Autonomous Driving Control and Field Tests

Summer Research Report

Lu Wen

Undergraduate student
Tsinghua University

Supervised by Professor Masayoshi Tomizuka
Ph.D student Wei Zhan
Ph.D student Chen Tang

UC Berkeley

July. 2017 – September. 2017

Contents

I. Introduction

This report introduces the whole work of a two-month internship. The final purpose of the work is to fulfill the trajectory following function of a real autonomous vehicle provided by Berkeley Deep Drive, by studying vehicle control.

Intelligent vehicle is a significant part of future intelligent traffic, and dynamic control is an indispensable and fundamental component. With satisfactory low-level control, the vehicle provides upper layer of planning and decision making with a platform that has eliminated other disturbance.

Vehicle low-level control contains lateral and longitudinal control. Longitudinal control involves navigating cruise at predefined and pre-computed velocity. Lateral control aims at presupposed trajectory tracking. It is hard to get a precise dynamic model by the inherent coupling and highly non-linearity of the vehicle, so the trajectory tracking remains a challenging problem.

This report presents most work on trajectory tracking work based on lateral control. Generally, common ways of trajectory tracking control are: PID, feed-forward and feedback control, LQR and MPC. PID is widely used in engineering owing that it dispenses with system model. However, a great deal of experimental work is need to get appropriate parameters. Feed-forward and feedback control is used more in recent years. Feed-forward controller majorly make compensation for disturbances caused by curvature changes, and feedback controller aims to minimize the influence of disturbances and model error. LQR gets linearized model by real-time linearizing tracking error system at each sampling time in receding horizons, and then solves LQR problems to get state control outputs. LQR is suitable for trajectory tracking under high-way scenes and most city scenes. When tires are working under the linearized region, LQR has rather good control performance, but would be sensitive to sudden change of curvature and get decrease in precision if tires work under non-linearized region. In addition, LQR fails to deal with constrains. MPC is known for great potential of solving problems with complex constrains, and the feasibility and reliability of which has been verified through both simulation and experiments. MPC is suited for this class of problems since it can handle Multi-Input Multi-Output systems with input and state constraints while taking into account the nonlinear dynamics of the vehicle. Our work mainly

involves the implement of LQR and MPC control.

The remainder of the report is structured as follows: In Section II, information about the experiment vehicle will be presented. Section III briefly introduces about the state estimation work. Section IV gives out the kinematic and dynamic models used in different algorithms. In Section V, the algorithms that have been implemented on the vehicle are introduced in detail, with some evaluations as well. Section VI discusses about some of our experiment results. Lastly, Section VII concludes with potential work and improvements for future research.

II. Vehicle Information

The experiment vehicle is a Lincoln MKZ, refit by Autonomous Stuff. It's implemented with some sensors and software to assist autonomous driving.

1) Hardware

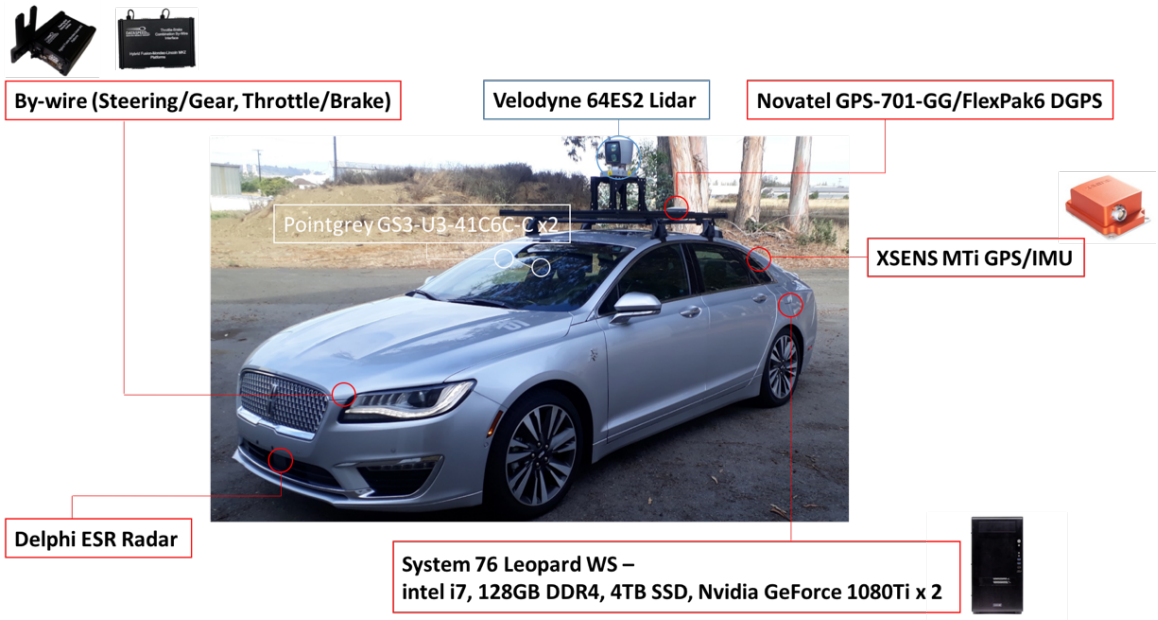

Figure 1. Sensors and computer implemented on MKZ

2) By-wire

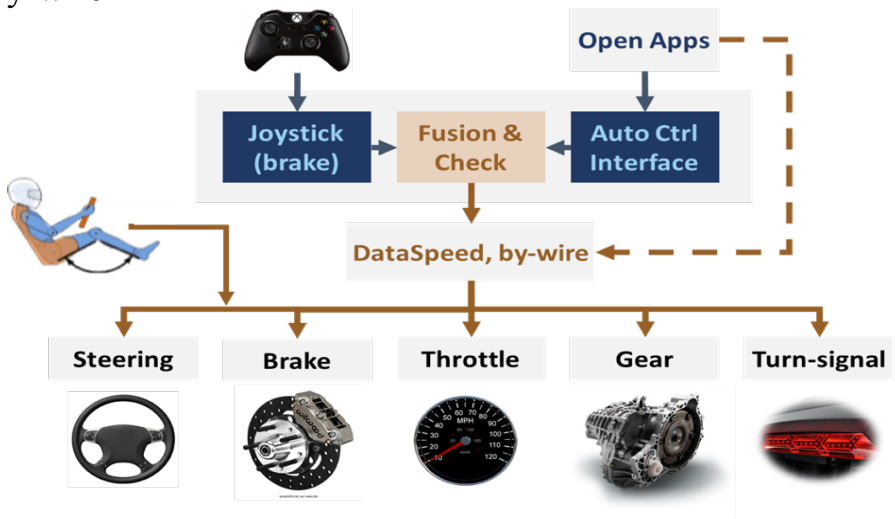

Figure 2.1 By-wire working flow chart supported by Data Speed

The by-wire kits receive the sensor data over the private Dataspeed CAN bus, and transmit the data to corresponding controller to get the command to the actuators of the vehicle. Besides, commands can also be sent directly to actuators.

Between upper commands and actuators, by-wire kits contain two controllers: speed controller and steering controller. Both are closed-loop PID controller, and structure of each is as follows:

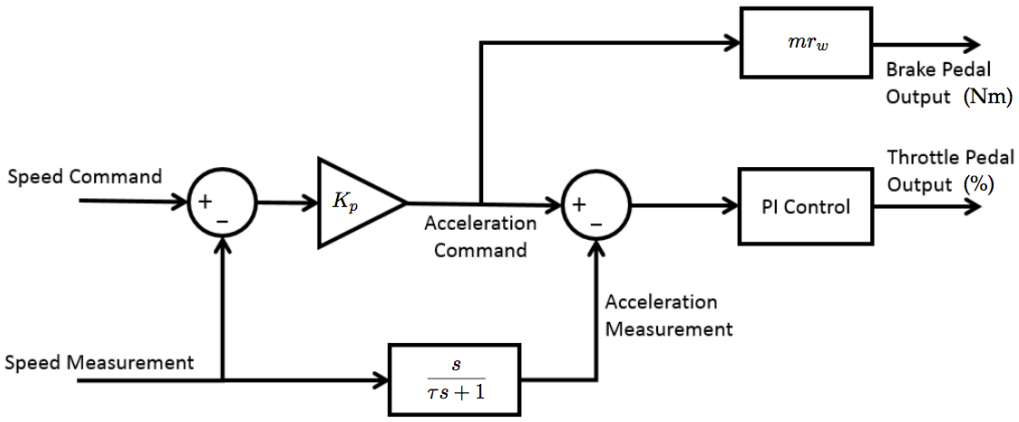

Figure 2.2 Structure of the speed controller

Speed controller: the vehicle speed measurement feedback comes from the **steering_report** topic. Acceleration feedback is derived by running the speed measurement through a differentiator and first-order low pass filter,

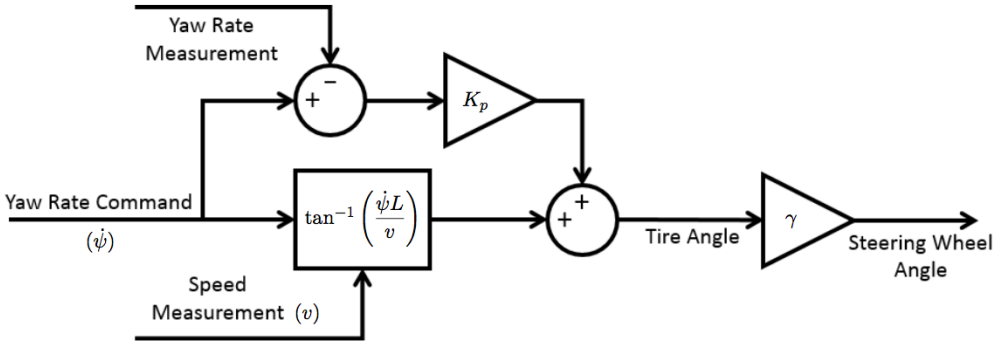

Figure 2.3 Structure of the steering controller

Steering controller: steering control is implemented with a feed-forward proportional controller, and the requested yaw rate and current speed measurement are used to compute a nominal steering angle based on a simple kinematic bicycle model

While turning with significant lateral acceleration, the car body roll affects the accuracy of the yaw rate measurement from the built-in vehicle gyro because the axis of rotation becomes slightly misaligned with the vehicle gyro's sensitive axis. Experiments have shown that the vehicle performs better in tighter turns with just the feed-forward control term because of this effect.

III. State Estimation

Extended Kalman filter (EKF) is used to fulfill the state estimation. The model used for state estimation is a non-linear kinetic model. The state consists of v_x, v_y, x, y, ψ, the input includes a_x, a_y, w_z and the output is the state at next time step. Formulations are as follows:

$$v_{x,k+1} = v_{x,k} + dt((a_x + n_0) + v_{y,k}(w_z + n_2)) \quad (1.1)$$

$$v_{y,k+1} = v_{y,k} + dt((a_y + n_1) + v_{x,k}(w_z + n_2)) \quad (1.2)$$

$$x_{k+1} = x_k + dt((v_{x,k}\cos(\psi_k) - v_{y,k}\sin(\psi_k) + n_3) \quad (1.3)$$

$$y_{k+1} = y_k + dt((v_{y,k}\cos(\psi_k) + v_{x,k}\sin(\psi_k) + n_4) \quad (1.4)$$

$$\psi_{k+1} = \psi_k + dt(w_z + n_1) \quad (1.5)$$

Where $v_{x,k}$ is the longitudinal velocity. $v_{y,k}$ is the lateral velocity. x_k and y_k are X and Y under universal transverse mercator (UTM) coordinate system respectively. ψ_k is the yaw angle. k is the discrete-time index.

a_x denotes longitudinal acceleration. a_y denotes lateral acceleration. w_z denotes yaw rate.

n is the measurement noise. Kalman filter assumes that n is a zero-mean, random variable.

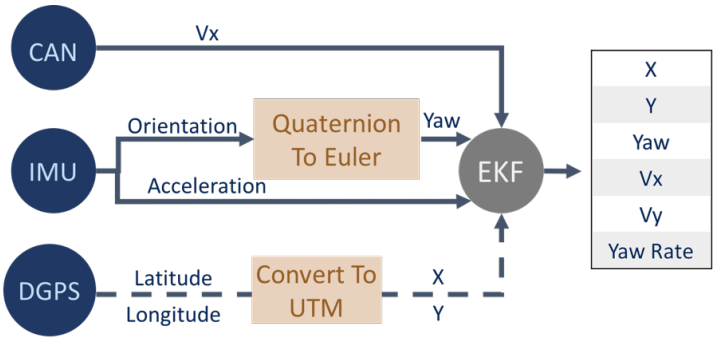

Figure 3. State Estimation.
EKF gets velocity from CAN data, yaw and acceleration from IMU, and x, y from DGPS.

There're two working modes for state estimation: 1) when there is location data from DGPS, EKF works with data from controller area network (CAN), inertial measurement unit (IMU) and differential global positioning system (DGPS); 2) when there is no DGPS data received, estimation simply relies on CAN and IMU.

IV. Vehicle Model

In this section, vehicle models used in control algorithms are introduced.

1) Kinematic model

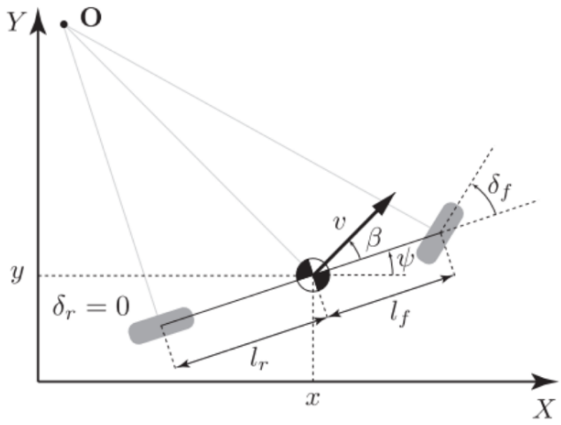

Figure 4. Kinematic bicycle model

The nonlinear continuous time equations that describe a kinematic bicycle model in an inertial frame are:

$$\dot{x} = v \cdot cos\,(\psi + \beta) \tag{2.1}$$

$$\dot{y} = v \cdot (\psi + \beta) \tag{2.2}$$

$$\dot{\psi} = \frac{v}{l_r} \cdot sin\,(\beta) \tag{2.3}$$

$$\dot{v} = a \tag{2.4}$$

$$\beta = tan^{-1}\left(\frac{l_r}{l_f + l_r} tan\left(\delta_f\right)\right) \tag{2.5}$$

where x and y are the coordinates of the center of mass in an inertial frame (x, y). ψ is the inertial heading and v is the speed of the vehicle. l_f and l_r represent the distance from the center of the mass of the vehicle to the front and rear axles, respectively. β is the angle of the current velocity of the center of mass with respect to the longitudinal axis of the car. a is the acceleration of the center of mass in the same direction as the velocity. The control inputs are the front and rear steering angles δ_f , and α. Since in most vehicles the rear wheels cannot be steered, we assume δ_r = 0.

2) Dynamic model

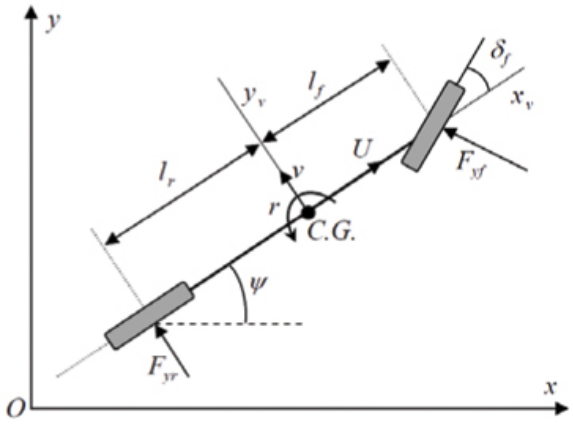

Figure 5. Lateral dynamic model of vehicle

$$\frac{dx}{dt} = v_x \cdot cos\theta\ - v_y \cdot sin\theta \tag{3.1}$$

$$\frac{dy}{dt} = v_x \cdot sin\theta + v_y \cdot cos\theta \tag{3.2}$$

$$\frac{dv_x}{dt} = a \tag{3.3}$$

$$\frac{d\theta}{dt} = \dot{\theta} \tag{3.4}$$

$$\frac{dv_y}{dt} = tan\delta\left(a - \dot{\theta} \cdot v_y\right) + \left(\frac{F_{yf}}{cos\delta} + F_{yr}\right)/m - \dot{\theta} \cdot v_x \tag{3.5}$$

$$\frac{d\dot{\theta}}{dt} = \frac{m \cdot l_f \cdot tan\delta}{I_z}\left(a - \dot{\theta} \cdot v_y\right) + \frac{l_a \cdot F_{yf}}{I_z \cdot cos\delta} + \frac{l_b \cdot F_{yr}}{I_z} \tag{3.6}$$

m and I_z denote the vehicle's mass and yaw inertia, respectively. $F_{c,f}$ and $F_{c,r}$ denote the lateral tire forces at the front and rear wheels, respectively, in coordinate frames aligned with the wheels.

V. Control Algorithm

1) Pure Pursuit

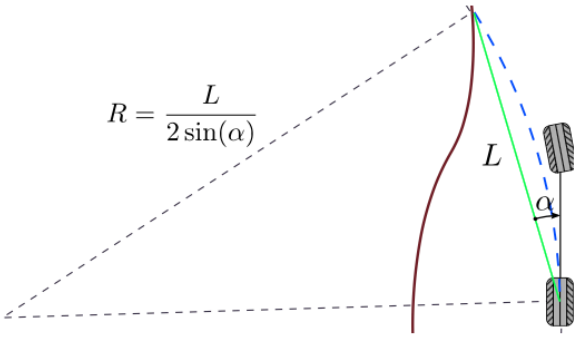

Figure 6. Pure Pursuit geometry

A circle (blue) is fit between rear wheel position and the reference path (brown) such that the chord length (green) is the look ahead distance L and the circle is tangent to the current heading direction.

The control law is based on fitting a semi-circle through the vehicle's current configuration to a point on the reference path ahead of the vehicle by a distance L called the look-ahead distance. Figure 4 illustrates the geometry. The circle is defined as passing through the position of the car and the point on the path ahead of the car by one look-ahead distance with the circle tangent to the car's heading. The curvature of the circle is given by

$$\kappa = \frac{2\sin(\alpha)}{L}$$

For a vehicle speed v_r , the commanded heading rate is

$$\omega = \frac{2v_r sin(\alpha)}{L}$$

α is given by

$$\alpha = arctan\left(\frac{y_{ref} - y_r}{x_{ref} - x_r}\right) - \theta$$

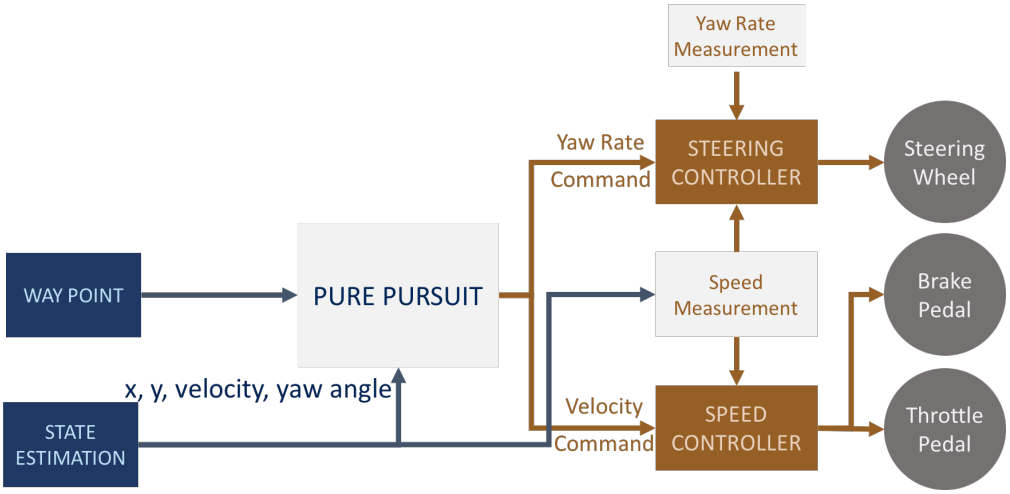

Figure 7. Pure pursuit working flow chart

2) Optimal Control

2.1) Discretization and linearization

Optimal control here is based on the dynamic model as shown in the section IV. The model is transformed to a linear, time-variant , discrete-time model for standard linear quadratic regulator (LQR) and model predictive control (MPC).

Discretization: the discretization is fulfilled with Runge-Kutta method:

$$s = \begin{bmatrix} x & y & \theta & v_x & v_y & \dot{\theta} \end{bmatrix}^T \tag{4.1}$$

$$\dot{s} = f(t,s): \begin{cases} \frac{dx}{dt} = v_x \cdot cos\theta - v_y \cdot sin\theta & (4.2) \\ \frac{dy}{dt} = v_x \cdot sin\theta + v_y \cdot cos\theta & (4.3) \\ \frac{d\theta}{dt} = \dot{\theta} & (4.4) \\ \frac{dv_x}{dt} = a & (4.5) \\ \frac{dv_y}{dt} = tan\delta\left(a - \dot{\theta} \cdot v_y\right) + \frac{\frac{F_{yf}}{cos\delta} + F_{yr}}{m} - \dot{\theta} \cdot v_x & (4.6) \\ \frac{d\dot{\theta}}{dt} = \frac{m \cdot l_f \cdot tan\delta}{I_z}\left(a - \dot{\theta} \cdot v_y\right) + \frac{l_a \cdot F_{yf}}{I_z \cdot cos\delta} + \frac{l_b \cdot F_{yr}}{I_z} & (4.7) \end{cases}$$

The length of horizon is N=15.

Then we can get the discretized model equation:

for n = 0, 1, 2, 3, …,N-1, using

$$s_{n+1} = s_n + \frac{dt}{6}(k_1 + 2k_2 + 2k_3 + k_4) \tag{5.1}$$

$$t_{n+1} = t_n + dt \tag{5.2}$$

$$k_1 = f(t_n, s_n) \tag{5.3}$$

$$k_2 = f(t_n + \frac{dt}{2}, s_n + \frac{dt}{2}k_1) \tag{5.4}$$

$$k_3 = f(t_n + \frac{dt}{2}, s_n + \frac{dt}{2}k_2) \tag{5.5}$$

$$k_4 = f(t_n + dt, s_n + dt \cdot k_3) \tag{5.6}$$

Linearization: we assume that the model used for computation is expressed through the following discrete-time nonlinear system :

$$s(t) = f(s(t), u(t)) \tag{6.1}$$

where s is the state vector, u is the control input vector, $f(\cdot, \cdot)$ is the state update function. System (6.1) is subject to the following state and input constraints:

$$s(t) \in \mathcal{X}, \quad u(t) \in \mathcal{U} \tag{6.2}$$

Consider the $s_0 \in \mathcal{X}$ and the input $u_0 \in \mathcal{U}$. Denote by $s_0(k)$ for $k \geq 0$ the state trajectory obtained by applying the input sequence $u(k) = u_0(k)$ for $k \geq 0$, to the system (6.1) with $s_0(0) = s_0$, i.e.:

$$s_0(k+1) = f(s_0(k), u(k)) \tag{6.3}$$

$$u(k) = u_0(k), \tag{6.4}$$

$$s_0(0) = s_0. \tag{6.5}$$

System (6.1) can be approximated by the following linearized system:

$$s(k+1) - s_0(k+1) = A_k\big(s(k) - s_0(k)\big) + B_k(u(k) - u_0(k)) \tag{6.6}$$

where A_k and B_k are defined as:

$$A_k = \frac{\partial f}{\partial s}\Big|_{s_0(k),\, u_0(k)}, \quad B_k = \frac{\partial f}{\partial u}\Big|_{s_0(k),\, u_0(k)} \tag{6.7}$$

The equation (6.7) describes the deviations of the nonlinear system (6.1) from the state trajectory $s_0(t)$, when an input sequence $u_0(t)$ is applied. And s_0, u_0 are optimal quantities programmed at last time step.

Alternatively, the system (6.6) can be rewritten as

$$s(k+1) = A_k s(k) + B_k u(k) + d_k(k) \tag{6.8}$$

where $d_k(k) = s_0(k+1) - A_k s_0(k) - B_k u_0(k)$ for $k \geq 0$.

Then we extend the model as:

$$\begin{bmatrix} s_{k+1} \\ u_k \\ u_{k-1} \end{bmatrix} = \begin{bmatrix} A_k & 0 & 0 \\ 0 & 0 & 0 \\ 0 & I & 0 \end{bmatrix} \begin{bmatrix} s_k \\ u_{k-1} \\ u_{k-2} \end{bmatrix} + \begin{bmatrix} B_k \\ I \\ 0 \end{bmatrix} u_k + \begin{bmatrix} d_k \\ 0 \\ 0 \end{bmatrix} \tag{6.9}$$

and rewrite the model equation as:

$$x_{k+1} = A_k^{aug} x_k + B_k^{aug} u_k + D_k \tag{6.10}$$

where $x_{k+1} = \begin{bmatrix} s_{k+1} \\ u_k \\ u_{k-1} \end{bmatrix}$, $D_k = \begin{bmatrix} d_k \\ 0 \\ 0 \end{bmatrix}$, $A_k^{aug} = \begin{bmatrix} A_k & 0 & 0 \\ 0 & 0 & 0 \\ 0 & I & 0 \end{bmatrix}$, $B_k^{aug} = \begin{bmatrix} B_k \\ I \\ 0 \end{bmatrix}$

2.2) Linear Quadratic Regulator

The objective function can be defined as:

$$J_0^*(x(0)) \triangleq \min_{U_0} (x_N' - x_{ref,N}') Q (x_N - x_{ref,N}) + \sum_{k=0}^{N-1} \left[(x_k' - x_{ref,k}') Q (x_k - x_{ref,k}) + u_k' R u_k \right] \tag{6.11}$$

subject to: $x_{k+1} = A_k^{aug} x_k + B_k^{aug} u_k + D_k$ (6.12)

$x0 = x(0)$ (6.13)

Q is the state weight matrix; R is the input weight matrix
Q and R are both symmetric positive semi-definite.

The Q weight matrix sets penalty on x, y under the UTM coordinate system. Usually the control and evaluation of vehicle's motion is referred in latitude and longitude way, so a rotation matrix is multiplied to the Q weight matrix.

$$Q = \begin{bmatrix} q_x & 0 & 0 \\ 0 & q_y & 0 \\ 0 & 0 & \ddots \end{bmatrix} \tag{7.1}$$

Define:

$$Q.block(0,0,2,2) = \begin{bmatrix} q_x & 0 \\ 0 & q_y \end{bmatrix} \tag{7.2}$$

Rotation Matrix $RT(\theta) = \begin{bmatrix} cos\theta & -sin\theta \\ sin\theta & cos\theta \end{bmatrix}$ (7.3)

The new Q matrix is:

$$Q.block(0,0,2,2) = RT(\theta)^T \cdot Q.block(0,0,2,2) \cdot RT(\theta) \tag{7.4}$$

The new Q weight matrix sets penalty on x, y under the relative coordinate system now.

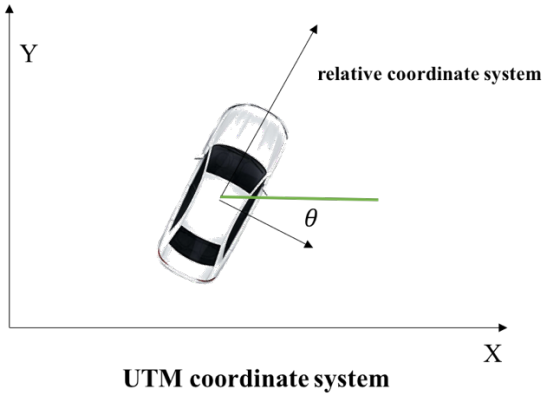

figure 8. coordinate system rotation

Since the LQR cannot enforce hard constraints on inputs, there may exist saturation of inputs. There's a simulation result illustrating the saturation of inputs in the VI. part.

2.3) Model Predictive Controller

Constrains on actuators should be considered during the real test, and the MPC can solve optimization problems with hard constrains.

MPC is based on the model that derived from the same discretization and linearization process, with conversion of Q weight matrix from UTM to relative coordinate system as well. And the objective function is almost the same, added with constrains:

$$J_0^*\left(x(0)\right) \triangleq \min_{U_0} \; x_N' P x_N + \sum_{k=0}^{N-1}[x_k' Q x_k + u_k' R u_k] \tag{8.1}$$

$$\text{subject to:} \quad x_{k+1} = A_k^{aug} x_k + B_k^{aug} u_k + D_k \tag{8.2}$$

$$x0 = x(0) \tag{8.3}$$

$$|u| = \begin{vmatrix} a \\ w \end{vmatrix} \leq \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} \tag{8.4}$$

$$|u_n.(w) - u_{n-1}.(w)| \leq b_3 \tag{8.5}$$

b_1: max acceleration; b_2: max wheel angle; b_3: max wheel angle rate

The input constraint (8.4) restricts the max acceleration and wheel angle; the input constraint (8.5) restricts the wheel angle rate.

We use an online QP solver – qpOASES, to get fast and reliable solution of convex quadratic programming problems in real-time.

VI. Experiment Results

1) State Estimation

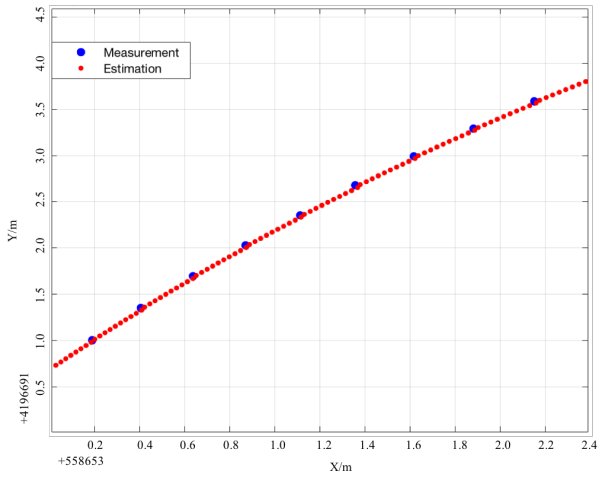

Figure 9.1 X,Y estimation result

Blue dots stand for measured location information from DGPS, at a frequency of 10 Hz. Red dots are estimation results obtained from state estimation node, whose working rate is 100 Hz.

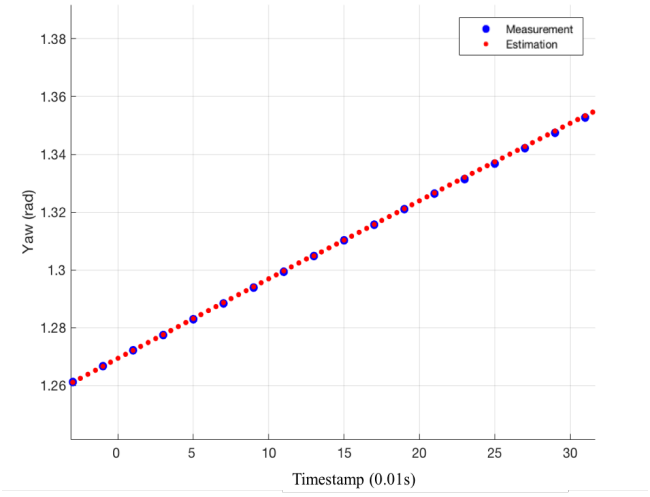

Figure 9.2 Yaw tracking results

Blue dots stand for measured yaw angles from IMU, at a frequency of 25 Hz. Red dots are estimation results obtained from state estimation node, whose working rate is 100 Hz.

From figure 9.1 and figure 9.2, it can be observed that estimation results play in accord with the trend of measurement on the whole.

2) LQR vs. MPC

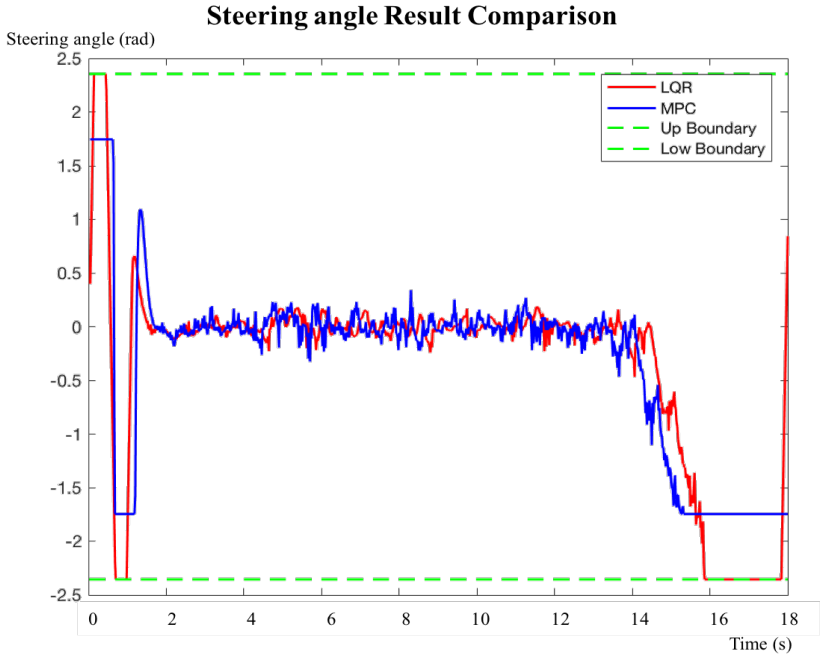

Figure 10. Steering angle Result Comparison

Figure 10 shows the steering angle result of a simulated trajectory tracing. The simulation is set with an initial error on x and y (-2 meters respectively). The result shows that there may be saturation of LQR optimization. As MPC has taken the hard constraints into account, it gives out better optimization results.

3) Tracking Tests

We mainly put MPC into the tracking of two different trajectories, each of which tests different performance on the vehicle. We get these two trajectories by driving along the trajectories to get data that are need to produce the trajectories. We collect location data -- x and y, lateral and longitude velocity, and yaw rate from state estimation at the rate of 100 Hz. Then add time label to each trajectory point from 0 second with a 0.01-second interval.

i. Trajectory I

The first trajectory consists of a 50-meter-straight-way, and a 90-degree-turn.

Figure 11.1 Trajectory One Map
At Partners for Advanced Transportation Technology (PATH, Berkeley)

The tracking results are evaluated in the following ways:

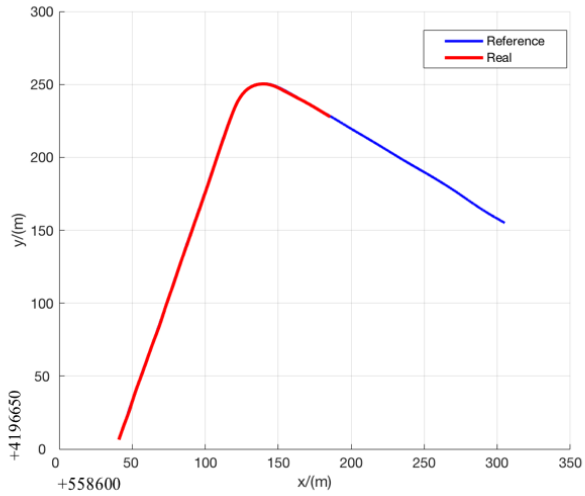

Figure 11.2 Trajectory tracking result

Figure 11.2 shows the X and Y tracking results. Numerically, the average error is 0.1917 meters, the max error is 0.3296 meters. (The accuracy of DGPS reaches 0.01 meter).

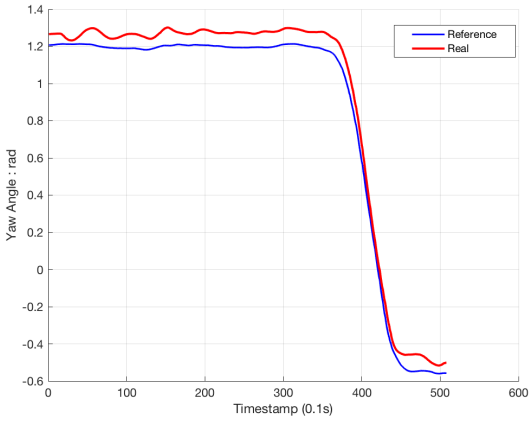

Figure 11.3 Yaw angle tracking result

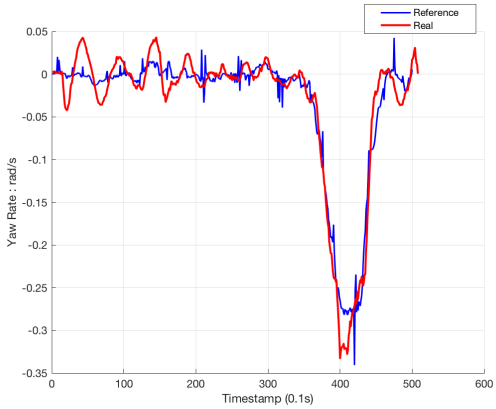

Figure 11.4 Yaw rate tracking result

From Figure 11.3, a steady error around 0.1 rad on yaw angle can be observed. It is speculated that the error partly lies on the unreliability of IMU measurement. During the data-collecting process to produce the referenced trajectories, the inaccuracy of IMU may lead to a inaccurate referenced yaw rate. Taking into account of this, we put little penalty on yaw rate during tracking experiments. Thus, the steady error is explicable in this sense.

ii. Trajectory II

Figure 12.1 Trajectory II Map
At Partners for Advanced Transportation Technology (PATH, Berkeley)

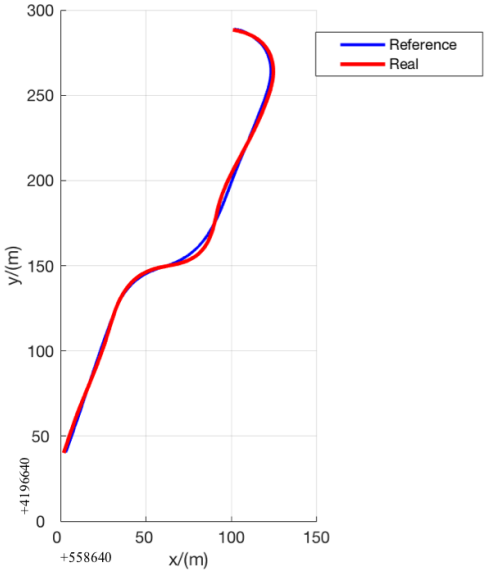

Figure 12.2 Trajectory tracking result

Figure 12.2 shows the X and Y tracking results. Numerically, the average error is 0.4944 meters, the max error is 1.252 meters. (The accuracy of DGPS reaches 0.01 meter).

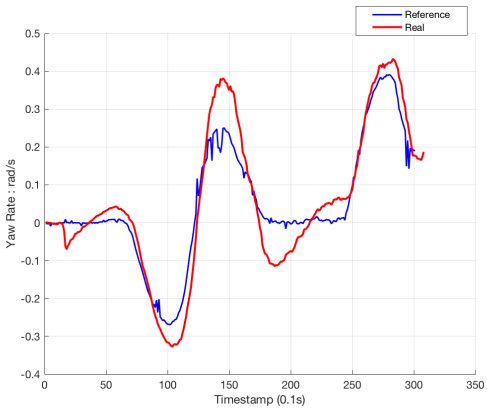

Figure 12.3 Yaw angle tracking result

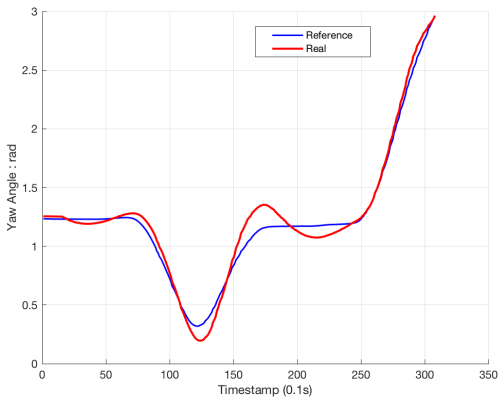

Figure 12.4 Yaw rate tracking result

The performance of trajectory II is obviously not as satisfactory as that of trajectory I. It has obvious larger overshoots on yaw angle and yaw rate at sharp curves, and thus bigger errors on X and Y.

The field test videos can be found here:
Field test videos.

VII. Future Work

1) Improvement of current work

There are some limitations of current work, and ways to improve current work are discussed in this section.

1.1) State Estimation

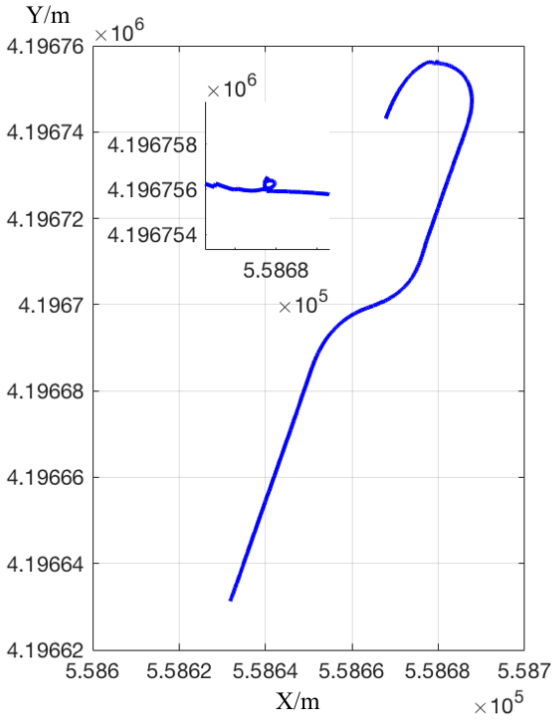

Figure 13. State estimation fault

As shown in the figure 13, some disturbance of the environment, e.g. a little heave on the ground, may exert bad results on state estimation. This may be avoided by using the dynamic model of vehicle.

1.2) Parameters Adjustment

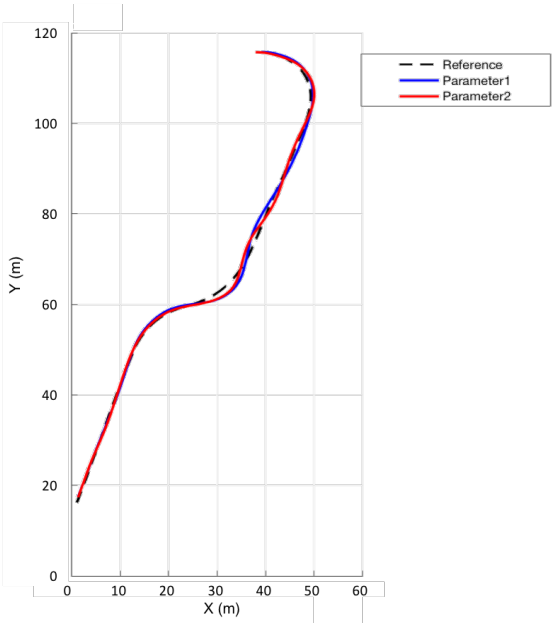

Figure 14.1 Tracking results comparison with different weight parameters

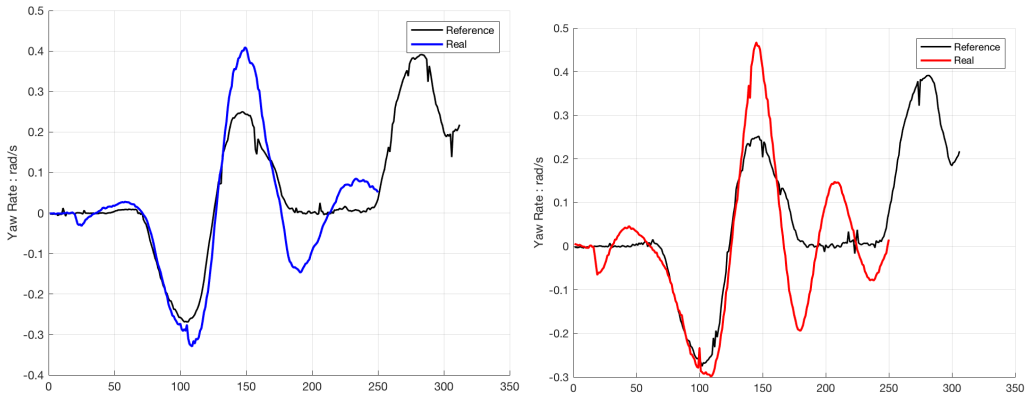

Figure 14.2 (left) Yaw rate tracking result of parameter1
Figure 14.3 (right) Yaw rate tracking result of parameter 2

As the non-linear system is approximately linearized, the effects that the shape of trajectory exerts on control system fail to be taken into account. Thus, MPC is very sensitive to curvature changes and overshoots are very common.

Different parameters of weight on inputs and states have different influence on the tracking performance. As shown in Figure 14, the performance of parameter1, with less penalty on X, Y, and yaw rate, gets smaller overshoots and less fluctuations. However, the improvement on tracking performances is not remarkable if we continue reducing the penalty on X, Y and yaw rate. Therefore, more research in deciding a better set of parameters for weight matrix is desired.

In addition, the length of prediction horizon and the working frequency of MPC are adjustable as well. Considering these parameters, the computationally expense should be also taken into account.

1.3) Matrix algorithm efficiency.

It's recorded that the time expense of LQR is less than 0.01 seconds, and the solving time of MPC is at an average of 0.03 seconds, among which matrix multiplications take up around 0.02 seconds. The expense of matrix multiplications hugely constrains the length of horizon, and thus the accuracy of tracking. A practicable way to increase the multiplication efficiency is using the sparse matrix. Besides, some other libraries have potential to improve matrix multiplication efficiency, e.g. Intel MKL matrix algorithm library.

2) Proposal of future work

While this report has presented the fulfillment of the fundamental trajectory-tracking function, many opportunities for extending the scope remain. Some of these directions with my personal research interest are presented as follow:

Robust MPC for Actuator-fault Tolerance

The object actuator fault-tolerant control (FTC) scheme is to maintain satisfactory performance for the controlled system even in the presence of faults. Some approaches has already been discussed in the last decade. However, among the researchers, most tackle the problem with reconfiguration separately, and are usually carried out on simulation examples. The experimental vehicle Lincoln MKZ with by-wire kits provides with proper research conditions.

Collision Avoidance and Stabilization for Autonomous Vehicles

Emergency scenarios may necessitate autonomous vehicle maneuvers up to their handling limits in order to avoid collisions. In these scenarios, vehicle stabilization becomes important to ensure that the vehicle does not lose control. Research integrates trajectory-tracking, vehicle stabilization, and collision avoidance and mediates potential conflicting objectives among them. This research interest can be conducted based on current work on the trajectory-tracking under a model predictive control framework.