DDLD产线：Data Driven Landmark Detection

ICC feature迭代 代转区的评测head

数据湖的使用

数据集：

labeled\_s\_minicell\_issue\_fix\_xushi\_241203\_v5

任务：构建评测集，测试虚实属性

评测集构建参考MCT，测试参考端到端生产流程

{'\_id':'labeled\_s\_minicell\_issue\_fix\_xushi\_241203\_v5'}能够看到处理前的GT数据

处理后可用的评测集如下表示：

处理后的数据分为bag和对于frames存储，如：

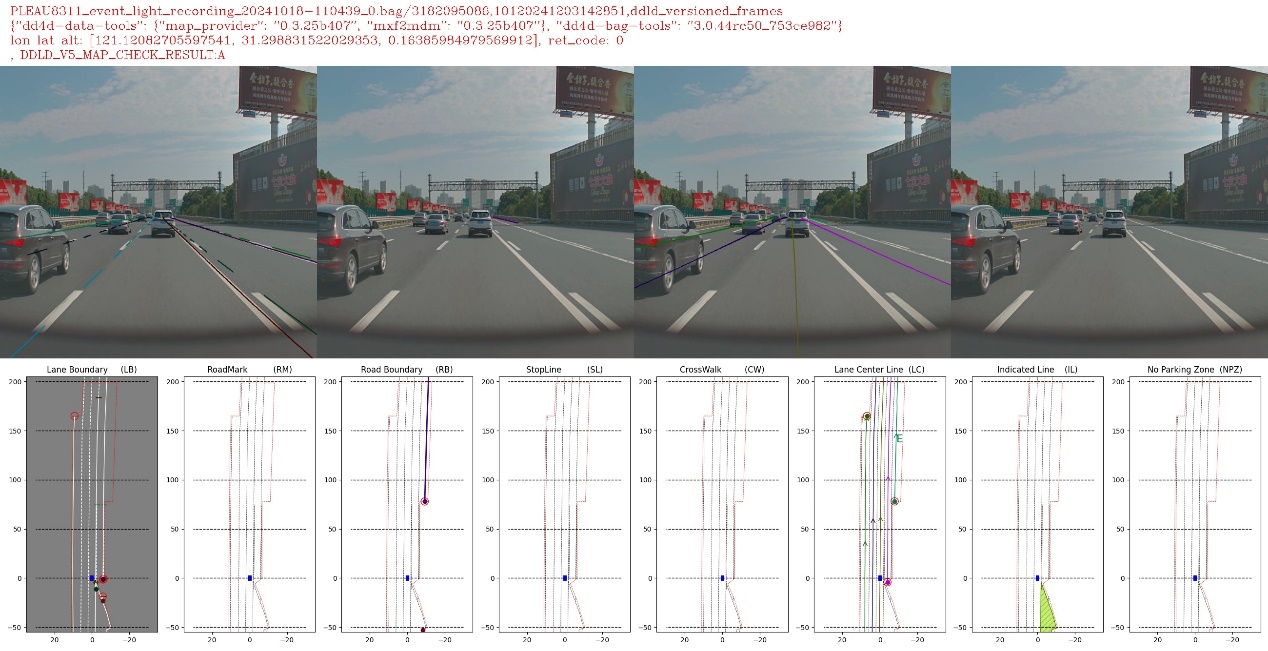
Ddld\_versioned\_bags

{'bag\_name':'PLEAU8311\_event\_light\_recording\_20241018-110439\_0.bag', 'version':'10120241203142851'}

Ddld\_versioned\_frames

{'bag\_name':'PLEAU8311\_event\_light\_recording\_20241018-110439\_0.bag', 'version':'10120241203142851'}

在frames中可以可视化：



LB lane boundary 车道线边界

RoadMark RM 路标

Road Boundary RB 道路边界

StopLine SL

CrossWalk CW

Lane Center Line LC车道中心线（没有实线，需要靠两个车道线界定）

Indicated Line IL 指示线（待转区，道路合并等等）

No Parking Zone NPZ 禁停区

可以看到搜索的命令和文件都是同一个

Region Of Vsibility ROV 相机透视后的真正可见范围

使用builid\_gt下的parse\_pep\_gt\_mct.py,以及pep\_gt\_cfgs

任务1：将cfg的输入中的dataset name，bag name改为显式输入

任务2：将pipeline中的参数和config.py中的参数对齐

任务3：将parse\_pep\_gt\_mct.py和caculate\_rov.py的处理流程集成到一个文件

Bev-lanDet,PersFormer,Geometry-guided Kernel Transformer(GKT)

刚申请或者重启的debug机器需要source ~/.bash\_debug 主要是改变默认环境python的位置

|  |
| --- |
| Parse\_pep\_gt\_mct.py中DdldPepGtParser的作用： |

Onnx模型文件的特点：

|  |
| --- |
| **1. FileSize（文件大小）**   * **含义**：指保存ONNX模型文件的物理存储空间（如磁盘占用），通常以MB/GB为单位。 * **决定因素**：   + 参数数量（如神经网络权重数量）。   + 参数精度（如FP32占4字节/参数，FP16占2字节/参数）。   + 模型结构复杂度（层数、连接方式等）。 * **示例**：ResNet-50的ONNX文件约100MB，而GPT-3可能达数百GB。 * **为什么重要**：影响存储、传输效率，尤其在嵌入式设备中需考虑存储限制。   **2. Memory（内存占用）**   * **含义**：模型运行时占用的内存（RAM），包括：   + **权重内存**：加载模型参数所需空间。   + **激活内存**：前向传播时中间计算结果（如特征图）的临时存储。 * **决定因素**：   + 参数量（权重内存：参数量×数据类型字节数）。   + 输入尺寸（激活内存：与输入分辨率、网络深度正相关）。 * **示例**：输入一张1080p图像，某些模型可能需1GB以上内存。 * **为什么重要**：设备内存不足会导致崩溃或降速（如手机端部署需轻量化）。   **边缘设备没有VRAM（GPU），只有CPU Memory**  **3. Complexity（复杂度）**   * **广义分类**：   + **计算复杂度**：通常用FLOPs衡量（见下文）。   + **空间复杂度**：与内存占用相关。   + **结构复杂度**：模型深度、宽度、连接复杂度（如残差连接）。 * **影响**：复杂度高的模型通常能力更强，但资源消耗更大，可能过拟合。 * **为什么重要**：需在性能（如准确率）与资源消耗间权衡。   **4. FLOPs（浮点运算次数）**   * **含义**：执行一次模型推理所需的浮点运算（加减乘除）总数，衡量计算量。 * **决定因素**：   + 卷积层：FLOPs = 输入高×输入宽×输入通道×输出通道×卷积核高×核宽。   + 全连接层：FLOPs = 输入维度×输出维度。 * **示例**：ResNet-50约3.8 GFLOPs（十亿次运算），GPT-3达数千万亿次。 * **为什么重要**：FLOPs越高，对算力要求越高，推理速度越慢（尤其在CPU/边缘设备）。 |

模型前向运行流程

|  |
| --- |
| 感知模型评测流程   1. 使用build\_gt从原始gt数据中获得可供mct使用的gt数据集，例如：   labeled\_s\_minicell\_issue\_fix\_xushi\_241203\_v5 -> wzh\_2025\_01\_16\_test\_dataset  具体可以参考：  /home/zhaohui1.wang/dd4d-workspace/build\_gt/pep\_gt\_cfgs/pep\_gt\_cfg\_ebm\_320.py  其中wzh\_2025\_01\_16\_test\_dataset为上传到数据湖的处理后GT文件   1. 使用模型inference对原始input数据进行inference获得dt数据，例如：   labeled\_s\_minicell\_issue\_fix\_xushi\_241203\_v5->pkl文件  具体可以参考：/home/zhaohui1.wang/mal/projects/ddod/examples/e2e\_4.x\_finetune\_ld\_random\_init\_bev/config\_bev\_seg\_long.py   1. 使用gt数据集和dtpkl文件进行MCT评测 |

EBM：End2end Big Model智驾大模型

|  |  |
| --- | --- |
| 2/24 | 1.完成宽窄数据UNKONWN\_WIDTH到NARROW的切换，实现效果说明如下：  原数据labeled\_s\_minicell\_issue\_fix\_xushi\_241203\_v5中所属PLEAU8311\_event\_light\_recording\_20241018-110439\_0.bag    原数据为UNKNOWN\_WIDTH  labeled\_s\_minicell\_issue\_fix\_xushi\_241203\_v5\_wzh\_remake中所属PLEAU8311\_event\_light\_recording\_20241018-110439\_0.bag  2  复制处理后的数据已经是NARROW,检查是否  2.model进行inference  Tag出现问题，query会把数据集滤除，猜想是复制数据集没有tag数据  Ddld\_train\_dataset\_samples中tag为空    修改方法：在samples读取函数中添加得到tags的读取方法，保留原数据集的tags    3．完成train\_data\_ddld\_issue\_fix\_data数据的批量处理 |

|  |
| --- |
| Waiting Area  torch.backends.cudnn.benchmark的使用要求：输入大小不发生变化  MMCV的使用方法  [MMCV 核心组件分析(五): Registry - 知乎](https://zhuanlan.zhihu.com/p/355271993)  Registry-@register\_module()-build\_from\_cfg 3个基本步骤  Dataset/Pv\_dataset  模型训练部分：    数据集读取全是在数据湖datalake上获取  模型结构方面：  Model第一个encoder采用MtMcBevAggByCurrentKV  Mult-Task-Multi-Camera    对应[B,T,N,4,4]  Encoder重要模块：  有编码，kv生成  使用transformer生成bev feature    Transformer挺多  最终分割的head在BevSegHead中，生成loss    仅在bev空间进行seg，没有深度高度信息， |