Optimal Trajectories & Control for Automobiles

Tim Wheeler

Abstract—This paper presents a method for the optimal motion planning and path-following of car-like vehicles in the absence of obstacles as a final project for the Stanford University AA222 Multi-Disciplinary Design Optimization class. The problem of optimal control to transition a car-like vehicle between two states is broken down into (a) solving a continuous curvature, minimum-distance path, (b) solving for the velocity profile along said path, and (c) solving the path tracking problem with a high-fidelity car model in the presence of noise. Analytical results for the motion planning problem are reviewed and the velocity profile problem is left for future work by assuming a constant velocity profile. The main contribution of this work is investigation in several methods for optimal trajectory control. A gain-scheduled linear controller, linear quadratic regulator control, and model predictive control are investigated.

1 Introduction

As automated safety systems and fully autonomous driving is introduced, it becomes increasingly necessary to develop tools for their efficient analysis. One of the chief problems in autonomous driving is that of efficiently moving between two states. This paper presents a method for decomposing the problem of optimal state transition into three subproblems: motion planning, the generation of an optimal feasible trajectory between a starting state and a goal state; velocity profile generation, the determination of velocity values over the desired trajectory; and path tracking, or following the desired trajectory with a more complicated car model in the presence of noise.

This paper is the author's final project in the Stanford Multidisciplinary Design Optimization Class for the Spring of 2014. Due to the optimization nature of the class, and the existence of an analytic solution to the motion planning problem, the majority of the project was spent on path tracking.

2 Problem Statement

Consider a car-like vehicle $\mathcal{A}$ whose state and control are described by the vectors $\boldsymbol{x} \in \mathbb{R}^7$ and $\boldsymbol{u} \in \mathbb{R}^2$:

$$\boldsymbol{x} = \begin{pmatrix} x \\ y \\ \theta \\ u \\ v \\ \dot{\theta} \\ \delta \end{pmatrix} \qquad \boldsymbol{u} = \begin{pmatrix} \dot{u} \\ \dot{\delta} \end{pmatrix}$$

where x, y, and θ denote the 2D state and orientation, u and v denote the longitudinal and lateral velocities aligned with the car frame, δ denotes the wheel turn angle, and the dotted variables represent the corresponding time derivatives. The equations of motion are [1]

$$\dot{v} = \tan(\delta)(\dot{u} - \dot{\theta}v) + \left(F_{yf}/\cos\delta + F_{yr}\right)/m - \dot{\theta}u$$

$$I\ddot{\theta} = ma\tan(\delta)(\dot{u} - \dot{\theta}v) + aF_{yf}/\cos\delta - bF_{yr}$$

where m is the mass of the car, a and b are the distance from the center of gravity to the front and rear axles respectively, I is the moment of inertia, and the lateral tire forces are given via a linear tire model with stiffness C,

$$F_{yf} = C\left(\tan^{-1}\left(\frac{v + \dot{\theta}a}{u}\right) - \delta\right)$$

$$F_{yf} = C\tan^{-1}\left(\frac{v + \dot{\theta}a}{u}\right).$$

Given a starting state $\boldsymbol{x_o}$ and a target state $\boldsymbol{x_f}$, find the optimal control $\boldsymbol{u}^\star(t)$ which solves the minimization problem:

• T. Wheeler is with the Department of Aeronautics and Astronautics, Stanford University, Stanford, CA, 94305.
E-mail: wheelert@stanford.edu

$$\begin{array}{ll} \text{inf} & \frac{1}{2}\int_{t_o}^{t_f} \boldsymbol{u}^T(t)R\boldsymbol{u}(t)dt \\ \text{subject to} & \dot{\boldsymbol{x}}(t) = a(\boldsymbol{x},\boldsymbol{u},t) \\ & \boldsymbol{x}(t) \in \mathcal{X} \quad \boldsymbol{u}(t) \in \mathcal{U} \\ & t_f > t_o, \quad \boldsymbol{x}(t_o) = \boldsymbol{x}_o, \quad \boldsymbol{x}(t_f) = \boldsymbol{x}_f \end{array}$$

where R is a potentially time-varying cost matrix, $\mathcal{X}$ is the set of admissible states, and $\mathcal{U}$ is the set of admissible controls.

The above formulation does not lend itself well to current optimization techniques. Dynamic programming, while theoretically useable, would be intractable, and analytical approaches will not be considered. The most common approach is not to directly solve this problem at all, but to split it into two phases, motion planning and trajectory tracking. Techniques are widely available in the literature to solve either of these subproblems, and together they form a near-optimal solution to the overarching problem [2], [3], [4], [5].

3 Motion Planning

The shortest path between two configurations for various wheeled cars have known analytical solutions [6], [7], [8]. Of particular interest is the continuous-curvature car (CC Car), a vehicle move only forward, described by:

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{\kappa} \end{pmatrix} = \begin{pmatrix} cos\theta \\ sin\theta \\ \kappa \\ 0 \end{pmatrix} u + \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} \sigma$$

where the controls are given by u, the forward velocity, and σ, the angular acceleration, which is related to $\dot{\phi}$, the steering velocity of the front wheels. The model is further constrained by setting $u = 1$ and $|\sigma| \leq \sigma_{\max}$. The resulting system has a continuous curvature profile and thus results in paths with continuous steering angles.

Like Dubins or Reeds-Shepp paths, optimal CC Car paths are composed entirely of straight lines and turns of maximum curvature. The main difference here is how the orientation of the car cannot change instantaneously. Instead, continuous-curvature turns (CC Turns) are made up of three parts: (a) a clothoid arc of sharpness $\sigma = \pm\sigma_{\max}$ whose curvature varies from 0 to $\pm\kappa_{\max}$, (b) a circular arc of radius κ^{-1}, and (c) a clothoid arc of sharpness $-\sigma$ whose curvature varies from $\pm\kappa_{\max}$ to 0. It was shown that optimal paths can be computed in much the same way as they are for the Dubins Car, only with line segments restricted to making an angle μ with the CC circles [7].

A good reference for generating clothoids is given here [9].

The end result are six possible optimal paths, each with a closed analytical form. In any given situation it is possible to solve all six equations and take the shortest resulting feasible path.

The CC Car solver thus produces a trajectory with minimum distance between two configurations. Notice, however, that the path contains no information on time or velocity. A constant velocity was used in this project for the sake of simplicity. This assumption is ill-advised for sharp turns at high speeds. For calm driving conditions with a set speed limit this is a reasonable choice.

The actual algorithm was not implemented in this research project due to the author having solved the Dubins and Reeds-Shepp problems previously and this one thus not being worth doing. A trajectory was generated in Julia using the clothoid equations for the purpose of having a feasible reference trajectory to track.

4 Trajectory Tracking

We now turn to the problem of tracking a given feasible trajectory. The tracking problem is a fundamental part of optimal control theory and lends itself to various approaches. The problem is typically formulated as follows:

$$\begin{array}{ll} \text{min} & \frac{1}{2}\Delta\boldsymbol{x}^T(t_f)H\Delta\boldsymbol{x}(t_f) + \\ & \quad +\frac{1}{2}\int_{t_o}^{t_f} \Delta\boldsymbol{x}^T(t)Q\Delta\boldsymbol{x}(t) + \boldsymbol{u}^T(t)R\boldsymbol{u}(t)dt \\ \text{subject to} & \dot{\boldsymbol{x}}(t) = a(\boldsymbol{x},\boldsymbol{u},t) \\ & \boldsymbol{x}(t) \in \mathcal{X} \quad \boldsymbol{u}(t) \in \mathcal{U} \\ & t_f > t_o, \quad \boldsymbol{x}(t_o) = \boldsymbol{x}_o, \quad \boldsymbol{x}(t_f) = \boldsymbol{x}_f \end{array}$$

where H and Q are potentially time-varying cost matrices for penalizing deviation from the path. Notice that this problem is always feasible provided that at least one set of admissible controls exists which keeps the system within $\mathcal{X}$.

First we consider use of a gain-scheduled linear controller for a Dubins car on a straight path. We then consider use of a Linear Quadratic Controller for the simple car. We conclude with an implementation of Model Predictive Control for a more complicated car model.

4.1 Gain-Scheduled Linear Control for a Simple Car

Assume that a feasible trajectory (x_d, u_d) has been provided which satisfies the system dynamics and

$r(t) = h(x_d(t), u_d(t))$. We define a corresponding error system with $e = x - x_d$ and $v = u - u_d$ with dynamics

$$\begin{aligned}\dot{e} &= \dot{x} - \dot{x_d} = f(x,u) - f(x_d, u_d) \\ &= f(e + x_d, v + u_d) - f(x_d) \equiv F(e, v, x_d(t), u_d(t)).\end{aligned}$$

In the case of trajectory tracking the error e is assumed to be small. The controller is expected to be performing well. The dynamics are linearized around $e = 0$:

$$\dot{e} \approx A(t)e + B(t)v$$

$$A(t) = \frac{\partial F}{\partial e}|_{(x_d(t), u_d(t))}, \quad B(t) = \frac{\partial F}{\partial v}|_{(x_d(t), u_d(t))}$$

In the simple case where the original system is linear, it can be shown that $A(t)$ and $B(t)$ are constant, and the error dynamics are simply

$$\dot{e} = Ae + Bv.$$

A control system of the form

$$v = -Ke + k_r r$$

can be used, with constant r. It is necessary to find gain matrix K such that the closed-loop dynamics have the desired behavior and k_r such that the desired output at equilibrium is achieved. Pole-placement is often used to find K. For desired output y = r, it has been shown that [10]

$$k_r = -1/\left(C(A - BK)^{-1}B\right).$$

Regulation of the linearized system $(A(x_d), B(x_d))$ can be achieved assuming x_d and u_d are constant or slowly-varying with respect to the performance criterion. If a state feedback controller is designed for every $K(x_d)$, then one can control the system given by feedback of the form

$$v = K(x_d)e.$$

This corresponds to

$$u = -K(x_d)(x - x_d) + u_d.$$

An example from the references was worked out in order to demonstrate this approach [10].

Consider the model of the Dubins Car, a car-like system defined only by the position of the center of the rear wheels and its orientation. Assume the car can only drive forward. The system dynamics take the form:

$$\begin{bmatrix}\dot{x} \\ \dot{y} \\ \dot{\theta}\end{bmatrix} = \begin{bmatrix} vsin\theta \\ vcos\theta \\ \frac{v}{l}tan\phi\end{bmatrix}$$

where v is the forward velocity, phi the steering angle, and l the wheelbase, or distance between the front and rear axles. The system inputs are v and ϕ.

Consider the problem of tracking a straight line given by $(x(t), y(t)) = (v_r {*} t, y_r)$. The corresponding desired state is $x_d = (v_r * t, y_r, 0)$ and desired input is $u_d = (v_r, 0)$. Note that the equations of motion are satisfied.

The corresponding linearized system is:

$$A_d = \frac{\partial f}{\partial x}|_{(x_d, u_d)} = \begin{bmatrix}0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0\end{bmatrix}$$

$$B_d = \frac{\partial f}{\partial u}|_{(x_d, u_d)} = \begin{bmatrix}1 & 0 \\ 0 & 0 \\ 0 & v_r/l\end{bmatrix}$$

The error dynamics are governed by $\dot{e} = Ae + Bw$, with $e = x - x_d$ and $w = u - u_d$. (Note that w is being used so as not to conflict with the velocity, v). This results in

$$\dot{e}_x = w_1, \quad \dot{e}_y = e_\theta, \quad \dot{e}_\theta = \frac{v_r}{l}w_2.$$

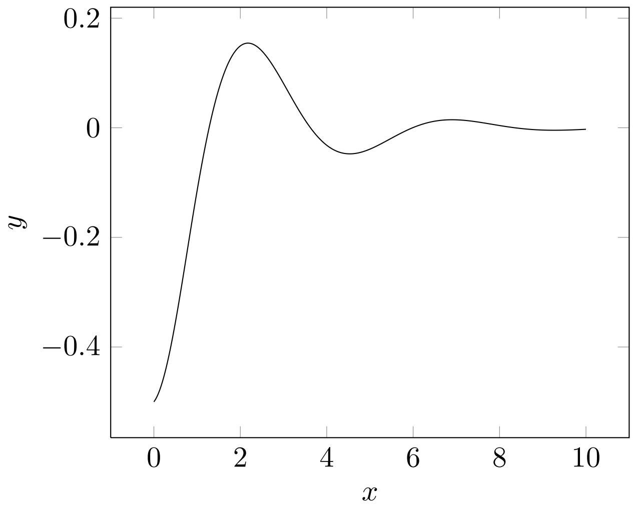
Notice that $\dot{e}_x$ is decoupled from the other two states. It is thus possible to specify separate control dynamics for the longitudinal and lateral dynamics. It was shown that if one desires the closed loop eigenvalues of the longitudinal dynamics (e_x) to be at λ_1 and the closed loop eigenvalues of the lateral dynamics (e_y, e_θ) to be (λ_2, λ_3), then we set [10]:

$$\begin{aligned} w_1 &= -\lambda_1 e_x \\ w_2 &= \frac{l}{v_r}\left(a_1 e_y + a_2 e_\theta\right) \\ a_1 &= -(\lambda_2 + \lambda_3) \\ a_2 &= \lambda_2 \lambda_3.\end{aligned}$$

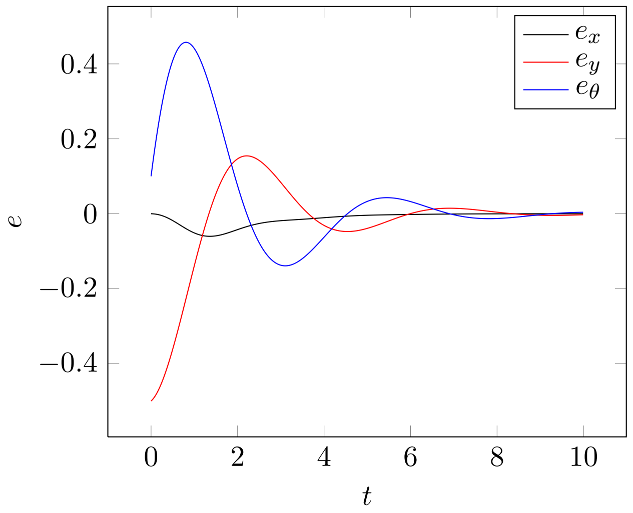
The resulting controller has the form:

$$\begin{aligned}\begin{bmatrix} v \\ \phi \end{bmatrix} &= -\begin{bmatrix}\lambda_1 & 0 & 0 \\ 0 & \frac{a_1 l}{v_r} & \frac{a_2 l}{v_r}\end{bmatrix}\begin{bmatrix} x - v_r t \\ y - y_r \\ \theta\end{bmatrix} + \begin{bmatrix} v_r \\ 0\end{bmatrix} \\ &= K_d e + u_d\end{aligned}$$

The system was simulated using $y_r = 0$, $v_r = 1$, $\lambda = \begin{bmatrix} 1 & -1 & -1 \end{bmatrix}$, $l = 0.05$ and initial state $x(t_o) = \begin{bmatrix} 0 & -0.5 & 0.1 \end{bmatrix}^T$.



The controller clearly stabilizes the system to follow the given trajectory.



We can see that the error for all three components decays towards zero over time.

One limitation of this approach is that a separate set of gains must be designed for every operating points x_d. One way around this is to design gains for a set of operating points and then interpolate the gains between such points for other operating conditions.

We would like to extend this to tracking arbitrary feasible paths. Instead of designing around several operating points we introduce a transformation to convert a given $(x_d(t), x(t))$ pair to a state $x_{rel}(t)$ relative to the zero state $(0, 0, 0)$. This is accomplished with a translation and rotation:

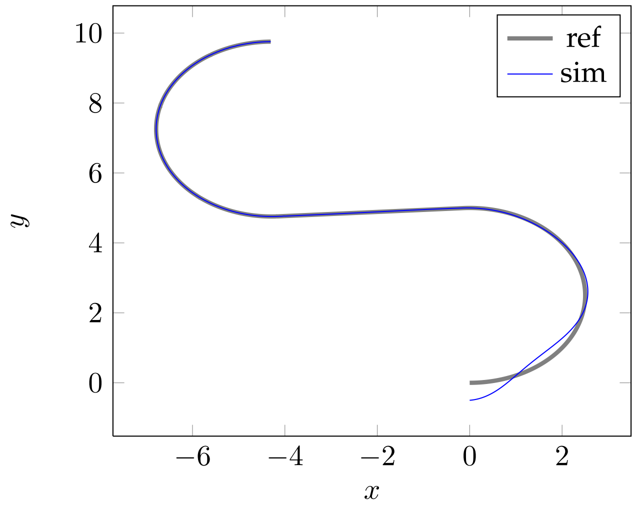
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x - x_d \\ y - y_d \end{bmatrix}, \quad \begin{bmatrix} x_{ref} \\ y_{ref} \\ \theta_{ref} \end{bmatrix} = \begin{bmatrix} x' cos\theta_d + y' sin\theta_d \\ -x' sin\theta_d + y' sin\theta_d \\ \theta - \theta_d \end{bmatrix}$$

The new relative state, $x_{rel}(t)$, acts as a sort of error relative to the zero state. We can thus design one controller, as already demonstrated, and then compute the control input using the same method using the relative state. Notice that we must now include the nominal steering angle when there are turns:

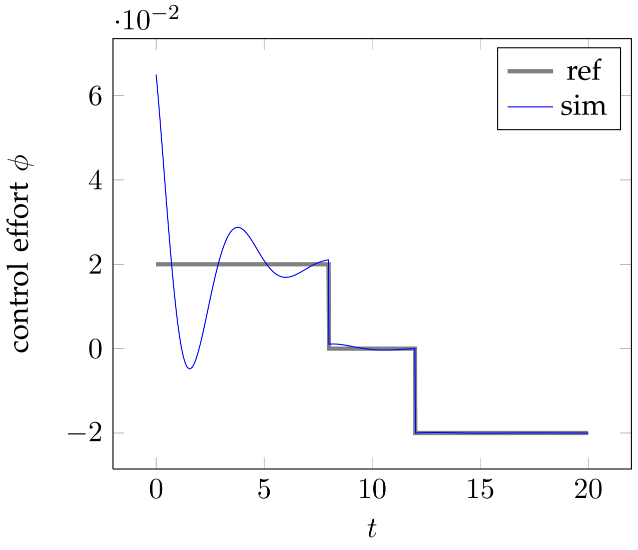
$$\begin{bmatrix} v \\ \phi \end{bmatrix} = K_d x_{rel} + u_d.$$

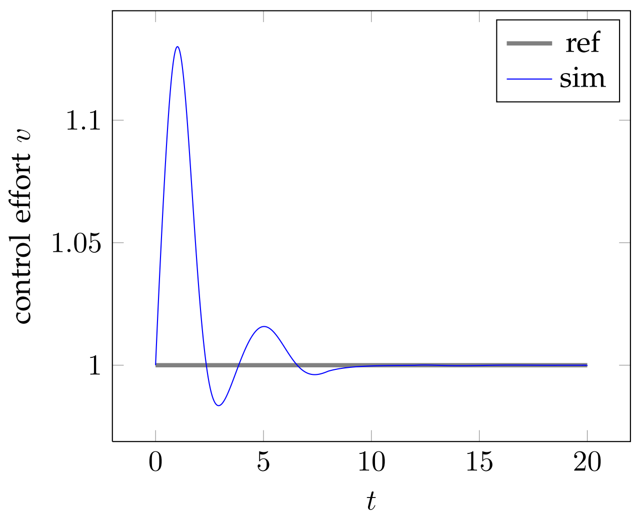
Note that K_d has not changed.

The system was simulated on a trajectory with a left turn, a straight section, and a right turn. The same car dynamics and initial condition were employed. The nominal trajectory and nominal input set are plotted along with the actual results.



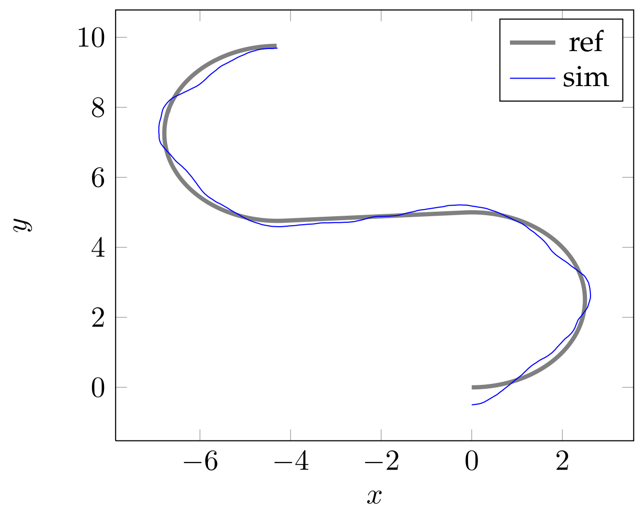
We notice that perfect tracking is recovered.



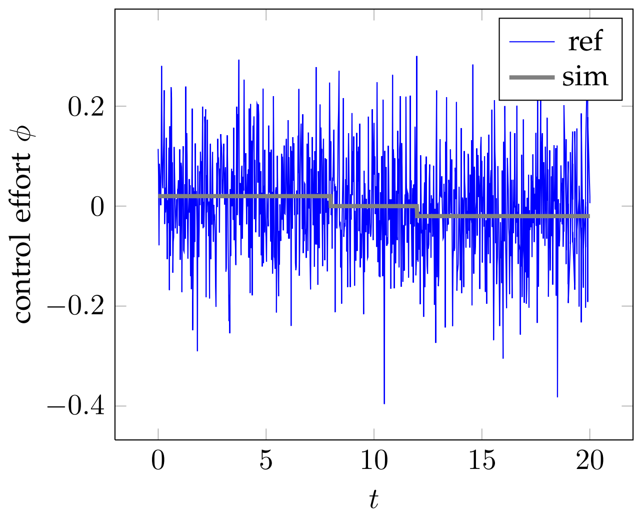


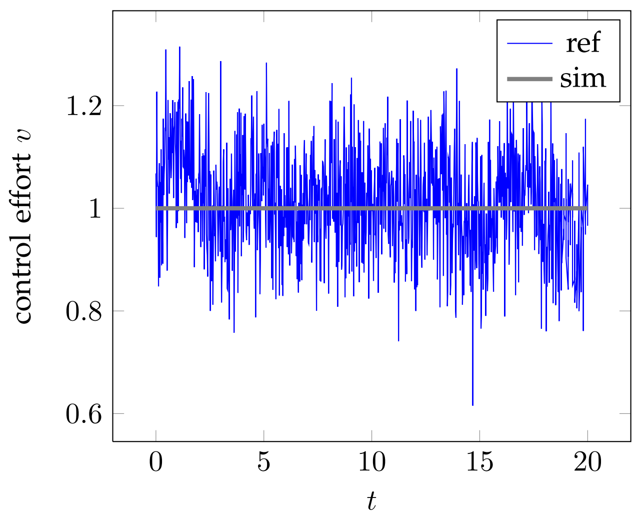
The control inputs quickly settle to the nominal control inputs, as is desired.

The system was also simulated in the presence of control noise. The same system was run with uncorrelated Gaussian noise with standard deviation 0.1 for both control inputs: v and ϕ.



We notice that tracking is still maintained despite this significant amount of noise.





While this control method is successful in controlling a simple car model, it still remains to be seen how it performs on the more complicated car model. Given more time it would have been interesting to use the same method and employ a PID controller to get the input dynamics from the more complicated dynamics to track the simple car values.

4.2 LQR Control for a Dubins Car

We have demonstrated the use of a locally stabilizing time-varying controller for a holonomic system. It is often the case that a nonholonomic system is controllable when linearized about a nonstationary trajectory[11]. The time-varying Linear Quadratic Regulator(LQR) is a popular tracking controller and is well established in the controls literature.

Several attempts were made at implementing such a controller by linearizing the car dynamics about the present location and solving the receding horizon optimal control problem. Unfortunately these methods were not fruitful and need further evaluation. The method of MPC control is a more general approach which did provide some interesting results.

4.3 MPC Control for a CC Car

Model Predictive Control (MPC) is another well-known controls method with attributes which make it a powerful tool for many applications. The fundamental theory of optimal control is split between dynamic programming, which results in closed-loop controllers, and calculus of variations, which results in open-loop controllers. The calculus of variations lend themselves well to simple systems but soon become analytically intractable. Dynamic programming is theoretically applicable to a vast set of control problems, but is often computationally intractable. MPC straddles the line between the two approaches, offering a means of controlling a system with numerical results offered in real time for a pseudo-feedback approach.

The idea behind MPC is to solve a discretized optimization problem at some time t_i to obtain the set of control inputs $\boldsymbol{u} = \left[u_1, \ldots, u_{N-1}\right]$ which minimizes a given cost function. The controller then applies u_1 from the period t_i to t_{i+1}. The MPC problem is then repeated at t_{i+1}.

Taken at a single time slice, MPC merely solves for an open-loop set of control inputs. When considered in succession it can be viewed as a closed-loop system, as it re-optimizes with each time slice.

The drawback with MPC is that it requires online computation of an optimization problem. If the problem is computational intractable or finding the optimal solution is not guaranteed then it may perform very poorly or not work at all. In addition, proving stability of MPC controllers is sometimes difficult.

Discretization is used to convert the general path planning problem formulation into a finite dimensional problem by discretizing the time domain into equal time intervals and imposing the constraints at the temporal nodes.

For any given time interval $\left[0,\ \ t_f\right]$, and time increment, Δt, the temporal nodes are given as

$$t_k = k\Delta t, \qquad k = 0, \ldots, N,$$

where $N\Delta t = t_f$. Next it is necessary to represent the control input, $\boldsymbol{u}(t)$, in terms of a basis function. Here we use piecewise constant basis functions, resulting in the zero-order-hold discretization of a non-linear time invariant system.

The problem is now reformulated as, given the discretized trajectory, at t_i solve:

$$\begin{array}{ll} \text{min} & \boldsymbol{\Delta x}_{i+N}^T H \boldsymbol{\Delta x}_{i+N} + \\ & \quad + \sum_{k=i}^{N-1} \boldsymbol{\Delta x}_k^T Q \boldsymbol{\Delta x}_k + \boldsymbol{u}_k^T R \boldsymbol{u}_k \\ \text{subject to} & \boldsymbol{x}_{i+1} = a(\boldsymbol{x_i}, \boldsymbol{u_i}, t_i) \\ & \boldsymbol{x}(t) \in \mathcal{X} \quad \boldsymbol{u}(t) \in \mathcal{U} \end{array}$$

This problem was implemented for both a CC Car model and for the more complicated car model including friction. Due to numerical effects the sideslip velocity in the latter case was forced to zero. In each case the same precomputed trajectory consisting of a single CC turn and a straight segment were used. The trajectory generation parameters were:

σ_{max}	0.25	Limit on curvature [rad]
ϕ_{max}	$\frac{\pi}{6}$	Limit on steering angle [rad]
Δt	0.1	Time discretization [sec]
v	0.5	Velocity [m/s]

4.3.1 MPC Simulation Results for CC Car Model

The same CC Car parameters were used in the simulation for the path tracker as were used in the generation of the trajectory. Once can therefore expect that the CC Car should be able to track the trajectory well.

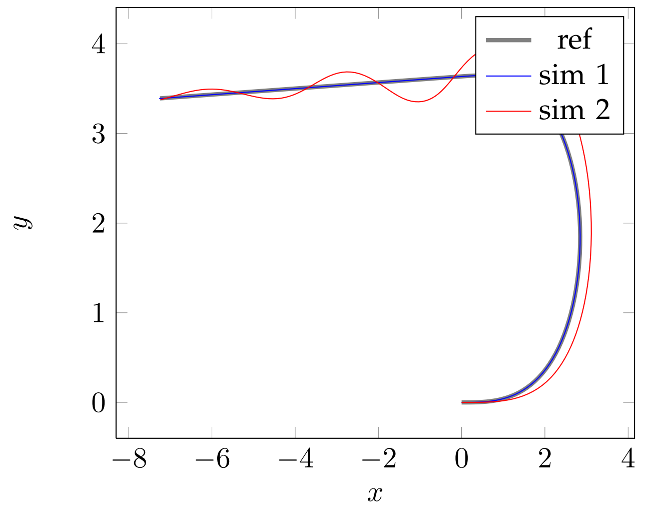
Cost matrices were chosen to primarily penalize deviation from (x, y), which smaller weights on orientation and velocity. Integrated state only considered positional errors to encourage close hugging of the desired trajectory. Two simulations are shown below, one without weight placed on control (1) and one with penalties on control (2). The corresponding cost matrices are:

$$Q = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \qquad N = 2$$

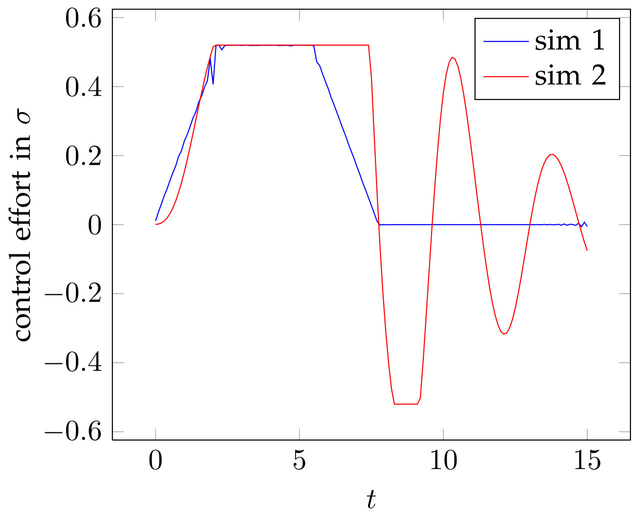
$$H_1 = \begin{bmatrix} 100 & 0 & 0 & 0 \\ 0 & 100 & 0 & 0 \\ 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 50 \end{bmatrix} \qquad H_2 = \begin{bmatrix} 100 & 0 & 0 & 0 \\ 0 & 100 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$R_1 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \qquad R_2 = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix}$$

The simulation was implemented in MATLAB using fmincon as the solver.



Above is the trajectory for both simulations overlaid on the optimal trajectory. We find that we reproduce the desired trajectory when control is not penalized. The second simulation, with an intentionally high penalty, has rather poor performance.



The first simulation nicely follows the curvature used to generate the original trajectory. The trapezoidal form is precisely what is expected. The second simulation was chosen to show how the control effort does adhere to its bounding limits. One can also see how it initially does not put enough effort into tracking the trajectory which results the quick accumulation of error.

4.3.2 MPC Simulation Results for Higher Fidelity Car Model

The higher fidelity car model has the same dynamics as the model initially described, except for numerical reasons the side velocity is set to zero.

The model parameters were:

a	0.3	Distance from cg to rear axle [m]
b	0.1	Distance from cg to front axle [m]
C	100	Cornering Stiffness [kg m2]
I	100	Moment of Inertia [m2]
m	1	Mass [kg]

This system had considerable difficulty tracking the trajectory. Part of this was due to the fact that the system is simply more complicated and the solver being used is not guaranteed to read a global minima (due to non-linear nature of the problem). Other factors include the additional model parameters which do not directly relate to the CC Car. It is not clear what the appropriate set of model values is.

Cost matrices were chosen to only penalize deviation from (x, y), due to numerical instability. Final position was given a much higher penalty than integrated positional error. This was to prevent chatter and allow for smoother path following. A larger horizon ($N = 10$) was used to get the car to follow properly.

Two simulations are shown below, one without weight placed on control (1) and one with penalties on control (2). The cost matrices are:

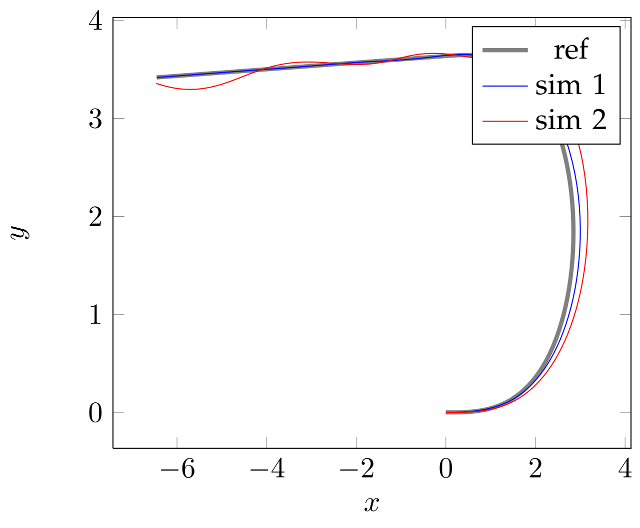
$$H = \text{diag}\begin{pmatrix}100 & 100 & 0 & 0 & 0 & 0\end{pmatrix}$$

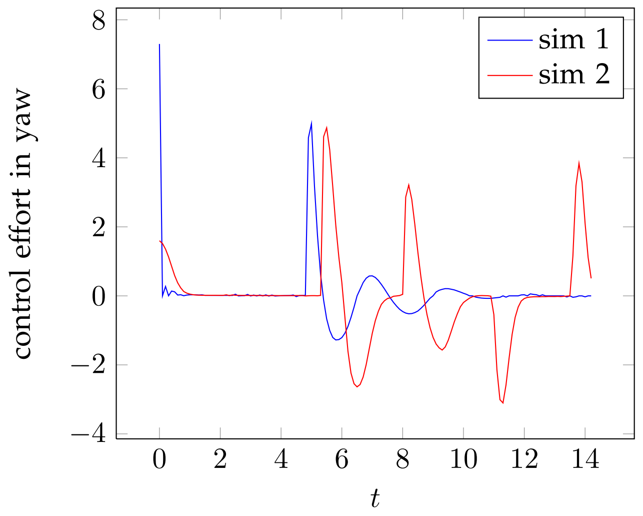
$$Q = \text{diag}\begin{pmatrix}1 & 1 & 0 & 0 & 0 & 0\end{pmatrix}$$

$$R_1 = \begin{bmatrix}0 & 0\\ 0 & 0\end{bmatrix} \quad R_2 = \begin{bmatrix}0.006 & 0\\ 0 & 0.006\end{bmatrix}$$

The simulation was also implemented in MATLAB using fmincon as the solver.

Below is the trajectory for both simulations overlaid on the optimal trajectory. We find that we do not reproduce the desired trajectory but do fairly well. As soon as a penalty is placed on control effort we start to do rather poorly. Getting a stable controller with weighted error was difficult. It is very easy for the car to get knocked off track far enough that it cannot correct itself adequately and the system goes haywire.





The control values here are less intuitive. We notice that the limits in control were sufficiently high to not affect the resulting trajectory. Again we find that the simulation which penalizes control effort begins with less control input, resulting in aggregated error. In the long run this results in far worse control.

5 Conclusion

In this paper, a general method for solving the optimal control problem of driving a car between two states has been proposed. The problem was divided into three components; motion planning, velocity profile generation, and path tracking. Motion planning was shown to be feasible between any two states in an obstacle-free space using continuous curvature paths composed of straight lines, clothoid segments, and arcs of maximum curvature. The problem of velocity profile generation was avoided by assuming a constant velocity profile.

Several methods were employed to solve the path tracking problem. A gain-scheduled linear controller and a relative state transformation was employed to get a simple car to track an arbitrary feasible trajectory in the presence of noise. Model predictive control was used to control both a simple car and a more complicated car model to track a feasible trajectory. Numerical stability was an issue.

Further research opportunities include linearization techniques, such as LQR, and extending the gain-scheduled linear controller to the more complicated car model.

Acknowledgment

The author would like to thank Dr. Kochenderfer and Youngjun Kim for their guidance, time, and commitment to the AA222 class.

References

[1] J. C. Gerdes and E. J. Rossetter, “A unified approach to driver assistance systems based on artificial potential fields,” *Journal of Dynamic Systems, Measurement, and Control*, vol. 123, no. 3, pp. 431–438, 2001.
[2] F. Borrelli, *Predictive Control for linear and hybrid systems*. in preparation, 2014.
[3] P. Morin and C. Samson, “Trajectory tracking for nonholonomic vehicles: overview and case study,” in *Kozlowski, K., éditeur, 4th Inter. Workshop on Robot Motion Control (RoMoCo)*, 2004, pp. 139–153.
[4] R. Pepy, A. Lambert, and H. Mounier, “Path planning using a dynamic vehicle model,” in *Information and Communication Technologies, 2006. ICTTA’06. 2nd*, vol. 1. IEEE, 2006, pp. 781–786.
[5] J. M. Snider, “Automatic steering methods for autonomous automobile path tracking,” *Robotics Institute, Pittsburgh, PA, Tech. Rep. CMU-RITR-09-08*, 2009.
[6] L. E. Dubins, “On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents,” *American Journal of mathematics*, pp. 497–516, 1957.
[7] T. Fraichard and A. Scheuer, “From reeds and shepp’s to continuous-curvature paths,” *Robotics, IEEE Transactions on*, vol. 20, no. 6, pp. 1025–1035, 2004.
[8] S. LaValle, *Planning Algorithms*. New York: Cambridge University Press, 2006.
[9] C. Rouff and M. Hinchey, *Experience from the DARPA Urban Challenge*. Springer-Verlag London Limited, 2011.
[10] R. Murray, *Optimization-Based Control: Trajectory Generation and Tracking*, 2nd ed. Pasadena: California Institute of Technology, 2010.
[11] A. W. Divelbiss and J. T. Wen, “Trajectory tracking control of a car-trailer system,” *Control Systems Technology, IEEE Transactions on*, vol. 5, no. 3, pp. 269–278, 1997.