Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 133 (2018) 839–846

Procedia Computer Science

www.elsevier.com/locate/procedia

International Conference on Robotics and Smart Manufacturing (RoSMa2018)

Vision Based Intelligent Vehicle Steering Control Using Single Camera for Automated Highway System

P.V.Manivannan[a], Pulidindi Ramakanth[b]

[a,b]Department of Mechanical Engineering,IIT Madras, Chennai-600036, India

Abstract

Generally, stereoscope vision is used for guiding autonomous intelligent road vehicles, as image depth can be calculated easily. However, when one of the camera fails, it will advantages to have suitable guidance algorithm that can detect the lane marking using single camera. The primary objective of this work is to develop and implement control algorithms for identifying and guiding the intelligent road vehicle in the assigned lane using image-processing techniques, using single camera. It deals with dividing the video being taken by the camera, into several number of frames. Then, the obtained frames are processed using Image acquisition techniques in Matlab. It detects the lane to be followed through image processing, calculates the angle of movement required for the robot to stay in the assigned path and commands the steering system with appropriate control algorithms. The identification of lane is done using color detection and boundary marking techniques. The image initially undergoes Inverse Perspective Mapping (IPM) for viewing the 3-D space in a 2-dimensional array. Once the IPM of the acquired image undergoes lane detection algorithm, the angle of any curve in the lane is detected using angle detection module. The robot is also made to move at the center of the path using Fuzzy Logic algorithms, where it takes distance from the left detected edge as the input and outputs the angle to be moved by the robot to stay in the center. The developed algorithm has been tested using P3-DX Pioneer mobile in laboratory condition. The experimental result obtained show that the algorithm used by the vehicle to follow the center of the lane, with any given angle or position of initiation, has worked all the times under given experimental conditions, with a maximum error of 2cm.

Keywords: Automated Highway System, Intelligent Vehicle, Inverse Perspective Mapping, Fuzzy Logic

1. Introduction

Automated Highway System (AHS) is a system, which collects information from sensors installed on roads & vehicles and uses that information to drive vehicles with minimal or no human intervention. Lane tracking forms one of the crucial part of the system, as the entire time of the process cycle depends on the time for the image processing cycle. In addition, Autonomous Guided Vehicle (AGV) is of great interest in various fields such as defense and manufacturing industry. They increase efficiency and reduce costs by helping to automate parts movement inside the manufacturing facility or warehouse. Navigating the robot using Light Detection and Ranging (LiDAR) is very well

10.1016/j.procs.2018.07.111

understood, developed and deployed. However, using cameras give additional flexibility of being selective as color of the different type of lane marking can be used to distinguish on specific path from another. Moreover, by transmitting the video to a remote location, the targets can be monitored efficiently by a remote host, which is free from any threat. Their first automated vehicle was built in 1962. There is progress in this research from then onwards. T.M. Jochem, C.E.Thorpe, et al, (1993) have used neural net approach with a large database of training images to automatically follow roads without any explicit feature recognition for the Autonomous Land Vehicle in a Neural Net (ALVINN) system. The POSTECH Road Vehicle (PRV II) developed by K.I.Kim, S.Y.Oh, et al (1995) uses the concept of color cameras and the assumption that the road will be homogeneously colored to extract the road region from images even on completely unstructured roads.

The Generic Obstacle and Lane Detection (GOLD) system implemented by M. Bertozzi and A.Broggi (1998) exploits the flat, planar road assumption to map the image into another domain representing a bird's eye view of the road (takes out the perspective like the RALPH system). According to research work of B.Southall and C.J.Taylor, (2001), the feature extraction is done by thresholding both pixel values and cross-correlation to dark-bright-dark function. S.Nedevschi, R. Schmidt, et al (2004) work presents a 3D lane detection method based on stereovision. The stereovision algorithm allows the elimination of the common assumptions: flat road, constant pitch angle or absence of roll angle. The lane is modeled as a 3D surface, defined by the vertical and horizontal clothoid curves, the lane width and the roll angle. The detection results are used to update the lane state through Kalman filtering. The monocular lane-vehicle detection and tracking system of King Hann, Li-Minn, et al (2009) deals with lane boundary detection, lane region tracking, and vehicle detection with a proposed vertical asymmetry measurement. It uses Inverse Perspective Mapping technique to remove perspective effect and reconstruct a 3-D mapping when the camera parameters and knowledge about the road are known. Autonomous vehicle guidance using stereoscopic vision and single camera has been successfully carried out by (Sristi Sundram, et al., 2013) and (Aakash Gupta, et al., 2013).

The autonomous robot navigation is achieved through wall following using ultrasonic sensors and image processing using a simple webcam (Alpha Daye (Dialloa, et al., 2015). They have used bitwise image processing comparison method introduced is writing in OpenCV and runs on the Raspberry Pi embedded controller. Thus, from the lane detection techniques mentioned above, under the assumptions of having a constant width path and a flat road, a monocular vision is good enough to obtain the lane tracking using the control algorithms. It provides us a simple solution in tracking the lane and thereby, in the motion of the vehicle along the assigned path. Where as in other situations where there is an occlusion of the road and the lane markers due to cars and other obstacles; and also the existence of yaw, pitch and roll on the road, a stereoscopic vision is necessary. So, the limitations of the project has made monocular vision sufficient for this project.

2. Digital image processing

Image Acquisition Toolbox provides graphical tools and a programmatic interface to help you work with image acquisition hardware in MATLAB. Image Acquisition Toolbox automatically detects compatible image and video acquisition devices. Matlab has inbuilt video adapters which are essential for accessing the image acquisition devices. The installed adaptors in Matlab can be known by typing the *imaqhwinfo* in the command terminal. The image captured can be stored in any format. While reading an image, the color spaces can be changed using functions like rgb2gray(). It converts RGB image to gray scale image. Here, the default YUY format has been converted to RGB format for further image processing. This system is coded in such a way that it acquires frames from the video, taken by the camera, for every 0.5 seconds and processes these images further.

Line detection algorithm, code has to identify a color line of any given RGB value (if any) in the image. This is accomplished by driving the problem into two parts. The first part is to separate the given color from the rest of the color and the second part is to identify lines in the residual image. Here, the color is chosen to be yellow. Color detection algorithm is used to separate yellow color from the rest of the colors. The image is read into Matlab using imread() command, as shown in Figure 1. The tolerance can also set in accordance with how close the user wants the colors to match. After the image is read by the system, the respective RGB values of the color combination are identified. All the pixels in the image with the same RGB values are converted to white, to distinguish these required pixels from the rest as shown in Fig.3.4.b). Boundary detection algorithm Then the image is converted from RGB color space to gray scale using the command rgb2gray(). This grayscale image is converted into a binary image using the command BW = im2bw(I, level). The output image BW replaces all pixels in the input image with luminance greater than level with the value 1 (white) and replaces all other pixels with the value 0 (black). Specify level in the range [0,1]. This range is relative to the signal levels possible for the image's class. Therefore, a level value of 0.5 is midway between black and white, regardless of class. If the input image is not a grayscale image, the Matlab routine (im2bw) converts the input image to grayscale, and then converts this grayscale image to binary by thresholding.

To identify the yellow colored lane of original image (Figure 1) as a single line, the boundaries of the required color combination, detected as white color in the gray image, are traced using the command *B = bwboundaries(BW).* This command traces the exterior boundaries of objects, as well as boundaries of holes inside these objects, in the binary image BW. The BW image must be a binary image where nonzero pixels belong to an object and zero pixels constitute the background. The boundaries detected in the image are converted into a single line by marking the line approximately along the middle of the boundary. The unnecessary lines are omitted from the image by providing a minimum length for the lines to be detected. This way, the required colored lane is detected as a single line in a given image, as shown in Figure 2.

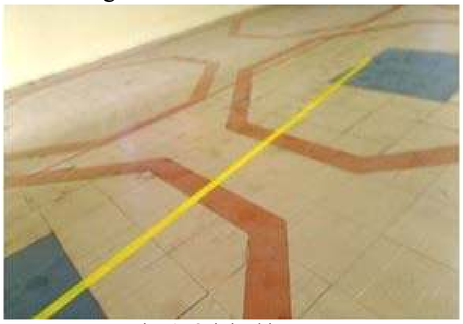


Fig. 1. Original image

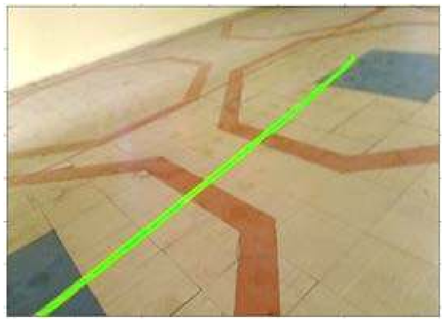


Fig. 2. Detected line boundaries

3. Theoretical background

The angle of view under which a scene is acquired and the distance of the objects from the camera (namely the perspective effect) contribute to associate a different information content to each pixel of an image. The perspective effect, in fact, must be taken into account when processing images in order to weigh each pixel according to its information content. To cope with this problem a geometrical transform Inverse Perspective Mapping (IPM), has been introduced. It allows to remove the perspective effect from the acquired image, remapping it into a new 2-dimensional domain (the remapped domain) in which the information content is homogeneously distributed among all pixels, thus allowing the efficient implementation of the processing steps. Obviously the application of the IPM transform requires the knowledge of the specific acquisition conditions (camera position, orientation etc.) and some assumption on the scene represented in the image. Thus the IPM transform can be of use in structured environments, where, for example, the camera is mounted in a fixed position or in situations where the calibration of the system and the surrounding environment can be sensed via other kind of sensor. Assuming the road in front of the vision system is planar, the use of IPM allows to obtain a bird's eye view of the scene. (AnuarMikdadMuad, AiniHussain, et al., 2004). Two equations (1) and (2), were derived using the triangulation and trigonometry of the IPM model.

$$u(x,0,z) = \frac{\gamma(x,0,z) - (Y - \alpha)}{\frac{2\alpha}{n-1}} \tag{1}$$

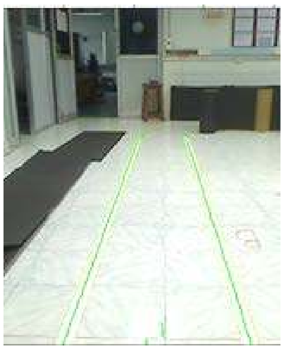
$$v(x,0,z) = \frac{\theta(x,0,z) - (\Phi - \alpha)}{\frac{2\alpha}{m-1}} \tag{2}$$

where, $\gamma = \tan^{-1}\frac{z}{x}$

$$\theta = \tan^{-1}\frac{h}{\sqrt{\left(x^2+z^2\right)}}$$

Y - angle between the projection of the optical axis on the flat plane
Φ- angle between the optical axis and the horizon,
A - camera angular aperture
n x m - resolution of the image in pixels.

The following are the few assumptions needs to be considered for using this method: The road in front of the vision system is planar and the camera is properly placed. Correct setting of yaw (γ) and tilt (θ) angle should be ensured in order to obtain accurate values of measurement. In this method, it is necessary to find the area under observance in the camera and also the real world coordinates for the same. The algorithm takes the image as an input and gives the IPM of the image as an output. For this purpose, it is needed to the find the real world co-ordinates of the target, which is done manually. To achieve this, the vehicle is placed at the center of the lane and image of the target is taken. The edges of the target are marked and their pixel coordinates are matched with points in the real world. The distances between the points are measured using a measuring tape manually, and they are fed into the code. The Matlab software has an inbuilt function, T = maketform (transformtype,...), which creates multidimensional spatial transformation structure called a TFORM struct. The following code, T= maketform ('projective',U,X), which builds a TFORMstructT for a two-dimensional projective transformation that maps each row of U to the corresponding row of X, has been used to obtain the inverse perspective image of the input. The U and X arguments define the corners of input and output quadrilaterals, which indicates the shape of the target. The limitation here is that, no three corners can be collinear. The following images, 3.a) and 3.b), indicate the functioning of IPM in Laboratory.


(a)

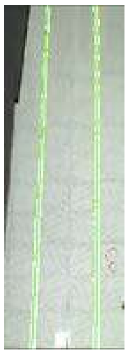

(b)

Fig. 3. Region of interest (a) Lane detected; (b) IPM of the image

The resolution of the video being taken is 640x480 pixels. The camera is placed at a height of 82 cm from the ground with an angle of depression about 36 degrees. Slight adjustments in the view, regarding centering of the image, are set using the option, Region of Interest (ROI). The region of interest considered for this project is as following: X - Offset = 0 pixels; Y - Offset = 350 pixels; Width = 560 pixels; Height = 530 pixels.

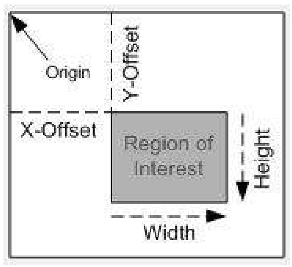


Fig. 4. Region of interest

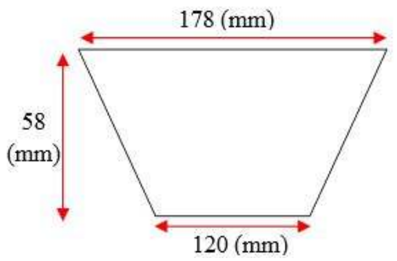


Fig. 5. Real world of lengths the image

Each term is related to the size of the video as shown below in the Figure 4. This ROI is mapped to the real world coordinates where the values of lengths measured are as shown in Figure 5. Usage of single camera to detect the lane: Instead of having a stereoscopic vision for detecting the lane marking far ahead of robot position, "Inverse Perspective Mapping" has been used, which makes a projective transform of the image. This transformation is used to change the captured images from a camera perspective to a bird's-eye view. In addition, this transformation enables a mapping between pixels in the image plane to world co-ordinates (meters). The abscissa of bird's eye-view denotes the width of view and the ordinate denotes the distance. The 'Region of Interest' can be set accordingly by denoting the necessary.

4. Algorithm for guiding robot

This control algorithm considers two cases i.e. a curved motion and a straight line motion. It involves the calculation of angle of movement for the vehicle, in case of a curved motion and involves moving the vehicle along the center of the path, in case of a straight line motion. To make robot to follow the center of the path during a straight line motion, Fuzzy Logic control algorithm has been used for the optimal steering of the robot. Fuzzy logic is a form of many-valued logic or probabilistic logic; it deals with reasoning that is approximate rather than fixed and exact. The left detected edge of the path is considered as the datum line for this algorithm. The distance from the left detected line is given as an input for the fuzzy logic and the output produced is the angle that needs to be rotated by the vehicle to maintain itself at the center of the path. The input variable 'distance' is considered as a Gaussian membership function and the output variable 'turn' is considered as a triangular membership function. The maximum/minimum angle of turn is obtained, since the distance of the robot from the center and the distance of area under observation from the robot are known from the measurements. The following calculations are made:

- The distance of area under observation is 110 cm.
- The distance of the robot from the center when placed at the edge is 11 cm.

So the maximum angle of turn from the any of the edges is 5.71 degrees ($90 - \tan^{-1}(110/11)$. The sign indicates the left or right side motion of the robot. Therefore, +5.71 degrees and -5.71 degrees are the maximum and minimum limits respectively, for the output value of 'turn' from the Fuzzy logic. To guide the robot on a curved path, the present algorithm under discussion assumes the borders or the boundaries of the path as obstacles. The aim of this algorithm is to avoid these obstacles. It inputs an image, calculates the turn required for the movement of the robot in the lane and outputs its value in terms of degrees. Assumption: It works on the assumption that the width of the lane is equal all through the path. To accomplish this algorithm, three images of the path are taken from the camera. The first image is taken by placing the robot at the center of the path (Fig.6.a). The second one is taken by placing the robot at the leftmost edge of the path (Fig.6.b) and the third one, from the rightmost edge of the path (Fig.6.c). Then the bird's eye view of these the three images captured, are observed. The following are the images captured from Laboratory.

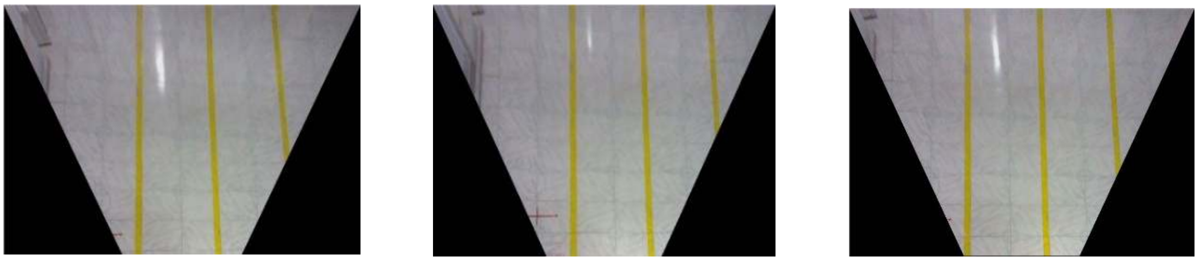


Fig. 6. Bird-Eye views (a) Image from Center; (b) Image from leftmost edge; (c) Image from rightmost lane

As can be observed from the above images, there is shift of lane towards right in the image taken from the leftmost edge, when compared with the one taken from the center. Similarly, the lane shifts towards left in the image taken from the rightmost edge. This forms an important observation for this algorithm. Each of the figures is divided into two halves, the left side and the right side. The left border of the path in the image taken from the leftmost edge forms a limit for the left side edge of the path. Similarly the right border of the path in the image taken from the rightmost edge forms a limit for the right side edge of the path. On considering these limits, lanes and overlapping all the three images, the final image obtained is shown in the Fig.7.

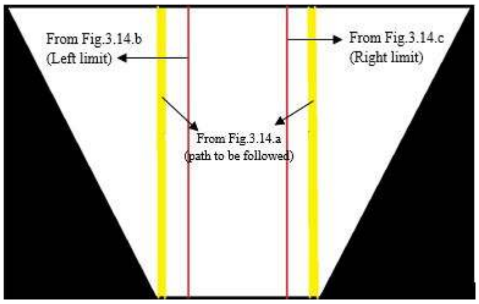

From Fig.3.14.b (Left limit)
From Fig.3.14.c (Right limit)
From Fig.3.14.a (path to be followed)

Fig. 7. Image obtained after overlapping the bird eye view images

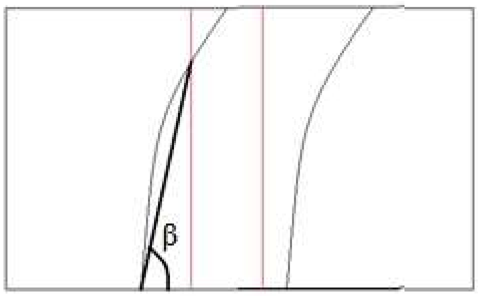


Fig. 8. Angle measurement

When any edge of the path intersects with the limit, a line is considered from the initial point of the edge to the intersection point of the edge with the limit. Calculating the angle (β) of the line with the horizontal, gives the angle of movement for the robot as shown in Figure 8. Since, the length of the Region of Interest under observation is very small, the lane curvature can be approximated as a straight line at an angle. The following measurements are made from the Figure 7:

- The limit for the leftmost edge is measured to be at 78 cm.
- The center of the lane is measured to be at 89 cm.
- The limit for the rightmost edge is measured to be at 100 cm.

The above measurements show that the distance between the limits is 22cm (100 – 78). The width of the P3-DX is 38 cm and the width of the lane in real world coordinates is 60 cm. The above measurement of distance between limits can be confirmed by deducting the width of the robot from the width of the lane i.e. 60 – 38 = 22cm. Since both these measurements show the same value, the initial assumption in IPM that there must be correct setting of yaw (γ) and tilt (θ) angle in order to obtain accurate values of measurement, is satisfied.

5. Experimental investigations

Fig. 9. Pioneer P3-DX Mobile Robot

Pioneer P3-DX mobile research robot, shown in Fig.9, is used for evaluating the developed algorithm. It works under the client-server environment. The low level details of mobile robotics are managed by servers embodied in the operating system software (ARCOS, AROS, P2OS) of the robot's micro controller. The client software that provides the high level control must run on a computer connected to the micro-controller. This can either be the on-board PC (that communicates with it directly through a serial RS-232 connection), or via a remote networked PC, which requires a server program to be running on the robot PC, providing the communication link between the remote PC and the micro-controller. This software is provided by the Pioneer manufacturers, MobileRobots (formerly ActivMedia) and can be used to control any of their models, e.g. AmigoBots, PeopleBots and Pioneers. ARIA is an object-oriented applications-programming interface, written in C++ and intended for the creation of intelligent high-level client-side software. It is essentially a library for C++ programmers. In addition, a programmer's own "action" classes may inherit from the base ArAction class. These classes run their own thread with the robot's current action being determined by an action resolver. This facilitates easy creation of a subsumption like architecture, (although this methodology does not have to be followed). ARIA is therefore technically architecture dependent. To transfer of value from Matlab to C++, the value of 'turn' obtained from Matlab is sent to a .yml file. It stores this value until the next image is taken. This .yml gets updated after a new value of turn is obtained. A Matlab parser is written to send the value from Matlab to .yml file. Using the XML/YAML file storages provided by the OpenCV class Filestorage, the variables from the .yml file are loaded to the C++ code for running the robot.The objective of the control algorithm is to guide the robot along the assigned path (which is determined by lane detection algorithm) using the 'turn' values sent to the vehicle. When the motors are enabled, P3-DX takes the current position as the origin and is oriented towards the positive x direction. The robot initiates and continues its movement at a constant speed of 100mm/s.

In this algorithm, Filestorage::READ command is used to read the contents in the .yml file located in the computer, which indicates the value of 'turn angle'. There are different sets of rotation actions and timers framed for different values of turn angle. For angles less than 10 degrees, robot.setDeltaHeading() has been used for the motion of the robot. For angles more than 10 degrees, robot.setRotVel() has been used for the motion of the robot. This is because, the usage of robot.setDeltaHeading() for larger angles, might result in jolting motion of the robot. To transfer of value from Matlab to C++, the value of 'turn' obtained from Matlab is sent to a .yml file. It stores this value until the next image is taken. This .yml gets updated after a new value of turn is obtained. A Matlab parser is written to send the value from Matlab to .yml file. Using the XML/YAML file storages provided by the OpenCV class Filestorage, the variables from the .yml file are loaded to the C++ code for running the robot. Experiments are conducted to test the motion of the robot under various conditions. The results are obtained for starting the motion from various positions along the width of the path and in various angles on the path. The following are the graphs which indicate the motion of the robot in various conditions.

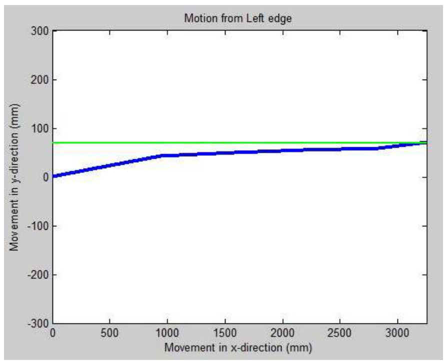


Fig. 11. Vehicle motion from lane left edge

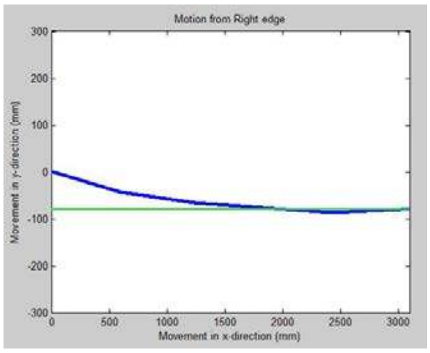


Fig. 12. Vehicle motion from lane right edge

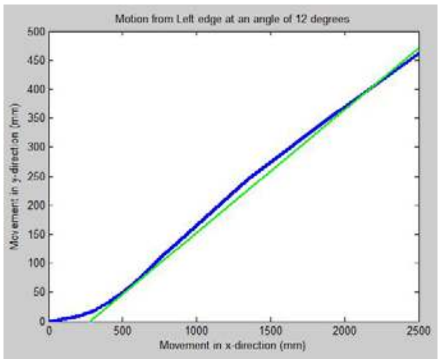


Fig. 13. At left edge with a heading 12 degrees towards the lane center

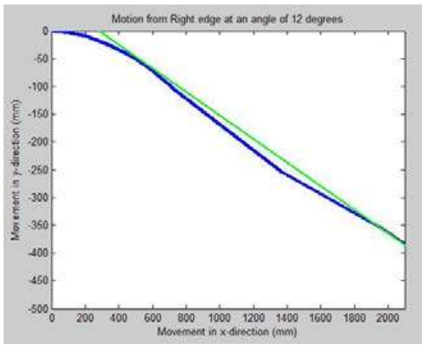


Fig. 14. At right edge with a heading 12 degrees towards the lane center

The following statements can be made based on the above results:

1. The large variation in the error, when the case of taking a turn on a constant width path is compared with the case of taking a turn on varying width path, indicates that the width of lane should be maintained constant all through the path for better functioning of the algorithm.
2. Advancements in the alignment of the camera has reduced the error by about 30 percent as clearly indicated in Fig. 5.8
3. The algorithms tested on a curved path showed results that initiated the curve motion, but did not progress further along the path. This is because the camera was unable to sense the path, further, after its initial turn. Hence, this code is limited to the paths with no sudden consecutive turn angles.
4. The light focused should be in such a way that, it produces no glare along the path. This reduces the occurrence of unnecessary boundary lines along the path.
5. Initial position of the robot should be placed on the path and should view both the margins of the lane

Conclusions

A vision based intelligent vehicle steering control was developed. The image acquisition and Color detection modules are developed in Matlab software were used marking boundaries and thus, leading to lane detection. These modules make use of the inverse perspective mapping for calculating the angle of movement which is sent to the C++ code with built-in ARIA libraries. The guiding module for the robot has been developed in OpenCV software in C++ programming language. A control algorithm measures the value of turn required for the motion of the robot. This computed value of turn is fed into the robot through the C++ programming. The entire integrated module has been tested on P3-DX mobile robot. From the experimental result obtained, following conclusions are made: The algorithm used by the vehicle to follow the center of the lane, with any given angle or position of initiation, has worked all the times under given experimental conditions, with a maximum error of 2cm. Advancements in the alignment of the camera has reduced the error by about 30 percent in the motion of the robot. The error, in case of lane with varying width, is much higher than the error, in case of lane with constant width. This is direct implication of the basic assumption on which the algorithm for movement of the vehicle along a turn is built.

References

[1] T.M. Jochem, C.E. Thorpe,D.A. Pomerleau,"MANIAC, A next generation neurally based autonomous road follower", in Proceedings of the Third International Conference on Intelligent Autonomous Systems, Pittsburgh, PA, February 1993.

[2] K.I. Kim, S.Y. Oh, S.W. Kim, H. Jeong, C.N. Lee, B.S. Kim, C.S. Kim, "An autonomous land vehicle PRV II: Progresses and performance enhancement", in Proceedings of the IEEE Intelligent Vehicles Symposium '95, Detroit, MI, September 1995, pp. 264–269.

[3] Bertozzi M and Broggi A. "GOLD: A parallel real-time stereo vision system for generic obstacle and lane detection", in Institute of Electrical and Electronics Engineers (IEEE) Transactions on Image Processing, 1998, vol. 7, no.1, pp. 62–81.

[4] B.Southall, C.J.Taylor, "Stochastic road shape estimation", University of Pennsylvania, 2001.

[5] S. Nedevschi, R. Schmidt, T. Graf, R. Danescu, D. Frentiu, T. Marita,F. Oniga, and C.Pocol, "3d lane detection system based on stereovision," in Proceedings of the Institute of Electrical and Electronics Engineers (IEEE) Intelligent Transportation Systems Conference, 2004.

[6] AnuarMikdadMuad, AiniHussain, Salina Abdul Samad,Mohd. Marzuki Mustaffa and Burhanuddin Yeop Majlis, "Implementation of inverse perspective mapping algorithm for the development of an automatic lane tracking system", in Institute of Electrical and Electronics Engineers (IEEE) Region 10 conference, 2004, vol.1, pp. 207-210.

[7] Amanda Whitbrook, "Controlling the Pioneer P3-DX robots", at Computer Science and IT School (CSiT), University of Nottingham, 2006.

[8] King Hann, Li Minn, KahPhooi and Siew Wen, "Lane-Vehicle Detection and Tracking", in Proceedings of the International MultiConference of Engineers and Computer Scientists (IMECS), Hong Kong, 2009, Vol. 2.

[9] Sristi Sundram, P.V.Manivannan, "*Design and development of a stereoscopic vision based 3D Targeting System for the Autonomously Guided Vehicles*", ICEET- International Conference on Electrical Engineering and Technology, New York, USA, 5-6th June, 2013.

[10] Aakash Gupta, P.V.Manivannan and M.Singaperumal, "*Development of Monocular Vision based Targeting System for an Autonomous Defence Vehicle*", Proceedings of the 1st International and 16th National Conference on Machines and Mechanisms (iNaCoMM2013), IIT Roorkee, India, December 18-20, 2013.

[11] Alpha Daye Dialloa, Suresh Gobeea, Vickneswari Durairajaha, "Autonomous Tour Guide Robot using embedded system control", IEEE International Symposium on Robotics and Intelligent Sensors (IRIS 2015).