Project 2

Prediction

实现了 CV 预测器

```python
def constant_velocity(self, object: TrackedObject) -> PredictedTrajectory:
    cv_probability = 1.0
    cv_waypoints = []
    if isinstance(object, Agent):
      for time in np.arange(self._ego_state.time_seconds, self._ego_state.time_seconds + self._duration, self._sample_time):
          x = object.center.x + time * object.velocity.magnitude() * np.cos(object.center.heading)
          y = object.center.y + time * object.velocity.magnitude() * np.sin(object.center.heading)
          cv_waypoints.append(Waypoint(time_point=time*1e6,
                                       oriented_box=OrientedBox.from_new_pose(object.box, StateSE2(x, y, object.center.heading)),
                                       velocity=object.velocity))
    return PredictedTrajectory(waypoints=cv_waypoints, probability=cv_probability)
  You, 3周前 • Implement cv predictor …
```

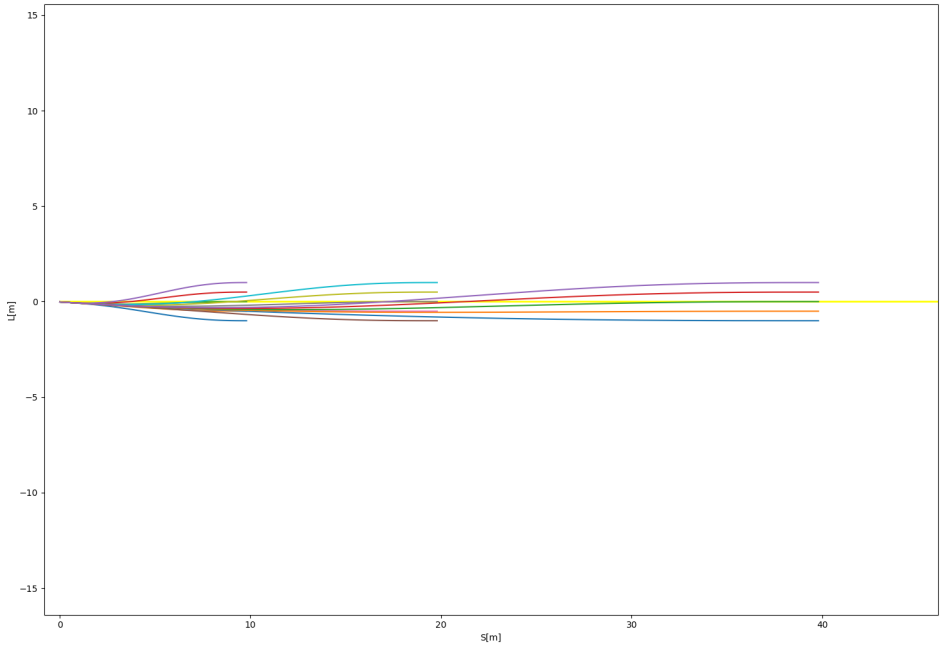
Path Planning

本项目的 Path planning 部分采用了 lattice planner，分为横向采样和纵向采样两部分。

1. 在横向上

- 横向的局部位姿采样：沿着 reference line 在 d 方向上间隔 0.5m 采样左 1m，右 1m: [-1.0m，1.0m，0.5m]，在 s 方向上采样 [10m，20m，40m]
- 横向的 path 采样：已知起点和终点（采样得到）位姿（s, l, l', l''），共有 6 个条件。可以利用五次多项式拟合这段 path(s, l)：

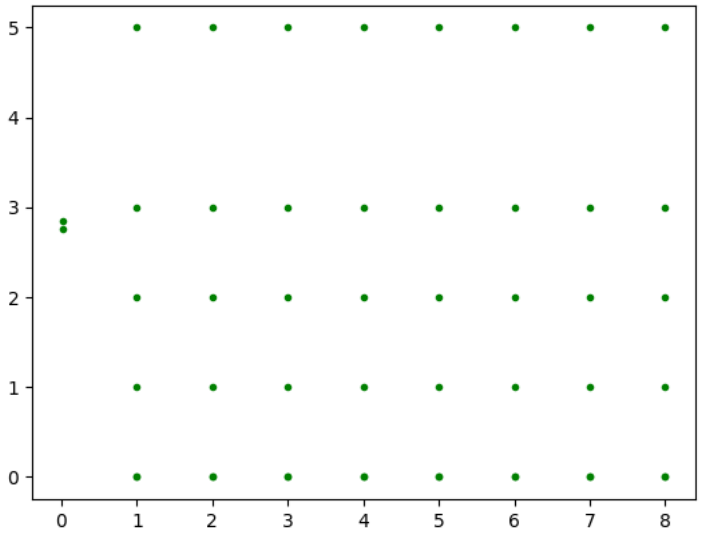
$$L(s)= a_0 + a_1 * s + a_2 * s^2 + a_3 * s^3 + a_4 * s^4 + a_5 * s^5$$

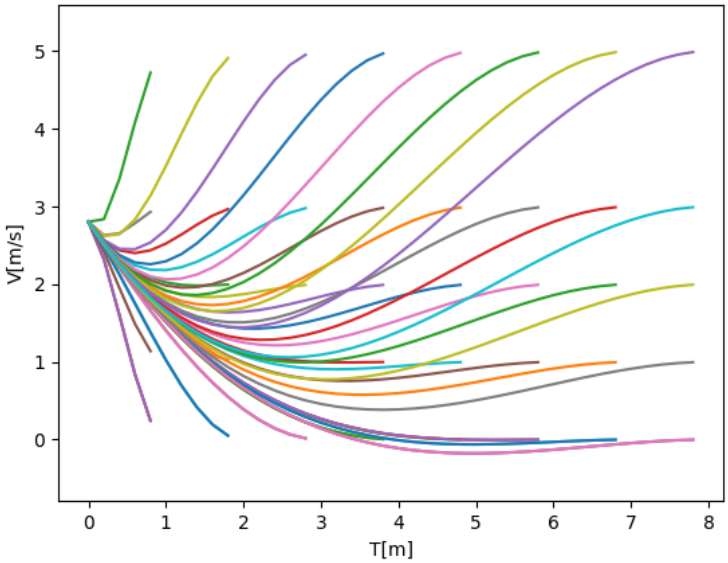


2. 在纵向上

- 纵向的速度采样：在 horizon time 范围内按照 1s 的间隔段，从当前位置的速度以 max_acc 和 max_dec 依次求出最大最小速度，然后按照 1m/s 的间隔采样得到每个时间对应的所有速度。
- 纵向的轨迹采样：已知起点（s, v, a）和终点（v, a）共 5 个条件，可以利用 4 次多项式拟合这段 speed profile：

  $S(t) = a_0 + a_1 * t + a_2 * t^2 + a_3 * t^3 + a_4 * t^4 + a_5 * t^5$





然后通过一一组合横向 path 和纵向 speed profile，通过评价函数求出最优的组合，评价函数设计了：平滑性（横纵向曲线的 3 阶导数 jerk），偏移参考线（L 的 value）和纵向的 progress（S 的 value）三种，远离障碍物作为 TODO。

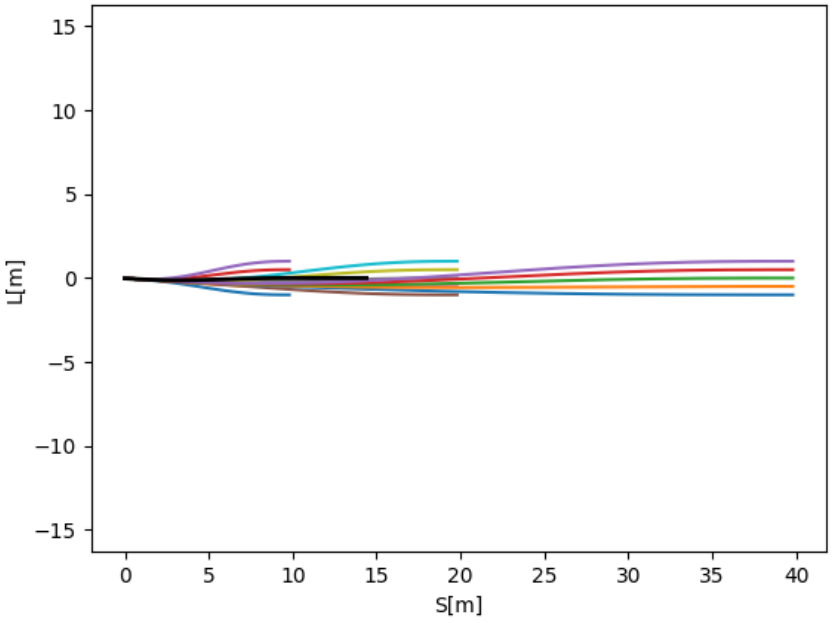
最后将选出的横向 path 和纵向 speed profile 按照 0.25s 的时间间隔进行合并得到最终的轨迹，具体方法是由 speed profile 求出对应时间下的 current_s，将 current_s 带入 L 方程得到 current_l。

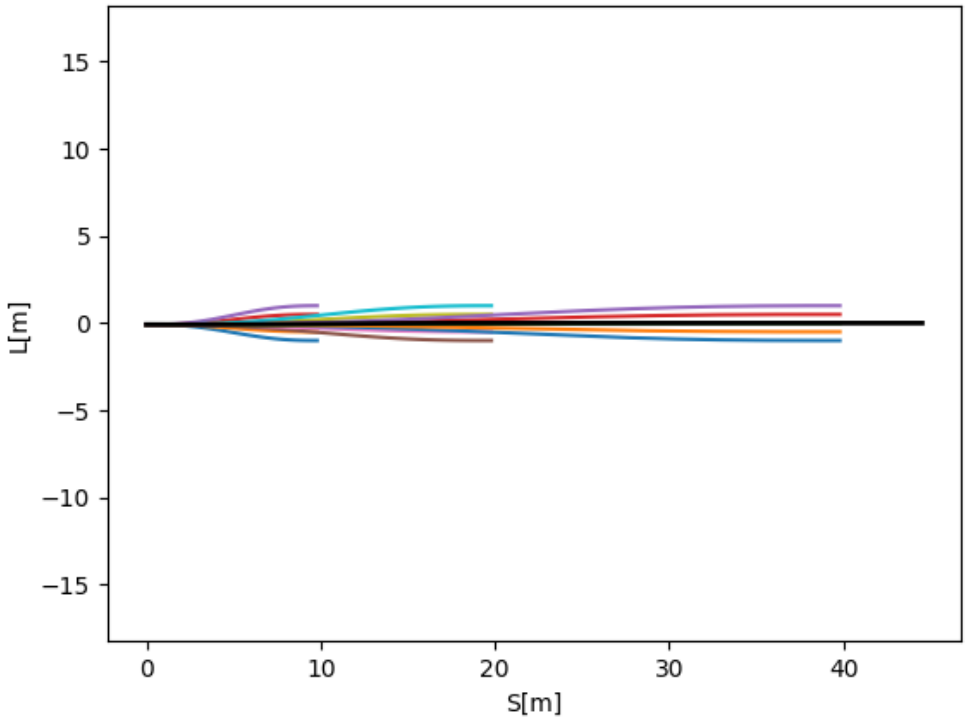
Speed Planning

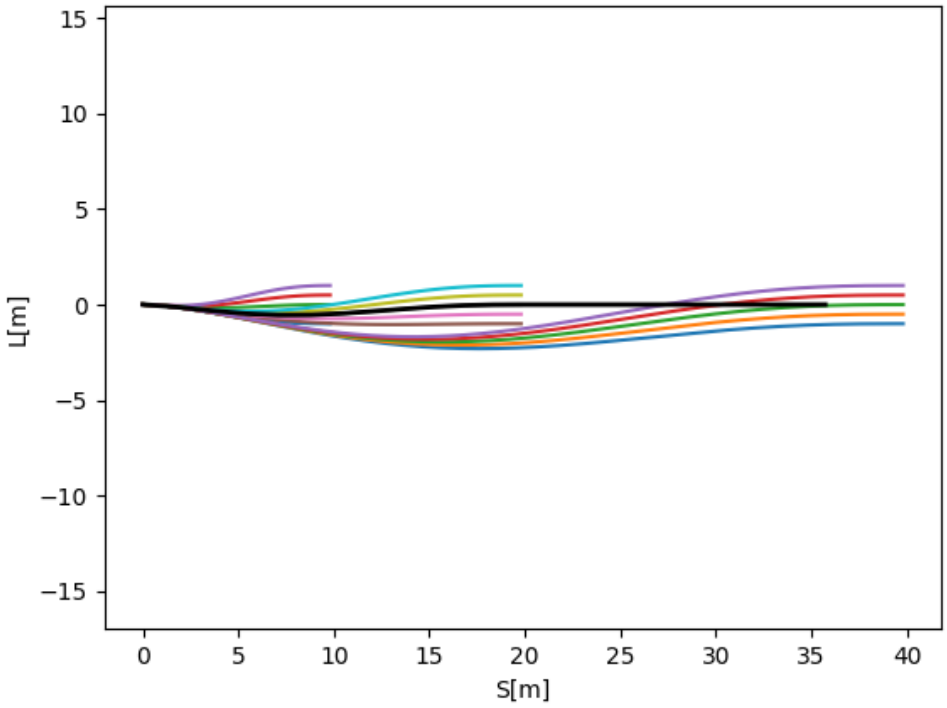
- 实现了一个匀速运动模型
- 尝试了 DpDecider，可以正常运行，但是求出的速度永远是 0，不知原因

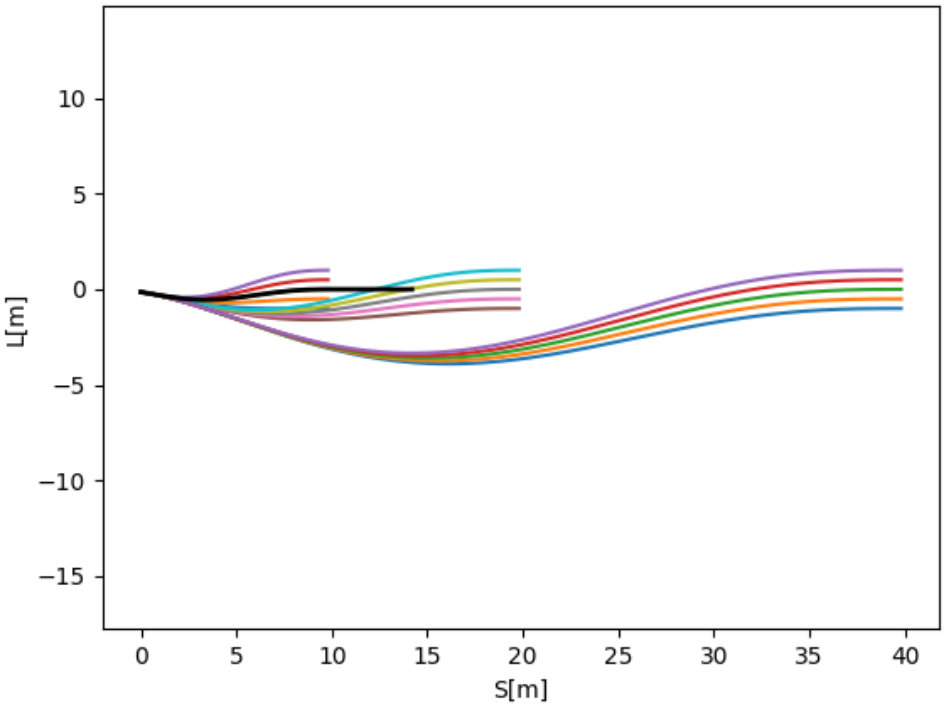
Question

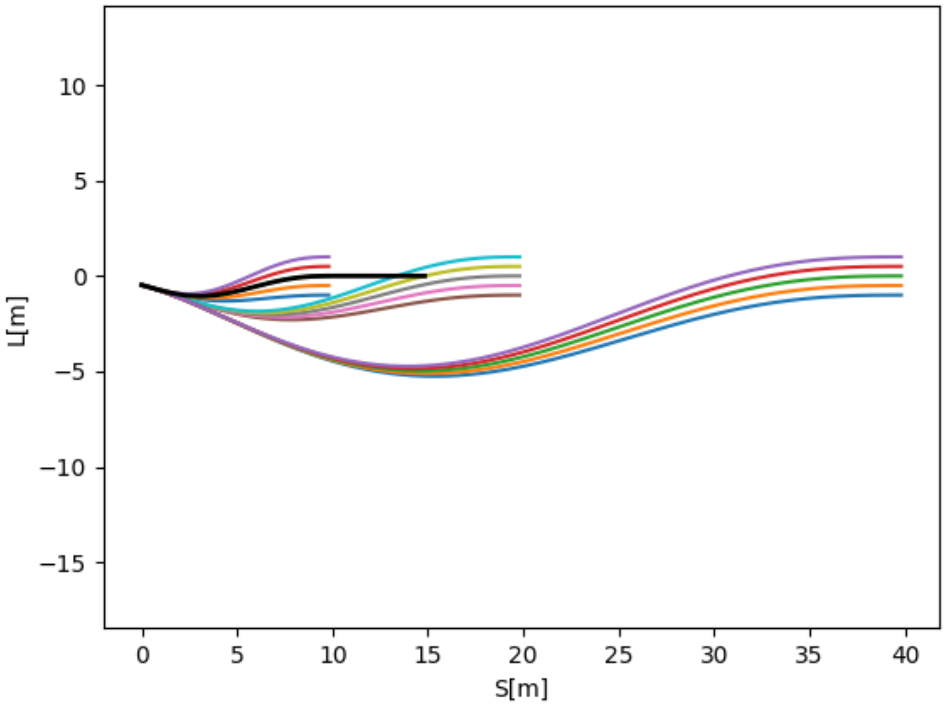
1. 在跑了几个场景之后，发现在直线场景中本算法可以较好的运行，但在遇到弯道时生成的 path 逐渐偏离参考线，如下图所示是连续几帧的采样结果，黑色为最优 path。可见在前三个图中最优 path 是较为贴合 reference line 的，之后起始位置越来越向下，采样 path 也越来越偏离 reference line，这个问题不知道是什么原因，请助教帮忙解答一下~

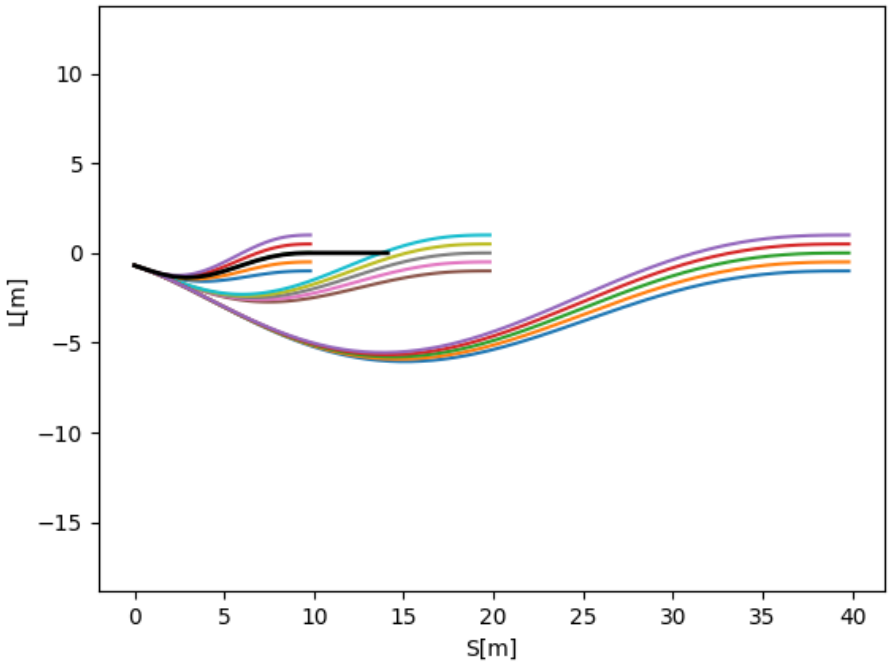


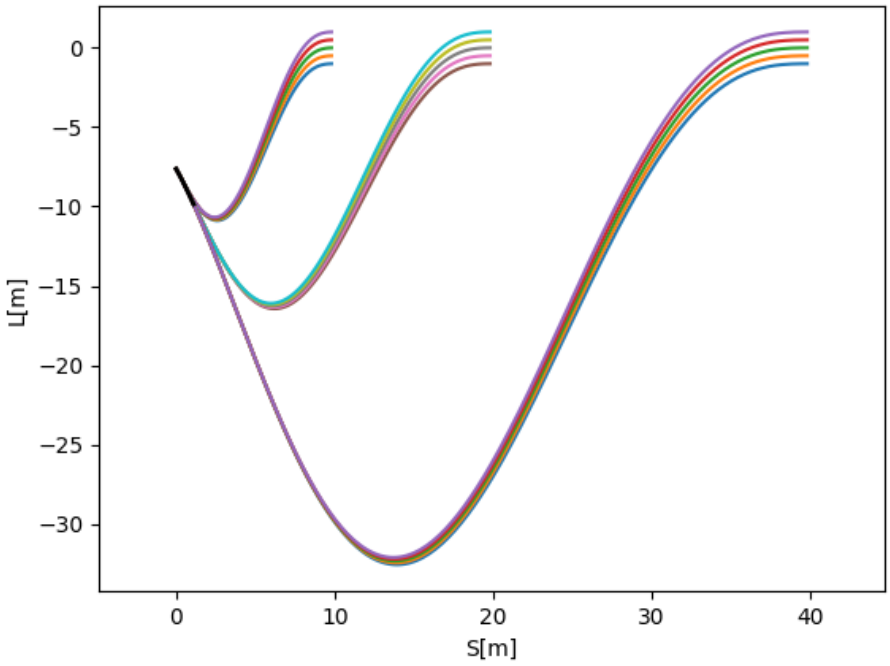

15
10
5
0
−5
−10
−15
L[m]
0
10
20
30
40
S[m]


15
10
5
0
−5
−10
−15
L[m]
0
5
10
15
20
25
30
35
40
S[m]


10
5
0
−5
−10
−15
L[m]
0
5
10
15
20
25
30
35
40
S[m]


10
5
0
−5
−10
−15
L[m]
0
5
10
15
20
25
30
35
40
S[m]


10
5
0
−5
−10
−15
L[m]
0
5
10
15
20
25
30
35
40
S[m]


0
−5
−10
−15
−20
−25
−30
L[m]
0
10
20
30
40
S[m]

2. Lattice planner 合并横纵向曲线后得到了一条轨迹，但是由于下游有 speed planning 模块，其中只使用了轨迹中的几何信息，因此想到是否可以只采样横向 path，通过离散 s 得到 l, dl, ddl, s 序列？以及这个方案与横纵向一起采样的方案适合什么场景使用？

```
# get the optimal trajectory in frenet
l = []
dl = []
ddl = []
s = []
# 方法1: 根据间隔时间，由纵向轨迹得到s，再由s得到l
# for t in np.arange(0, self.horizon_time.time_s, self.sampling_time.time_s):
#     if (t > optimal_lon_trajectory.get_time()):
#         break
#     # 由速度曲线得到 s，再由 s 得到 l.
#     sampled_s = optimal_lon_trajectory.get_point(t)
#     l.append(optimal_lat_trajectory.get_point(sampled_s))
#     dl.append(optimal_lat_trajectory.get_first_derivative(sampled_s))
#     ddl.append(optimal_lat_trajectory.get_second_derivative(sampled_s))
#     s.append(sampled_s)
# 方法2: 离散 s，直接由横向轨迹得到
for sampled_s in np.arange(0, optimal_lat_trajectory.get_param(), 0.2):
    # 由速度曲线得到 s，再由 s 得到 l.
    l.append(optimal_lat_trajectory.get_point(sampled_s))
    dl.append(optimal_lat_trajectory.get_first_derivative(sampled_s))
    ddl.append(optimal_lat_trajectory.get_second_derivative(sampled_s))
    s.append(sampled_s)
plt.plot(s, l, 'r-')
return l, dl, ddl, s      You, 5天前 • version 1.0 …
```