Lateral & Longitudinal Controller Design For Autonomous Vehicles

Bhardwaj, Aditya
Dept. of Mechanical Engineering
Texas A&M University
College Station,Texas

adityab@tamu.edu

***Abstract*— Considering the safety and reliability, path and velocity tracking are one of the most crucial aspects in the development of autonomous vehicles. This paper utilizes the theory of control system design by developing two different types of controllers and then comparing their efficiency based on various metrics. The architecture of the first controller is modelled as combining two different SISO controllers (Stanley and PID) for path and velocity tracking respectively whilst the second design uses a MIMO LQR controller for tracking both the parameters.**

Keywords—Autonomous Vehicles, self-driving cars, LQR, PID

I. Introduction

Car like robots or self-driving cars are becoming most popular research areas in the field of of autonomous mobility. With the advancement in the field of computation it is comparatively easier than the earlier days to run complex control system design algorithms to accomplish myriad of tasks. The problem now changes from choosing a controller to overcome the limitations of computations to choosing a controller which provides better safety, occupant comfort and best performance.

The primary focus of the paper is to design two different types of controllers and in order to evaluate different metrics to conclude a better design both the controllers are connected to an autonomous vehicle simulator.

A. Abbreviations:

x – Lateral position

y – Longitudinal Position

δ – Steering angle

v – velocity

a_x – acceleration

β – slip angle

F – Forces

PID – Proportional Integral Derivative

LQR – Linear Quadratic regulator

Kp – PID Gain

K – LQR Gain

k – stanley Gain

e – crosstrack error

ψ – heading error

v_e – velocity error

B. CARLA

CARLA[4]: **Car** **L**earning to **A**ct is an open source autonomous vehicle simulator which consists of various tracks and simulation environments and is built upon unreal engine. It has gained popularity for testing algorithm designs typically used for developing self-driving cars. It is a reliable testing environment as it is capable of emulating actual vehicle scenarios.

For our purpose, we will utilize a race-track environment (shown in fig 1) in CARLA where there are no pedestrians and traffic lights, climate setting to daylight noon and no cars. We have done so because we are interested to test only the controllers' path and velocity tracking capabilities and not perception behavior of the vehicle.

Fig 1 Racetrack Scenario in CARLA

C. Kinematic Bicycle Model

A kinematic bicycle model approximates the kinematics of a four wheeled car to kinematics of two wheeled bicycle as shown in fig 2 The controller designed using this model is easier to work with in terms of linearization of the non-linear model.

$$
\begin{aligned}
\dot{x} &= v\cos(\psi+\beta) \\
\dot{y} &= v\sin(\psi+\beta) \\
\dot{\psi} &= \frac{v}{l_r}\sin(\beta) \\
\dot{v} &= a \\
\beta &= \tan^{-1}\left(\frac{l_r}{l_f+l_r}\tan(\delta_f)\right)
\end{aligned}
$$

The parameters in the kinematic design can be easily updated after designing a controller. For eg. a controller designed for a vehicle using the kinematic controller can be used with other vehicles by updating only a few parameters.

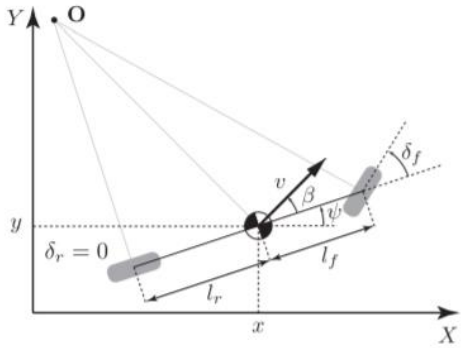

Fig 2 Kinematic Bicycle Model

D. Dynamic Bicycle Model

Similarly, the dynamic bicycle model helps to design a controller for velocity tracking.

$$\begin{aligned}
\ddot{x} &= \dot{\psi}\dot{y} + a_x \\
\ddot{y} &= -\dot{\psi}\dot{x} + \frac{2}{m}\left(F_{c,f}\cos\delta_f + F_{c,r}\right) \\
\ddot{\psi} &= \frac{2}{I_z}\left(l_f F_{c,f} - l_r F_{c,r}\right) \\
\dot{X} &= \dot{x}\cos\psi - \dot{y}\sin\psi \\
\dot{Y} &= \dot{x}\sin\psi + \dot{y}\cos\psi,
\end{aligned}$$

$F_{C,f}$ and $F_{C,r}$ denote the lateral tire forces at the front and rear wheels, respectively, in coordinate frames aligned with the wheels.

E. Method 1:

1. Stanley Controller(For path Tracking):

The Stanley controller is a geometric path tracking approach used by Stanford University's autonomous vehicle entry in the DARPA Grand Challenge, Stanley.[2]. The Stanley method is a nonlinear feedback function of the cross-track error measured from the center of the front axle to the nearest path point for which exponential convergence can be shown. Co-locating the point of control with the steered front wheels allows for an intuitive control law, where the first term simply keeps the wheels aligned with the given path by setting the steering angle δ equal to the heading error.

The corrected steering angle is given by

$$= \begin{cases} \psi(t) + \arctan\frac{k\,e(t)}{v(t)} & \text{if } |\psi(t) + \arctan\frac{k\,e(t)}{v(t)}| < \delta_{max} \\ \delta_{max} & \text{if } \psi(t) + \arctan\frac{k\,e(t)}{v(t)} \geq \delta_{max} \\ -\delta_{max} & \text{if } \psi(t) + \arctan\frac{k\,e(t)}{v(t)} \leq -\delta_{max} \end{cases}$$

2. PID Control(Velocity Tracking)

It is one of the most common and widely used controller. It continuously utilizes the proportional ,integral and derivative error and applies the correct action by generating the gain value required for the control gain. The throttle is computed as:

a = Kp*v_e

F. Method 2

LQR is an optimal controller which provides the optimal gain based on the cost of inputs and system states. In order to find the gain, Infinite time Algebraic Ricaati(ARE) equations are solved at each time step.

x[k+1] = A x[k] + B u[k]

cost = sum($x[k]^T$ *Q*x[k] + $u[k]^T$ *R*u[k])

where A is $\begin{bmatrix} 1 & dt & 0 & 0 & 0 \\ 0 & 0 & v & 0 & 0 \\ 0 & 0 & 1 & dt & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$

B is $\begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ v/L & 0 \\ dt & 0 \end{bmatrix}$ and x is $\begin{bmatrix} e_k \\ e_k - e_{k-1} \\ \Psi_k \\ \Psi_k - \Psi_{k-1} \\ v\_e \end{bmatrix}$

Q = Identity Matrix of size(5x5)

R = Identity Matrix of size(2x2)

And ARE equations represented by :

$$X_{t-1} = Q + A^T X_t A - A^T X_t B (B^T X_t B + R)^{-1} B^T X_t A$$

is solved using Linear Matrix Inequality at each timestep. In order to reduce the computation time a limit of maximum 450 iterations is set to solve the ARE and convergence criteria of ARE is set to 10^{-4}.

The gain is computed using,

$$K = (R + B^T X B)^{-1} B^T X A$$

U* = -K * X

A = U*[1,0]

δ = U*[0,0]

G. System State Update:

After each timestep in method 1 and Method 2 the inputs are mapped to the maximum and minimum prescribed values which are accepted by the simulator actuators. The acceleration (throttle) is mapped with the range [0,1], whilst the steering angle is mapped in the range of [$-\pi/2$, $\pi/2$].

After the actuator input in provided the system states are updated using the linearized bicycle models.

H. Results:

a) Path Tracking:Both the controller seems to track the complete path properly. Stanley controller is is able to *track 94.20% of the path whilst LQR is able to track 96.14% of the path. The results of the path tracking are shown in fig.3 and fig.4*

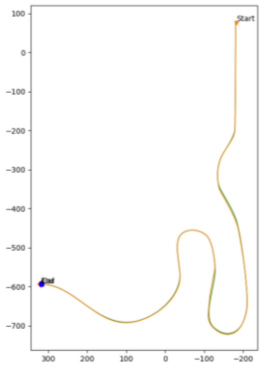

Fig 3 LQR

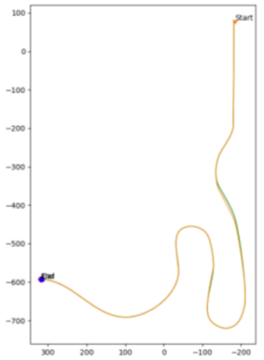

Fig 4 PID + Stanley

b) Velocity Tracking: Similarly, the velocity tracking performance is also reasonably good for both the controllers as shown in fig 5 and fig 6.

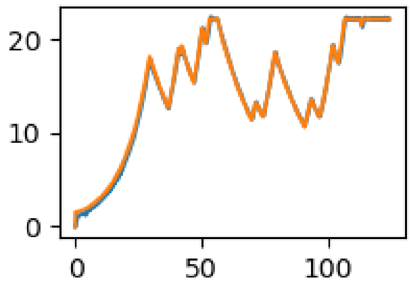

Fig 5 LQR

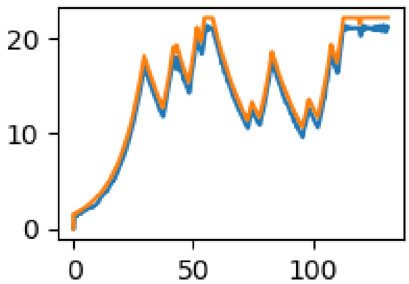

Fig 6 PID + Stanley

c) Behaviour: Although both controllers seem to quantitatively perform well, but the simulation results display a very undesirable behviour in case of stanley controller. The vehicle wobbles a lot which can be dangerous at high speed manuevers.

I. Conclusion:

The LQR controller is more safe, incorporates rider comfort as it takes smooth turns at high speed and low raidius road curvatures and also marginally outperforms the stanley and PID controller in terms of path and velocity tracking. But the issue with LQR is computational power. The PID and Stanley controller requires far less computation power as compared to LQR controller which means that not only sophisticated computation devices are required to run LQR but there will also be a need for more battery storage to support the computations. This being suggested LQR can be utilized for transportations where safety and high speed manueveurs are both prime requirements viz. public transportations whilst PID and stanley can be used in places where safety and accuracy are of not prime concern such as low speed autonomous golf carts, delievery vehicles,etc.

J. Future Work:

This comparison can be further expanded by including other types of controllers such as Model Predictive controller.

The assumption of more computation power requirement by the LQR is made by checking the simulator's refresh rate in case of both the controllers. More accurate comparison for the requirements of the computation power by different controllers can be done which might further help to take a decision in chossing a controller.

References

[1] Jingjing Jiang and Alessandro Astolf Lateral Control of an Autonomous Vehicle, IEEE TRANSACTIONS ON INTELLIGENT VEHICLES, VOL. 3, NO. 2, JUNE 2018.

[2] Jarrod M. Snider Automatic Steering Methods for Autonomous CMU-RI-TR-09-08, Robotics Institute,Carnegie Mellon University,Pittsburgh, Pennsylvania,February 2009

[3] Sebastian Thrun, Mike Montemerlo, Hendrik Dahlkamp, David Stavens, Andrei Aron, James Diebel, Philip Fong, John Gale, Morgan Halpenny, Gabriel Hoffmann, Kenny Lau, Celia Oakley, Mark Palatucci, Vaughan Pratt, Pascal Stang, Sven Strohband, Cedric Dupont, Lars-Erik Jendrossek, Christian Koelen, Charles Markey, Carlo Rummel, Joe van Niekerk, Eric Jensen, Philippe Alessandrini, Gary Bradski, Bob Davies, Scott Ettinger, Adrian Kaehler, Ara Nefian, and Pamela Mahoney. Stanley: The robot that won the DARPA grand challenge. Journal of Field Robotics, 23(9):661–692, 2006.

[4] [3] Alexey Dosovitskiy and German Ros and Felipe Codevilla and Antonio Lopez and Vladlen Koltun CARLA: An Open Urban Driving Simulator, Proceedings of the 1st Annual Conference on Robot Learning, 2017.