Paper: ASAT-17-023-MO

17th International Conference on
AEROSPACE SCIENCES & AVIATION TECHNOLOGY,
ASAT - 17 – April 11 - 13, 2017, E-Mail: asat@mtc.edu.eg
Military Technical College, Kobry Elkobbah, Cairo, Egypt
Tel: +(202) 24025292 – 24036138, Fax: +(202) 22621908

Modeling and Simulation of a Vision-Based Autonomous Vehicle

A. Desoky[*], A. Bayoumy[†], G. Hassaan[‡]

Abstract: This paper presents a comprehensive mathematical simulation for the trajectory of a vision-based autonomous vehicle during moving between lane lines of the structured road. The simulation accomplished by using MATLAB/Simulink software. This simulation mimics the existence of an actual digital camera by using a novel 3D-vision block to simulate the actual images that assumed to be provided by a digital camera connected to an embedded computer. The 3D-vision block uses mathematical equations, execution sequence and logical conditions to create a virtual captured image. So, this virtual image is then used to detect the lane in the front of the vehicle depending on the virtual camera position and its parameters. Inside simulation environment that based on the kinematic model of the vehicle and vision model, the controller has been designed that will be coded in the embedded computer. Several evaluations are shown and discussed about the lane detection and 3D vision simulation.

Keywords: vision-based, lane detection, autonomous vehicle, vision modeling and simulation.

Nomenclature

ar	The road slope
br	The road y-intercept
b	Rear wheelbase from C.G
$D_{average}$	Average of offset distance errors ratio
D_{FP}	Predicted Offset Distance Error
D_{FI}	Offset Distance Error by Virtual Image
D_{FF}	Offset Distance Error by Frustum
D_F	Real Offset Distance Error
F_{yf} F_{xf}	Forces acting in the longitudinal and lateral direction
$\mathrm{H_{cam}}$	Camera height from the ground
I_z	Car moment of inertia
L	Wheel base.
L_f	Distance from the front axle to the center of gravity
L_r	Distance from the rear axle to the center of gravity
m	Vehicle mass
n	Vehicle wheel speed
P1,P2,P3,P4	Frustum Field of view points on the ground.
$\mathrm{P_{1-inter-L}}$	Intersection point between P1 and lane left line

[*] MSc Student, Faculty of Engineering, Cairo university, Giza, Egypt, ahmeddesokyeg@gmail.com
[†] Assistant Professor, MSA University, Giza, Egypt, ambayoumy@msa.eun.eg
[‡] Emeritus Professor, Faculty of Engineering, Cairo university, Giza, Egypt, galalhassaan@ymail.com

$P_{1-\text{inter}-R}$	Intersection point between P1 and lane right line
$P_{2-\text{inter}-L}$	Intersection point between P2 and lane left line
$P_{2-\text{inter}-R}$	Intersection point between P2 and lane right line
$P_{3-\text{inter}-L}$	Intersection point between P3 and lane left line
$P_{3-\text{inter}-R}$	Intersection point between P3 and lane right line
$P_{4-\text{inter}-L}$	Intersection point between P4 and lane left line
$P_{4-\text{inter}-R}$	Intersection point between P4 and lane right line
r	Radius of vehicle wheel
R	Instantaneous radius of the vehicle curvature
S	Image matrix
v	Vehicle longitudinal speed
V_L	The tangential velocity of the left wheel on rear
v_x, v_y	Longitudinal and lateral speed
X_{car}, Y_{car}	Global coordinate of the midway of the rear wheels.
X_{cam}, Y_{cam}	Global coordinate of the camera.
X_c	The bottom point of the road centerline location in x direction
$x_{\text{Im bottom}}$	Location of P1 and P2 in x direction
$x_{\text{Im top}}$	Location of P3 and P4 in x direction
y_{p1}	Location of P1 in y direction
y_{p3}	Location of P3 in y direction
δ	Vehicle steering angle.
$\theta_{average}$	Average of heading angle errors ratio
θ_{HP}	Predicted Heading Angle Error
θ_{HI}	Heading Angle Error by Virtual Image
θ_H	Real Heading Angle Error
θ_{road}	Road angle with respect to horizontal
θ_{car}	Vehicle orientation in the global frame.
θ_{cam}	Camera orientation in the global frame.
$\dot{\theta}_{\text{car}}$	Vehicle yaw rate [vehicle angular velocity]
θ_1	The angle of right lane line w.r.t horizontal
θ_2	The angle of left lane line w.r.t horizontal
ω	Angular velocity of the vehicle wheel
α	Horizontal half field of view
β	Vertical half field of view
γ	Camera inclination angle (pitch)

1. Introduction

Autonomous vehicle are a type of vehicle, which can perform the desired tasks without human guidance. Researchers are being done to create such vehicle that can cope with a variety of environment, typically, on land, underwater, air borne, underground, or in space. From there, the car would take over and drive to destination with no human input. The car would be able to

sense its environment to make steering and speed changes as necessary. Therefore, to develop an autonomous vehicle it will involve automated driving, navigating and monitoring systems.
Many projects on autonomous driving are in development in industry. In 1980, Mercedes-Benz was the first to develop an autonomous road vehicle. It drove on empty highways at speeds around 100 km/h.
In the 1990s, DARPA created an autonomous land vehicle that was able to drive cross-country. Since then they organized their famous DARPA Grand Challenge, a competition for autonomous vehicles.
In the last years, almost all car manufacturers are busy developing some sort of self-driving car that can maneuver in real world traffic [1].
Recently there are many researches until now about the autonomous driving systems used a lane-keeping vision system to detect the road lane and compute road information such as heading angle and offset displacement, which are then sent to the controller operating the steering wheel to ensure that the vehicles driving in the inside lane.
The process of estimating the position and heading direction of the vehicle inside the lane is essential for many autonomous vehicle applications. The developments of techniques for autonomous navigation and control of autonomous vehicles has become an important and active research topic. By autonomous navigation, we mean the vehicle's capability of purposefully moving without any human intervention.

1.1 Related Work

We will investigate some of the major research projects in the field of automated driving, with a strong focus on sensing. Automotive companies carry out most of the research in the field of automated driving; hence, there is not much public information available.

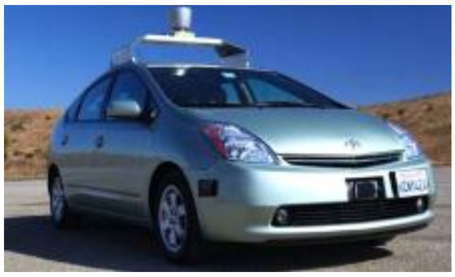

Fig.1. One of Google's automated Toyota Priuse [2]

In (Fig.1) Google's automated Toyota Prius vehicles have traveled more than 300.000 km on public roads. The vehicle's main sensor is a large Velodyne HDL 64E LIDAR unit mounted on the roof, which creates a high-resolution map of the surroundings, 360 degrees around the vehicle. In addition, four RADAR units are used (three on the front bumper, one on the rear bumper), with sufficiently large range for the detection of vehicles on highways. A camera sensor mounted behind the rearview mirror, which is used to detect traffic lights. Furthermore, a GPS / INS and wheel encoder are used to accurately determine the vehicle position. In order to be able to autonomously drive in a certain area, the vehicle must be driven through it manually first, so it can create a map of the environment. When the vehicle drives autonomously, it compares fresh data with the environment map, which allows it to differentiate with pedestrians and stationary objects [2].
In 2004, DARPA offered a $1 million prize to the first autonomous vehicle capable of negotiating a 150 mile course through the Mojave Desert. For the first time, the vehicles were required to be fully autonomous with no humans allowed in the vehicle during the competition. By this time, GPS systems were widely available, significantly improving navigational abilities. Despite several high profile teams, and general advances in computing

technology, the competition proved to be a disappointment with the most successful team reaching only 7 miles before stopping.
The following year, DARPA re-held the competition. This time the outcome was very different with five vehicles completing the 132 mile course and all but one of the 23 finalists surpassing the seven miles achieved the previous year. Stanley won the competition.
Buoyed by this success, DARPA announced a new challenge, to be held in 2007 named the Urban Challenge. This would see the competition move from the desert to an urban environment autonomous land vehicle that was able to drive cross-country, with the vehicles having to negotiate junctions and maneuver in the presence of both autonomous and human driven vehicles [1].

Fig.2. The “Boss” Vehicle won the 2007 DARPA urban challenge [1]

In the DARPA Urban Challenge, fully automated vehicles have to maneuver their way through an urban environment. These vehicles have to perform regular driving tasks such as turning and keeping headway, as well as special maneuvers such as parking, U-turns, etc. Many teams from universities all over the world compete in this challenge. (Fig.2) shows the Boss vehicle. It uses 11 LIDAR sensors (4 different types), 5 long-range dynamic RADARs (Continental ARS300), and 2 cameras.
In [3] it presents a comprehensive experimental and theoretical study of a proposed mechatronics system for an autonomous ground vehicle. This vehicle has the capability of lane detection and tracking it. The system considered employs a computer vision technique in which real data are collected by a single calibrated camera. Further processing and analysis to the images captured by the camera are carried out to recognize the lane lines. The numerical simulation is implemented in the MATLAB/Simulink environment.
In [4] An intelligent vehicle control system is designed and embedded in a digital signal processing (DSP) platform (eZdspTMF2812). A golf cart is used as an installation platform for the overall system, including steering wheel alternating current (AC) serve motor, brake actuator, throttle driving circuit and sensors. Digital image processing technology is also used to enable the autonomous driving system to achieve multi-mode lane-keeping. The overall system is tested and evaluated on a university campus.

1.2 Contribution

This paper has two contributions:

a- Developing and simulating the vision system of an autonomous vehicle to detect the road lane and create virtual image using MATLAB/Simulink environment.
b- Vehicle steering control system of the proposed simulation using successive loop closure with optimal control gains by using optimization toolbox in MATLAB/Simulink environment.

2. Platform Description

In this paper, the proposed simulation of an autonomous vehicle is consider for a vehicle model with four rigid wheels that had been used in [3] as shown in Fig.3 with features as shown in

Table 1.The vehicle is driven through a DC-electric motor which is installed at the rear-left wheel of the vehicle without any differentiation between its wheels. The front axle of the vehicle is steered through another DC-electric motor that controls the direction of the vehicle. The system is able to identify and track the lanes on real time. The autonomous driving system is equipped with a vision system (digital camera) with features as shown in

Table 2 that collects road data such as lateral offset displacement and heading angle.

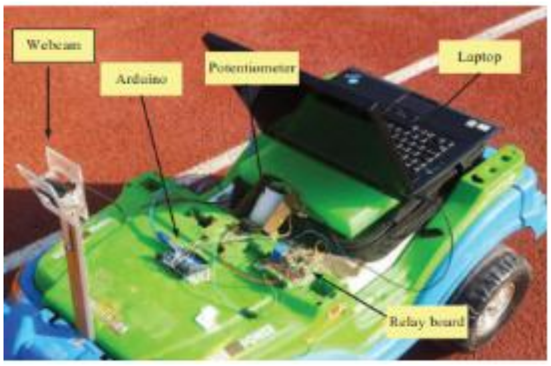

Fig.3. Experimental Platform Description [3]

Table 1. Experimental Platform features [3]

Parameters		Value	
1	Vehicle width	64	cm
2	Vehicle length	112	cm
3	Wheel base (L)	55	cm
4	Wheel track (b)	48	cm
5	Wheel radius (r)	13	cm
6	Tire width (w)	7	cm
7	Max Wheel Steering angle	±15	degree
8	Vehicle speed (v)	0.5	m/s

Table 2. Camera features [3]

1	Average Half Vertical FoV (α)	28/2	degree
2	Average Half Horizontal FoV (β)	36.8/2	degree
3	Camera Inclination Angle(γ)	30	degree

3. Proposed Method

The autonomous driving system is equipped with a vision system. The vision system is able to detects and track lane-markings and provides look-ahead road information such as lateral offset and displacement of angle relates to maintain in the centerline of the road, with its heading.
The vision system includes an embedded platform and a digital camera. The embedded platform performs image processing whereas the digital camera captures the image data.
This paper supposes existence the actual digital camera in the front of experimental vehicle used in [3] that capture the image and the embedded platform perform image processing. This work simulates the vision-based autonomous vehicle trajectory to be able to detect and track a road lane from the simulated images that assumed to be provided by digital camera connected to the embedded platform.

The proposed method simulates the following four subsystems as shown in (Fig.4):

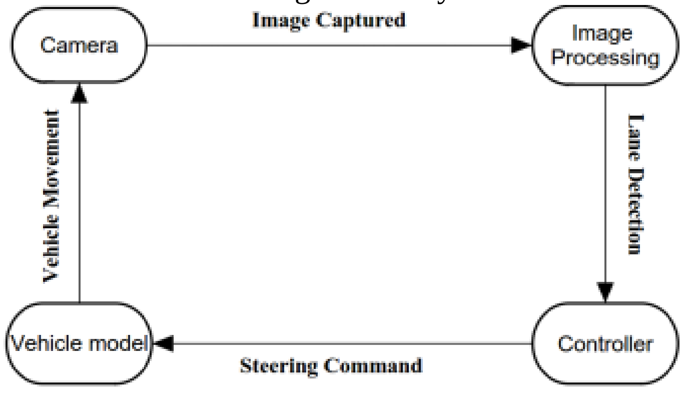

Fig.4. Subsystems of a Vision-Based Autonomous Vehicle

The first subsystem is the sensor (the digital camera); this camera provides the information of the environment by capturing the image in front of the vehicle during driving. The second subsystem is image-processing subsystem that captures images frame by frame. After that, lane detection algorithm that embedded into the laptop that used in [3] extracts the information about vehicle position with respect to the lane lines. The results are angle between the road lane midline and orientation of the vehicle (the vehicle-heading angle) and lateral offset distance.
This paper simulates the first and the second subsystem using a novel 3D vision block that has mathematical equations and logical conditions to create a virtual image and detect the lane in the front of the vehicle depending on the virtual camera position and its parameter. The simulation environment based on the kinematic model of the vehicle and vision model that simulate the digital camera and image processing algorithm that will be into the embedded platform.
The controller subsystem is the third subsystem uses the resulted heading angle and offset distance errors from lane detection algorithm. Designing a vehicle controller requires a model of vehicle's behavior.
Therefore, vehicle subsystem is the fourth subsystem, in this work vehicle's kinematics and dynamics behaviors are used as a vehicle model. As the experimental vehicle used in [3] has very low and constant speed (0.5 m/s) acceleration and consequently forces will be small, and measuring the tire stiffness coefficients and the slip angles, which relay on the ground roughness, tire material and shape, need advanced unavailable equipment thus the kinematic model will be sufficient for vehicle modeling.

4. Mathematical Model

4.1 Vehicle Kinematic Model

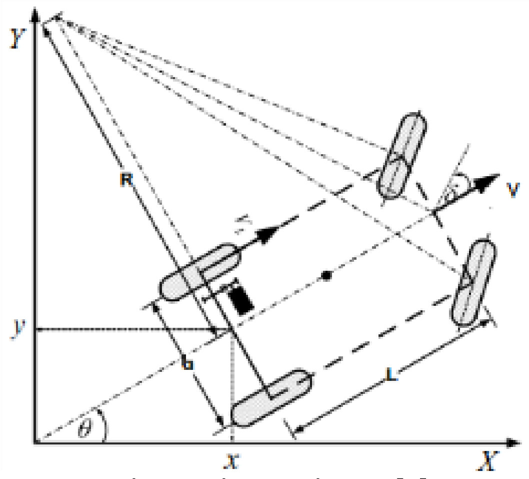

Fig.5. Kinematic model

This section presents a simple vehicle kinematic model which is used in many path planning works[3, 5]. The vehicle moves in a configuration global space (X_G, Y_G) that is given by (x_{car}, y_{car}, θ_{car}) as shown in (Fig.5). This system implies the moving without skidding assumption. The instantaneous radius of curvature R of the vehicle determined from v and $\dot{\theta}$:

$$R = \frac{v}{\dot{\theta}_{car}} = \frac{L}{\tan\delta} \tag{3.1}$$

$$V_L = \omega.r \tag{3.2}$$

Curvature radius of the trajectory described by left wheel δ>0
Curvature radius of the trajectory described by right wheel δ<0

$$R' = \begin{cases} R - \dfrac{b}{2} & \delta > 0 \\ R + \dfrac{b}{2} & \delta < 0 \end{cases} \tag{3.3}$$

$$\dot{\theta}_{car} = \begin{cases} \dfrac{V_L}{R - \frac{b}{2}} & \delta > 0 \\ \dfrac{V_L}{R + \frac{b}{2}} & \delta < 0 \end{cases} \tag{3.4}$$

$$\dot{\theta}_{car} = \frac{V_L}{R'} \tag{3.5}$$

$$\dot{\theta}_{car} = \frac{v}{R} \tag{3.6}$$

$$v = \dot{\theta}_{car} R \tag{3.7}$$

$$v = \frac{V_L}{R'} \frac{L}{tan\ \delta} \tag{3.8}$$

(x, y) may be expressed as:

$$\dot{x}_{car} = v\cos\theta_{car} \tag{3.9}$$

$$\dot{y}_{car} = v\sin\theta_{car} \tag{3.10}$$

4.2 Vision System Model (3D Vision Model)

The vision model simulates the first and the second subsystem that mentioned in section (3).This model simulates the actual digital camera as shown in Fig.6 and Fig.7 and generates virtual images as shown in Fig.8 with assuming ideal lane detection.

This section develops and simulates the vision system of an autonomous vehicle to detect the road lane then create virtual image. This image depend on camera parameter and its position, this virtual image contains the lines of the road, from this image the simulation can compute and analysis road information such as heading angle and offset displacement, which are then sent to the controller to produce the required steering wheel angle to ensure that the vehicle is driving inside the lane.

The location of the camera with respect to the location of the center of the rear axle of the vehicle that is given by (x_{car}, y_{car}, θ_{car}) as shown in Fig.5 can be defined by the following equations:

$$x_{cam} = L\cos\theta_{car} + x_{car} \tag{3.11}$$

$$y_{cam} = L\sin\theta_{car} + y_{car} \tag{3.12}$$

$$\theta_{cam} = \theta_{car} \tag{3.13}$$

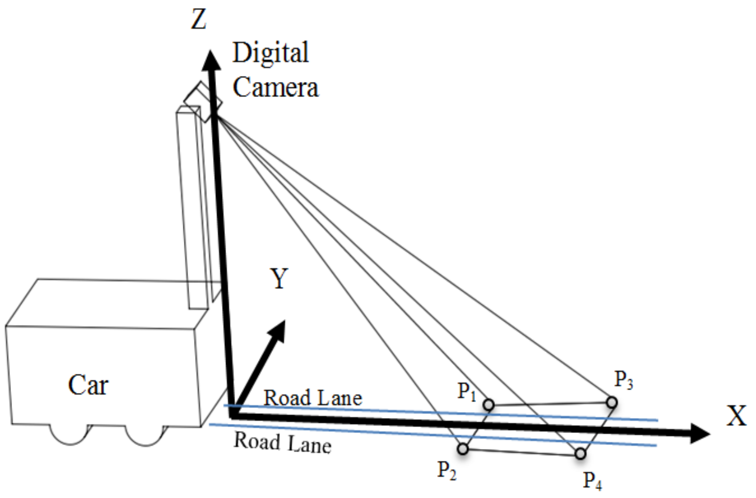

Fig.6. Vehicle Camera Viewing Frustum

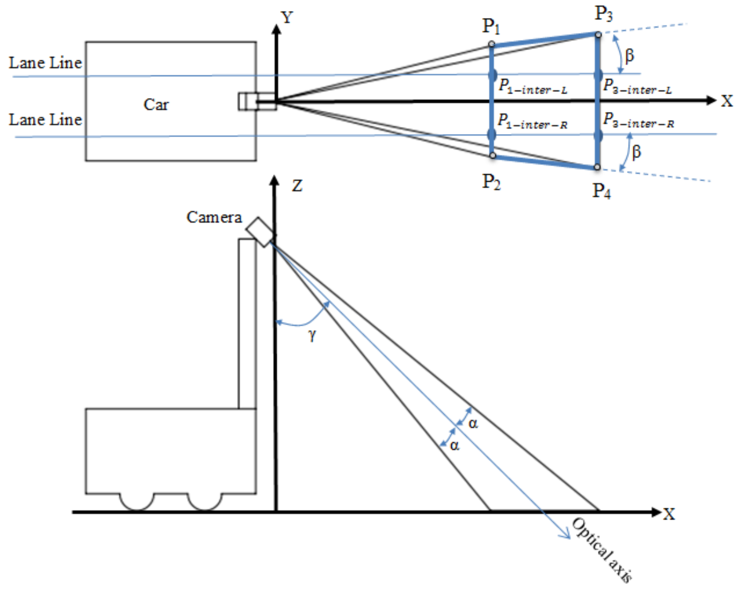

Fig.7. Viewing frustum from Top and side view

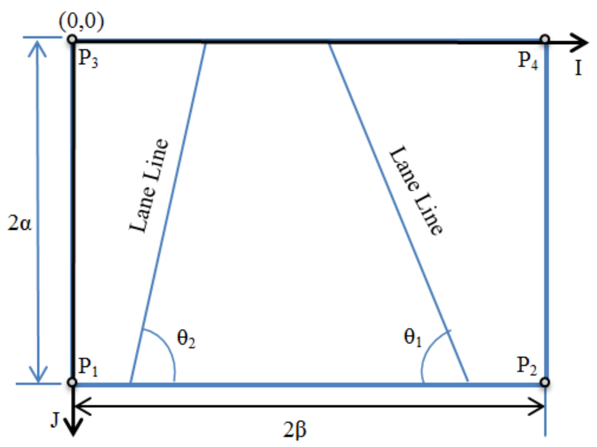

Fig.8. Road lane in the Virtual image

The goal of lane detection is to detect the lane in the front of the vehicle depending on the virtual camera position (x_{cam}, y_{cam} and θ_{cam}) and its parameter (α, β and γ) with respect to the position of the lane (x_{road}, y_{road}, and θ_{road}) and calculate lane errors (θ_H , D_F) respectively.
The vision-based lane detection system used in the proposed simulation depends on camera viewing frustum and horizontal and vertical field of view as they were mentioned in [6, 7].
The actual vehicle is outfitted with camera located at the front, so the vehicle can find its lateral error from the lane centerline this error depend on heading angle and lateral offset distance (θ_H , D_F).
The frustum points as shown in Fig.6 and Fig.7 with ground with respect to the local coordinate system (vehicle coordinate system) can be defined by the following equations:

$$x_{\text{Im bottom}} = H_{cam} \times \tan(\gamma - \alpha) \tag{3.14}$$

$$x_{\text{Im top}} = H_{cam} \times \tan(\gamma + \alpha) \tag{3.15}$$

$$y_{p1} = \frac{H_{cam}}{\cos(\gamma - \alpha)} \times \tan(\beta) \tag{3.16}$$

$$y_{p3} = \frac{H_{cam}}{\cos(\gamma + \alpha)} \times \tan(\beta) \tag{3.17}$$

The transformation of the frustum points on ground from the local coordinate system (X_L, Y_L) (vehicle coordinate system) to global coordinate system (X_G, Y_G) can be transformed by the following matrix:

$$\begin{bmatrix} x_G \\ y_G \end{bmatrix} = \begin{bmatrix} \cos(\theta_{cam}) & -\sin(\theta_{cam}) \\ \sin(\theta_{cam}) & \cos(\theta_{cam}) \end{bmatrix} \begin{bmatrix} x_L \\ y_L \end{bmatrix} + \begin{bmatrix} x_{cam} \\ y_{cam} \end{bmatrix} \tag{3.18}$$

The projection of the camera frustum on the ground creates a trapezoid with points (P1, P2, P3, and P4) as shown in (Fig.7) and the ground trapezoid represents the rectangular of simulated image as shown in (Fig.8).
Then, we can get the four points of intersection of the four lines of trapezoid with the two lines of the lane. However, there are many possibilities of the location of the four points of intersection in the simulated image, hence, each side of the simulated image may contain two points of intersection, and these possibilities depend on the vehicle pose with the lane. Therefore, a novel image matrix in equation (4.19) that is represented by:

$$S = \begin{bmatrix} - & - & 0 & 0 & - & - & 2\beta & 2\beta \\ -2\alpha & -2\alpha & - & - & 0 & 0 & - & - \end{bmatrix}_{2\times 8} \tag{3.19}$$

where;

The first row represents the coordination of points on image in I direction, and the second row represents the coordination of points on image in J direction as shown in Fig. 10 .
Each column represents a point of intersection as shown in Fig. 10 and as represented in Table 3:

Table 3: Explanation of the image matrix

Column Number	Intersection Point Number	Related Lane line	Related Image Side
1	1	Left line	Lower side of the image
2	2	Right line	
3	3	Left line	Left side of the image
4	4	Right line	
5	5	Left line	Top side of the image
6	6	Right line	
7	7	Left line	Right side of the image
8	8	Right line	

- The matrix elements (1, 3), (1, 4), (1, 7), (1, 8) represent the image frame limits in I direction, and the matrix elements (2, 1), (2, 2), (2, 5), (2, 6) represent the image frame limits in J direction.
- The first two elements in the first row of the matrix represent the distance in I direction of the intersection point of left and right lane lines respectively with the lower side of the trapezoid.
- The 3rd and 4th elements in the second row of the matrix represent the distance in J direction of the intersection point of left and right lane lines respectively with the left side of the trapezoid.
- The 5th and 6th elements in the first row of the matrix represent the distance in I direction of the intersection point of left and right lane lines respectively with the top side of the trapezoid.
- The 7th and 8th elements in the second row of the matrix represent the distance in J direction of the intersection point of left and right lane lines respectively with the right side of the trapezoid.

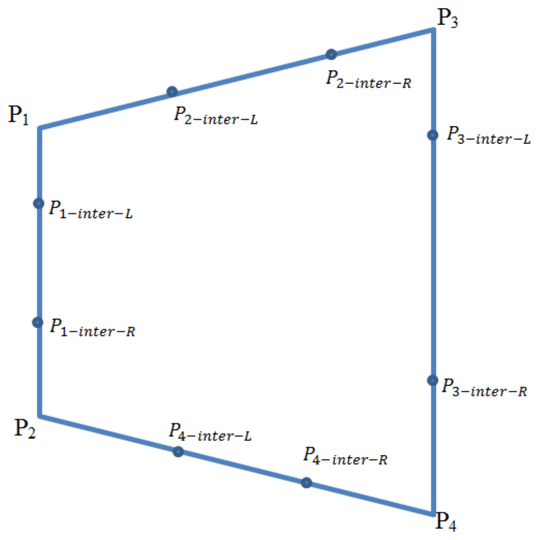

Fig. 9. The Location Possibility of the Intersection Points in the Trapezoid

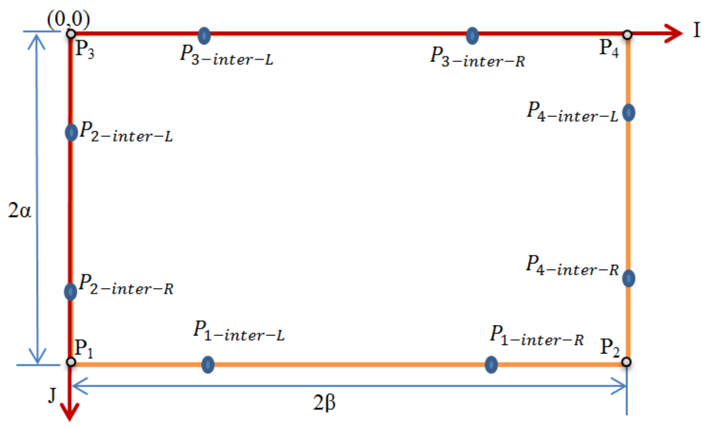

Fig. 10. The Location Possibilities of the Intersection Points of in the Virtual Image

From mapping and similarity between the trapezoid as shown in (Fig. 9) and rectangular image as shown in (Fig. 10), the coordination of the intersection points in the rectangular simulated image can be computed by the following similarity equations:

$$S(1,1)=\frac{(2\beta)*(\text{distance between } P_1 \text{and} P_{1-\text{inter}-L})}{(\text{distance between } P_1 \text{and} P_2)} \qquad (3.20)$$

$$S(1,2)=\frac{(2\beta)*(\text{distance between } P_1 \text{and} P_{1-\text{inter}-R})}{(\text{distance between } P_1 \text{and} P_2)} \qquad (3.21)$$

$$S(2,3)=\frac{-(2\alpha)*(\text{distance between } P_3 \text{and} P_{2-\text{inter}-L})}{(\text{distance between } P_1 \text{and} P_3)} \qquad (3.22)$$

$$S(2,4)=\frac{-(2\alpha)*(\text{distance between } P_3 \text{and} P_{2-\text{inter}-R})}{(\text{distance between } P_1 \text{and} P_3)} \qquad (3.23)$$

$$S(1,5)=\frac{(2\beta)*(\text{distance between } P_3 \text{and} P_{3-\text{inter}-L})}{(\text{distance between } P_3 \text{and} P_4)} \qquad (3.24)$$

$$S(1,6)=\frac{(2\beta)*(\text{distance between } P_3 \text{and} P_{3-\text{inter}-R})}{(\text{distance between } P_3 \text{and} P_4)} \qquad (3.25)$$

$$S(2,7)=\frac{-(2\alpha)*(\text{distance between } P_4 \text{and} P_{4-\text{inter}-L})}{(\text{distance between } P_2 \text{and} P_4)} \qquad (3.26)$$

$$S(2,8)=\frac{-(2\alpha)*(\text{distance between } P_4 \text{and} P_{4-\text{inter}-R})}{(\text{distance between } P_2 \text{and} P_4)} \qquad (3.27)$$

By substitution in the equation (3.19) from the equations (3.20) to (3.27) that represent all possibilities of the intersection points location points in the rectangular simulated image, and remove all empty element. Hence, the image matrix that mentioned above in equation (3.19) will be reduced to [2×4] or [2×2] in case of two lines or one line respectively has been detected.

The position of the lane lines in the virtual image and also their (slope and y-intercept) with respect to the virtual image axis can be computed.

Then the lateral position and orientation (θ_{HI} , D_{FI}) of the vehicle with respect to the center of roadway as shown in (Fig.8) and (Fig. 12) can be estimated by the following equations:

$$D_{FI}=\beta-X_C \qquad (3.28)$$

$$\theta_{HI}=\frac{\theta_1+\theta_2}{2} \qquad (3.29)$$

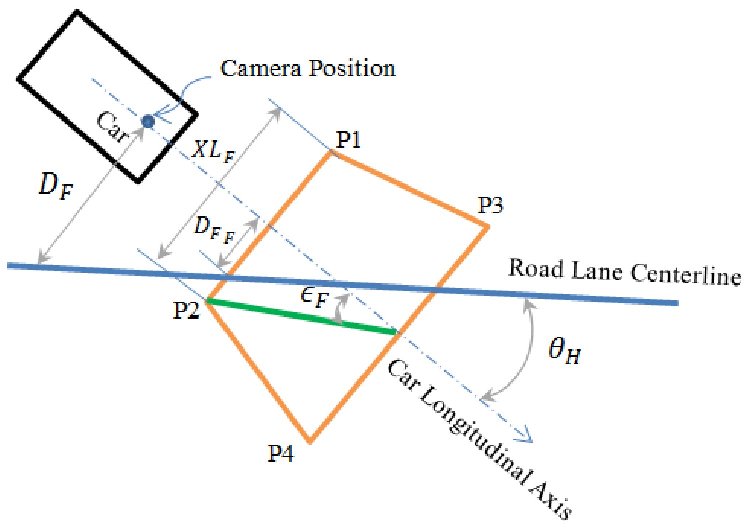

Fig. 11. Errors Associated with Vehicle in Frustum

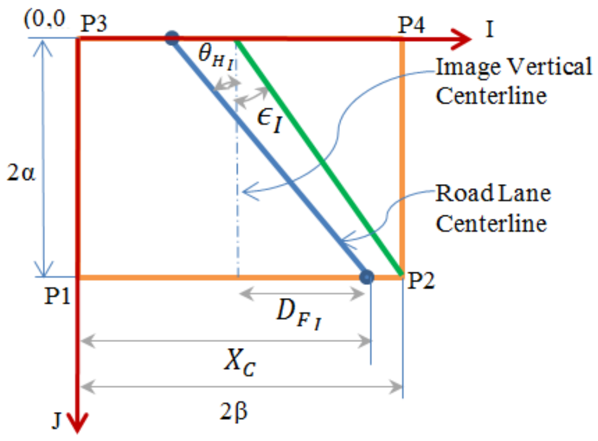

Fig. 12. Errors Associated with Vehicle in Simulated Image

From mapping and similarity between the trapezoid as shown in (Fig. 11) and rectangular image as shown in (Fig. 12), the heading angle and lateral offset distance can be estimated by (θ_{HP} , D_{FP}) as shown in (Fig. 13) by the following equations:

$$D_{FP} = \frac{(2\beta)\times D_{FF}}{XL_F} \qquad (3.30)$$

$$\theta_{HP} = \frac{\theta_H \times \epsilon_I}{\epsilon_F} \qquad (3.31)$$

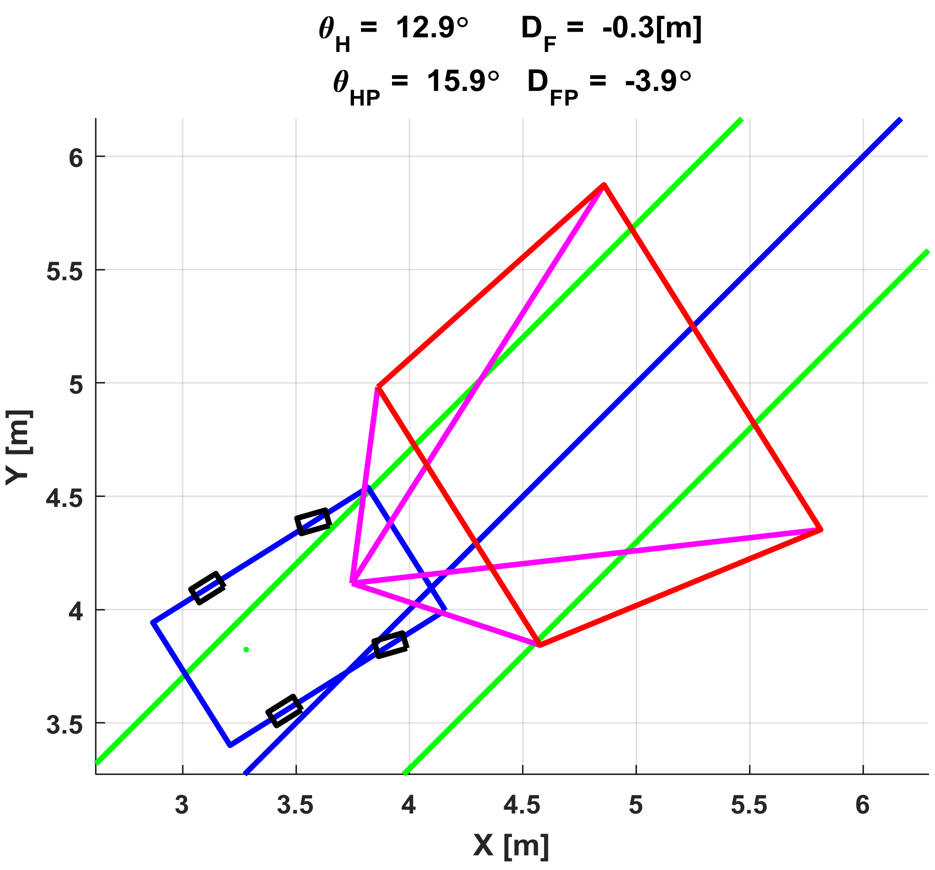

Fig. 13. Plotting the Vehicle and the Frustum w.r.t Road using MATLAB

By applying equations (3.28) and (3.29), The orientation and lateral offset (θ_{HI} , D_{FI}) can be calculated as shown in (Fig. 14)

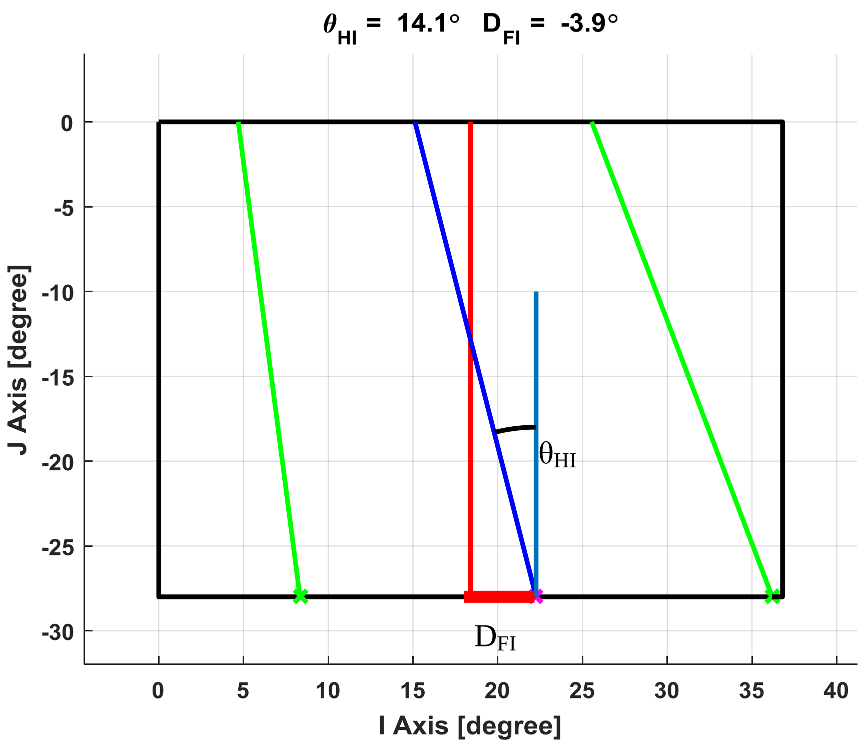

Fig. 14. Plotting the Simulated Image using MATLAB

5. Design of the Lateral Controller

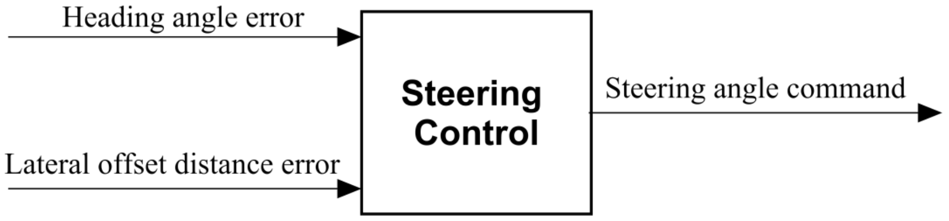

Fig.15. Control system block diagram

This section is concerned with the vehicle navigating along a specific trajectory. The steering control as shown in (Fig.15) is not only involves ensuring that the vehicle is in the correct position but also has the correct heading.
The lateral controller has to achieve two concerns:

- To ensure the vehicle is facing in the correct direction.
- To minimize the distance between the current position and the desired position (the lane middle).

From virtual image, we can obtain the desired position and heading w.r.t to the lane middle. The steering angle is set to the difference between the current vehicle heading and the road heading. This has the effect of ensuring the vehicle navigates a path that is parallel to the desired path.
In order to minimize the lateral offset between the vehicle and the desired path (the lane middle), the distance to the closest point on the trajectory is calculated. The previously calculated steering angle is then modified proportionally to steer towards the desired position.
The paper presents a simple controller, the vehicle steering control system uses successive loop closure and optimizes the control gains by optimization toolbox in MATLAB.
In order to implement the controllers designed in the model, a few variables must be known. These variables are the lateral deviation (D_F) and the error in the heading angle of the vehicle (θ_H), the steering controller is designed to minimize error.

5.1 Successive Loop Closure

The basic idea behind successive loop closure is to close several simple feedback loops in succession around the open-loop plant dynamics rather than designing a single (presumably more complicated) control system [8]
The controller is designed by successive loop closure technique as shown in (Fig.16). In all these controllers, the outer loops control the slower variables and the inner loops control the faster ones.
The outermost loop tries to rectify, through feedback, the error between desired and actual values of the slowest changing variable by computing through proportional blocks what the desired value of faster variable should be. The inner loops handle errors in the faster variables with the innermost loop dealing with the fastest variable.

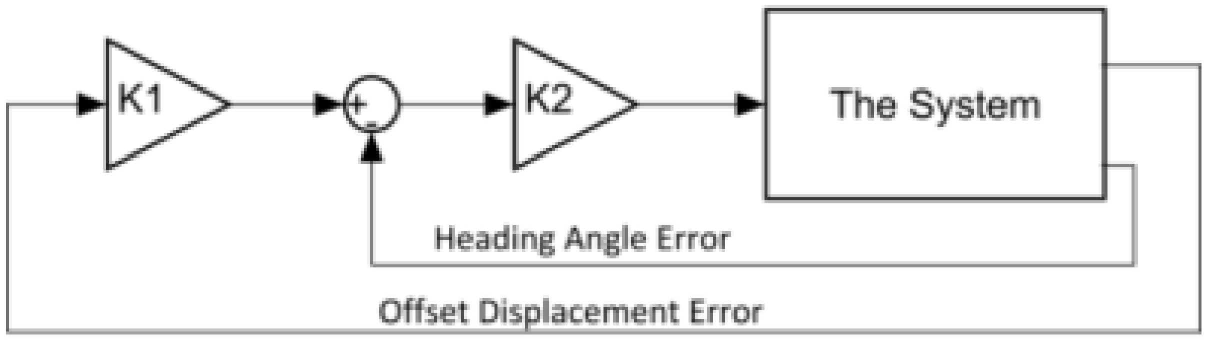

Fig.16. Successive Loop Closure Diagram

To illustrate how this approach can be applied, first, we are interested in controlling the output (heading angle error) by using its feedback without (offset displacement error); by tuning this step, we can get the compensator (K2) in succession.

Then, we apply the output (offset displacement error) by multiplying it with gain (K1) to make a difference with the output of (heading angle error); by tuning this step, we can get the compensator (K1) in succession.

5.2 Optimization Method

Mathematical optimization can be used to arrive at the best solution for a given problem in an efficient way. Optimization method by optimization toolbox in MATLAB [9] determines the control gains in the form of constants K1, K2. Optimization Toolbox provides functions for finding parameters that minimize or maximize objectives while satisfying constraints. The toolbox includes solvers for linear programming, mixed-integer linear programming, quadratic programming, nonlinear optimization, and nonlinear least squares. We can use these solvers to find optimal solutions to continuous and discrete problems, perform tradeoff analyses, and incorporate optimization methods into algorithms and applications [9]. To represent our optimization problem for solution, we generally put these steps:

- Create an objective function, the function we want to minimize.
- Create constraints.

6. Mathematical Simulation

This section describes the simulation that made the mathematical models (the vehicle and the vision models) that have been implemented in MATLAB/Simulink environment.

6.1 Error Calculation and Correlation Analysis

The error of (heading angle and offset displacement) is calculated in static condition without vehicle kinematic model and controller block at many random vehicle initial positions (X, Y, θ) in case of 2D and 3D vision simulation. Hence 2D model simulation is used to verify the proposed 3D vision model that mentioned in section (0).

2D model is a simple simulation that is only depends on vehicle and road position. The vision in 2D model is from top view with infinity field of view; Hence the 2D simulation does not have a simulation for the camera vision. Simply, in this simulation the vehicle position is a point and the road is a line, consequently the distance between point and line can be calculated simply by equation (6.2) and the heading angle can be calculated by equation (6.3). The objective of the error calculation is to get two important values (the correlation coefficient and the average value of the errors ratio) of the two cases 2D and 3D vision simulation.

6.1.1 Error in 2D Simulation :

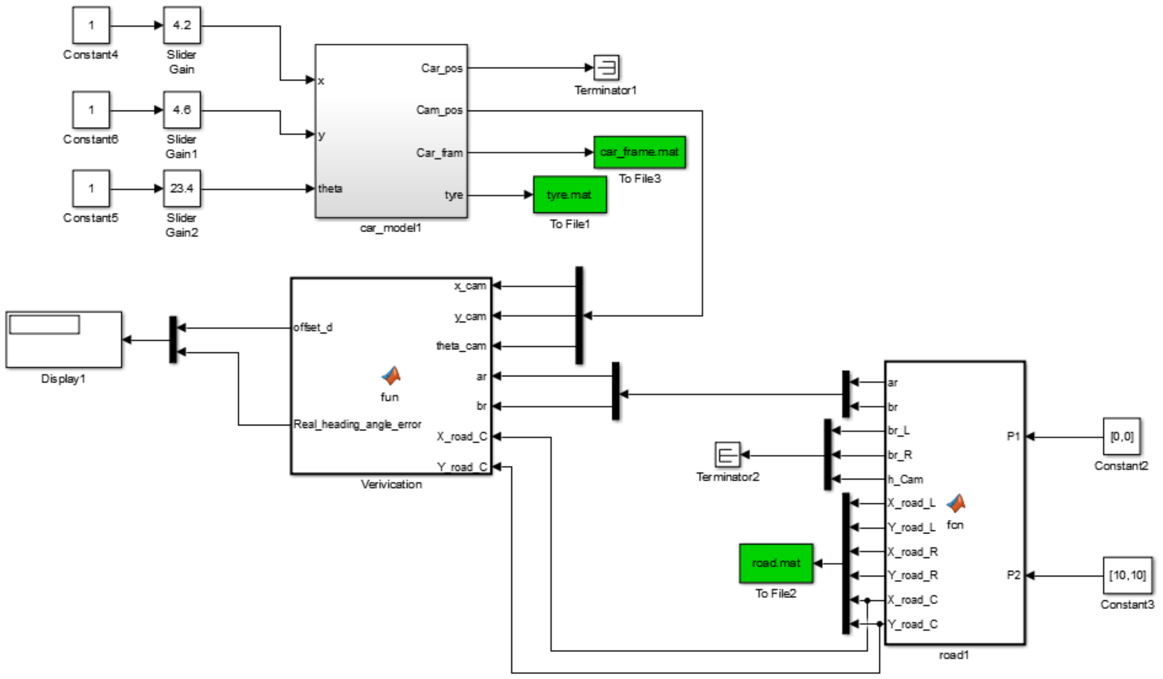

Fig.17. Error in 2D Vision Simulation

In (Fig.17) shows a simple simulation that is only depends on vehicle and road position, This simulation has three blocks (road position/cam position(X,Y,θ)/2D model calculation block). From this simulation we can get the real offset distance and required heading angle.
The perpendicular distance from a the location of the vehicle (point) to the road line
The distance from a point (m, n) to the line $Ax + By + C = 0$ is given as shown in (Fig.18):

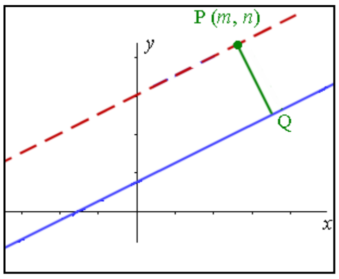

Fig.18. Distance from a point to the line

$$d = \frac{|Am + Bn + C|}{\sqrt{A^2 + B^2}} \tag{5.1}$$

$$D_F = \frac{\text{ar*x\_cam} + (-1)\text{*y\_cam} + \text{br}}{\sqrt{\text{ar}^2 + (-1)^2}} \quad [10] \tag{5.2}$$

Moreover, the heading angle can be calculated by:

$$\theta_H = \theta_{road} - \theta_{cam} \tag{5.3}$$

6.1.2 Error in 3D Vision Simulation :

This simulation as shown in (Fig.19) likes the previous simulation but the calculation block contains camera simulation (Vision System Model) that mentioned in section (0)
The 3D simulation depends on the vehicle position, road position, and camera (horizontal/vertical) field of view.
From two simulations in 2D and 3D vision, the errors of (heading angle and offset displacement) in static position (X, Y, θ) have been calculated without vehicle kinematic.

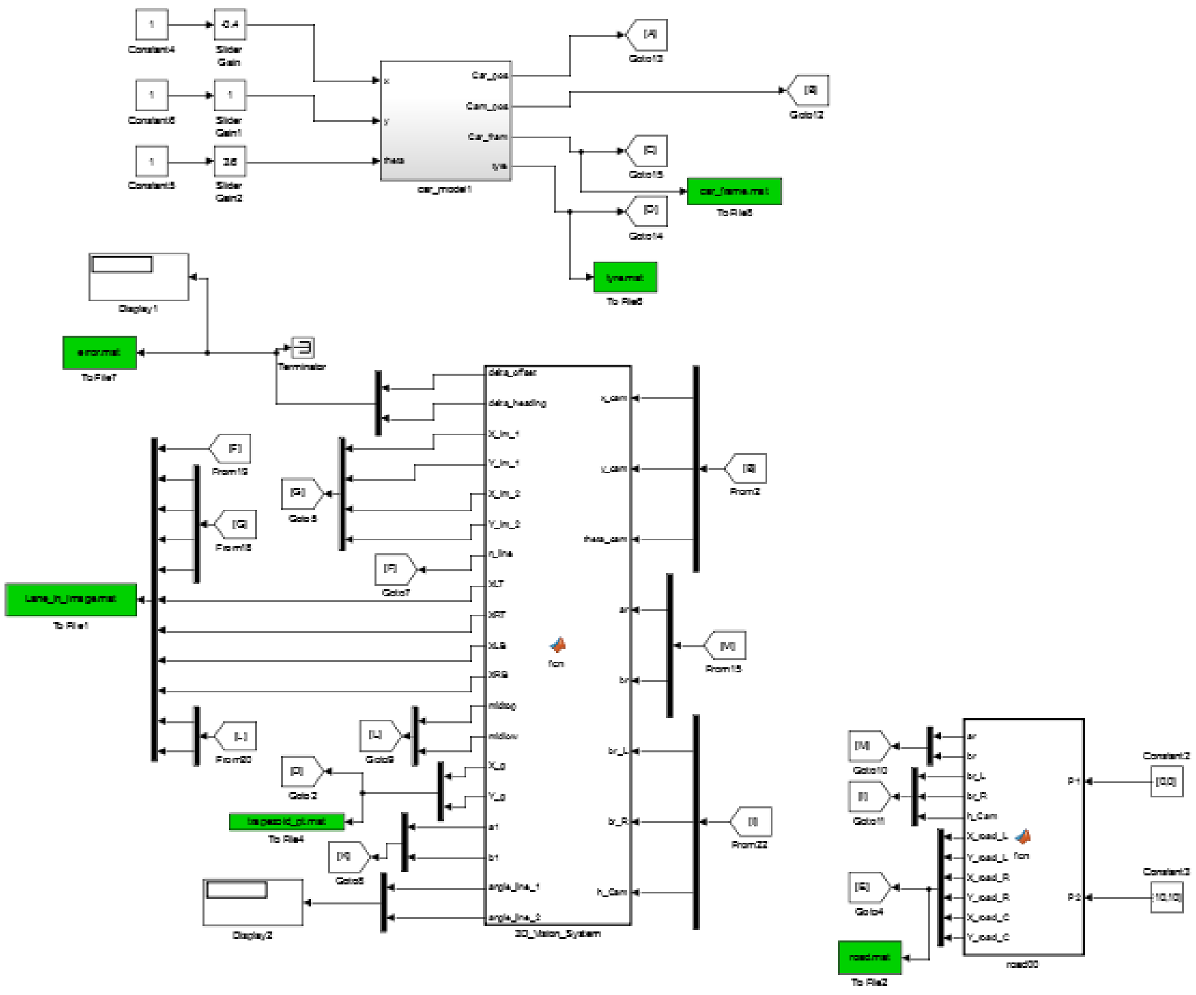

Fig.19. Error in 3D Vision Simulation

In addition, the ratio of the errors (heading angle and offset displacement) in 2D and 3D Simulation could be compute.

$$\text{Average of Offset Distance Errors Ratio} = \text{average}\sum\left(\frac{\text{offset distance in3D}}{\text{offset distance in2D}}\right) = 20 \tag{5.4}$$

$$\text{Average of Heading Angle Errors Ratio} = \text{average}\sum\left(\frac{\text{heading angle in3D}}{\text{heading angle in2D}}\right) = 1.1 \tag{5.5}$$

By multiplying these values by each value in (heading angle and offset displacement) of 2D vision and plotting the results, we get similar attitude from the 2D and 3D results that is shown in

Fig.20 and Fig.21 . These figures show that the heading angle errors and offset distance errors in case of (2D and 3D) vision have a strong positive relation this relation can be represented by correlation coefficient [11].

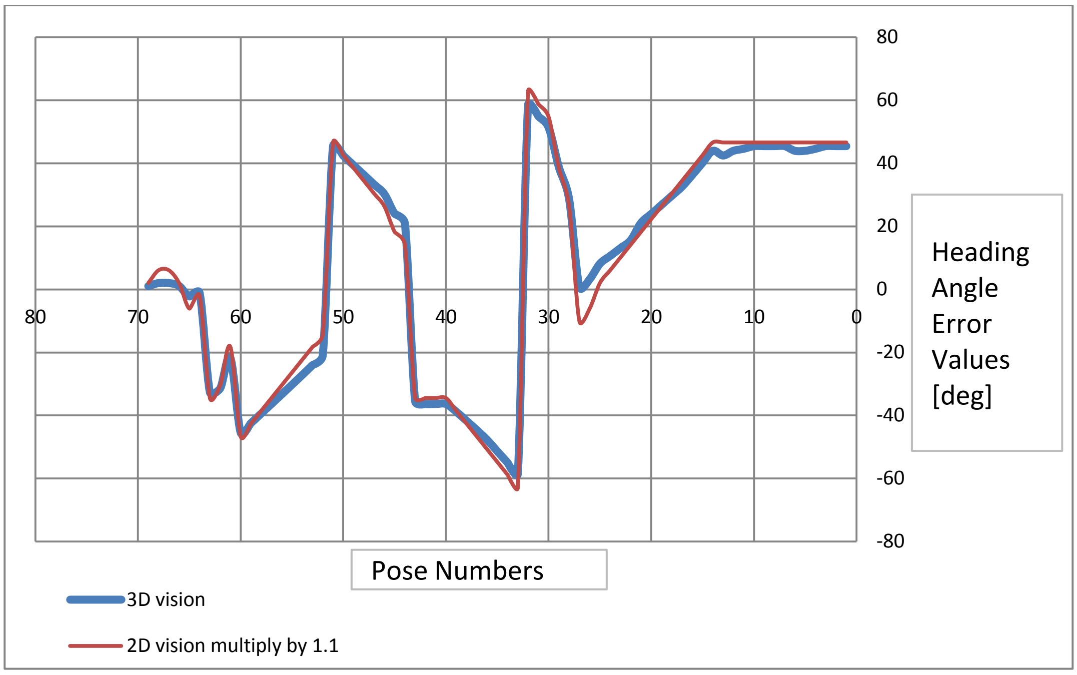

Fig.20. Heading Angle Error in 3D Vision vs. 2D Vision

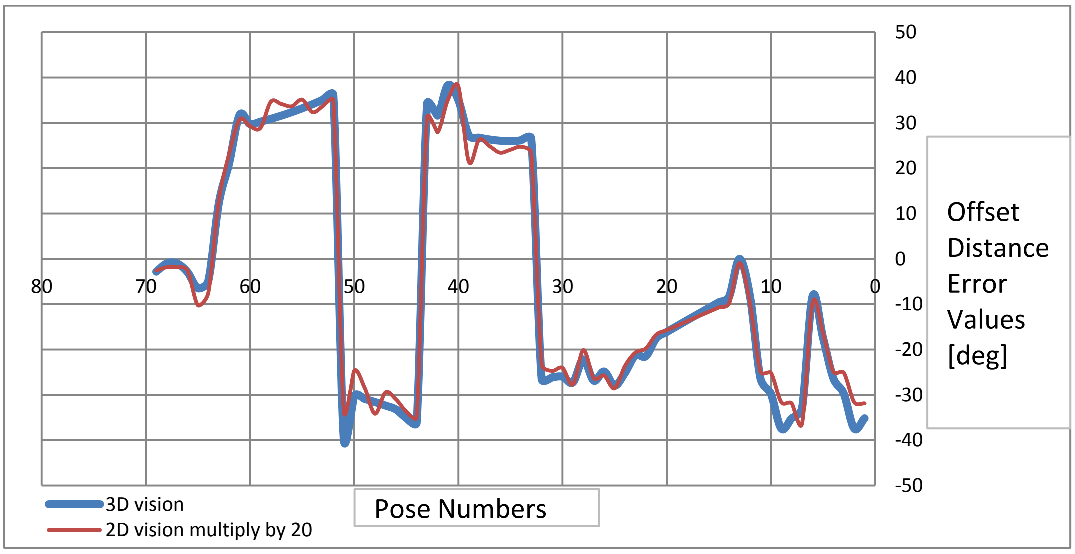

Fig.21. Offset Distance Error in 3D Vision vs. 2D Vision

The correlation coefficient is a mathematical representation of the mathematical relationship between two values or sets of data. The correlation coefficient is the statistical measurement of degree to which the change in one of the measurements affects the change in another set of measurements.
One of the most commonly used formulas in stats is Pearson's correlation coefficient formula.

$$r = \frac{n\left(\sum xy\right) - (\sum x)(\sum y)}{\sqrt{\left[n\sum x^2 - (\sum x)^2\right]\left[n\sum y^2 - (\sum y)^2\right]}} \tag{5.6}$$

where:
n is the number of pairs of data that equals 69 trails.
x and y are results of measurements that contain measurement error.
Correlation coefficient formulas are used to find how strong a relationship is between data. The formulas return a value between [-1: 1], where (1) Indicates a strong positive relationship

between the data, (-1) Indicates a strong negative relationship between the data, and (zero) indicates no relationship between the data.
By error analysis, we find that there is a strong positive relation between the 2D and 3D vision simulation results in (heading angle error and offset distance error).
Correlation coefficient for heading angle error = 0.99
Correlation coefficient for offset distance error = 0.98

6.2 Simulation in 2D Vision

In this simulation as shown in (Fig.22), the kinematic and control blocks that were mentioned in sections 0 and 0 have been added to the simulation that was mentioned in section 0.
The optimization method has been used by optimization toolbox in MATLAB [9] to determine the control gains in the form of constants K1, K2. To get smother and stable path for the vehicle to the lane, the objective function of the optimization method could be assumed as $600 * D_F{}^2 + \theta_H{}^2$

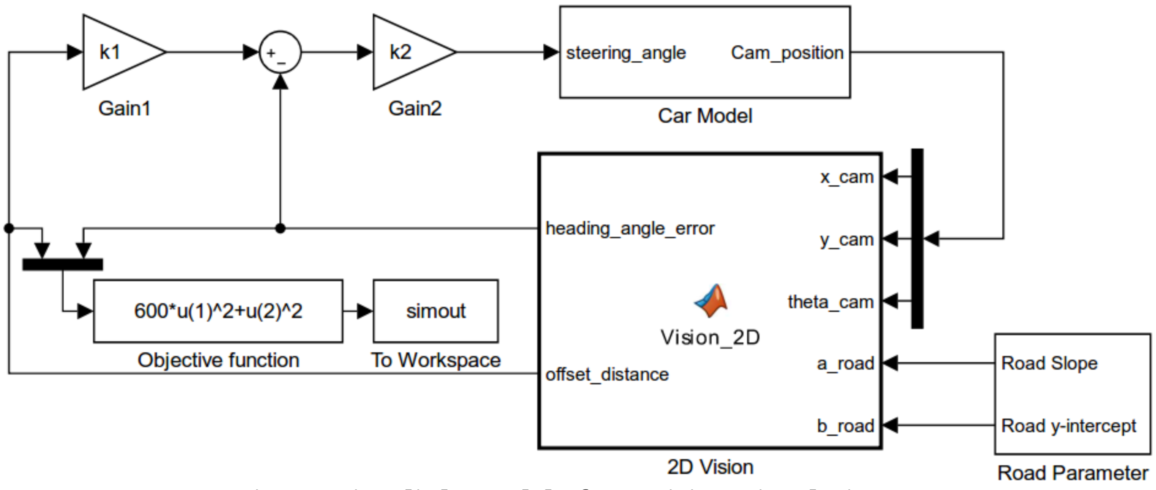

Fig.22. Simulink Model of 2D Vision Simulation

6.3 Simulation in 3D Vision

In this simulation as shown in (Fig.23), the same simulation of 2D vision model that is mentioned in section0) could be used with replacing its 2D vision block with 3D vision block that is mentioned in section (0).
By using optimization method in 2D vision simulation model, the control gains K1 and K2 could be determined, they have values -22.5205 and -0.4754 respectively; these gains are used in 3D vision simulation.

In addition to the average errors ratio of the (heading angle error and offset distance error) that calculated before in (section 0) is multiplied to 3D vision simulation output.

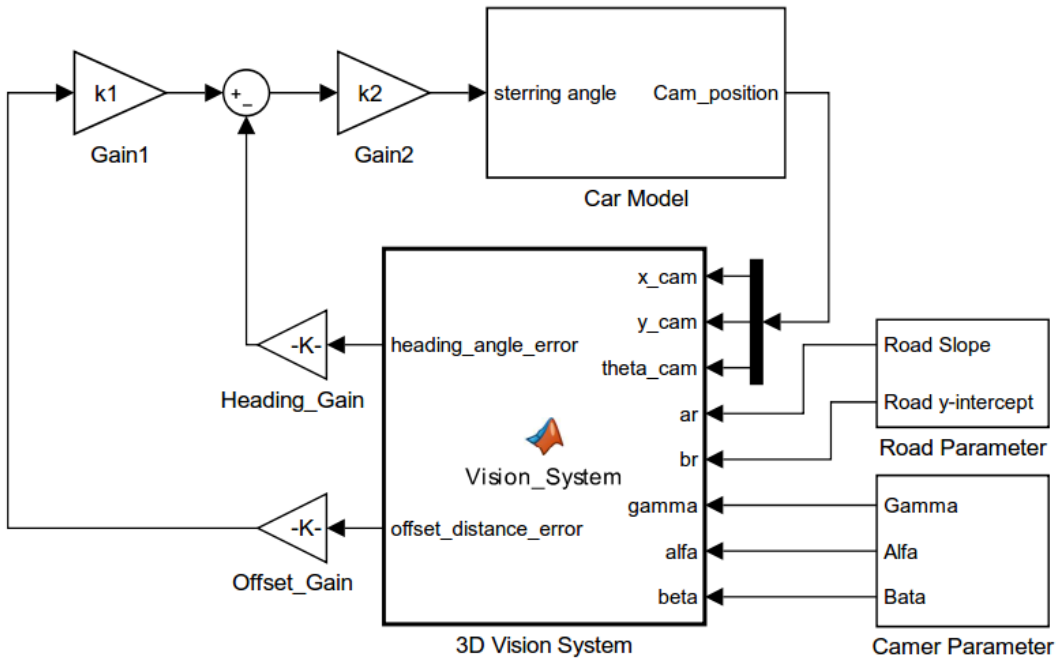

Fig.23. Simulink Model of 3D Vision Simulation

7. Simulation Results Analysis

The simulation model is performed for 2D and 3D simulation to monitor the expected response of the vehicle. In this simulation, the car and the camera parameter are chosen from [3] while the road width 96 cm and its angle with X axis is 45 degree

The initial condition for 2d and 3d simulation models are the same. The road is set little out of the camera field of view to check the 3d vision model in both cases by seeing the road.

Fig. 24 and Fig. 25 show heading angle error and offset distance error respectively versus time. From figures it's noticed that the value of heading angle and offset distance errors in 3d vision simulation model are zero from 0 to 0.7021 sec which attributed to the inexistence of lane lines in the simulated image because the road lane line is still out from the camera frustum. While as soon as lines get into the camera field of view, the errors become have values.

On the other hand, the errors in 2d vision simulation model read errors from time 0 sec, due to the assumption that the vision in 2D model is from top view with infinity field of view.

Fig. 26 shows the steering angle that comes out from controller. The steering angle in 2D model begins from a negative value, while in 3D vision model experienced sudden drop at time 0.7021 sec once the road became visible to the camera field of view and exist in the simulated image, these drop in values caused by the heading angle to go higher at this time and then converge to zero. This action is attributed to the controller that has two inputs and one output and produce gains come from optimization method.

Fig. 27 to Fig. 29 simulate the vehicle trajectory and show simple vehicle drawing and its front wheel steering angle in different position.

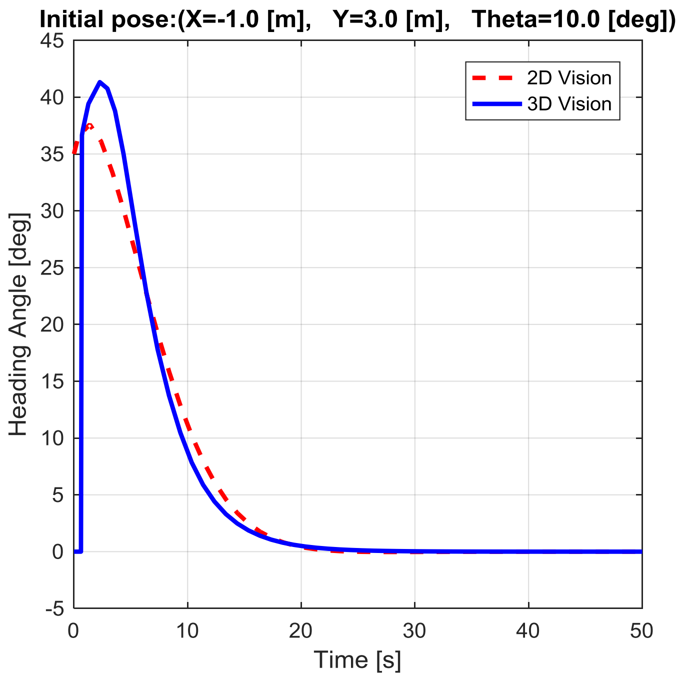

Fig. 24. Heading angle error vs. Time

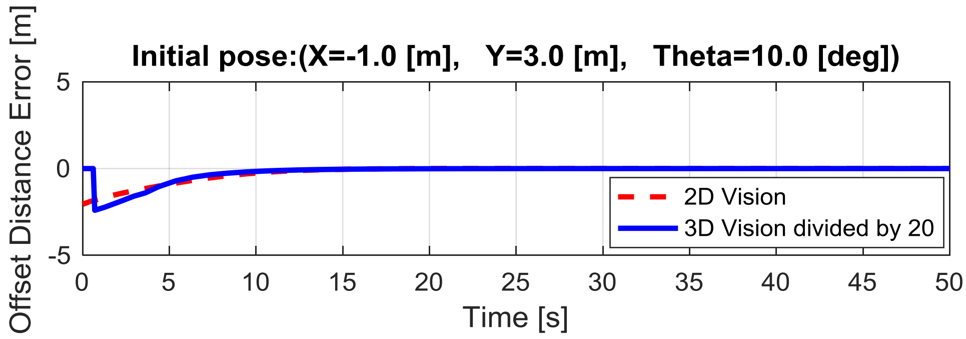

Fig. 25. Offset Distance Error vs. Time

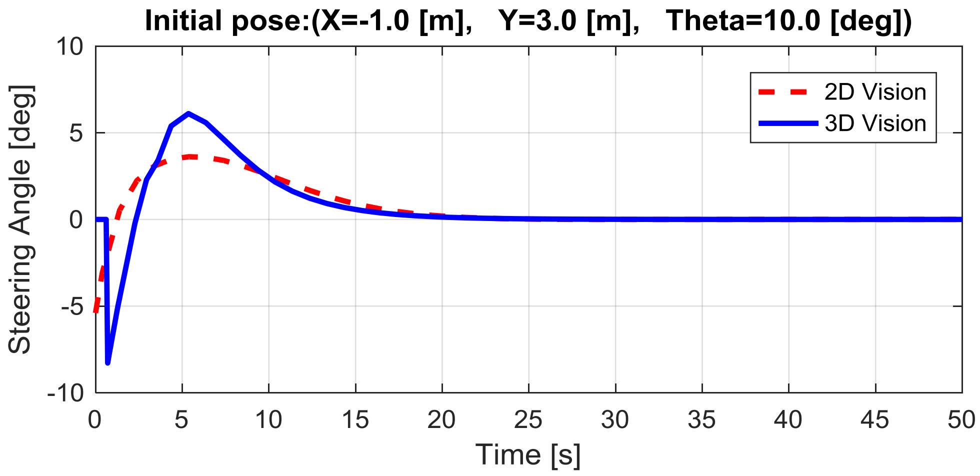

Fig. 26. Steering Angle vs. Time

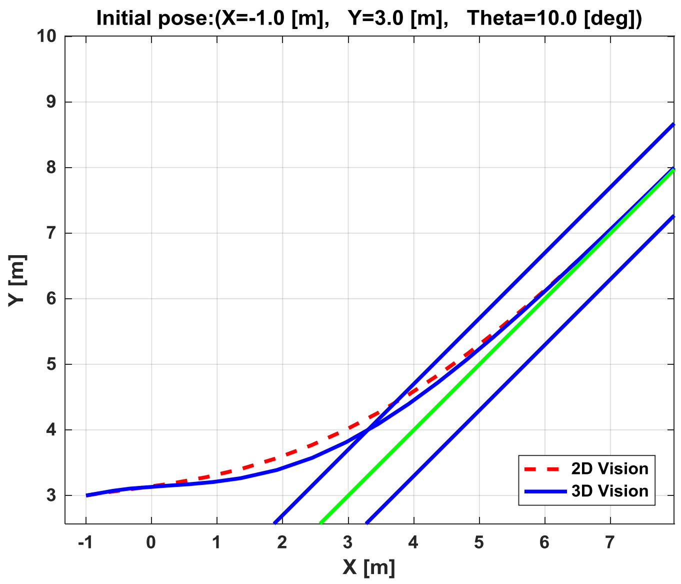

Fig. 27. Vehicle Trajectory

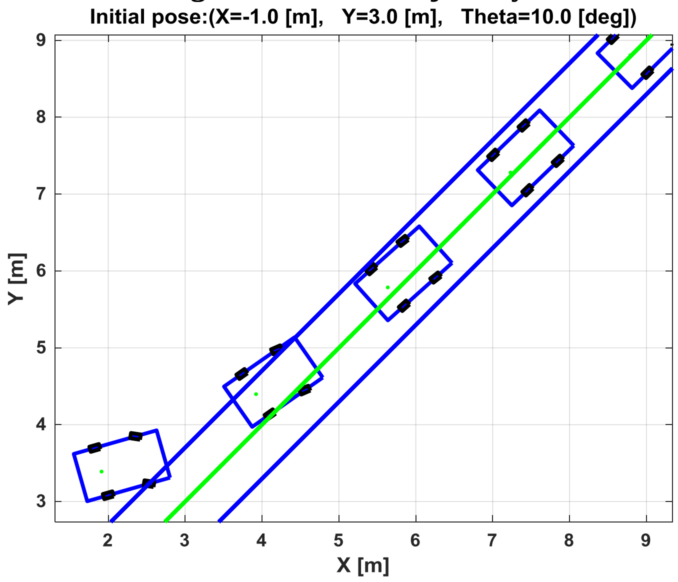

Fig. 28. Vehicle Trajectory with vehicle body

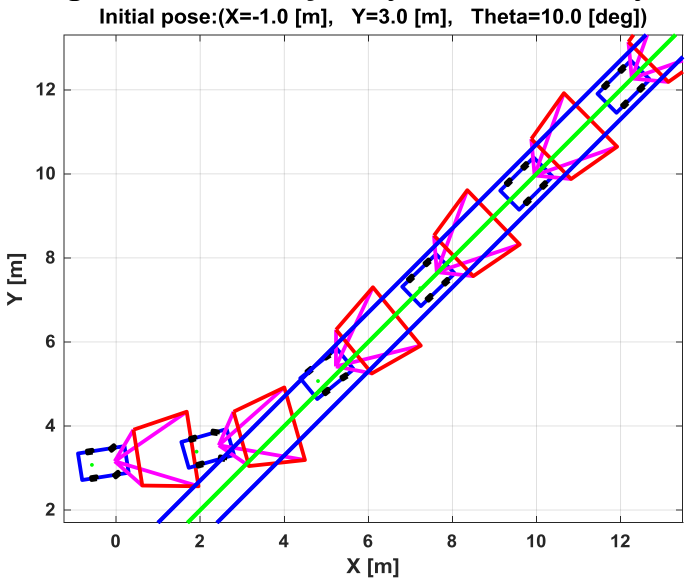

Fig. 29. Vehicle trajectory with vehicle and frustum

8. Conclusion

In this paper, a complete simulation model is developed to model vision-based autonomous vehicle. The simulation model enables finding the trajectory of the vehicle in different situations.

The objective of this simulation is to prove that a novel 3D vision model can determine the steering command for the vehicle lateral control.

9. References

[1] C. Rouff and M. Hinchey, *Experience from the DARPA urban challenge*: Springer Science & Business Media, 2011.

[2] E. Guizzo, "How google's self-driving car works," *IEEE Spectrum Online, October,* vol. 18, 2011.

[3] T. S. A. Al-Zaher, A. M. Bayoumy, A.-H. M. Sharaf, and Y. H. H. El-din, "Lane tracking and obstacle avoidance for Autonomous Ground Vehicles," in *Mechatronics (MECATRONICS), 2012 9th France-Japan & 7th Europe-Asia Congress on and Research and Education in Mechatronics (REM), 2012 13th Int'l Workshop on*, 2012, pp. 264-271.

[4] J.-W. Perng, Y.-H. Wen, G.-Y. Chen, and W.-J. Chang, "Intelligent Vehicle Driving Controller Design and Implementation Evaluation," *Journal of Mechanics Engineering and Automation*, vol. 2, pp. 422-430, 2012.

[5] R. Rajamani, *Vehicle dynamics and control*: Springer Science & Business Media, 2011.

[6] M. Bertozzi and A. Broggi, "GOLD: A parallel real-time stereo vision system for generic obstacle and lane detection," *IEEE transactions on image processing*, vol. 7, pp. 62-81, 1998.

[7] A. Borkar, M. Hayes, and M. T. Smith, "A novel lane detection system with efficient ground truth generation," *IEEE Transactions on Intelligent Transportation Systems,* vol. 13, pp. 365-374, 2012.

[8] R. W. Beard and T. W. McLain, *Small unmanned aircraft: Theory and practice*: Princeton university press, 2012.

[9] T. Coleman, M. A. Branch, and A. Grace, "Optimization toolbox," *For Use with MATLAB. User's Guide for MATLAB 5, Version 2, Relaese II,* 1999.

[10] H. Anton, *Elementary linear algebra*: John Wiley & Sons, 2010.

[11] B. Grewal, *Higher engineering mathematics*: Khanna Publisher, New Delhi, 1996.