Syrian Arab Republic
Ministry of Higher Education
Manara University
Faculty of Engineering
Department of Robotics and Intelligent Systems

Path Tracking Control System

Edited by:
Ali K. Deeb
Saleh A. Rabea

Supervised by:
Dr. Hasan AL Ahmad
Dr. Samer Suleiman
Eng. Wassim Ahmad

Date
15/July/2023

Contents

1-Introduction 1

1-1- Related Works 1

2-Lateral Kinematic Model 2

2-1- Assumptions 2

2-2- Lateral Kinematic Equations 3

2-3- Remark 4

2-3-1- Right Turn 4

2-3-2- Left Turn 4

3-Controlling Strategy 5

3-1- Orientation Error Handling 5

3-2- Cross Track Error Handling 8

3-3- Orientation Error & Cross Track Error Handling 11

4-Simulation 14

4-1- Matlab and Simulink 14

4-1-1- Matlab 14

4-1-2- Simulink 14

4-2- Block Diagram 14

4-2-1- Road Block 15

4-2-2- Sensors Block 15

4-2-3- Controller Block 16

4-2-4- Kinematic Model Block 16

5-Conclusion 19

6-References 20

Figures Index

Figure (2–1) Kinematics of lateral vehicle motion 2
Figure (2–2) Turning geometry 4
Figure (3–1) orientation error & CTE 5
Figure (3–2) orientation-based control 6
Figure (3–3) Orientation error versus time 6
Figure (3–4) Cross track error versus time 6
Figure (3–5) steering angle versus time 6
Figure (3–6)) orientation-based control failure 7
Figure (3–7) orientation error versus time 7
Figure (3–8) cross track error versus time 7
Figure (3–9) steering angle versus time 8
Figure (3–10) CTE-based control 8
Figure (3–11) orientation error versus time 9
Figure (3–12) cross track error versus time 9
Figure (3–13) steering angle versus time 9
Figure (3–14) CTE-based control failure 10
Figure (3–15) orientation error versus time 10
Figure (3–16) cross track error versus time 10
Figure (3–17) steering angle versus time 11
Figure (3–18) orientation & cross track based control 11
Figure (3–19) orientation error versus time 12
Figure (3–20) cross track error versus time 12
Figure (3–21) steering angle versus time 12
Figure (3–22) control strategy 13
Figure (4–1) block diagram 15
Figure (4–2) road block 15
Figure (4–3) sensors block 16
Figure (4–4) controller block 16
Figure (4–5) kinematic model block 17
Figure (4–6) X coordinate block 17
Figure (4–7) Y coordinate block 18
Figure (4–8) orientation angle block 18

Tables Index

Table(2–1) symbols' indications 2

Abstract

This project presents a steering control method for driving an autonomous vehicle to track a smooth and continuous desired path. First, it provides a kinematic model of a vehicle's motion based on its geometry without considering forces or torques, which is valid for low-speed motion, large path curvature, and zero inclination and bank angles. The project adopts a bicycle model in which the two left and right front wheels are represented by a single central wheel and so the rear wheels. The project also proposes a control strategy that combines two methods for handling two types of errors: The orientation error, and the cross-track error CTE. The first control method uses a geometric procedure to relate the steering angle with the orientation error. The second control method uses a Proportional-Integral-Derivative (PID) controller to modify the input signal (steering angle) according to the CTE. The strategy makes the velocity vector of vehicle's center of gravity parallel to the road based on the orientation error first, then rotates it towards the centerline based on the CTE. This project contains a Matlab simulation of the system that shows the results of the controlling methods and the motion of the vehicle. The project concludes that the combined method solves the problems of the orientation-error-based controller and the CTE-based controller.

Introduction

Autonomous vehicles (AVs) have been a topic of intense research and development in recent years. The ability of these vehicles to operate without human intervention has the potential to revolutionize transportation systems by improving traffic flow, increasing accessibility for people who cannot drive and improving safety on the roads. One of the key features of AVs is their ability to perform path tracking, which involves the vehicle staying on its designated path without human assistance. Path tracking is a critical component of autonomous driving, as it ensures that the vehicle stays on course. To achieve this level of autonomy, autonomous vehicles rely on a variety of technologies, including sensors such as cameras, lidar, and radar, as well as advanced algorithms and multi-disciplinary approaches in areas such as robotics, artificial intelligence, computer vision, software engineering, and control systems. One critical aspect of autonomous vehicle control is the ability to accurately and safely adjust the vehicle's lateral motion. This project contributes to this area of research by proposing a control method for handling two types of errors that can affect the vehicle's position and orientation on the road. This control method can be integrated into autonomous vehicle systems to ensure accurate and safe control of the vehicle's lateral motion. [1]

1-1-Related Works

1. “A Nested PID Steering Control for Lane Keeping in Vision Based Autonomous Vehicles” by Riccardo Marino and others. This paper proposes a nested PID control method to perform path following in the case of roads with uncertain curvature. [1]
2. “Design and Testing of a Controller for Autonomous Vehicle Path Tracking Using GPS/INS Sensors” by J. Kang and others. In this paper a steering controller, which uses finite preview optimal control methods, is proposed to control the measured lateral offset, the yaw angle and their derivatives. [2]
3. “Combined Automatic Lane Keeping and Driver’s Steering Through a 2-DOF Control Strategy” by V. Cerone and others. A closed loop control strategy which feeds back the lateral offset is proposed: an automatic lane keeping is combined with the driver’s steering with no need of switching strategies between the driver and the lane keeping control. [3]
4. “Experimental results in vision-based lane keeping for highway vehicles” by V. Cerone and others. In this paper a control system based on the loop shaping technique is tested by experiments using feedback from the lateral offset. [4]
5. “Predictive Active Steering Control for Autonomous Vehicle Systems” by P. Falcone and others. A model predictive control approach is followed: the controlled outputs are the lateral offset, the yaw rate and the yaw angle; the controller is designed both on a nonlinear and a linear vehicle model using lateral and longitudinal vehicle speed, yaw angle and yaw rate measurements. [5]
6. “Application of an Optimal Preview Control for Simulation of Closed-Loop Automobile Driving” by C.C. MacAdam. A model predictive steering controller is used to emulate the driver behavior in the CarSim environment: it is designed on the basis of a simplified linear model and on longitudinal and lateral speed, yaw angle and yaw rate measurements to predict the error with respect to a given target path. [6]

Lateral Kinematic Model

2-1- Assumptions

Kinematic model of the vehicle provides a mathematical representation of the motion of the vehicle based on its geometry only without considering forces or torques. Such model is considered to be valid under the assumptions of low-speed motion and large path curvature, where the velocity vector of each wheel is in the direction of the wheel. This is reasonable because the lateral acceleration $a_y = \frac{V^2}{R}$,where R is the instantaneous radius of curvature, and V is the speed of the vehicle, is negligible ($a_y < 0.2\mathrm{m.s^{-2}}$). Hence, the slip angle of each wheel, which is the angle between wheel's direction and its velocity vector, is also negligible since it is positively correlated with the lateral acceleration. Further assumptions, the path which the vehicle has to follow is considered to be smooth and continuous. In addition, the bank angle and inclination angle are considered to be zero. In this project, a bicycle model is adopted in which the two left and right front wheels are represented by a single central wheel and so the rear wheels as shown in Figure (2-1). [7]

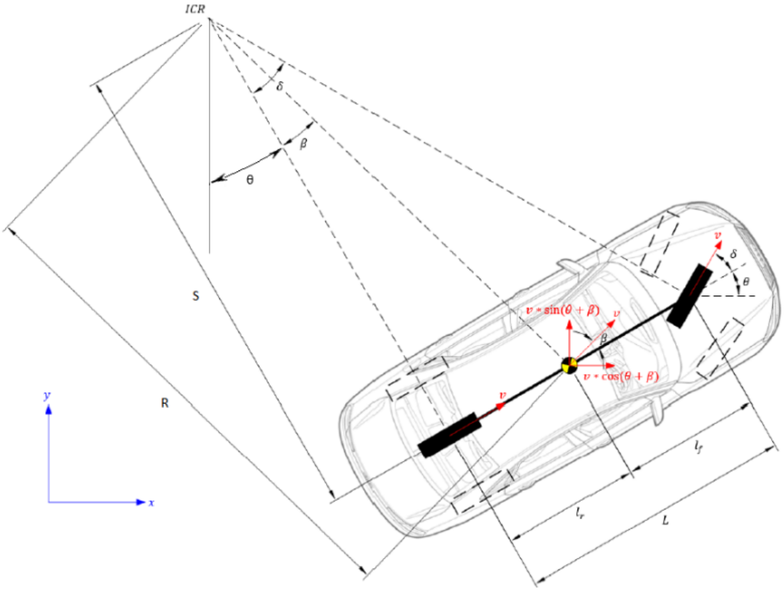

Figure (2–1) Kinematics of lateral vehicle motion [8]

Table(2–1) symbols' indications

L	The distance between front and rear axles.
l_f	The distance between front axle and center of gravity CG.
l_r	The distance between rear axle and CG.
θ	The orientation angle of the vehicle.
v	Velocity vector of CG.
ICR	Instantaneous center of rolling.
R	Instantaneous radius of rolling.
S	The distance between ICR and the vehicle's longitudinal axis.
δ	The steering angle of the front wheel.
β	The angle between velocity vector of CG and the vehicle's longitudinal axis. (Slip angle).

2-2-Lateral Kinematic Equations

The kinematic equations are relations between the system input and the system output under the assumptions discussed earlier. Assuming the velocity to be constant, the input of this system is the steering angle δ and the outputs are the position coordinates X, Y and the orientation θ. These outputs are related to input signal as follow:

The velocity of the CG is represented by

$$\vec{v} = \overrightarrow{v_x} + \overrightarrow{v_y} = \vec{\dot{X}} + \vec{\dot{Y}} \qquad (1)$$

$$\dot{X} = v.\cos(\beta + \theta) \qquad (2)$$

$$\dot{Y} = v.\sin(\beta + \theta) \qquad (3)$$

Where $\dot{X}$, $\dot{Y}$ denotes the global X, Y components of velocity.

When the vehicle moves on any arbitrary path, it can be considered to be rotating instantaneously around its ICR. Thus, the instantaneous angular velocity of the vehicle, which describes the change of orientation of the vehicle, can be written as:

$$\dot{\theta} = \frac{v}{R} \qquad (4)$$

From the vehicle's geometry,

$$\cos(\beta) = \frac{S}{R} \qquad (5)$$

$$\tan(\delta) = \frac{L}{S} \qquad (6)$$

Substituting S from equation (5) into (6),

$$R = \frac{L}{\tan(\delta) \times \cos(\beta)} \qquad (7)$$

Substituting R into equation (4),

$$\dot{\theta} = v \times \frac{\tan(\delta) \times \cos(\beta)}{L} \qquad (8)$$

Thus, the kinematic model of the vehicle's lateral motion is

$$\dot{X} = v.\cos(\beta + \theta)$$

$$\dot{Y} = v.\sin(\beta + \theta)$$

$$\dot{\theta} = v \times \frac{\tan(\delta) \times \cos(\beta)}{L}$$

Which represent time changes in position and orientation of the vehicle in terms of the steering angle δ , the slip angle β, and the constant velocity of the CG v. However, the only input of the system is the steering angle δ. The slip angle β can be written in terms of δ as follows: [7] [8]

$$\tan(\beta) = \frac{l_r}{S} \qquad (9)$$

Using equation (6):

$$\beta = \tan^{-1}\left(\frac{l_r \times \tan(\delta)}{L}\right) \qquad (10)$$

2-3-Remark

It should be mentioned that the steering angles of the left and right wheels will not be exactly the same since the radius of the path of each wheel is different.

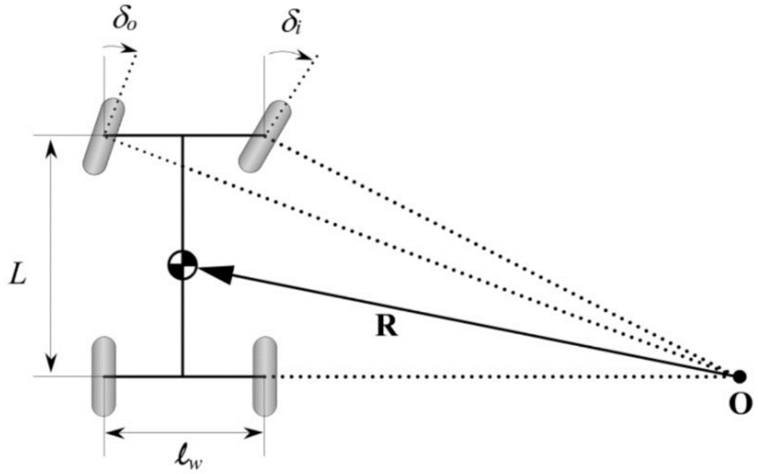

Figure (2–2) Turning geometry

If the steering motor is mounted on the right wheel, then a relation between the bicycle model steering angle δ and the right steering angle δ_r should be established considering that the reference of each angle is the longitudinal axis of the vehicle and the positive direction is counter-clockwise. There are two cases to discuss, right turn case and left turn case. [7]

2-3-1-Right Turn

In this case, the right wheel is the inner wheel $\delta_r = \delta_i$ and has a negative direction.

$$\tan(\delta) = -\frac{L}{S}$$

$$\tan(\delta_r) = -\frac{L}{S - \frac{l_w}{2}}$$

$$\delta_r = \tan^{-1}\left(\frac{L}{\frac{L}{\tan(\delta)} + \frac{l_w}{2}}\right) \qquad (11)$$

2-3-2-Left Turn

In this case, the right wheel is the outer wheel $\delta_r = \delta_o$ and has a positive direction.

$$\tan(\delta) = \frac{L}{S}$$

$$\tan(\delta_r) = \frac{L}{S + \frac{l_w}{2}}$$

Hence, as concluded in equation (11), the steering angle of the right wheel remains as follows

$$\delta_r = \tan^{-1}\left(\frac{L}{\frac{L}{\tan(\delta)} + \frac{l_w}{2}}\right)$$

Controlling Strategy

The controlling strategy used in this project depends on handling two types of errors. The first type is orientation error e_{θ}, which indicates the difference between the orientation of the vehicle and the orientation of the path's tangent at the point corresponding to the vehicle's position. Handling this type of error leads the CG's velocity vector to be parallel to the path's tangent. The second type is the cross-track error CTE, which is the distance between the vehicle's CG and the desired path. Handling this type of error rotates the CG's velocity vector toward the path instantaneously. The less these two errors, the more accurate the tracking.

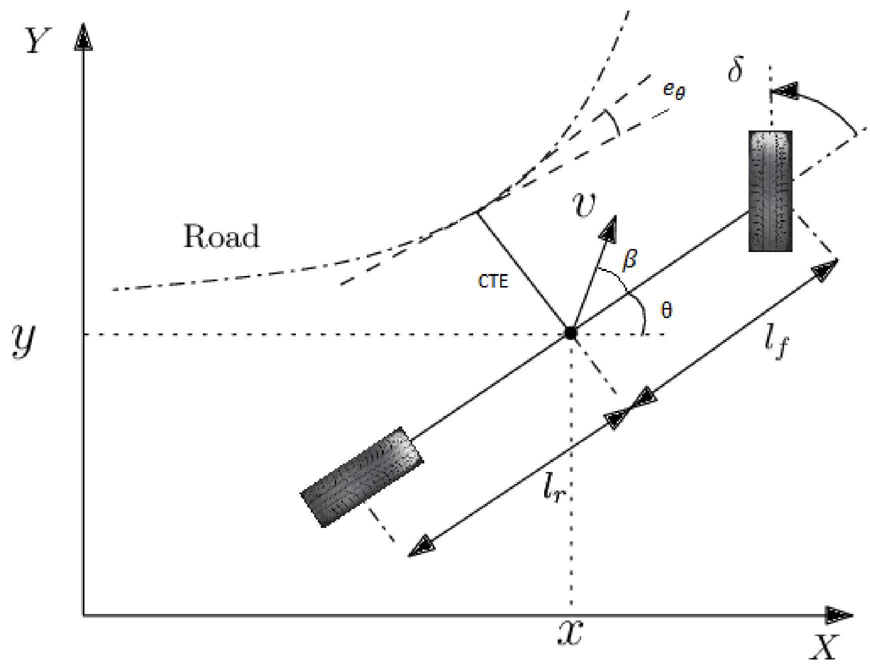

Figure (3–1) orientation error & CTE [9]

3-1-Orientation Error Handling

The orientation error as discussed earlier indicates the difference between the vehicle's orientation and the path's tangent orientation. To make the vehicle track a path, the velocity vector of CG should always be tangent to the path. To make this possible, the angle between the velocity vector and the vehicle's longitudinal axis β should be equal to the angle between the path's tangent and the same axis which is the orientation error itself e_{θ}. Moreover, the steering angle δ which represent the input signal should every instant provide the needed slip angle which is β. Thus, using equation (10)

$$\beta = \tan^{-1}\left(\frac{l_r \times \tan(\delta)}{L}\right)$$

$$\delta = \tan^{-1}((L * \frac{\tan(\beta)}{lr}) \qquad (12)$$

and knowing that

$$\beta = e_{\theta}$$

The needed input signal is

$$\delta = \tan^{-1}((L * \frac{\tan(e_{\theta})}{lr})$$

The result of this approach is shown in the following figures where the path is sinusoidal.

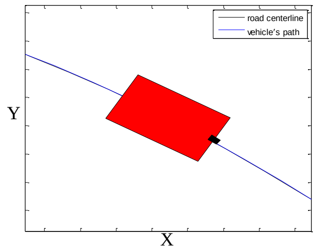

Figure (3–2) orientation-based control

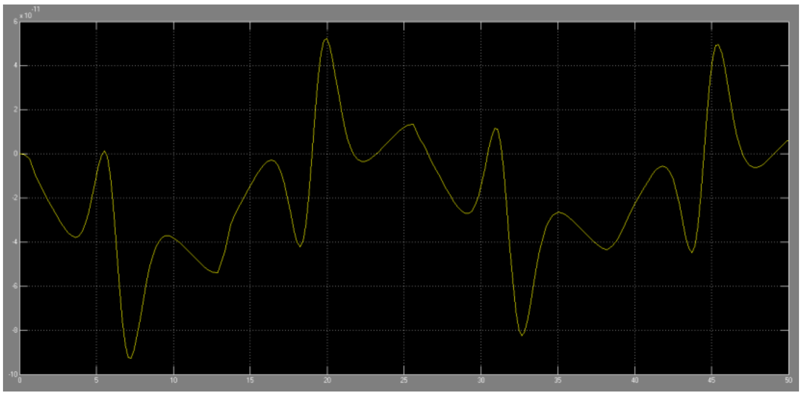

Figure (3–3) Orientation error versus time

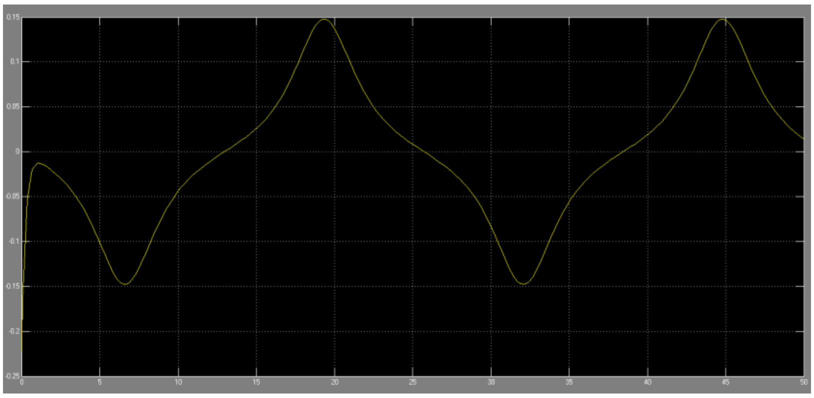

Figure (3–4) Cross track error versus time

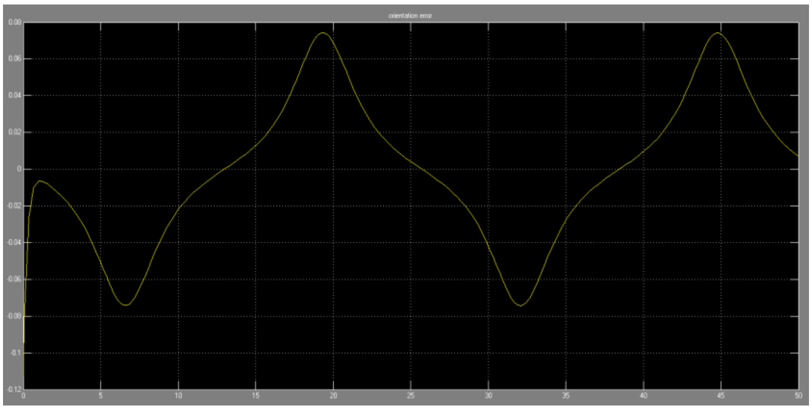

Figure (3–5) steering angle versus time

The main disadvantage of this method is considering the orientation of the vehicle only. When the velocity vector of the CG is parallel to the path the error will be zero and the vehicle will go straight forward even if there is a distance between the CG and the path. So, the vehicle may take a path parallel to the road as shown in the figure (3-6).

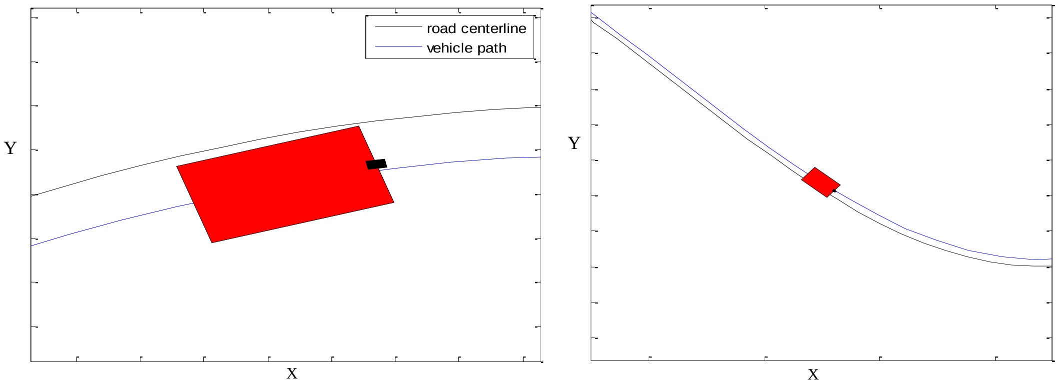

Figure (3–6)) orientation-based control failure

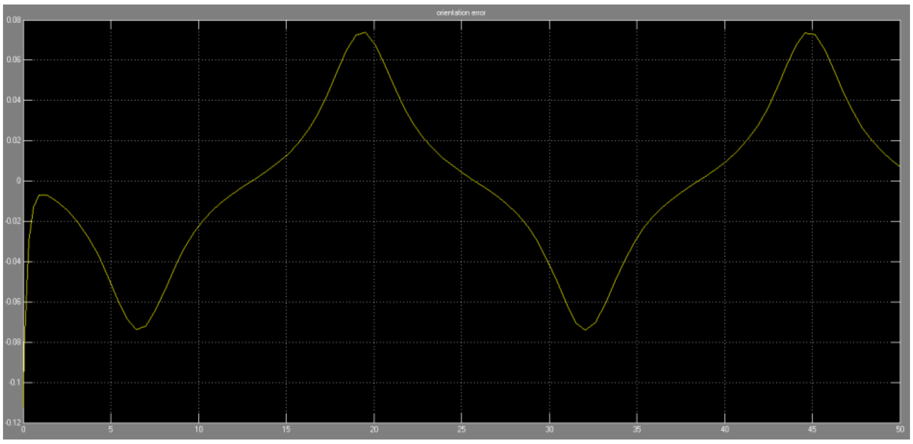

Figure (3–7) orientation error versus time

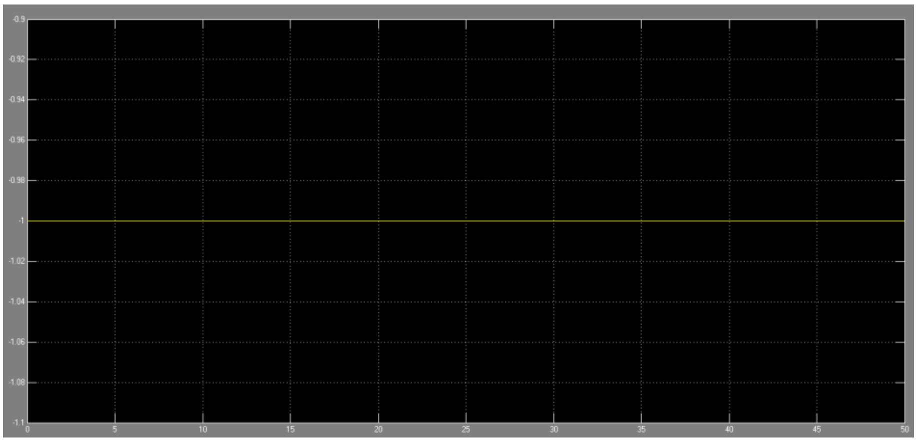

Figure (3–8) cross track error versus time

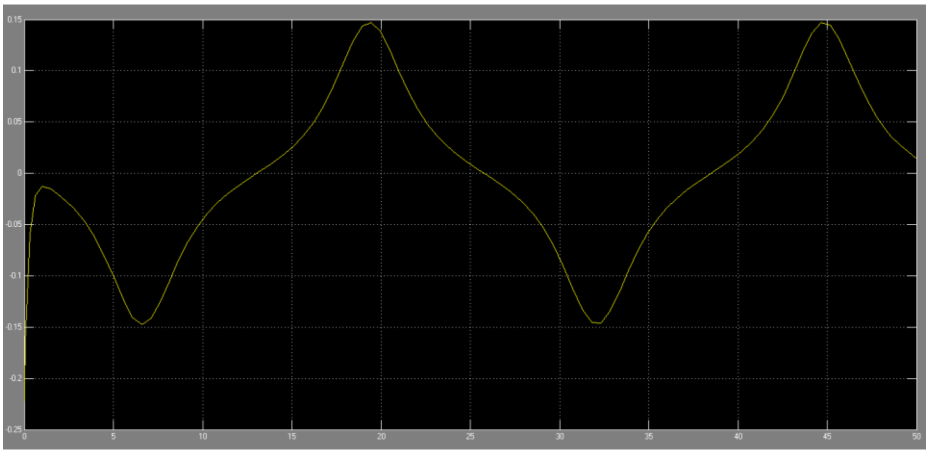

Figure (3–9) steering angle versus time

3-2-Cross Track Error Handling

CTE as discussed earlier indicates the distance between the vehicle's CG and the path. The vehicle should always return to the path at a speed proportional to CTE. The larger the CTE, the greater the speed of return. In addition, the vehicle should return to the path smoothly to avoid over crossing the path and missing the track again. Furthermore, due to safety requirements, the vehicle should quickly go toward the path determined especially in cases that the initial position of the vehicle is far from the path. For those requirements, a Proportional-Integral-Derivative PID controller is used. Moreover, from a practical point of view, the integral term of the controller helps in amplifying small error signals by accumulating this error so it can affect low-sensitive actuators. The results of applying this controlling method are shown in figures (3-10), (3-11), (3-12), (3-13) using a sinusoidal path. [10]

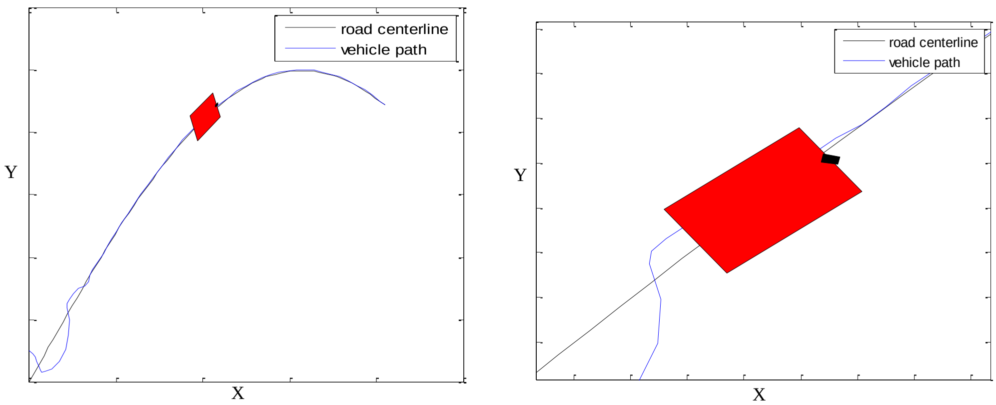

Figure (3–10) CTE-based control

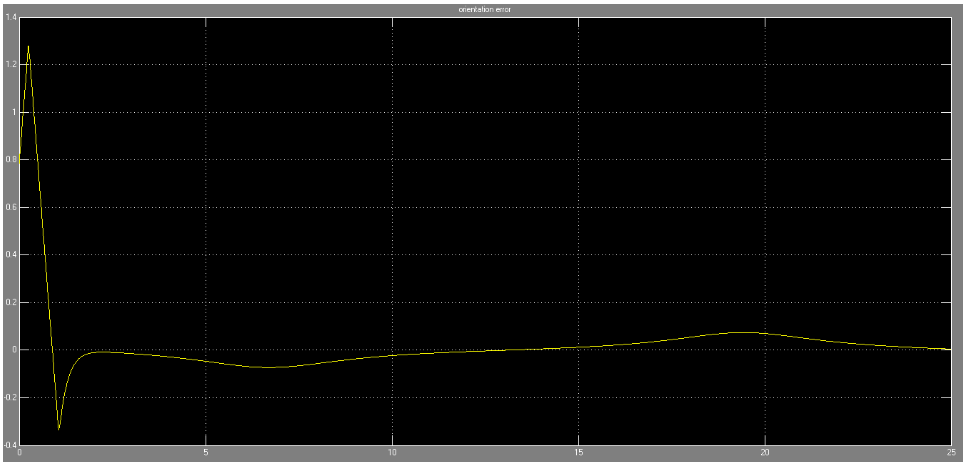

Figure (3–11) orientation error versus time

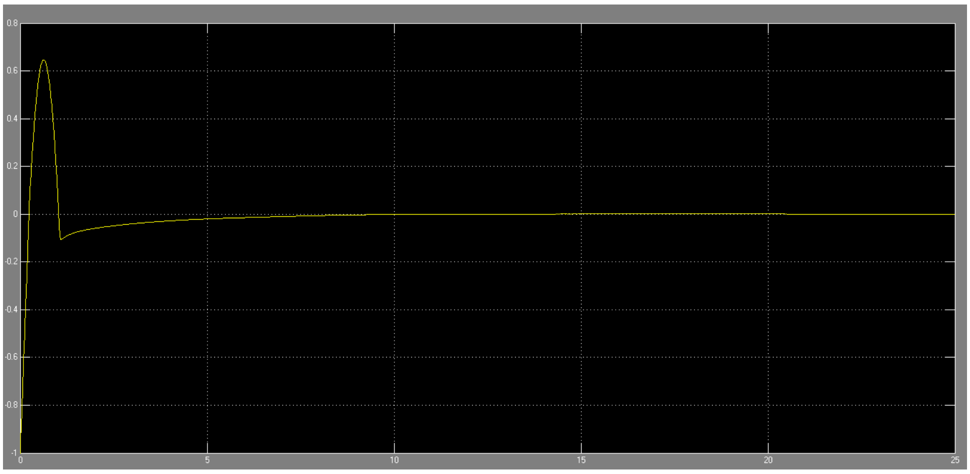

Figure (3–12) cross track error versus time

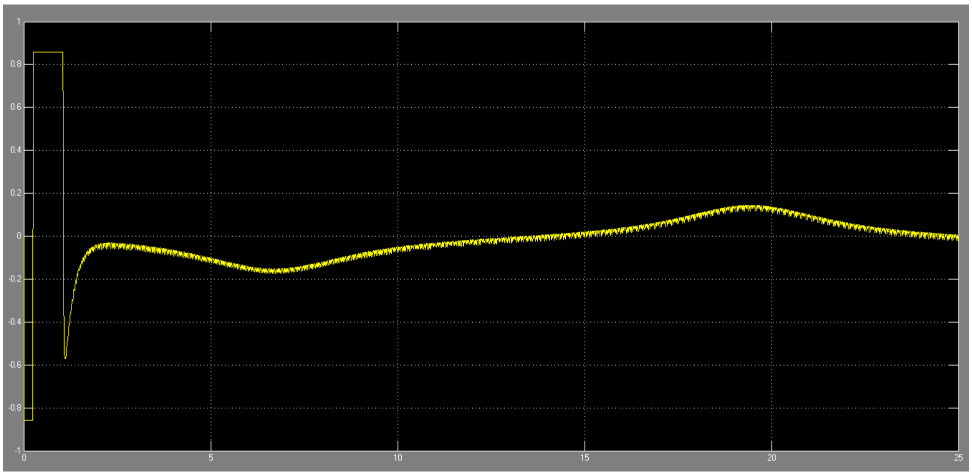

Figure (3–13) steering angle versus time

Figure (3-10) shows an acceptable result via applying this method. However, there are some situations in which PID controller can’t accomplish the task properly. For example, if the vehicle is far enough from the path, the large error will steer the wheel toward the path resulting a change in the orientation. But, since the PID controls the steering angle regardless of the orientation of the vehicle, its large output will remain turning the wheel and increasing orientation. Thus, the vehicle may become perpendicular to the path and start to move in the opposite direction instead of the desired one. The following figures illustrates the situation using a straight path.

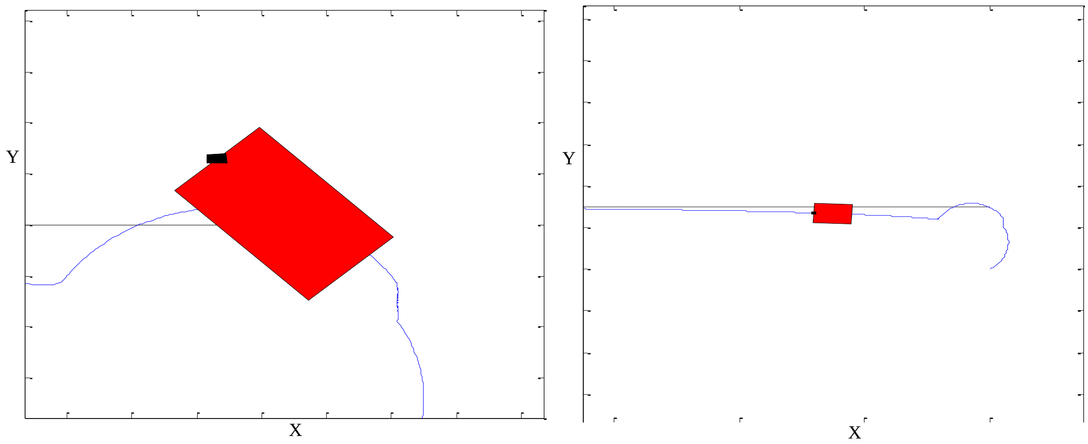

Figure (3–14) CTE-based control failure

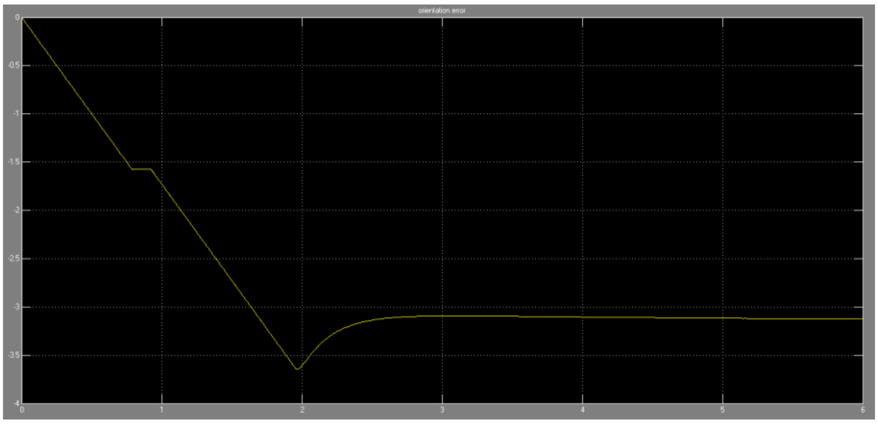

Figure (3–15) orientation error versus time

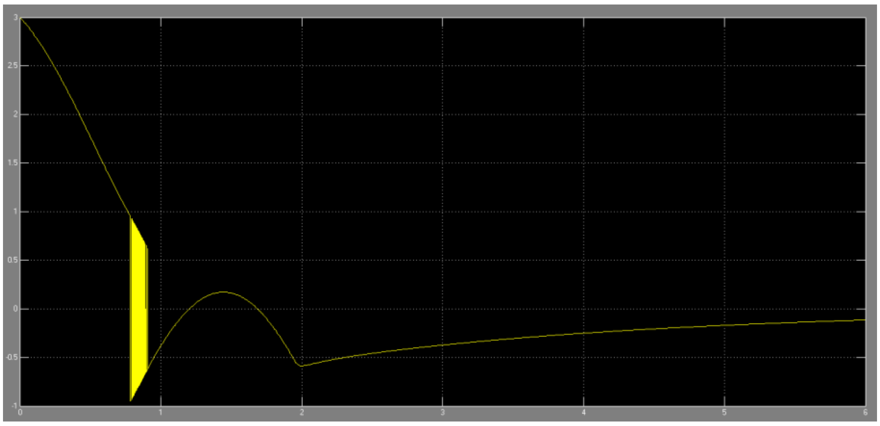

Figure (3–16) cross track error versus time

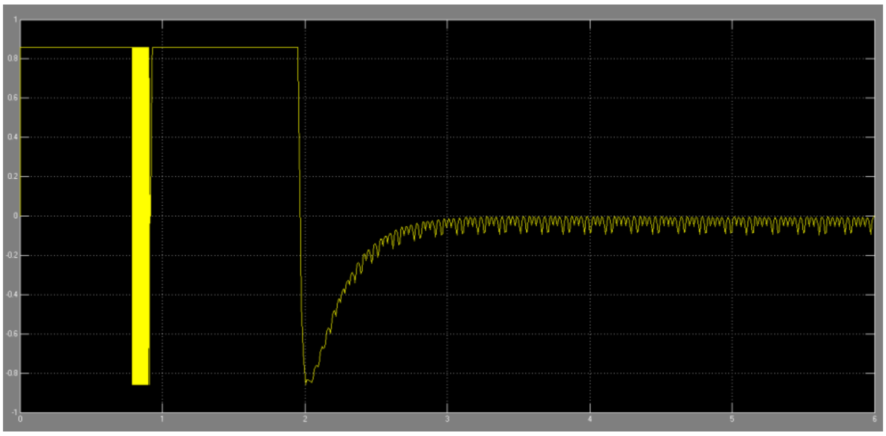

Figure (3–17) steering angle versus time

In figure (3-14), the vehicle had to track the black line and move along it to the right. But, as shown, the vehicle started to move backward to the left.

3-3-Orientation Error & Cross Track Error Handling

The method used to handle both the orientation error and the cross-track error is a combination of the two methods mentioned in sections 3-1 and 3-2. The idea is to make the velocity vector at CG parallel to the path depending on the orientation error first, then to rotate it toward the road depending on the cross-track error. This means that the slip angle of the vehicle β consists of the addition of two angles β_p that makes the velocity vector parallel to the path, and β_c that drives the vector toward the path. The first one equals the orientation error and the second one is the result of the PID controller due to CTE. Then the steering angle δ is calculated using equation (12). The following figures shows the results with a sinusoidal path.

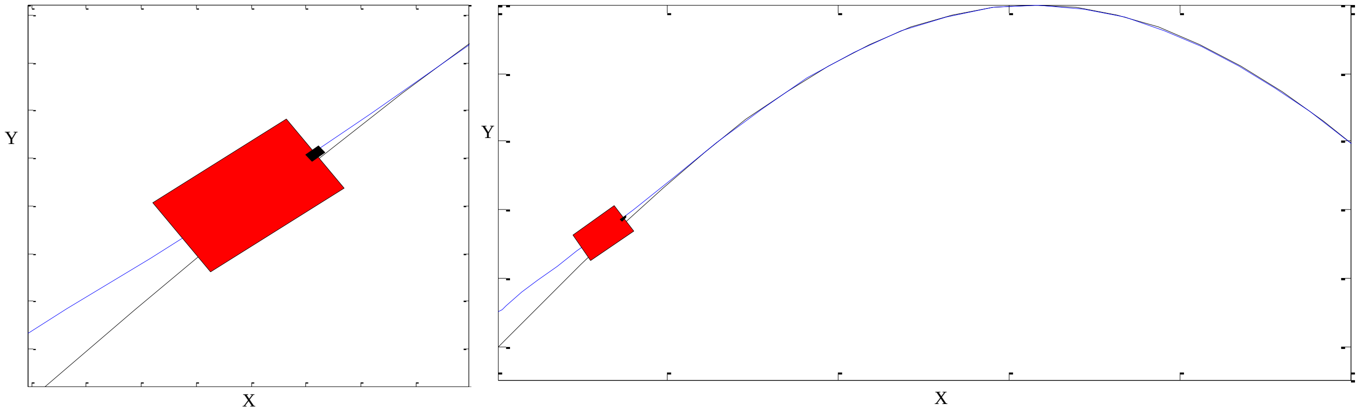

Figure (3–18) orientation & cross track based control

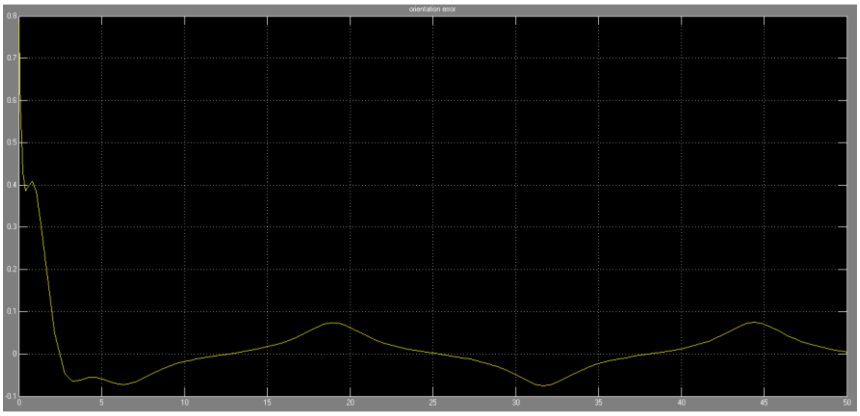

Figure (3–19) orientation error versus time

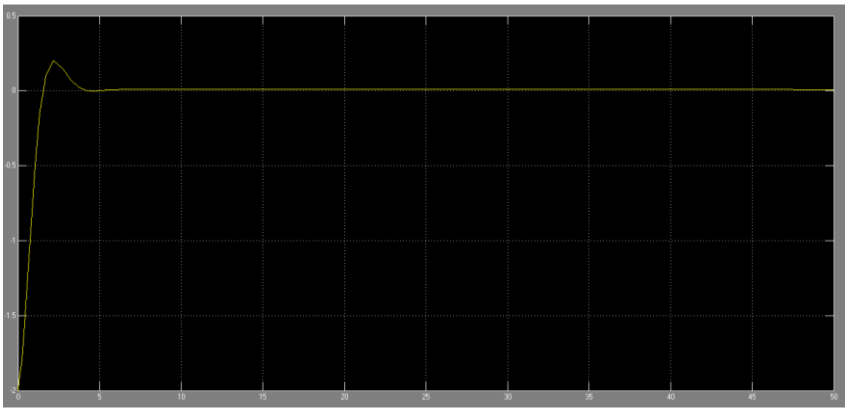

Figure (3–20) cross track error versus time

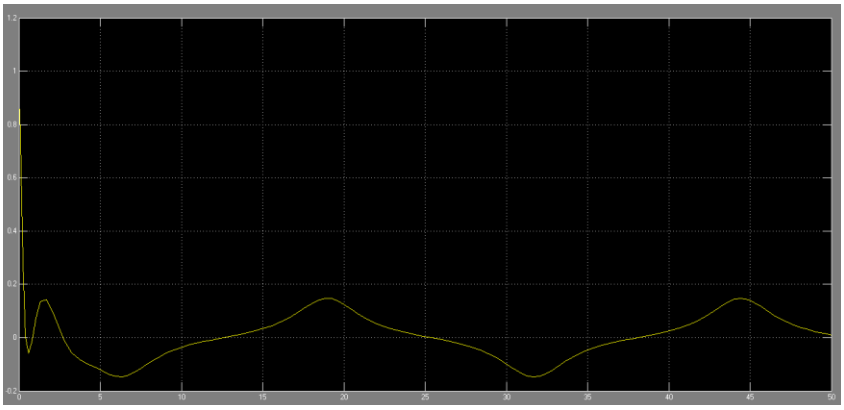

Figure (3–21) steering angle versus time

This method solves the problem of the orientation error based controller since the vehicle is moving toward the path by the effect of β_c and solves the problem of the CTE based controller since the angle β_p prevents the overall slip angle from driving the vehicle in an opposite direction. The following flowchart illustrates the control strategy used.

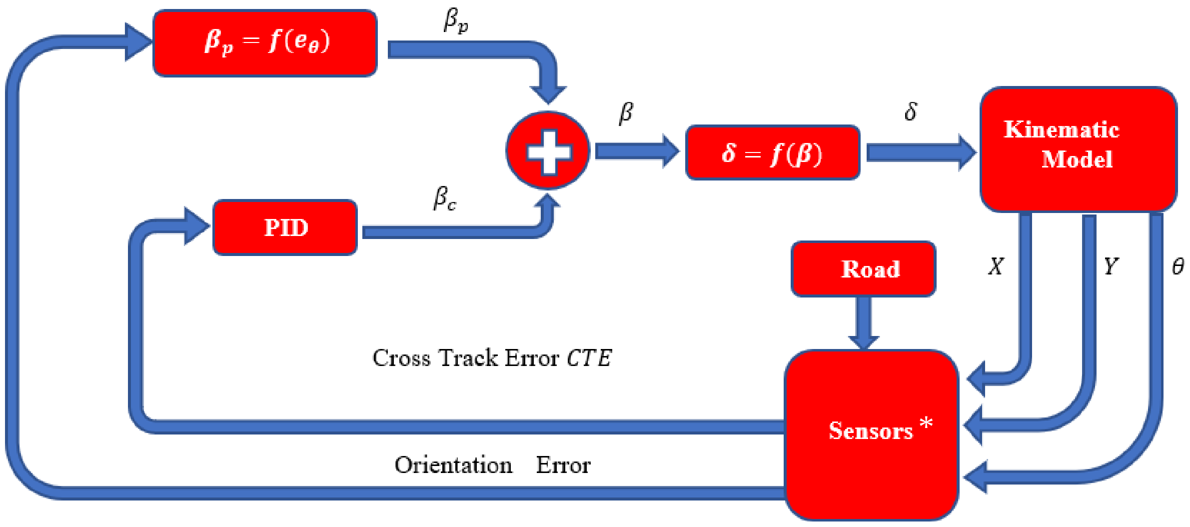

Figure (3–22) control strategy

*Sensors provide the orientation error and CTE using many different methods such as image processing and pre-built map with Inertial Navigation System INS. This part is out of the scope of this project. [6]

Simulation

4-1-Matlab and Simulink

4-1-1-Matlab

MATLAB is a high-level programming language and interactive environment used for numerical computation, visualization, and programming. It has an extensive library of pre-built functions and toolboxes that make it easy to prototype and test algorithms. MATLAB also has a powerful graphics system for creating complex visualizations, and complementary products for modeling, simulation, and deployment. Overall, MATLAB is a versatile tool used in engineering, science, and mathematics for data analysis, image processing, control systems, and more.

4-1-2-Simulink

Simulink is a block diagram environment for simulation and model-based design that is integrated with MATLAB. It allows you to model and simulate dynamic systems using a graphical programming approach, and provides tools for analyzing and visualizing simulation results. Simulink is widely used in industry and academia for designing and testing complex systems, particularly for control system design.

4-2-Block Diagram

This project is simulated using Matlab Simulink environment. The path is supposed to have a mathematical formula presented by $y_p = f(x)$ with its derivative $\frac{dy_p}{dx} = \frac{df(x)}{dx}$. The errors are calculated instantaneously using the road formula and the current outputs which are the position (X, Y) and the orientation θ. The measured errors are inserted in a control block where the control strategy presented earlier was applied. The controller calculates the slip angle β then determines the corresponding steering angle δ. Then, the kinematic model modifies the position X, Y and the orientation θ according the angles δ and β. Finally, these outputs are fed back to the controller by the sensors at which point the cycle repeats.

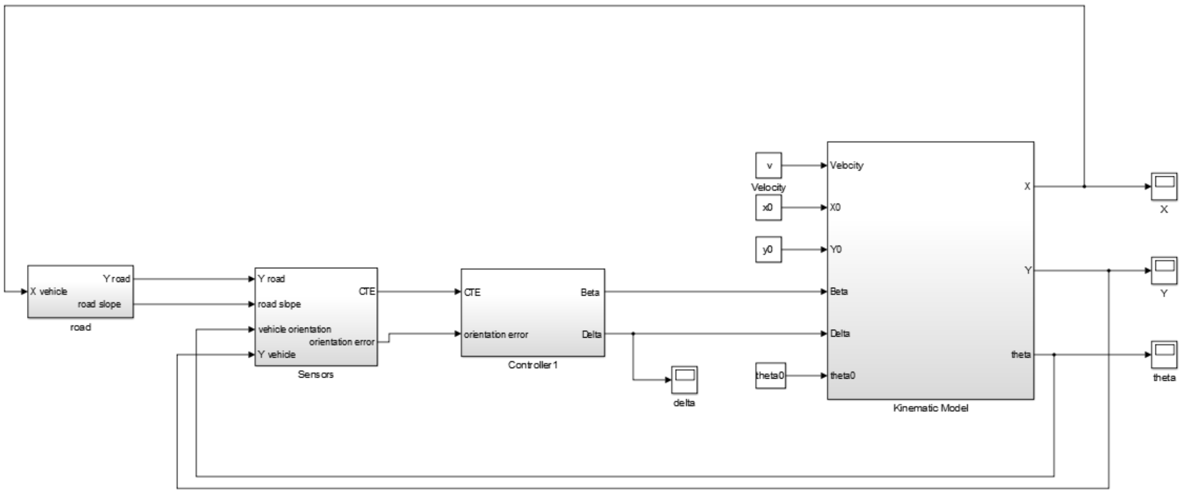

Figure (4–1) block diagram

4-2-1-Road Block

This block contains the mathematical formula of the road and its derivative. The input of this block is the X coordinate of the vehicle and its output is the corresponding Y coordinate of the road y_r and road slope.

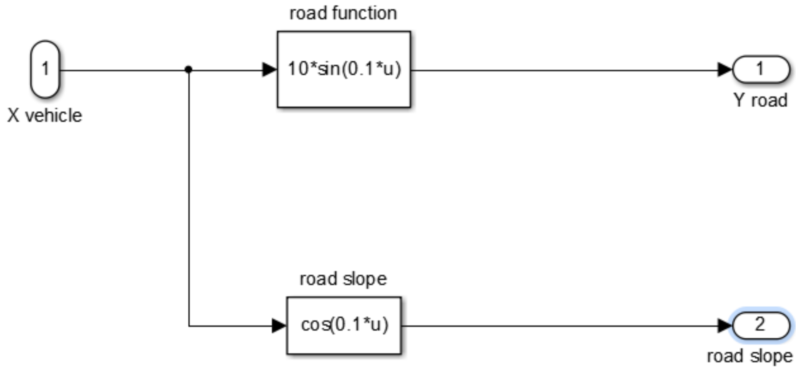

Figure (4–2) road block

4-2-2-Sensors Block

This block represents any system that's able to measure orientation error e_θ and cross track error CTE. In this simulation, the road mathematical expression is used to find the errors as follows:

$$e_\theta = \tan^{-1}\left(\frac{df(X)}{dX}\right) - \theta \qquad (13)$$

where X is the vehicle's X coordinate instantaneously.

The second error CTE is the distance between the vehicle and the road. It should be mentioned that the point of study of the path must be the nearest point to the vehicle's CG. In this case, CTE is approximated to be the distance along Y axis. In other words, CTE is calculated as the difference between the Y coordinate of the vehicle and road's Y coordinate at the point corresponding to the vehicle's X coordinate y_r. This approximation simply determines in which side the vehicle is located relative to the road.

$$CTE = y_r - Y \qquad (14)$$

However, the sign of CTE must be flipped when the vehicle moves in the opposite direction of X axis. So, the interval test block examines the orientation of the vehicle. If the orientation belongs to the interval $[\frac{\pi}{2}, \frac{3\pi}{2}]$ then it outputs 1 indicating that the sign must be reversed.

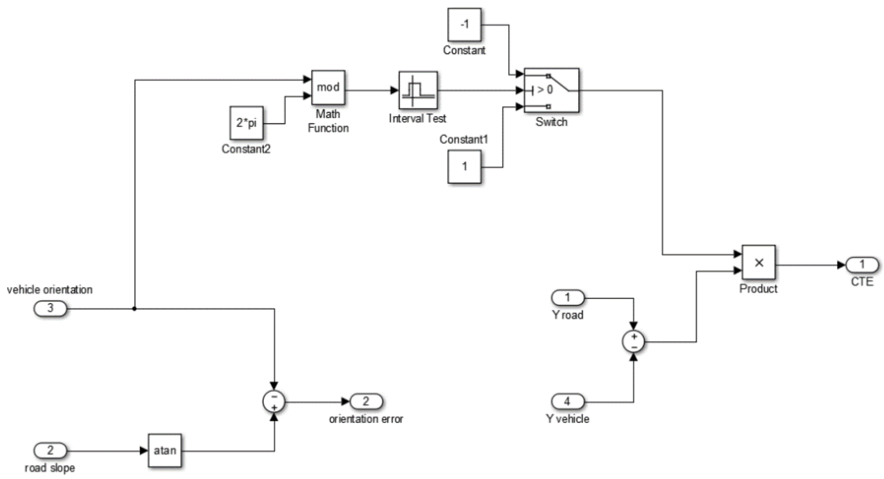

Figure (4–3) sensors block

4-2-3-Controller Block

This block includes the controlling strategy illustrated in section 3-3. First, a PID controller is applied to the CTE to determine β_c which saturated into the interval $[-\frac{\pi}{6}, \frac{\pi}{6}]$. PID parameters are calculated using the automatic tune provided by Matlab Simulink. Second, the orientation error β_p is added to β_c to determine the slip angle β. The result is saturated again into the interval $[-\frac{\pi}{6}, \frac{\pi}{6}]$ to satisfy the assumptions mentioned earlier. Finally, the steering angle δ is calculated from β using equation (12).

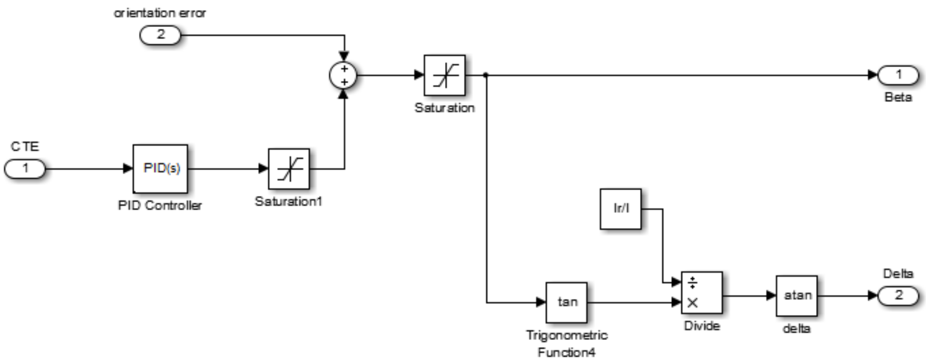

Figure (4–4) controller block

4-2-4-Kinematic Model Block

This block simulates the motion of the vehicle using the lateral kinematic model equations (2), (3), (8). The input of this block is the constant velocity of the CG, the steering angle δ, the slip angle β, along with the initial configuration of the vehicle X_0, Y_0 and θ_0. This block outputs the X, Y coordinates of CG and the orientation of the vehicle θ at each instant.

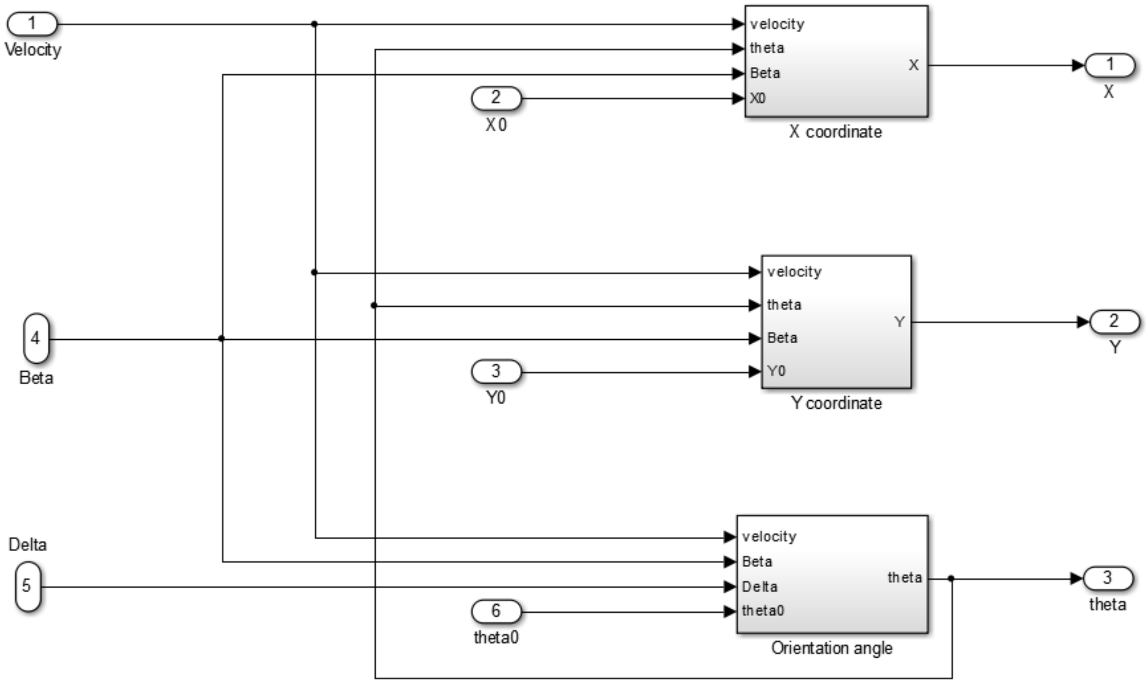

Figure (4–5) kinematic model block

4-2-4-1- X Coordinate Block

This block represents equation (2) that relates the time change of the X coordinate of the vehicle to the system's input and variables.

$$\dot{X} = v.\cos(\beta + \theta)$$

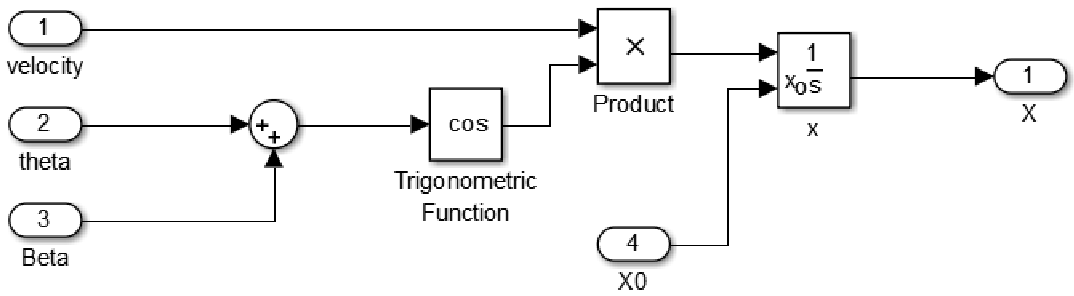

Figure (4–6) X coordinate block

4-2-4-2- Y Coordinate Block

This block represents equation (3) that relates the time change of the Y coordinate of the vehicle to the system's input and variables.

$$\dot{Y} = v.\sin(\beta + \theta)$$

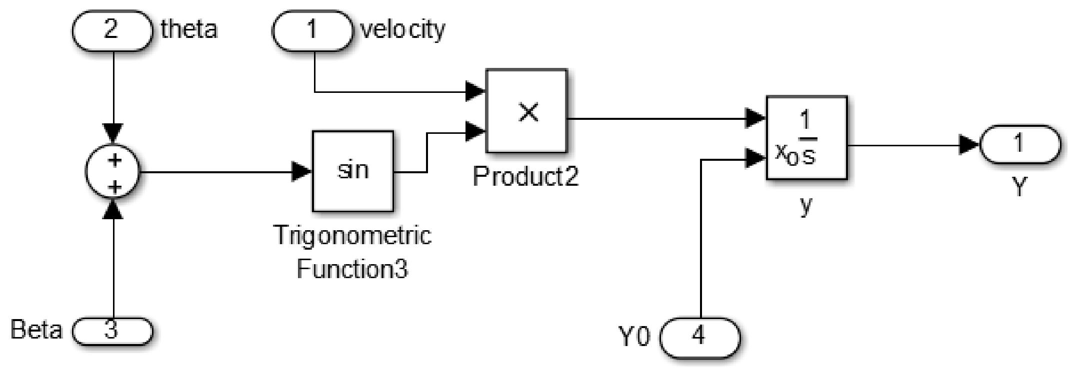

Figure (4–7) Y coordinate block

4-2-4-3-Orientation Angle Block

This block represents equation (8) that relates the time change of the orientation angle of the vehicle to the system's input and variables.

$$\dot{\theta} = v \times \frac{\tan(\delta) \times \cos(\beta)}{L}$$

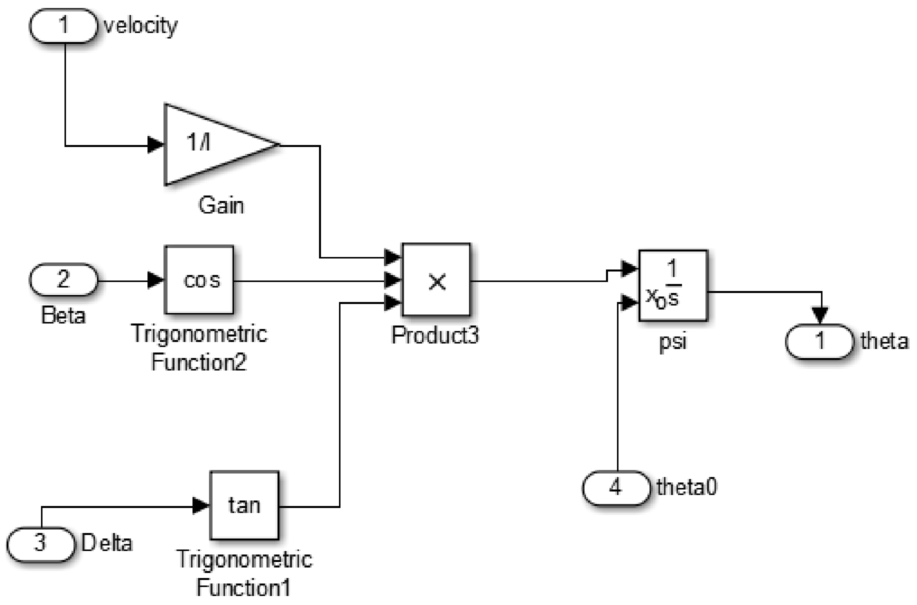

Figure (4–8) orientation angle block

Conclusion

This project presents a control algorithm that aims to steer an autonomous vehicle to follow a continuous and smooth path with small curvature. The algorithm addresses two types of errors: orientation error and cross-track error. The orientation error is handled using a geometric approach, while the cross-track error is addressed using a PID controller. To use the geometric approach alone, the initial position of the vehicle must be on the path with an orientation angle close enough to the path's orientation; otherwise, the vehicle will follow a parallel path that is not suitable for tracking the desired path. Similarly, to use the PID controller alone, the initial position of the vehicle must be close enough to the path; otherwise, the vehicle may move in the opposite direction to the desired path. The project shows that the combined algorithm that combines both methods can solve the issues of each individual approach and achieve acceptable results in path tracking. The Matlab Simulink environment is used to simulate the algorithm and control methods based on the kinematic model and mathematical formula for the path. Finally, the application of this project is essential to develop a realistic mathematical model of the vehicle and the control algorithm that reflects real-world conditions.

References

[1] Marino, Riccardo ; Scalzi, Stefano; & others;, "A Nested PID Steering Control for Lane Keeping in Vision Based Autonomous Vehicles," in *American Control Conference* , St. Louis, MO, USA , 2009.

[2] J. Kang and Y. Hindiyeh, " Design and Testing of a Controller for Autonomous Vehicle Path Tracking Using GPS/INS Sensors," in *IFAC World Congress*, Seoul, Korea, 2008.

[3] V. Cerone and M. Milanese, "Combined Automatic Lane Keeping and Driver's Steering Through a 2-DOF Control Strategy," *IEEE Transactions on Control Systems Technology,* vol. 17, pp. 135-142, 2009.

[4] V. Cerone and A. Chinu, " Experimental results in vision-based lane keeping for highway vehicles," in *American Control Conference,* Anchorage, 2002.

[5] P. Falcone and F. Borrelli, "Predictive Active Steering Control for Autonomous Vehicle Systems," *IEEE Transactions on Control Systems Technology,* vol. 15, pp. 566-580, 2007.

[6] C. MacAdam, " Application of an Optimal Preview Control for Simulation of Closed-Loop Automobile Driving," *IEEE Transactions on Systems, Man, and Cybernetics,* vol. 11, pp. 393-399, 1981.

[7] R. Rajamani, Vehicle Dynamics and Control, Minnesota: springer, 2006.

[8] C. Samak, T. Samak and S. Kandhasamy, CONTROL STRATEGIES FOR AUTONOMOUS VEHICLES, India, 2021.

[9] S. Taherian and S. Fallah, "Autonomous Emergency Collision Avoidance and Stabilisation in Structured Environments," *JOURNAL OF LATEX CLASS FILES,* vol. 14, p. 2, 2015.

[10] N. Nise, Control Systems Engineering, California: John Wiley & Sons, Inc, 2011.