Modified Dynamic Programming Algorithms for Vehicles Moving in Lane-Free Environments

# Discrete Differential Dynamic Programming (DDDP)

## Problem Formulation

The discrete-time vehicle dynamics are described as

where are the state variables, corresponding to the longitudinal and lateral positions and the longitudinal speed, respectively. The control variables reflect on the longitudinal acceleration and the lateral speed, while is the time step.

State and control variables are bounded within feasible region as follows:

The objective function to be minimized is described in (6). Note that collision with other vehicles/obstacles in not considered within the objective function, however any unsafe position is considered as infeasible area.

The corresponding recursive Bellman equation for reads

with boundary condition .

## Numerical Solution

### Discretization

With given control discretization intervals and , the state discretization intervals are as follows

This is quite abstract. Probably needs to change with something more sophisticated.

### State and Control Domains

State and control domains are considered as follows

where

# Differential Dynamic Programming (DDP)

## Problem Formulation

The discrete-time vehicle dynamics are described as

where are the state variables, corresponding to the longitudinal and lateral positions and speeds, respectively. The control variables reflect on the longitudinal and lateral accelerations, while is the time step.

State-dependent and constant bounds on control variables are applied as follows:

where

The objective function to be minimized reads

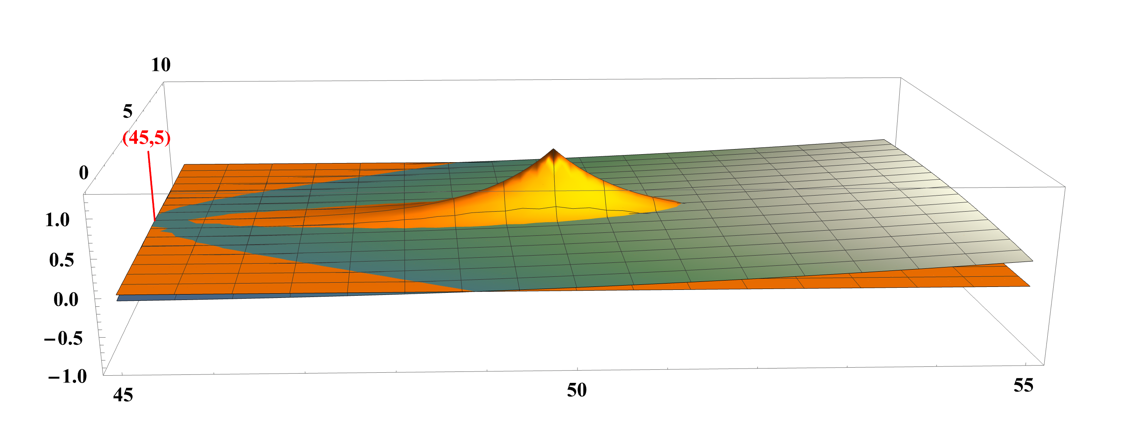
where is the collision avoidance term for each obstacle , defined as

In the current initial tests, the collision avoidance term does not consider any time-gap logic, however this can be modified easily.

Subsequently, the corresponding recursive Bellman equation for reads

with boundary condition .

At each time-step of each iteration, DDP requires a quadratic approximation of the term to be minimized in the recursive Bellman equation. To this end, a second order Taylor approximation is applied around nominal trajectories . Moreover, the state equations and the constraints are approximated using first order Taylor approximation around the same nominal trajectories .



A graph of a graph showing a point

Description automatically generated with medium confidence

A green and orange graph

Description automatically generated with medium confidence

A graph of a graph

Description automatically generated

A graph of a mountain

Description automatically generated

Figure 2nd Order Taylor Approximation of collision avoidance term (green surface) versus the original function (orange surface). The red vertical line and text correspond to the nominal point for the approximation. The center point of the obstacle is located at ()=(50,5).

# Results

## Scenario

Scenario setup

|  |  |
| --- | --- |
| Time Horizon | 25 s |
| Time Step () | 0.25 s |
| Longitudinal Desired speed () | 30 m/s |
| Initial Speed () | 25.0 m/s |
| Obstacle Speeds | 25.0 m/s |
| Weights | [1.0, 1.0, 0.5, 0.1, 20.0] |

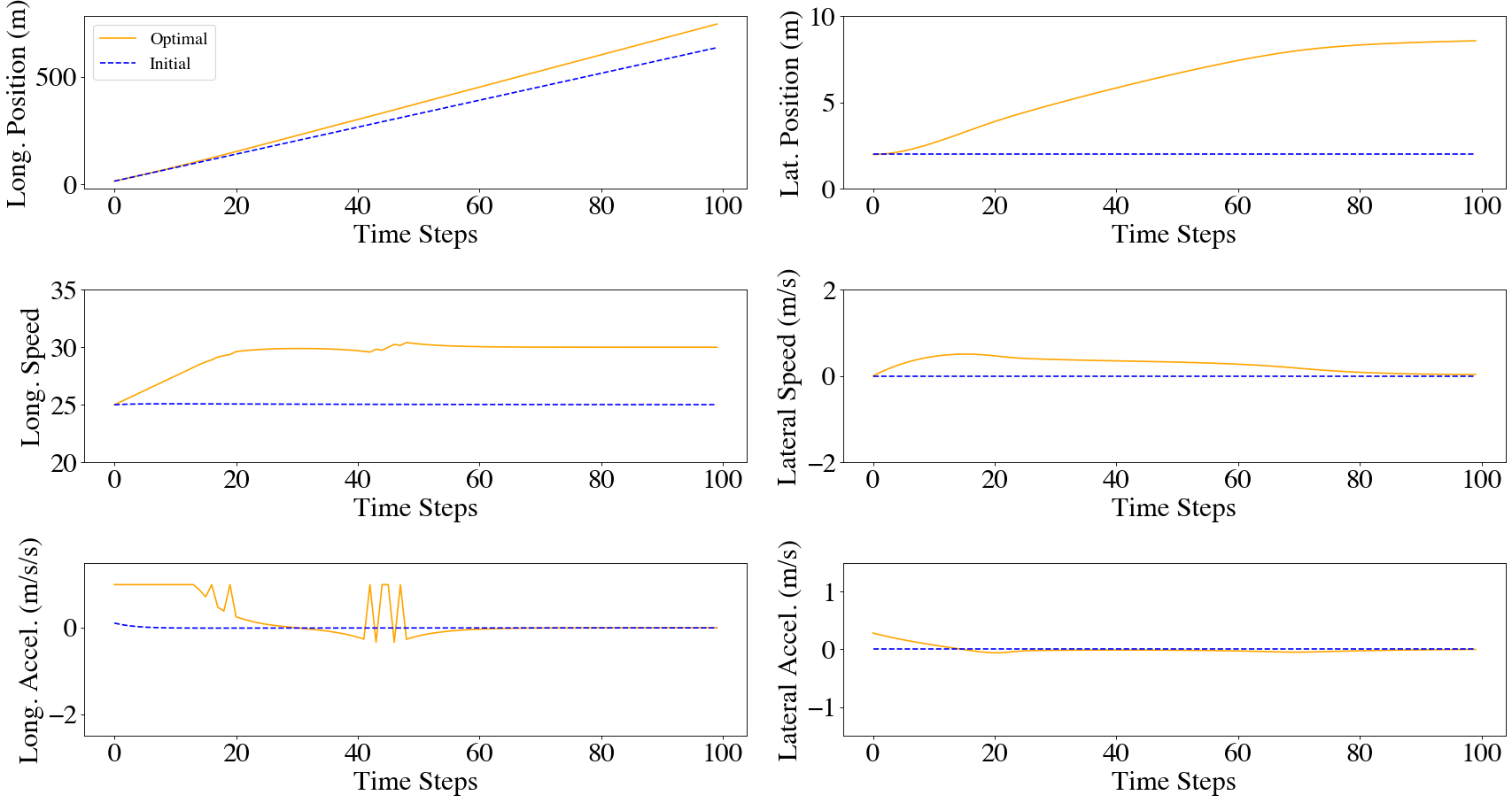
A yellow and black rectangle with a red and black stripe

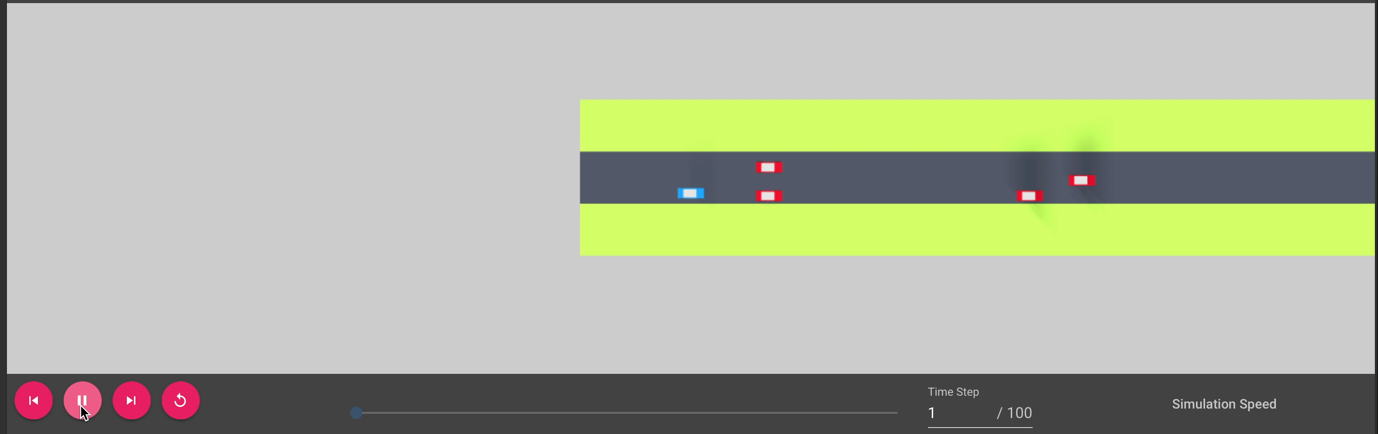
Description automatically generated

Figure 2 Scenario representation at initial state

## DDP

### Optimal trajectories using as initial trajectory the **IDM**





### Optimal trajectories using as initial trajectory the **Feedback** **controller**

A group of graphs showing different steps

Description automatically generated

A screenshot of a video

Description automatically generated

### Optimal trajectories using as initial trajectory the optimal solution of **DDDP with and**

A group of graphs showing different steps

Description automatically generated

A screenshot of a video

Description automatically generated

### Optimal trajectories using as initial trajectory the optimal solution of **DDDP with and**

A group of graphs showing different steps

Description automatically generated

A screenshot of a video

Description automatically generated

### Optimal trajectories using as initial trajectory the optimal solution of **DDDP with and**

A group of graphs showing different steps

Description automatically generated

A screenshot of a video

Description automatically generated

### CPU TImes

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **CPU Time (in s.)** | | | | |
|  | **IDM** | **Feedback** | **DDDP** | **DDP** | **Total** |
| **IDM + DDP** | 0.001 |  |  | 0.018 (11 Iter.) | 0.019 |
| **Feedback + DDP** |  | 0.001 |  | 0.011 (8 Iter.) | 0.012 |
| **DDDP + DDP (K=100 and T=0.25)** |  | 0.001 | 23.428 (13 Iter.) | 0.005 (4 Iter.) | 23.445 |
| **DDDP + DDP (K=50 and T=0.5)** |  | 0.001 | 0.94 (14 Iter.) | 0.003 (4 Iter.) | 0.954 |
| **DDDP + DDP (K=25 and T=1.0)** |  | 0.001 | 0.1 (37 Iter.) | 0.001 (4 Iter.) | 0.11 |