LQG Steering Control for Auto Lane Changing

Cong Fu, Zixin Liu

September 24, 2020

Abstract

This project focuses on the LQG controller design methodology based on the lateral error dynamics of vehicle model and constant longitudinal velocity assumption. Also, a feedforward controller is used to improve the control performance. The proposed solution is then implemented in MATLAB/Simulink to control the vehicle model for tracking a predefined half-sine reference trajectory to mimic the real-world lane changing scenario. In this simulation, disturbance and sensor noises are also added to the system, and the control system still performances well showing the robustness and ability of noises rejection.

Keywords- lane changing, error dynamics, Kalman filter, feedforward control, LQR control

1 Introduction

Self-driving car is a hot topic in both automotive industry and research field. To fully takeover the control authority from drivers, self-driving cars should be able to handle every possible driving situations on the road, among which lane changing is of great importance, as it can serve as the primitive of the autonomous vehicle to perform other complicated operation such as cutting in, lane merging or entering highway[2]. To achieve auto lane changing, a well-designed control system is essential, including model analysis, motion planning, sensor filtering, controller design, etc. This project will focus on the model analysis and controller design using the discrete system control design methods, and in the meantime, some filtering technique will also be used to achieve noise suppression and obtain a clean signal for the feedback system.

2 Problem Formulation

To capture the vehicle dynamics, this project uses a 5 states bicycle model as the system plant. For lane changing problem, lateral dynamics should be emphasized, so we assume longitudinal velocity to be constant but with some small perturbation due to disturbance.

Also, we can simplify our control design by using a 4 states bicycle model which only captures the lateral dynamics of the vehicle. The control technique used in this project is LQR (linear quadratic regulator) which will drive all the system states back to the origin while minimizes a cost function. However, lane changing problem is a tracking problem. So, we have to change the realization of the system. For example, we can add the integral state into the dynamics which is the integration of the error between reference and the output, so that the system can track the reference with zero steady-state error when the regulator forces the integral state back to zero. Also, we can change all 4 states into error states, which means we consider the error dynamics of the system along a certain trajectory. In this project, error dynamics method is used to design the controller.

In real world application, sensors signals are always come with noise, and not all the states are measurable due to physical constraint or huge cost. So, some estimation techniques should be used to provide "clean" information for the feedback controller. This project uses a discrete Kalman filter as the state observer.

3 Vehicle Modeling

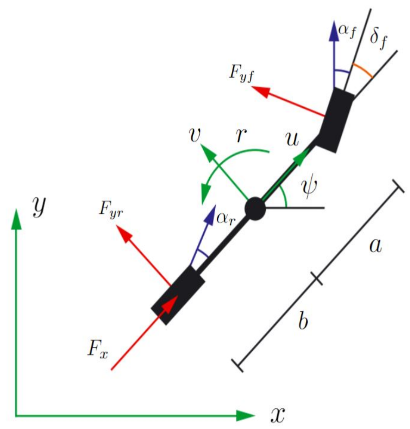

Figure 1: Vehicle bicycle model

The dynamics for a bicycle model, assuming constant longitudinal velocity v_x, are given by:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\phi} \\ \dot{v}_y \\ \dot{r} \end{bmatrix} = \begin{bmatrix} v_x cos(\phi) - v_y sin(\phi) \\ v_y cos(\phi) + v_x sin(\phi) \\ r \\ \frac{1}{m}(F_{yr} + F_{yf} cos(\delta) - m v_x r) \\ \frac{1}{I_z}(-bF_{yr} + aF_{yf} cos(\delta)) \end{bmatrix} \tag{1}$$

where δ is the front wheel angle, v_y is the lateral velocity, r is the yaw rate, F_{yf} and F_{yr} are the lateral tire force at the front and rear tires, a is the distance to the center of mass from the front axle, and b is the distance to the center of mass from the rear axle.

The Pacejka Tire model, sometimes referred to as the "Magic Tire Model", can be used to model the lateral forces as functions of the slip angles at the front and rear tires α_f and α_r is shown as follows:

$$F_{yf} = F_{zf} D sin(C arctan(B(1-E)\alpha_f + E acrtan(B\alpha_f))) \tag{2}$$
$$F_{yr} = F_{zr} D sin(C arctan(B(1-E)\alpha_r + E acrtan(B\alpha_r))) \tag{3}$$
$$F_{zf} = \frac{b}{L} mg \tag{4}$$
$$F_{zr} = \frac{a}{L} mg \tag{5}$$
$$\alpha_f = \delta - arctan(\frac{v_y + ar}{v_x}) \tag{6}$$
$$\alpha_r = -arctan(\frac{v_y - br}{v_x}) \tag{7}$$

Where F_{zf} and F_{zr} are the vertical tire forces at the front and rear tire, L is the wheelbase, and g is the acceleration of gravity. The remaining coefficients B, C, D, E are empirically found values. And for simplicity the tire dynamics are often assumed to be linear with respect to the slip angle $F_{yi} \approx C_{\alpha_i} \alpha_i$. The cornering stiffness C_{α_i} is defined as

$$C_{\alpha_i} = \frac{\partial F_{yi}}{\partial \alpha_i}\bigg|_{\alpha_i = 0} \tag{8}$$

The parameters of the vehicle we use is shown as follows:

Table 1: Typical Vehicle Parameter Table

m	$1500kg$	total mass of vehicle
L	$2.54m$	wheel base
a	$1.14m$	cg to front axle distance
b	$1.40m$	cg to rear axle distance
I_z	$2420.0kg \cdot m^2$	moment of inertia
C_{af}	$105440N/rad$	cornering stiffness(front)
C_{rf}	$85857N/rad$	cornering stiffness(rear)
g	$9.81N/kg$	gravitational parameter
v_x	$20m/s$	rear wheel linear speed

These are nonlinear differential equations, so they should be linearized along a nominal trajectory which is the real-time generated reference path by some motion planning techniques. The result of the linearization would be a time-variant linear system. Though this kind of systems can be controlled using finite horizon LQR, it will be less computational if we can use a pre-calculated feedback gain instead of updating it on-line. To explore the probability of doing so, we dig into the frequency response of the system at different points in the nominal trajectory generated by applying a constant steering command to the vehicle as shown in figure 2 .

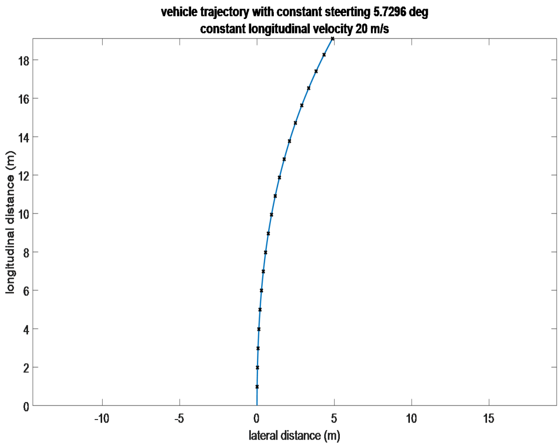

Figure 2: vehicle trajectory

We pick 20 points evenly located in the trajectory and linearize the nonlinear model at these points. Then we plot the bode plots of all these 20 linearized systems as shown in figure 3.

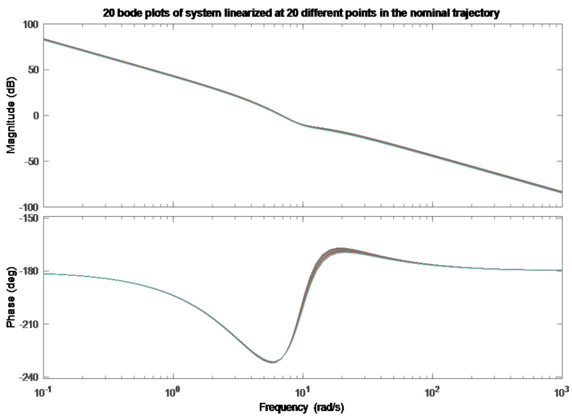

Figure 3: bode plots

We can see that all these 20 systems have very similar frequency response which means we can simplify this time-variant system into a time-invariant system with some modeling uncertainties. So, we choose the model linearized at point $[v_x\ 0\ 0\ 0\ 0]^T$ as our design model (10).

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\phi} \\ \dot{v}_y \\ \dot{r} \end{bmatrix} = \begin{bmatrix} 0 & 0 & -v_x sin\phi - v_y cos(\phi) & -sin\phi & 0 \\ 0 & 0 & -v_y sin(\phi) + v_x cos(\phi) & cos\phi & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & \frac{-v_x(C_{ar}+C_{af})}{m} & \frac{v_x(bC_{ar}-aC_{af})}{m} - v_x \\ 0 & 0 & 0 & \frac{v_x(bC_{ar}-aC_{af})}{I_z} & \frac{-v_x(b^2C_{ar}+a^2C_{af})}{I_z} \end{bmatrix} \begin{bmatrix} x \\ y \\ \phi \\ v_y \\ r \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ \frac{C_{af}}{m} \\ \frac{aC_{af}}{I_z} \end{bmatrix} \delta \tag{9}$$

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\phi} \\ \dot{v}_y \\ \dot{r} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 20 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & -6.3765 & -20 \\ 0 & 0 & 0 & 0 & -6.3080 \end{bmatrix} \begin{bmatrix} x \\ y \\ \phi \\ v_y \\ r \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 70.2921 \\ 49.6692 \end{bmatrix} \delta \tag{10}$$

Then we discretize this model using Zero-Order-Hold method:

$$\begin{bmatrix} x(k+1) \\ y(k+1) \\ \phi(k+1) \\ v_y(k+1) \\ r(k+1) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0.2 & 0.01 & 0 \\ 0 & 0 & 1 & 0 & 0.01 \\ 0 & 0 & 0 & 0.9362 & -0.2 \\ 0 & 0 & 0 & 0 & 0.9369 \end{bmatrix} \begin{bmatrix} x(k) \\ y(k) \\ \phi(k) \\ v_y(k) \\ r(k) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0.7029 \\ 0.4967 \end{bmatrix} \delta(k) \tag{11}$$

For the 4 states error dynamics model, it is time-invariant once we assume constant longitudinal velocity, which is shown as follows:

$$\frac{d}{dt}\begin{bmatrix} e_y \\ \dot{e}_y \\ e_\phi \\ \dot{e}_\phi \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -\frac{C_{af}+C_{ar}}{mu} & \frac{C_{af}+C_{ar}}{m} & \frac{bC_{af}-aC_{ar}}{mu} \\ 0 & 0 & 0 & 1 \\ 0 & \frac{bC_{af}-aC_{ar}}{I_z u} & \frac{aC_{af}-bC_{ar}}{I_z} & -\frac{a^2C_{af}+b^2C_{ar}}{I_z u} \end{bmatrix} \begin{bmatrix} e_y \\ \dot{e}_y \\ e_\phi \\ \dot{e}_\phi \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{C_{af}}{m} \\ 0 \\ \frac{aC_{af}}{I_z} \end{bmatrix} \delta_f \tag{12}$$

where

$$e_y = \frac{y - y_d}{cos\phi} \tag{13}$$

$$\dot{e}_y = v + ue_\phi \tag{14}$$

$$e_\phi = \phi - \phi_d \tag{15}$$

$$\dot{e}_\phi = r - r_d \tag{16}$$

This model has some inherent benefits. First, we can only control the states that are important for path tracking for which the lateral displacement and the heading angle are the states we want to control. Second, the control objective is to make the states to zero, which is compatible with the control objective of LQR. Third, this model is a linear time invariant model, which means that we can, instead of using finite horizon LQR, we can calculate the feedback gain off-line by infinite horizon LQR. After substituting all the parameter values in the above model, what we get is the following:

$$\frac{d}{dt}\begin{bmatrix} e_y \\ \dot{e}_y \\ e_\phi \\ \dot{e}_\phi \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -6.3766 & 127.5313 & -0.0001 \\ 0 & 0 & 0 & 1 \\ 0 & -0.0000 & 0.0007 & -6.3080 \end{bmatrix} \begin{bmatrix} e_y \\ \dot{e}_y \\ e_\phi \\ \dot{e}_\phi \end{bmatrix} + \begin{bmatrix} 0 \\ 70.2933 \\ 0 \\ 49.6701 \end{bmatrix} \delta_f \tag{17}$$

Also, we discretize the model using Zero-Order-Hold method:

$$\begin{bmatrix} e_y(k+1) \\ \dot{e}_y(k+1) \\ e_\phi(k+1) \\ \dot{e}_\phi(k+1) \end{bmatrix} = \begin{bmatrix} 1 & 0.0049 & 0.0016 & 0 \\ 0 & 0.9686 & 0.6276 & 0.0016 \\ 0 & 0 & 1 & 0.0049 \\ 0 & 0.0000 & -0.0000 & 0.9690 \end{bmatrix} \begin{bmatrix} e_y(k) \\ \dot{e}_y(k) \\ e_\phi(k) \\ \dot{e}_\phi(k) \end{bmatrix} + \begin{bmatrix} 0.0009 \\ 0.3460 \\ 0.0006 \\ 0.2445 \end{bmatrix} \delta_f(k) \tag{18}$$

4 Estimator Design

To control a vehicle, feedback should be applied, which requires the system should have the ability to measure the vehicle states.

(x, y) coordinate can be obtained by GPS, and Yaw angle can also be measured by gyroscope directly. However, it is hard to measure the lateral velocity and yaw rate of a vehicle. Usually, yaw rate can be estimated from the data of yaw angle, but noise problem retains. And lateral velocity can be measured using real-time RTK from OxTS, but it is hard to be used in commercial vehicles due to the huge price. So, this project uses discrete Kalman filter to estimate these two states from the bicycle model.

Kalman filter is an optimal linear estimator assuming the measurement and process both contain gaussian noises. The update scheme is as follow:

$$\bar{\mu} = A_t \mu_{t-1} + B_t u_t \tag{19}$$
$$\bar{\Sigma_t} = A_t \Sigma_{t-1} A_t^T + R_t \tag{20}$$
$$K_t = \bar{\Sigma_t} C_t^T (C_t \bar{\Sigma_t} C_t^T)^{-1} \tag{21}$$
$$\mu_t = \bar{\mu_t} + K_t (z_t - C_t \bar{\mu_t}) \tag{22}$$
$$\Sigma_t = (I - K_t C_t) \bar{\Sigma_t} \tag{23}$$

We choose R matrix as the covariance matrix of the measurement noise. In this project, we measure three states, x coordinate, y coordinate and yaw angle. The sensor noises are set to be the following:

$$\mu_x = 0 \tag{24}$$
$$\sigma_x = 0.0011 \tag{25}$$
$$\mu_y = 0 \tag{26}$$
$$\sigma_y = 0.0011 \tag{27}$$
$$\mu_{yaw} = 0 \tag{28}$$
$$\sigma_{yaw} = 2.1 \times 10^{-6} \tag{29}$$

In practice, we can first sample the noises signals and calculate the covariance matrix of them. In this project, the covariance matrix is

$$\begin{bmatrix} 0.001119762 & -0.000021168 & 0.000000587 \\ -0.000021168 & 0.00114099 & -0.000000524 \\ 0.000000586 & -0.00000052 & 0.000002125 \end{bmatrix}$$

After setting the R matrix as the above matrix and assign a relatively small value to Q, the result of Kalman filter test is shown in figure 4. The actual trajectory is generated by the feedforward control input which will be talked about in the next section. The blue dash line is the direct measurement using GPS and gyroscope, so it is very noisy. After applying Kalman filter, the states we get are much more "clean" and close to the actual trajectory.

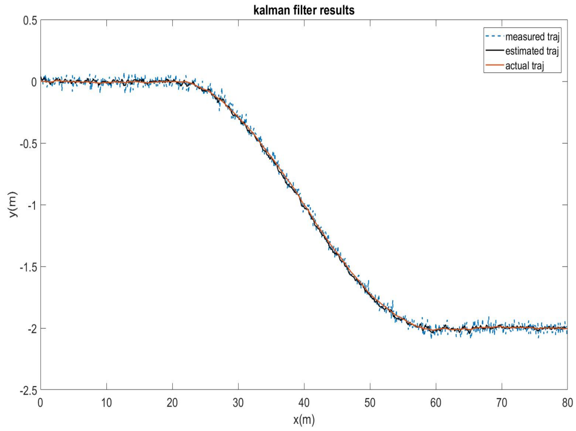

Figure 4: kalman filter result

5 Controller design

In this project we implement the LQG control approach. LQG is the combination of a Kalman filter(also known as LQE) with a linear quadratic regulator(LQR). This control method concerns uncertain linear systems which have incomplete state information(not all states can be measured) and disturbed by additive white Gaussian noise in both sensor and process model. The whole control block diagram is shown as follows:

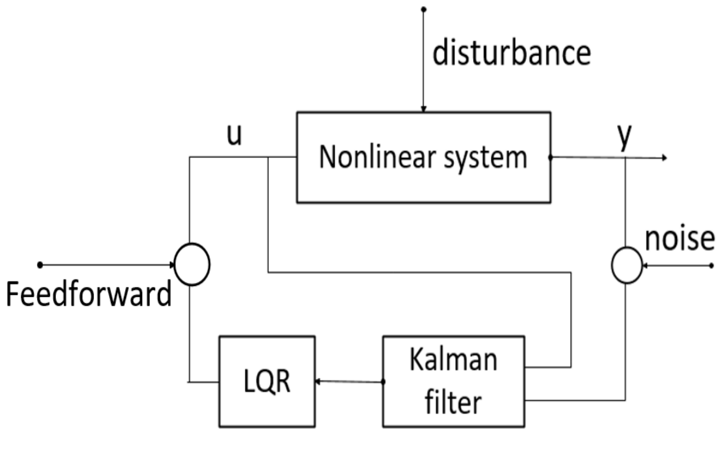

Figure 5: Control block diagram

5.1 Feedforward Control

Together with states feedback, this project also uses feedforward to improve the control performance. Feedforward control directly depends on the reference input. Doing so allows the control signal to response more rapidly to the command so that the system might be speed up. In this project, the feedforward is calculated based on the curvature of the reference trajectory.

$$\delta_{ff} = \frac{L}{R} + K_v \frac{u^2}{R} \tag{30}$$

Where,

$$K_v = \frac{mb}{LC_{\alpha f}} - \frac{ma}{LC_{\alpha r}} \tag{31}$$

The control result is shown in figure[]. With this feedforward controller, though there still is an obvious tracking error, the vehicle can response to the reference correctly. Then, in the following section, we will design a LQR controller to correct this error.

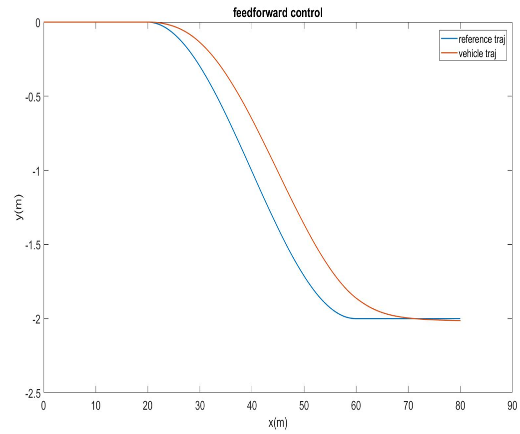

Figure 6: Control block diagram

5.2 Infinite Horizon LQR Control

The controller is designed based on the 4 states error dynamics model because what we care about in the lane changing control is the lateral dynamics (the errors along lateral direction and yaw angle). So the essence of this problem is to design a regulator to drive the error states back to the origin. Linear quadratic regulator (LQR) approach is naturally considered to be a very suitable tool. To control a vehicle to track a predefined reference path with constant longitudinal velocity, the cost function is formulated as the following:

$$J = \int^{\infty} (e^T Q e + v_x^T R u) \; dt \tag{32}$$

where Q is the positive definite matrix which penalize the error states and R is the semi-positive definite matrix which penalize the steering input. The main idea here is that we want the error states converge to zero as quickly as possible while the input value should be as small as possible. LQR provides a method to deal with the trade off between these two requirements and deals to the optimal control gain.

The optimal control minimizing 32 is given by

$$u(t) = -Kx(t), K = R^{-1} B^T P \tag{33}$$

The matrix P is a positive semidefinite solution to the Discrete Algebraic Riccati Equation(ARE):

$$P = A^T P A - (A^T P B)(R + B^T P B)^{-1}(B^T P A) + Q \tag{34}$$

6 Results and Analysis

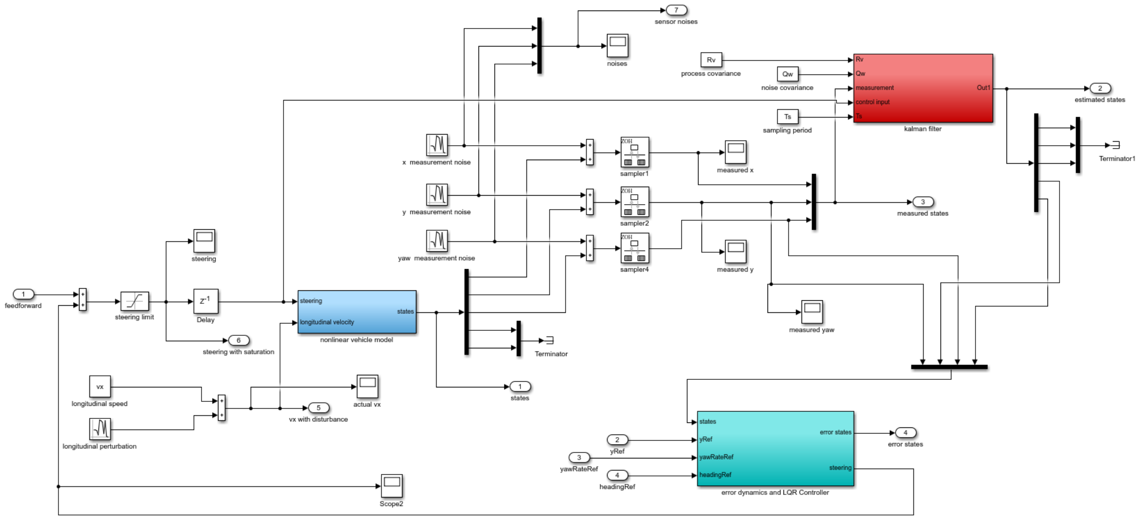

Figure 7: Simulink model of the whole control system

To verify the above designs, this project builds a Simulink/MATLAB model as shown in figure 7. The sampling rate is 0.005s, disturbance and noises are shown in figure 9 and 8.

Though we assume constant longitudinal velocity, it is, actually, subject to some perturbations such that it varies from the assumed value. To deal with this kind of perturbation, for example, we can try to model its dynamics and do a disturbance estimation [3]. But because the LTI control model used above will change according to different longitudinal velocities, in this project, the perturbation is treated as the model uncertainty which can also test the robustness of the controller. For the measurement noises, we set the noises of x and y directions are the same because they are both measured by GPS.

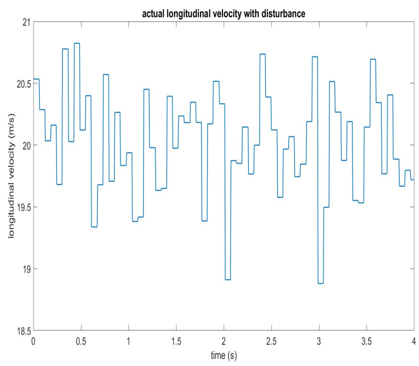

Figure 8: Longitudinal velocity with disturbance

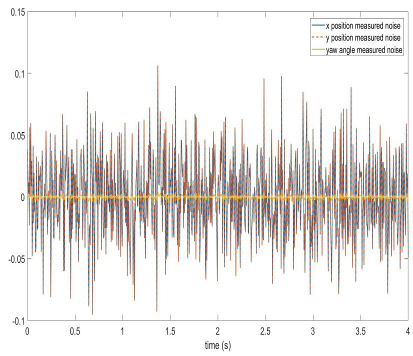

Figure 9: measurement noises

Using the discrete LQR technique with the Q and R matrices below, the feedback we obtained is:

$$k = [21.7993\ 2.4137\ 15.6107\ 0.4730] \tag{35}$$

$$Q = \begin{bmatrix} 100 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{36}$$

$$R = 10 \tag{37}$$

For the Kalman filter, we choose the initial uncertainty matrix, process covariance matrix, and noise covariance matrix as the following:

$$\Sigma_0 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{38}$$

$$Q_v = 1 \times 10^{-4} \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{39}$$

$$R_w = \begin{bmatrix} 0.001119762 & -0.000021168 & 0.000000587 \\ -0.000021168 & 0.00114099 & -0.000000524 \\ 0.000000586 & -0.00000052 & 0.000002125 \end{bmatrix} \tag{40}$$

With all the parameters above, the simulation results are shown in the figures below. In figure 10, we can see that the noises are suppressed by the Kalman filter such that the estimated trajectory does not vary from the actual one too much as the measured trajectory does. These estimated states are used as the input of the LQR controller. By applying feedforward and LQR together, the vehicle can track the reference lane changing reference trajectory with small lateral position and yaw angle errors during the transient, but it converges very fast and smoothly which is good for the passengers' experience as shown in figure 11.

Also, in the lane changing situation, large steering should be avoided to ensure the comfort of the passengers. In this project, a steering limit is set for the control command as shown in figure 12. To avoid saturation which might cause oscillation or even unstability problem, we penalize the control input with a relatively "large" matrix R. However, though the steering does not saturated, it contains a lot high frequency components which means the steering wheel is turned very frequently. This problem is due to the sensor noises. Kalman filter suppresses the majority of the noises, but the black line in figure 10 is still not smooth enough, as a result, the controller would try to minimize the error between this not "clean" estimation and the reference and then lead to this "noisy" steering command. Figure 13 shows the steering command when there are no sensors noises.

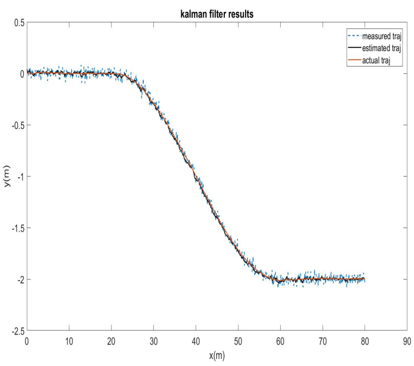

Figure 10: Kalman filtering results

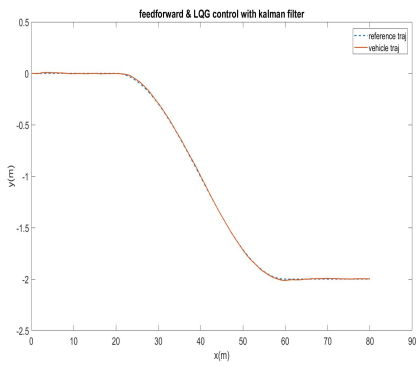

Figure 11: LQR and feedforward control

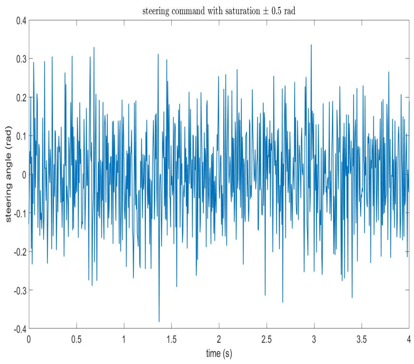

Figure 12: Steering Command with sensor noise

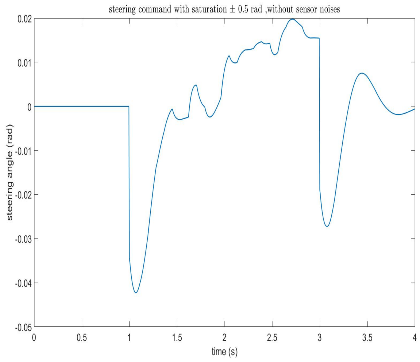

Figure 13: Steering Command without sensor noises

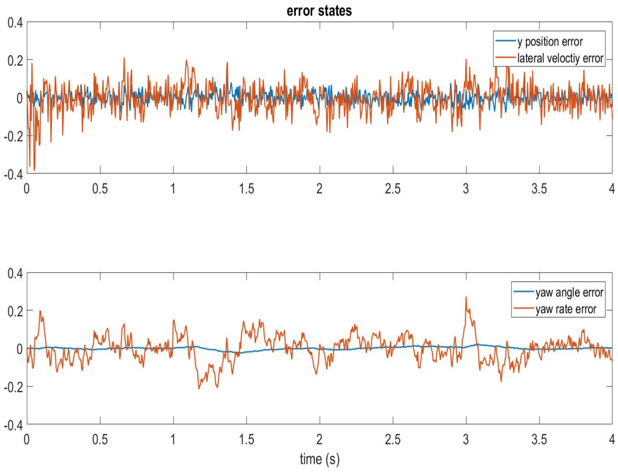

Figure 14: Error states

7 Conclusion

This project design a auto lane changing controller for the autonomous vehicle using both feedforward and LQR control, what's more, Kalman filter is applied to deal with the unmeasurable states and noise suppression. The control result is acceptable. To improve the control performance, there are several things can be done. First, design a good sensor filter to filter out the high frequency noises of the measurement before letting the signals enter the Kalman filter. Second, a more feasible reference trajectory can be find by applying some path planning techniques instead of just using a half-sine track.

References

[1] Phillips, Charles L., and H. Troy Nagle. Digital control system analysis and design. Prentice Hall Press, 2007.

[2] Hatipoglu, Cem, Umit Ozguner, and Keith A. Redmill. "Automated lane change controller design." IEEE transactions on intelligent transportation systems 4.1 (2003): 13-22.

[3] Kravaris, Costas, et al. "Nonlinear observer design for state and disturbance estimation." Systems and Control Letters 56.11-12 (2007): 730-735.