。

步骤3 单击逻辑场景库分类后的，用户将新建一个逻辑场景库。

- 逻辑场景库分类：当前场景组分类名称。
- 逻辑场景库名称：只能包含数字、英文、中文、下划线、中划线，不得超过64个字符。
- 逻辑场景库描述（非必填）：不能包含“@#$%^&*< > \”，不得超过255个字符。

步骤4 添加逻辑场景

单击新建的逻辑场景库名称，在右侧场景列表单击“添加场景”。

- 可勾选逻辑场景前勾选框选择逻辑场景。
- 可筛选历史逻辑场景库中已有的逻辑场景到当前的逻辑场景库中。
- 如果场景数量较多，可搜索逻辑场景或基于标签筛选场景后选择。

步骤5 单击“确定”，提示“添加逻辑场景成功”，在场景列表页面展示添加的场景。

----结束

逻辑场景库相关操作

在“逻辑场景库”页签，还可进行以下操作。

表 7-8 逻辑场景库相关操作

任务	操作步骤
查看逻辑场景库信息	单击左侧逻辑场景库名称，查看右侧该场景库信息以及场景库包含场景信息。 ● 场景库信息：场景库名称、创建人、描述和创建时间等信息。 ● 场景列表：该逻辑场景库中包含的所有场景。逻辑场景的具体操作请参考**逻辑场景管理**。
修改逻辑场景库信息/逻辑场景库分类信息	单击场景库名称或场景库分类后的，修改逻辑场景库或场景库分类的信息。
删除逻辑场景库/逻辑场景库分类	单击场景库名称或场景库分类后的，删除指定逻辑场景库或场景库分类。
查询逻辑场景	根据“逻辑场景名称”或“创建人”，输入搜索条件，查询逻辑场景列表中的场景。

7.5.5 逻辑场景管理

逻辑场景功能可批量生成仿真场景，扩充仿真场景库，为自动驾驶开发提供海量仿真场景。自定义逻辑场景的可以进行修改、删除操作，并支持仿真器B。

逻辑场景中场景是完全独立存储与使用的，在逻辑场景生成的泛化场景非常多，其中有价值的场景比例低，如果存储到场景中会对场景模块造成冲击，当前如果认为逻辑场景中保存的个别场景价值高，可以采用下载场景文件到本地再上传到场景模块的方式保留该逻辑场景。

7.5.5.1 逻辑场景

创建逻辑场景

步骤1 在左侧菜单栏中选择“仿真服务 > 场景管理”。

步骤2 选择“逻辑场景”页签，单击“新增逻辑场景”。

- 逻辑场景名称：只能包含数字、英文、中文、下划线、中划线、点，且不支持以点结尾，不得超过256个字符。
- 描述（非必填）：简单描述场景。
- 添加场景标签，标签数量不超过50个。
  a. 直接选择：单击“添加标签”，从场景标签中直接选择标签，也可新建标签。
     多项搜索：可根据需要决定是否启用多项搜索，输入多个关键字，中间用“;”隔开，可搜索多个关键字。
  b. Json文件导入：单击“选择json文件”，可选择本地的json文件，直接导入标签。

步骤3 选择文件上传。

1. **文件版本为：OpenSCENARIO 2.0.0时：**
   - 场景文件版本：选择OpenSCENARIO 2.0.0。
   - 动态场景文件：仿真场景动态描绘文件（文件格式为“.osc”）。
   - 地图文件：开放格式和实际上的公路网络在驾驶模拟应用程序中的描述标准或者使用DSL语言描述的逻辑地图（文件格式为“.xodr”，“.odr”）。

     说明

     - 添加地图文件为“.xodr”格式后，勾选“我已阅读并同意《八爪鱼自动驾驶云服务使用声明》”。
     - 详情界面中，“xodr”实际的地图文件可以在地图中查看，“odr”逻辑的地图文件在详情中不会展示地图文件。
2. **文件版本为：OpenSCENARIO 1.1.1时：**

   图 7-12 OpenSCENARIO 1.1.1

   - 场景文件版本：选择OpenSCENARIO 1.1.1。
   - 动态场景文件：OpenSCENARIO1.1具体场景文件（文件格式为“.xosc”）。
   - 地图文件：开放格式和实际上的公路网络在驾驶模拟应用程序中的描述标准（文件格式为“.xodr”）。
   - 场景参数文件：逻辑场景泛化参数的分布文件（文件格式为“.xosc”）。

说明

- 添加地图文件后，勾选“我已阅读并同意《八爪鱼自动驾驶云服务使用声明》”。
- 详情界面中，“xodr”实际的地图文件可以在地图中查看。

步骤4 单击“创建”，逻辑场景创建成功。在逻辑场景列表可查看逻辑场景的基本信息。

----结束

逻辑场景相关操作

在“逻辑场景列表”，还可以进行以下操作。

表 7-9 逻辑场景相关操作

任务	操作步骤
查询逻辑场景	选择“逻辑场景名称”、“场景ID”或“创建人”，在搜索输入框中输入搜索条件，按回车键即可查询。也可按照 " 标签筛选 " 查询场景具体可参考**标签筛选**。
删除逻辑场景	• 单击逻辑场景名称后操作栏内的“ 删除”，删除该场景。 • 勾选多个逻辑场景前勾选框，单击场景列表上方的“删除”，可批量删除场景。 **说明** 被任务使用的逻辑场景不可被删除。
编辑逻辑场景	单击操作栏中的“编辑”，可编辑逻辑场景基本信息。
查看逻辑场景详情	单击逻辑场景名称，可查看逻辑场景详情。 • 基本信息：场景名称、创建时间，解析状态等信息。 • 场景参数：包括动态场景、静态场景信息，可单击文件列表后的“下载”或“替换文件”，将文件下载本地或替换场景文件。 • 泛化任务：平台支持逻辑场景生成泛化任务，具体操作参考**泛化任务**。 • 场景预览：根据场景文件的不同情况，场景预览会以不同的方式呈现。详见**场景预览**。

场景预览

场景预览当前有两种呈现方式：动态场景预览和地图场景预览。

- 动态场景预览：版本为OpenSCENARIO1.1.1的场景预览，存在完整的逻辑场景文件时显示。
- 地图场景预览：逻辑场景文件缺失或部分缺失，逻辑场景文件解析失败或其他不支持动态场景预览的情况时显示。

动态场景预览

逻辑场景的动态场景预览同具体场景的动态场景预览相同，可参考具体场景的动态**场景预览**。

地图场景预览

逻辑场景的地图场景预览同具体场景的地图场景预览相同，可参考具体场景的地图**场景预览**。

7.5.5.2 泛化场景

创建泛化任务

说明

1. 当逻辑场景状态为“解析失败”和“正在解析”时，无法创建泛化任务。
2. 上传动态和静态场景文件的说明请查看**Open SCENARIO2.0场景说明**。
3. 当逻辑场景文件版本为OpenSCENARIO2.0.0时，创建的泛化任务支持仿真器B。

步骤1 在左侧菜单栏中单击“仿真服务 > 场景管理”。

步骤2 选择“逻辑场景”页签，单击逻辑场景名称，进入逻辑场景详情页。

步骤3 选择“泛化任务”页签，单击“创建场景泛化”，参考下表填写泛化任务基本信息。

表 7-10 场景泛化参数

参数	说明
泛化任务名称	包含中英文、数字、“_”“-”，不得超过256个字符。
仿真器	选择在线仿真器，当前支持仿真器B。
优先级	当前支持S、A、B、C。级别顺序为：S > A > B > C，默认级别是C。
老化时间	选择场景有效期，场景将在选择日期的当日24时老化。
描述	简单描述泛化场景。

步骤4 参考下表配置泛化参数。

图 7-13 泛化参数

表 7-11 参数说明

参数	说明
泛化类型	场景泛化的类型，即生成的场景类型。
是否开启参数设置	如开启，根据选择泛化类型，展示可以设置的场景泛化参数。自建逻辑场景默认开启参数设置。
采样方式	可选择“均匀采样”“蒙特卡洛采样”“拉丁超立方采样”“联合概率分布采样”“重要性采样”。具体可参考“**7.9 采样方式介绍**”。
场景文件版本	泛化任务支持的场景文件版本。
场景泛化数量	泛化后生成的场景数量（0，10000］。
参数设置	分为动态和静态两种泛化参数。 相关系数：仅针对选择“联合概率分布采样”时显示，可新增相关参数[-1，1]，支持两位小数，变量不可重复选择。
选择仿真任务	选择关联到该逻辑场景下泛化任务的仿真任务，仅针对选择“重要性采样”时显示。
敏感性分析	根据需要可选择敏感性分析，具体请参考**敏感性分析**。

步骤5 单击“创建”，泛化任务创建成功，可在泛化任务列表页查看新建泛化任务。

步骤6 删除泛化任务。

单击操作栏中的“删除”，可删除泛化任务。

步骤7 关联仿真任务对比。

单击泛化任务名称后操作栏内的“关联仿真任务对比”。选择需要对比的任务名称，单击“确认”。页面跳转至“任务对比”页面，即可查看任务对比结果。

步骤8 查看泛化任务详情。

泛化任务处于“已完成”状态后，可单击任务名称，查看泛化任务详情。

- 基本信息：泛化任务名称、泛化数量、所属逻辑场景、仿真器、场景文件版本、采样方式等信息。
- 参数详情：泛化任务的参数，场景ID、具体参数、场景标签等信息。

  说明

  被批量仿真任务使用的泛化场景不允许老化。
- 导入具体场景：可选择泛化场景导入至场景列表中。

步骤9 查看泛化场景详情。

单击场景ID，可查看该场景详细信息。

步骤10 查询泛化任务。

在场景泛化列表中，搜索框内输入搜索内容进行查找。

步骤11 删除泛化任务。

单击任务名称后“操作”栏内的“删除”，即可删除指定泛化任务，删除时同步删除对应的场景。用户也可勾选泛化任务名称前的勾选框，单击“删除”，对泛化任务批量删除。

说明

- 删除后不可恢复，请谨慎操作。
- 被批量仿真使用的泛化场景不允许删除。

----结束

敏感性分析

Octopus平台支持基于参数组合、回归训练、敏捷性评定三个参数空间分析得到的敏感性分析结果，主要对逻辑场景的参数空间进行敏感性分析。在泛化任务完成的批量仿真任务后加上敏感性分析，然后把敏感性分析结果展现在新的泛化任务创建界面上，来帮助客户更好调节参数创建泛化任务。

步骤1 进入到创建泛化任务界面，参数设置，选择参数设置后的“灵敏度分析”。

步骤2 选择关联到该逻辑场景下泛化任务的仿真任务。

步骤3 单击“确定”，界面出现上次任务的敏感性分析结果，展示每个参数的灵敏度参数，该值的范围在[0,1]，该值越大，说明该参数的变化对评测分数的影响越大。

步骤4 用户可根据敏感性分析结果，设置当前各项参数的数值。

----结束

泛化任务队列管理

用户可以对泛化任务进行优先级排序。

步骤1 在左侧菜单栏中单击“仿真服务 > 场景管理”。

步骤2 在界面的右上角，单击“泛化任务管理”，可对需要管理的任务进行上移，置顶等操作。

----结束

7.5.6 测试套件管理

测试套件模块提供测试套件管理功能，支持对测试套件的增删改查功能。自建测试套件将场景格式相同的测试用例，便于用户进行开发。

创建测试套件

步骤1 在左侧菜单栏中单击“仿真服务 > 场景管理”。

步骤2 选择“测试套件”页签，单击“测试套件列表”后的“新增套件”。

- 测试套件名称：包含中英文、数字、“_”“-”，不得超过64个字符。
- 仿真器：支持仿真器B。
- 描述（非必填）：不能包含“@#$%^&*< > \”，不得超过255个字符。

步骤3 单击“确定”，套件创建成功。

步骤4 单击测试套件文件夹名称，右侧显示套件的基本信息以及套件内的用例列表。

----结束

测试套件相关操作

在“测试套件列表”，还可以进行以下操作。

表 7-12 测试套件相关操作

任务	操作步骤
复制测试套件	单击测试套件文件夹后的，输入测试套件名称，可复制测试套件。
删除测试套件	单击测试套件文件夹后的，可删除测试套件。
编辑测试套件	单击测试套件文件夹后的，可编辑测试套件基本信息。
查询测试套件	在搜索框中输入搜索内容对测试套件进行查询。

套件用例管理

Octopus提供测试套件的管理功能，将多个用例放入到一个测试套件中，后续可以直接基于测试套件创建仿真任务。

步骤1 单击测试套件名称，在用例列表单击“添加用例”。

- 可勾选测试用例名称前勾选框选择用例。

- 如果用例数量较多，可根据“标签筛选”搜索用例后选择。

步骤2 单击“确定”，提示“添加用例成功”，在用例列表页面展示添加的用例。

步骤3 查看用例信息。

单击用例名称，查看该用例详细信息。

----结束

套件用例相关操作

在“用例列表”，还可以进行以下操作。

表 7-13 套件用例相关操作

任务	操作步骤
在线查看用例	单击用例列表后的操作栏中的“查看场景”，可选择在线仿真器，在线查看用例。
复用用例	单击操作栏中“另存为”，输入名称，即可生成另一个名称的用例。
删除用例	单击操作栏中“从套件中删除”，删除用例，删除后无法恢复，请谨慎操作。
查询用例	在页面右侧，可按照“用例名称”、“用例ID”查询用例，也可在搜索框中输入搜索内容对用例进行查询。

7.5.7 测试用例管理

测试用例模块支持对单个用例的增删改查操作。用户可根据场景类型，本地选择对应的场景文件。用例创建完毕后，用户可选择在线仿真机器加载测试用例，通过仿真器内置算法检验测试用例质量。

7.5.7.1 创建用例

测试用例支持用户上传符合仿真器用例规范的自定义用例。添加用例的步骤可参考如下：

步骤1 在左侧菜单栏中选择“仿真服务 > 场景管理”。

步骤2 选择“测试用例”页签，单击“创建用例”，填写基本信息。

- 测试用例名称：只能包含数字、英文、中文、下划线、中划线、点，且不支持以点结尾，不得超过256个字符。
- 用例标签：添加场景标签，标签数量不超过50个。
  - 直接选择：单击“添加标签”，从场景标签中直接选择标签，也可新建标签。

    多项搜索：可根据需要决定是否启用多项搜索，输入多个关键字，中间用“;”隔开，可搜索多个关键字。
  - Json文件导入：单击“选择json文件”，可选择本地的json文件，直接导入标签。

- 优先级：当前支持S、A、B、C、D。级别顺序为：S > A > B > C > D。
- 仿真器：支持仿真器B。
- 测试套件：可直接选择套件。
- 测试用例描述：简要描述用例，不包含“@#$%^&*< > \”，不超过255个字符。

步骤3 选择场景。

选择场景，也可通过场景标签筛选选择场景。

步骤4 单击“创建”，页面提示创建成功，并在测试用例管理页面看到新建的测试用例信息。

----结束

测试用例相关操作

在“测试用例”页签，可对用例进行以下操作。

表 7-14 测试用例相关操作

任务	操作步骤
查询用例	可按照 " 用例ID " 、 " 用例名称 " 查询测试用例，也可在搜索框中输入搜索内容对用例进行查询。
删除用例	对于自定义用例，用户可单击用例名称后“操作”栏下的“删除”，删除该场景测试用例。 **说明** • 删除后该用例将不再应用于测试套件中，请谨慎操作。 • 用例状态为“已关闭”和“已发布”的测试用例不允许删除。
批量操作	对于自定义用例，可勾选多个测试用例前勾选框，单击“批量操作”下拉框，批量发布，批量删除，批量关闭多个测试用例。 **说明** • 用例状态为“已关闭”和“已发布”的测试用例不允许删除和发布。 • 用例状态为“已关闭”和“编辑中”的测试用例不允许关闭。
用例再利用	单击用例名称后“操作”栏内的“另存为”，输入新的测试用例名称，可以测试用例进行复制。
历史结果对比	1. 单击用例名称后操作栏内的“更多 > 历史结果对比”。 2. 选择需要对比的任务名称，单击“确认”。 3. 页面跳转至“任务对比”页面，即可查看任务对比结果。

7.5.7.2 用例管理

在测试用例的详情页，可以进行以下操作：

步骤1 查看测试用例详情。

单击用例名称查看该用例详细信息。

步骤2 编辑测试脚本。

可根据需要添加测试脚本，在测试用例详情页，单击测试脚本后面的“创建镜像”。

- 镜像选择：下拉选择镜像。
- 镜像版本：下拉选择镜像版本。
- 运行命令：输入镜像的运行命令，具体命令根据镜像启动脚本确定。示例命令如：bash start.sh，python main.py等。

  说明

  运行命令需满足以下条件：
  - 不能为空。
  - 必须是满足ASCII码的字符串。
  - 不能包含特殊字符\@#$%^&*<>。
  - 不能超过255个字符。

步骤3 用例发布。

当用例状态为“编辑中”时，用例详情页面，单击用例名称后“发布”，可对用例进行发布。

步骤4 关闭用例。

当用例状态为“已发布”时，用例详情页面，单击用例名称后“关闭用例”，可对用例进行关闭。

步骤5 修改用例。

单击用例名称，进入用例详情页，单击右上角“编辑”，即可修改用例信息。

说明

- 用例状态为 " 已关闭 " 和 " 已发布 " 时，不支持编辑用例基本信息。
- 用例编辑中/发布/关闭的使用逻辑：

  测试用例在被创建时的默认状态是编辑中，当测试用例中的场景和测试脚本以及其他相关信息都确认达到测试条件时，可以选择发布测试用例，发布后的测试用例才能被用于创建仿真任务，并且用例一旦发布，就不允许删除，从而保证测试结果的可追溯。当发布后的测试用例后续不会再进行测试时，可以选择关闭测试用例，关闭后的测试用例不能用于创建仿真任务，并且同样关闭后的测试用例不能被删除，从而保证结果的可追溯性。

----结束

7.5.8 标签管理

7.5.8.1 新增标签

Octopus平台提供ODD筛选对场景分类，同时还支持用户根据业务所需创建自定义标签，从其他维度分类场景，更高效管理场景库。在创建仿真评测任务时，基于分类标签体系，可快速筛选出符合评测任务需求的场景。

ODD筛选：每个自动驾驶系统运行的前提条件和适用范围可能不同，因此在进行场景测试时，对应的场景是属于自动驾驶系统设计运行域内至关重要。ODD（operational design domain，设计运行域）筛选提供按设计运行条件进行场景筛选的功能，ODD筛选类别的建立参考NHTSA（美国国家公路交通安全管理局）发布的《A Framework

for Automated Driving System Testable Cases and Scenarios》文献和德国Pegasus项目的场景分类体系，从道路、交通条件、环境条件等层次描述场景。

用户可根据业务所需扩展标签体系，创建自定义标签，自定义标签支持新建、编辑、删除等管理功能，支持树状分层结构，最大支持10层。

导入标签

步骤1 在左侧菜单栏中单击“仿真服务 > 场景管理”。

步骤2 单击页面右上角的“标签管理”，单击“ 导入标签”。

步骤3 单击 “添加文件”，单击“确定”，将编写好的.ttl格式的文件（标签树）上传。上传成功后刷新页面，可以在页面上看到新导入的标签树。

说明

- .ttl格式文件可参考导出已有的标签树文件。
- .ttl文件规格不大于200kb，标签数量不大于500个。
- 根节点的名称必须是唯一的，标签树中同一父节点下不能有同名标签，且根节点数量不超过10个。

----结束

标签管理相关操作

在“标签管理”界面，还可以进行以下操作。

表 7-15 标签管理相关操作

任务	操作步骤
新建根标签	单击“新增标签”，输入标签名称。
新建标签	单击标签后的  ，输入标签名称。
修改标签	单击标签后的  ，可对标签的名称进行修改。
删除标签	单击标签后的  ，可删除标签或其子节点标签。 **说明** 平台预置场景标签暂不支持编辑、删除。
导出标签	单击根标签后的  ，可导出标签至本地。
搜索标签	在搜索框输入搜索内容，可模糊搜索标签。 多项搜索：可根据需要决定是否启用多项搜索，输入多个关键字，中间用“;”隔开，可搜索多个关键字。

7.5.8.2 标签筛选

在场景、逻辑场景和测试用例，以及添加场景、逻辑场景和测试用例时，可以通过标签筛选，快速筛选出场景或用例。

标签筛选

步骤1 在左侧菜单栏中单击“仿真服务 > 场景管理”。

步骤2 在页面中单击“标签筛选”。

步骤3 在筛选框中输入内容，查找目标标签。也可在左侧标签树列表中选择目标标签。

步骤4 勾选左侧目标标签后，右侧界面会显示左侧勾选的目标标签。

多项搜索：可根据需要决定是否启用多项搜索，输入多个关键字，中间用“;”隔开，可搜索多个关键字。

步骤5 单击“确认”，界面展示含有目标标签的场景。

----结束

标签筛选相关操作

在“标签筛选”界面，还可以进行以下操作。

表 7-16 标签筛选相关操作

任务	操作步骤
新建标签	单击标签后的⊕或搜索框后的“新增标签”，输入标签名称。
修改标签	单击标签后的✎，可对标签的名称进行修改。
删除标签	单击标签后的🗑，可对标签进行删除。 **说明** 平台预置场景标签暂不支持编辑、删除。

7.6 并行仿真

Octopus平台的并行仿真模块分为任务配置和仿真任务两部分。用户在任务配置模块，可使用自研仿真算法，根据Octopus自研仿真评测体系，从行车安全、驾驶行为、乘员舒适性等多维度测评在多种条件下的仿真场景中控制算法控制质量。在仿真任务模块，可将仿真任务运行中关键指标变化绘制成图表，直观形象。

说明

基础版服务不包含并行仿真，在使用并行仿真之前需要提前购买扩展资源包。

7.6.1 任务配置

任务配置主要由仿真算法、评测项配置和场景三部分组成。仿真算法提供自动驾驶控制算法。评测配置提供评测依据。场景作为自动驾驶模拟场景，测试仿真算法的控制效果。

创建任务配置

创建仿真任务配置时需要完成基本配置、算法配置，评测项配置和选择场景几部分的内容。创建任务配置的步骤可参考如下：

步骤1 在左侧菜单栏中单击“仿真服务 > 并行仿真”。

步骤2 选择“任务配置”页签，单击“新建任务配置”，参考下表配置仿真任务基本信息。

表 7-17 仿真任务配置基础信息说明

参数	说明
任务配置名称	包含中英文、数字、“_”“-”，不得超过64个字符。
任务配置描述	简要描述任务内容，不得超过255个字符。
最大运行时长	仿真任务中一个仿真场景的运行时间或一个场景组中每个场景的运行时间，单位为秒。运行时长选择范围[60，600]。
重复次数	同一个场景在一个任务中多次运行指定次数。重复次数选择范围[1，1000000]。
录制策略	回放场景直观查看主车在仿真场景中的运行情况。目前支持的录制策略：不录制、录制所有场景。 录制所有场景，录制的是一个回放文件，保存时间为永久。删除任务时，同步删除回放文件。
优先级	当前支持S、A、B、C。级别顺序为：S > A > B > C。

步骤3 仿真器配置。

表 7-18 仿真器配置信息说明

参数	说明
仿真器	选择在线仿真器，当前支持仿真器B、仿真器C（支持的仿真器列表取自基础包选择的仿真器类型和扩展包选择的仿真器类型的并集）。
仿真器来源	可选择“内置仿真器”和“仿真器镜像”。
仿真器配置文件	单击“编辑”，展开仿真器配置文件编辑窗，支持自定义仿真器配置文件。
仿真器镜像	选择仿真器镜像时，需要选择仿真器镜像配置。
仿真器版本	选择仿真器镜像时，选择仿真器镜像后，需要配置仿真器版本。

步骤4 算法配置。

仿真任务支持用户使用内置算法，用户也可选择自研控制算法。

- 算法配置：如果使用自定义算法，请选择在算法管理中创建成功的仿真算法。自定义仿真算法创建请参考**算法创建**。如果不选择，则默认使用仿真器内置驾驶员模型。
- 使用Datahub：勾选该项后，仿真任务在3d回放时可展示预测、规控、定位等算法内部信息，用户还能根据算法内部数据的pb文件实现算法的白盒化评测。

  说明

  Datahub模式，目前暂不支持配合仿真器的帧同步模式。

步骤5 评测项配置。

用户可选择“内置评测配置”和“自定义评测镜像”，设置评测标准。

- 内置评测配置：请选择“评测管理”服务中创建成功的内置评测配置。
- 自定义评测镜像：请选择在“评测管理”服务中创建成功的自定义评测镜像。创建评测镜像请参考**创建评测镜像**。
- 测试结果通过标准。可自定义测试结果通过标准的分数。区间为0-100分。

步骤6 选择场景。

支持场景库、测试套件和逻辑场景。选择相应的场景库或测试套件，如果数量过多，可在搜索框内输入搜索内容，进行筛选。

步骤7 单击“创建”。

----结束

任务配置相关操作

在“任务配置”列表，可对仿真任务配置进行以下操作。

表 7-19 任务配置相关操作

任务	操作步骤
查看算法详情	单击该任务配置所使用的算法名称，即可查看算法详情。无算法接入时页面显示“--”，使用仿真器自带的驾驶员模型控制主车。
查看评测详情	单击该任务配置所使用的评测名称，即可查看评测详情。
查看任务配置详情	单击任务配置名称，可查看任务配置详情。
查询任务配置	可按照＂任务配置ID＂、＂任务配置名称＂、＂算法名称＂、＂评测名称＂、＂任务配置描述＂查询任务配置。
删除任务配置	单击操作栏“ 删除”，删除该任务配置。删除后不可恢复，请谨慎操作。

任务	操作步骤
编辑任务配置	单击操作栏“编辑”，以修改任务配置名称和描述信息。

7.6.2 仿真任务

仿真任务列表提供对仿真任务的管理功能，支持停止任务、删除任务、修改任务等操作。

7.6.2.1 仿真任务创建

创建仿真任务

步骤1 在左侧菜单栏中，选择“仿真服务 > 并行仿真”。

步骤2 在“任务配置”页签，单击仿真任务配置名称，进入仿真任务配置详情页。

步骤3 在“仿真任务列表”，单击“运行”，根据界面提示填写相关信息。

步骤4 单击“确认”，仿真任务创建完成。

步骤5 创建完毕的任务，在任务配置详情页的“仿真任务列表”可查看详情。也可在“仿真任务”页签查看详情。

----结束

任务配置相关操作

在任务配置详情页的“仿真任务列表”或“仿真任务”页签，可对仿真任务进行以下操作。

表 7-20 任务配置相关操作

任务	操作步骤
查看任务进度	将鼠标移动至进度条处，显示当前任务所包含场景中成功/失败/总场景数量。
查看任务状态	可在列表根据任务状态查询任务。
查询任务	可按照“任务ID”、“任务名称”、“任务状态”、“任务描述”、“算法版本”查询任务。
停止任务	对于任务状态为“运行中”“排队中”和“调度中”的任务，因为某种原因需要停止运行时，可单击操作栏“停止”选项，即可停止该任务。
删除任务	在任务创建有误，或任务生命周期结束，可删除任务。处于“运行中”的任务不可删除。单击操作栏“ 删除”选项，删除该任务。删除后不可恢复，请谨慎操作。

任务	操作步骤
编辑任务	单击操作栏“编辑”以修改任务名称和描述信息。

7.6.2.2 仿真任务详情

在任务管理页面，用户可以看到多批次任务的实时进度以及所处状态。单击指定任务名称，用户可以看到该任务的详细信息。在任务详情页面，用户可以看到该任务的创建时间、启动时间、运行时间、完成时间以及任务的状态等信息，同时展示任务下的所有场景的仿真评测结果，综合得分和运行结果，支持仿真软件回放场景以及将场景的结果进行信号查看。

任务分析

仿真任务得分高低，该任务是否通过检测，该任务使用的仿真算法控制质量如何由多个仿真评测指标从多个角度衡量评判。

仿真任务包含的仿真场景运行成功后，用户可以关注该仿真场景的得分，以及是否通过评测指标的检测。

评测项异常分布

在仿真任务详情页的任务分析模块，可查看各评测项异常分布。

步骤1 在左侧菜单栏中单击“仿真服务 > 并行仿真”。

步骤2 单击“仿真任务”页签，单击任务名称。或者通过“任务配置”页签进入仿真任务详情页。

步骤3 单击子任务的评测项的结果按钮（通过/未通过/无效）。

步骤4 查看评测指标的子指标的异常分布。异常分布类型分为：POINT_TYPE_POINT、POINT_TYPE_REGION、POINT_TYPE_ALL、POINT_TYPE_NORMAL，详情请参考**内置评测指标说明**。

左侧显示子指标是否成功，右侧时间轴显示指标异常的时间段，鼠标悬停异常时间轴上，单击“进入回放”，可选择进入回放查看。

步骤5 单击“信号查看器”，页面跳转至信号查看器页面，可查看关联数据，一个框图只支持显示10条信号。

如果无关联数据，则“查看关联数据”按钮置灰。

内置评测指标的可视化序列数据请查看**内置评测指标说明**。

----结束

部分子任务重试

在仿真任务详情页，支持部分子任务重新生成新的仿真任务。

步骤1 在左侧菜单栏中单击“仿真服务 > 并行仿真”。

步骤2 单击“仿真任务”页签，单击任务名称。或者通过“任务配置”页签进入仿真任务详情页。

步骤3 在“任务分析”模块，选择子任务前的复选框，单击“部分子任务重试”，填写任务名称和描述信息。

步骤4 单击“确认”，创建仿真任务。

----结束

仿真任务结果

仿真算法的质量由仿真结果呈现，仿真服务模块以自动驾驶过程中关键指标的检测结果为依据对仿真算法进行评测，回放可查看场景，信号查看器是对仿真结果的展示。

须知

1. 仿真任务结果展示可以有以下几种，按钮可单击条件判断为如下：（如果文件不存在，界面会报错提示）
   - 回放按钮：录制模式下即可单击。
   - 算法日志按钮：录制模式下，关联了算法的情况下可单击。
   - 评测日志按钮：录制模式下，关联了自定义评测镜像的情况下可单击。
   - 仿真pb文件按钮：录制模式下可单击。
   - 评测pb文件按钮：任何模式下都可单击。
   - 信号查看器按钮：任何模式下都可单击。
   - 算法pb文件按钮：录制模式下使用自定义算法，并开启Datahub才可单击下载。
2. 所有可下载的结果均不建议使用过长场景名称，如果场景名过长，浏览器会有文件名截取行为，具体截取长度因浏览器、操作系统以及文件下载路径不同而有所区别，以实际情况为准。

- 查看回放：场景已运行成功且创建任务时选择录制所有场景，则支持回放场景。仿真场景在操作栏单击“回放”，选择进行3D回放进行在线仿真回放，具体操作参考**场景回放**。
- 历史结果对比：平台支持多个历史结果进行对比，具体操作请参考**历史结果对比**。
- 算法日志下载：用户可将仿真任务的日志下载至本地查看。

  说明

  自定义算法支持下载日志，内置算法不支持日志下载。
- 评测日志下载：任务运行成功，且录制模式下关联了自定义评测镜像的情况下，可下载评测日志至本地。
- 仿真pb下载：任务运行成功，且录制策略选择录制所有场景，用户可以下载仿真pb文件。仿真pb文件以osi标准存储了仿真器在整个仿真过程中的数据，用户可以利用该仿真pb进行数据分析。
- 评测pb下载：任务运行成功后，用户可以下载评测pb文件。评测pb文件用于存储评测输出的结果，支持用户自定义评测指标输出为Octopus的评测pb格式，从而在前端进行展示。
- 算法pb下载：任务运行成功后，用户可以下载算法pb文件。算法pb文件包含感知、规控、定位等算法信息。

- 信号查看器：在已完成的任务中，在任务详情页，单击操作栏中的“信号查看器”，页面跳转至信号查看器页面，以图表的形式展现该场景自动驾驶过程中的关键数据的变化。具体操作请参考**信号查看器**。

历史结果对比

在任务详情页面，支持多个任务进行对比分析。

步骤1 在左侧菜单栏中单击“仿真服务 > 并行仿真”。

步骤2 单击仿真配置名称，进入仿真配置详情页。

步骤3 在“仿真任务列表”，单击任务名称，进入任务详情页面。

步骤4 在“任务分析”模块，单击操作栏中的“更多 > 历史结果对比”。

步骤5 选择任务，单击“确定”。页面跳转至“任务对比”页面，即可查看任务对比结果。

----结束

7.6.3 3D 回放

3D回放页面支持加载本地OSI，OpenDRIVE文件，以便于回放本地场景。

步骤1 在左侧菜单栏中选择“仿真服务 > 并行仿真”。

步骤2 在界面的右上角，单击“3D回放”，进入到3D回放界面。

步骤3 单击界面右上角“加载本地文件”。

步骤4 添加需要加载的本地文件，单击“确认”。界面回放本地文件。

----结束

7.6.4 排队任务管理

不同用户可以对自己的任务进行优先级排序。

步骤1 在左侧菜单栏中单击“仿真服务 > 并行仿真”。

步骤2 在界面的右上角，单击“排队任务管理”，可对需要管理的任务进行上移，置顶等操作。

----结束

7.6.5 信号查看器

评测任务运行过程中，将一些衡量自动驾驶质量的关键指标，如速度、加速度等数据在仿真场景中变化量绘制成图表，便于用户直观考量规控算法的控制质量。用户也可将业务所需关注的数据集合成新的图表，对比查看。

说明

查看器一次只支持载入5个任务场景，页面最多支持5个框图展示，一个框图最多支持10条信息选择。

步骤1 在左侧菜单栏中单击“仿真服务 > 并行仿真”。

步骤2 在界面的右上角，单击“信号查看器”，进入到信号查看器界面。

步骤3 单击“载入数据”，可根据“任务名称”、“算法名称”、“算法版本”搜索任务，选择对比分析数据。

步骤4 单击“载入”后，左侧会出现选择的数据。

步骤5 在右侧选择“添加框图”，右侧出现空白的框图。

步骤6 选择左侧的数据，右侧将高亮显示对应的数据，内置评测信号参数请参考**内置评测信号参数**。

步骤7 可根据需要添加数据和框图，也可根据需要选择框图的横向和纵向布局。

步骤8 单击 可删除不需要的框图和数据。单击“清空数据”，可清空页面上所有的数据。

----结束

内置评测信号参数

表 7-21 内置评测信号参数

内置评测信号参数	对应中文
jerkX	纵向加加速度
jerkY	侧向加加速度
speedX	纵向速度
speedY	侧向速度
accX	纵向加速度
accY	侧向加速度
speedYaw	横摆速度
relativeYaw	相对车道中心线偏离角
lateralOffset	相对车道中心线偏离距离
relativeSpeed	相对前车的速度
relativeDistance	相对前车的距离
TH	相对前车的车头时距
TTC	相对前车ttc
rmsAccX	纵向加速度均方根值
rmsAccY	侧向加速度均方根值
varianceSpeed	速度方差

7.6.6 场景回放

7.6.6.1 仿真器回放

当场景已运行成功且创建任务时选择录制所有场景时，仿真任务支持回放场景。

Octopus平台需要有可以正常使用的仿真器。

仿真器回放页面可自动播放车辆的行动场景。

具体仿真器操作请参考**7.2.1 仿真器**。

7.6.6.2 3D 回放

当场景已运行成功且创建任务时选择录制所有场景时，仿真任务支持回放场景。用户可选用3D回放进行回放，回放页面可自动播放车辆的行动场景。

前提要求

3D回放对回放机器配置有以下要求：

1. 回放机器需要GPU硬件。硬件加速的方式：在chrome设置-高级中打开硬件加速 。
2. 机器的参考配置（低配）：8核cpu 、UHD620的gpu 、16G内存 、100Mbps带宽。

查看 3D 回放

3D回放页面详细说明如下：

图 7-14 3D 回放页面

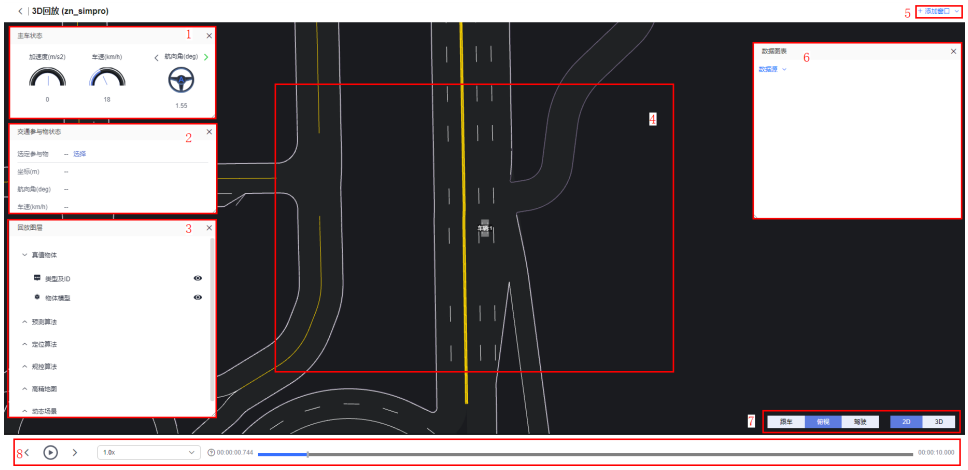


表 7-22 3D 回放页面详细说明

序号	区域名称	说明
1	主车运行状态数据	汽车驾驶的相关数据信息。如加速度、车速和航向角数据，数据随着车辆的行驶动态变化。 单击“速度”字样可以对速度的单位进行切换。
2	交通参与物状态	可根据需要选择显示参与物状态。 单击“选定参与物”后的“选择”，单击回放区域中的参与物，选择“保存”，即可显示参与物的数据。 单击“重新选择”，单击回放区域的其他参与物，选择“保存”，即可切换参与物显示数据。 **说明** 当选择项为其他物品时，主车探测不到此物品的数据时，相关值展示为空。
3	回放图层	参考**回放图层**信息。
4	回放区域	车辆的行驶轨迹，随着主车的行驶感知在主车辆周围出现的其他物体，如其他车辆、行人和交通信号等。目前可感知的物体类型请见**感知物体类型**。鼠标移动至道路时，道路会变红。遇到信号灯，车辆会按照红绿灯指示行驶。
5	添加窗口	平台支持通过关闭窗口和“添加窗口”自定义选择窗口，在3D回放界面进行显示： 单击“添加窗口”选择需要添加的窗口名称，3D回放界面即可添加新的窗口，“交通参与物状态”“数据图表”类型的窗口最多支持添加5个。
6	数据图表	可以选择多个数据源。详情参考**数据图表**。
7	场景切换	视角切换：跟车、俯视、驾驶，自由。当在3D回放页面拖动鼠标时，即可切换为自由视角。 2D/3D切换：可单击“2D/3D”，可切换2D，3D场景。
8	视频播放控件	控制视频播放暂停回放按钮，支持逐帧和倍速播放。

回放图层

在回放图层区域，可以选择不同图层，多层次观看3D回放视频。

- 真值物体：包含选项有真值物体类型及ID、物体模型，可控制真值物体显示或隐藏。
- 预测算法：可显示或隐藏除主车以外，他车的朝向前的行驶轨迹。
- 定位算法：可显示或隐藏主车的定位（通过算法计算出的信息）和真值（实际信息）的坐标信息。
- 规控算法：可显示或隐藏“主车局部规划轨迹”和“主车全局规划轨迹”。
- 高精地图：显示“路面”、“车道线”、“车道参考线”、“中心车道线”、“交通信号”，可控制对应功能显示或者隐藏。
- 动态场景：显示场景trigger信息，目前支持设置了Distance，ReachPosition，End Of Road，Collision，OffRoad，Time Head Way，Time To Collision，Acceleration，StandStill，Speed，Relative Speed，Traveled Distance，

Relative Distance这几类trigger。地图会以  的图标展现，鼠标单击图标时，地图中会出现trigger的详细信息。

说明

1. 当有trigger作用于某个交通参与物时，且该trigger自身没有position。该交通参与物模型上会显示trigger图标。（注：Time To Collision类型的trigger可能包含position信息，但是该position代表目标位置，不是其本身位置，所以此类trigger也会显示在交通参与物上）。
2. 一个trigger作用于多个交通参与物时，涉及到的交通参与物都会带有trigger图标。
3. 一个交通参与物带有多个trigger时，只会显示一个trigger图标，鼠标悬浮上去，会在右上角列出所有trigger的内容。

数据图表

步骤1 单击“数据源”，选择“添加数据源”。

步骤2 选择数据源后，数据图表显示选择的评测对象。

步骤3 当3D回放在播放的同时，数据图表中会有标准线跟随时间轴同时移动。

----结束

感知物体类型

须知

- 由于各类仿真器对交通参与物类型的覆盖面不同，可能产生仿真器不支持某种类型，或仿真器与仿真回放中类型不匹配。
- 以下模型均为副车的颜色，主车ego的颜色为白色，其他车比主车颜色更深。

表 7-23 车辆和行人

类别	标签	三维感知框/未选中	三维感知框/选中	3D模型
未知车辆/其他车辆/小型车/紧凑型轿车/中型车/豪华车				
送货车/重型卡车/半挂式拖车/拖车				

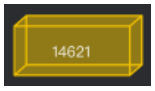
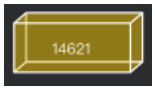
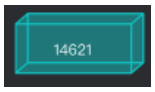
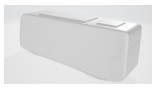
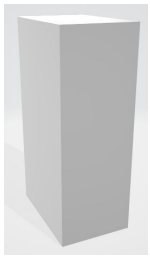
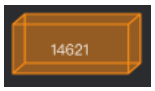
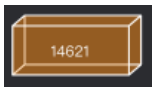
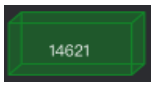
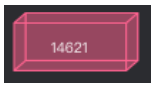
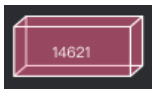
类别	标签	三维感知框/未选中	三维感知框/选中	3D模型
行人				
公交车				
未知/其他/轮椅				
摩托车/自行车				
电车/火车				暂无
动物				暂无

表 7-24 交通信号

类型	名称	图标
信号灯	红黄绿灯	
	方向灯-箭头向上	
	方向灯-箭头向左	
	方向灯-箭头向前向左	

类型	名称	图标
	方向灯-箭头向右	
	方向灯-箭头向前向右	
交通标识牌	注意安全	
	人行横道	
	两侧变窄	
	右侧变窄	
	左侧变窄	
	减速让行	
	停车让行	
	禁止驶入	
	禁止掉头	
	禁止停车	
	禁止长时停放	
	注意落石	

7.7 批量仿真调优

仿真服务可以提供基于云资源的大规模并行仿真。仿真过程中，涉及资源配置与调度。合理配置任务的资源占用，可以尽可能的提高资源利用率，进而提高仿真测试并行度，增加算法测试的里程数。

获取并配置算法实际资源占用

创建仿真算法时，需要填写算法镜像占用的CPU以及内存。这个将影响批量仿真子任务的资源调度，在相同资源情况下，算法资源配置越高，同时运行的任务数越少，因此尽可能配置实际资源占用，可以提高集群的利用效率。

先在本地启动算法容器，等算法全部启动运行后，使用以下方法估计资源占用情况：

```
docker stats my_container
```

其中my_container是运行容器的名称。

假设输出如下：

```
CONTAINER ID   NAME          CPU %     MEM USAGE / LIMIT     MEM %     NET I/O         BLOCK I/O         PIDS
b5ecde01ad2d   my_container  25.00%    256MiB / 500MiB       25.00%    1.2kB / 1.2kB   0B / 0B           71
```

观察并记录容器的CPU以及MEM占用情况，假设最大占用情况为：

- CPU：25.00%
- MEM USAGE：256MiB

根据上述信息，可以稍有余量的设置CPU和内存：

- CPU：25%的CPU使用率意味着容器使用了0.25个CPU核心，因此可以设置CPU为0.3核。
- 内存：256MiB，因此可以设置内存值为300MiB。

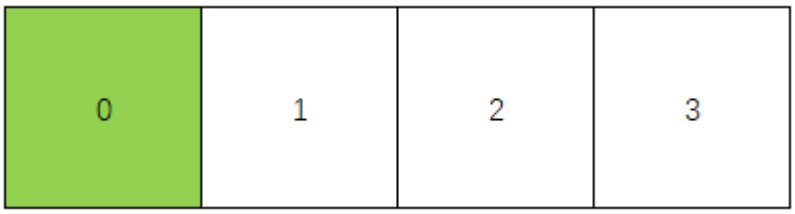
这样可以尽可能减小单个任务的资源占用，从而提高并行任务数量，提升批量仿真性能。

注意：如果算法容器的值超过了前端界面提示的最大值，那么需要考虑提升集群节点的CPU或者内存配置。例如：界面显示CPU最大12.19，如果算法占用15核，那么当前集群是无法满足调度的，需要提升集群节点配置。

并行仿真任务资源利用说明

当前批量仿真任务同一用户仅允许同时运行一个批量仿真任务。假设有一个4节点的集群，某个任务只有一个场景，那么仿真会创建一个子任务，此时，任务只会被调度到一个节点上，示意图如图7-15所示。

图 7-15 调度节点



这样某一台机器工作，其他3台机器资源会空闲，造成资源浪费。因此建议批量仿真时，尽量增大批量仿真的子任务数（与被测场景数量有关），使得集群节点内的所有机器资源被充分利用，减少机器空闲状态。

7.8 Open SCENARIO2.0 场景说明

7.8.1 动态场景

7.8.1.1 场景组成

场景文件结构

场景文件结构样例

```
OpenSCENARIO2.0
├─ RoadNetwork
│   ├─ LogicFile
│   └─ SceneGraphFile (0..1)
├─ ParameterDeclarations (0..1)
├─ Entities
│   └─ Entity (1+)
└─ Storyboard
    ├─ StopTrigger
    ├─ InitActions
    │   ├─ AssignInitSpeedAction
    │   └─ AssignInitPositionAction
    └─ Story
        └─ Act (1+)
            ├─ StartTrigger (Wait)
            │   └─ ConditionGroup
            │       └─ Condition (1+)
            └─ Action
```

场景组成说明

场景文件的主体是一个场景剧本storyboard，用户需要在storyboard前先声明将会使用的路网RoadNetwork、参数Parameter，和实体Entities。

然后在Storyboard中通过InitActions对实体进行初始化（给定初始速度和位置）。

通过场景故事Story中实体Entities间的每个动作集Act来展开场景。对于动作集Act内的每个行为Action，用户还可以通过Wait设置一个或多个触发条件。

7.8.1.2 场景样例（Scenario Examples）

如下介绍具体场景样例和逻辑场景样例。

- 具体场景样例

```
import standard

scenario my_scenario:
    # Road Network
    map: map
    map.set_map_file("./road.xodr")

    # Parameter Declarations
    m_z: length = 5.0m
    LeadVehicle_InitDistance: length = 20.0m
    Ego_InitSpeed: speed = 16.66666667mps
    delay1: time = 20s
    delay2: time = 5s
    Swerve_Offset: length = -1.0m
    Swerve_MaxLateralAcc: speed = 0.3mps
    m_sinusoidal: dynamics_shape = sinusoidal
    m_right: side_left_right = right
    m_odr: odr_point = map.create_odr_point(road_id: '0', lane_id: '-4', s: 5.0m, t: 0.0m)
    Ego_InitPosition: pose_3d with:
        keep(it.odr_point == m_odr)
    m_xyz: xyz_point = map.create_xyz_point(x: 0.0m, y: 10.0m,z: m_z)
    person_InitPosition: pose_3d with:
        keep(it.xyz_point == m_xyz)
    m_mode: distance_mode = reference_points
```



```
    m_direction: distance_direction = euclidianDistance
    m_profile: dynamics_shape = step
    m_side: lane_change_side = same

    # Entities
    Ego_name: string = "Audi_A3_2009_black"
    Side_vehicle_name: string = "Audi_A3_2009_red"
    Ego_controller: string = "DefaultDriver"
    Ego: vehicle with:
        keep(it.name == Ego_name)
        keep(it.initial_bm == Ego_controller)
    lead_vehicle: vehicle with:
        keep(it.name == Side_vehicle_name)
    woman: person with:
        keep(it.name == "Christian")
        keep(it.model == "female_adult")

    # Storyboard
    do parallel:
        # InitActions
        Ego.assign_init_position(position: Ego_InitPosition)
        Ego.assign_init_speed(Ego_InitSpeed)
        lead_vehicle.assign_init_position() with:
            lane(lane: 0, side_of: Ego, side: m_right)
            position(LeadVehicle_InitDistance, ahead_of: Ego)
        lead_vehicle.assign_init_speed() with:
            speed(same_as: Ego)
        woman.assign_init_position(person_InitPosition)

        # Story
        serial:
            # action 1: ActivateALKSController
            wait elapsed(delay1)
            Ego.activate_controller(true, true)
        serial:
            # action 2: SwerveAction
            wait elapsed(delay2)
            lead_vehicle.lane_offset(Swerve_Offset, Swerve_MaxLateralAcc, m_sinusoidal)
        serial:
            wait lead_vehicle.object_distance(reference: Ego, direction: m_direction, mode: m_mode) < 5m
            parallel:
                Ego.change_lane(side:m_side, reference: lead_vehicle, side:m_side, rate_profile: m_profile,
rate_peak: 0.3mps)
                lead_vehicle.change_speed(target: 0.2mps, rate_profile: m_sinusoidal, rate_peak: -0.1mpss)
```

- 逻辑场景样例

```
import standard

scenario my_scenario:
    # Road Network
    map: map
    map.set_map_file("./road.xodr")

    # Parameter Declarations
    m_z: length = [0.0m..5.0m]
    LeadVehicle_InitDistance: length = 20.0m
    Ego_InitSpeed: speed = 16.66666667mps
    delay1: time = [10s, 20s, 60s]
    delay2: time = 5s
    Swerve_Offset: length = -1.0m
    Swerve_MaxLateralAcc: speed = 0.3mps
    m_sinusoidal: dynamics_shape = [linear, sinusoidal]
    m_right: side_left_right = [left, right]
    m_odr: odr_point = map.create_odr_point(road_id: '0', lane_id: '-4', s: 5.0m, t: 0.0m)
    Ego_InitPosition: pose_3d with:
        keep(it.odr_point == m_odr)
    m_xyz: xyz_point = map.create_xyz_point(x: 0.0m, y: 10.0m,z: m_z)
    person_InitPosition: pose_3d with:
        keep(it.xyz_point == m_xyz)
    m_mode: distance_mode = reference_points
```



```
m_direction: distance_direction = euclidianDistance
m_profile: dynamics_shape = [step, linear]
m_side: lane_change_side = [same, right]

# Entities
Ego_name: string = "Audi_A3_2009_black"
Side_vehicle_name: string = "Audi_A3_2009_red"
Ego_controller: string = ["DefaultDriver", "ComfortableDriver"]
Ego: vehicle with:
    keep(it.name == Ego_name)
    keep(it.initial_bm == Ego_controller)
lead_vehicle: vehicle with:
    keep(it.name == Side_vehicle_name)
woman: person with:
    keep(it.name == "Christian")
    keep(it.model == "female_adult")

# Storyboard
do parallel:
    # InitActions
    Ego.assign_init_position(position: Ego_InitPosition)
    Ego.assign_init_speed(Ego_InitSpeed)
    lead_vehicle.assign_init_position() with:
        lane(lane: 0, side_of: Ego, side: m_right)
        position(LeadVehicle_InitDistance, ahead_of: Ego)
    lead_vehicle.assign_init_speed() with:
        speed(same_as: Ego)
    woman.assign_init_position(person_InitPosition)

    # Story
    serial:
        # action 1: ActivateALKSController
        wait elapsed(delay1)
        Ego.activate_controller(true, true)
    serial:
        # action 2: SwerveAction
        wait elapsed(delay2)
        lead_vehicle.lane_offset(Swerve_Offset, Swerve_MaxLateralAcc, m_sinusoidal)
    serial:
        wait lead_vehicle.object_distance(reference: Ego, direction: m_direction, mode: m_mode) < 5m
        parallel:
            Ego.change_lane(side:m_side, reference: lead_vehicle, side:m_side, rate_profile: m_profile,
rate_peak: 0.3mps)
            lead_vehicle.change_speed(target: 0.2mps, rate_profile: m_sinusoidal, rate_peak: -0.1mpss)
```

7.8.1.3 代码样例（Code Examples）

7.8.1.3.1 路网设置（Road Network）

路网设置的相关文件都需要在交互页面上传，如果语句设置文件与上传文件有出入，以上传文件为准。

地图文件（Logic file）

地图文件（xodr文件）使用set_map_file语句指定。

具体场景使用xodr文件，逻辑场景使用odr文件。

例1：具体地图

```
map: map
map.set_map_file("./road.xodr")
```

例2：逻辑地图


```
map: map
map.set_map_file("./road.odr")
```

7.8.1.3.2 参数声明（Parameter Declarations）

参数声明格式为“参数名: 参数类型”，可以使用“=”为参数赋值，例如“m_distance: length = 10.0m”。

须知

无论是具体场景还是逻辑场景，都不要对一个参数重复赋值。

具体场景（Concrete scenario）

具体场景的参数声明支持**基础类型**、**标量(scalar)类型**、**枚举(enum)类型**，和**结构(struct)类型**：

- **基础类型**包含int、float、bool，和string类型。可以直接在等号后赋值。
- **scalar类型**包含speed、acceleration、length、time，和angle类型。赋值时需要在值后加上OSC2.0支持的单位（见附录**Scalar units**）。
- **enum类型**包含side_left_right、distance_direction、distance_mode、lane_change_side、 dynamics_shape和catalog。赋值时必须使用枚举列表内的值（见附录**Enum list**） 。
- **struct类型**包含odr_point、xyz_point、position_3d、road_point、orientation_3d和pose_3d等，可以使用keep创建。此外，odr_point、xyz_point，和road_point分别可以使用map的成员函数create_odr_point 、create_xyz_point和create_road_point来创建（OSC2.0支持的struct类型详见附录**Struct**）。

须知

- 给bool型赋值时必须使用“true”和“false”，如果使用“True”或“False”或拼写错误将使语句无效。

例1：基础类型

```
m_road_id: string = '0'
Ego_name: string = "Audi_A3_2009_black"
m_lateral: bool = true
```

例2：scalar类型

```
v: speed = 5mps
delay: time = 40s
distance: length = 30.0m
m_a: acceleration = 0.01mpss
```

例3：enum类型

```
m_side: side_left_right = right
m_direction: distance_direction = longitudinal
```

例4：struct类型，keep创建

```
my_odr: odr_point with:
    keep(it.road_id == 0)
```

```
    keep(it.lane_id == -1)
    keep(it.s == 3.0m)
    keep(it.t == 0.0m)

my_pos: position_3d with:
    keep(it.x == 1.0m)
    keep(it.y == 2.0m)
    keep(it.z == 3.0m)

my_xyz: xyz_point with:
    keep(it.position == my_pos)

my_road: road_point with:
    keep(it.road_id == '1')
    keep(it.s == 3.0m)
    keep(it.t == 0.0m)

my_orientation: orientation_3d with:
    keep(it.roll == 1.0rad)
    keep(it.pitch == 2.0rad)
    keep(it.yaw == 3.0rad)

my_pose: pose_3d with:
    keep(it.xyz_point == my_xyz)
    keep(it.orientation == m_orientation)

m_trajectory: trajectory with:
    keep(it.points == [my_pose1, my_pose2, my_pose3])
    keep(it.time_stamps == [0s, 5s, 20s] )
```

例5：struct类型，map创建

```
my_odr: odr_point = map.create_odr_point(road_id: m_road_id,lane_id:'-4',s: 5.0m, t: 0.0m)
my_xyz: xyz_point = map.create_xyz_point(x: 2.5m, y: 10.0m ,z: 0.0m)
my_point: road_point = map.create_road_point(road_id: '1', s: 5.0m, t: 0.0m)
```

逻辑场景（Logical scenario）

逻辑场景的参数声明通过：范围型[最小值..最大值] 和枚举型[值1, 值2] 的方式来实现泛化：

- 范围型支持float和scalar类型。
- 枚举型支持int, float, bool, str, enum和scalar类型，且需要保证枚举列表中的元素均为相同类型。

例1：范围型

```
m_value: float = [2.0..3.0]
v: speed = [5mps..10mps]
delay: time = [40s..60s]
m_a: acceleration = [0.00mpss..0.03mpss]
```

例2：枚举型

```
m_ id: int = [-1, 2]
m_value: float = [2.0, 3.0]
m_on_road: bool = [true, false]
Ego_name: string = ["Audi_A3_2009_black", "Audi_A3_2009_red"]
m_shape: dynamics_shape = [linear, sinusoidal]
m_side: side_left_right = [left, right]
v: speed = [5mps, 7mps, 10mps]
m_a: acceleration = [0.01mpss, 0.03mpss]
delay: time = [40s, 60s, 100s]
```

- 通过泛化参数，可以实现实体entity 、动作act ，以及修饰器modifier的泛化。
- Struct类型不能直接泛化，而是通过泛化参数来进行泛化。

例1：entity泛化

```
Ego_name: string = ["Audi_A3_2009_black", "Audi_A3_2009_red"]
Ego_controller: string = "DefaultDriver"
Ego: vehicle with:
    keep(it.name == Ego_name)
    keep(it.initial_bm == Ego_controller)
```

例2：act泛化

```
m_lateral: bool = [true, false]
m_speed: speed = [5mps..15mps]
m_rate_profile: dynamics_shape = linear
Ego.activate_controller(m_lateral, true)
Ego.change_speed(target: m_speed, rate_peak: 0.0mpss, rate_profile: m_rate_profile)
```

例3：modifier泛化

```
m_speed: speed = [5mps..15mps]
Ego.assign_init_speed() with:
    speed(speed: m_speed)
```

例4：Struct类型泛化

```
m_lane_id: int = [-1, 2]
m_odr: odr_point = map.create_odr_point(road_id: '0', lane_id:m_lane_id, s: 5.0m, t: 0.0m)

m_x: distance = [2m..7.5m]
m_position_3d: position_3d = map.create_xyz_point(x: m_x, y: 10.0m ,z: 0.0m)

m_pose_3d: pose_3d with:
    keep(it.odr_point == m_odr)
```

7.8.1.3.3 实体设置（Entities）

车辆和控制器（Vehicle and controller）

通过***车辆名: vehicle***的方式来为车辆命名。通过***keep(it.name == 指定车型名称)***的方式来指定车辆类型。通过***keep(it.initial_bm == 指定controller名称)***的方式来指定controller类型，目前均只能支持仿真器B内置的车辆和controller类型。生成文件会自动适配车辆信息。

name，initial_bm等车辆属性需要在仿真器的catalog列表中定义，不同仿真器的预定义的catalog内容有所不同，需要在撰写场景文件时确认使用的车型、controller名称在仿真器catalog中已经存在。

须知

- name为必选项，initial_bm非必选项。
- 主车必须命名为Ego，否则仿真器B将无法识别。

例1：主车，指定initial_bm

```
Ego_name: string = "Audi_A3_2009_black"
Ego_controller: string = "DefaultDriver"
Ego: vehicle with:
    keep(it.name == Ego_name)
    keep(it.initial_bm == Ego_controller)
```

例2：非主车，不指定initial_bm


```
cut_in_vehicle: vehicle with:
    keep(it.name == "Audi_A3_2009_red")
```

例3(仿真器B)：主车，指定initial_bm

```
cut_in_vehicle: vehicle with:
    keep(it.name == "Saimo")
    keep(it.initial_bm == "默认驾驶员")
```

例4(仿真器B)：非主车，不指定initial_bm

```
cut_in_vehicle: vehicle with:
    keep(it.name == "Saimo")
```

行人（Pedestrian）

通过***行人名: person***的方式来为行人命名。通过***keep(it.name == 指定行人名称)***的方式来指定行人类型。通过***keep(it. model == 指定行人特征)***的方式来指定行人的性别、年龄特征。

样例

```
Dude: person with:
    keep(it.name == "Christian")
    keep(it.model == "male_adult")
```

7.8.1.3.4 场景剧本（StoryBoard）

执行顺序（Execution sequence）

OSC2.0场景剧本StoryBoard通过执行顺序Execution Sequence和触发器Trigger来支持用户设计各种场景。StoryBoard中有parallel和serial两种执行指令，最外层执行指令之前需要加上do来使场景剧本生效。其中：

- parallel：**同步执行**下方代码块内的动作action。
- serial：**依次执行**下方代码块内的动作act。

例如下方样例中，do parallel:下的assign_init_speed，assign_init_position和wait elapsed(10s) 是同步执行的。而serial:下的lead_vehicle.change_speed在Ego.activate_controller完成之后执行。

须知

由于初始动作InitAction内的action同步执行，且InitAction与story之间不涉及顺序执行，建议场景最外层统一使用parallel。

样例

```
m_profile: dynamics_shape = linear
do parallel:
    # InitAction
    Ego.assign_init_speed(15mps)
    Ego.assign_init_position(position: Ego_InitPosition)
    # Story
    serial:
        # act1
        Ego.activate_controller(true, true)
        lead_vehicle.change_speed(target: 20mps, rate_profile: m_profile, rate_peak: 0.3mpss)
    serial:
        # act2
        wait elapsed(10s)
        lead_vehicle.activate_controller(true, true)
```


7.8.1.3.5 触发器与触发条件（Trigger and condition）

用户可以使用*wait+触发条件*的方式来设置动作的触发条件condition，可以使用的触发条件有：**触发条件（elapsed）**、**触发条件（object_distance）**、**触发条件（point_distance）**、**触发条件（time_to_collision）**、**触发条件（time_headway）**、**触发条件（speed）**和**触发条件（acceleration）**。如果不满足等待时长或触发条件，后续动作将无法执行。

⚠ 注意

使用wait来设定触发条件condition时，必须在serial下执行，否则wait无效。

触发条件（elapsed）

使用***wait elapsed(time)***的方式设定等待时间，不返回值。

- 使用方法：经过设定的等待时间后，触发动作。

样例

```
serial:
    # action1
    wait elapsed(10s)
    Ego.activate_controller(true, true)
```

触发条件（object_distance）

返回语句中涉及的两个实体entity之间的相对距离。

表 7-25 object_distance 参数

Parameter	Type	Mandatory	Description
reference	entity	yes	The reference entity
direction	**distance_direction**	yes	Measure the distance in the x-coordinate/y-coordinate/xy-coordinate
mode	**distance_mode**	yes	Use reference_points/bounding_boxes to measure the distance between the reference points/bounding boxes

- 使用方法：当动作主体与参考实体之间的距离>或< 或==某个长度值时，触发动作。

样例

```
m_direction: distance_direction = euclidianDistance
m_mode: distance_mode = reference_points
do parallel:
   # Story
   serial:
      # action1:
      wait lead_vehicle.object_distance(reference: Ego, direction: m_direction, mode: m_mode) < 5m
      Ego.activate_controller(true, true)
```

触发条件（point_distance）

返回主体与参考点的距离。

- 参数：

表 7-26 point_distance 参数

Parameter	Type	Mandatory	Description
reference	pose_3d	yes	The reference point
direction	**distance_direction**	yes	Measure the distance in the x-coordinate/y-coordinate/xy-coordinate
mode	**distance_mode**	yes	Use reference_points/bounding_boxes to measure the distance between the reference points/bounding boxes

使用方法：当动作主体与参考点之间的距离>或<或==某个长度值时，触发动作。

样例

```
my_xyz: xyz_point = map.create_xyz_point(x: 0.0m, y: 10.0m,z: m_z)
my_pose3d: pose_3d with:
   keep(it.xyz_point == my_xyz)
m_direction: distance_direction = euclidianDistance
m_mode: distance_mode = reference_points
do parallel:
   # Story
   serial:
      # action1
      wait Ego.point_distance(my_pose3d, m_direction, m_mode)  >  10.0m
      Ego.activate_controller(true, true)
```

触发条件（time_to_collision）

假设语句中涉及的两个实体entity都以当前速度移动，返回直到两者边界框**bounding_box**碰撞的时间（两车的车距/两车相对距离）。

- 参数：

表 7-27 time_to_collision 参数

Parameter	Type	Mandatory	Description
reference	entity	yes	The reference entity

使用方法：当距离两者碰撞的时间>或<或==某个值时，触发动作。

样例

```
m_profile: dynamics_shape = linear
do parallel:
   # Story
   serial:
      # action1:
      wait cut_in_vehicle.time_to_collision(reference:Ego) < 3s
      Ego.change_speed(target: 0.0mps, rate_profile: m_profile, rate_peak: -0.3mpss)
```

触发条件（time_headway）

假设语句中涉及的两个实体entity都以当前速度移动，返回两者参考点**reference point**之间沿s坐标的时间距离（两车的车距/本车的车速）。

- 参数：

表 7-28 time_headway 参数

Parameter	Type	Mandatory	Description
reference	entity	yes	The reference entity

- 使用方法：当两者的时间距离>或<或==某个值时，触发动作。

样例

```
m_profile: dynamics_shape = linear
do parallel:
   # Story
   serial:
      # action1:
      wait cut_in_vehicle.time_headway(reference:Ego) < 3s
      Ego.change_speed(target: 0.0mps, rate_profile: m_profile, rate_peak: -0.3mpss)
```

触发条件（speed）

返回实体的速度。

- 使用方法：当动作主体的速度>或<或==某个值时，触发动作。

样例

```
do parallel:
   # Story
   serial:
      # action1
```

```
    wait Ego.speed  >  30.0mps
    Ego.activate_controller(true, true)
```

触发条件（acceleration）

返回实体的加速度。

- 使用方法：当动作主体的加速度>或<或==某个值时，触发动作。

样例

```
do parallel:
  # Story
  serial:
    # action1
    wait Ego.acceleration  >  1.0mpss
    Ego.activate_controller(true, true)
```

7.8.1.3.6 动作（Actions）

用户首先使用初始动作InitActions来初始化实体entities，接着使用其他动作Actions展开实体entities的场景故事story。

初始动作（assign_init_speed）

- **动作主体**：车辆vehicle或行人pedestrian 。
- **结束时间**：当动作主体actor达到指定的速度speed时，动作结束。
- **是否支持modifier**：是
- **参数**：参数如下表

表 7-29 assign_init_speed 参数

Parameter	Type	Mandatory	Description
target	speed	yes	Desired (scalar) speed assigned by the user

注意

- assign_init_speed支持设置绝对速度和相对速度，设置相对速度时使用**修饰器（speed）**来给出相对值。
- 设置初始速度时，初始速度不能超过该主体所在道路的限速。
- 如果所需场景起始速度为0，无需使用assign_init_speed动作。

绝对速度

```
init_speed: speed = 10.0mps
Ego.assign_init_speed(init_speed)
```

相对速度

```
cut_in_vehicle.assign_init_speed() with:
  speed(speed: 5mps, faster_than: Ego)
```

初始动作（assign_init_position）

- **动作主体**：车辆vehicle或行人pedestrian。
- **结束时间**：当动作主体actor到达指定的位置坐标时，动作结束。
- **是否支持modifier**：是
- **参数**：参数如下表

表 7-30 assign_init_position 参数

Parameter	Type	Mandatory	Description
position	pose_3d	no	Desired position assigned by the user

注意

- assign_init_position支持设置绝对位置和相对位置，设置相对位置时使用修饰器（position）和修饰器（lane）来给出相对值。
- 当初始位置需要车辆转换朝向时，通过设置pose_3d的orientation来设定朝向，以与所在车道朝向一致，例如车辆所在车道和road0呈90°夹角时，设置orientation的yaw为1.57rad，否则车辆启动后会有转向行为，影响场景的正常执行。
- 设置初始位置时所采用的地图元素必须是对应的地图中有的元素，比如设置绝对位置create_odr_point('0', '-4', 5.0m, 0.0m)时，地图上必须有id为'0'的road，该road上必须有lane_id为 '-4'的lane，该lane至少有5.0m以上的长度。如果设置的初始位置找不到地图中的对应元素，将泛化出无效的场景。

绝对位置

```
m_odr: odr_point = map.create_odr_point('0', '-4', 5.0m, 0.0m)
m_orientation: orientation_3d with:
    keep(it.roll == 0.0rad)
    keep(it.pitch == 0.0rad)
    keep(it.yaw == 1.57rad)
init_position: pose_3d with:
    keep(it.odr_point == m_odr)
    keep(it.orientation == m_orientation)
Ego.assign_init_position(position: init_position)
```

相对位置

```
cut_in_vehicle.assign_init_position() with:
    lane(lane: raletive_lane_id, side_of: Ego, side: left)
    position(distance: 85.0m, behind: Ego)
```

初始动作（acquire_position_init）

- **动作主体**：车辆vehicle或行人pedestrian。
- **结束时间**：当动作主体actor获取目标位置position时，动作结束。
- **是否支持modifier**：否
- **参数表**： 参数如下表，pose_3d是point和orientation的组合结构，point可以使用xyz_point或odr_point或road_point中的任意一个，orientation非必选项。

表 7-31 acquire_position_init 参数

Parameter	Type	Mandatory	Description
target	pose_3d	yes	target position

⚠ 注意

- 目标位置必须在地图设定的道路上，且是可达的。

xyz_point，有方向要求

```
m_x: length = 0.0m
m_xyz: xyz_point = map.create_xyz_point(x: m_x, y: 10.0m ,z: 0.0m)
m_heading: angle = 1.57rad
m_orientation: orientation_3d with:
    keep(it.roll == 0.0rad)
    keep(it.pitch == 0.0rad)
    keep(it.yaw == m_heading)
m_position: pose_3d with:
    keep(it.xyz_point == m_xyz)
    keep(it.orientation == m_orientation)
Ego.acquire_position_init(target: m_position)
```

odr_point，无方向要求

```
m_odr: odr_point = map.create_odr_point(road_id: '0', lane_id: '-4', s: 5.0m, t: 0.0m)
m_position: pose_3d with:
    keep(it.odr_point == m_odr)
Ego.acquire_position_init(target: m_position)
```

road_point，无方向要求

```
m_road: road_point = map.create_road_point(road_id: '0', s: 5.0m, t: 0.0m)
m_position: pose_3d with:
    keep(it.road_point == m_road)
Ego.acquire_position_init(target: m_position)
```

动作（change_speed）

- **动作主体**：车辆vehicle或行人pedestrian。
- **结束时间**：当动作主体actor达到目标速度target ，动作结束。
- **是否支持modifier**：否
- **参数**：参数如下表，支持位置参数和关键字参数。

表 7-32 change_speed 参数

Parameter	Type	Mandatory	Description
reference	entity	no	Default=it.actor. Reference to the entity that is used to determine the target speed. If this argument is omitted, the actor itself is used as reference

Parameter	Type	Mandatory	Description
target	speed	yes	Target value for the speed at the end of the action
rate_profile	dynamics_shape	yes	Assign a shape for the change of the speed variable. This profile affects the acceleration during action execution
rate_peak	acceleration	yes	Target value for the peak acceleration that must be achieved during the action

⚠ 注意

- 目标速度不能超出所在道路的限速值。
- 当rate_profile为step时，瞬间达到目标速度，不会受到rate_peak值的影响。

样例

```
m_profile: dynamics_shape = sinusoidal
Ego.change_speed(reference: lead_vehicle, target: -5.0mps, rate_profile: m_profile, rate_peak: 2mpss)
```

动作（change_lane）

- **动作主体**：车辆vehicle
- **结束时间**：当动作主体actor位于目标车道lane中、目标偏移offset处时，动作结束。
- **是否支持modifier**：否
- **参数**：参数如下表，支持关键字参数。rate_peak和rate_profile是必选项，用于设置osc1中的dynamics、target和reference必须设置且只设置其中之一，前者用于指定绝对车道，后者用于指定相对车道。

表 7-33 change_lane 参数

Parameter	Type	Mandatory	Description
number_of_lanes	uint	no	The target lane is "num_of_lanes" to the side of the reference entity. Use in conjunction with "side"
side	lane_change_side	no	Select on which side of the reference entity
reference	entity	no	Default=it.actor. Reference to the entity that is used to determine the target lane. If this argument is omitted, the actor itself is used as reference

Parameter	Type	Mandatory	Description
offset	length	no	Default=0.0. Target offset from center of the target lane that the actor follows at the end of the action
rate_profile	dynamics_shape	yes	Assign a shape for the change of the lateral position variable (t-axis). This profile affects the lateral velocity during action execution
rate_peak	speed	yes	Target value for the peak lateral velocity that must be achieved during the action
target	lane_id(uint)	no	The actor starts and finishes the action in the target lane

⚠ 注意

- 目标lane必须在地图上。
- 中心线左侧的lane_id为正，如'1'，'2'。右侧的lane_id为负，如'-1'，'-3'。绝对值越大，距离中心线越远。
- offset值不能超出当前所在lane的宽度范围。
- 使用相对位置时，参考对象reference必须是车辆。
- rate_profile只能选择linear或step。

绝对车道1

```
my_lane: lane with:
    keep(it.lane_id == '1')
m_profile: dynamics_shape = linear
side_vehicle.change_lane(target: my_lane, rate_profile: m_profile, rate_peak: 0.3mps)
```

绝对车道2

```
my_lane: lane with:
    keep(it.lane_id == '-1')
m_profile: dynamics_shape = linear
side_vehicle.change_lane(target: my_lane, offset: -0.2m, rate_profile: m_profile, rate_peak: 0.4mps)
```

相对车道1

```
m_profile: dynamics_shape = step
m_side: lane_change_side = left
m_profile: dynamics_shape = step
Ego.change_lane(number_of_lanes: 2, side: m_side, reference: side_vehicle, offset: 0.8m, rate_profile:
m_profile, rate_peak: 0.3mps)
```

相对车道2

```
m_profile: dynamics_shape = step
m_side: lane_change_side = same
m_profile: dynamics_shape = step
Ego.change_lane(number_of_lanes: 1, reference: side_vehicle, side:m_side, rate_profile: m_profile, rate_peak:
0.3mps)
```

动作（activate_controller）

- **动作主体**：车辆vehicle
- **结束时间**：激活或停用控制器controller后，动作结束。
- **是否支持modifier**：否
- **参数**：参数如下表，支持位置参数和关键字参数。

表 7-34 activate_controller 参数

Parameter	Type	Mandatory	Description
lateral	bool	yes	In lateral domain: Activate or deactivate controller defined (e.g. automated, autonomous) behavior
longitudinal	bool	yes	In longitudinal domain: Activate or deactivate autonomous behavior

代码样例：

```
Ego.activate_controller(lateral:true, longitudinal:true)
```

动作（lane_offset）

- **动作主体**：车辆vehicle
- **结束时间**：当动作主体actor到达目标偏移offset处时，动作结束。
- **是否支持modifier**：否
- **参数**：参数如下表，支持位置参数和关键字参数。参数target非必选项，不设置target时采用绝对偏移，设置target时采用相对偏移。

表 7-35 lane_offset 参数

Parameter	Type	Mandatory	Description
offset	length	yes	Signed number in meters the vehicle should respect as an offset from the center of the current lane
rate_peak	speed	yes	Target value for the peak lateral velocity that must be achieved during the action
dynamics_shape	dynamics_shape	yes	Geometrical shape of the LaneOffsetAction's dynamics
target	entity	no	Reference entity

⚠ 注意

- offset值不能超出当前所在lane的宽度范围。

绝对偏移

```
m_shape: dynamics_shape = step
Ego.lane_offset(0.8m, 0.5mps, m_shape)
```

相对偏移

```
m_shape: dynamics_shape = linear
side_vehicle.lane_offset(offset:1.0m, rate_peak: 1.5mps, dynamics_shape:m_shape, target: Ego)
```

动作（acquire_position）

- **动作主体**：车辆vehicle或行人pedestrian。
- **结束时间**：当动作主体actor获取目标位置position时，动作结束。
- **是否支持modifier**：否
- **参数表**： 参数如下表，pose_3d是point和orientation的组合结构， point可以使用xyz_point或odr_point或road_point中的任意一个，orientation非必选项。

表 7-36 acquire_position 参数

Parameter	Type	Mandatory	Description
target	pose_3d	yes	target position

⚠ 注意

- acquire_position与acquire_position_init的使用方法相同，但由于不是初始动作，可以设置触发条件。
- 仿真器B尚未支持acquire_position动作。
- 代码样例见**初始动作（acquire_position_init）**。

7.8.1.3.7 修饰器（Modifiers）

修饰器（speed）

用途：设定动作主体actor在当前阶段的速度。可以修饰初始动作 **assign_init_speed** 。

参数：

表 7-37 speed 参数

Parameter	Type	Mandatory	Description
speed	speed	no	A target speed value, including a speed unit
faster_than	entity	no	set the speed target faster than another entity
slower_than	entity	no	set the speed target slower than another entity
same_as	entity	no	set the speed target same as another entity

⚠ 注意

- faster_than、slower_than和same_as必须设置且仅设置一个。
- 使用faster_than和slower_than时配合speed来设置相对速度。

使用speed+faster_than

```
cut_in_vehicle.assign_init_speed() with:
    speed(speed: 5mps, faster_than: Ego)
```

使用speed+slower_than

```
cut_in_vehicle.assign_init_speed() with:
    speed(speed: 5mps, slower_than: Ego)
```

使用same_as

```
cut_in_vehicle.assign_init_speed() with:
    speed(same_as: Ego)
```

修饰器（lane）

用途：设定动作主体actor所处的车道。可以修饰初始动作**assign_init_position**。

参数:

表 7-38 lane 参数

Parameter	Type	Mandatory	Description
lane	int	no	Relative value of the target lane_id
same_as	entity	no	Option to specify that the vehicle must be in the same lane as the referenced vehicle

Parameter	Type	Mandatory	Description
side_of	entity	no	Option to specify that the vehicle must be in another lane than the referenced vehicle
side	side_left_right	no	Depending on the value the actor shall be on the right or left side of the referenced entity. How many lanes right or left of that entity is specified by the lane-parameter
offset	length	no	Lateral offset to the target lane

⚠ 注意

- side_of和same_as必须设置且仅设置一个。
- 使用side_of来设置车道时，必须同时使用lane和side。

使用lane+side_of

```
m_left: side_left_right = left
cut_in_vehicle.assign_init_position() with:
    lane(lane: 1, side_of: Ego, side: m_left)
    position(distance: 85.0m, behind: ego)
```

使用same_as

```
m_left: side_left_right = left
cut_in_vehicle.assign_init_position() with:
    lane(same_as: Ego)
    position(distance: 85.0m, behind: ego)
```

修饰器（position）

用途：设定动作主体actor所处的车道。可以修饰初始动作assign_init_position 。

参数:

表 7-39 position 参数

Parameter	Type	Mandatory	Description
distance	length	no	A target length value including a length unit. The distance is calculated using the route-based s-coordinate
ahead_of	entity	no	specified by the lane-parameter
behind	entity	no	When behind is specified, the actor must be behind the entity by the specified value

代码样例

```
cut_in_vehicle.assign_init_position() with:
    lane(lane: raletive_lane_id, side_of: ego, side: left)
    position(distance: 85.0m, behind: ego)
```

7.8.1.4 附录（Appendix）

7.8.1.4.1 Scalar Units

Units单位详解：

- **speed units**

```
SPEED_UNIT = {
    "meter_per_second": 1.0,
    "mps": 1.0,
    "kilometer_per_hour": 0.277777778,
    "kmph": 0.277777778,
    "kph": 0.277777778,
    "mile_per_hour": 0.447038889,
    "mph": 0.447038889,
    "miph": 0.447038889,
    "mmph": 0.000000278,
    "millimeter_per_hour": 0.000000278
}
```

- **acceleration units**

```
ACCELERATION_UNIT = {
    "meter_per_sec_sqr": 1.0,
    "mpsps": 1.0,
    "mpss": 1.0,
    "kilometer_per_hour_per_sec": 0.5270462769,
    "kmphps": 0.5270462769,
    "mile_per_hour_per_sec": 0.6686096686,
    "miphps": 0.6686096686
}
```

- **length units**

```
LENGTH_UNIT = {
    "nanometer": 0.000000001,
    "nm": 0.000000001,
    "micrometer": 0.000001,
    "millimeter": 0.001,
    "mm": 0.001,
    "centimeter": 0.01,
    "cm": 0.01,
    "meter": 1.0,
    "m": 1.0,
    "kilometer": 1000.0,
    "km": 1000.0,
    "inch": 0.0254,
    "feet": 0.3048,
    "mile": 1609.344,
    "mi": 1609.344
}
```

- **time units**

```
TIME_UNIT = {
    "millisecond": 0.001,
    "ms": 0.001,
    "second": 1.0,
    "sec": 1.0,
    "s": 1.0,
    "minute": 60.0,
    "min": 60.0,
    "hour": 3600.0,
```



```
    "h": 3600.0
}
```

- **angle units**

```
ANGLE_UNIT = {
    "degree": 57.295779513,
    "deg": 57.295779513,
    "radian": 1.0,
    "rad": 1.0
}
```

7.8.1.4.2 Enum Lists

side_left_right

用于修饰器**lane**。

side_left_right list

```
ENUM_SIDE_LEFT_RIGHT = ("left", "right")
```

- left：在车道的左侧
- right：在车道的右侧

distance_direction

用于触发条件**object_distance**和**point_distance**。

distance_direction list

```
ENUM_DISTANCE_DIRECTION = ("longitudinal", "lateral", "euclidianDistance")
```

- longitudinal：在x坐标中测量距离。正表示引用位于参考实体的前面。
- lateral：在y坐标中测量距离，正表示引用位于参考实体的左侧。
- euclidianDistance：欧氏距离。

distance_mode

用于触发条件**object_distance**和**point_distance**。

distance_mode list

```
ENUM_DISTANCE_MODE = ("reference_points", "bounding_boxes")
```

- reference_points：测量参考点之间的距离
- bounding_boxes：测量边界框之间的距离

lane_change_side

用于动作**change_lane**。

lane_change_side list

```
ENUM_LANE_CHANGE_SIDE = ("left", "right", "inside", "outside", "same")
```

- left：参考实体左侧的车道
- right：参考实体右侧的车道
- inside：参考实体内侧的车道

- outside：参考实体外侧的车道
- same：与参考实体相同的车道

dynamics_shape

表示给定变量随时间或距离的变化，用于动作**change_speed**，**change_lane**。

dynamics_shape list

```
ENUM_DYNAMICS_SHAPE = ("linear", "cubic", "sinusoidal", "step")
```

- linear：变化曲线是一个线性linear函数f(x) = f_0 +变化速率*x。
- cubic：变化曲线是一个三次变迁Cubical transition函数f(x)=A*x3+B*x2+C*x+D，约束梯度在开始和结束时必须为零。
- sinusoidal：变化曲线是一个正弦变迁Sinusoidal transition函数f(x)=A*正弦(x)+B，约束梯度在开始和结束时必须为零。
- step：变化曲线是一个阶段变迁Step transition函数。

catalog

目录catalog可使一些元素得以重复使用，在目录catalog中参数化类型是可维护的。

catalog list

```
ENUM_CATALOG = ("vehicle_catalog", "controller_catalog", "pedestrian_catalog", "misc_object_catalog")
```

- vehicle_catalog：场景中可复用的车辆类型列表。
- controller_catalog：场景中可复用的控制器类型列表。
- pedestrian_catalog：场景中可复用的行人类型列表。
- misc_object_catalog：场景中可复用的杂项对象类型列表。

7.8.1.4.3 Struct

struct类型，又称结构类型，是一种由简单类型（例如int、float、string类型，scalar类型，简单的struct类型等）构建的复杂类型，一般用于表示抽象的道路结构，与地图文件中的具体的道路结构建立关联。osc2.0支持的struct类型有：odr_point、position_3d、road_point、orientation_3d和pose_3d。

position_3d

- **定义**：笛卡尔（XYZ）坐标系中的**三维位置**（position）。
- **用途**：设置坐标系中的三维位置，用于构成xyz_point。
- **参数**：参数如下表

表 7-40 position_3d 参数

Parameter	Type	Mandatory	Description
x	length	yes	position on the x-axis

Parameter	Type	Mandatory	Description
y	length	yes	position on the y-axis
z	length	yes	position on the z-axis

代码样例

```
my_position: position_3d with:
    keep(it.x == 150.0m)
    keep(it.y == 200.0m)
    keep(it.z == 0.0m)
```

xyz_point

- **定义：**笛卡尔（XYZ）坐标系中的特定**位置点**（point）。
- **用途**：设置实体位置，用于构成pose_3d。
- **参数**：参数如下表

表 7-41 xyz_point 参数

Parameter	Type	Mandatory	Description
position	position_3d	yes	Position in Cartesian (XYZ) coordinates

keep创建

```
my_pos: position_3d with:
    keep(it.x == 150.0m)
    keep(it.y == 200.0m)
    keep(it.z == 0.0m)
my_xyz: xyz_point with:
    keep(it.position == my_pos)
```

create创建

```
my_xyz: xyz_point = map.create_xyz_point(x: 150.0m, y: 200.0m ,z: 0.0m)
```

road_point

- **定义：**路网s-t坐标系中的特定**位置点**（point）。
- **用途**：设置实体位置，用于构成pose_3d。
- **参数**：参数如下表

表 7-42 road_point 参数

Parameter	Type	Mandatory	Description
road_id	string	yes	identifier for the road in which this point is located
s	length	yes	Coordinate along the s-axis of the corresponding road
t	length	yes	Coordinate along the t-axis of the corresponding road

keep创建

```
my_road: road_point with
    keep(it.road_id == '1')
    keep(it.s == 5.0m)
    keep(it.t == 0.0m)
```

create创建

```
my_point: road_point = map.create_road_point(road_id: '1', s: 5.0m, t: 0.0m)
```

odr_point

- **定义：**ASAM OpenDRIVE坐标系中的**位置点**（point）。
- **用途**：设置实体位置，用于构成pose_3d。
- **参数**：参数如下表

表 7-43 odr_point 参数

Parameter	Type	Mandatory	Description
road_id	string	yes	ASAM OpenDRIVE identifier for the road
lane_id	string	yes	ASAM OpenDRIVE identifier for the lane
s	length	yes	Coordinate along the ASAM OpenDRIVE s-axis
t	length	yes	Coordinate along the ASAM OpenDRIVE t-axis, the t-coordinate is measured from the lane centerline

keep创建

```
my_odr: odr_point with:
    keep(it.road_id == '1')
```



```
    keep(it.lane_id == '-2')
    keep(it.s == 3.0m)
    keep(it.t == 0.0m)
```

create创建

```
my_odr: odr_point = map.create_odr_point(road_id: '1',lane_id:'-2',s: 3.0m, t: 0.0m)
```

orientation_3d

- **定义：**由Tait–Bryan角度的三个参数roll（横滚角，围绕x轴的角度）、pitch（俯仰角，围绕y轴的角度）和yaw（偏航角，围绕z轴的角度）定义的**三维角度**。
- **用途**：设置实体的朝向角度、用于构成pose_3d。
- **参数**：参数如下表

表 7-44 orientation_3d 参数

Parameter	Type	Mandatory	Description
roll	angle	yes	rotation angle around the x-axis
pitch	angle	yes	rotation angle around the y-axis
yaw	angle	yes	rotation angle around the z-axis

⚠ 注意

- 根据ISO 8855的定义，角度旋转的顺序是：首先进行yaw（围绕z轴）、接着pitch（围绕新y轴），最后roll（围绕新x轴）。
- 当实体的朝向与road0的方向相同时，无需设置orientation_3d。
- angle的单位一般为rad（弧度）而非degree（角度），rad = degree*pi/180，1rad约等于57.3度（详见**scalar units**中的angle units一节）。

与road 0的方向相反（相差180°）

```
m_orientation: orientation_3d with:
    keep(it.roll == 0.0rad)
    keep(it.pitch == 0.0rad)
    keep(it.yaw == 3.14rad)
```

pose_3d

- **定义：**三维空间的**复合位置**，包含**位置点**（odr_point或position_3d或road_point）和**方向**（orientation_3d）两个参数
- **用途**：设置实体的初始位置（assign_init_speed动作）、目标位置（acquire_position动作）
- **参数**：参数如下表

表 7-45 pose_3d 参数

Parameter	Type	Mandatory	Description
xyz_point	xyz_point	no	a pose in space specified in Cartesian (XYZ) coordinates
odr_point	odr_point	no	a point expressed in ASAM OpenDRIVE coordinates
road_point	road_point	no	a point on route network specified in S-T coordinates
orientation	orientation_3d	no	three-dimensional orientation

⚠ 注意

- xyz_point、odr_point和road_point必须设置且仅设置一个，用以提供位置信息。
- orientation非必选项，当不设置orientation时，对应roll、pitch、yaw均为0时的方向。

使用xyz_point、设置orientation

```
my_xyz: xyz_point = map.create_xyz_point(x: 150.0m, y: 200.0m ,z: 0.0m)
m_orientation: orientation_3d with:
    keep(it.roll == 0.0rad)
    keep(it.pitch == 0.0rad)
    keep(it.yaw == 1.57rad)
my_pose: pose_3d with:
    keep(it.xyz_point == my_xyz)
    keep(it.orientation == m_orientation)
```

使用odr_point、不设置orientation

```
my_odr: odr_point = map.create_odr_point(road_id: '1',lane_id:'-2',s: 3.0m, t: 0.0m)
my_pose: pose_3d with:
    keep(it.odr_point == my_odr)
```

使用road_point、不设置orientation

```
my_road: road_point with
    keep(it.road_id == '1')
    keep(it.s == 5.0m)
    keep(it.t == 0.0m)
my_pose: pose_3d with:
    keep(it.road_point == my_road)
```

7.8.1.4.4 ALKS 样例

根据官方提供的ALKS样例，提供了一些osc2.0的场景（osc文件）及其转化结果（xosc文件）。考虑到仿真器的支持程度，建议在转换时选择osc1.0版本。

FreeDriving

简述：主车Ego按照初始速度匀速行驶，10000s后激活controller，300s后停止场景（激活时间与场景停止时间可修改）。

osc2.0场景

```
import standard

scenario FreeDriving:
    # map
    map: map
    map.set_map_file("./ALKS_Road_Different_Curvatures.xodr")
    # parameter
    m_odr: odr_point = map.create_odr_point(road_id: '0', lane_id: '-4', s: 5.0m, t: 0.0m)
    Ego_InitPosition: pose_3d with:
        keep(it.odr_point == m_odr)
    Ego_InitSpeed_Ve0: speed = 16.66666667mps
    Duration: time = 300s
    # entity
    Ego_Name: string = "Audi_A3_2009_black"
    Ego_Controller: string = "DefaultDriver"
    Ego: vehicle with:
        keep(it.name == Ego_Name)
        keep(it.initial_bm == Ego_Controller)
    # storyboard
    do parallel(duration: Duration):
        # init
        Ego.assign_init_position(position: Ego_InitPosition)
        Ego.assign_init_speed(Ego_InitSpeed_Ve0)
        # act1: ActivateALKSControllerAct
        serial:
            wait elapsed(10000s)
            Ego.activate_controller(true, true) # args: lateral, longitudinal
```

转化结果

```
<OpenSCENARIO >
    <FileHeader author="Octopus/Simulation" date="2022-10-28T07:45:48" description="" revMajor="1"
revMinor="0"/ >
    <CatalogLocations >
        <VehicleCatalog >
            <Directory path="Distros/Current/Config/Players/Vehicles"/ >
        </VehicleCatalog >
        <ControllerCatalog >
            <Directory path="Distros/Current/Config/Players/driverCfg.xml"/ >
        </ControllerCatalog >
        <PedestrianCatalog >
            <Directory path="Distros/Current/Config/Players/Pedestrians"/ >
        </PedestrianCatalog >
        <MiscObjectCatalog >
            <Directory path="Distros/Current/Config/Players/Objects"/ >
        </MiscObjectCatalog >
    </CatalogLocations >
    <RoadNetwork >
        <LogicFile filepath="./ALKS_Road_Different_Curvatures.xodr"/ >
        <SceneGraphFile filepath=""/ >
    </RoadNetwork >
    <Entities >
        <ScenarioObject name="Ego" >
            <Vehicle name="Audi_A3_2009_black" vehicleCategory="car" >
                <BoundingBox >
                    <Center x="1.5" y="0" z="0.9"/ >
                    <Dimensions height="2.1" length="4.5" width="1.8"/ >
                </BoundingBox >
                <Performance maxAcceleration="8" maxDeceleration="8" maxSpeed="60"/ >
                <Axles >
                    <FrontAxle maxSteering="0.5" positionX="3.1" positionZ="0.3" trackWidth="1.8"
wheelDiameter="0.6"/ >
```



```
                <RearAxle maxSteering="0" positionX="0" positionZ="0.3" trackWidth="1.8"
wheelDiameter="0.6"/ >
            </Axles >
            <Properties >
               <Property name="control" value="external"/ >
            </Properties >
         </Vehicle >
         <ObjectController >
            <Controller name="DefaultDriver" >
               <Properties/ >
            </Controller >
         </ObjectController >
      </ScenarioObject >
   </Entities >
   <Storyboard >
      <Init >
         <Actions >
            <Private entityRef="Ego" >
               <PrivateAction >
                  <TeleportAction >
                     <Position >
                        <LanePosition laneId="-4" offset="0" roadId="0" s="5"/ >
                     </Position >
                  </TeleportAction >
               </PrivateAction >
               <PrivateAction >
                  <LongitudinalAction >
                     <SpeedAction >
                        <SpeedActionDynamics dynamicsDimension="time" dynamicsShape="step"
value="0"/ >
                        <SpeedActionTarget >
                           <AbsoluteTargetSpeed value="16.66666667"/ >
                        </SpeedActionTarget >
                     </SpeedAction >
                  </LongitudinalAction >
               </PrivateAction >
            </Private >
         </Actions >
      </Init >
      <Story name="FreeDrivingStory" >
         <Act name="Ego_activate_controller_7_act" >
            <ManeuverGroup maximumExecutionCount="1" name="Ego_activate_controller_7_maneuvers" >
               <Actors selectTriggeringEntities="false" >
                  <EntityRef entityRef="Ego"/ >
               </Actors >
               <Maneuver name="Ego_activate_controller_7_maneuver" >
                  <Event maximumExecutionCount="1" name="Ego_activate_controller_7_event"
priority="overwrite" >
                     <Action name="Ego_activate_controller_7" >
                        <PrivateAction >
                           <ActivateControllerAction lateral="true" longitudinal="true"/ >
                        </PrivateAction >
                     </Action >
                     <StartTrigger >
                        <ConditionGroup >
                           <Condition conditionEdge="rising" delay="0"
name="Ego_activate_controller_7_condition_1" >
                              <ByValueCondition >
                                 <SimulationTimeCondition rule="greaterThan" value="0"/ >
                              </ByValueCondition >
                           </Condition >
                        </ConditionGroup >
                     </StartTrigger >
                  </Event >
               </Maneuver >
            </ManeuverGroup >
            <StartTrigger >
               <ConditionGroup >
                  <Condition conditionEdge="rising" delay="0" name="Ego_activate_controller_7_condition"
```



```
>
                    <ByValueCondition >
                        <SimulationTimeCondition rule="greaterThan" value="10000"/ >
                    </ByValueCondition >
                </Condition >
            </ConditionGroup >
        </StartTrigger >
    </Act >
</Story >
<StopTrigger >
    <ConditionGroup >
        <Condition conditionEdge="rising" delay="0" name="story_end" >
            <ByValueCondition >
                <SimulationTimeCondition rule="greaterThan" value="300"/ >
            </ByValueCondition >
        </Condition >
    </ConditionGroup >
</StopTrigger >
</Storyboard > </OpenSCENARIO>
```

FullyBlockingTarget

简述：主车Ego按照初始速度匀速行驶，正前方495m处有一位行人，10000s后激活Ego的controller，40s后停止场景（激活时间与场景停止时间可修改）。

osc2.0场景

```
import standard

scenario FullyBlockingTarget:
    # map
    map: map
    map.set_map_file("./ALKS_Road.xodr")

    # parameter
    Ego_InitPosition_LaneId: string = '-4'
    Ego_Odr: odr_point = map.create_odr_point(road_id: '0', lane_id: Ego_InitPosition_LaneId, s: 5.0m, t: 0.0m)
    Ego_InitPosition: pose_3d with:
        keep(it.odr_point == Ego_Odr)
    TargetBlocking_Odr: odr_point = map.create_odr_point(road_id: '0', lane_id: Ego_InitPosition_LaneId, s:
500.0m, t: 0.0m)
    TargetBlocking_InitPosition: pose_3d with:
        keep(it.odr_point == TargetBlocking_Odr)
    Ego_InitSpeed_Ve0: speed = 16.66666667mps
    Duration: time = 40s

    # entity
    Ego_Name: string = "Audi_A3_2009_black"
    Ego_Controller: string = "DefaultDriver"
    Ego: vehicle with:
        keep(it.name == Ego_Name)
        keep(it.initial_bm == Ego_Controller)
    TargetBlocking: person with:
        keep(it.name == "Christian")
        keep(it.model == "male_adult")

    # storyboard
    do parallel(duration: Duration):
        # init
        Ego.assign_init_position(position: Ego_InitPosition)
        Ego.assign_init_speed(Ego_InitSpeed_Ve0)
        TargetBlocking.assign_init_position(position: TargetBlocking_InitPosition)

        # act1: ActivateALKSControllerAct
        serial:
            wait elapsed(10000s)
            Ego.activate_controller(true, true) # args: lateral, longitudinal
```

转化结果


```
<OpenSCENARIO >
   <FileHeader author="Octopus/Simulation" date="2022-10-28T07:58:37" description="" revMajor="1"
revMinor="0"/ >
   <CatalogLocations >
      <VehicleCatalog >
         <Directory path="Distros/Current/Config/Players/Vehicles"/ >
      </VehicleCatalog >
      <ControllerCatalog >
         <Directory path="Distros/Current/Config/Players/driverCfg.xml"/ >
      </ControllerCatalog >
      <PedestrianCatalog >
         <Directory path="Distros/Current/Config/Players/Pedestrians"/ >
      </PedestrianCatalog >
      <MiscObjectCatalog >
         <Directory path="Distros/Current/Config/Players/Objects"/ >
      </MiscObjectCatalog >
   </CatalogLocations >
   <RoadNetwork >
      <LogicFile filepath="./ALKS_Road.xodr"/ >
      <SceneGraphFile filepath=""/ >
   </RoadNetwork >
   <Entities >
      <ScenarioObject name="Ego" >
         <Vehicle name="Audi_A3_2009_black" vehicleCategory="car" >
            <BoundingBox >
               <Center x="1.5" y="0" z="0.9"/ >
               <Dimensions height="2.1" length="4.5" width="1.8"/ >
            </BoundingBox >
            <Performance maxAcceleration="8" maxDeceleration="8" maxSpeed="60"/ >
            <Axles >
               <FrontAxle maxSteering="0.5" positionX="3.1" positionZ="0.3" trackWidth="1.8"
wheelDiameter="0.6"/ >
               <RearAxle maxSteering="0" positionX="0" positionZ="0.3" trackWidth="1.8"
wheelDiameter="0.6"/ >
            </Axles >
            <Properties >
               <Property name="control" value="external"/ >
            </Properties >
         </Vehicle >
         <ObjectController >
            <Controller name="DefaultDriver" >
               <Properties/ >
            </Controller >
         </ObjectController >
      </ScenarioObject >
      <ScenarioObject name="TargetBlocking" >
         <Pedestrian mass="80" model="male_adult" name="Christian" pedestrianCategory="pedestrian" >
            <BoundingBox >
               <Center x="1.5" y="0" z="0.9"/ >
               <Dimensions height="2.1" length="4.5" width="1.8"/ >
            </BoundingBox >
            <Properties/ >
         </Pedestrian >
      </ScenarioObject >
   </Entities >
   <Storyboard >
      <Init >
         <Actions >
            <Private entityRef="Ego" >
               <PrivateAction >
                  <TeleportAction >
                     <Position >
                        <LanePosition laneId="-4" offset="0" roadId="0" s="5"/ >
                     </Position >
                  </TeleportAction >
               </PrivateAction >
               <PrivateAction >
                  <LongitudinalAction >
                     <SpeedAction >
```



```
                        <SpeedActionDynamics dynamicsDimension="time" dynamicsShape="step"
value="0"/ >
                        <SpeedActionTarget >
                            <AbsoluteTargetSpeed value="16.66666667"/ >
                        </SpeedActionTarget >
                    </SpeedAction >
                </LongitudinalAction >
            </PrivateAction >
        </Private >
        <Private entityRef="TargetBlocking" >
            <PrivateAction >
                <TeleportAction >
                    <Position >
                        <LanePosition laneId="-4" offset="0" roadId="0" s="500"/ >
                    </Position >
                </TeleportAction >
            </PrivateAction >
        </Private >
    </Actions >
</Init >
<Story name="FullyBlockingTargetStory" >
    <Act name="Ego_activate_controller_7_act" >
        <ManeuverGroup maximumExecutionCount="1" name="Ego_activate_controller_7_maneuvers" >
            <Actors selectTriggeringEntities="false" >
                <EntityRef entityRef="Ego"/ >
            </Actors >
            <Maneuver name="Ego_activate_controller_7_maneuver" >
                <Event maximumExecutionCount="1" name="Ego_activate_controller_7_event"
priority="overwrite" >
                    <Action name="Ego_activate_controller_7" >
                        <PrivateAction >
                            <ActivateControllerAction lateral="true" longitudinal="true"/ >
                        </PrivateAction >
                    </Action >
                    <StartTrigger >
                        <ConditionGroup >
                            <Condition conditionEdge="rising" delay="0"
name="Ego_activate_controller_7_condition_1" >
                                <ByValueCondition >
                                    <SimulationTimeCondition rule="greaterThan" value="0"/ >
                                </ByValueCondition >
                            </Condition >
                        </ConditionGroup >
                    </StartTrigger >
                </Event >
            </Maneuver >
        </ManeuverGroup >
        <StartTrigger >
            <ConditionGroup >
                <Condition conditionEdge="rising" delay="0" name="Ego_activate_controller_7_condition"
>
                    <ByValueCondition >
                        <SimulationTimeCondition rule="greaterThan" value="10000"/ >
                    </ByValueCondition >
                </Condition >
            </ConditionGroup >
        </StartTrigger >
    </Act >
</Story >
<StopTrigger >
    <ConditionGroup >
        <Condition conditionEdge="rising" delay="0" name="story_end" >
            <ByValueCondition >
                <SimulationTimeCondition rule="greaterThan" value="40"/ >
            </ByValueCondition >
        </Condition >
    </ConditionGroup >
</StopTrigger >
</Storyboard > </OpenSCENARIO>
```


PartiallyBlockingTarget

简述：主车Ego按照初始速度匀速行驶，斜前方495米处有一位行人，10000s后激活Ego的controller，40s后停止场景（激活时间与场景停止时间可修改）。

osc2.0场景

```
import standard

scenario PartiallyBlockingTarget:
    # map
    map: map
    map.set_map_file("./ALKS_Road.xodr")

    # parameter
    Ego_InitPosition_LaneId: string = '-4'
    Ego_Odr: odr_point = map.create_odr_point(road_id: '0', lane_id: Ego_InitPosition_LaneId, s: 5.0m, t: 0.0m)
    Ego_InitPosition: pose_3d with:
        keep(it.odr_point == Ego_Odr)
    TargetBlocking_InitPosition_Offset: length = -1.5m
    TargetBlocking_Odr: odr_point = map.create_odr_point(road_id: '0', lane_id: Ego_InitPosition_LaneId, s:
500.0m, t: TargetBlocking_InitPosition_Offset)
    TargetBlocking_InitPosition: pose_3d with:
        keep(it.odr_point == TargetBlocking_Odr)
    Ego_InitSpeed_Ve0: speed = 16.66666667mps
    Duration: time = 40s

    # entity
    Ego_Name: string = "Audi_A3_2009_black"
    Ego_Controller: string = "DefaultDriver"
    Ego: vehicle with:
        keep(it.name == Ego_Name)
        keep(it.initial_bm == Ego_Controller)
    TargetBlocking: person with:
        keep(it.name == "Christian")
        keep(it.model == "male_adult")

    # storyboard
    do parallel(duration: Duration):
        # init
        Ego.assign_init_position(position: Ego_InitPosition)
        Ego.assign_init_speed(Ego_InitSpeed_Ve0)
        TargetBlocking.assign_init_position(position: TargetBlocking_InitPosition)

        # act1: ActivateALKSControllerAct
        serial:
            wait elapsed(10000s)
            Ego.activate_controller(true, true) # args: lateral, longitudinal
```

转化结果

```
<OpenSCENARIO >
    <FileHeader author="Octopus/Simulation" date="2022-10-28T08:09:41" description="" revMajor="1"
revMinor="0"/ >
    <CatalogLocations >
        <VehicleCatalog >
            <Directory path="Distros/Current/Config/Players/Vehicles"/ >
        </VehicleCatalog >
        <ControllerCatalog >
            <Directory path="Distros/Current/Config/Players/driverCfg.xml"/ >
        </ControllerCatalog >
        <PedestrianCatalog >
            <Directory path="Distros/Current/Config/Players/Pedestrians"/ >
        </PedestrianCatalog >
        <MiscObjectCatalog >
            <Directory path="Distros/Current/Config/Players/Objects"/ >
        </MiscObjectCatalog >
    </CatalogLocations >
    <RoadNetwork >
```



```
    <LogicFile filepath="./ALKS_Road.xodr"/ >
    <SceneGraphFile filepath=""/ >
  </RoadNetwork >
  <Entities >
    <ScenarioObject name="Ego" >
      <Vehicle name="Audi_A3_2009_black" vehicleCategory="car" >
        <BoundingBox >
          <Center x="1.5" y="0" z="0.9"/ >
          <Dimensions height="2.1" length="4.5" width="1.8"/ >
        </BoundingBox >
        <Performance maxAcceleration="8" maxDeceleration="8" maxSpeed="60"/ >
        <Axles >
          <FrontAxle maxSteering="0.5" positionX="3.1" positionZ="0.3" trackWidth="1.8"
wheelDiameter="0.6"/ >
          <RearAxle maxSteering="0" positionX="0" positionZ="0.3" trackWidth="1.8"
wheelDiameter="0.6"/ >
        </Axles >
        <Properties >
          <Property name="control" value="external"/ >
        </Properties >
      </Vehicle >
      <ObjectController >
        <Controller name="DefaultDriver" >
          <Properties/ >
        </Controller >
      </ObjectController >
    </ScenarioObject >
    <ScenarioObject name="TargetBlocking" >
      <Pedestrian mass="80" model="male_adult" name="Christian" pedestrianCategory="pedestrian" >
        <BoundingBox >
          <Center x="1.5" y="0" z="0.9"/ >
          <Dimensions height="2.1" length="4.5" width="1.8"/ >
        </BoundingBox >
        <Properties/ >
      </Pedestrian >
    </ScenarioObject >
  </Entities >
  <Storyboard >
    <Init >
      <Actions >
        <Private entityRef="Ego" >
          <PrivateAction >
            <TeleportAction >
              <Position >
                <LanePosition laneId="-4" offset="0" roadId="0" s="5"/ >
              </Position >
            </TeleportAction >
          </PrivateAction >
          <PrivateAction >
            <LongitudinalAction >
              <SpeedAction >
                <SpeedActionDynamics dynamicsDimension="time" dynamicsShape="step"
value="0"/ >
                <SpeedActionTarget >
                  <AbsoluteTargetSpeed value="16.67"/ >
                </SpeedActionTarget >
              </SpeedAction >
            </LongitudinalAction >
          </PrivateAction >
        </Private >
        <Private entityRef="TargetBlocking" >
          <PrivateAction >
            <TeleportAction >
              <Position >
                <LanePosition laneId="-4" offset="-1.5" roadId="0" s="500"/ >
              </Position >
            </TeleportAction >
          </PrivateAction >
        </Private >
```



```
        </Actions >
      </Init >
      <Story name="PartiallyBlockingTargetStory" >
        <Act name="Ego_activate_controller_act" >
          <ManeuverGroup maximumExecutionCount="1" name="Ego_activate_controller_maneuvers" >
            <Actors selectTriggeringEntities="false" >
              <EntityRef entityRef="Ego"/ >
            </Actors >
            <Maneuver name="Ego_activate_controller_maneuver" >
              <Event maximumExecutionCount="1" name="Ego_activate_controller_event"
priority="overwrite" >
                <Action name="Ego_activate_controller" >
                  <PrivateAction >
                    <ActivateControllerAction lateral="true" longitudinal="true"/ >
                  </PrivateAction >
                </Action >
                <StartTrigger >
                  <ConditionGroup >
                    <Condition conditionEdge="rising" delay="0"
name="Ego_activate_controller_condition_1" >
                      <ByValueCondition >
                        <SimulationTimeCondition rule="greaterThan" value="0"/ >
                      </ByValueCondition >
                    </Condition >
                  </ConditionGroup >
                </StartTrigger >
              </Event >
            </Maneuver >
          </ManeuverGroup >
          <StartTrigger >
            <ConditionGroup >
              <Condition conditionEdge="rising" delay="0" name="Ego_activate_controller_condition" >
                <ByValueCondition >
                  <SimulationTimeCondition rule="greaterThan" value="10000"/ >
                </ByValueCondition >
              </Condition >
            </ConditionGroup >
          </StartTrigger >
        </Act >
      </Story >
      <StopTrigger >
        <ConditionGroup >
          <Condition conditionEdge="rising" delay="0" name="story_end" >
            <ByValueCondition >
              <SimulationTimeCondition rule="greaterThan" value="40"/ >
            </ByValueCondition >
          </Condition >
        </ConditionGroup >
      </StopTrigger >
    </Storyboard > </OpenSCENARIO>
```

FollowLeadVehicleEmergencyBrake

简述：主车Ego和正前方33.33m的头车lead_vehicle按照相同的初始速度匀速行驶，前车突然刹车，10000s后激活Ego的controller，40s后停止场景（激活时间与场景停止时间可修改）。

osc2.0场景

```
import standard

scenario FollowLeadVehicleEmergencyBrake:
    # map
    map: map
    map.set_map_file("./ALKS_Road.xodr")

    # parameter
    Ego_InitPosition_LaneId: string = '-4'
```



```
    m_odr: odr_point = map.create_odr_point(road_id: '0', lane_id: Ego_InitPosition_LaneId, s: 5.0m, t: 0.0m)
    Ego_InitPosition: pose_3d with:
        keep(it.odr_point == m_odr)
    Ego_InitSpeed_Ve0: speed = 16.66666667mps
    LeadVehicle_InitDistance_dx0: length = 33.33333333m
    LeadVehicle_BrakeRate_Gx_max: acceleration = 6.0mpss
    m_rate_profile: dynamics_shape = linear
    Duration: time = 40s

    # entity
    Ego_Name: string = "Audi_A3_2009_black"
    LeadVehicle_Name: string = "Audi_A3_2009_red"
    Ego_Controller: string = "DefaultDriver"
    Ego: vehicle with:
        keep(it.name == Ego_Name)
        keep(it.initial_bm == Ego_Controller)
    lead_vehicle: vehicle with:
        keep(it.name == LeadVehicle_Name)

    # storyboard
    do parallel(duration: Duration):
        # init
        Ego.assign_init_position(position: Ego_InitPosition)
        Ego.assign_init_speed(Ego_InitSpeed_Ve0)
        lead_vehicle.assign_init_position() with:
            lane(same_as: Ego)
            position(LeadVehicle_InitDistance_dx0, ahead_of: Ego)
        lead_vehicle.assign_init_speed(Ego_InitSpeed_Ve0)

        # act1: ActivateALKSControllerAct
        serial:
            wait elapsed(10000s)
            Ego.activate_controller(true, true) # args: lateral, longitudinal
        # act2: BrakeAct
        # VaryingSpeedEvent
        serial:
            wait elapsed(10s)
            lead_vehicle.change_speed(target: 0.0mps, rate_profile: m_rate_profile, rate_peak:
LeadVehicle_BrakeRate_Gx_max)
```

转化结果

```
<OpenSCENARIO >
    <FileHeader author="Octopus/Simulation" date="2022-10-28T08:12:34" description="" revMajor="1"
revMinor="0"/ >
    <CatalogLocations >
        <VehicleCatalog >
            <Directory path="Distros/Current/Config/Players/Vehicles"/ >
        </VehicleCatalog >
        <ControllerCatalog >
            <Directory path="Distros/Current/Config/Players/driverCfg.xml"/ >
        </ControllerCatalog >
        <PedestrianCatalog >
            <Directory path="Distros/Current/Config/Players/Pedestrians"/ >
        </PedestrianCatalog >
        <MiscObjectCatalog >
            <Directory path="Distros/Current/Config/Players/Objects"/ >
        </MiscObjectCatalog >
    </CatalogLocations >
    <RoadNetwork >
        <LogicFile filepath="./ALKS_Road.xodr"/ >
        <SceneGraphFile filepath=""/ >
    </RoadNetwork >
    <Entities >
        <ScenarioObject name="Ego" >
            <Vehicle name="Audi_A3_2009_black" vehicleCategory="car" >
                <BoundingBox >
                    <Center x="1.5" y="0" z="0.9"/ >
                    <Dimensions height="2.1" length="4.5" width="1.8"/ >
                </BoundingBox >
```



```
            <Performance maxAcceleration="8" maxDeceleration="8" maxSpeed="60"/ >
            <Axles >
               <FrontAxle maxSteering="0.5" positionX="3.1" positionZ="0.3" trackWidth="1.8"
wheelDiameter="0.6"/ >
               <RearAxle maxSteering="0" positionX="0" positionZ="0.3" trackWidth="1.8"
wheelDiameter="0.6"/ >
            </Axles >
            <Properties >
               <Property name="control" value="external"/ >
            </Properties >
         </Vehicle >
         <ObjectController >
            <Controller name="DefaultDriver" >
               <Properties/ >
            </Controller >
         </ObjectController >
      </ScenarioObject >
      <ScenarioObject name="lead_vehicle" >
         <Vehicle name="Audi_A3_2009_red" vehicleCategory="car" >
            <BoundingBox >
               <Center x="1.5" y="0" z="0.9"/ >
               <Dimensions height="2.1" length="4.5" width="1.8"/ >
            </BoundingBox >
            <Performance maxAcceleration="8" maxDeceleration="8" maxSpeed="60"/ >
            <Axles >
               <FrontAxle maxSteering="0.5" positionX="3.1" positionZ="0.3" trackWidth="1.8"
wheelDiameter="0.6"/ >
               <RearAxle maxSteering="0" positionX="0" positionZ="0.3" trackWidth="1.8"
wheelDiameter="0.6"/ >
            </Axles >
            <Properties/ >
         </Vehicle >
      </ScenarioObject >
   </Entities >
   <Storyboard >
      <Init >
         <Actions >
            <Private entityRef="Ego" >
               <PrivateAction >
                  <TeleportAction >
                     <Position >
                        <LanePosition laneId="-4" offset="0" roadId="0" s="5"/ >
                     </Position >
                  </TeleportAction >
               </PrivateAction >
               <PrivateAction >
                  <LongitudinalAction >
                     <SpeedAction >
                        <SpeedActionDynamics dynamicsDimension="time" dynamicsShape="step"
value="0"/ >
                        <SpeedActionTarget >
                           <AbsoluteTargetSpeed value="16.66666667"/ >
                        </SpeedActionTarget >
                     </SpeedAction >
                  </LongitudinalAction >
               </PrivateAction >
            </Private >
            <Private entityRef="lead_vehicle" >
               <PrivateAction >
                  <TeleportAction >
                     <Position >
                        <RelativeLanePosition dLane="0" ds="33.33333333" entityRef="Ego"/ >
                     </Position >
                  </TeleportAction >
               </PrivateAction >
               <PrivateAction >
                  <LongitudinalAction >
                     <SpeedAction >
                        <SpeedActionDynamics dynamicsDimension="time" dynamicsShape="step"
```



```
value="0"/ >
                        <SpeedActionTarget >
                            <AbsoluteTargetSpeed value="16.66666667"/ >
                        </SpeedActionTarget >
                    </SpeedAction >
                </LongitudinalAction >
            </PrivateAction >
        </Private >
    </Actions >
</Init >
<Story name="FollowLeadVehicleEmergencyBrakeStory" >
    <Act name="Ego_activate_controller_7_act" >
        <ManeuverGroup maximumExecutionCount="1" name="Ego_activate_controller_7_maneuvers" >
            <Actors selectTriggeringEntities="false" >
                <EntityRef entityRef="Ego"/ >
            </Actors >
            <Maneuver name="Ego_activate_controller_7_maneuver" >
                <Event maximumExecutionCount="1" name="Ego_activate_controller_7_event"
priority="overwrite" >
                    <Action name="Ego_activate_controller_7" >
                        <PrivateAction >
                            <ActivateControllerAction lateral="true" longitudinal="true"/ >
                        </PrivateAction >
                    </Action >
                    <StartTrigger >
                        <ConditionGroup >
                            <Condition conditionEdge="rising" delay="0"
name="Ego_activate_controller_7_condition_1" >
                                <ByValueCondition >
                                    <SimulationTimeCondition rule="greaterThan" value="0"/ >
                                </ByValueCondition >
                            </Condition >
                        </ConditionGroup >
                    </StartTrigger >
                </Event >
            </Maneuver >
        </ManeuverGroup >
        <StartTrigger >
            <ConditionGroup >
                <Condition conditionEdge="rising" delay="0" name="Ego_activate_controller_7_condition"
>
                    <ByValueCondition >
                        <SimulationTimeCondition rule="greaterThan" value="10000"/ >
                    </ByValueCondition >
                </Condition >
            </ConditionGroup >
        </StartTrigger >
    </Act >
    <Act name="lead_vehicle_change_speed_1_act" >
        <ManeuverGroup maximumExecutionCount="1"
name="lead_vehicle_change_speed_1_maneuvers" >
            <Actors selectTriggeringEntities="false" >
                <EntityRef entityRef="lead_vehicle"/ >
            </Actors >
            <Maneuver name="lead_vehicle_change_speed_1_maneuver" >
                <Event maximumExecutionCount="1" name="lead_vehicle_change_speed_1_event"
priority="overwrite" >
                    <Action name="lead_vehicle_change_speed_1" >
                        <PrivateAction >
                            <LongitudinalAction >
                                <SpeedAction >
                                    <SpeedActionDynamics dynamicsDimension="rate" dynamicsShape="linear"
value="6"/ >
                                    <SpeedActionTarget >
                                        <AbsoluteTargetSpeed value="0"/ >
                                    </SpeedActionTarget >
                                </SpeedAction >
                            </LongitudinalAction >
                        </PrivateAction >
```



```
                    </Action >
                    <StartTrigger >
                        <ConditionGroup >
                            <Condition conditionEdge="rising" delay="0"
name="lead_vehicle_change_speed_1_condition_1" >
                                <ByValueCondition >
                                    <SimulationTimeCondition rule="greaterThan" value="0"/ >
                                </ByValueCondition >
                            </Condition >
                        </ConditionGroup >
                    </StartTrigger >
                </Event >
            </Maneuver >
        </ManeuverGroup >
        <StartTrigger >
            <ConditionGroup >
                <Condition conditionEdge="rising" delay="0"
name="lead_vehicle_change_speed_1_condition" >
                    <ByValueCondition >
                        <SimulationTimeCondition rule="greaterThan" value="10"/ >
                    </ByValueCondition >
                </Condition >
            </ConditionGroup >
        </StartTrigger >
    </Act >
  </Story >
  <StopTrigger >
    <ConditionGroup >
      <Condition conditionEdge="rising" delay="0" name="story_end" >
        <ByValueCondition >
          <SimulationTimeCondition rule="greaterThan" value="40"/ >
        </ByValueCondition >
      </Condition >
    </ConditionGroup >
  </StopTrigger >
</Storyboard > </OpenSCENARIO>
```

CutInNoCollision

简述：主车Ego按照初始速度16.667mps匀速行驶，前方85m右1车道的切入车辆cut_in_vehicle以慢于Ego5.55mps的速度行驶，当两车距离小于30m时，cut_in_vehicle开始以15mps为目标加速（加速动态受到加速度值CutInVehicle_SpeedChange_RatePeak的影响，此处ALKS设置的默认加速度为0.0mps，可以根据需要更改），同时变道切入Ego所在车道，10000s后激活Ego的controller，40s后停止场景（激活时间与场景停止时间可修改）。

osc2.0场景

```
import standard

scenario CutInNoCollision:
  # map
  map: map
  map.set_map_file('./ALKS_Road.xodr')

  # parameter
  m_odr: odr_point = map.create_odr_point(road_id: '0', lane_id: '-4', s: 5.0m, t: 0.0m)
  Ego_InitPosition: pose_3d with:
    keep(it.odr_point == m_odr)
  Ego_InitSpeed_Ve0: speed = 16.66666667mps
  CutInVehicle_InitPosition_RelativeLaneId: int = 1  # -1 for right, 1 for left
  CutInVehicle_HeadwayDistanceTrigger_dx0: length = 30.0m
  CutInVehicle_LaneChange_MaxLateralVelocity_Vy: speed = 2.0mps
  CutInVehicle_SpeedChange_RatePeak: acceleration = 0.0mpss
  CutInVehicle_Acceleration_Target: speed = 15.0mps
  m_right: side_left_right = right
  m_direction: distance_direction = longitudinal
```



```
m_mode: distance_mode = bounding_boxes
m_side: lane_change_side = same
m_rate_profile_lane: dynamics_shape = linear
m_rate_profile_speed: dynamics_shape = linear
Duration: time = 40s

# entity
Ego_Name: string = "Audi_A3_2009_black"
CutInVehicle_Name: string = "Audi_A3_2009_red"
Ego_Controller: string = "DefaultDriver"
Ego: vehicle with:
    keep(it.name == Ego_Name)
    keep(it.initial_bm == Ego_Controller)
cut_in_vehicle: vehicle with:
    keep(it.name == CutInVehicle_Name)

# storyboard
do parallel(duration: Duration):
    # init
    Ego.assign_init_position(position: Ego_InitPosition)
    Ego.assign_init_speed(target: Ego_InitSpeed_Ve0)
    cut_in_vehicle.assign_init_position() with:
        lane(lane: CutInVehicle_InitPosition_RelativeLaneId, side_of: Ego, side: m_right)
        position(distance: 85.0m, ahead_of: Ego)
    cut_in_vehicle.assign_init_speed(11.1111mps)

    # story
    serial:
        # act1: ActivateALKSControllerAct
        wait elapsed(10000s)
        Ego.activate_controller(lateral: true, longitudinal: true)
    serial:
        # act2: CutInAct
        wait cut_in_vehicle.object_distance(reference: Ego, direction: m_direction, mode: m_mode) <
CutInVehicle_HeadwayDistanceTrigger_dx0
        parallel:
            cut_in_vehicle.change_lane(side: m_side, reference: Ego, rate_peak:
CutInVehicle_LaneChange_MaxLateralVelocity_Vy, rate_profile: m_rate_profile_lane)
            cut_in_vehicle.change_speed(target: CutInVehicle_Acceleration_Target, rate_peak:
CutInVehicle_SpeedChange_RatePeak, rate_profile: m_rate_profile_speed)
```

转化结果

```
<OpenSCENARIO >
  <FileHeader author="Octopus/Simulation" date="2022-10-28T08:55:16" description="" revMajor="1"
revMinor="0"/ >
  <CatalogLocations >
    <VehicleCatalog >
      <Directory path="Distros/Current/Config/Players/Vehicles"/ >
    </VehicleCatalog >
    <ControllerCatalog >
      <Directory path="Distros/Current/Config/Players/driverCfg.xml"/ >
    </ControllerCatalog >
    <PedestrianCatalog >
      <Directory path="Distros/Current/Config/Players/Pedestrians"/ >
    </PedestrianCatalog >
    <MiscObjectCatalog >
      <Directory path="Distros/Current/Config/Players/Objects"/ >
    </MiscObjectCatalog >
  </CatalogLocations >
  <RoadNetwork >
    <LogicFile filepath="./ALKS_Road.xodr"/ >
    <SceneGraphFile filepath=""/ >
  </RoadNetwork >
  <Entities >
    <ScenarioObject name="Ego" >
      <Vehicle name="Audi_A3_2009_black" vehicleCategory="car" >
        <BoundingBox >
          <Center x="1.5" y="0" z="0.9"/ >
          <Dimensions height="2.1" length="4.5" width="1.8"/ >
```



```
            </BoundingBox >
            <Performance maxAcceleration="8" maxDeceleration="8" maxSpeed="60"/ >
            <Axles >
               <FrontAxle maxSteering="0.5" positionX="3.1" positionZ="0.3" trackWidth="1.8"
wheelDiameter="0.6"/ >
               <RearAxle maxSteering="0" positionX="0" positionZ="0.3" trackWidth="1.8"
wheelDiameter="0.6"/ >
            </Axles >
            <Properties >
               <Property name="control" value="external"/ >
            </Properties >
         </Vehicle >
         <ObjectController >
            <Controller name="DefaultDriver" >
               <Properties/ >
            </Controller >
         </ObjectController >
      </ScenarioObject >
      <ScenarioObject name="cut_in_vehicle" >
         <Vehicle name="Audi_A3_2009_red" vehicleCategory="car" >
            <BoundingBox >
               <Center x="1.5" y="0" z="0.9"/ >
               <Dimensions height="2.1" length="4.5" width="1.8"/ >
            </BoundingBox >
            <Performance maxAcceleration="8" maxDeceleration="8" maxSpeed="60"/ >
            <Axles >
               <FrontAxle maxSteering="0.5" positionX="3.1" positionZ="0.3" trackWidth="1.8"
wheelDiameter="0.6"/ >
               <RearAxle maxSteering="0" positionX="0" positionZ="0.3" trackWidth="1.8"
wheelDiameter="0.6"/ >
            </Axles >
            <Properties/ >
         </Vehicle >
      </ScenarioObject >
   </Entities >
   <Storyboard >
      <Init >
         <Actions >
            <Private entityRef="Ego" >
               <PrivateAction >
                  <TeleportAction >
                     <Position >
                        <LanePosition laneId="-4" offset="0" roadId="0" s="5"/ >
                     </Position >
                  </TeleportAction >
               </PrivateAction >
               <PrivateAction >
                  <LongitudinalAction >
                     <SpeedAction >
                        <SpeedActionDynamics dynamicsDimension="time" dynamicsShape="step"
value="0"/ >
                        <SpeedActionTarget >
                           <AbsoluteTargetSpeed value="16.67"/ >
                        </SpeedActionTarget >
                     </SpeedAction >
                  </LongitudinalAction >
               </PrivateAction >
            </Private >
            <Private entityRef="cut_in_vehicle" >
               <PrivateAction >
                  <TeleportAction >
                     <Position >
                        <RelativeLanePosition dLane="-1" ds="85" entityRef="Ego"/ >
                     </Position >
                  </TeleportAction >
               </PrivateAction >
               <PrivateAction >
                  <LongitudinalAction >
                     <SpeedAction >
```



```
                        <SpeedActionDynamics dynamicsDimension="time" dynamicsShape="step"
value="0"/ >
                        <SpeedActionTarget >
                            <AbsoluteTargetSpeed value="11.11"/ >
                        </SpeedActionTarget >
                    </SpeedAction >
                </LongitudinalAction >
            </PrivateAction >
        </Private >
    </Actions >
</Init >
<Story name="CutInNoCollisionStory" >
    <Act name="Ego_activate_controller_act" >
        <ManeuverGroup maximumExecutionCount="1" name="Ego_activate_controller_maneuvers" >
            <Actors selectTriggeringEntities="false" >
                <EntityRef entityRef="Ego"/ >
            </Actors >
            <Maneuver name="Ego_activate_controller_maneuver" >
                <Event maximumExecutionCount="1" name="Ego_activate_controller_event"
priority="overwrite" >
                    <Action name="Ego_activate_controller" >
                        <PrivateAction >
                            <ActivateControllerAction lateral="true" longitudinal="true"/ >
                        </PrivateAction >
                    </Action >
                    <StartTrigger >
                        <ConditionGroup >
                            <Condition conditionEdge="rising" delay="0"
name="Ego_activate_controller_condition_1" >
                                <ByValueCondition >
                                    <SimulationTimeCondition rule="greaterThan" value="0"/ >
                                </ByValueCondition >
                            </Condition >
                        </ConditionGroup >
                    </StartTrigger >
                </Event >
            </Maneuver >
        </ManeuverGroup >
        <StartTrigger >
            <ConditionGroup >
                <Condition conditionEdge="rising" delay="0" name="Ego_activate_controller_condition" >
                    <ByValueCondition >
                        <SimulationTimeCondition rule="greaterThan" value="10000"/ >
                    </ByValueCondition >
                </Condition >
            </ConditionGroup >
        </StartTrigger >
    </Act >
    <Act name="cut_in_vehicle_change_lane_act" >
        <ManeuverGroup maximumExecutionCount="1" name="cut_in_vehicle_change_lane_maneuvers"
>
            <Actors selectTriggeringEntities="false" >
                <EntityRef entityRef="cut_in_vehicle"/ >
            </Actors >
            <Maneuver name="cut_in_vehicle_change_lane_maneuver" >
                <Event maximumExecutionCount="1" name="cut_in_vehicle_change_lane_event"
priority="overwrite" >
                    <Action name="cut_in_vehicle_change_lane" >
                        <PrivateAction >
                            <LateralAction >
                                <LaneChangeAction >
                                    <LaneChangeActionDynamics dynamicsDimension="rate"
dynamicsShape="linear" value="2"/ >
                                    <LaneChangeTarget >
                                        <RelativeTargetLane entityRef="Ego" value="0"/ >
                                    </LaneChangeTarget >
                                </LaneChangeAction >
                            </LateralAction >
                        </PrivateAction >
```



```
                    </Action >
                    <StartTrigger >
                        <ConditionGroup >
                            <Condition conditionEdge="rising" delay="0"
name="cut_in_vehicle_change_lane_condition" >
                                <ByEntityCondition >
                                    <TriggeringEntities triggeringEntitiesRule="any" >
                                        <EntityRef entityRef="cut_in_vehicle"/ >
                                    </TriggeringEntities >
                                    <EntityCondition >
                                        <DistanceCondition alongRoute="true" freespace="true" rule="lessThan"
value="30" >
                                            <Position >
                                                <RelativeObjectPosition dx="0" dy="0" dz="0" entityRef="Ego" >
                                                    <Orientation h="0" p="0" r="0" type="absolute"/ >
                                                </RelativeObjectPosition >
                                            </Position >
                                        </DistanceCondition >
                                    </EntityCondition >
                                </ByEntityCondition >
                            </Condition >
                        </ConditionGroup >
                    </StartTrigger >
                </Event >
            </Maneuver >
        </ManeuverGroup >
        <StartTrigger >
            <ConditionGroup >
                <Condition conditionEdge="rising" delay="0"
name="cut_in_vehicle_change_lane_condition_1" >
                    <ByValueCondition >
                        <SimulationTimeCondition rule="greaterThan" value="0"/ >
                    </ByValueCondition >
                </Condition >
            </ConditionGroup >
        </StartTrigger >
    </Act >
    <Act name="cut_in_vehicle_change_speed_act" >
        <ManeuverGroup maximumExecutionCount="1"
name="cut_in_vehicle_change_speed_maneuvers" >
            <Actors selectTriggeringEntities="false" >
                <EntityRef entityRef="cut_in_vehicle"/ >
            </Actors >
            <Maneuver name="cut_in_vehicle_change_speed_maneuver" >
                <Event maximumExecutionCount="1" name="cut_in_vehicle_change_speed_event"
priority="overwrite" >
                    <Action name="cut_in_vehicle_change_speed" >
                        <PrivateAction >
                            <LongitudinalAction >
                                <SpeedAction >
                                    <SpeedActionDynamics dynamicsDimension="rate" dynamicsShape="linear"
value="0"/ >
                                    <SpeedActionTarget >
                                        <AbsoluteTargetSpeed value="15"/ >
                                    </SpeedActionTarget >
                                </SpeedAction >
                            </LongitudinalAction >
                        </PrivateAction >
                    </Action >
                    <StartTrigger >
                        <ConditionGroup >
                            <Condition conditionEdge="rising" delay="0"
name="cut_in_vehicle_change_speed_condition" >
                                <ByEntityCondition >
                                    <TriggeringEntities triggeringEntitiesRule="any" >
                                        <EntityRef entityRef="cut_in_vehicle"/ >
                                    </TriggeringEntities >
                                    <EntityCondition >
                                        <DistanceCondition alongRoute="true" freespace="true" rule="lessThan"
```



```
value="30" >
                                        <Position >
                                            <RelativeObjectPosition dx="0" dy="0" dz="0" entityRef="Ego" >
                                                <Orientation h="0" p="0" r="0" type="absolute"/ >
                                            </RelativeObjectPosition >
                                        </Position >
                                    </DistanceCondition >
                                </EntityCondition >
                            </ByEntityCondition >
                        </Condition >
                    </ConditionGroup >
                </StartTrigger >
            </Event >
        </Maneuver >
    </ManeuverGroup >
    <StartTrigger >
        <ConditionGroup >
            <Condition conditionEdge="rising" delay="0"
name="cut_in_vehicle_change_speed_condition_1" >
                <ByValueCondition >
                    <SimulationTimeCondition rule="greaterThan" value="0"/ >
                </ByValueCondition >
            </Condition >
        </ConditionGroup >
    </StartTrigger >
</Act >
</Story >
<StopTrigger >
    <ConditionGroup >
        <Condition conditionEdge="rising" delay="0" name="story_end" >
            <ByValueCondition >
                <SimulationTimeCondition rule="greaterThan" value="40"/ >
            </ByValueCondition >
        </Condition >
    </ConditionGroup >
</StopTrigger >
</Storyboard > </OpenSCENARIO>
```

CutInUnavoidableCollision

简述：主车Ego按照初始速度16.667mps匀速行驶，前方55m右1车道的切入车辆cut_in_vehicle以慢于Ego5.55mps的速度行驶，当两车距离小于10m时，cut_in_vehicle开始以15mps为目标加速，同时变道切入Ego所在车道，10000s后激活Ego的controller，40s后停止场景（激活时间与场景停止时间可修改）。

osc2.0场景

```
import standard

scenario CutInUnavoidableCollision:
    # map
    map: map
    map.set_map_file("./ALKS_Road.xodr")

    # parameter
    m_odr: odr_point = map.create_odr_point(road_id: '0', lane_id: '-4', s: 5.0m, t: 0.0m)
    Ego_InitPosition: pose_3d with:
        keep(it.odr_point == m_odr)
    Ego_InitSpeed_Ve0: speed = 16.66666667mps
    CutInVehicle_InitPosition_RelativeLaneId: int = 1  # -1 for right, 1 for left
    CutInVehicle_HeadwayDistanceTrigger_dx0: length = 10.0m
    CutInVehicle_LaneChange_MaxLateralVelocity_Vy: speed = 3.0mps
    CutInVehicle_SpeedChange_RatePeak: acceleration  = 0.0mpss
    CutInVehicle_Acceleration_Target: speed = 15.0mps
    CutInVehicle_SpeedChange_RateProfile: dynamics_shape = linear
    m_right: side_left_right = right
    m_direction: distance_direction = longitudinal
    m_mode: distance_mode = bounding_boxes
```



```
m_side: lane_change_side = same
m_rate_profile_lane: dynamics_shape = linear
m_rate_profile_speed: dynamics_shape = linear
Duration: time = 40s

# entity
Ego_Name: string = "Audi_A3_2009_black"
CutInVehicle_Name: string = "Audi_A3_2009_red"
Ego_Controller: string = "DefaultDriver"
Ego: vehicle with:
    keep(it.name == Ego_Name)
    keep(it.initial_bm == Ego_Controller)
cut_in_vehicle: vehicle with:
    keep(it.name == CutInVehicle_Name)

# storyboard
do parallel(duration: Duration):
    # init
    Ego.assign_init_position(position: Ego_InitPosition)
    Ego.assign_init_speed(Ego_InitSpeed_Ve0)
    cut_in_vehicle.assign_init_position() with:
        lane(lane: CutInVehicle_InitPosition_RelativeLaneId, side_of: Ego, side: m_right)
        position(distance: 55.0m, ahead_of: Ego)
    cut_in_vehicle.assign_init_speed(11.1111mps)

    # story
    serial:
        # act1: ActivateALKSControllerAct
        wait elapsed(10000s)
        Ego.activate_controller(true, true)  # lateral, longitudinal   # act 1: ActivateALKSControllerStory
    serial:
        # act2: CutInAct
        wait cut_in_vehicle.object_distance(reference: Ego, direction: m_direction, mode: m_mode) <
CutInVehicle_HeadwayDistanceTrigger_dx0
        parallel:
            cut_in_vehicle.change_lane(side: m_side, reference: Ego, rate_peak:
CutInVehicle_LaneChange_MaxLateralVelocity_Vy, rate_profile: m_rate_profile_lane)
            cut_in_vehicle.change_speed(target: CutInVehicle_Acceleration_Target, rate_peak:
CutInVehicle_SpeedChange_RatePeak, rate_profile: m_rate_profile_speed)
```

转化结果

```
<OpenSCENARIO >
  <FileHeader author="Octopus/Simulation" date="2022-10-28T09:18:33" description="" revMajor="1"
revMinor="0"/ >
  <CatalogLocations >
    <VehicleCatalog >
      <Directory path="Distros/Current/Config/Players/Vehicles"/ >
    </VehicleCatalog >
    <ControllerCatalog >
      <Directory path="Distros/Current/Config/Players/driverCfg.xml"/ >
    </ControllerCatalog >
    <PedestrianCatalog >
      <Directory path="Distros/Current/Config/Players/Pedestrians"/ >
    </PedestrianCatalog >
    <MiscObjectCatalog >
      <Directory path="Distros/Current/Config/Players/Objects"/ >
    </MiscObjectCatalog >
  </CatalogLocations >
  <RoadNetwork >
    <LogicFile filepath="./ALKS_Road.xodr"/ >
    <SceneGraphFile filepath=""/ >
  </RoadNetwork >
  <Entities >
    <ScenarioObject name="Ego" >
      <Vehicle name="Audi_A3_2009_black" vehicleCategory="car" >
        <BoundingBox >
          <Center x="1.5" y="0" z="0.9"/ >
          <Dimensions height="2.1" length="4.5" width="1.8"/ >
        </BoundingBox >
```



```
            <Performance maxAcceleration="8" maxDeceleration="8" maxSpeed="60"/ >
            <Axles >
               <FrontAxle maxSteering="0.5" positionX="3.1" positionZ="0.3" trackWidth="1.8"
wheelDiameter="0.6"/ >
               <RearAxle maxSteering="0" positionX="0" positionZ="0.3" trackWidth="1.8"
wheelDiameter="0.6"/ >
            </Axles >
            <Properties >
               <Property name="control" value="external"/ >
            </Properties >
         </Vehicle >
         <ObjectController >
            <Controller name="DefaultDriver" >
               <Properties/ >
            </Controller >
         </ObjectController >
      </ScenarioObject >
      <ScenarioObject name="cut_in_vehicle" >
         <Vehicle name="Audi_A3_2009_red" vehicleCategory="car" >
            <BoundingBox >
               <Center x="1.5" y="0" z="0.9"/ >
               <Dimensions height="2.1" length="4.5" width="1.8"/ >
            </BoundingBox >
            <Performance maxAcceleration="8" maxDeceleration="8" maxSpeed="60"/ >
            <Axles >
               <FrontAxle maxSteering="0.5" positionX="3.1" positionZ="0.3" trackWidth="1.8"
wheelDiameter="0.6"/ >
               <RearAxle maxSteering="0" positionX="0" positionZ="0.3" trackWidth="1.8"
wheelDiameter="0.6"/ >
            </Axles >
            <Properties/ >
         </Vehicle >
      </ScenarioObject >
   </Entities >
   <Storyboard >
      <Init >
         <Actions >
            <Private entityRef="Ego" >
               <PrivateAction >
                  <TeleportAction >
                     <Position >
                        <LanePosition laneId="-4" offset="0" roadId="0" s="5"/ >
                     </Position >
                  </TeleportAction >
               </PrivateAction >
               <PrivateAction >
                  <LongitudinalAction >
                     <SpeedAction >
                        <SpeedActionDynamics dynamicsDimension="time" dynamicsShape="step"
value="0"/ >
                        <SpeedActionTarget >
                           <AbsoluteTargetSpeed value="16.66666667"/ >
                        </SpeedActionTarget >
                     </SpeedAction >
                  </LongitudinalAction >
               </PrivateAction >
            </Private >
            <Private entityRef="cut_in_vehicle" >
               <PrivateAction >
                  <TeleportAction >
                     <Position >
                        <RelativeLanePosition dLane="-1" ds="55" entityRef="Ego"/ >
                     </Position >
                  </TeleportAction >
               </PrivateAction >
               <PrivateAction >
                  <LongitudinalAction >
                     <SpeedAction >
                        <SpeedActionDynamics dynamicsDimension="time" dynamicsShape="step"
```



```
value="0"/ >
                        <SpeedActionTarget >
                            <AbsoluteTargetSpeed value="11.1111"/ >
                        </SpeedActionTarget >
                    </SpeedAction >
                </LongitudinalAction >
            </PrivateAction >
        </Private >
    </Actions >
</Init >
<Story name="CutInUnavoidableCollisionStory" >
    <Act name="Ego_activate_controller_7_act" >
        <ManeuverGroup maximumExecutionCount="1" name="Ego_activate_controller_7_maneuvers" >
            <Actors selectTriggeringEntities="false" >
                <EntityRef entityRef="Ego"/ >
            </Actors >
            <Maneuver name="Ego_activate_controller_7_maneuver" >
                <Event maximumExecutionCount="1" name="Ego_activate_controller_7_event"
priority="overwrite" >
                    <Action name="Ego_activate_controller_7" >
                        <PrivateAction >
                            <ActivateControllerAction lateral="true" longitudinal="true"/ >
                        </PrivateAction >
                    </Action >
                    <StartTrigger >
                        <ConditionGroup >
                            <Condition conditionEdge="rising" delay="0"
name="Ego_activate_controller_7_condition_1" >
                                <ByValueCondition >
                                    <SimulationTimeCondition rule="greaterThan" value="0"/ >
                                </ByValueCondition >
                            </Condition >
                        </ConditionGroup >
                    </StartTrigger >
                </Event >
            </Maneuver >
        </ManeuverGroup >
        <StartTrigger >
            <ConditionGroup >
                <Condition conditionEdge="rising" delay="0" name="Ego_activate_controller_7_condition"
>
                    <ByValueCondition >
                        <SimulationTimeCondition rule="greaterThan" value="10000"/ >
                    </ByValueCondition >
                </Condition >
            </ConditionGroup >
        </StartTrigger >
    </Act >
    <Act name="cut_in_vehicle_change_lane_act" >
        <ManeuverGroup maximumExecutionCount="1" name="cut_in_vehicle_change_lane_maneuvers"
>
            <Actors selectTriggeringEntities="false" >
                <EntityRef entityRef="cut_in_vehicle"/ >
            </Actors >
            <Maneuver name="cut_in_vehicle_change_lane_maneuver" >
                <Event maximumExecutionCount="1" name="cut_in_vehicle_change_lane_event"
priority="overwrite" >
                    <Action name="cut_in_vehicle_change_lane" >
                        <PrivateAction >
                            <LateralAction >
                                <LaneChangeAction >
                                    <LaneChangeActionDynamics dynamicsDimension="rate"
dynamicsShape="linear" value="3"/ >
                                    <LaneChangeTarget >
                                        <RelativeTargetLane entityRef="Ego" value="0"/ >
                                    </LaneChangeTarget >
                                </LaneChangeAction >
                            </LateralAction >
                        </PrivateAction >
```



```
                    </Action >
                    <StartTrigger >
                        <ConditionGroup >
                            <Condition conditionEdge="rising" delay="0"
name="cut_in_vehicle_change_lane_condition" >
                                <ByEntityCondition >
                                    <TriggeringEntities triggeringEntitiesRule="any" >
                                        <EntityRef entityRef="cut_in_vehicle"/ >
                                    </TriggeringEntities >
                                    <EntityCondition >
                                        <DistanceCondition alongRoute="true" freespace="true" rule="lessThan"
value="10" >
                                            <Position >
                                                <RelativeObjectPosition dx="0" dy="0" dz="0" entityRef="Ego" >
                                                    <Orientation h="0" p="0" r="0" type="absolute"/ >
                                                </RelativeObjectPosition >
                                            </Position >
                                        </DistanceCondition >
                                    </EntityCondition >
                                </ByEntityCondition >
                            </Condition >
                        </ConditionGroup >
                    </StartTrigger >
                </Event >
            </Maneuver >
        </ManeuverGroup >
        <StartTrigger >
            <ConditionGroup >
                <Condition conditionEdge="rising" delay="0"
name="cut_in_vehicle_change_lane_condition_1" >
                    <ByValueCondition >
                        <SimulationTimeCondition rule="greaterThan" value="0"/ >
                    </ByValueCondition >
                </Condition >
            </ConditionGroup >
        </StartTrigger >
    </Act >
    <Act name="cut_in_vehicle_change_speed_act" >
        <ManeuverGroup maximumExecutionCount="1"
name="cut_in_vehicle_change_speed_maneuvers" >
            <Actors selectTriggeringEntities="false" >
                <EntityRef entityRef="cut_in_vehicle"/ >
            </Actors >
            <Maneuver name="cut_in_vehicle_change_speed_maneuver" >
                <Event maximumExecutionCount="1" name="cut_in_vehicle_change_speed_event"
priority="overwrite" >
                    <Action name="cut_in_vehicle_change_speed" >
                        <PrivateAction >
                            <LongitudinalAction >
                                <SpeedAction >
                                    <SpeedActionDynamics dynamicsDimension="rate" dynamicsShape="linear"
value="0"/ >
                                    <SpeedActionTarget >
                                        <AbsoluteTargetSpeed value="15"/ >
                                    </SpeedActionTarget >
                                </SpeedAction >
                            </LongitudinalAction >
                        </PrivateAction >
                    </Action >
                    <StartTrigger >
                        <ConditionGroup >
                            <Condition conditionEdge="rising" delay="0"
name="cut_in_vehicle_change_speed_condition" >
                                <ByEntityCondition >
                                    <TriggeringEntities triggeringEntitiesRule="any" >
                                        <EntityRef entityRef="cut_in_vehicle"/ >
                                    </TriggeringEntities >
                                    <EntityCondition >
                                        <DistanceCondition alongRoute="true" freespace="true" rule="lessThan"
```



```
value="10" >
                                        <Position >
                                            <RelativeObjectPosition dx="0" dy="0" dz="0" entityRef="Ego" >
                                                <Orientation h="0" p="0" r="0" type="absolute"/ >
                                            </RelativeObjectPosition >
                                        </Position >
                                    </DistanceCondition >
                                </EntityCondition >
                            </ByEntityCondition >
                        </Condition >
                    </ConditionGroup >
                </StartTrigger >
            </Event >
        </Maneuver >
    </ManeuverGroup >
    <StartTrigger >
        <ConditionGroup >
            <Condition conditionEdge="rising" delay="0"
name="cut_in_vehicle_change_speed_condition_1" >
                <ByValueCondition >
                    <SimulationTimeCondition rule="greaterThan" value="0"/ >
                </ByValueCondition >
            </Condition >
        </ConditionGroup >
    </StartTrigger >
</Act >
</Story >
<StopTrigger >
    <ConditionGroup >
        <Condition conditionEdge="rising" delay="0" name="story_end" >
            <ByValueCondition >
                <SimulationTimeCondition rule="greaterThan" value="40"/ >
            </ByValueCondition >
        </Condition >
    </ConditionGroup >
</StopTrigger >
</Storyboard > </OpenSCENARIO>
```

CutOutFullyBlocking

简述：主车Ego与同车道前方33.33m的头车lead_vehicle都按照初始速度16.667mps匀速行驶，正前方约500m处有一个行人target_blocking，lead_vehicle在与行人距离小于50m时往左变道躲避行人，10000s后激活Ego的controller，40s后停止场景（激活时间与场景停止时间可修改）。

osc2.0场景

```
import standard

scenario CutOutFullyBlocking:
    # map
    map: map
    map.set_map_file("./ALKS_Road.xodr")

    # parameter
    Ego_InitPosition_LaneId: string = '-4'
    Ego_InitSpeed_Ve0: speed = 16.66666667mps
    LeadVehicle_InitDistance_dx0: length = 38.33333333m
    Ego_Odr: odr_point = map.create_odr_point(road_id: '0', lane_id: Ego_InitPosition_LaneId, s: 5.0m, t: 0.0m)
    Ego_InitPosition: pose_3d with:
        keep(it.odr_point == Ego_Odr)
    Blocking_Odr: odr_point = map.create_odr_point(road_id: '0', lane_id: Ego_InitPosition_LaneId, s: 500.0m,
t: 0.0m)
    Blocking_InitPosition: pose_3d with:
        keep(it.odr_point == Blocking_Odr)
    LeadVehicle_Odr: odr_point = map.create_odr_point(road_id: '0', lane_id: Ego_InitPosition_LaneId, s:
LeadVehicle_InitDistance_dx0, t: 0.0m)
    LeadVehicle_InitPosition: pose_3d with:
```



```
    keep(it.odr_point == LeadVehicle_Odr)
  FrontOfLead_Distance_dx0_f: length = 50.0m
  Lateral_Velocity_Vy: speed = 2.0mps
  m_direction : distance_direction = longitudinal
  m_mode : distance_mode = bounding_boxes
  m_rate_profile: dynamics_shape = linear
  m_side: lane_change_side = left
  Duration: time = 40s

  # entity
  Ego_Name: string = "Audi_A3_2009_black"
  LeadVehicle_Name: string = "Audi_A3_2009_red"
  Ego_Controller: string = "DefaultDriver"
  Ego: vehicle with:
    keep(it.name == Ego_Name)
    keep(it.initial_bm == Ego_Controller)
  target_blocking: person with:
    keep(it.name == "Christian")
    keep(it.model == "male_adult")
  lead_vehicle: vehicle with:
    keep(it.name == LeadVehicle_Name)

  # storyboard
  do parallel(duration: Duration):
    # init
    Ego.assign_init_position(position: Ego_InitPosition)
    Ego.assign_init_speed(Ego_InitSpeed_Ve0)
    target_blocking.assign_init_position(position: Blocking_InitPosition)
    lead_vehicle.assign_init_position(position: LeadVehicle_InitPosition)
    lead_vehicle.assign_init_speed(Ego_InitSpeed_Ve0)

    serial:
      # act1: ActivateALKSControllerAct
      wait elapsed(10000s)
      Ego.activate_controller(true, true)  # lateral, longitudinal
    serial:
      # act2: CutOutAct
      wait lead_vehicle.object_distance(reference: target_blocking, direction: m_direction, mode: m_mode)
< FrontOfLead_Distance_dx0_f
      lead_vehicle.change_lane(number_of_lanes: 1, side: m_side, reference: lead_vehicle, rate_peak:
Lateral_Velocity_Vy, rate_profile: m_rate_profile)
```

转化结果

```
<OpenSCENARIO>
  <FileHeader author="Octopus/Simulation" date="2022-10-29T01:31:30" description="" revMajor="1"
revMinor="0"/>
  <CatalogLocations>
    <VehicleCatalog>
      <Directory path="Distros/Current/Config/Players/Vehicles"/>
    </VehicleCatalog>
    <ControllerCatalog>
      <Directory path="Distros/Current/Config/Players/driverCfg.xml"/>
    </ControllerCatalog>
    <PedestrianCatalog>
      <Directory path="Distros/Current/Config/Players/Pedestrians"/>
    </PedestrianCatalog>
    <MiscObjectCatalog>
      <Directory path="Distros/Current/Config/Players/Objects"/>
    </MiscObjectCatalog>
  </CatalogLocations>
  <RoadNetwork>
    <LogicFile filepath="./ALKS_Road.xodr"/>
    <SceneGraphFile filepath=""/>
  </RoadNetwork>
  <Entities>
    <ScenarioObject name="Ego">
      <Vehicle name="Audi_A3_2009_black" vehicleCategory="car">
        <BoundingBox>
          <Center x="1.5" y="0" z="0.9"/>
```



```
            <Dimensions height="2.1" length="4.5" width="1.8"/>
         </BoundingBox>
         <Performance maxAcceleration="8" maxDeceleration="8" maxSpeed="60"/>
         <Axles>
            <FrontAxle maxSteering="0.5" positionX="3.1" positionZ="0.3" trackWidth="1.8"
wheelDiameter="0.6"/>
            <RearAxle maxSteering="0" positionX="0" positionZ="0.3" trackWidth="1.8"
wheelDiameter="0.6"/>
         </Axles>
         <Properties>
            <Property name="control" value="external"/>
         </Properties>
      </Vehicle>
      <ObjectController>
         <Controller name="DefaultDriver">
            <Properties/>
         </Controller>
      </ObjectController>
   </ScenarioObject>
   <ScenarioObject name="lead_vehicle">
      <Vehicle name="Audi_A3_2009_red" vehicleCategory="car">
         <BoundingBox>
            <Center x="1.5" y="0" z="0.9"/>
            <Dimensions height="2.1" length="4.5" width="1.8"/>
         </BoundingBox>
         <Performance maxAcceleration="8" maxDeceleration="8" maxSpeed="60"/>
         <Axles>
            <FrontAxle maxSteering="0.5" positionX="3.1" positionZ="0.3" trackWidth="1.8"
wheelDiameter="0.6"/>
            <RearAxle maxSteering="0" positionX="0" positionZ="0.3" trackWidth="1.8"
wheelDiameter="0.6"/>
         </Axles>
         <Properties/>
      </Vehicle>
   </ScenarioObject>
   <ScenarioObject name="target_blocking">
      <Pedestrian mass="80" model="male_adult" name="Christian" pedestrianCategory="pedestrian">
         <BoundingBox>
            <Center x="1.5" y="0" z="0.9"/>
            <Dimensions height="2.1" length="4.5" width="1.8"/>
         </BoundingBox>
         <Properties/>
      </Pedestrian>
   </ScenarioObject>
</Entities>
<Storyboard>
   <Init>
      <Actions>
         <Private entityRef="Ego">
            <PrivateAction>
               <TeleportAction>
                  <Position>
                     <LanePosition laneId="-4" offset="0" roadId="0" s="5"/>
                  </Position>
               </TeleportAction>
            </PrivateAction>
            <PrivateAction>
               <LongitudinalAction>
                  <SpeedAction>
                     <SpeedActionDynamics dynamicsDimension="time" dynamicsShape="step"
value="0"/>
                     <SpeedActionTarget>
                        <AbsoluteTargetSpeed value="16.66666667"/>
                     </SpeedActionTarget>
                  </SpeedAction>
               </LongitudinalAction>
            </PrivateAction>
         </Private>
         <Private entityRef="lead_vehicle">
```



```
                <PrivateAction>
                    <TeleportAction>
                        <Position>
                            <LanePosition laneId="-4" offset="0" roadId="0" s="38.33333333"/>
                        </Position>
                    </TeleportAction>
                </PrivateAction>
                <PrivateAction>
                    <LongitudinalAction>
                        <SpeedAction>
                            <SpeedActionDynamics dynamicsDimension="time" dynamicsShape="step"
value="0"/>
                            <SpeedActionTarget>
                                <AbsoluteTargetSpeed value="16.66666667"/>
                            </SpeedActionTarget>
                        </SpeedAction>
                    </LongitudinalAction>
                </PrivateAction>
            </Private>
            <Private entityRef="target_blocking">
                <PrivateAction>
                    <TeleportAction>
                        <Position>
                            <LanePosition laneId="-4" offset="0" roadId="0" s="500"/>
                        </Position>
                    </TeleportAction>
                </PrivateAction>
            </Private>
        </Actions>
    </Init>
    <Story name="CutOutFullyBlockingStory">
        <Act name="Ego_activate_controller_4_act">
            <ManeuverGroup maximumExecutionCount="1" name="Ego_activate_controller_4_maneuvers">
                <Actors selectTriggeringEntities="false">
                    <EntityRef entityRef="Ego"/>
                </Actors>
                <Maneuver name="Ego_activate_controller_4_maneuver">
                    <Event maximumExecutionCount="1" name="Ego_activate_controller_4_event"
priority="overwrite">
                        <Action name="Ego_activate_controller_4">
                            <PrivateAction>
                                <ActivateControllerAction lateral="true" longitudinal="true"/>
                            </PrivateAction>
                        </Action>
                        <StartTrigger>
                            <ConditionGroup>
                                <Condition conditionEdge="rising" delay="0"
name="Ego_activate_controller_4_condition_1">
                                    <ByValueCondition>
                                        <SimulationTimeCondition rule="greaterThan" value="0"/>
                                    </ByValueCondition>
                                </Condition>
                            </ConditionGroup>
                        </StartTrigger>
                    </Event>
                </Maneuver>
            </ManeuverGroup>
            <StartTrigger>
                <ConditionGroup>
                    <Condition conditionEdge="rising" delay="0" name="Ego_activate_controller_4_condition">
                        <ByValueCondition>
                            <SimulationTimeCondition rule="greaterThan" value="10000"/>
                        </ByValueCondition>
                    </Condition>
                </ConditionGroup>
            </StartTrigger>
        </Act>
        <Act name="lead_vehicle_change_lane_act">
            <ManeuverGroup maximumExecutionCount="1" name="lead_vehicle_change_lane_maneuvers">
```



```
            <Actors selectTriggeringEntities="false">
               <EntityRef entityRef="lead_vehicle"/>
            </Actors>
            <Maneuver name="lead_vehicle_change_lane_maneuver">
               <Event maximumExecutionCount="1" name="lead_vehicle_change_lane_event"
priority="overwrite">
                  <Action name="lead_vehicle_change_lane">
                     <PrivateAction>
                        <LateralAction>
                           <LaneChangeAction>
                              <LaneChangeActionDynamics dynamicsDimension="rate"
dynamicsShape="linear" value="2"/>
                              <LaneChangeTarget>
                                 <RelativeTargetLane entityRef="lead_vehicle" value="1"/>
                              </LaneChangeTarget>
                           </LaneChangeAction>
                        </LateralAction>
                     </PrivateAction>
                  </Action>
                  <StartTrigger>
                     <ConditionGroup>
                        <Condition conditionEdge="rising" delay="0"
name="lead_vehicle_change_lane_condition">
                           <ByEntityCondition>
                              <TriggeringEntities triggeringEntitiesRule="any">
                                 <EntityRef entityRef="lead_vehicle"/>
                              </TriggeringEntities>
                              <EntityCondition>
                                 <DistanceCondition alongRoute="true" freespace="true" rule="lessThan"
value="50">
                                    <Position>
                                       <RelativeObjectPosition dx="0" dy="0" dz="0"
entityRef="target_blocking">
                                          <Orientation h="0" p="0" r="0" type="absolute"/>
                                       </RelativeObjectPosition>
                                    </Position>
                                 </DistanceCondition>
                              </EntityCondition>
                           </ByEntityCondition>
                        </Condition>
                     </ConditionGroup>
                  </StartTrigger>
               </Event>
            </Maneuver>
         </ManeuverGroup>
         <StartTrigger>
            <ConditionGroup>
               <Condition conditionEdge="rising" delay="0"
name="lead_vehicle_change_lane_condition_1">
                  <ByValueCondition>
                     <SimulationTimeCondition rule="greaterThan" value="0"/>
                  </ByValueCondition>
               </Condition>
            </ConditionGroup>
         </StartTrigger>
      </Act>
   </Story>
   <StopTrigger>
      <ConditionGroup>
         <Condition conditionEdge="rising" delay="0" name="story_end">
            <ByValueCondition>
               <SimulationTimeCondition rule="greaterThan" value="40"/>
            </ByValueCondition>
         </Condition>
      </ConditionGroup>
   </StopTrigger>
</Storyboard>
</OpenSCENARIO>
```


CutOutMultipleBlockingTargets

简述：主车Ego与同车道前方33.33m的头车lead_vehicle都按照初始速度16.667mps匀速行驶，正前方约500m处有一个行人target_blocking1，约515m处有一个障碍车辆target_blocking2，lead_vehicle在与行人target_blocking1距离小于50m时往左变道避障，10000s后激活Ego的controller，40s后停止场景（激活时间与场景停止时间可修改）。

osc2.0场景

```
import standard

scenario CutOutMultipleBlockingTargets:
    # map
    map: map
    map.set_map_file("./ALKS_Road.xodr")

    # parameter
    Ego_InitPosition_LaneId: string = "-4"
    Ego_InitSpeed_Ve0: speed = 16.66666667mps
    LeadVehicle_InitDistance_dx0: length = 38.33333333m
    Ego_Odr: odr_point = map.create_odr_point(road_id: '0', lane_id: Ego_InitPosition_LaneId, s: 5.0m, t: 0.0m)
    Ego_InitPosition: pose_3d with:
        keep(it.odr_point == Ego_Odr)
    Blocking1_Odr: odr_point = map.create_odr_point(road_id: '0', lane_id: Ego_InitPosition_LaneId, s: 500.0m,
t: 0.0m)
    Blocking1_InitPosition: pose_3d with:
        keep(it.odr_point == Blocking1_Odr)
    Blocking2_Odr: odr_point = map.create_odr_point(road_id: '0', lane_id: Ego_InitPosition_LaneId, s: 515.0m,
t: 0.0m)
    Blocking2_InitPosition: pose_3d with:
        keep(it.odr_point == Blocking2_Odr)
    LeadVehicle_Odr: odr_point = map.create_odr_point(road_id: '0', lane_id: Ego_InitPosition_LaneId, s:
LeadVehicle_InitDistance_dx0, t: 0.0m)
    LeadVehicle_InitPosition: pose_3d with:
        keep(it.odr_point == LeadVehicle_Odr)
    FrontOfLead_Distance_dx0_f: length = 50.0m
    Lateral_Velocity_Vy: speed = 2.0mps
    m_direction : distance_direction = longitudinal
    m_mode : distance_mode = bounding_boxes
    m_rate_profile: dynamics_shape = linear
    m_side: lane_change_side = left
    Duration: time = 40s

    # entity
    Ego_Name: string = "Audi_A3_2009_black"
    LeadVehicle_Name: string = "Audi_A3_2009_red"
    TargetBlocking2_Name: string = "Audi_A3_2009_blue"
    Ego_Controller: string = "DefaultDriver"
    Ego: vehicle with:
        keep(it.name == Ego_Name)
        keep(it.initial_bm == Ego_Controller)
    target_blocking1: person with:
        keep(it.name == "Christian")
        keep(it.model == "male_adult")
    target_blocking2: vehicle with:
        keep(it.name == TargetBlocking2_Name)
    lead_vehicle: vehicle with:
        keep(it.name == LeadVehicle_Name)

    # storyboard
    do parallel(duration: Duration):
        # init
        Ego.assign_init_position(position: Ego_InitPosition)
        Ego.assign_init_speed(Ego_InitSpeed_Ve0)
        target_blocking1.assign_init_position(position: Blocking1_InitPosition)
        target_blocking2.assign_init_position(position: Blocking2_InitPosition)
        lead_vehicle.assign_init_position(position: LeadVehicle_InitPosition)
```



```
    lead_vehicle.assign_init_speed(Ego_InitSpeed_Ve0)

    serial:
        # act1: ActivateALKSControllerAct
        wait elapsed(10000s)
        Ego.activate_controller(true, true)  # lateral, longitudinal
    serial:
        # act2: CutOutAct
        wait lead_vehicle.object_distance(reference: target_blocking1, direction: m_direction, mode:
m_mode) < FrontOfLead_Distance_dx0_f
        lead_vehicle.change_lane(number_of_lanes: 1, side: m_side, reference: lead_vehicle, rate_peak:
Lateral_Velocity_Vy, rate_profile: m_rate_profile)
```

转化结果

```
<OpenSCENARIO>
  <FileHeader author="Octopus/Simulation" date="2022-10-29T01:26:41" description="" revMajor="1"
revMinor="0"/>
  <CatalogLocations>
    <VehicleCatalog>
      <Directory path="Distros/Current/Config/Players/Vehicles"/>
    </VehicleCatalog>
    <ControllerCatalog>
      <Directory path="Distros/Current/Config/Players/driverCfg.xml"/>
    </ControllerCatalog>
    <PedestrianCatalog>
      <Directory path="Distros/Current/Config/Players/Pedestrians"/>
    </PedestrianCatalog>
    <MiscObjectCatalog>
      <Directory path="Distros/Current/Config/Players/Objects"/>
    </MiscObjectCatalog>
  </CatalogLocations>
  <RoadNetwork>
    <LogicFile filepath="./ALKS_Road.xodr"/>
    <SceneGraphFile filepath=""/>
  </RoadNetwork>
  <Entities>
    <ScenarioObject name="Ego">
      <Vehicle name="Audi_A3_2009_black" vehicleCategory="car">
        <BoundingBox>
          <Center x="1.5" y="0" z="0.9"/>
          <Dimensions height="2.1" length="4.5" width="1.8"/>
        </BoundingBox>
        <Performance maxAcceleration="8" maxDeceleration="8" maxSpeed="60"/>
        <Axles>
          <FrontAxle maxSteering="0.5" positionX="3.1" positionZ="0.3" trackWidth="1.8"
wheelDiameter="0.6"/>
          <RearAxle maxSteering="0" positionX="0" positionZ="0.3" trackWidth="1.8"
wheelDiameter="0.6"/>
        </Axles>
        <Properties>
          <Property name="control" value="external"/>
        </Properties>
      </Vehicle>
      <ObjectController>
        <Controller name="DefaultDriver">
          <Properties/>
        </Controller>
      </ObjectController>
    </ScenarioObject>
    <ScenarioObject name="lead_vehicle">
      <Vehicle name="Audi_A3_2009_red" vehicleCategory="car">
        <BoundingBox>
          <Center x="1.5" y="0" z="0.9"/>
          <Dimensions height="2.1" length="4.5" width="1.8"/>
        </BoundingBox>
        <Performance maxAcceleration="8" maxDeceleration="8" maxSpeed="60"/>
        <Axles>
          <FrontAxle maxSteering="0.5" positionX="3.1" positionZ="0.3" trackWidth="1.8"
wheelDiameter="0.6"/>
```



```
            <RearAxle maxSteering="0" positionX="0" positionZ="0.3" trackWidth="1.8"
wheelDiameter="0.6"/>
          </Axles>
          <Properties/>
        </Vehicle>
      </ScenarioObject>
      <ScenarioObject name="target_blocking1">
        <Pedestrian mass="80" model="male_adult" name="Christian" pedestrianCategory="pedestrian">
          <BoundingBox>
            <Center x="1.5" y="0" z="0.9"/>
            <Dimensions height="2.1" length="4.5" width="1.8"/>
          </BoundingBox>
          <Properties/>
        </Pedestrian>
      </ScenarioObject>
      <ScenarioObject name="target_blocking2">
        <Vehicle name="Audi_A3_2009_blue" vehicleCategory="car">
          <BoundingBox>
            <Center x="1.5" y="0" z="0.9"/>
            <Dimensions height="2.1" length="4.5" width="1.8"/>
          </BoundingBox>
          <Performance maxAcceleration="8" maxDeceleration="8" maxSpeed="60"/>
          <Axles>
            <FrontAxle maxSteering="0.5" positionX="3.1" positionZ="0.3" trackWidth="1.8"
wheelDiameter="0.6"/>
            <RearAxle maxSteering="0" positionX="0" positionZ="0.3" trackWidth="1.8"
wheelDiameter="0.6"/>
          </Axles>
          <Properties/>
        </Vehicle>
      </ScenarioObject>
    </Entities>
    <Storyboard>
      <Init>
        <Actions>
          <Private entityRef="Ego">
            <PrivateAction>
              <TeleportAction>
                <Position>
                  <LanePosition laneId="-4" offset="0" roadId="0" s="5"/>
                </Position>
              </TeleportAction>
            </PrivateAction>
            <PrivateAction>
              <LongitudinalAction>
                <SpeedAction>
                  <SpeedActionDynamics dynamicsDimension="time" dynamicsShape="step"
value="0"/>
                  <SpeedActionTarget>
                    <AbsoluteTargetSpeed value="16.66666667"/>
                  </SpeedActionTarget>
                </SpeedAction>
              </LongitudinalAction>
            </PrivateAction>
          </Private>
          <Private entityRef="lead_vehicle">
            <PrivateAction>
              <TeleportAction>
                <Position>
                  <LanePosition laneId="-4" offset="0" roadId="0" s="38.33333333"/>
                </Position>
              </TeleportAction>
            </PrivateAction>
            <PrivateAction>
              <LongitudinalAction>
                <SpeedAction>
                  <SpeedActionDynamics dynamicsDimension="time" dynamicsShape="step"
value="0"/>
                  <SpeedActionTarget>
```



```
                        <AbsoluteTargetSpeed value="16.66666667"/>
                    </SpeedActionTarget>
                </SpeedAction>
            </LongitudinalAction>
        </PrivateAction>
    </Private>
    <Private entityRef="target_blocking1">
        <PrivateAction>
            <TeleportAction>
                <Position>
                    <LanePosition laneId="-4" offset="0" roadId="0" s="500"/>
                </Position>
            </TeleportAction>
        </PrivateAction>
    </Private>
    <Private entityRef="target_blocking2">
        <PrivateAction>
            <TeleportAction>
                <Position>
                    <LanePosition laneId="-4" offset="0" roadId="0" s="515"/>
                </Position>
            </TeleportAction>
        </PrivateAction>
    </Private>
  </Actions>
 </Init>
 <Story name="CutOutMultipleBlockingTargetsStory">
  <Act name="Ego_activate_controller_7_act">
    <ManeuverGroup maximumExecutionCount="1" name="Ego_activate_controller_7_maneuvers">
        <Actors selectTriggeringEntities="false">
            <EntityRef entityRef="Ego"/>
        </Actors>
        <Maneuver name="Ego_activate_controller_7_maneuver">
            <Event maximumExecutionCount="1" name="Ego_activate_controller_7_event"
priority="overwrite">
                <Action name="Ego_activate_controller_7">
                    <PrivateAction>
                        <ActivateControllerAction lateral="true" longitudinal="true"/>
                    </PrivateAction>
                </Action>
                <StartTrigger>
                    <ConditionGroup>
                        <Condition conditionEdge="rising" delay="0"
name="Ego_activate_controller_7_condition_1">
                            <ByValueCondition>
                                <SimulationTimeCondition rule="greaterThan" value="0"/>
                            </ByValueCondition>
                        </Condition>
                    </ConditionGroup>
                </StartTrigger>
            </Event>
        </Maneuver>
    </ManeuverGroup>
    <StartTrigger>
        <ConditionGroup>
            <Condition conditionEdge="rising" delay="0" name="Ego_activate_controller_7_condition">
                <ByValueCondition>
                    <SimulationTimeCondition rule="greaterThan" value="10000"/>
                </ByValueCondition>
            </Condition>
        </ConditionGroup>
    </StartTrigger>
  </Act>
  <Act name="lead_vehicle_change_lane_1_act">
    <ManeuverGroup maximumExecutionCount="1"
name="lead_vehicle_change_lane_1_maneuvers">
        <Actors selectTriggeringEntities="false">
            <EntityRef entityRef="lead_vehicle"/>
        </Actors>
```



```
            <Maneuver name="lead_vehicle_change_lane_1_maneuver">
               <Event maximumExecutionCount="1" name="lead_vehicle_change_lane_1_event"
priority="overwrite">
                  <Action name="lead_vehicle_change_lane_1">
                     <PrivateAction>
                        <LateralAction>
                           <LaneChangeAction>
                              <LaneChangeActionDynamics dynamicsDimension="rate"
dynamicsShape="linear" value="2"/>
                              <LaneChangeTarget>
                                 <RelativeTargetLane entityRef="lead_vehicle" value="1"/>
                              </LaneChangeTarget>
                           </LaneChangeAction>
                        </LateralAction>
                     </PrivateAction>
                  </Action>
                  <StartTrigger>
                     <ConditionGroup>
                        <Condition conditionEdge="rising" delay="0"
name="lead_vehicle_change_lane_1_condition">
                           <ByEntityCondition>
                              <TriggeringEntities triggeringEntitiesRule="any">
                                 <EntityRef entityRef="lead_vehicle"/>
                              </TriggeringEntities>
                              <EntityCondition>
                                 <DistanceCondition alongRoute="true" freespace="true" rule="lessThan"
value="50">
                                    <Position>
                                       <RelativeObjectPosition dx="0" dy="0" dz="0"
entityRef="target_blocking1">
                                          <Orientation h="0" p="0" r="0" type="absolute"/>
                                       </RelativeObjectPosition>
                                    </Position>
                                 </DistanceCondition>
                              </EntityCondition>
                           </ByEntityCondition>
                        </Condition>
                     </ConditionGroup>
                  </StartTrigger>
               </Event>
            </Maneuver>
         </ManeuverGroup>
         <StartTrigger>
            <ConditionGroup>
               <Condition conditionEdge="rising" delay="0"
name="lead_vehicle_change_lane_1_condition_1">
                  <ByValueCondition>
                     <SimulationTimeCondition rule="greaterThan" value="0"/>
                  </ByValueCondition>
               </Condition>
            </ConditionGroup>
         </StartTrigger>
      </Act>
   </Story>
   <StopTrigger>
      <ConditionGroup>
         <Condition conditionEdge="rising" delay="0" name="story_end">
            <ByValueCondition>
               <SimulationTimeCondition rule="greaterThan" value="40"/>
            </ByValueCondition>
         </Condition>
      </ConditionGroup>
   </StopTrigger>
</Storyboard>
</OpenSCENARIO>
```


ForwardDetectionRange

简述：主车Ego按照初始速度16.667mps匀速行驶，右前方（偏移5.25m）约500m处有一个行人target_blocking，10000s后激活Ego的controller，40s后停止场景（激活时间与场景停止时间可修改）。

osc2.0场景

```
import standard

scenario ForwardDetectionRange:
    # map
    map: map
    map.set_map_file("./ALKS_Road.xodr")

    # parameter
    Ego_InitPosition_LaneId: string = "-4"
    Ego_Odr: odr_point = map.create_odr_point(road_id: '0', lane_id: Ego_InitPosition_LaneId, s: 5.0m, t: 0.0m)
    Ego_InitPosition: pose_3d with:
        keep(it.odr_point == Ego_Odr)
    Ego_InitSpeed_Ve0: speed = 16.66666667mps
    TargetBlocking_InitPosition_Offset: length = -5.25m
    TargetBlocking_Odr: odr_point = map.create_odr_point(road_id: '0', lane_id: Ego_InitPosition_LaneId, s:
500.0m, t: TargetBlocking_InitPosition_Offset)
    TargetBlocking_InitPosition: pose_3d with:
        keep(it.odr_point == TargetBlocking_Odr)
    m_vehicle_catalog: catalog = vehicle_catalog
    m_target_blocking_catalog: catalog = pedestrian_catalog
    Duration: time = 40s

    # entity
    Ego_Name: string = "Audi_A3_2009_black"
    Ego_Controller: string = "DefaultDriver"
    Ego: vehicle with:
        keep(it.name == Ego_Name)
        keep(it.initial_bm == Ego_Controller)
    target_blocking: person with:
        keep(it.name == "Christian")
        keep(it.model == "male_adult")

    # storyboard
    do parallel(duration: Duration):
        # init
        Ego.assign_init_position(position: Ego_InitPosition)
        Ego.assign_init_speed(Ego_InitSpeed_Ve0)
        target_blocking.assign_init_position(position: TargetBlocking_InitPosition)

        # story
        serial:
            # act 1: ActivateALKSControllerStory
            wait elapsed(10000s)
            Ego.activate_controller(true, true)  # lateral, longitudinal
```

转化结果

```
<OpenSCENARIO>
    <FileHeader author="Octopus/Simulation" date="2022-10-29T01:46:19" description="" revMajor="1"
revMinor="0"/>
    <CatalogLocations>
        <VehicleCatalog>
            <Directory path="Distros/Current/Config/Players/Vehicles"/>
        </VehicleCatalog>
        <ControllerCatalog>
            <Directory path="Distros/Current/Config/Players/driverCfg.xml"/>
        </ControllerCatalog>
        <PedestrianCatalog>
            <Directory path="Distros/Current/Config/Players/Pedestrians"/>
        </PedestrianCatalog>
        <MiscObjectCatalog>
```



```
        <Directory path="Distros/Current/Config/Players/Objects"/>
      </MiscObjectCatalog>
   </CatalogLocations>
   <RoadNetwork>
      <LogicFile filepath="./ALKS_Road.xodr"/>
      <SceneGraphFile filepath=""/>
   </RoadNetwork>
   <Entities>
      <ScenarioObject name="Ego">
         <Vehicle name="Audi_A3_2009_black" vehicleCategory="car">
            <BoundingBox>
               <Center x="1.5" y="0" z="0.9"/>
               <Dimensions height="2.1" length="4.5" width="1.8"/>
            </BoundingBox>
            <Performance maxAcceleration="8" maxDeceleration="8" maxSpeed="60"/>
            <Axles>
               <FrontAxle maxSteering="0.5" positionX="3.1" positionZ="0.3" trackWidth="1.8"
wheelDiameter="0.6"/>
               <RearAxle maxSteering="0" positionX="0" positionZ="0.3" trackWidth="1.8"
wheelDiameter="0.6"/>
            </Axles>
            <Properties>
               <Property name="control" value="external"/>
            </Properties>
         </Vehicle>
         <ObjectController>
            <Controller name="DefaultDriver">
               <Properties/>
            </Controller>
         </ObjectController>
      </ScenarioObject>
      <ScenarioObject name="target_blocking">
         <Pedestrian mass="80" model="male_adult" name="Christian" pedestrianCategory="pedestrian">
            <BoundingBox>
               <Center x="1.5" y="0" z="0.9"/>
               <Dimensions height="2.1" length="4.5" width="1.8"/>
            </BoundingBox>
            <Properties/>
         </Pedestrian>
      </ScenarioObject>
   </Entities>
   <Storyboard>
      <Init>
         <Actions>
            <Private entityRef="Ego">
               <PrivateAction>
                  <TeleportAction>
                     <Position>
                        <LanePosition laneId="-4" offset="0" roadId="0" s="5"/>
                     </Position>
                  </TeleportAction>
               </PrivateAction>
               <PrivateAction>
                  <LongitudinalAction>
                     <SpeedAction>
                        <SpeedActionDynamics dynamicsDimension="time" dynamicsShape="step"
value="0"/>
                        <SpeedActionTarget>
                           <AbsoluteTargetSpeed value="16.66666667"/>
                        </SpeedActionTarget>
                     </SpeedAction>
                  </LongitudinalAction>
               </PrivateAction>
            </Private>
            <Private entityRef="target_blocking">
               <PrivateAction>
                  <TeleportAction>
                     <Position>
                        <LanePosition laneId="-4" offset="-5.25" roadId="0" s="500"/>
```



```
                    </Position>
                </TeleportAction>
            </PrivateAction>
        </Private>
    </Actions>
</Init>
<Story name="ForwardDetectionRangeStory">
    <Act name="Ego_activate_controller_7_act">
        <ManeuverGroup maximumExecutionCount="1" name="Ego_activate_controller_7_maneuvers">
            <Actors selectTriggeringEntities="false">
                <EntityRef entityRef="Ego"/>
            </Actors>
            <Maneuver name="Ego_activate_controller_7_maneuver">
                <Event maximumExecutionCount="1" name="Ego_activate_controller_7_event"
priority="overwrite">
                    <Action name="Ego_activate_controller_7">
                        <PrivateAction>
                            <ActivateControllerAction lateral="true" longitudinal="true"/>
                        </PrivateAction>
                    </Action>
                    <StartTrigger>
                        <ConditionGroup>
                            <Condition conditionEdge="rising" delay="0"
name="Ego_activate_controller_7_condition_1">
                                <ByValueCondition>
                                    <SimulationTimeCondition rule="greaterThan" value="0"/>
                                </ByValueCondition>
                            </Condition>
                        </ConditionGroup>
                    </StartTrigger>
                </Event>
            </Maneuver>
        </ManeuverGroup>
        <StartTrigger>
            <ConditionGroup>
                <Condition conditionEdge="rising" delay="0" name="Ego_activate_controller_7_condition">
                    <ByValueCondition>
                        <SimulationTimeCondition rule="greaterThan" value="10000"/>
                    </ByValueCondition>
                </Condition>
            </ConditionGroup>
        </StartTrigger>
    </Act>
</Story>
<StopTrigger>
    <ConditionGroup>
        <Condition conditionEdge="rising" delay="0" name="story_end">
            <ByValueCondition>
                <SimulationTimeCondition rule="greaterThan" value="40"/>
            </ByValueCondition>
        </Condition>
    </ConditionGroup>
</StopTrigger>
</Storyboard>
</OpenSCENARIO>
```

逻辑场景（以 CutOutMultipleBlockingTargets 为例）

逻辑场景的ALKS样例，本文只介绍CutOutMultipleBlockingTargets，更多参数介绍，请查看官方文档。

osc2.0场景

```
import standard

scenario CutOutMultipleBlockingTargets:
    # map
    map: map
```



```
    map.set_map_file("./ALKS_Road.xodr")

    # parameter
    Ego_InitPosition_LaneId: string = ['-4', '-1']
    Ego_InitSpeed_Ve0: speed = [20mps, 40mps, 60mps, 80mps]
    LeadVehicle_InitDistance_dx0: length = [40m..80m]
    Ego_Odr: odr_point = map.create_odr_point(road_id: '0', lane_id: Ego_InitPosition_LaneId, s: 5.0m, t: 0.0m)
    Ego_InitPosition: pose_3d with:
        keep(it.odr_point == Ego_Odr)
    Blocking1_Odr: odr_point = map.create_odr_point(road_id: '0', lane_id: Ego_InitPosition_LaneId, s: 500.0m,
t: 0.0m)
    Blocking1_InitPosition: pose_3d with:
        keep(it.odr_point == Blocking1_Odr)
    Blocking2_Odr: odr_point = map.create_odr_point(road_id: '0', lane_id: Ego_InitPosition_LaneId, s: 515.0m,
t: 0.0m)
    Blocking2_InitPosition: pose_3d with:
        keep(it.odr_point == Blocking2_Odr)
    LeadVehicle_Odr: odr_point = map.create_odr_point(road_id: '0', lane_id: Ego_InitPosition_LaneId, s:
LeadVehicle_InitDistance_dx0, t: 0.0m)
    LeadVehicle_InitPosition: pose_3d with:
        keep(it.odr_point == LeadVehicle_Odr)
    FrontOfLead_Distance_dx0_f: length = 50.0m
    Lateral_Velocity_Vy: speed = [0.1mps..1.0mps]
    m_direction : distance_direction = longitudinal
    m_mode : distance_mode = bounding_boxes
    m_rate_profile: dynamics_shape = [step, linear]
    m_dimension: dynamics_dimension = rate
    m_side: lane_change_side = [left, right]
    Duration: time = [30s, 40s, 50s]

    # entity
    Ego_Name: string = ["Audi_A3_2009_black", "Audi_A3_2009_red"]
    LeadVehicle_Name: string = "Audi_A3_2009_red"
    TargetBlocking2_Name: string = "Audi_A3_2009_blue"
    Ego_Controller: string = "DefaultDriver"
    Ego: vehicle with:
        keep(it.name == Ego_Name)
        keep(it.initial_bm == Ego_Controller)
    target_blocking1: person with:
        keep(it.name == "Christian")
        keep(it.model == "male_adult")
    target_blocking2: vehicle with:
        keep(it.name == TargetBlocking2_Name)
    lead_vehicle: vehicle with:
        keep(it.name == LeadVehicle_Name)

    # storyboard
    do parallel(duration: Duration):
        # init
        Ego.assign_init_position(position: Ego_InitPosition)
        Ego.assign_init_speed(Ego_InitSpeed_Ve0)
        target_blocking1.assign_init_position(position: Blocking1_InitPosition)
        target_blocking2.assign_init_position(position: Blocking2_InitPosition)
        lead_vehicle.assign_init_position(position: LeadVehicle_InitPosition)
        lead_vehicle.assign_init_speed(Ego_InitSpeed_Ve0)

        serial:
            # act1: ActivateALKSControllerAct
            wait elapsed(10000s)
            Ego.activate_controller(true, true)  # lateral, longitudinal
        serial:
            # act2: CutOutAct
            wait lead_vehicle.object_distance(reference: target_blocking1, direction: m_direction, mode:
m_mode) < FrontOfLead_Distance_dx0_f
            lead_vehicle.change_lane(number_of_lanes: 1, side: m_side, reference: lead_vehicle, rate_peak:
Lateral_Velocity_Vy, rate_profile: m_rate_profile, dimension: m_dimension)
```

7.8.2 静态场景（地图）

7.8.2.1 场景组成

静态场景提供了4个种子场景，静态场景的生成和泛化都基于这些种子场景。

四个种子场景分别为：

- straight城区直行
- merge匝道合流
- split匝道分流
- junction路口

7.8.2.2 领域模型设计

以下为4个种子场景可以泛化的参数及范围，请确保在编写odr文件时各参数在以下描述范围内。

7.8.2.2.1 straight 城区直行

- **用途**：创建straight城区直行的静态场景（ 地图 ）
- **参数**：参数如下表

表 7-46 straight 城区直行参数

Parameter	Type	Mandatory	Description
lane_width	length	yes	每个车道宽度。
left_lane_num	int	yes	左侧车道数量。
right_lane_num	int	yes	右侧车道数量。
bikeway	bool	yes	是否有自行车道。
sidewalk	bool	yes	是否有人行道。
main_speed	speed	yes	主路的限速值。
road_length	length	yes	主路的长度。

- **参数取值范围：**

```
lane_width: length = [3m..4m]
left_lane_num: int = [0, 1, 2, 3, 4]
right_lane_num: int = [1, 2, 3, 4]
bikeway: bool = [true, false]
sidewalk: bool = [true, false]
main_speed: speed = [40kph..60kph]
road_length: length = [150m..500m]
```

keep创建

```
my_straight: straight with:
    keep(it.lane_width == 3m)
    keep(it.left_lane_num == 0)
    keep(it.right_lane_num == 3)
    keep(it.bikeway == false)
    keep(it.sidewalk == false)
    keep(it.main_speed == 60kph)
    keep(it.road_length == 500m)
```

create创建

```
my_straight: straight = scenery.create_straight(lane_width: 3m, left_lane_num: 0, right_lane_num: 3,
bikeway: false, sidewalk: false, main_speed: 60kph, road_length: 500m)
```

7.8.2.2.2 merge 匝道合流

- **用途**：创建merge高速匝道合流的静态场景（地图）
- **参数**：参数如下表

表 7-47 merge 参数

Parameter	Type	Mandatory	Description
lane_width	length	yes	每个车道宽度。
left_lane_num	int	yes	左侧车道数量。
right_lane_num	int	yes	右侧车道数量。
ramp_lane_num	int	yes	匝道车道数量。
main_speed	speed	yes	主路的限速值。
ramp_speed	speed	yes	匝道的限速值。
radius_of_curvature	length	yes	匝道的曲率半径。
ramp_length	length	yes	匝道的长度。
road_aids_type	road_aids_type	yes	道路辅助设施类型，匝道合流时匝道和主路的连接方式，目前共有三种：DType-1（直接式1）、DType-2（直接式2）和PType（平行式）。

- **参数取值范围：**

```
lane_width: length = [3m..4m]
left_lane_num: int = [0, 1, 2, 3, 4]
right_lane_num: int = [1, 2, 3, 4]
ramp_lane_num: int = [1, 2]
```



```
main_speed: speed = [80kph..120kph]
ramp_speed: speed = [40kph..60kph]
radius_of_curvature: length = [150m..1000m]
ramp_length: length = [200m..500m]
road_aids_type: road_aids_type = ["DType-1", "DType-2", "PType"]
```

keep创建

```
my_road_aids_type: road_aids_type = "DType-1"
my_merge: merge with:
    keep(it.lane_width == 3m)
    keep(it.left_lane_num == 0)
    keep(it.right_lane_num == 3)
    keep(it.ramp_lane_num == 1)
    keep(it.main_speed == 120kph)
    keep(it.ramp_speed == 60kph)
    keep(it.radius_of_curvature == 200m)
    keep(it.ramp_length == 250m)
    keep(it.road_aids_type == my_road_aids_type)
```

create创建

```
my_road_aids_type: road_aids_type = "DType-1"
my_merge: merge = scenery.create_merge(lane_width: 3m, left_lane_num: 0, right_lane_num: 3,
main_speed: 120kph, ramp_speed: 60kph, radius_of_curvature: 200m, ramp_length: 250m, road_aids_type:
my_road_aids_type)
```

7.8.2.2.3 split 匝道分流

- **用途**：创建split高速匝道合流的静态场景（地图）
- **参数**：参数如下表

表 7-48 split 参数

Parameter	Type	Mandatory	Description
lane_width	length	yes	每个车道宽度
left_lane_num	int	yes	左侧车道数量
right_lane_num	int	yes	右侧车道数量
ramp_lane_num	int	yes	匝道车道数量
main_speed	speed	yes	主路的限速值
ramp_speed	speed	yes	匝道的限速值
radius_of_curvature	length	yes	匝道的曲率半径
ramp_length	length	yes	匝道的长度

Parameter	Type	Mandatory	Description
road_aids_type	road_aids_type	yes	道路辅助设施类型，匝道分流时匝道和主路的连接方式，目前共有三种：DType-1（直接式1）、DType-2（直接式2）和PType（平行式）。

- **参数取值范围：**

```
lane_width: length = [3m..4m]
left_lane_num: int = [0, 1, 2, 3, 4]
right_lane_num: int = [3, 4]
ramp_lane_num: int = [1, 2]
main_speed: speed = [80kph..120kph]
ramp_speed: speed = [40kph..60kph]
radius_of_curvature: length = [150m..1000m]
ramp_length: length = [200m..500m]
road_aids_type: road_aids_type = ["DType-1", "DType-2", "PType"]
```

keep创建

```
my_road_aids_type: road_aids_type = "DType-1"
my_merge: merge with:
    keep(it.lane_width == 3m)
    keep(it.left_lane_num == 0)
    keep(it.right_lane_num == 3)
    keep(it.ramp_lane_num == 1)
    keep(it.main_speed == 120kph)
    keep(it.ramp_speed == 60kph)
    keep(it.radius_of_curvature == 200m)
    keep(it.ramp_length == 250m)
    keep(it.road_aids_type == my_road_aids_type)
```

create创建

```
my_road_aids_type: road_aids_type = "DType-1"
my_merge: merge = scenery.create_merge(lane_width: 3m, left_lane_num: 0, right_lane_num: 3,
main_speed: 120kph, ramp_speed: 60kph, radius_of_curvature: 200m, ramp_length: 250m, road_aids_type:
my_road_aids_type)
```

说明

- merge和split场景使用的road_aids_type为enum类型，详见附录Enum lists的**road_aids_type**一节。
- 在road_aids_type为"DType-2"时，right_lane_num必须大于ramp_lane_num。
- 如果希望车道属性需要搭建同一方向，不能存在left和right属性同时出现，可以将left_lane_num设置为0。

7.8.2.2.4 junction 路口

- **用途**：创建junction城区路口的静态场景（地图）
- **参数**：参数如下表

Parameter	Type	Mandatory	Description
lane_num	int	yes	单方向车道数量。
bikeway	bool	yes	是否有自行车道。

Parameter	Type	Mandatory	Description
sidewalk	bool	yes	是否有人行道。
junction_type	junction_type	yes	路口的类型，目前支持两种类型crossroad（十字路口）和T-junction（丁字路口）。

- **参数取值范围：**

```
lane_num: int = [1, 2, 3, 4]
bikeway: bool = [true, false]
sidewalk: bool = [true, false]
junction_type: junction_type = ["crossroad", "T-junction"]
```

keep创建

```
my_junction_type: junction_type = "crossroad"
my_junction: junction with:
    keep(it.lane_num == 2)
    keep(it.bikeway == false)
    keep(it.sidewalk == false)
    keep(it.junction_type == my_junction_type)
```

create创建

```
my_junction_type: junction_type = "crossroad"
my_junction: junction = scenery.create_junction(lane_num: 2, bikeway: false, sidewalk: false, junction_type: my_junction_type)
```

7.8.2.2.5 one_way_junction 单行线路口

单行线路口是指该路口的所有road均为单行线，比如一个普通的十字路口由8条road组成.

- **用途**：创建junction城区路口的静态场景（地图）
- **参数**：参数如下表.

Parameter	Type	Mandatory	Description
lane_num	int	yes	单方向车道数量.
bikeway	bool	yes	是否有自行车道.
sidewalk	bool	yes	是否有人行道.
junction_type	junction_type	yes	路口的类型，目前支持两种类型crossroad（十字路口）和T-junction（丁字路口）

- **参数取值范围：**

```
lane_num: int = [1, 2, 3, 4]
bikeway: bool = [true, false]
sidewalk: bool = [true, false]
junction_type: junction_type = ["crossroad", "T-junction"]
```

keep创建

```
my_junction_type: junction_type = "crossroad"
my_junction: one_way_junction with:
    keep(it.lane_num == 2)
```



```
    keep(it.bikeway == false)
    keep(it.sidewalk == false)
    keep(it.junction_type == my_junction_type)
```

7.8.2.3 静态场景样例

下文提供keep创建和create创建两种写法。

声明所有要泛化的变量，即为本例中第2-4行。

关键字merge说明是匝道合流的种子场景，即为本例中的第6行。

明确本场景中所有参数的具体值，即为本例中的第7行到最后。

```
scenario HighwayMerge:
    lane_width: length = [3m, 4m]
    left_lane_num: int = [0]
    ramp_length: length = [200m, 500m]
    road_aids_type: road_aids_type == "DType-1"

    merge_1: merge with:
        keep(it.lane_width == lane_width)
        keep(it.left_lane_num == left_lane_num)
        keep(it.right_lane_num == 2)
        keep(it.ramp_lane_num == 1)
        keep(it.main_speed == 120kph)
        keep(it.ramp_speed == 60kph)
        keep(it.radius_of_curvature == 200m)
        keep(it.ramp_length == ramp_length)
        keep(it.road_aids_type == road_aids_type)
```

声明本场景为静态场景，即为本例中第2行。

声明所有要泛化的变量，即为本例中第4行。

create_merge说明是匝道合流的种子场景，即为本例中的第6行。

函数create_merge的入参即指定本场景中的所有参数具体值，即为本例中的第6-7行。

```
scenario Merge:
    m_scene: scenery

    lane_width: length = [3m, 4m]
    my_road_aids_type: road_aids_type == "DType-1"

    m_straight: merge = m_scene.create_merge(lane_width: lane_width, left_lane_num: 0, right_lane_num: 2,
main_speed: 120kph, ramp_speed: 60kph, radius_of_curvature: 200m, ramp_length: 300m, ramp_lane_num:
1, road_aids_type: my_road_aids_type)
```

7.8.2.4 附录

7.8.2.4.1 Enum Lists

road_aids_type

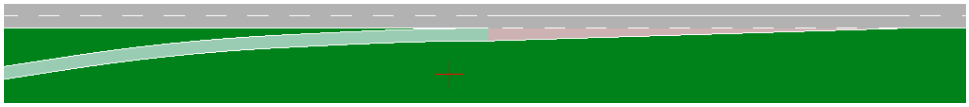
匝道类型，用于静态场景的**split场景**和**merge场景**。

road_aids_type list

```
ENUM_ROAD_AIDS_TYPE = ("DType-1", "DType-2", "PType")
```

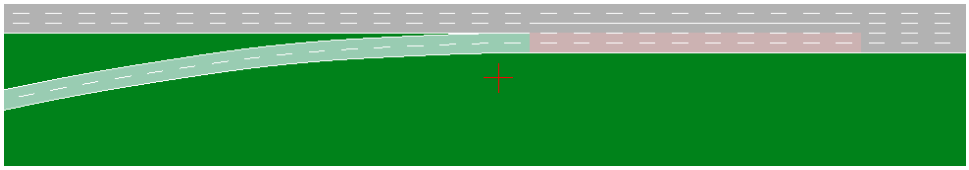
- DType-1（直接式1）：

  DType-1（直接式1）是指匝道合入（分流）时直接汇入（离开）主路，并且匝道部分逐渐收缩（扩展）直到合入（离开）主路，如下图所示：



- DType-2（直接式2）：

  DType-2（直接式2）是指匝道合入（分流）时直接汇入（离开）主路，主路的车道数也随之发生变化，匝道部分不进行收缩（扩展），如下图所示：



- PType（平行式）：

  PType（平行式）是指匝道合入（分流）时先有一段平行于主路，然后匝道再逐渐收缩（扩展）直到合入（离开）主路，如下图所示：

junction_type

junction（交叉口）类型，用于静态场景的**junction场景**

road_aids_type list

```
ENUM_ROAD_AIDS_TYPE = ("DType-1", "DType-2", "PType")
```

- crossroad：十字路口
- T-junction：丁字路口

7.8.3 动静态配套样例

7.8.3.1 种子地图的逻辑场景样例（仿真器 B）

配合静态场景的种子场景，在本节提供了对应的适配仿真器B的逻辑场景样例。

7.8.3.1.1 straight

简述：地图场景为直道。lead_vehicle和主车Ego在主道上分别以40kph和Ego_InitSpeed_Ve0的初始速度一前一后行驶，Ego设定了目标在主道右2车道上的目标点Target_position，同时激活Ego控制器（控制器会影响Ego去往Target_position的寻路算法，但目前仿真器B尚不支持寻路动作acquire_position），控制器有时会根据lead_vehicle的位置更改主车Ego的速度。

地图文件（odr）

```
scenario Straight:
    m_scene: scenery

    lane_width: length = [3m..4m]
    right_lane_num: int = [2, 3]
    bikeway: bool = [true, false]
    sidewalk: bool = [true, false]
    main_speed: speed = 60kph
    road_length: length = [550m, 600m]
```

```
    straight_1: straight with:
        keep(it.lane_width == lane_width)
        keep(it.left_lane_num == 0)
        keep(it.right_lane_num == right_lane_num)
        keep(it.bikeway == bikeway)
        keep(it.sidewalk == sidewalk)
        keep(it.main_speed == main_speed)
        keep(it.road_length == road_length)
```

场景文件（osc）

```
import standard

scenario Straight:
    # map
    map: map
    map.set_map_file("./straight.odr")

    # parameter
    Ego_InitSpeed_Ve0: speed = [55kph..60kph]
    Ego_InitPosition_LaneId: string = ['-1', '-2']
    Ego_InitPosition_s: length = [0m..30m]
    Ego_Odr: odr_point = map.create_odr_point(road_id: '1', lane_id: Ego_InitPosition_LaneId, s:
Ego_InitPosition_s, t: 0.0m)
    Ego_InitPosition: pose_3d with:
        keep(it.odr_point == Ego_Odr)
    m_distance: length = [50m..80m]
    LeadVehicle_Odr: odr_point = map.create_odr_point(road_id: '1', lane_id: '-1', s: m_distance, t: 0.0m)
    LeadVehicle_InitPosition: pose_3d with:
        keep(it.odr_point == LeadVehicle_Odr)
    Target_xyz: xyz_point = map.create_xyz_point(x: 450m, y: -4.5m ,z: 0.0m)
    Target_position: pose_3d with:
        keep(it.xyz_point == Target_xyz)
    Duration: time = 100s

    # entity
    Ego: vehicle with:
        keep(it.name == "Saimo")
        keep(it.initial_bm == "默认驾驶员")
    lead_vehicle: vehicle with:
        keep(it.name == "Saimo")
        keep(it.initial_bm == "默认驾驶员")

    # storyboard
    do parallel(duration: Duration):
        # init
        Ego.assign_init_position(position: Ego_InitPosition)
        Ego.assign_init_speed(Ego_InitSpeed_Ve0)
        lead_vehicle.assign_init_position(position: LeadVehicle_InitPosition)
        lead_vehicle.assign_init_speed(40kph)

        Ego.activate_controller(true, true)
        Ego.acquire_position(target: Target_position)
```

7.8.3.1.2 merge

简述：地图场景为匝道合流。主车Ego在主道行驶，初始速度为Ego_InitSpeed_Ve0，Ego设定了目标在主道右侧2车道上的目标点Target_position，仿真开始后激活Ego控制器（控制器会影响Ego去往Target_position的寻路算法，但目前仿真器B尚不支持寻路动作acquire_position），从车side_vehicle在匝道行驶，初始速度为SideVehicle_InitSpeed_Ve0.side_vehicle从匝道汇入主道。控制器有时会根据side_vehicle的位置更改主车Ego的速度。

地图文件（odr）

```
scenario Merge:
    m_scene: scenery
```



```
    lane_width: length = [3m, 4m]
    radius_of_curvature: length = [200m..1000m]
    ramp_lane_num: int = 1
    ramp_length: length = [200m..300m]
    main_speed: speed = 120kph
    ramp_speed: speed = 60kph
    road_aids_type: road_aids_type = ["DType-1", "DType-2", "PType"]

    merge_1: merge with:
        keep(it.lane_width == lane_width)
        keep(it.left_lane_num == 0)
        keep(it.right_lane_num == 2)
        keep(it.ramp_lane_num == ramp_lane_num)
        keep(it.main_speed == main_speed)
        keep(it.ramp_speed == ramp_speed)
        keep(it.radius_of_curvature == radius_of_curvature)
        keep(it.ramp_length == ramp_length)
        keep(it.road_aids_type == road_aids_type)
```

场景文件（osc）

```
import standard

scenario Merge:
    # map
    map: map
    map.set_map_file("./merge.odr")

    # parameter
    Ego_InitSpeed_Ve0: speed = [90kph..110kph]
    Ego_InitPosition_LaneId: string = ['-1', '-2']
    Ego_InitPosition_s: length = [0m..30m]
    Ego_Odr: odr_point = map.create_odr_point(road_id: '10', lane_id: Ego_InitPosition_LaneId, s:
Ego_InitPosition_s, t: 0.0m)
    Ego_InitPosition: pose_3d with:
        keep(it.odr_point == Ego_Odr)
    SideVehicle_InitSpeed_Ve0: speed = [45kph, 50kph, 55kph]
    SideVehicle_s: length = [30.0m..80.0m]
    SideVehicle_Odr: odr_point = map.create_odr_point(road_id: '1', lane_id: '-1', s: SideVehicle_s, t: 0.0m)
    SideVehicle_InitPosition: pose_3d with:
        keep(it.odr_point == SideVehicle_Odr)
    Target_xyz: xyz_point = map.create_xyz_point(x: 530m, y: -2m ,z: 0.0m)
    Target_position: pose_3d with:
        keep(it.xyz_point == Target_xyz)
    Duration: time = 100s

    # entity
    Ego: vehicle with:
        keep(it.name == "Saimo")
        keep(it.initial_bm == "默认驾驶员")
    side_vehicle: vehicle with:
        keep(it.name == "Saimo")
        keep(it.initial_bm == "默认驾驶员")

    # storyboard
    do parallel(duration: Duration):
        # init
        Ego.assign_init_position(position: Ego_InitPosition)
        Ego.assign_init_speed(Ego_InitSpeed_Ve0)
        side_vehicle.assign_init_position(position: SideVehicle_InitPosition)
        side_vehicle.assign_init_speed(SideVehicle_InitSpeed_Ve0)

        Ego.activate_controller(true, true)
        Ego.acquire_position(target: Target_position)
```

7.8.3.1.3 split

简述：地图场景为匝道分流。lead_vehicle和主车Ego在主道的同一车道上分别以35kph和Ego_InitSpeed_Ve0的初始速度一前一后行驶，Ego设定了目标在匝道上的目

标点Target_position，仿真开始后激活Ego控制器（控制器会影响Ego去往Target_position的寻路算法，但目前仿真器B尚不支持寻路动作acquire_position）。控制器有时会根据lead_vehicle的位置更改主车Ego的速度。

须知

使用xyz坐标创建终点时，由于匝道地图泛化会使终点偏移，建议在创建测评任务时为检测终点设置合适的半径，例如"到达半径5m"。

地图文件（odr）

```
scenario Split:
    m_scene: scenery

    lane_width: length = [3m..4m]
    main_speed: speed = 120kph
    ramp_speed: speed = 60kph
    ramp_length: length = [300m..350m]
    road_aids_type: road_aids_type = "DType-2"

    split_1: split with:
        keep(it.lane_width == lane_width)
        keep(it.left_lane_num == 0)
        keep(it.right_lane_num == 3)
        keep(it.ramp_lane_num == 1)
        keep(it.main_speed == main_speed)
        keep(it.ramp_speed == ramp_speed)
        keep(it.radius_of_curvature == 200m)
        keep(it.ramp_length == ramp_length)
        keep(it.road_aids_type == road_aids_type)
```

场景文件（osc）

```
import standard

scenario Split:
    # map
    map: map
    map.set_map_file("./split.odr")

    # parameter
    Ego_InitSpeed_Ve0: speed = [90kph..110kph]
    Ego_InitPosition_LaneId: string = ['-1', '-2']
    Ego_InitPosition_s: length = [0m..100m]
    Ego_Odr: odr_point = map.create_odr_point(road_id: '10', lane_id: Ego_InitPosition_LaneId, s:
Ego_InitPosition_s, t: 0.0m)
    Ego_InitPosition: pose_3d with:
        keep(it.odr_point == Ego_Odr)
    m_InitDistance: length = [60m..100m]
    Target_xyz: xyz_point = map.create_xyz_point(x: 445m, y: -46.5m ,z: 0.0m)
    Target_position: pose_3d with:
        keep(it.xyz_point == Target_xyz)
    Duration: time = 100s

    # entity
    Ego: vehicle with:
        keep(it.name == "Saimo")
        keep(it.initial_bm == "默认驾驶员")
    lead_vehicle: vehicle with:
        keep(it.name == "Saimo")
        keep(it.initial_bm == "默认驾驶员")

    # storyboard
    do parallel(duration: Duration):
        # init
```

```
    Ego.assign_init_position(position: Ego_InitPosition)
    Ego.assign_init_speed(Ego_InitSpeed_Ve0)
    lead_vehicle.assign_init_position() with:
        lane(same_as: Ego)
        position(distance: m_InitDistance, ahead_of: Ego)
    lead_vehicle.assign_init_speed(35kph)

    Ego.activate_controller(true, true)
    Ego.acquire_position(target: Target_position)
```

7.8.3.1.4 junction

简述：地图场景为交叉口。lead_vehicle和主车Ego一前一后分别以LeadVehicle_TargetSpeed_Ve0和Ego_TargetSpeed_Ve0的初始速度向交叉口行驶，Ego设定了目标在右转车道上的目标点Target_position，仿真开始后激活Ego控制器（控制器会影响Ego去往Target_position的寻路算法，但目前仿真器B尚不支持寻路动作acquire_position）。另一路段上有一辆车vehicle1，也以LeadVehicle_TargetSpeed_Ve0的速度朝交叉口行驶。控制器有时会根据环境车的位置更改主车Ego的速度。

地图文件（odr）

```
scenario Junction:
    m_scene: scenery

    lane_num: int = [2, 3, 4]
    bikeway: bool = false
    sidewalk: bool = false
    junction_type: junction_type = ["crossroad", "T-junction"]

    junction_1: junction with:
        keep(it.lane_num == lane_num)
        keep(it.bikeway == bikeway)
        keep(it.sidewalk == sidewalk)
        keep(it.junction_type == junction_type)
```

场景文件（osc）

```
import standard

scenario Junction:
    # map
    map: map
    map.set_map_file("./junction.odr")

    # parameter
    Ego_TargetSpeed_Ve0: speed = [50kph..60kph]
    Ego_InitPosition_RoadId: string = '1'
    Ego_InitPosition_LaneId: string = ['-1', '-2']
    Ego_InitPosition_s: length = [0m..100m]
    Ego_Odr: odr_point = map.create_odr_point(road_id: Ego_InitPosition_RoadId, lane_id:
Ego_InitPosition_LaneId, s: Ego_InitPosition_s, t: 0.0m)
    Ego_InitPosition: pose_3d with:
        keep(it.odr_point == Ego_Odr)
    Target_xyz: xyz_point = map.create_xyz_point(x: 225m, y: -100m ,z: 0.0m)
    Target_position: pose_3d with:
        keep(it.xyz_point == Target_xyz)
    LeadVehicle_TargetSpeed_Ve0: speed = 45kph
    Vehicle1_Odr: odr_point = map.create_odr_point(road_id: '2', lane_id: '2', s: 100.0m, t: 0.0m)
    m_orientation: orientation_3d with:
        keep(it.roll == 0.0rad)
        keep(it.pitch == 0.0rad)
        keep(it.yaw == -1.57rad)
    Vehicle1_InitPosition: pose_3d with:
        keep(it.odr_point == Vehicle1_Odr)
        keep(it.orientation == m_orientation)
    m_profile: dynamics_shape = [sinusoidal, linear, step]
```



```
m_rate_peak: acceleration = [5kmphps, 10kmphps]
Duration: time = [100s, 120s]

# entity
Ego: vehicle with:
    keep(it.name == "Saimo")
    keep(it.initial_bm == "默认驾驶员")
lead_vehicle: vehicle with:
    keep(it.name == "Saimo")
    keep(it.initial_bm == "默认驾驶员")
vehicle1: vehicle with:
    keep(it.name == "Saimo")
    keep(it.initial_bm == "默认驾驶员")

# storyboard
do parallel(duration: Duration):
    # init
    Ego.assign_init_position(position: Ego_InitPosition)
    Ego.assign_init_speed(Ego_TargetSpeed_Ve0)
    lead_vehicle.assign_init_position() with:
        lane(same_as: Ego)
        position(distance: 30.0m, ahead_of: Ego)
    lead_vehicle.assign_init_speed(LeadVehicle_TargetSpeed_Ve0)
    vehicle1.assign_init_position(position: Vehicle1_InitPosition)
    vehicle1.assign_init_speed(LeadVehicle_TargetSpeed_Ve0)

    Ego.activate_controller(true, true)
    Ego.acquire_position(target: Target_position)
```

7.9 采样方式介绍

蒙特卡洛采样

蒙特卡洛采样是一种简单的随机抽样，根据概率分布进行采样，如对样本服从μ=0，δ=1的正态分布，通过蒙特卡洛采样进行采样，采样得到的点能满足正态分布要求，如下图所示，采样得到的点会集中μ=0附近，要想采样得到更边界的点，需要进行大量采样。

图 7-16 蒙特卡洛采样

拉丁超立方采样

拉丁超立方采样的目的是用较少的采样次数，来达到与多次蒙特卡洛采样相同的结果，并且涵盖更全面的边界点。

如下图所示，同样对于μ=0，δ=1的正态分布，可以利用更少的采样点得到相同的分布，并且不会产生明显的聚集现象，边界值也能更容易获取到。

图 7-17 拉丁超立方采样

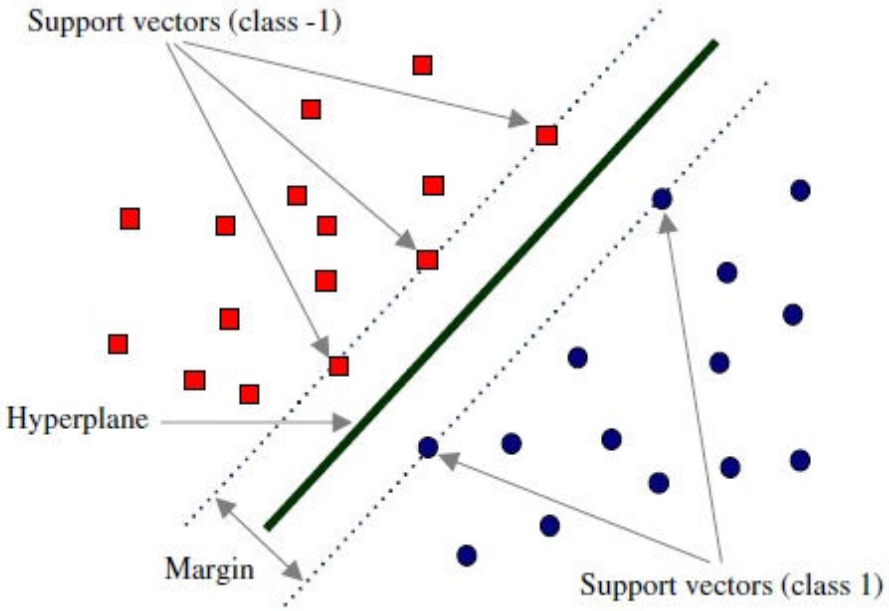
联合概率分布采样

联合概率分布采样假设连续型参数符合正态分布，支持录入连续型参数之间的相关系数（值为1时，表示变量完全正相关。值为0时，表示变量间独立。值为-1时，表示变量完全负相关），并根据参数分布和相关系数进行联合概率分布采样。而离散型参数根据给定的取值列表进行随机采样。

重要型采样

重要性采样是在优化目标边界附近进行采样，利用上一次泛化场景仿真后得到的评测分数进行训练拟合，找到边界后不断在边界附近进行采样。

图 7-18 重要型采样



8 智驾模型服务

8.1 智驾模型简介

通过与AI模型、大模型的结合，提供高精度自动标注能力，大幅度降低传统人工标注数据真值的成本。提供场景数据集生成能力，帮助自动驾驶模型训练快速扩充数据集，低成本获取难例数据集。提供多模态场景理解和检索能力，帮助客户在海量样本库快速、智能的分类和检索。

智驾模型授权操作步骤

步骤1 数据准备。

- 开通相应服务并购买套餐包，操作请参考**1.5.6 开通智驾模型微调服务**、**1.5.7 开通我的模型和购买套餐包**。
- 在使用模型微调、2D图像生成等功能之前，需要用户授权访问用户的OBS对象存储服务。

  说明

  开通我的模型服务后，当没有购买套餐包，平台将按照任务实际调用次数按需计费。

步骤2 进入Octopus控制台，鼠标悬停界面右上角用户名。

步骤3 单击“统一身份认证”，进入统一身份认证服务。

步骤4 选择“委托 > 创建委托”，填写委托参数。

- 委托名称：自定义名称，建议为： octopus_obs_agency。
- 委托类型：选择“云服务”。
- 云服务：选择“自动驾驶云服务”。
- 持续时间：可自定义持续时间。
- 描述：简要描述委托信息。

步骤5 单击“完成”，在创建成功的弹框里单击“立即授权”。

步骤6 在授权页面，单击“新建策略”，填写新建策略的参数。

- 策略名称：自定义名称，建议为： octopus_obs_agency_policy。
- 策略配置方式：选择“JSON视图”。

- 策略内容：配置为如下内容。

```
{
    "Version": "1.1",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "obs:object:GetObject",
                "obs:object:DeleteObject",
                "obs:bucket:ListAllMyBuckets",
                "obs:bucket:ListBucket",
                "obs:object:PutObject"
            ]
        }
    ]
}
```

- 策略描述：简要描述策略信息。

步骤7 单击“下一步”，在策略创建成功确认弹框中单击“确定”。勾选新建的策略，单击“下一步”。

步骤8 选择授权范围方案后，单击“确定”按钮后根据界面提示确认提示内容。

步骤9 授权后需等待15-30分钟才可生效。

步骤10 单击“完成”，授权完成。

在“授权记录”界面，可查看授权记录。

步骤11 返回“模型微调”或“2D图像生成”等功能界面，单击委托名称后的“修改”，填写并确认新建的委托名称。

----结束

8.2 多模态检索

多模态检索通过大模型算法，对用户视频库进行特征提取，再通过多模态视频检索功能，检索出满足文本或者视频搜索条件的视频片段。

视频库配置

当前一个用户只支持配置一个视频库。

步骤1 在服务控制台“总览 > 我的模型”区域，开通“多模态检索”服务。

步骤2 在左侧菜单栏中单击“智驾模型服务 > 多模态检索”。

步骤3 单击“视频库配置”，输入基本信息。

- 名称：只能包含数字、英文、中文、下划线、中划线，输入长度不得超过64个字符。
- 描述：简要描述任务，不能包含“@^\#$%&*<>'|"/`”，输入长度不能超过256个字符。
- 视频文件：请选择视频库的路径。

步骤4 单击“生成视频特征”，平台根据大模型算法自动生成视频特性。生成新视频特征会覆盖原视频库信息，请谨慎操作。

----结束

视频检索

步骤1 在左侧菜单栏中单击“智驾模型服务 > 多模态检索”。

步骤2 单击多模态检索任务操作栏中的“检索”，在检索弹出框中填写基本信息。

- 检索输入：
  - 当检索类型为“文本”时，直接输入有效检索文本。
  - 当检索类型为“视频”时，上传一个视频文件，文件大小不能超过5MB。
- 检索类型：可选择“文本”和“视频”。
- 检索结果个数：设置检索个数。每次最多可以检索出100个结果。

步骤3 单击“检索”，将从视频库中按置性率检索出符合检索特征的视频路径及视频片段。

步骤4 单击“预览”，预览检索视频片段。

步骤5 单击“保存检索结果CSV文件”，将搜索结果下载到本地保存。

----结束

8.3 模型微调

模型微调是在不依赖训练服务的情况下对模型进行二次训练，达到二次调优的效果。调优后的迭代模型可以在智驾模型管理列表中查看。

新建模型微调任务

步骤1 完成OBS授权委托，具体操作步骤请参考**授权操作步骤**。

步骤2 在服务控制台“总览 > 模型微调”，开通“智驾模型微调”服务。

步骤3 在左侧菜单栏中选择“智驾模型服务 > 模型微调”。

步骤4 单击“新建任务”。在新建模型微调任务页面，填写基本信息。

- 名称：只能包含数字、英文、中文、下划线、中划线，输入长度不得超过64个字符。
- 描述：简要描述任务，不能包含“@^\#$%&*<>'|"/`”，输入长度不能超过256个字符。

步骤5 完成模型配置。

- 模型级别：可选择“初始模型”和“迭代模型”。“初始模型”为Octopus平台提供的内置模型，“迭代模型”是用户二次微调后的模型。
- 输入模型：选择需要进行微调的模型和版本。初始模型提供的内置模型如下：

表 8-1 Octopus 平台提供的内置模型

模型名称	说明
hw_model_3d_detection	3D检测模型。
hw_model_2dgen_weather	2D图像生成-天气模型。
hw_model_autolabel	2D全景分割模型。

模型名称	说明
hw_model_4dbev_lane_line_detection	车道线预标注模型。
hw_model_4dbev_road_mark_detection	路面预标识模型。
hw_model_2d_3d_detection_fusion	2D3D融合检测模型。

- 输出模型：模型微调后存储的模型仓库。
- 模型版本描述：简要描述模型版本，不能包含“@^\#$%&*<>'|"/`”，输入长度不能超过256个字符。

步骤6 完成算法配置。

- 参数列表：选择输入模型及版本后，将自动关联显示参数列表信息。支持自定义value值。
- 环境变量列表：选择输入模型及版本后，将自动关联显示环境变量列表信息。用户可以自定义value值。

  当预标注模型的key为semantic_categories、instance_categories时，value值是数据集中所有的标注类型名称（允许中文，不允许遗漏某个标注类型），按语义分割和实例分割，分开统计后的值。示例如下：

  - instance_categories：“交通警示物,骑行者,其它车辆,两轮车,消防栓,柱子,地面标识,限位块,行人,车辆,地锁,减速带”
  - semantic_categories：“可行驶区域,路沿,车道线,车位线”
- 数据集：选择训练的数据集。最多支持选择5个数据集。数据集中的标注数据必须是八爪鱼格式的。格式参考：**车道线预标注模型和路面预标识模型数据集格式**。
- 资源规格：选择训练使用的资源池和资源规格。当前支持ModelArt资源池的Vnt1、Ant8、pool-octopus-ma-gpu-ant03、octopus-v100规格。

步骤7 单击“确认”，下发模型微调任务。

----结束

模型微调任务相关操作

在模型微调任务列表页，还可以完成以下操作。

表 8-2 模型微调任务管理相关操作

任务	操作步骤
查看任务详情	单击任务名称，在任务详情页面查看模型微调任务详情，包括任务基本信息、参数详情以及任务日志。
停止任务	当任务状态为“排队中”或“运行中”时，可单击操作栏内的“停止”，进行停止任务操作。
复制任务	单击操作栏内的“复制”，可基于现有任务创建新任务。

任务	操作步骤
单个删除任务	单击操作栏内的“删除”可单个删除任务。模型微调任务删除后无法恢复，请谨慎操作。
批量删除任务	勾选名称前面的复选框，单击列表上方的“删除”，可批量删除任务。模型微调任务删除后无法恢复，请谨慎操作。
查询模型微调任务	在搜索输入框中输入搜索条件，按回车键即可查询目标任务。

车道线预标注模型和路面预标识模型数据集格式

当模型微调选择hw_model_4dbev_road_mark_detection或是hw_model_4dbev_lane_line_detection时的输入数据格式。

```
|--- TestSet
   |--- folder_0
      |--- 4DAnnotation
         |--- timestamps0_timestamps1.json
      |--- pointcloud_fusion
         |--- timestamps0_timestamps1.pcd
         |--- Metadata.xml
      |--- pose
         |--- lidar_pose.txt
   |--- folder_1
      |--- 4DAnnotation
         |--- timestamps0_timestamps1.json
      |--- pointcloud_fusion
         |--- timestamps0_timestamps1.pcd
         |--- Metadata.xml
      |--- pose
         |--- lidar_pose.txt
   |--- ...

|---TrainningSet
   |--- folder_0
      |--- 4DAnnotation
         |--- timestamps0_timestamps1.json
      |--- pointcloud_fusion
         |--- timestamps0_timestamps1.pcd
         |--- Metadata.xml
      |--- pose
         |--- lidar_pose.txt
   |--- folder_1
      |--- 4DAnnotation
         |--- timestamps0_timestamps1.json
      |--- pointcloud_fusion
         |--- timestamps0_timestamps1.pcd
         |--- Metadata.xml
      |--- pose
         |--- lidar_pose.txt
   |--- ...

|---ValidationSet
   |--- folder_0
      |--- 4DAnnotation
         |--- timestamps0_timestamps1.json
      |--- pointcloud_fusion
         |--- timestamps0_timestamps1.pcd
         |--- Metadata.xml
      |--- pose
         |--- lidar_pose.txt
   |--- folder_1
```



```
    |--- 4DAnnotation
        |--- timestamps0_timestamps1.json
    |--- pointcloud_fusion
        |--- timestamps0_timestamps1.pcd
        |--- Metadata.xml
    |--- pose
        |--- lidar_pose.txt
|--- ...
```

2D3D 融合检测模型数据集格式

- 当模型微调选择hw_model_2d_3d_detection_fusion时的输入格式：

```
|--- raw-data
    |--- clip_folder1
        |--- data  #数据文件夹
            |--- <timestamps1> #一帧数据
                                |--- <pcd_timestamp>.pcd  #点云文件
                                |--- <pcd_timestamp>.json  #真值结果文件
                                |--- front.jpg #前视
                                |--- rear.jpg #后视
                                |--- left_backward.jpg #左后
                                |--- left_forward.jpg #左前
                                |--- right_backward.jpg #右后
                                |--- right_forward.jpg #右前
                        |--- ... ...
        |--- param  #内外参文件夹
            |--- front.yaml  #前视相机内外参
            |--- left_backward.yaml  #左后相机内外参
                        |--- left_forward.yaml #左前相机内外参
            |--- right_backward.yaml #右后相机内外参
            |--- right_forward.yaml #右前相机内外参
                        |--- rear.yaml #后视相机内外参
            |--- lidar.yaml  #雷达外参

    |--- clip_folder2
                |--- data
            |--- <timestamps1>
                                |--- <pcd_timestamp>.pcd
                                |--- <pcd_timestamp>.json
                                |--- front.jpg
                                |--- rear.jpg
                                |--- left_backward.jpg
                                |--- left_forward.jpg
                                |--- right_backward.jpg
                                |--- right_forward.jpg
                        |--- ... ...
        |--- param
            |--- front.yaml
            |--- left_backward.yaml
                        |--- left_forward.yaml
            |--- right_backward.yaml
            |--- right_forward.yaml
                        |--- rear.yaml
            |--- lidar.yaml

    |--- ...
```

- 相机内外参定义

```
相机内外参文件名：camview.yaml, camview从白名单中选取；
相机内参包括：内参和畸变参数；
相机外参定义为：相机坐标系到IMU坐标系的变换矩阵；
<camview>.yaml文件定义如下：
---
cameraType: # 0-perspective, 1-fisheye
  type: 0
K: #opencv-matrix
  rows:3
  cols:3
  dt:1
```



```
  data:[1.0, 0.0, 1.0, 0.0, 1.0, 1.0, 0.0, 0.0, 1.0] # fx,fy,cx,cy为相机焦距和主点
D: #opencv-matrix
  rows: 4 #根据实际畸变参数个数填写
  cols: 1
  dt: 1
  data: [0.0, 0.0, 0.0, 0.0]  #填入实际的畸变参数
Imagesize: [1, 1] #填入图像宽，高
Transformation: # Camera to IMU
  rows: 4
  cols: 4
  dt: 1.0
  data: [0.0, 0.0, 0.0, 0.0,
        0.0, 0.0, 0.0, 0.0,
        0.0, 0.0, 0.0, 0.0,
        0.0, 0.0, 0.0, 1.0] #请填入转换矩阵的实际值
```

- 激光雷达内外参定义：

```
文件名：lidar.yaml
外参定义：Lidar坐标系到IMU坐标系的变换矩阵
lidar.yaml文件内容定义：
---
Transformation:
   cols: 4
   rows: 4
   dt: d
   data: [0.0, 0.0, 0.0, 0.0,
        0.0, 0.0, 0.0, 0.0,
        0.0, 0.0, 0.0, 0.0,
        0.0, 0.0, 0.0, 1.0] #根据实际内容填写
```

8.4 2D 图像生成

步骤1 完成OBS授权委托，具体操作步骤请参考**授权操作步骤**。

步骤2 在服务控制台“总览 > 我的模型”区域，开通“2D图像生成”服务。

步骤3 在左侧菜单栏中选择“智驾模型服务 > 2D图像生成”。

步骤4 单击“新建任务”，填写信息。

- 名称：只包含英文，数字，下划线（ _ ）和中划线（ - ），并且以英文开头的名称，不得超过64个字符。
- 描述：简要描述2D图像，不包含“@^\#$%&*<>'|"/`”，不得超过500个字符。
- 输出途径：可手动输入或者下拉选择授权的OBS桶路径。OBS桶路径获取方法请参考**创建对象存储服务**。

步骤5 根据需要生成的图片设置对应的参数。

系统会根据设置的参数查询可选对象框模板，可以直接选择一个对象框模板，再在模板中选择对象框。如果系统未查到可选的对象框模板，建议尝试切换相机视角，切换对象类型或者减少对象数量。

图 8-1 创建 2D 图像参数

图 8-2 选择对象框模板

- 相机视角：选择相机视角。对象类型选择“地锁”时，只能选择“后视”视角。
- 时间段：选择时间段。
- 天气：选择天气。
- 批量生成数：设置生成图像的数量。
- 生成图像分辨率：选择生成图像的分辨率。
- 对象类型/对象数量：请选择对象类型，并设置对象的数量。支持同时选择3个对象类型，但不能同时选择“地锁”和其他类型的对象。

说明

- 当前控制台上地锁只支持指定单个对象框生成图像，其他类型支持多个对象框。
- API支持多个对象框。
- 人车模型最多支持17个框。

步骤6 单击“创建”，可在列表查看模型生成的2D图像。

----结束

2D 图像列表相关操作

在2D图像列表，还可以进行以下操作。

表 8-3 2D 图像列表相关操作

任务	操作步骤
搜索模型生成的2D图像	在搜索框中输入关键字搜索相关2D图像生成作业名称。
查看模型生成的2D图像详情	单击模型生成的2D图像生成作业名称，即可查看模型生成的2D图像详情页。 ● 显示位置框：可选择是否显示位置框。 ● 下载图片：鼠标悬停图片，可选择单张下载图片至本地。 ● 删除图片：可选择单张或批量删除图片。 ● 清理失效图片：如果有失效的图片，可选择单击右上角“清理失效图片”，清理失效图片。
终止模型生成2D图像	单击操作栏的“终止”，即可终止状态为“执行中”的模型。
删除模型生成的2D图像	单击操作栏的“删除”，即可删除模型生成的2D图像信息。 **说明** 删除后无法恢复，请谨慎操作（OBS中的图片数据需用户手动清理）。

8.5 2D 预标注

2D预标注默认使用服务内置的初始模型部署的在线服务，也可以修改“当前生效在线服务”，修改为模型微调后的迭代模型部署的“运行中”的在线服务。2D预标注当前支持多种预标注功能：

- 目标检测：主要用于鱼眼图片的预标注。
- 语义分割（混合）：支持鱼眼图片和普通图片的预标注。
- 车道线检测：能够快速标注车道线的位置和类别。

2D 预标注支持的标注类别

- 目标检测：

  可行驶区域、车道线、车位线、路沿、地面标识、减速带、消防栓、柱子、地锁、限位块、警示物、骑行者、行人、车辆、其他车辆、两轮车。
- 车道线检测：

  白虚线、黄虚线、白实线、黄实线、路沿线、停止线。
- 语义分割（混合）：

- 成年人、手推车、面包车、工程车、卡车、公交车、小汽车、骑行者。
- 直行右转箭头、右转导向箭头、左转导向箭头、直行导向箭头、路面字符、其它导向箭头。
- 停车线、禁停止线、斑马线、导流线、停止线、双实线、双虚线、虚线、黄实线、实线。
- 水马、锥桶、矮墙、隔音墙、路面栅栏、消防栓、路沿、减速带、柱子、井盖、限位块、地锁、杆状物、臂障。

创建 2D 预标注任务步骤

步骤1 在服务控制台“总览 > 我的模型”区域，开通“2D图像生成”服务。

步骤2 在左侧菜单栏中选择“智驾模型服务 > 2D预标注”。

步骤3 单击“添加文件”。

步骤4 选择预标注功能场景，然后单击“添加文件”，添加本地保存的文件，单击“确认”。

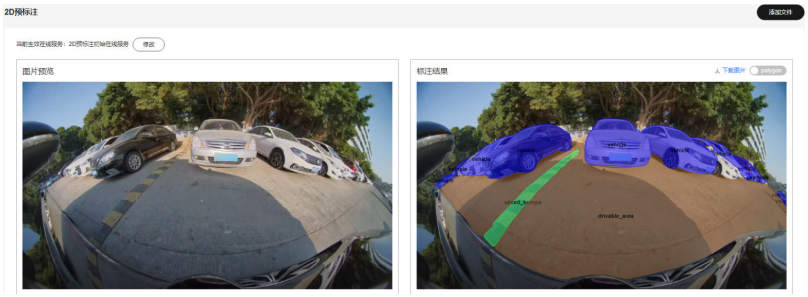
说明

- 图片格式：只能选择JPG/JPEG/PNG文件，图片大小不能超过7MB，且不能超过10,000,000像素。其中目标检测场景推荐上传鱼眼图片。
- 标注结果：目标检测和语义分割（混合）场景直接在原始图片叠加预标注结果显示，详情可参考下图。

步骤5 单击“开始识别”，生成相关返回值。以下为标注结果示例。

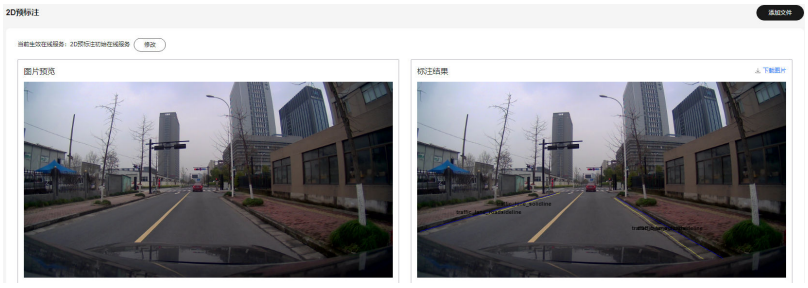
- **目标检测**

图 8-3 目标检测标注结果



- **车道线检测**

图 8-4 车道线检测标注结果



- **语义分割（混合）**

图 8-5 语义分割（混合）标注结果

步骤6 （可选）管理相关结果。

- 修改当前生效的在线服务：

  单击当前生效在线服务名称后的“修改”，重新选择在线服务，切换当前生效的在线服务。

- 下载标注结果后的图片：

  2D预标注结果生成后，单击标注结果右上角“下载图片”，可以将标注后的图片下载到本地。

----结束

8.6 3D 预标注

8.6.1 3D 预标注

3D预标注当前支持目标检测和目标分割两种标注功能。支持的3D预标注类别如下：

- 目标检测：行人、自行车、摩托车、卡车、公交车、小汽车。
- 目标分割：Pedestrian（行人）、Bicycle（自行车）、Motorcycle（摩托车）、Truck（卡车）、Bus（公交车）、Car（小汽车）、Trailer（拖车）、Construction vehicle（工程车）、Drivable surface（可行驶路面）、Terrain（地带）、Sidewalk（人行道）、Vegetation（草木）、Other flat（其他）、Barrier（路面栅栏）、Traffic cone（锥桶）、Manmade（建筑）。

创建 3D 预标注任务步骤

步骤1 在服务控制台“总览 > 我的模型”区域，开通“预标注”服务，具体操作步骤请参考**开通我的模型和购买套餐包**。

步骤2 在左侧菜单栏中选择“智驾模型服务 > 3D预标注”。

步骤3 选择“3D预标注”页签。

步骤4 单击“添加文件”，根据需要设置预标注功能，参考如下：

目标检测

- 预标注功能：此处选择“目标检测”。
- 添加文件：上传本地点云文件。只能选择PCD点云文件，文件大小不能超过7MB。

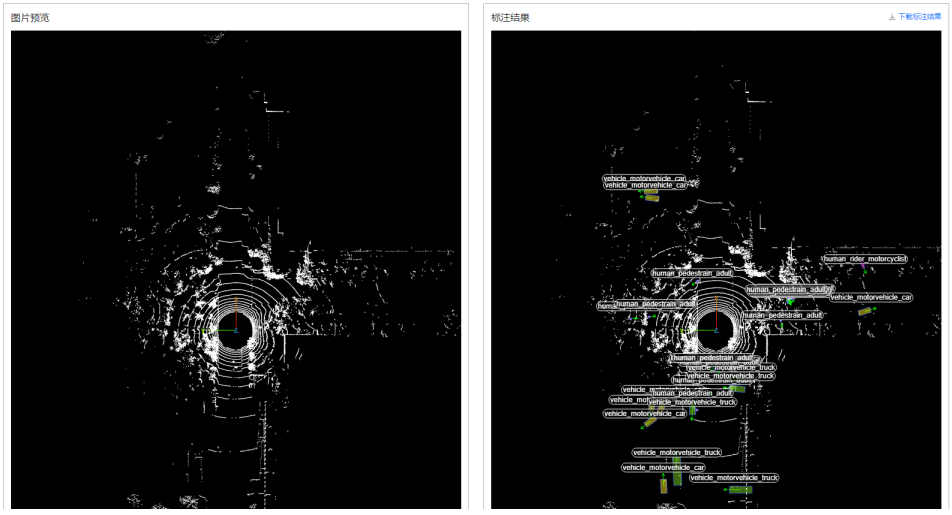
目标分割

- 预标注功能：此处选择“目标分割”。
- 添加文件：上传本地点云文件。只能选择PCD点云文件，文件大小不能超过2MB。

步骤5 单击“确认”，生成相关返回值。3D预标注结果生成后，单击标注结果右上角“下载标注结果”，可以将标注后的图片下载到本地保存。

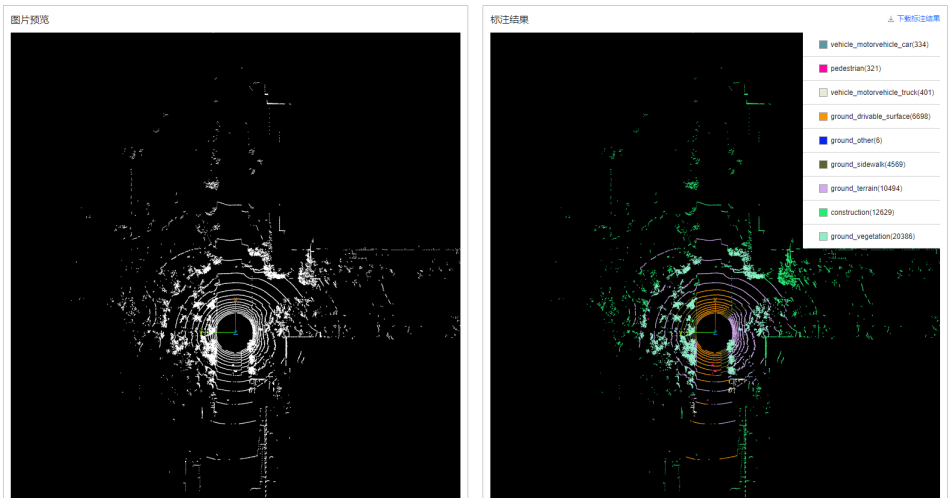
目标检测

图 8-6 标注结果



目标分割

图 8-7 标注结果



----结束

8.6.2 2D3D 融合预标注

自动驾驶传感器中，各个模态有各自的优势和劣势。比如相机模态对visual appearance的感知更为准确，激光雷达模态对距离感知更为有效。然后当LiDAR扫描

线数过低时，经常无法甄别物体的类型，但是此时如果能结合LiDAR扫描和2D图像检测，则可以由3D扫描确定目标大致位置，然后用2D图像检测来识别物体类别。

通过2D3D的融合，可以弥补各自模态的不足，扬长避短，提升目标检测的整体精度。在3D检测的基础上，通过2D cross-check提升3D检测类别的精度提升。

3D标注物支持的类别详情如下：

person（行人）、car（小汽车）、bus（公交车）、traffic_cone（交通锥）、motorcycle（摩托车）、bicycle（自行车）、unknown（未知障碍物）、tricycle（三轮车）、truck（卡车）。

创建 2D3D 融合预标注任务步骤

步骤1 完成OBS授权委托，具体操作步骤请参考**授权操作步骤**。

步骤2 在服务控制台“总览 > 我的模型”区域，开通“场景识别”和“2D图像生成”服务。

步骤3 在左侧菜单栏中选择“智驾模型服务 > 3D预标注”。

步骤4 选择“2D3D融合预标注”页签。

步骤5 单击“新建任务”，参考如下填写基本信息。

- 任务名称：填写任务名称，只能包含数字、英文、中文、下划线、中划线，输入长度不得超过64个字符。
- 任务类型：选择任务类型。分为“目标检测”和“目标追踪”。
  - 目标检测：对采集的点云数据，预测每帧点云数据中物体的框及类别。
  - 目标追踪：对采集的连续帧点云数据，预测每帧点云数据中物体的框、类别、追踪ID， 前后帧中的同一个物体，追踪ID保持一致。
- 输入路径：选择OBS输入路径。输入文件必须满足2D3D融合预标注输入输出文件格式要求。
- 输出路径：选择OBS输出路径。输出文件满足2D3D融合预标注输入输出文件格式要求。
- 描述：简要描述任务，不能包含“@^\#$%&*<>'|"/`”，输入长度不能超过256个字符。

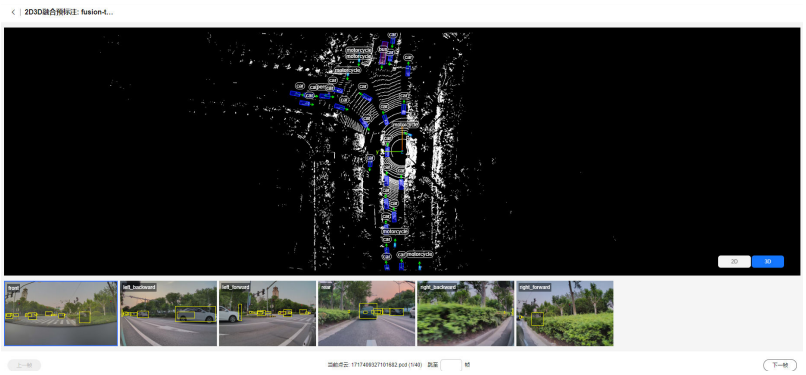
步骤6 单击“确认”，创建一个2D3D融合预标注任务。

步骤7 （可选）管理相关任务。

- 预览任务：

  单击操作栏中的“预览”，预览标注结果。只有执行成功的任务，才支持预览操作。

图 8-8 2D3D 融合预标注结果



- 删除任务：

  单击操作栏中的“删除”，删除任务。任务删除后无法恢复，请谨慎操作。

- 查询任务：

  在搜索输入框中输入搜索条件，按回车键即可查询目标任务。

----结束

2D3D 输入输出文件格式要求

2D3D融合预标注源数据文件要求：

需包含点云文件，六视角图片文件（六视角包含front、front_tele、left_backward、left_forward、right_backward、right_forward、rear），lidar文件和六视角相机的标定文件。追踪任务需要包含位姿文件。

- 文件结构要求

  输入数据的目录结构和文件内容格式需满足octopus数据要求，输入数据在OBS文件组织形式如下所示：

```
|--- Raw
   |--- pointcloud #点云文件夹
      |--- <pcd_timestamp>.pcd
      |--- ...
   |--- image  #图片文件夹
      |--- front
         |--- <img_view_timestamp>.jpg
         |--- ...
      |--- rear
         |--- <img_view_timestamp>.jpg
         |--- ...
      |--- left_backward
         |--- <img_view_timestamp>.jpg
         |--- ...
      |--- left_forward
         |--- <img_view_timestamp>.jpg
         |--- ...
      |--- right_backward
         |--- <img_view_timestamp>.jpg
         |--- ...
      |--- right_forward
         |--- <img_view_timestamp>.jpg
         |--- ...
   |--- param #标定文件夹
      |--- lidar.yaml
      |--- front.yaml
      |--- left_backward.yaml
      |--- left_forward.yaml
      |--- right_backward.yaml
      |--- right_forward.yaml
      |--- rear.yaml
|--- SLAM_Result
   |---pose  #位姿文件夹
      |---lidar_pose.txt
```

- 雷达标定文件要求

```
#lidar.yaml
Transformation:
   data：[0.0, 0.0, 0.0, 0.0,
         0.0, 0.0, 0.0, 0.0,
         0.0, 0.0, 0.0, 0.0,
         0.0, 0.0, 0.0, 1.0]
```

- 相机标定文件要求

```
#<camview>.yaml
cameraType: # 0-pinhole,1-fisheye
```



```
  type: 0
K: #opencv-matrix
  rows: 3
  cols: 3
  dt: d
  data: [fx, 0.0, cx, 0.0, fy, cy, 0.0, 0.0, 1.0]  #fx,fy,cx,cy为相机焦距和主点
D: #opencv-matrix
  rows: 6 #根据实际畸变参数个数填写
  cols: 1
  dt: d
  data: [0.0, 0.0, 0.0, 0.0, 0.0] #填入实际的畸变参数
Imagesize: [width, height] #填入图像宽，高
Transformation: #opencv-matrix
  rows: 4
  cols: 4
  dt: d
  data: [0.0, 0.0, 0.0, 0.0,
      0.0, 0.0, 0.0, 0.0,
      0.0, 0.0, 0.0, 0.0,
      0.0, 0.0, 0.0, 1.0] #请填入转换矩阵的实际值
```

- 位姿文件格式要求

  第一行如下所示，其他行每行代表一个时刻的传感器位姿文件。

  ```
  #timestamp latitude/deg longitude/deg altitude/m utm-x/m utm-y/m utm-z/m x/m y/m z/m r00 r01 r02 r10 r11 r12 r20 r21 r22;
  ```

  说明

  - r00-r22的9个元素代表从局部坐标系到全局坐标系的旋转矩阵，按行排列。
  - 为保持兼容性，数据中存入三种坐标系的位置数据。

    一是经纬高的WGS84坐标系位置数据。

    二是UTM坐标系位置数据，如果无经纬高和UTM坐标信息，该部分字段设置为0。

    三是局部SLAM坐标系位置数据，以米为度量单位。如果提供了经纬高和UTM，则x、y、z字段可置为零，如果无法提供经纬高和UTM，则经纬高和UTM置为零，x、y、z需要为有效数据。（经纬高、UTM）为一组，x、y、z为一组。两组之间二选一。如果两组都有效（均不为0），优先使用第一组全局坐标系的坐标。
  - 时间戳为点云时间戳和点云文件命名中的时间戳保持一致。

- 输出文件格式要求：

  输出数据的文件目录结构满足ocotpus数据目录结构要求，数据目录结构如下所示：

  ```
  |--- BEV_Result
     |--- 3D_object
        |--- 3D_2D_reprojection #3D投影2D框结果
           |--- <pcd_timestamp>
              |---front.json
              |---rear.json
              |---left_backward.json
              |---left_forward.json
              |---right_backward.json
              |---right_forward.json
        |--- detect_tracking_result #3D框结果
           |--- <pcd_timestamp>.json
  ```

8.6.3 3D 多帧预标注

3D多帧预标注当前支持多帧检测和连续帧检测两种标注功能。支持的3D预标注类别为：行人、自行车、摩托车、卡车、公交车、小汽车。

创建 3D 多帧预标注任务步骤

步骤1 在服务控制台“总览 > 我的模型”区域，开通“预标注”服务，具体操作步骤请参考**开通我的模型和购买套餐包**。

步骤2 在左侧菜单栏中选择“智驾模型服务 > 3D预标注”。

步骤3 选择“3D多帧预标注”页签。

步骤4 单击“新建任务”，填写任务名称和描述信息，其他可参考如下填写。

- 任务类型：选择任务类型。
  - 目标检测：对采集的点云数据，预测每帧点云数据中物体的框及类别。
  - 目标追踪：对采集的连续帧点云数据，预测每帧点云数据中物体的框、类别、追踪ID， 前后帧中的同一个物体，追踪ID保持一致。
- 输入路径：选择OBS输入路径。输入文件必须满足3D多帧预标注输入输出文件格式要求
- 输出路径：选择OBS输出路径。输出文件满足3D多帧预标注输出文件格式要求。

步骤5 单击“确认”，创建一个3D多帧预标注任务。

步骤6 （可选）管理3D多帧预标注任务。

- 预览任务

  单击任务操作栏中的“预览”，预览标注结果。仅支持预览执行成功的任务。

  图 8-9 3D 多帧预标注结果

- 删除任务

  单击任务操作栏中的“删除”，根据界面提示删除任务。任务删除后无法恢复，请谨慎操作。

- 查询任务

  在搜索输入框中输入搜索条件，按回车键即可查询任务。

----结束

3D 多帧预标注输入输出文件格式要求

- 输入文件格式

  输入数据的目录结构和文件内容格式需满足octopus数据要求，所选的输入路径文件夹下必须包含Raw子文件夹，Raw文件夹下必须包含pointcloud子文件夹，

pointcloud目录下是待检测的所有pcd文件，文件命名规范为：<时间戳>.pcd，示例如下：

```
|--- Raw
   |--- pointcloud #点云文件夹
      |--- <timestamp>.pcd
      |--- ...
```

- 输出文件格式

  输出数据的文件目录结构满足ocotpus数据目录结构要求，输出文件将在所选输出路径文件夹下自动新建子文件夹，输出文件命名规范为：输出数据文件夹/<任务名称>_<创建时间>/BEV_Result/3D_object/detect_tracking_result/<时间戳>.json，示例如下：

```
|--- BEV_Result
   |--- 3D_object
      |--- detect_tracking_result
         |--- <timestamp>.json
```

8.6.4 4D 分割预标注

4D分割预标注支持对大点云输入文件进行标注。支持的3D预标注类别如下：

Pedestrian（行人）、Bicycle（自行车）、Motorcycle（摩托车）、Truck（卡车）、Bus（公交车）、Car（小汽车）、Trailer（拖车）、Construction vehicle（工程车）、Drivable surface（可行驶路面）、Terrain（地带）、Sidewalk（人行道）、Vegetation（草木）、Other flat（其他）、Barrier（路面栅栏）、Traffic cone（锥桶）、Manmade（建筑）。

创建 4D 分割预标注任务步骤

步骤1 在服务控制台“总览 > 我的模型”区域，开通“预标注”服务，具体操作步骤请参考**开通我的模型和购买套餐包**。

步骤2 在左侧菜单栏中选择“智驾模型服务 > 3D预标注”。

步骤3 选择“4D分割预标注”页签。

步骤4 单击“新建任务”，填写任务名称和描述信息，其他可参考如下填写。

- 输入路径：选择OBS输入路径。输入文件必须满足4D分割预标注输入输出文件格式要求。
- 输出路径：选择OBS输出路径。输出文件满足4D分割预标注输出文件格式要求。

步骤5 单击“确认”，创建一个4D分割预标注任务。

步骤6 （可选）管理4D分割预标注任务。

- 查看标注结果

  单击任务的OBS输出路径，跳转至OBS对应输出目录下，执行成功的任务会在该目录下生成对应的json文件。

图 8-10 OBS 输出文件

- 删除任务
  单击任务操作栏中的“删除”，根据界面提示删除任务。任务删除后无法恢复，请谨慎操作。
- 查询任务
  在搜索输入框中输入搜索条件，按回车键即可查询目标任务。

----结束

4D 分割预标注输入输出文件格式要求

- 输入文件格式
  所选的输入OBS路径目录下须包含所有待检测的pcd文件。
- 输出文件格式
  输出文件将在所选输出路径下新建子文件夹，数据目录结构如下所示：

```
|--- <任务名称>_<时间戳>
    |--- <pcd文件名>.json
```

8.7 3D 预标注车道线检测

8.7.1 3D 车道线检测

上传融合点云、车辆轨迹、坐标偏移等数据，利用AI模型提取车道线和路面标识元素并整理融合输出完整线面矢量元素 。

创建 3D 预标注车道线检测任务

步骤1 在服务控制台“总览 > 我的模型”区域，开通“3D预标注车道线检测”服务。

步骤2 在左侧菜单栏中选择“智驾模型服务 > 3D预标注车道线检测”。

步骤3 在“3D车道线检测”页签单击“新建任务”。

步骤4 填写任务名称，输入路径，输出路径、描述和强度值信息。

说明

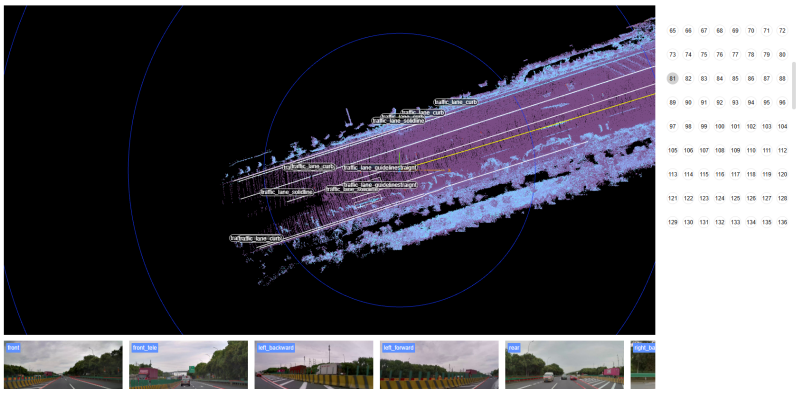
- 输入路径、输出路径：选择OBS输入路径。车道线检测输入文件必须满足车道线检测输入输出文件格式要求，格式可参考**3D预标注车道线检测输入输出文件格式要求**。
- 强度值：点云强度值，取值范围[6-15]的整数，默认值8。

步骤5 单击“确认”，完成3D预标注车道线检测的创建。

步骤6 （可选）创建成功后还可以进行以下操作。

- 停止任务：只有排队中的任务才能执行停止操作。
- 删除任务：只有不是运行中的任务才能执行删除操作。
- 查看任务详情：单击任务名称，在任务界面查看任务基本信息及车道线检测预标注效果图。

图 8-11 3D 预标注车道线预标注效果图



- 查询任务：在搜索输入框中输入搜索条件，按回车键即可查询目标任务。

----结束

3D 预标注车道线检测输入输出文件格式要求

- 输入数据在OBS文件组织形式：

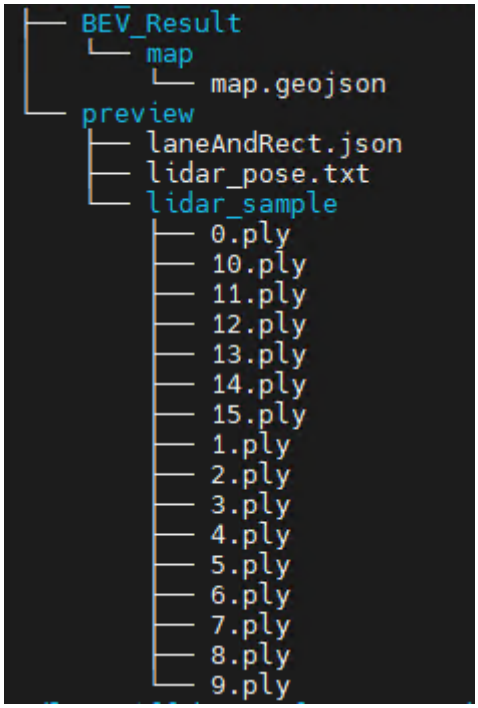
```
|--- Raw
   |--- pointcloud #点云文件夹
      |--- timestamp1.pcd
      |--- ...
   |--- image  #图片文件夹
      |--- front
         |--- <img_view_timestamp>.jpg
         |--- ...
      |--- rear
         |--- <img_view_timestamp>.jpg
         |--- ...
      |--- left_backward
         |--- <img_view_timestamp>.jpg
         |--- ...
      |--- left_forward
         |--- <img_view_timestamp>.jpg
         |--- ...
      |--- right_backward
         |--- <img_view_timestamp>.jpg
         |--- ...
      |--- right_forward
```



```
        |--- <img_view_timestamp>.jpg
        |--- ...
  |--- param #标定文件夹
      |--- lidar.yaml
      |--- front.yaml
      |--- front_tele.yaml
      |--- left_forward.yaml
      |--- right_backward.yaml
      |--- right_forward.yaml
      |--- gnss.yaml
|--- SLAM_Result
   |---pose  #位姿文件夹
      |---lidar_pose.txt
   |---pointcloud_fusion  #融合点云
      |---starttime-endtime.pcd
      |---Metadata.xml
```

- 输出结果在OBS文件组织形式：

图 8-12 OBS 文件输出数据组织形式



8.7.2 Tiff 强度拉伸

点云强度值会影响车道线和路面标识检测结果，对于不同的点云数据，需要进行不同强度值的拉伸，以达到更好的检测效果。

本章节介绍如何创建Tiff强度拉伸任务，并调整任务的强度值。仅任务状态为“执行成功”的任务支持调整强度值。

创建 Tiff 强度拉伸任务

步骤1 在服务控制台“总览 > 我的模型”区域，开通“3D预标注车道线检测”服务。

步骤2 在左侧菜单栏中单击“智驾模型服务 > 3D预标注车道线检测”。

步骤3 在“Tiff强度拉伸”页签单击“新建任务”。

步骤4 填写任务名称，输入路径，输出路径、描述和强度值信息。

说明

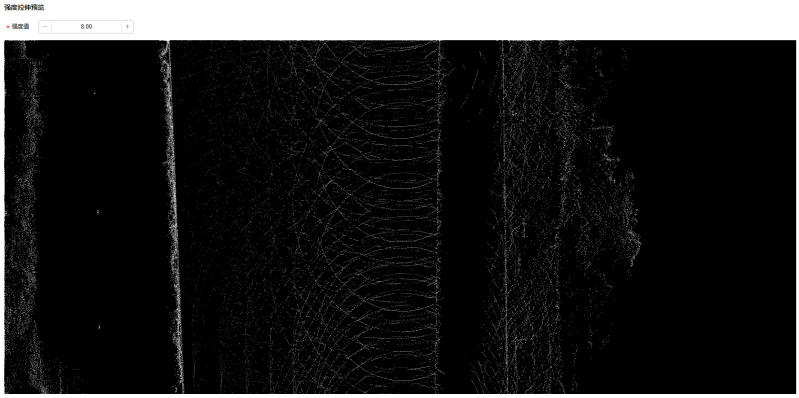
- 输入路径、输出路径：选择OBS输入路径。车道线检测输入文件必须满足车道线检测输入输出文件格式要求，格式可参考**Tiff强度拉伸输入输出文件格式要求**。
- 强度值：点云强度值，取值范围[6-15]，默认值8.00。

步骤5 单击“确认”，完成Tiff强度拉伸的创建。

步骤6 （可选）创建成功后还可以进行以下操作。

- 调整Tiff强度值：单击任务状态为“执行成功”的任务名称，在强度拉伸预览页面可调整任务的强度值，输入[6-15]之间的数值，然后单击空白处等待任务调整成功。
- 停止任务：只有排队中的任务才能执行停止操作。
- 删除任务：不支持删除运行中的任务。
- 查看任务详情：单击任务名称，在任务界面查看任务基本信息及车道线检测预标注效果图。

图 8-13 车道线强度拉伸效果图



- 查询任务：在搜索输入框中输入搜索条件，按回车键即可查询目标任务。

----结束

Tiff 强度拉伸输入输出文件格式要求

- 输入数据在OBS文件组织形式

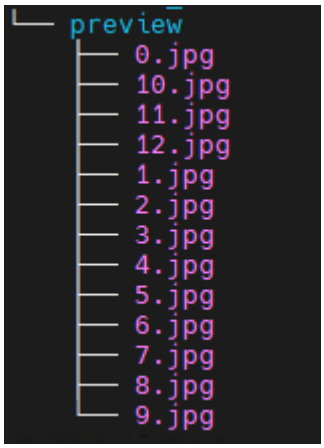
```
|--- Raw
   |--- pointcloud #点云文件夹
      |--- timestamp1.pcd
      |--- ...
   |--- image  #图片文件夹
      |--- front
         |--- <img_view_timestamp>.jpg
         |--- ...
      |--- rear
         |--- <img_view_timestamp>.jpg
         |--- ...
      |--- left_backward
         |--- <img_view_timestamp>.jpg
         |--- ...
      |--- left_forward
         |--- <img_view_timestamp>.jpg
         |--- ...
      |--- right_backward
```



```
        |--- <img_view_timestamp>.jpg
        |--- ...
    |--- right_forward
        |--- <img_view_timestamp>.jpg
        |--- ...
  |--- param #标定文件夹
    |--- lidar.yaml
    |--- front.yaml
    |--- front_tele.yaml
    |--- left_forward.yaml
    |--- right_backward.yaml
    |--- right_forward.yaml
    |--- gnss.yaml
|--- SLAM_Result
  |---pose  #位姿文件夹
    |---lidar_pose.txt
  |---pointcloud_fusion  #融合点云
    |---starttime-endtime.pcd
    |---Metadata.xml
```

- 输出数据在OBS文件组织形式：

  图 8-14 OBS 文件输出数据组织形式

8.8 服务监控

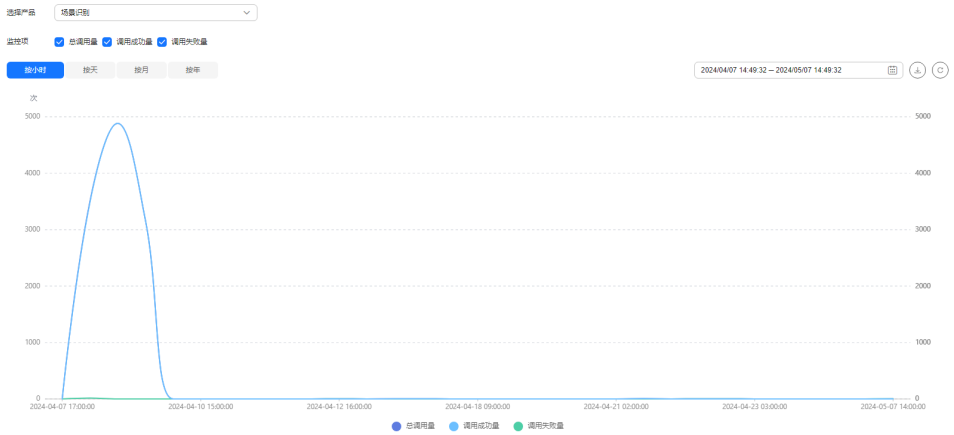
此模块支持选择不同产品、监控项、时间段，查看API的总调用量、调用成功量、调用失败量。

步骤1 在左侧菜单栏中单击“智驾模型服务”。

步骤2 选择“服务监控”页签，按需求选择产品、监控项以及时间段。

步骤3 界面以曲线图的形式显示API的总调用量、调用成功量、调用失败量。

图 8-15 服务监控



----结束

8.9 SLAM 构图

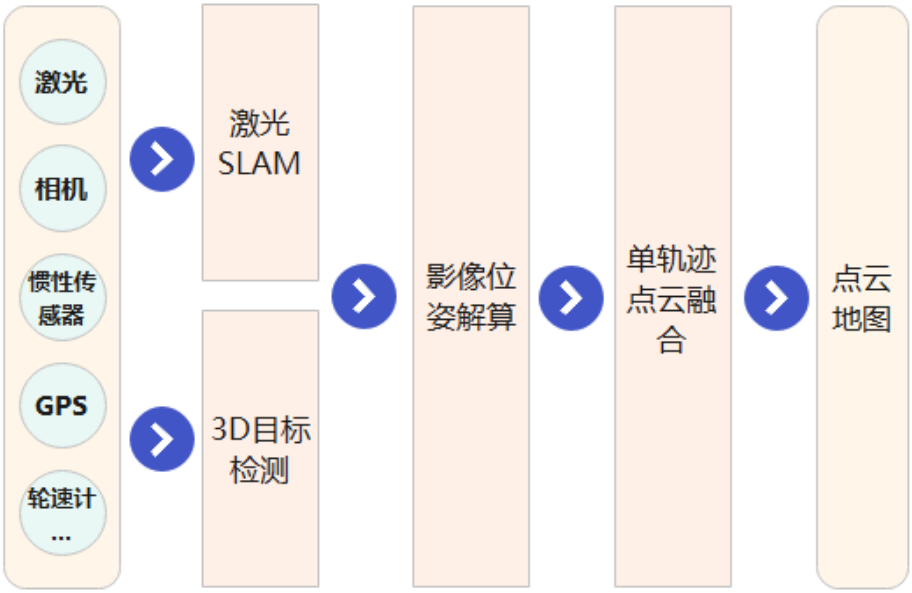
8.9.1 SLAM 构图简介

本任务旨在为4D-BEV数据自动化预标注提供点云地图，进而辅助车企构建自动驾驶车端BEV算法训练提供数据真值生成能力。

本任务将依赖融合定位、运动畸变校正、闭环检测和点云融合等能力构建，对多源传感器数据进行SLAM位姿解算，并在八爪鱼平台上展示激光点云融合结果。

SLAM构图简介如下所示：

图 8-16 SLAM 构图简介



8.9.2 创建 SLAM 构图任务

创建 SLAM 构图任务

步骤1 完成OBS授权委托，具体操作步骤请参考**授权操作步骤**。

步骤2 在服务控制台“总览 > 我的模型”区域，开通“SLAM构图”服务。

步骤3 在左侧菜单栏中单击“智驾模型服务 > SLAM构图”。

步骤4 单击“创建”，填写名称，输入路径，输出路径和描述信息。

输入路径、输出路径：选择OBS输入路径和输出路径，必须满足SLAM构图输入输出文件格式要求，格式可参考**SLAM构图输入输出文件格式要求**。

说明

- 上传的数据为经过时间同步后的数据，时间同步精度要求为0.1毫秒。
- 上传的数据需具备绝对位置信息（通常需要来自于GNSS等传感器）和IMU信息。
- 单轨迹数据长度建议不超过2km，且点云不超过2000帧。
- 点云采集频率10Hz或以上，GNSS和IMU等传感器采集频率建议100Hz或以上。

步骤5 单击“确认”创建任务。

步骤6 （可选）创建成功后在SLAM构图任务列表页可以进行以下操作。

- 查看任务详情：单击任务名称，可查看SLAM构图任务基本信息及点云地图构建结果。
- 停止任务：当任务状态为“排队中”时，可单击操作栏内的“停止”。
- 查询任务：在搜索输入框中输入搜索条件，按回车键即可查询目标任务。

----结束

SLAM 构图输入输出文件格式要求

- 全流程包含原始数据SLAM构图，OBS单轨迹数据结构如下：

```
|--- /track-XXX
   |--- Raw
      |--- param
         |--- fisheye_front.yaml
         |--- fisheye_left.yaml
         |--- fisheye_rear.yaml
         |--- fisheye_right.yaml
         |--- front.yaml
         |--- front_tele.yaml
         |--- left_backward.yaml
         |--- left_forward.yaml
         |--- rear.yaml
         |--- right_backward.yaml
         |--- right_forward.yaml
         |--- lidar.yaml
         |--- gnss.yaml
      |--- image
         |--- fisheye_front
                |--- timestamps0.jpg
                |--- timestamps1.jpg
                |--- timestamps2.jpg
                |--- ...
         |--- fisheye_left
                |--- timestamps0.jpg
                |--- timestamps1.jpg
```

```
                |--- timestamps2.jpg
                |--- ...
        |--- fisheye_rear
                |--- timestamps0.jpg
                |--- timestamps1.jpg
                |--- timestamps2.jpg
                |--- ...
        |--- fisheye_right
                |--- timestamps0.jpg
                |--- timestamps1.jpg
                |--- timestamps2.jpg
                |--- ...
        |--- front
                |--- timestamps0.jpg
                |--- timestamps1.jpg
                |--- timestamps2.jpg
                |--- ...
        |--- front_tele
                |--- timestamps0.jpg
                |--- timestamps1.jpg
                |--- timestamps2.jpg
                |--- ...
        |--- left_backward
                |--- timestamps0.jpg
                |--- timestamps1.jpg
                |--- timestamps2.jpg
                |--- ...
        |--- left_forward
                |--- timestamps0.jpg
                |--- timestamps1.jpg
                |--- timestamps2.jpg
                |--- ...
        |--- rear
                |--- timestamps0.jpg
                |--- timestamps1.jpg
                |--- timestamps2.jpg
                |--- ...
        |--- right_backward
                |--- timestamps0.jpg
                |--- timestamps1.jpg
                |--- timestamps2.jpg
                |--- ...
        |--- right_forward
                |--- timestamps0.jpg
                |--- timestamps1.jpg
                |--- timestamps2.jpg
                |--- ...
    |--- gnss
        |--- gnss.txt
    |--- imu
        |--- imu.txt
    |--- pointcloud
        |--- timestamps0.pcd
        |--- timestamps1.pcd
        |--- timestamps2.pcd
        |--- ...
    |--- pose(可选，如有则提供)
        |--- init_lidar_pose.txt
```

- 输出数据在OBS的数据结构：

```
|--- /track-XXX
    |--- SLAM_Result
            |--- pointcloud_fusion
                |--- timestamps0-timestampsn.pcd
                |--- Metadata.xml
            |--- pose
                |--- lidar_pose.txt
                |--- fisheye_front.txt
                |--- fisheye_left.txt
                |--- fisheye_rear.txt
```



```
                |--- fisheye_right.txt
                |--- front.txt
                |--- front_tele.txt
                |--- left_backward.txt
                |--- left_forward.txt
                |--- rear.txt
                |--- right_backward.txt
                |--- right_forward.txt
```

8.10 智驾模型管理

8.10.1 智驾模型

智驾模型微调后，需要对模型进行评测，并把模型部署到环境上进行推理，方便用户使用自己的数据训练得到专属模型进行推理，及比对训练效果。

智驾模型列表中展示服务内置的初始模型和用户进行模型微调后的迭代模型。

图 8-17 智驾模型列表

智驾模型列表相关操作

智驾模型列表可以进行以下操作。

表 8-4 智驾模型列表相关操作

任务	操作步骤
编辑模型	单击操作栏中的“编辑”，编辑智驾模型。仅支持修改描述信息。初始模型不支持编辑。
删除模型	单击操作栏中的“删除”，删除智驾模型。初始模型不支持删除。

任务	操作步骤
查看模型详情	1. 单击模型名称，查看模型详情。 2. 在模型详情页，查看模型基本信息和模型版本列表。可以对模型版本进行“模型微调”、“创建在线服务”和“删除”操作。 **说明** • 初始模型不支持“创建在线服务”和“删除”。 • 只有“创建成功”的模型版本支持“模型微调”和“创建在线服务”。
查询模型	在搜索输入框中输入搜索条件，按回车键即可查询。

8.10.2 在线服务

微调后的迭代模型创建在线服务，进行推理，查看预测效果。每类迭代模型，只能同时运行一个在线服务，如果已经存在“运行中”的在线服务，将无法继续创建该类在线服务。

创建在线服务

步骤1 在左侧菜单栏中选择“智驾模型服务 > 智驾模型管理”。

步骤2 选择“在线服务”页签，单击“创建在线服务”。

步骤3 完成在线服务基本信息和模型仓库及配置。

- 名称：输入在线服务名称，只能包含数字、英文、中文、下划线、中划线，输入长度不能超过64个字符。
- 运行时间：设置在线服务运行时间，运行时间到期后，在线服务将自动停止。在线服务的最晚停止时间为模型套餐包的最晚失效时间。模型套餐包的最晚失效时间，可以在“总览”界面，“我的模型”区域查看。
- 描述：输入在线服务的简要描述，不包含“@^\#$%&*<>'|"/`”，不得超过500个字符。
- 模型版本：选择模型微调后的迭代模型及版本。支持同时选择车道线预标注模型、路面预标识模型和2D3D融合检测模型。

步骤4 单击“创建”，确认配置信息后，单击“确认”下发创建在线服务任务。

----结束

在线服务列表相关操作

在线服务列表可以进行以下操作。

表 8-5 在线服务列表相关操作

任务	操作步骤
修改在线服务	单击操作栏中的“修改”，修改在线服务。“启动中”的在线服务不支持修改。

任务	操作步骤
预测在线服务	单击操作栏中的“预测”，跳转至相关在线服务界面，进行预测，测试模型的推理能力。只有“运行中”的在线服务支持预测。
启动在线服务	单击操作栏中的“更多 > 启动”，启动在线服务。
停止在线服务	单击操作栏中的“更多 > 停止”，停止在线服务。
删除在线服务	支持单个、批量删除在线服务。删除操作无法恢复，请谨慎操作。 ● 单击操作栏中的“更多 > 删除”，单个删除在线服务。 ● 选择需要删除的在线服务，单击列表上方的“删除”，批量删除在线服务。 内置在线服务不支持删除。
查看在线服务详情	1. 单击在线服务名称，查看在线服务详情。 2. 在线服务详情页，查看在线服务基本信息、调用接口和配置信息。
查询在线服务	在搜索输入框中输入搜索条件，按回车键即可查询。

在线服务相关操作与任务所处状态约束关系请见下表：

说明

内置在线服务不支持“修改”、“启动”、“停止”、“重启”和“删除”操作。
以下列表只展示迭代模型创建的在线服务的操作状态与约束关系。

表 8-6 操作与状态约束关系

状态	修改	预测	启动	停止	重启	删除
运行中	√	√	x	√	√	√
停止	√	x	√	x	x	√
告警	√	x	√	√	x	√
部署中	x	x	x	√	x	√
失败	√	x	√	x	x	√
资源排队中	x	x	x	x	x	√

9 镜像仓库

9.1 镜像仓库

Octopus平台各服务均提供用户自定义镜像功能，镜像仓库管理模块对镜像提供了统一管理。

新建镜像仓库

步骤1 用平台管理员账号登录Octopus平台。

步骤2 在左侧菜单栏中，单击“镜像仓库 ”。

步骤3 单击“新建”，填写基本信息。

- 名称：输入镜像仓库的名称，只能包含数字、英文、中文、下划线、中划线。
- 描述：简单描述镜像仓库，最大长度为255。
- 用途：根据需求在下拉框选择用途。
- 使用范围：仅支持团队，即租户内所有配置了该镜像相关权限的用户都可见可编辑。

步骤4 单击“确认”，在镜像仓库列表即可查看新建的镜像仓库。

步骤5 （可选）管理已创建的镜像仓库。

在“镜像仓库”列表，还可以完成以下操作。

- 查看镜像仓库详情：单击操作栏内的“详情”，可查看镜像仓库详情。
- 查询镜像仓库：在搜索框中输入搜索条件，按回车键即可查询。
- 删除镜像仓库：单击操作栏内的“删除”，可删除镜像仓库。
- 编辑镜像仓库：单击操作栏内的“编辑”，可编辑镜像仓库名称和描述。

----结束

新建镜像版本

步骤1 在左侧菜单栏中，单击“镜像仓库 ”。

步骤2 在镜像仓库列表，单击操作栏中的“详情”，进入到镜像仓库的详情页。

步骤3 单击左上角“新建”，填写镜像版本描述信息。

步骤4 单击“确认”，在镜像版本列表，可查看镜像版本信息。

----结束

镜像版本相关操作

在“镜像版本”列表，还可以完成以下操作。

表 9-1 镜像版本相关操作

任务	操作步骤
推送镜像	1. 单击指定镜像版本“操作”栏内的“推送”。 2. 复制登录指令，登录镜像仓库。 3. 在docker客户端，用“docker tag”命令将要推送上库的本地镜像打标签为推送指令“docker push”后的镜像名称。例如： `docker tag 本地镜像 odrp-beta.Octopus.ias.huawei.com/Octopus/11ffec1e` 4. 在docker客户端，复制推送指令，将镜像推送至Octopus平台。
拉取镜像	支持将平台的镜像拉取至本地进行调试开发。仅支持创建成功的镜像版本进行拉取镜像操作。 1. 单击指定镜像版本“操作”栏内的“拉取”。 2. 复制拉取指令，将镜像拉取至本地。
删除镜像版本	单击镜像版本操作栏内的“更多>删除”，可删除镜像版本。
取消镜像推送	单击镜像版本操作栏内的“更多>取消推送”，可取消推送镜像版本。
二次推送镜像	单击操作栏内的“推送”，可对状态为“创建完成”的镜像版本进行二次推送。（推理服务镜像不建议使用二次推送镜像功能。）

9.2 制作镜像（数据集）

9.2.1 Dockerfile 示例

用户可使用命令行模式或Dockerfile模式进行构建。

数据集镜像不支持调用GPU资源。

以数据集自定义镜像为例，一般的镜像制作Dockerfile示例如下：

```
# 载入基础镜像，用户可根据数据集处理任务的实际需要选择合适的镜像环境，例如ubuntu:latest
FROM {基础镜像}

# 设置工作目录【可选】默认为ROOT，用户可修改USER及PATH
WORKDIR /root/workspace

# 按需安装用户APT环境；如果需要修改/etc/apt/sources.list可替换
COPY /path/to/sources.list /etc/apt/sources.list
RUN apt-get install vim
```

```
# 按需安装用户算法环境；如果需要修改~/.pip/pip.conf可替换；用户也可安装miniconda进行包管理
COPY /path/to/pip.conf /root/.pip/pip.conf
COPY /path/to/requirements.txt /root
RUN pip install –r /root/requirements.txt

# 设置环境变量【可选】
ENV PYTHONUNBUFFERED 1
```

对于Dockerfile的统一构建方式如下：

docker build –f [DockerfileName] –t [ImageName:ImageVersion] .

9.2.2 环境变量使用说明

镜像运行时，会向运行环境注入部分默认文件配置:

表 9-2 环境变量说明

任务名称	文件名	环境变量
数据集任务	数据集待筛选数据目录	OCTPS_DATASET_DIR
	数据集筛选结果数据目录	TARGET_RESULT_DIR="/tmp/…/result/data"
	数据集日志文件目录	TARGET_LOG_DIR="/tmp/…/log"
	数据集待筛选数据文件目录（注：标注任务与通用存储生成数据集需通过所在目录json文件获取所需筛选数据）	SOURCE_DATASET_FILE_DIR
	数据集筛选路径	OCTOPUS_INPUT_FILTER_PATH_LIST

OCTPS_DATASET_DIR

OCTPS_DATASET_DIR为数据集源数据的数据路径，根据不同的数据来源，所挂路径不同， 示例：

```
本地：/tmp/dataset-temp/local_import/6f91947c-cd47-434b-b654-8332da961d7a/
f7c9a054-3c9e-49c7-8934-a1e1d668eb12/
标注：/tmp/label-data/
数据仓库：/tmp/warehouse/
生成子集，视图：/tmp/dataset-new/6f91947c-cd47-434b-b654-8332da961d7a/dataset/
```

OBS需通过用户桶的ak，sk依据OBS相关的sdk获取到用户所需筛选的源数据，示例：

图 9-1 示例图 1

图 9-2 示例图 2

TARGET_RESULT_DIR

TARGET_RESULT_DIR为存放筛选或者格式转换后数据的路径，本地路径示例：

```
/tmp/temp-data/dataset/c8a73760-b5df-4f61-81d7-17e144fa6d69/result/data/
```

对应OBS中raw桶路径为：

```
temp-data/dataset/c8a73760-b5df-4f61-81d7-17e144fa6d69/result/data/
```

TARGET_LOG_DIR

TARGET_LOG_DIR为数据集镜像筛选日志文件的存储路径，示例：

```
/tmp/dataset/6f91947c-cd47-434b-b654-8332da961d7a/log
```

如果将日志文件存放在此路径，创建任务成功后可以在平台看到此日志文件。

SOURCE_DATASET_FILE_DIR

SOURCE_DATASET_FILE_DIR为标注或通用存储生成数据集时的源数据索引json文件，示例：

标注：

```
/tmp/dataset-temp/{versionId}/f7c9a054-3c9e-49c7-8934-a1e1d668eb12/result_frame.json
```

Json文件内容示例：

通用存储：

```
/tmp/data-warehouse/warehouse-dataset/
```

注：通用存储可能存在多个索引json文件，需遍历。（file_attributes_1.json, file_attributes_2.json…… ）

Json文件内容示例：

```
{
    "/tmp/warehouse/1342/segments.csv": {
        "tags": [
            "sjf",
            "lxc"
        ],
        "custom_attributes": [
            {
                "lxc": "sjf"
            },
            {
                "sjf": "lxc"
            }
        ],
        "id": "RD1z5IcBnzBAZ6qGoGy6",
        "sort": [
            "1342/",
            "1683164995766",
            "segments.csv"
        ],
        "file_name": "segments.csv",
        "file_type": "text",
        "data_warehouse_id": "f4234b59-6e62-4b6f-91b3-e9b99e7aef55",
        "raw_data_id": "",
        "data_process_task_id": "1342",
        "create_time": 1683164995766,
        "file_folder_path": "1342/"
    }
}
```


OCTOPUS_INPUT_FILTER_PATH_LIST

创建通用存储类型的数据集，在新建版本时，选择自定义镜像后筛选的路径，示例：

```
/tmp/warehouse/process-job-1,/tmp/warehouse/process-job-2/folder1/,/tmp/warehouse/process-job-2/folder1/file.png
```

注意事项

创建Octopus格式数据集需在自定义镜像内将数据处理为Octopus格式存入TARGET_LOG_DIR内，标注来源的数据集创建只支持Octopus格式。

9.3 制作镜像（标注）

Octopus平台依赖算子镜像内的/bin/bash、stdbuf、tee软件，请确保基础镜像内包含上述软件且能通过PATH找到。

9.3.1 Dockerfile 示例

一般情况下，引擎主要包含预标注算法或预审核算法运行所需要的基本依赖环境，用户也可将预标注算法或预审核算法包内置在镜像仓库的镜像中。

用户可使用命令行模式或Dockerfile模式进行构建。

以预标注自定义镜像为例，一般的镜像制作Dockerfile示例如下：

```
# 载入基础镜像，训练或评测引擎一般需包含cuda/cudnn等算法基础环境。用户可手动制作或拉取官方镜像（如xxx/cuda:11.0.3-devel-ubuntu18.04）
FROM {基础镜像}
```

```
# 设置工作目录【可选】默认为ROOT，用户可修改USER及PATH
WORKDIR /root/workspace

# 如果是内置预标注算法的自定义镜像，需要把预标注算法复制到工作目录下，
COPY /path/to/algorithom /path/to/algorithom

# 按需安装用户APT环境。如果需要修改/etc/apt/sources.list可替换
COPY /path/to/sources.list /etc/apt/sources.list
RUN apt-get install vim

# 按需安装用户算法环境。如果需要修改~/.pip/pip.conf可替换。用户也可安装miniconda进行包管理
COPY /path/to/pip.conf /root/.pip/pip.conf
COPY /path/to/requirements.txt /root
RUN pip install –r /root/requirements.txt

# 设置环境变量【可选】
ENV PYTHONUNBUFFERED 1
```

对于Dockerfile的统一构建方式如下：

```
docker build –f [DockerfileName] –t [ImageName:ImageVersion] .
```

9.3.2 环境变量使用说明

镜像运行时，会向运行环境注入部分默认文件配置:

表 9-3 环境变量说明

任务名称	文件名	环境变量
预标注任务	待标注数据集目录	OCTPS_DATASET_DIR="/tmp/…/data"
	模型仓库中的模型算法目录（非内置）	OCTPS_MODEL_DIR="/tmp/label/source/model"
	模型版本关联标注物文件目录（填写预标注结果时，需要根据类别填写对应的标签id）	OCTPS_META_PATH="/tmp/…/meta/label_meta_infos.json"
	预标注结果数据目录	TARGET_RESULT_DIR="/tmp/…/result/data"
	预标注日志文件目录	TARGET_LOG_DIR="/tmp/…/result/log"
预审核任务	全量数据集目录	OCTPS_DATASET_DIR="/tmp/…/data"
	模型仓库中的模型算法目录（非内置）	OCTPS_MODEL_DIR="/tmp/label/source/model"
	模型版本关联标注物文件目录	OCTPS_META_PATH="/tmp/…/meta/label_meta_infos.json"
	审核属性字段目录	OCTPS_INSPECTION_ATTRI_DIR = /tmp/…/attribute/inspection_attribute.json

任务名称	文件名	环境变量
	待审核的数据帧索引文件目录（用于从全量数据集中筛选出需要审核的数据）	OCTPS_DATASET_INDEX_PATH= /tmp/…/index/index.json
	预审核结果数据目录	TARGET_RESULT_DIR="/tmp/…/result/data"
	预审核规则数据目录	TARGET_RULES_DIR=/tmp/…/rule/rules.json
	预审核日志文件目录	TARGET_LOG_DIR="/tmp/…/result/log"

OCTPS_DATASET_DIR

- **预标注任务**

  OCTPS_DATASET_DIR为待标注的数据路径，示例：/tmp/label/task-2022/source/data。每一帧数据存放在单独的子文件夹，文件组织结构如下所示。

  2D数据帧文件组织结构：

```
OCTPS_DATASET_DIR/XX/…/XX/├─子文件夹1
                          ├─图片1.jpg  #标注图片
                          ├─标注1.json #图片1的标注信息，非必有
                        ├─子文件夹2
                          ├─图片2.jpg  #标注图片
                          ├─标注2.json  #图片2的标注信息，非必有
```

  3D数据帧文件组织结构：

```
OCTPS_DATASET_DIR/XX/…/XX/├─子文件夹1
                          ├─点云1.pcd #已标注点云
                          ├─图片1.jpg #点云1对应的同时刻图片，非必
有
                          ├─标注1.json #点云1的标注信息，非必有
                        ├─子文件夹2
                          ├─点云2.pcd #已标注点云
                          ├─图片2.jpg #点云2对应的同时刻图片，非必有
                          ├─标注2.json #点云2的所有标注信息，非必有
```

  注：“XX/…/XX/”部分存在嵌套多个文件夹的情况，具体嵌套层数和上传的数据集目录结构有关系。

- **预审核任务**

  OCTPS_DATASET_DIR为待审核的全量数据路径，示例：/tmp/label/task-2022/source/data。每一帧数据存放在单独的子文件夹，文件组织结构如下所示。

  2D数据帧文件组织结构：

```
OCTPS_DATASET_DIR/├─子文件夹1
                  ├─图片1.jpg  #已标注图片
                  ├─labels.json  #图片1的所有标注信息
                ├─子文件夹2
                  ├─图片2.jpg  #已标注图片
                  ├─labels.json  #图片2的所有标注信息
```

  3D数据帧文件组织结构：

```
OCTPS_DATASET_DIR/├─子文件夹1
                  ├─点云1.pcd  #已标注点云
                  ├─图片1.jpg  #点云1对应的同时刻图片
                  ├─labels.json  #点云1的所有标注信息
```

```
        ├─子文件夹2
        │  ├─点云2.pcd  #已标注点云
        │  ├─图片2.jpg  #点云2对应的同时刻图片
        │  ├─labels.json  #点云2的所有标注信息
```

每帧数据的标注结果存放在labels.json中，具体格式及内容说明参考**2.7.4 OCTOPUS数据集格式说明**中的labels字段。

OCTPS_META_PATH

OCTPS_META_PATH为标注物文件路径，示例：/tmp/label/task-2022/source/meta/label_meta_infos.json。标签文件中包含了当前任务所选择的所有标注物的基本信息。预标注结果中每个标注对象所需的标注物id，可通过此文件中的id字段获取。标注物文件内容示例如下。

```
[
 {
   "id": 2085, # 平台上所建标注物的ID
   "name": "Car", # 平台上所建标注物的名称
   "color": "#d0021b",
   "label_shape_type": "bndbox",
   "attribute": "{}",
   "description": "car",
     ... ...
 }
... ...
]
```

OCTPS_MODEL_DIR

OCTPS_MODEL_DIR为模型仓库中的模型文件在镜像中的存放路径，示例：/tmp/label/source/model。

OCTPS_INSPECTION_ATTRI_DIR

OCTPS_INSPECTION_ATTRI_DIR为审核属性字段文件路径，示例：/tmp/…/attribute/inspection_attribute.json。文件内容示例如下：

```
{
  "inspection": {
    "miss_label_error": false,  #漏标
    "vehicle_direction_error": false, #车头方向错误
    "error_desc": "",
    "attribute_error": false,  #属性错误
    "out_range_label_error": false,  #未贴合
    "anchor_error": false, #锚点错误
    "classification_error": false,  #类别错误
    "extra_label_error": false #多标
  }
}
```

OCTPS_DATASET_INDEX_PATH

OCTPS_DATASET_INDEX_PATH为待审核的数据帧索引文件目录，示例：/tmp/…/index/index.json。文件内包含所需数据所在的子文件夹名称，用于从全量数据集（OCTPS_DATASET_DIR）中筛选出需要审核的数据。文件内容示例如下：

```
["子文件夹1","子文件夹2"]
```

TARGET_RESULT_DIR

TARGET_RESULT_DIR为存放预标注/预审核结果的路径，示例：/tmp/label/task-2022/result/data。预标注/预审核结果按照特定格式保存在json文件中。3D数据

帧中json文件的路径及命名和点云文件保持一致，2D数据帧中json文件的路径及命名和图片文件保持一致。以上述2D数据帧为例，结果文件组织结构如下所示：

```
TARGET_RESULT_DIR /XX/…/XX/ ├─子文件夹1
                            ├─图片1.json  #图片1的结果信息
                          ├─子文件夹2
                            ├─图片2.json  #图片2的结果信息
```

注：“XX/…/XX/子文件夹i/”路径和待处理数据集（OCTPS_DATASET_DIR）中的此部分路径保持一致。

TARGET_LOG_DIR

TARGET_LOG_DIR为模型推理日志文件的存储路径，示例：/tmp/label/task-2022/result/log。如果将日志文件存放在此路径，创建任务成功后可以在平台看到此日志文件。

TARGET_RULES_DIR

TARGET_RULES_DIR为模型使用的审核规则的存储路径，示例：/tmp/…/rule/rules.json。审核规则信息按照特定格式（详见“预审核规则格式说明”）存在json文件中，创建任务成功后可以在平台报告中看到此规则信息。

9.4 制作镜像（训练）

9.4.1 制作 CCE 集群训练镜像

Octopus平台依赖算子镜像内的/bin/bash、stdbuf、tee软件，请确保基础镜像内包含上述软件且能通过PATH找到。

一般情况下，训练与评测定义为同一个引擎，主要包括算法或评测脚本运行所需要的基本依赖环境。用户可使用命令行模式或Dockerfile模式进行构建。

以训练、评测镜像为例，一般的镜像制作Dockerfile示例如下：

```
# 载入基础镜像，训练或评测引擎一般需包含cuda/cudnn等算法基础环境。用户可手动制作或拉取官方镜像（如 xxx/cuda:11.0.3-devel-ubuntu18.04 ）
FROM {基础镜像}

# 设置工作目录【可选】默认为ROOT，用户可修改USER及PATH
WORKDIR /root/workspace

# 安装用户APT环境。如果需要修改/etc/apt/sources.list可替换
COPY /path/to/sources.list /etc/apt/sources.list
RUN apt-get install vim

# 安装用户算法环境。如果需要修改~/.pip/pip.conf可替换。用户也可安装miniconda进行包管理
COPY /path/to/pip.conf /root/.pip/pip.conf
COPY /path/to/requirements.txt /root
RUN pip install –r /root/requirements.txt

# 设置环境变量【可选】
ENV PYTHONUNBUFFERED 1
```

编译镜像类似上述训练、评测镜像制作方式，但一般不包含cuda/cudnn库，需替换为用户的编译环境。

对于Dockerfile的统一构建方式如下：

docker build –f [DockerfileName] –t [ImageName:ImageVersion] .

镜像运行时，会向运行环境注入部分默认文件配置:

表 9-4 环境变量说明

任务名称	文件名	环境变量
训练任务	增量模型目录	MODEL="/tmp/data/model"
	训练产物目录	RESULT= "/tmp/result"
	数据集目录	DATASET="/tmp/data/dataset/"
评测任务	评测结果目录	EVAL_RESULT= "/tmp/result"
	模型版本文件目录	MODEL="/tmp/data/model"
	数据集目录	DATASET= "/tmp/data/dataset/dataset-0"
编译任务	模型版本文件目录	MODEL_PATH="/tmp/data/model"
	编译产物目录	TMP_RESULT_PATH= "/tmp/result"
预标注任务	数据集目录	OCTPS_DATASET_DIR="/tmp/…/data"
	模型版本文件目录	OCTPS_MODEL_DIR="/tmp/…/model"
	模型版本关联标注物文件路径	OCTPS_META_PATH="/tmp/…/meta/label_meta_infos.json"
	预标注结果数据目录	TARGET_RESULT_DIR="/tmp/…/result/data"
	预标注日志文件目录	TARGET_LOG_DIR="/tmp/…/result/log"
推理服务	模型版本文件目录	OCTOPUS_MODEL="/home/mind/model"

9.4.2 制作 ModelArts 集群训练镜像

制作训练镜像

八爪鱼训练镜像使用**ma-user**用户运行，用户需保证镜像内已创建**ma-user**用户，且训练过程中使用到的python环境或其他依赖对**ma-user**具有权限。本地镜像构建完成后，需要将镜像上传到八爪鱼平台镜像仓库。

- 如果使用dockerfile构建，参考命令为**docker build -f Dockerfile -t new_image:1.0**
- 如果使用容器命令行方式构建，参考命令为**docker commit** *{container-id}* **new_image:1.0**

以下分别介绍“从0到1构建”和“从已有镜像迁移”两种制作方式。

- 从0到1构建训练镜像。

  用户可以docker run -it {image-id-or-name} bash在容器内依次执行命令后commit为新的镜像。

  或者采用dockerfile构建，参考如下：

```
# 1、查看算法依赖，一般需要考虑使用的ubuntu版本、cuda版本、cudnn版本
FROM xxx/cuda:11.3.1-cudnn8-devel-ubuntu20.04

# 2、安装python环境 如果使用其他方式安装，可替换
USER root
RUN apt update && \
    apt install python3 python3-pip -y && \
    # 可选，安装opencv库时需要安装以下依赖，缺少其他库请按实际需求添加依赖
    # apt install libgl1-mesa-glx -y && \
    ln -sf /usr/bin/python3 /usr/bin/python && \
    ln -sf /usr/bin/pip3 /usr/bin/pip && \
    # 如果在root下安装pip依赖，在步骤3后需要对python库添加权限，如
    # chmod -R 777 ${PYTHONPATH}/lib/python3.8 或者
    # chown -R ma-user:100 ${PYTHONPATH}/lib/python3.8
    # pip install xxx

# 3、增加ma-user(必选)
RUN useradd -m -d /home/ma-user -s /bin/bash -g 100 -u 1000 ma-user

# 4、安装python依赖（或可在步骤2中安装）
USER ma-user
RUN pip install xxx

ENV PYTHONPATH ${PYTHONPATH}:{YOUR NEW PYTHON BIN PATH}
```

- 从已有镜像迁移。

```
# 将已有镜像导入
FROM {YOUR OWN IMAGE}
USER root
RUN useradd -m -d /home/ma-user -s /bin/bash -g 100 -u 1000 ma-user

# 此处添加对使用python库赋权限，参考方式一
# 例如，用户的python依赖均在root属主下，需要对ma-user开放权限

USER ma-user
ENV PYTHONPATH ${PYTHONPATH}:{YOUR PYTHON BIN PATH}
```

制作推理服务镜像

- 推理服务镜像同训练任务镜像一样，必须内置一个用户名为“**ma-user**”，组名为“**ma-group**”的普通用户，且必须确保该用户的uid=1000、gid=100。
- 需要明确设置镜像的启动命令。执行命令为：**CMD sh /home/mind/run.sh**
- 服务端必须使用https协议， 且暴露在所有网络平面（0.0.0.0）的“8080”端口。
- 在“8080”端口，提供URL路径为“/health”的健康检查接口供健康检查使用。
- 接口仅支持POST、GET、PUT、DELETE四种方法。

  Dockerfile示例：

```
FROM python:3.11

USER root
RUN useradd -m -d /home/ma-user -s /bin/bash -g 100 -u 1000 ma-user

USER ma-user
RUN pip install --progress-bar off flask cryptography

WORKDIR /home/ma-user
COPY server.py .
CMD python server.py
```


HTTPS Server示例：

```
from flask import Flask, request
import json

app = Flask(__name__)

@app.route('/greet', methods=['POST'])
def say_hello_func():
    print("------------ in hello func ----------")
    data = json.loads(request.get_data(as_text=True))
    print(data)
    username = data['name']
    rsp_msg = 'Hello, {}!'.format(username)
    return json.dumps({"response":rsp_msg}, indent=4)

@app.route('/goodbye', methods=['GET'])
def say_goodbye_func():
    print("------------ in goodbye func ----------")
    return '\nGoodbye!\n'

@app.route('/', methods=['POST'])
def default_func():
    print("------------ in default func ----------")
    data = json.loads(request.get_data(as_text=True))
    return '\n called default func !\n {} \n'.format(str(data))

@app.route('/health', methods=['GET'])
def healthy():
    return "{\"status\": \"OK\"}"

# host must be "0.0.0.0", port must be 8080
if __name__ == '__main__':
    # 访问创建推理服务时选择的模型版本文件
    torchserve = PTVisionService(model_path=os.getenv('OCTOPUS_MODEL') + '/best.pt')

    # host must be "0.0.0.0", port must be 8080
    app.run(host="0.0.0.0", port=8080, ssl_context='adhoc')
```

注意：在创建推理服务时选择的模型版本文件将在服务启动时下载到OCTOPUS_MODEL对应的路径下：

环境变量名称	含义	默认值（默认值可能会随版本变化，不建议直接使用）
OCTOPUS_MODEL	模型版本文件下载目录	/home/mind/model

在本地机器调试

自定义引擎的规范可以在安装有docker的本地机器上通过以下步骤提前验证：

a. 将自定义引擎镜像下载至本地机器，假设镜像名为“custom_engine:v1”。
b. 将模型版本文件夹复制到本地机器，假设模型包文件夹名字为“model”。
c. 在模型文件夹的同级目录下验证如下命令启动服务：

   docker run -u 1000:100 -p 8080:8080 -v /home/model:/home/mind/model -e OCTOPUS_MODEL=/home/mind/model custom_engine:v1

d. 在本地机器上启动另一个终端，执行以下验证指令，得到符合预期的推理结果。

   curl -k https://127.0.0.1:8080/*${推理服务的请求路径}*

说明

本地调试完成后，每次都需将镜像推送至新版本，使用二次推送镜像功能会导致推理服务镜像更新不生效。

制作开发环境镜像

需要在Dockerfile文件中添加uid为1000的用户**ma-user**和gid为100的用户组**ma-group**。如果基础镜像中uid 1000或者gid 100已经被其他用户和用户组占用，需要将其对应的用户和用户组删除。制作开发环境的自定义镜像时，基础镜像需满足如下规范：

- 使用Dockerhub等官方源发布的镜像。
- 使用Ubuntu 18.04、Ubuntu 20.04、Ubuntu 22.04作为基础镜像。

Dockfile示例如下，以Ubuntu 20.04为例：

```
# Replace it with the actual image version.
FROM ubuntu:20.04

# Set the user ma-user whose UID is 1000 and the user group ma-group whose GID is 100
USER root
RUN default_user=$(getent passwd 1000 | awk -F ':' '{print $1}') || echo "uid: 1000 does not exist" && \
    default_group=$(getent group 100 | awk -F ':' '{print $1}') || echo "gid: 100 does not exist" && \
    if [ ! -z ${default_user} ] && [ ${default_user} != "ma-user" ]; then \
        userdel -r ${default_user}; \
    fi && \
    if [ ! -z ${default_group} ] && [ ${default_group} != "ma-group" ]; then \
        groupdel -f ${default_group}; \
    fi && \
    groupadd -g 100 ma-group && useradd -d /home/ma-user -m -u 1000 -g 100 -s /bin/bash ma-user && \

# Grant the read, write, and execute permissions on the target directory to the user ma-user.
chmod -R 750 /home/ma-user

#Configure the APT source and install the ZIP and Wget tools (required for installing conda).
RUN mv /etc/apt/sources.list /etc/apt/sources.list.bak && \
    echo "deb http://repo.huaweicloud.com/ubuntu/ bionic main restricted\ndeb http://
repo.huaweicloud.com/ubuntu/ bionic-updates main restricted\ndeb http://repo.huaweicloud.com/ubuntu/
bionic universe\ndeb http://repo.huaweicloud.com/ubuntu/ bionic-updates universe\ndeb http://
repo.huaweicloud.com/ubuntu/ bionic multiverse\ndeb http://repo.huaweicloud.com/ubuntu/ bionic-updates
multiverse\ndeb http://repo.huaweicloud.com/ubuntu/ bionic-backports main restricted universe multiverse
\ndeb http://repo.huaweicloud.com/ubuntu bionic-security main restricted\ndeb http://
repo.huaweicloud.com/ubuntu bionic-security universe\ndeb http://repo.huaweicloud.com/ubuntu bionic-
security multivers e" > /etc/apt/sources.list && \
apt-get update && \
apt-get install -y zip wget

#Modifying the system Configuration of the image (required for creating the Conda environment)
RUN rm /bin/sh && ln -s /bin/bash /bin/sh

#Switch to user ma-user , download miniconda from the Tsinghua repository, and install miniconda in /
home/ma-user.
USER ma-user
RUN cd /home/ma-user/ && \
    wget --no-check-certificate https://mirrors.tuna.tsinghua.edu.cn/anaconda/miniconda/Miniconda3-4.6.14-
Linux-x86_64.sh && \
    bash Miniconda3-4.6.14-Linux-x86_64.sh -b -p /home/ma-user/anaconda3 && \
    rm -rf Miniconda3-4.6.14-Linux-x86_64.sh

#Configure the conda and pip sources
RUN mkdir -p /home/ma-user/.pip && \
    echo -e "channels:\n  - defaults\nshow_channel_urls: true\ndefault_channels:\n  - https://
mirrors.tuna.tsinghua.edu.cn/anaconda/pkgs/main\n  - https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgs/r
\n  - https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgs/msys2" > /home/ma-user/.condarc && \
    echo -e "[global]\nindex-url = https://pypi.tuna.tsinghua.edu.cn/simple\n[install]\ntrusted-host = https://
```

```
pypi.tuna.tsinghua.edu.cn" > /home/ma-user/.pip/pip.conf

#Create the conda environment and install the Python third-party package. The ipykernel package is mandatory for starting a kernel.
RUN source /home/ma-user/anaconda3/bin/activate && \
    conda create -y --name pytorch_1_8 python=3.7 && \
    conda activate pytorch_1_8 && \
    pip install torch==1.8.1 torchvision==0.9.1 && \
    pip install ipykernel==6.7.0 && \
    conda init bash && \
    conda deactivate

#Install packages like FFmpeg and GCC
USER root
RUN apt-get -y install ffmpeg && \
    apt -y install gcc-8 g++-8
```

说明

- 如需使用Jupyter功能还需手动添加IPython Kernel并绑定到当前的Python环境保证交互式指令能正常运行，否则打开“JupyterLab > 新建Notebook”会选不到kernel。

  具体步骤可参考**在Notebook中添加自定义IPython Kernel**。
- 不满足以上镜像规范，所制作的镜像可能会出现故障。常见故障排查：
  - 用户自定义镜像没有ma-user用户及ma-group用户组；
  - 用户自定义镜像中/home/ma-user目录，属主和用户组不是ma-user和ma-group；
  - 用户自定义镜像必须满足用户目录/home/ma-user权限为750，不能为其他权限；
  - 用户自定义镜像使用远程SSH功能，OpenSSH版本要兼容或高于8.0；
  - 用户制作的自定义镜像，在本地执行docker run启动，无法正常运行；
  - 用户自行安装了Jupyterlab服务导致冲突的，需要用户本地使用Jupyterlab命令罗列出相关的静态文件路径，删除并且卸载镜像中的Jupyterlab服务；
  - 用户自己业务占用了开发环境官方的8888、8889端口的，需要用户修改自己的进程端口号；
  - 用户的镜像指定了PYTHONPATH、sys.path导致服务启动调用冲突的，需在实例启动后，再指定PYTHONPATH、sys.path；
  - 用户使用了已开启sudo权限的专属池，使用自定义镜像时，sudo工具未安装或安装错误；
  - 用户使用的cann、cuda环境有兼容性问题；
  - 用户的docker镜像配置错误、网络或防火墙限制、镜像构建问题（文件权限、依赖缺失或构建命令错误）等原因导致的。

9.5 制作仿真镜像

9.5.1 自定义评测镜像制作

Octopus仿真服务平台定义了一些proto接口，用于支持用户自定义评测等功能。这些自定义功能通常以镜像的形式上传到云仿真平台，然后参与到业务运行流程中。

本文档对常见业务功能的镜像制作进行指导说明。

说明

- 自定义评测镜像涉及的样例代码，如有需要，请联系相关人员。
- 算法镜像和评测镜像cmd以用户创建项目时输入的运行命令为准，仿真器镜像不支持cmd自定义以后台默认的运行命令为准。Entrypoint只在算法镜像生效，评测镜像和仿真器镜像不支持配置entrypoint。
- 算法与仿真器需要采用TCP协议进行通信。

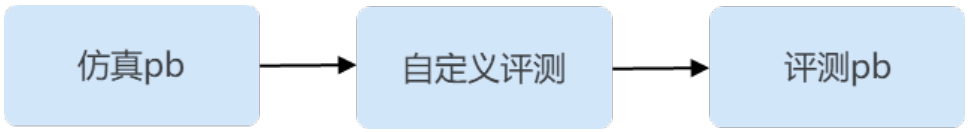
延时评测

仿真器输出的仿真过程数据会按照OSI的GroundTruth格式存储为pb文件，根据创建任务配置时是否选择使用datahub，该仿真pb有两种格式：

1. 使用datahub时，该仿真pb的每帧数据是GroundTruth结构，然后按照OSI标准推荐的存储格式，先按小端顺序存储4个字节，该4个字节表示下一帧GroundTruth总的字节数，然后继续4个字节，更下一帧GroundTruth的字节数，以此类推。
2. 未使用datahub时，该仿真pb按照sim_osi.proto定义的格式进行存储，所有仿真数据帧GroundTruth存储到一个列表字段“frames”中。

仿真平台定义了sim_osi.proto和eva.proto，用于支持用户自定义评测的功能，具体的proto字段说明见请联系接口人获取仿真服务的proto协议。

图 9-3 自定义评测流程



如上图所示，自定义评测算法以仿真pb文件作为输入，进行评测逻辑处理后，将评测结果写成评测pb。

其中仿真pb是通过八爪鱼提供的sim_osi.proto进行序列化和反序列化，评测pb是通过八爪鱼提供的eva.proto进行序列化和反序列化的。

自定义评测算法的实现有如下几个步骤：

步骤1 在代码内通过SIM_OSI_PATH环境变量获取仿真pb路径，通过EVA_PATH环境变量获取评测pb路径。

通过文件Open的方式打开仿真pb路径，读取字节流，根据文件后缀读取仿真pb文件，如果是.osi后缀结尾的文件，则表示该仿真pb是按照OSI标准格式存储，需要先读取4个字节，获取下一帧GroundTruth长度，然后再读取一定长度的帧数据字节，用GroundTruth进行反序列化。如果是.pb结尾的文件，则利用sim_osi.proto中的SimData反序列化仿真pb中的内容。该步骤会得到一个SimData的内存对象，用户通过访问对象中的字段即可获取自己关注的数据。

步骤2 仿真pb包含仿真器输出的整个仿真过程数据，用户处理根据自身评测逻辑处理所有帧数据。

步骤3 用户自定义的评测指标包含通过，不通过等结果，将该结果写入到eva.proto中的Evaluation类中，然后通过文件Open的形式打开评测pb路径，将评测结果写成评测pb文件。

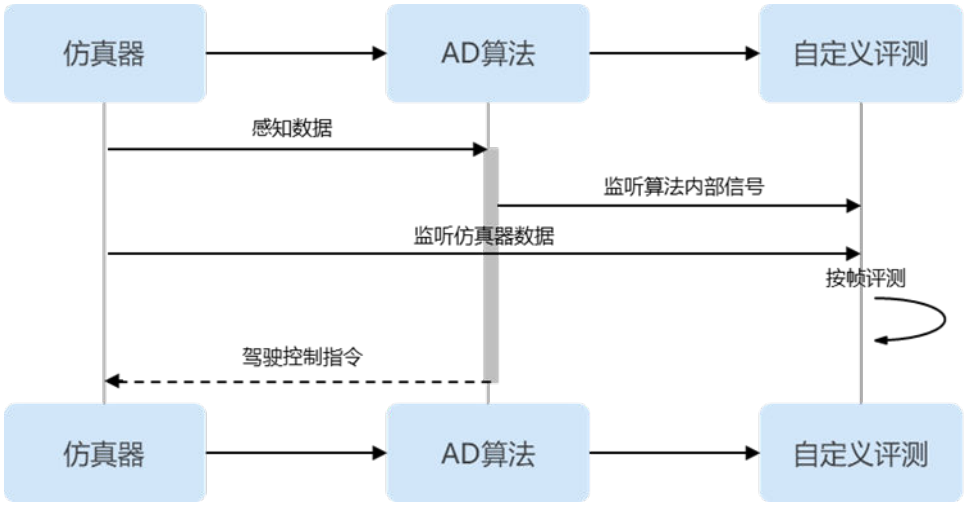
步骤4 写成评测pb文件后，自定义评测镜像的工作就完成了，仿真平台的控制程序在运行自定义评测容器时会主动将评测pb文件上传到对象存储中，前端通过下载该评测pb文件进行解析，可以将自定义评测结果和内置评测结果一样完全兼容地进行展示。

----结束

评测算法代码开发完成后，将代码构建成算法镜像上传到仿真平台评测管理模块即可被仿真任务使用。在制作评测算法镜像的Dockerfile中，建议将评测代码编译成的二进制文件COPY到系统的/usr/bin目录下，便于在前端界面填写评测镜像的运行命令时直接填写该二进制文件的名称即可。在镜像中新建一个shell脚本来运行评测代码也是可以接受的方案。

实时评测

图 9-4 实时评测



仿真平台定义了eva.proto，用于支持用户自定义实时评测的功能，具体的proto字段说明见其他相关文档。

自定义实时评测算法的实现有如下几个步骤：

步骤1 代码内实现与仿真器的通信，实时接收仿真器的帧数据，也可同时接收仿真器和AD算法的数据。同时，自定义评测算法也可选择对接八爪鱼提供的Datahub服务，通过Grpc协议，连接`SubscribeGroundTruth`接口实时获取GroundTruth数据。

步骤2 用户自定义的评测指标包含通过，不通过等结果，将该结果写入到eva.proto中的Evaluation类中。

步骤3 处理每帧数据，不断更新评测结果。具体是更新`eva.proto`中的Evaluation`对象的各个字段值。

步骤4 捕获SIGINT信号，当捕获到该信号时，代表仿真结束，将结束时的这帧的评测结果作为最终的评测结果，通过EVA_PATH环境变量获取评测pb路径，将评测结果写入到评测pb文件中。

步骤5 写成评测pb文件后，自定义实时评测镜像的工作就完成了。仿真平台的控制程序，会主动将自定义评测结果和内置评测结果融合后的评测pb文件上传到对象存储中，前端可通过下载pb文件进行解析。

----结束

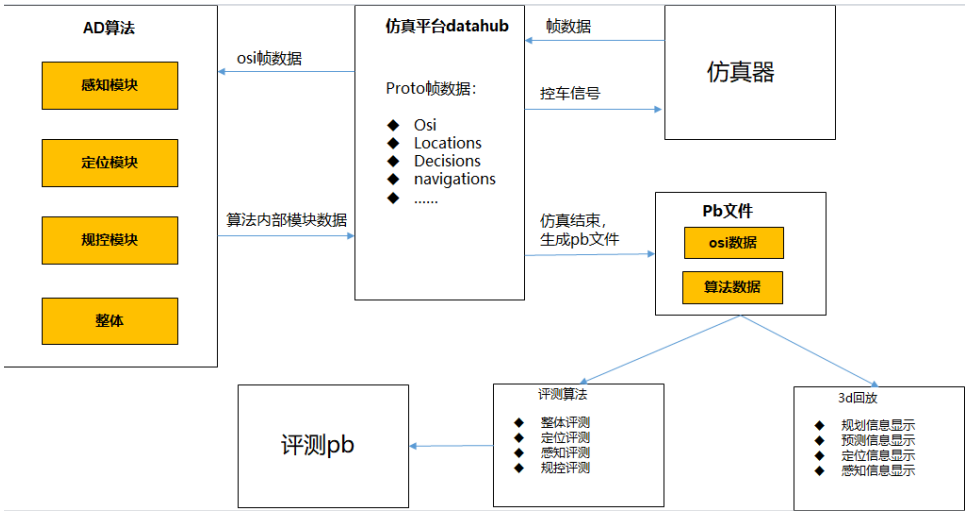
评测算法代码开发完成后，将代码构建成算法镜像上传到仿真平台评测管理模块即可被仿真任务使用。在制作评测算法镜像的Dockerfile中，建议将评测代码编译成的二进制文件COPY到系统的/usr/bin目录下，便于在前端界面填写评测镜像的运行命令时直接填写该二进制文件的名称即可。在镜像中新建一个shell脚本来运行评测代码也是可以接受的方案。

另外，对于仿真器B的configure.xml文件的以下两个字段需要更改为：

```
<ControlInLoop value="1" type="1" />
<CoSimuSocketMode value="2" />
```

9.5.2 与 datahub 对接的算法镜像制作

图 9-5 算法镜像



如上图所示，算法与仿真平台datahub通过grpc连接，通过接收osi数据作为输入，并将算法内部信号输出到datahub。

仿真平台可以生成仿真的osi和算法pb，用于3d回放展示和算法的白盒化评测。

具体grpc连接datahub的代码可以参考八爪鱼提供的demo样例。

9.5.3 评测算法的自研 proto 接口

背景

Octopus内置一套评测算法，用于对自动驾驶系统的性能表现进行多维度评测。内置评测算法的评测结果按照eva.proto中的定义，序列化成pb文件保存起来。

Octopus仿真平台的前端通过解析评测pb对评测结果进行展示，目前控制台展示主要分为两大方面：

- 各个评测指标的通过/未通过/无效的结果展示。
- 仿真过程中关键数据的时间序列曲线图展示。

另外，对于用户自研的评测算法的评测结果，也可以按照eva.proto，序列化成pb文件保存起来，这样Octopus的仿真平台前端能够展示用户自研评测算法的评测结果。

eva.proto 的关键字段解释

在利用Octopus仿真平台进行批量仿真时，每个场景都会有一个评测结果，单个场景的评测结果全部保存在如数据类图所示的Evaluation类中。

Evaluation类包含以下的7个字段。

表 9-5 Evaluation 类包含的字段

字段	说明
version	用于表示当前Octopus_eva.proto的版本。
score	用户保存批量仿真中单个场景下评测算法计算出的评测得分，Octopus的评分取值为[0, 100]。
avg_speed	单个场景下自动驾驶算法控制下主车的平均车速，单位m/s。
distance	单个场景下自动驾驶算法控制下主车的行驶里程，单位m。
vis	主车关键状态量的（如速度、加速度）的时间序列数据，用于前端进行可视化曲线展示。
metrics	Octopus内置了一批大类评测指标，并且每个大类评测指标下会多个子类指标。 该字段用于保存每个大类和子类评测指标的通过/未通过/无效的结果状态。 其中无效状态表示该指标在特定场景下是没意义的，如在没有交通灯的场景下，主车在交通灯前的行为检测是没有意义的。
source	是一个Source枚举类型，表示该评测算法的结果来源类型。 source共有6种类型，具体参考**source**。

表 9-6 source

字段	说明
SOURCE_UNSPECIFIED	表示评测算法类型未定义。
SOURCE_CUSTOMIZED_REALTIME	用户自定义实时评测算法结果。
SOURCE_CUSTOMIZED_OFFLINE	用户自定义延时评测算法结果。
SOURCE_DEFAULT_REALTIME	八爪鱼内置实时评测算法结果。
SOURCE_DEFAULT_OFFLINE	八爪鱼内置延时评测结果。

字段	说明
SOURCE_MERGED	融合后的评测结果。

Evaluation中两个比较关键且复杂的字段是vis和metrics，其中vis字段类型是Visualization，metrics字段的类型是repeated Metric。

Visualization和Metric的成员组成如上数据类图所示，各个关键成员的含义如下所示。

Visualization类型包含的各个字段及其含义如下所示。

表 9-7 Visualization 类型字段

字段	说明
sim_times	仿真过程的时间序列点，以仿真开始时刻（t0）为0点，相邻时刻的时间间隔为仿真器的周期，单位为秒（s）。
frame_nums	每个仿真时刻对应的数据帧数，表明这是第几帧的数据。
stats	统计类型的数据值，是指某数据在一段时间内统计值才有展示分析的意义，如加速度均方根值（arms），随着仿真时刻不断后移，需要不断计算初始时刻（t0）到当前仿真时刻（t）的arms值。 stats字段的类型是Statistic，Statistic类型的成员具体参考**stats**。
vector	向量类型的数据值，是指每个时刻都会有一个对应的数据值，如加速度、速度等物理量。 vector字段的类型是Vector，Vector类型的成员具体参考**vector**。

表 9-8 stats

字段	说明
type	是一个Type的枚举类型，表示该数据是哪种Octopus内置的Statistic类型。
value	该数据在每个仿真时刻对应的数值。
display_name	存储用户自定义的Statistic类型名称，用户自定义的数据无需对type赋值。
source	表示该数据的评测结果来源类型。
importance	表示该数据是重要类型还是次要类型。
module	表示该数据是来源于哪个算法模块。
performance	表示该数据是表示的哪个类别的算法性能。

表 9-9 vector

字段	说明
type	是一个Type的枚举类型，表示该数据是哪种Octopus内置的Vector类型。
value	该数据在每个仿真时刻对应的数值。
display_name	存储用户自定义的Vector类型名称，用户自定义的数据无需对type赋值。
source	表示该数据的评测结果来源类型。
importance	表示该数据是重要类型还是次要类型。
module	表示该数据是来源于哪个算法模块。
performance	表示该数据是表示的哪个类别的算法性能。

Metric类型包含的各个字段及其含义如下所示。

表 9-10 Metric 类型字段

字段	说明
type	是一个Enum类型，表示该数据是哪种Octopus内置的大类指标类型。
status	是一个Result的枚举类型，用于表示该大类指标的评测结果。 其中Result有三种类型，分别为： ● RESULT_UNSPECIFIED：该状态的结果表示评测指标是无效的，如对于红绿灯检测指标，如果仿真完成后评测函数发现场景中不存在红绿灯，则该项指标的评测是没有意义的，故评测结果展示为无效。 ● RESULT_PASSED: 该状态的结果表示评测结果为通过。 ● RESULT_FAILED：该状态的结果表示评测结果为未通过。

字段	说明
anomalies	用于保存每个大类指标下对应的子类指标发生异常的时间点，其数据类型为repeated Anomaly，其中Anomaly类型成员如下所示： • status: 是一个Result类型，与上述大类指标的status的表示相同，该字段用于存储子类指标的三种不同状态结果。 • subtype：是一个Subtype的枚举类型，表示该子类指标是哪种Octopus内置的子类指标类型。 其中，对于某些没有子类指标的大类指标来说，其subtype赋值为SUBTYPE_UNSPECIFIED。 • point_type：是一个PointType的枚举类型，表示该子类指标发生特殊状态（一般是指发生异常）时的时刻点用哪种形式存储起来。 其中PointType共有5种类型，参考**PointType**。 • Points：是repeated float类型，保存每个子类指标发生特殊状态的具体时间点的值。 • Stats_indices：是repeated ui32类型，表示该子指标关联的统计类数值列表。 • Vector_indices：是repeated ui32类型，表示该子指标关联的时间序列数值列表。
display_name	存储用户自定义的Anomaly类型名称，表示其自定义的子类指标，用户自定义的数据无需对sub_type赋值。
importance	表示该指标的重要程度。具体参考**importance**。
source	表示该数据的评测结果来源类型。
module	表示该数据是来源于哪个算法模块。具体参考**module**。
performance	表示该数据是表示的哪个类别的算法性能。具体参考**performance**。

表 9-11 PointType

字段	说明
POINT_TYPE_UNSPECIFIED	未定义的类型，遵循proto定义格式，Octopus未使用该类型，用户可根据需要对该类型赋予特定含义。
POINT_TYPE_POINT	表示该子类指标的异常时间点是离散的时间点形式，在任何时刻都可能发生异常。
POINT_TYPE_REGION	表示该子类指标的异常时间点是区间形式，一旦在某个时刻开始发生异常，则在随后一段时间内都会处于异常状态。
POINT_TYPE_ALL	表示该类指标的异常时间点是布尔形式的，从仿真开始到当前时刻的状态要么是完全通过，要么全过程都是异常的，统计类型的指标需要以这种形式表示。

字段	说明
POINT_TYPE_NORMAL	该类型与其他类型相反，如果该类型的点存在，则表示对应的子类指标是通过的，Octopus用该类型保存主车到达终点的时间值。

表 9-12 importance

字段	取值	说明
CATEGORY_UNSPECIFIED	0	未定义
CATEGORY_MAJOR	1	主要
CATEGORY_MINOR	2	次要

表 9-13 module

字段	取值	说明
MODULE_UNSPECIFIED	0	未定义
MODULE_NAVIGATION	1	导航
MODULE_LOCATION	2	定位
MODULE_PERCEPTION	3	感知
MODULE_PREDICTION	4	预测
MODULE_DECISION	5	决策
MODULE_PLANNING	6	规划
MODULE_CONTROL	7	控制
MODULE_WHOLE	8	整车

表 9-14 performance

字段	取值	说明
PERFORMANCE_UNSPECIFIED	0	未定义
PERFORMANCE_SAFETY	1	安全性
PERFORMANCE_REGULATION	2	合规性
PERFORMANCE_COMFORT	3	舒适性
PERFORMANCE_INTELLIGENCE	4	智能型

10 运维配置

10.1 集群纳管

10.1.1 查看集群纳管

运维配置提供集群纳管，由平台管理员账号进行管理和配置。集群提供多种节点的混合部署，基于高性能网络模型提供全方位、多场景、安全稳定的容器运行环境，平台可以将集群统一纳管，更方便查看节点资源使用量和修改节点用途，以及设置资源规格。其中，cce-user-job集群需强制纳管以适配资源规格。如果用户直接通过后台操作k8s集群及节点造成环境错乱，则需承担责任。

集群纳管相关操作

在“集群纳管”列表，可对纳管进行以下操作。

表 10-1 集群纳管相关操作

任务	操作步骤
查询纳管	在搜索输入框中输入搜索条件，按回车键即可查询。
筛选纳管	在“用途”栏，可按用途类型筛选展示集群纳管。
查看纳管详情	单击集群名称cce-user-job，查看纳管详情。
编辑纳管集群	单击操作栏“编辑”，对纳管进行编辑，可以修改纳管用途。

10.1.2 资源管理

集群纳管详情中展示集群的节点资源和资源规格分配情况。给节点增加标签，确认节点的用途，在资源规格中，给各个资源用途分配资源规格，支撑任务执行。资源不足时，用户需要单独**购买扩展资源包**。

步骤1 用平台管理员账号登录Octopus平台。

步骤2 在左侧菜单栏中单击“运维配置 > 集群纳管 ”。

步骤3 单击CCE类型的集群名称，如“cce-user-job”，进入纳管详情界面。

步骤4 单击节点列表操作栏中的“修改标签”，可对节点的用途和标签进行修改。

步骤5 选择资源规格页签，单击“新增规格”，选择并填写必要参数，即可新建一种资源规格，可以创建仅含CPU、内存的资源规格或者包含GPU、CPU、内存的资源规格。

- 资源用途：下拉选择，当前可选择数据转换、回放仿真、训练任务、模型评测、模型编译、预标注和数据脱敏。
- 节点规格：下拉选择，为新建的资源规格定上限，确保资源规格有节点适配，根据用途从集群中查询。
- 资源规格： 填写资源用途所需要的资源量，各任务推荐最小资源规格如下，用户可在节点规格范围内灵活配置。

表 10-2 资源规格

资源用途	规格
数据转换	1Core_1GiB、1Gpu_2Core_4GiB
回放仿真	1Core_1GiB、1Gpu_2Core_4GiB
训练任务	1Gpu_4Core_16GiB
模型评测	1Gpu_2Core_8GiB
模型编译	2Core_4GiB
预标注	1GPU_2Core_8GiB
数据脱敏	1Core_1GiB **说明** 内置容器需要1Gpu_12Core_48GiB。

说明

建议规格中的GPU类型和实际使用的GPU一致，避免管理混乱。

步骤6 选择资源规格页签，单击操作栏“删除”，可删除规格。

----结束

说明

并行仿真的资源规格为仿真器、算法等多个程序的资源规格总和。默认的最小资源规格如下：仿真器1Core_1GiB，内置算法2Core_1GiB，默认控制、评测等程序在0.1-0.2Core_256MB；当不接入算法和datahub时，支持的最小资源规格约1.3Core_1.5GiB，当接入算法时，用户需要根据自己的算法所需资源规格来推定并行仿真的总资源规格。

资源规格相关操作

在“资源规格管理”列表，可对纳管进行以下操作。

表 10-3 资源规格相关操作

任务	操作步骤
筛选规格	在“资源用途”栏，可按用途类型筛选规格并展示目标规格。
删除规格	单击操作栏“删除”，可删除规格。

11 工作空间

11.1 工作空间简介

在使用Octopus自动驾驶云服务之前，用户可通过工作空间实现资源隔离管理。

default工作空间为系统预置默认主工作空间，授权类型为Public，默认主账号和所有子账号子用户可用，不支持创建、编辑和删除。

Octopus预置3种系统策略类型的角色，方便用户创建用户组时进行授权。

- Octopus FullAccess：包含Octopus服务和依赖的云服务（不包含OBS）所有权限。被授权该角色的应为八爪鱼管理员用户组。
- Octopus CommonOperations：包含Octopus服务除“工作空间管理”外所有权限和依赖的云服务（不包含OBS）所有权限。被授权该角色的应为八爪鱼普通用户组。
- Octopus ReadOnlyAccess：包含Octopus服务和依赖云服务（不包含OBS）的只读权限。被授权该角色的应为八爪鱼浏览者或演示用户组。

说明

当前仿真服务、运维配置暂未适配工作空间。

11.2 创建工作空间

新建工作空间

步骤1 在左侧菜单栏中，单击“工作空间”。

步骤2 单击“新建工作空间”。

步骤3 完成工作空间基本信息。

- 空间名称：输入工作空间的名称，只能包含数字、英文、中文、下划线、中划线。
- 描述：简单描述空间名称，最大长度为255。
- 授权类型：当前可选择“Internal”。
  工作空间授权类型分为“Public”和“Internal”。

  - “Public”仅在租户（主账号和所有子账号）内部访问。
  - “Internal”为创建者、主账号、指定的子账号可访问。当授权类型为Internal时，需要指定可访问的子账号的账号名或者账号id，可选择多个。
- 授权对象：请选择需要授权的对象。当创建的IAM子用户没有IAM管理权限时，无法选择授权对象，推荐为IAM子用户添加IAM ReadOnlyAccess策略。

步骤4 单击“创建”，在工作空间列表查看创建好的工作空间。

----结束

工作空间相关操作

表 11-1 工作空间相关操作

任务	操作步骤
编辑工作空间	单击操作栏中的“编辑”，编辑工作空间。支持修改空间名称、描述和授权对象。授权最长10分钟后生效，如遇授权对象访问错误，请稍后重试。
删除工作空间	单击操作栏中的“删除”，删除工作空间。工作空间删除后会默认清理该工作空间下的所有资源。删除操作无法恢复，请谨慎操作。
查询工作空间	支持按状态和输入关键字两种方式查询工作空间。 ● 按状态查询：在“请选择状态”下拉列表选择工作空间状态，即可查询该状态的工作空间。 ● 输入关键字查询：在搜索输入框中输入搜索条件，按回车键即可查询工作空间。
查看工作空间详情	单击工作空间名称，查看工作空间详情。

11.3 查看授权对象

步骤1 在左侧菜单栏中，单击“工作空间”。

步骤2 单击目标工作空间名称，查看工作空间详情。

步骤3 在工作详情界面，查看当前工作空间的授权对象。

----结束

12 审计日志

12.1 支持云审计的关键操作

通过云审计服务，您可以记录与Octopus相关的操作事件，便于日后的查询、审计和回溯。

前提条件

已开通云审计服务。

支持审计的关键操作列表

表 12-1 云审计服务支持的公共服务关键操作列表

操作名称	资源类型	事件名称
新增纳管集群	octopus	addCluster
更新纳管集群信息	octopus	updateCluster
解除纳管集群	octopus	deleteCluster
新建资源规格	octopus	createResourceSpec
删除资源规格	octopus	deleteResourceSpec
更新集群节点标签	octopus	updateClusterNodeLabel
创建镜像仓库	octopus	createImageRepo
创建镜像仓库版本	octopus	createImageVersion
创建镜像推送账号	octopus	createDockerAccount
更新镜像仓库	octopus	updateImageRepo
删除镜像仓库	octopus	deleteImageRepo
删除镜像仓库版本	octopus	deleteImageVersion
更新镜像仓库版本	octopus	updateImageVersion

操作名称	资源类型	事件名称
创建标签	octopus	createDataTag
批量导入标签	octopus	importDataTags
删除标签	octopus	deleteDataTag
编辑标签	octopus	updateDataTag
批量创建标签	octopus	createDataTags

表 12-2 云审计服务支持的数据资产关键操作列表

操作名称	资源类型	事件名称
下载地图	octopus	downloadDataMap
创建地图信息	octopus	createDataMap
更新地图信息	octopus	updateDataMap
删除地图包	octopus	deleteDataMap
创建车辆	octopus	createVehicle
更新车辆	octopus	updateVehicle
删除车辆	octopus	deleteVehicle
创建车队	octopus	createVehicleFleet
更新车队	octopus	updateVehicleFleet
删除车队	octopus	deleteVehicleFleet
创建标定	octopus	createCalibration
删除标定	octopus	deleteCalibration
创建标定项	octopus	createCalibrationItem
删除标定项	octopus	deleteCalibrationItem
创建仿真场景	octopus	createDataEpisodeScenario
创建场景片段	octopus	createDataScenarioSegment
删除场景片段	octopus	deleteDataScenarioSegment
下载日志	octopus	downloadDataLog
下载回放数据	octopus	downloadReplayDataBlocks
下载resim回放数据	octopus	downloadReplayResimBlocks
下载mp4回放数据	octopus	downloadReplayVideoSlice

操作名称	资源类型	事件名称
下载回放数据	octopus	downloadReplayData
创建递送订单	octopus	createExpressOrder
触发订单传输数据	octopus	startTransferExpressOrderData
更新订单任务配置	octopus	updateExpressOrderTaskConfig
创建数据集	octopus	createDataset
修改数据集	octopus	updateDataset
删除数据集	octopus	deleteDataset
创建数据集导出任务	octopus	createDatasetExportTask
下载数据集版本的日志文件	octopus	downloadDatasetVersionLog
下载数据集版本文件数据	octopus	downloadDatasetVersionData
创建数据集版本	octopus	createDatasetVersion
删除数据集版本	octopus	deleteDatasetVersion
创建版本导出任务	octopus	createDatasetVersionExportTask
删除数据集导出任务	octopus	deleteDatasetExportTask
创建数据集格式	octopus	createDatasetFormatType
删除数据集格式	octopus	deleteDatasetFormatType
创建通用存储	octopus	createDataWarehouse
修改通用存储	octopus	updateDataWarehouse
批量删除通用存储	octopus	deleteDataWarehouses
批量删除某个通用存储中的多个数据处理任务	octopus	deleteDataWarehouseTasks
创建通用存储数据	octopus	createDataWarehouseData
更新通用存储的指定数据	octopus	updateDataWarehouseData
创建通用存储中视频文件分片	octopus	createDataWarehouseVideoSlice
删除通用存储中指定数据	octopus	deleteDataWarehouseData

操作名称	资源类型	事件名称
创建通用存储导出任务	octopus	createDataWarehouseExportTask
删除通用存储关联的导出任务	octopus	deleteDataWarehouseExportTask
批量删除通用存储自定义属性	octopus	deleteDataWarehouseCustomAttributes
新增通用存储自定义属性	octopus	createDataWarehouseCustomAttribute
下载通用存储文件数据	octopus	downloadDataWarehouseData
删除缓存数据记录	octopus	deleteDataCache
数据集缓存加速	octopus	publishDataset
创建模型仓库	octopus	createModelRepository
创建模型版本	octopus	createModelVersion
删除模型版本	octopus	deleteModelVersion
删除模型仓库	octopus	deleteModelRepository
更新模型仓库	octopus	updateModelRepository
打包模型版本	octopus	packageModelVersion
更新模型版本打包状态	octopus	updateModelVersionPackagingResult
删除模型版本中的文件	octopus	deleteModelVersionFile
保存模型版本文件	octopus	saveModelVersionFile
更新模型版本上传状态	octopus	updateModelVersionUploadResult
下载模型版本文件	octopus	downloadModelVersionFile
更新模型版本状态	octopus	updateModelVersionStatus

表 12-3 云审计服务支持的数据处理关键操作列表

操作名称	资源类型	事件名称
导入数据包	octopus	importPackage
更新数据包状态	octopus	updatePackageStatus
删除数据包	octopus	deletePackage

操作名称	资源类型	事件名称
更新数据包标签	octopus	updatePackageTags
删除导入任务	octopus	deleteImportPackageTask
创建数据包上传任务	octopus	createPackageUploadTask
创建分片上传链接	octopus	createPartSignedUrl
完成分片上传	octopus	completePackageUpload
上传数据包元数据	octopus	uploadPackageMetaInfo
创建算子	octopus	createDataProcessor
更新算子	octopus	updateDataProcessor
删除算子	octopus	deleteDataProcessor
创建算子作业	octopus	createDataJob
更新算子作业	octopus	updateDataJob
批量更新算子作业	octopus	updateDataJobs
删除算子作业	octopus	deleteDataJob
批量删除作业	octopus	deleteDataJobs
创建内置作业	octopus	createDataSystemJob
操作队列	octopus	updateDataJobPriority
批量操作队列	octopus	updateDataJobsPriority

表 12-4 云审计服务支持的标注服务关键操作列表

操作名称	资源类型	事件名称
新建标注团队	octopus	createAnnotationGroup
编辑标注团队	octopus	updateAnnotationGroup
删除标注团队	octopus	deleteAnnotationGroup
从数据集创建标注任务	octopus	createAnnotationBatchTaskFromDataset
导出批次任务列表报告	octopus	downloadAnnotationBatchTasksReport
批量删除标注批次任务	octopus	deleteAnnotationBatchTasks
创建拆分的批次任务	octopus	createAnnotationSubTasksFromBatchTask

操作名称	资源类型	事件名称
下载子任务统计报告	octopus	downloadAnnotationSubTaskStatisticReport
更新批次任务的标注规范	octopus	updateAnnotationBatchTaskSpecification
增加需求方字典值	octopus	createAnnotationRequester
删除需求方字典值	octopus	deleteAnnotationRequester
保存标注信息	octopus	updateAnnotationData
将3D坐标点转换成2D坐标点	octopus	updateAnnotationCube3dToCube2d
将2D坐标点转换成3D坐标点	octopus	updateAnnotationCube2dToCube3d
创建标注物	octopus	createAnnotationMeta
批量删除标注物	octopus	deleteAnnotationMetas
编辑标注物	octopus	updateAnnotationMeta
保存标注时间	octopus	updateAnnotationSubTaskTime
创建标注项目	octopus	createAnnotationProject
批量删除标注项目	octopus	deleteAnnotationProjects
修改标注项目	octopus	updateAnnotationProject
下载标注项目人员统计报告	octopus	downloadAnnotationProjectUserStatisticReport
创建标注脚本	octopus	createAnnotationScript
编辑标注脚本	octopus	updateAnnotationScript
批量删除标注脚本	octopus	deleteAnnotationScripts
批量删除标注任务	octopus	deleteAnnotationSubTasks
设置标注任务的审核比例	octopus	updateAnnotationSubTaskQualityInspectionRatio
更新标注子任务流程	octopus	updateAnnotationSubTaskProcedure
导出标注数据成数据集	octopus	downloadAnnotationDataToDataset
创建标注项目规范	octopus	createAnnotationProjectSpecification
批量删除标注项目规范	octopus	deleteAnnotationProjectSpecifications
更新任务队列顺序	octopus	updateAnnotationTaskQueue

操作名称	资源类型	事件名称
批量修改任务优先级	octopus	updateAnnotationTaskPriority
创建标注模板	octopus	createAnnotationTemplate
批量删除标注模板	octopus	deleteAnnotationTemplates
编辑标注模板	octopus	updateAnnotationTemplate
创建标注用户	octopus	createAnnotationUser
编辑标注用户	octopus	updateAnnotationUser
批量删除标注用户	octopus	deleteAnnotationUsers
获取标注文件	octopus	downloadAnnotationFile
新增团队中用户	octopus	createAnnotationGroupUser
删除团队中用户	octopus	deleteAnnotationGroupUser
下载标注批次任务日志	octopus	downloadAnnotationBatchTaskLog
批量创建标注预审核任务	octopus	createAnnotationAutoInspectTasks
批量删除标注预审核任务	octopus	deleteAnnotationAutoInspectTasks
下载标注预审核报告、日志	octopus	downloadAnnotationInspectLogReport

表 12-5 云审计服务支持的训练服务关键操作列表

操作名称	资源类型	事件名称
创建训练算法	octopus	createTrainAlgorithm
更新训练算法信息	octopus	updateTrainAlgorithm
删除训练算法	octopus	deleteTrainAlgorithm
删除训练算法文件	octopus	deleteTrainAlgorithmFile
保存训练算法文件	octopus	saveTrainAlgorithmFile
打包训练算法	octopus	packageTrainAlgorithm
更新训练算法打包结果	octopus	updateTrainAlgorithmPackagingResult
下载算法压缩包	octopus	downloadTrainAlgorithmPackage

操作名称	资源类型	事件名称
更新训练算法上传结果	octopus	updateTrainAlgorithmUploadResult
删除编译模型	octopus	deleteCompiledModel
打包编译模型	octopus	packageCompiledModel
更新打包编译模型打包结果	octopus	updateCompiledModelPackagingResult
下载编译模型	octopus	getDownloadPackagePath
创建模型编译机	octopus	createModelCompiler
删除模型编译镜像	octopus	deleteModelCompiler
更新模型编译镜像	octopus	updateModelCompiler
创建模型编译任务	octopus	createModelCompileTask
重启模型编译任务	octopus	restartModelCompileTask
停止模型编译任务	octopus	stopModelCompileTask
删除模型编译任务	octopus	deleteModelCompileTasks
修改模型编译任务优先级	octopus	updateModelCompileTaskPriority
下载模型编译任务日志	octopus	downloadModelCompileTaskLog
创建模型评测对比任务	octopus	createEvaluationComparisonTask
停止模型评测对比任务	octopus	stopEvaluationComparisonTask
重启模型评测对比任务	octopus	restartEvaluationComparisonTask
删除模型评测对比任务	octopus	deleteEvaluationComparisonTask
下载模型评测对比文件	octopus	downloadEvaluationComparisonTaskFiles
创建模型评测脚本	octopus	createModelEvaluationScript
更新模型评测脚本	octopus	updateModelEvaluationScript
删除模型评测脚本	octopus	deleteModelEvaluationScript
删除模型评测脚本文件	octopus	deleteModelEvaluationScriptFile

操作名称	资源类型	事件名称
保存模型评测脚本文件	octopus	saveModelEvaluationScriptFile
打包模型评测脚本文件	octopus	packageModelEvaluationScript
更新模型评估脚本打包结果	octopus	updateModelEvaluationScriptPackagingResult
上报打包模型评估脚本状态	octopus	updateModelUploadEvaluationScriptResult
下载已打包模型评估脚本	octopus	downloadModelEvaluationScriptPackagePath
创建模型评估任务	octopus	createModelEvaluationTask
重启模型评估任务	octopus	restartModelEvaluationTask
停止模型评估任务	octopus	stopModelEvaluationTask
继续模型评估任务	octopus	resumeModelEvaluationTask
删除模型评估任务	octopus	deleteModelEvaluationTasks
修改模型评测任务优先级	octopus	updateModelEvaluationTaskPriority
下载模型评测任务日志	octopus	downloadEvaluationTaskLog
保存坏例判别结果为数据集	octopus	saveBadcaseResultAsDatasets
创建训练任务	octopus	createTrainingTask
重启训练任务	octopus	restartTrainingTask
停止训练任务	octopus	stopTrainingTask
删除训练任务	octopus	deleteTrainingTasks
修改训练任务优先级	octopus	updateTrainingTaskPriority

表 12-6 云审计服务支持的仿真服务关键操作列表

操作名称	资源类型	事件名称
创建在线仿真加载任务	octopus	createSimLoadTasks
创建在线仿真保存任务	octopus	createSimSaveTasks

操作名称	资源类型	事件名称
创建在线仿真回放任务	octopus	createSimReplayTasks
下载仿真任务报告	octopus	downloadSimulationReports
创建算法镜像	octopus	createSimAlgorithmImages
更新算法镜像	octopus	updateSimAlgorithmImages
删除算法镜像	octopus	deleteSimAlgorithmImages
创建算法	octopus	createSimAlgorithms
更新算法	octopus	updateSimAlgorithms
删除算法	octopus	deleteSimAlgorithms
创建仿真配置	octopus	createSimBatchConfigs
更新仿真配置	octopus	updateSimBatchConfigs
删除仿真配置	octopus	deleteSimBatchConfigs
创建仿真任务	octopus	createSimBatches
更新仿真任务	octopus	updateSimBatches
删除仿真任务	octopus	deleteSimBatches
调整仿真任务队列优先级	octopus	updateSimBatchPriority
仿真pb下载	octopus	downloadSimOsi
算法日志下载	octopus	downloadAlgorithmLog
评测日志下载	octopus	downloadEvaluationLog
评测pb下载	octopus	downloadEvaluation
算法pb下载	octopus	downloadAlgorithmPb
仿真meta下载	octopus	downloadOsiMeta
算法meta下载	octopus	downloadAlgorithmMeta
创建评测镜像	octopus	createSimEvaluationImages
更新评测镜像	octopus	updateSimEvaluationImages
删除评测镜像	octopus	deleteSimEvaluationImages
创建评测	octopus	createSimEvaluations
更新评测	octopus	updateSimEvaluations
删除评测	octopus	deleteSimEvaluations
创建场景库分类	octopus	createSimCategories

操作名称	资源类型	事件名称
更新场景库分类	octopus	updateSimCategories
删除场景库分类	octopus	deleteSimCategories
创建逻辑场景库分类	octopus	createLogicalCategories
更新逻辑场景库分类	octopus	updateLogicalCategories
删除逻辑场景库分类	octopus	deleteLogicalCategories
创建场景库	octopus	createSimGroups
更新场景库	octopus	updateSimGroups
删除场景库	octopus	deleteSimGroups
场景库中添加场景	octopus	updateSimGroupsScenarios
场景库删除场景	octopus	deleteSimGroupsScenarios
创建逻辑场景库	octopus	createLogicalGroups
更新逻辑场景库	octopus	updateLogicalGroups
逻辑场景库中添加场景	octopus	updateLogicalGroupsScenarios
删除逻辑场景库	octopus	deleteLogicalGroups
删除逻辑场景库中的场景	octopus	deleteLogicalGroupsScenarios
创建场景地图	octopus	createSimMaps
更新场景地图文件	octopus	updateSimMaps
下载地图文件	octopus	downloadSimMaps
创建逻辑场景地图	octopus	createLogicalMaps
更新逻辑场景地图文件	octopus	updateLogicalMaps
创建场景3D模型	octopus	createSimModels
更新场景3D模型	octopus	updateSimModels
下载场景3D模型文件	octopus	downloadSimModels
创建测试套件	octopus	createSimSuits
更新测试套件	octopus	updateSimSuits
删除测试套件	octopus	deleteSimSuits

操作名称	资源类型	事件名称
套件添加用例	octopus	addCasesToSuits
从套件删除用例	octopus	deleteCasesFromSuit
创建测试用例	octopus	createSimTestcases
更新测试用例	octopus	updateSimTestcases
删除测试用例	octopus	deleteSimTestcases
测试用例添加单个标签	octopus	updateSimTestcasesLabels
测试用例删除单个标签	octopus	deleteSimTestcasesLabels
创建场景	octopus	createSimScenarios
更新场景	octopus	updateSimScenarios
删除场景	octopus	deleteSimScenarios
下载场景文件	octopus	downloadSimScenarios
创建场景文件	octopus	createSimScenariosFiles
修改场景文件	octopus	updateSimScenariosFiles
场景绑定单个标签	octopus	updateSimScenariosLabels
场景解绑标签	octopus	deleteSimScenariosLabels
创建逻辑场景	octopus	createLogicalScenarios
更新逻辑场景	octopus	updateLogicalScenarios
删除逻辑场景	octopus	deleteLogicalScenarios
下载逻辑场景文件	octopus	downloadLogicalScenarios
创建逻辑场景文件	octopus	createLogicalScenariosFiles
更新逻辑场景文件	octopus	updateLogicalScenariosFiles
创建标签	octopus	createSimLabels
更新标签	octopus	updateSimLabels
删除标签	octopus	deleteSimLabels
更新逻辑场景标签	octopus	updateLogicalScenariosLabels
删除逻辑场景标签	octopus	deleteLogicalScenariosLabels
创建参数泛化资源	octopus	createLogicalParameters
下载逻辑参数文件	octopus	downloadLogicalParameters

操作名称	资源类型	事件名称
更新参数泛化资源文件	octopus	updateLogicalParametersFiles
导入标签	octopus	createSimTurtles
创建泛化任务	octopus	createLogicalGeneralizations
删除泛化任务	octopus	deleteLogicalGeneralizations

表 12-7 云审计服务支持的智驾模型服务关键操作列表

操作名称	资源类型	事件名称
设置委托名称	octopus	setAgency
删除委托名称	octopus	deleteAgency
进行2D图片预标注	octopus	createImageAnnotation
创建2D图片生成任务	octopus	createImageGenJob
终止2D生成任务	octopus	terminateImageGenJob
删除2D生成图片任务	octopus	deleteImageGenJob
删除生成的2D图片	octopus	deleteGenImage
批量删除生成的图片	octopus	deleteGenImages
清理已删掉的图片	octopus	cleanGenImage

12.2 查看审计日志

在您开启了云审计服务后，系统会记录Octopus的相关操作，且控制台保存最近7天的操作记录。本节介绍如何在云审计服务管理控制台查看最近7天的操作记录。

操作步骤

步骤1 登录云审计服务管理控制台。

步骤2 在管理控制台左上角单击 图标，选择区域。

步骤3 在左侧导航栏中，单击“事件列表”，进入“事件列表”页面。

步骤4 事件列表支持通过筛选来查询对应的操作事件。当前事件列表支持四个维度的组合查询，详细信息如下：

- 事件来源、资源类型和筛选类型。

在下拉框中选择查询条件。

其中筛选类型选择事件名称时，还需选择某个具体的事件名称。

选择资源ID时，还需输入某个具体的资源ID。

选择资源名称时，还需选择或手动输入某个具体的资源名称。

- 操作用户：在下拉框中选择某一具体的操作用户，此操作用户指用户级别，而非租户级别。
- 事件级别：可选项为“所有事件级别”、“normal”、“warning”、“incident”，只可选择其中一项。
- 时间范围：可选择查询最近七天内任意时间段的操作事件。

步骤5 单击需要查看的事件名称，展开该事件的事件概览，可查看事件的基本信息，单击“了解事件结构”，可以查看该事件结构的详细信息。

----结束

