Fakultät für Informatik
Lehrstuhl für Echtzeitsysteme und Robotik

Title

First name and last name

Practical Course *Motion Planning for Autonomous Vehicles* WS 2017/2018

Advisor:	Name of advisor
Supervisor:	Prof. Dr.-Ing. Matthias Althoff
Submission:	01. January 2017

Continuous optimization of an trajectory - A non-convex optimization problem

Arash Kiani, Mubashir Hanif, Marc Schmid
Technische Universität München
Email: Your email

Abstract—**In this paper, we present the implementation and tests of a continuous trajectory planning algorithm, used in an autonomous vehicle from Mercedes-Benz (B̈ERTHA)̈. The algorithm is based on non-linear optimization, which objective function can be designed changed to the wished circumstances as driving fast or very safe. En contraire to the algorithm of BERTHA, there is no guarantee tho a single, global optimum in our algorithm, as we defined our constraints differently, because we use the commonroad scenarios as our environment.**

I. Introduction

In the last years, autonomous driving became a growing topic in Informatics as the computing power increased that we can compute dificult problems in an agreeable time. With this increase in hardware technology, many companies, like Google, Tesla, BMW or Mercedes-Benz, as well as Universities started to invest in autonomous driving. As autonomous driving may become the solution for urban mobility, investing in a safe and optimal trajectory just becomes pressing. In this Practical Course we investigated an algorithm for a trajectory planning task of an IEEE paper written by Julius Ziegler, Philipp Bender, Thao Dang and Christoph Stiller. The trajectory planning problem is a nonlinear optimization problem with nonlinear inequality constraints. The objective function is quadratic. A Newton Type method was used to solve for the trajectory. In this paper we show what we have to change in commonroad to run the solver, as well as the different constraints and different problems we have to solve regarding the algorithm and the scenario.

mds
January 2018

II. The Objective Function and it's constraints

A. Prelude

The scenario of the optimization task is provided by the commonroad model. In the model data for the lanelets, geometry, dynamic and static obstacles, speed limit and vehicle properties are given. The trajectory is computed in the scenarios coordinate frame. No sensory data has to be provided. The bounds of the problem are declared by the scenarios maximal and minimal coordinates, while the driving corridor depends on the definition. As commonroad provides highway scenarios, the driving corridor is large, most of the time over more than four lines. Later we will see, why this is important. The initial trajectory is some kind of handcrafted line with n points, while n equals the scenarios time t devided by the timestep ∇t.

B. Objective function

The non convex minimization is done by the python library scipy.optimize. It returns the local minimum of the Objective function in the form of $\mathbf{x}(t) = (x(t), y(t))^T$ for the centerpoint of the vehicle. The orientation of the vehicle $\psi(t)$ and the curvature$\kappa(t)$ of the trajectory are defined as

$$\psi(t) = arctan\frac{\dot{y}(t)}{\dot{x}(t)} \tag{1}$$

$$\kappa(t) = \frac{\dot{x}(t)\ddot{y}(t) - \dot{y}(t)\ddot{x}(t)}{\sqrt[3]{\dot{x}^2(t) + \dot{y}(t)}} \tag{2}$$

The oobjectiv function for the optimal trajectory is defined as the one that minimzes the integral

$$J\left[\mathbf{x}(t)\right] = \int_{t_0}^{t_0+T} L(\mathbf{x}, \dot{\mathbf{x}}, \ddot{\mathbf{x}}, \dddot{\mathbf{x}},)dt \tag{3}$$

with the summand L :

$$L = \mathbf{w}^T(j_{offs}, j_{vel}, j_{acc}, j_{jerk}, j_{jawr}) \tag{4}$$

$$\mathbf{w} = (w_{offs}, w_{vel}, w_{acc}, w_{jerk}, w_{jawr}) \tag{5}$$

The time t_0 is the starting time of the scenario, while the time $t_0 + T$ equals the length in time of the scenario. The vector $\mathbf{w}$ contains the different weighting factors of the individual summands, which have to be handchosen for different weightings of the summands (given, not optimized). Following the summands of the integrand L will be discussed.

$$j_{offs}(\mathbf{x}(t)) = \left|\frac{1}{2}(d_{left}(\mathbf{x}(t)) + (d_{right}(\mathbf{x}(t))\right|^2$$

pulls the trajectory to the center of the driving corridor. d_{left} and d_{right} are the distance functions to the left and to the right of the driving corridor. One of those has to be negative and one has to be positive, so that the ego vehicle passes in the center of the driving corridor. If the obstacles and the driving corridor are convex, its possible to use the euclidian distane, if the obstacles and driving corridor are non convex, the usage of a pseudo distance function is recommended. In section (adkfjeo) a pseudo distance function is introduced.

$$j_{vel}(\mathbf{x}(t)) = |v_{des}(\mathbf{x}(t) - \dot{\mathbf{x}}(t)|^2$$

describes the error of the velocity vector of the trajectory with respect to a dsired velocity vector. The desired velocity can be handcrafted or is related to the speed limits of the scenario.

If the pseudo distance function is used, the derivative of the function is orthogonal to the desired velocity, so it has to be shifted in the right direction.

$$\mathbf{v}_{des}(\mathbf{x}) = v_{des} \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} \frac{1}{2} \left(\nabla d_{left}(\mathbf{x}) + \nabla d_{right}(\mathbf{x}) \right)$$

These two terms describe the behavior of the trajectory, by developing the position and direction of the velocity. The last 3 terms are smoothness terms, which generate the driving dynamics and comfort.

$$j_{acc}(\mathbf{x}(t)) = |\ddot{\mathbf{x}}(\mathbf{t})|^2$$

$$j_{jerk}(\mathbf{x}(t)) = |\dddot{\mathbf{x}}(\mathbf{t})|^2$$

$$j_{jawr}(\mathbf{x}(t)) = \dot{\psi}^2(t)$$

By minimizing j_{acc}, j_{jerk}, j_{jawr} the forces to the passengers get minimized and the trajectory gets further smoother, by minimizing the jawrate and changes in acceleration.

C. Constraint functions

In the real world, the car is constaint to some physical bounds and as a human, we also would like to not crash into an other vehicle or a wall. So the objective function has to be optimized with respect to some constraints. The constraints for the vehicle are the steering geometry, the maximal acceleration and the maximal velocity. The maximal velocitiy is determined by,

$$|\dot{\mathbf{x}}(t)| \leq v_{max}$$

and the maximal acceleration by,

$$|\ddot{\mathbf{x}}(t)| \leq a_{max}$$

For low velocities, the curvature of the trajectory is limited to the steering angle,

$$-\kappa_{max} \leq \kappa(t) \leq \kappa_{max}$$

at higher velocities to the friction limit of the tires, which can be reduced to a circle of forces at any time

$$||\ddot{\mathbf{x}}(t)||^2 \leq a_{max}^2$$

We can reduce the external constraint for ”not crashing” to a constraint that the distance of the trajectory at time t has to be larger than zero plus some safety- and the vehicle shape($0 + d_s + r_v$) distances to every obstacle. Therefore both distance functions can be used.

$$\mathbf{d}(\mathbf{x}(t), \mathbf{o}(t)) \geq 0 + d_s + r_v$$

where $\mathbf{0}(t)$ is the position of the obstacle at time t.
As part of the practical course, we used different definitions to stay in the driving corridor. As the s̈oft constraintöf the objective function should pull the ego vehicle in the middle of the driving corridor, we have to make sure it’s not finding a local optimum outside the driving corridor. There are 2 methods to guarantee this. Either we set the boundaries of the problem to the driving corridor boundaries, or we use some distance constraint for example if the car drives from left to right,

$$p_{ego} \geq bound_{left}$$

$$p_{ego} \leq bound_{right}$$

or

$$dist(p_{ego}, bound_{left}) \leq 0$$

$$dist(p_{ego}, bound_{right}) \leq 0$$

The second set of constraints just work with the signed distance function, as the euclidian distance does not distinguish between being left or right of an object.

III. Ego vehicle shaping

The ego vehicle is modeled by circles with the diameter of the width of the ego vehicle along the longitudinal axis. The Midpoint of the vehicle is the axis of rotation of the vehicle in the model. In 1 we can see how the vehicle is modeled on

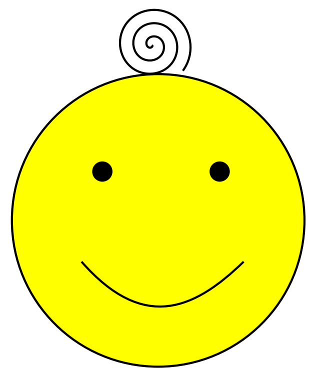

Fig. 1. A vector graphic loaded from a PDF file

the trajectory. If we choose the time step to big, our model may crash due to the first circle of the timestep t-1 and the last circle of t do not intersect anymore, so there might be the option that a car can drive between the timesteps t and t-1 into the trajectory. The timestep in the simulation is choosen so small, that this will never happen.

IV. Solver design

For optimization with constraints, the problem looks like

$$\min_{\mathbf{x}} f(\mathbf{x}) \tag{6}$$

w.r.t.

$$a(\mathbf{x}) \geq 0$$

$$b(\mathbf{x}) = 0$$

The lagrangian with the lagrange multipliers λ, α looks like:

$$\mathcal{L}(\mathbf{x}, \lambda, \alpha) = f(x) - \lambda^T a(\mathbf{x} - \alpha^T b(\mathbf{x}) \tag{7}$$

The solver has to be choosen so that a non convex problem can be solved. The objective function and constraints are two times continuosly differentiable (as long as we use convex polygone shaping this also holds for the euclidean distance). The `scipy.optimize`-toolbox of python has a lot of options. As this is a minimization problem, `scipy.optimize.minimize` was chosen. Constraints are just defined for the'COBYLA'(Constrained Optimization BY Linear Approximation) and 'SLSQP'(Sequential Least SQuares Programming) methods. Further the algorithm 'SLSQP' was chosen, as 'COBYLA' just supports non equality constraints. To lead our vehicle to the goal region of commonroad, we sometimes used equality constraints.

V. Distance function

As the Euclidian distance to a non convex polygone is not differentiable twice. We introduce a pseudo distance function which guarantees the differentiability. A line of a polygone is defined by two adjacent corner points $\mathbf{p}_1$ and $\mathbf{p}_2$ with $\mathbf{t}_1$ and $\mathbf{t}_2$ as corresponding tangent vectors. A pseudo tangent vector is created by interpolating the corner tangent vectors along the segment $p_1\bar{p}_2$

$$\mathbf{t}_\lambda = \lambda \mathbf{t}_2 + (1-\lambda)\mathbf{t}_1$$

which corresponds to the point

$$\mathbf{p}_\lambda = \lambda \mathbf{p}_2 + (1-\lambda)\mathbf{p}_1$$

for $\lambda \in [0, 1]$. For projecting $\mathbf{x}$ we determine λ such that the pseudo tangent is perpendicular to the pseudo normal

$$\mathbf{n}_\lambda \mathbf{t}_\lambda = 0$$

In general, to determine λ, a quadratic term has to be solved.

$$\lambda \mathbf{x}\mathbf{t}_2 + (1-\lambda)\mathbf{x}\mathbf{t}_1 -$$
$$(\lambda^2 \mathbf{p}_2\mathbf{t}_2 + (\lambda - \lambda^2)(\mathbf{p}_2\mathbf{t}_1 + \mathbf{p}_1\mathbf{t}_2) + (1-\lambda)^2\mathbf{p}_1\mathbf{t}_1) = 0$$

ARASH DESCRIBE HOW U CHOOSE LAMBDA ....

VI. Driving corridor and Commonroad

A. Driving corridor

An important point for the objective function is, how to choose the driving corridor. If the driving corridor is wide, the ego vehicle can overtake slow cars and leave the lane, if the driving corridor is narrow, it stays on it's given lanelet and it's hard to overtake slow cars, because the objective function will increase due to j_{offs}. Nevertheless, if the driving corridor is wide, j_{offs} is going to pull the car in the middle of the driving corridor, which might be between two or more lanelets, depending on the scenario.

B. Commonroad

Commonroad is a project of the Technical University Munich, providing realistic data for motion plannning on roads. It includes models of streets, traffic scenarios and tasks an ego vehicle has to perform. This paper focus on the US highway scenarios and simple models of roads. The algorithm does not pay attention to traffic rules, safety rules or behavior of other traffic participants. It's just optimizing the ego trajectory with the given data. There are no probabilistic predictionsif the traffic participants trajectories. They will be treated as given by the commonroad scenario.

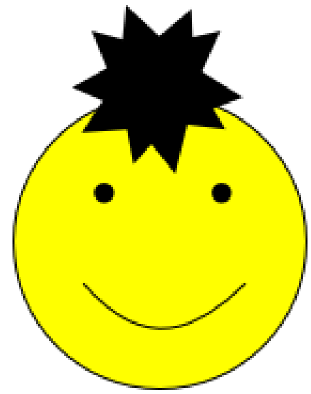

Fig. 2. A bitmap graphic loaded from a PNG file

VII. Conclusion

The conclusion goes here.

References

[1] H. Kopka and P. W. Daly, *A Guide to LaTeX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.