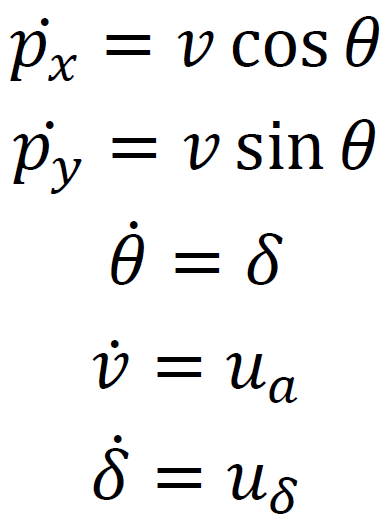
Final Project

Automatic Parking

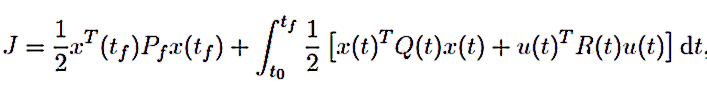
Jiahe Xu(jxu109), Anxun Zhang(azhang49)

1. Model description

A simple car model with dynamics defined with the below parameters is used in this case. Where (px, py) is the planar position of the car, v is the forward velocity, θ is the car’s angle concerning the px-axis, and δ is the steering angle.



The standard quadratic cost function is



with

For the simulation setting: red squares are obstacles, and the car (black circle) should reach the goal (blue start) in the picture. Small circles represent red squares; the path is two times as wide as the car’s diameter.

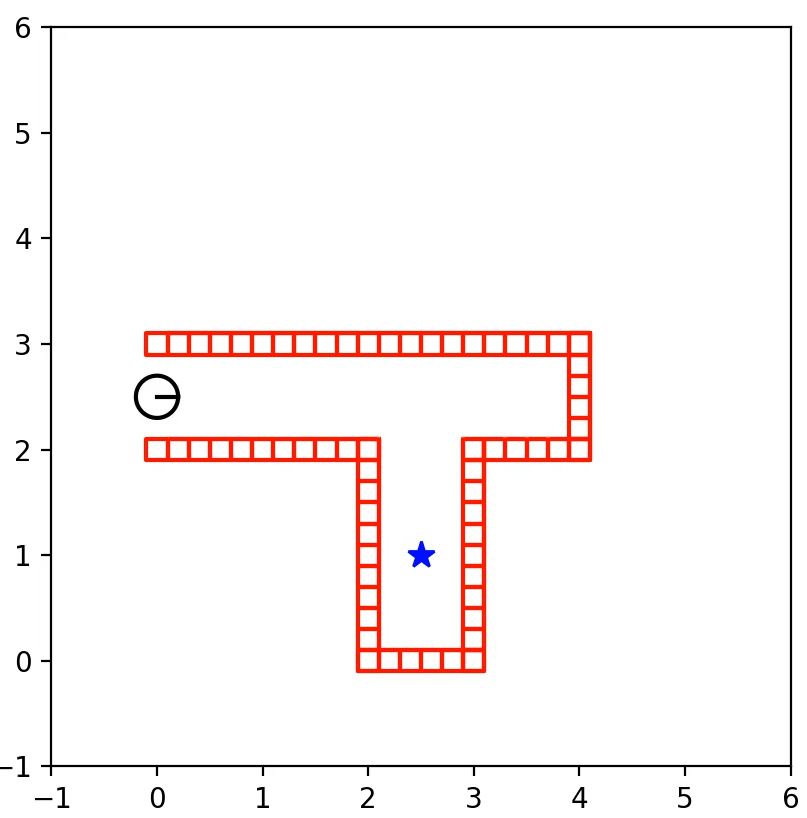


Figure 1: obstacle set

To better simulate the real-world situation, we use ***Runge-Kutta*** method to calculate the **actual position** of our car in the environment. We only use the Euler First-order approximation to calculate the position in the planning procedure. **This forces us to use MPC’s frame-like methods to solve each step**. Except for MPPI methods, DDP iLQR and CEM methods use an MPC frame to produce a control sequence and execute the first control pair.

1. Control Method

We used gradient-based methods (DDP and iLQR) and sample-based methods to do the test. However, since most obstacles are tangent to nearby ones, there will be **many local minimal points, gradient-based methods never work**. As a result, we will add waypoints to guide the car.

1. Results
2. Successful result for our problem situation:

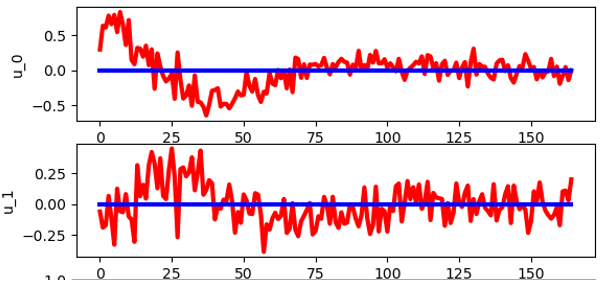
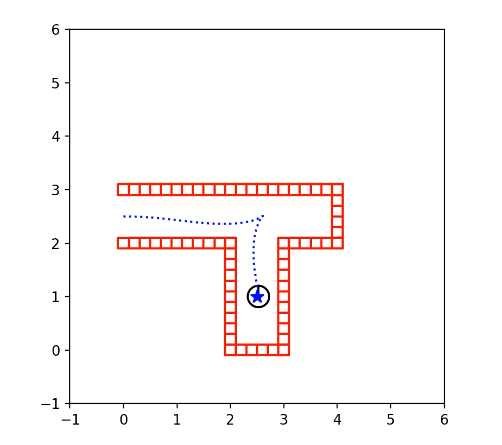
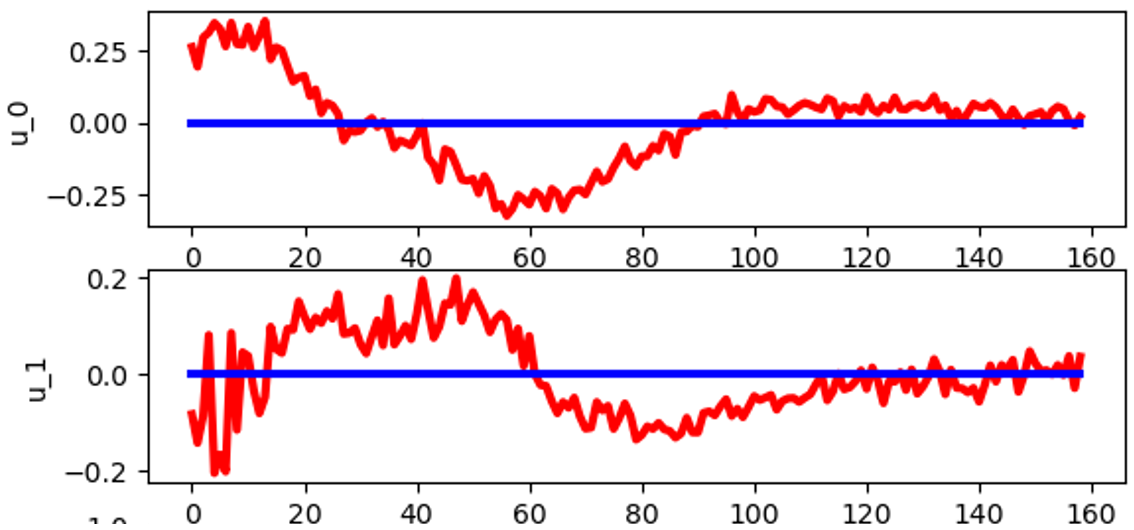
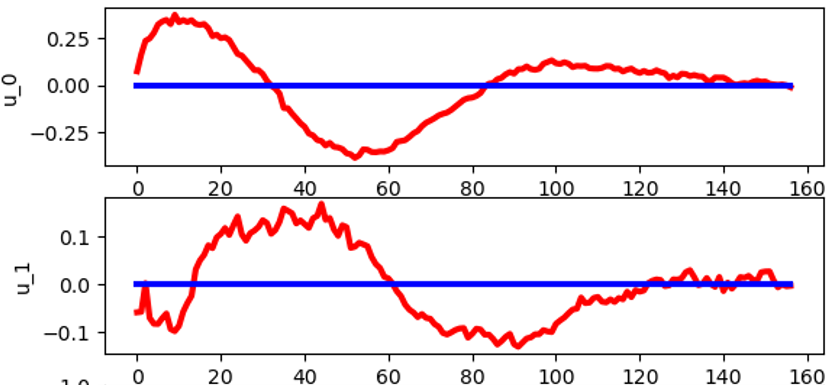
 

Figure 2: control of CEM (horizon size:50) Figure 3: the trajectory of sample-based methods

 Figure 4: control of MPPI-William method (horizon size:50) Figure 5: control of MPPI method (horizon size:50)

|  |  |  |
| --- | --- | --- |
| Methods | Time per-iteration (horizon size:50) | Total Cost (horizon size:50) |
| CEM | ms |  |
| MPPI | 43.9.5ms |  |
| MPPI Williams | 44.00.4ms |  |
| Methods | Time per-iteration (horizon size:20) | Total Cost (horizon size:20) |
| CEM | 21.1ms |  |
| MPPI | 18.2ms |  |
| MPPI Williams | 18.0ms |  |

Table1 sample-based methods results

We take 500 sample trajectories in each iteration (more than 500 samples doesn’t improve the performance too much), forum points make horizons more signific than 50, the result doesn’t change much. For horizons smaller than 20, the car might never reach the goal position. The initial variance of noise in control is always 0.2. All the sample-based methods have similar results when the horizon is long enough. The CEM method has a more extended time cost on each iteration because it needs to sort the costs and relatively worse performance on the total cost when the horizon is short since it doesn’t combine information-related weights. MPPI and MPPI-William’s[1] outcomes are expected to be similar since they have a different way of measuring the cost of trajectories. Overall, the result is as expected. **We didn’t use the result of sample-based methods to be the initial guess of gradient-based methods since the running time is too slow. The whole project was implemented in python; even with an excellent initial guess, the mechanism of python makes it inevitable to use three “for” loops when considering obstacles (3087.45 ms for each step with a horizon with 20 steps), and Matlab has similar results.**

2)**Manually add waypoints**

To make gradient-based methods work, we added waypoints. In our case, adding waypoints is easy, but in general it could be tricky since the path should be obstacle-free. The car will follow a series of waypoints, and the cost function uses the current goal to calculate the state cost in each time interval.

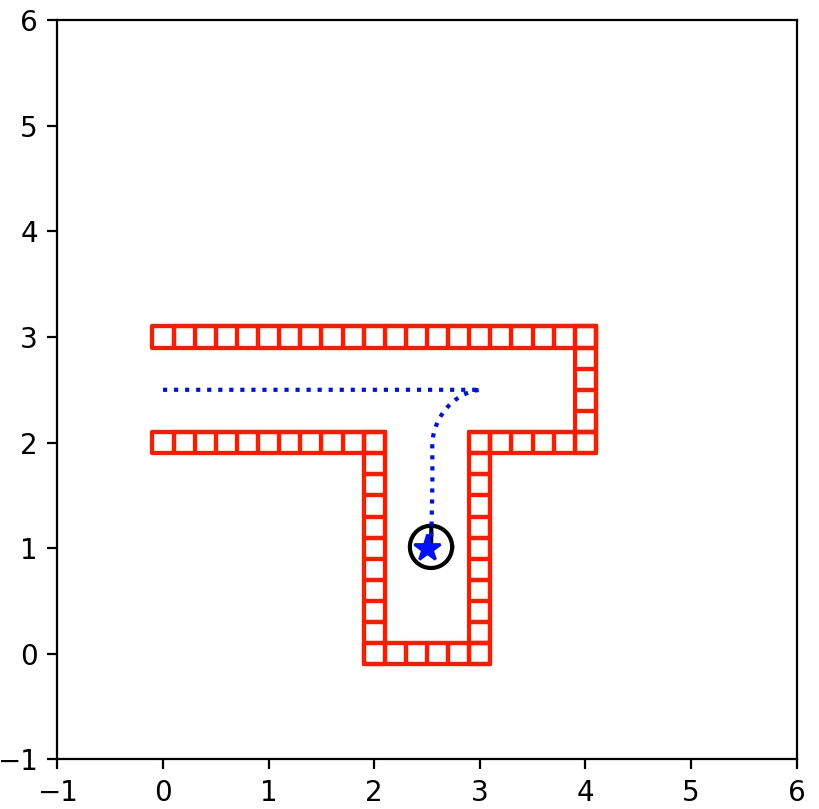
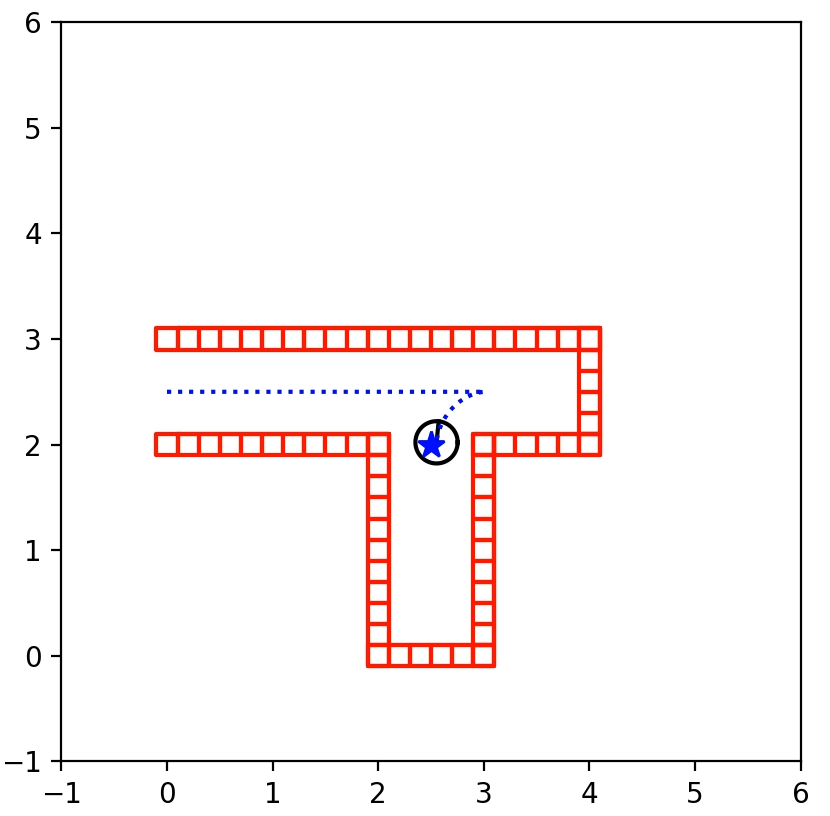
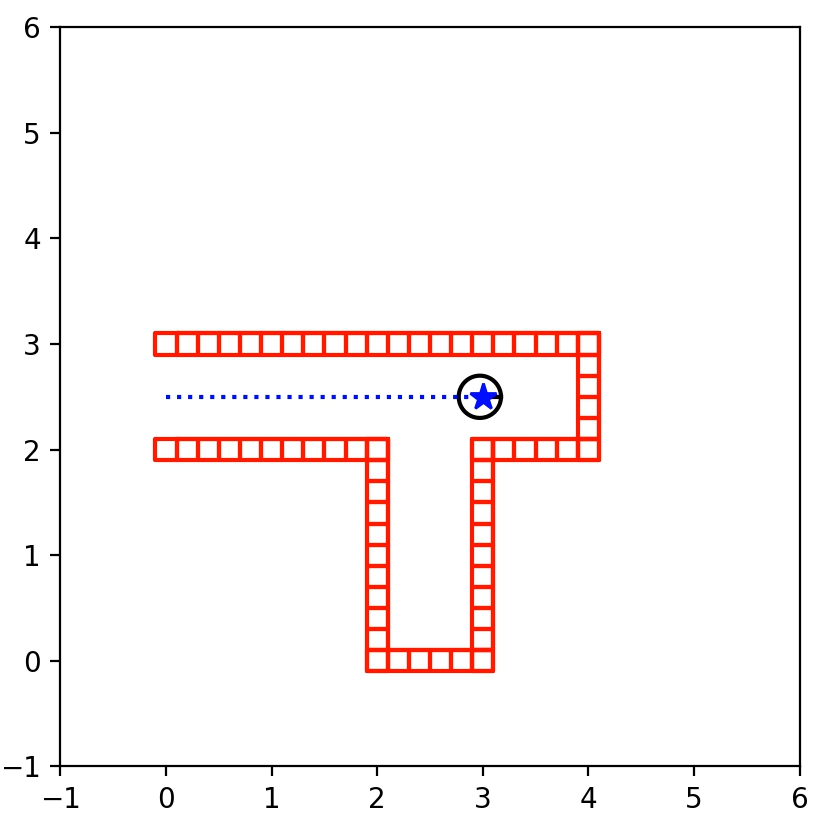


Figure 6: the trajectory with waypoints involved

|  |  |  |
| --- | --- | --- |
| Methods | Time per-iteration (horizon size:50) | Total Cost (horizon size:50) |
| CEM | 13.3ms |  |
| MPPI | 8.50.5ms |  |
| MPPI Williams | 7.50.6ms |  |
| DDP | 19ms | 18.44 |
| iLQR | 11ms | 18.44 |

|  |  |  |
| --- | --- | --- |
| Methods | Time per-iteration (horizon size:20) | Total Cost (horizon size:20) |
| CEM | 5.3ms |  |
| MPPI | 2.80.1ms |  |
| MPPI Williams | 7.50.1ms | 0.5 |
| DDP | ms | 20.07 |
| iLQR | ms | 20.07 |

Table2 all methods’ results

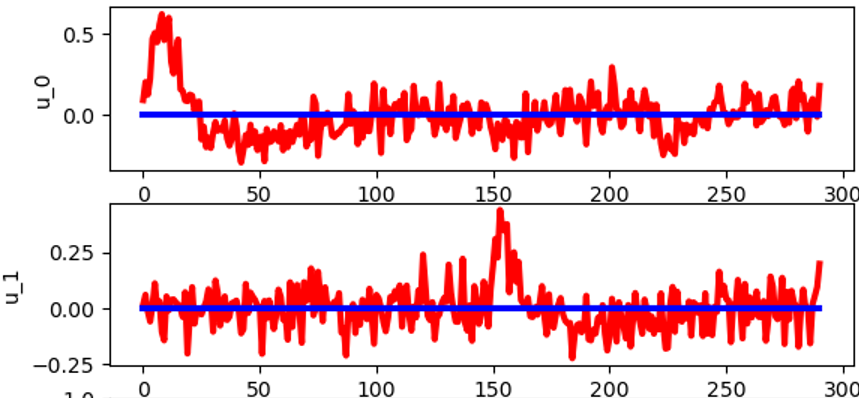
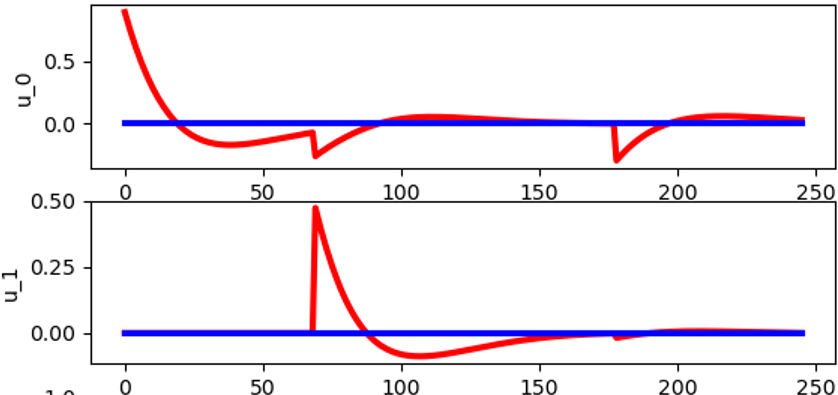


Figure7: control of DDP and iLQR (horizon size:50) Figure8: control of CEM(horizon size:50)

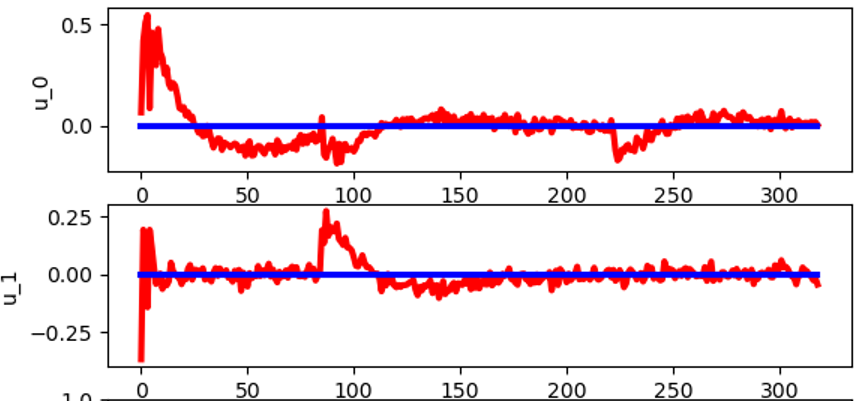
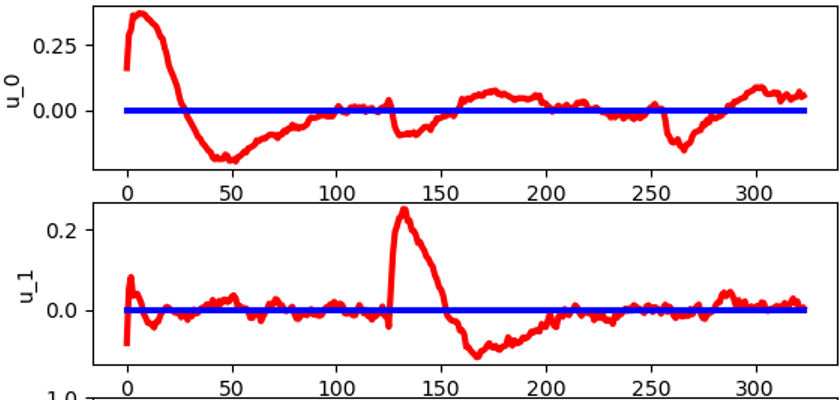


Figure9: control of MPPI (horizon size:50) Figure10: control of MPPI-William(horizon size:50)

The setting of sample based-methods is the same as the no-waypoints method. From Table2 we can see that when it is obstacle-free, gradient-based methods have better results than sample-based methods, but sample-based methods are faster. In two different horizons, DDP and iLQR have the same results (so we just used one figure), in general iLQR is faster, which is as expected. From the plots, sample-based methods are close to the result of gradient-based methods, this means our result is reliable.

1. Reference

[1] Williams, G., Wagener, N., Goldfain, B., Drews, P., Rehg, J. M., Boots, B., & Theodorou, E. A. (2017, May). Information theoretic MPC for model-based reinforcement learning. In 2017 IEEE International Conference on Robotics and Automation (ICRA) (pp. 1714-1721). IEEE.