Vision-based Driverless Car in the Condition of limited Computing Resource: Perspectives from a Student Competition

Khanh Duy Tung Nguyen[1], Long Duy Nguyen[1], Son Hai Le[1], Thu Van Le[2], Van-Giang Nguyen[1]
[1]*Faculty of Information Technology,* [2]*Faculty of Control Engineering,*

Le Quy Don Technical University,
236 Hoang Quoc Viet Str, Bac Tu Liem, Hanoi, Vietnam

***Abstract*—The "Digital race - Driverless car" was a student competition where each team was given a model car with fixed specification in both mechanical and electronics. The mission was to program the car to run automatically in the condition of road having obstacles, missing lane lines and bridges with completely out lane lines in the field of view. This paper aims to address the issues facing in the competition as well as a sustainable solution to make car run autonomously and rapidly in the condition of real time processing and limited computing resource. The vision-based solution consists of modules to infer the center of lane from images in normal flat road as well as road having obstacles and bridge. It also features a fuzzy controller and a PID controller to control the car accordingly. The proposed solution has been proven in practice by reaching the first position of the competition.**

Keywords— driverless car, lane detection, obstacle avoidance

I. INTRODUCTION

Along with deep learning, driverless car is dominating the industry and the academia in recent years. The huge interest in the field is due to the anticipated benefits of automated cars, such as reduction in deaths from traffic accidents as well as avoiding the fatal mistakes due to the drivers and other factors. [1]. Though many companies have joined the race to self-driving cars (Tesla, Google, Toyota, GM,...), it is hard for the universities and students to have full-scale autonomous car and access to the full framework to sense the surrounding enviroment and drive the car autonomously.

The "Digital Race - Driverless Car" was a nation-wide competition organized by FPT Corporation from December 2016 to May 2017 [2] for undergraduate students. The regulations are similar to other NXP-Cup challenges where each team was given a model car and the mission was to program so that the camera can detect the lane and steer the race car accordingly. What make Digital Race competition differs from NXP-Cup challenges is that the track has missing lane lines, obstacles and high bridge.

The autonomous controlling framework for driverless car usually consists of two major parts: a lane localization module and a motor controller. While commercial cars can be armed with LIDAR, GPS or high resolution digital maps; it is not the case for the model car due to its limited size and low price.

Fortunately, the relatively low-cost vision approaches have potential to localize the lane (though it come with complex computation).

There exist several vision-based methods to (See [3, 4] for a literature review) lane detection. Despite its high accuracy in severe conditions, those methods are hard to apply to model car with very limited computing resource in real time.

The objective of this paper is to propose a reliable vision-based solution to detect and infer the lane for autonomous model car, followed by a method to control the model car precisely. To accurately determine the center of lane, we analyze the lane lines (in left and right region of the image) from which inference is made to handle all possible cases. We also generalize it to adjust the center of lane in the condition of missing lane lines, the appearance of obstacle in the track, and the extreme bridge. To control the model car smoothly, we use a fuzzy controller for rotary actuator and a PID controller for movement motor.

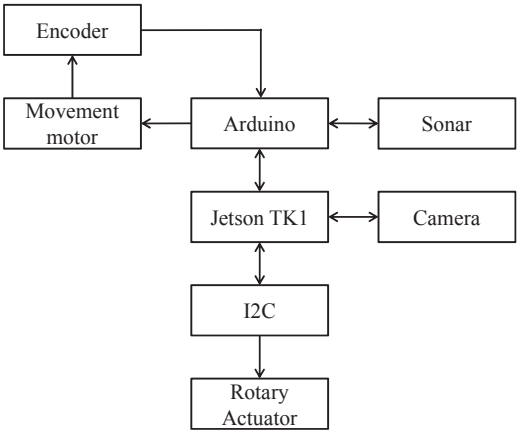
The remaining of the paper is organized as follows. Section II provides the information about the model car and the tracks. Section III presents method to efficiently estimate the center of lane in different conditions. This section also briefly describes the controlling model to control the car. Section IV presents our experimental studies. Section V concludes our work.

II. SYSTEM SPECIFICATION

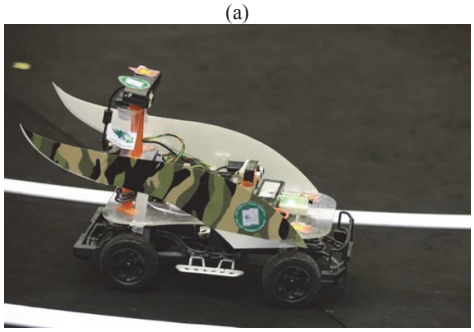
A. System configuration

The hardware for the competition was provided by FPT Corporation, which also dictates the rules teams must adhere to.

978-1-5386-0743-5/17/$31.00 ©2017 IEEE



(a)



(b)

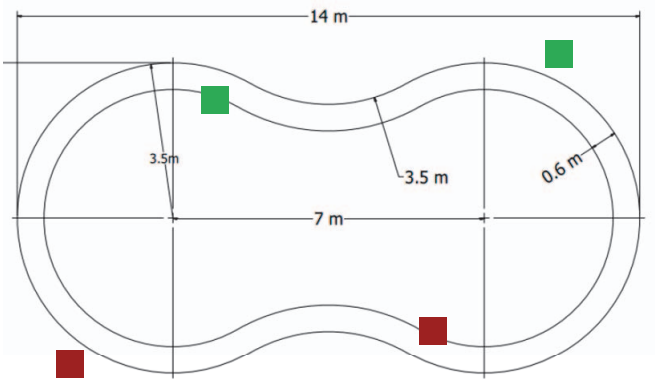
Fig. 1. (a) The functional diagram of the components involved in the model car; (b) An actual model car.

The basic kit includes: (i) A battery-powered Crawler HG-P402 model car having Lipo battery (Crawler brushed motor 20T, serve drive with maximum speed of 25km/h, going uphill up to 45 degree); (ii) An Orbbec Astra 3D camera having coverage angle of 60 degrees in horizontal direction and 49.5 degrees in vertical direction and the range of view is from 0.6 to 8 meters (Its frame rates is 10 fps for 1280×960 images and 30 fps for 640×480 images); (iii) A NVIDIA Jetson TK1 board [5] having Cortex-A15 CPU 2.3 GHz and Kepler GK20A GPU; (iv) A I2C module to control servo (rotary actuator) of car; (v) a sonar detector to sense the obstacle in the range from 2cm to 4m; (vi) An encoder to measure the speed of motor;(vii) an arduino uno microcontroller to control the motor. The functional diagram of the components involved in the kit is summarized in the Fig. 1. The Jetson TK1 board comes with pre-installed Linux4Tegra OS (a basic Ubuntu 14.04 with pre-configured drivers) in which additional programming frameworks such as gcc, OpenCV have been installed by our team.

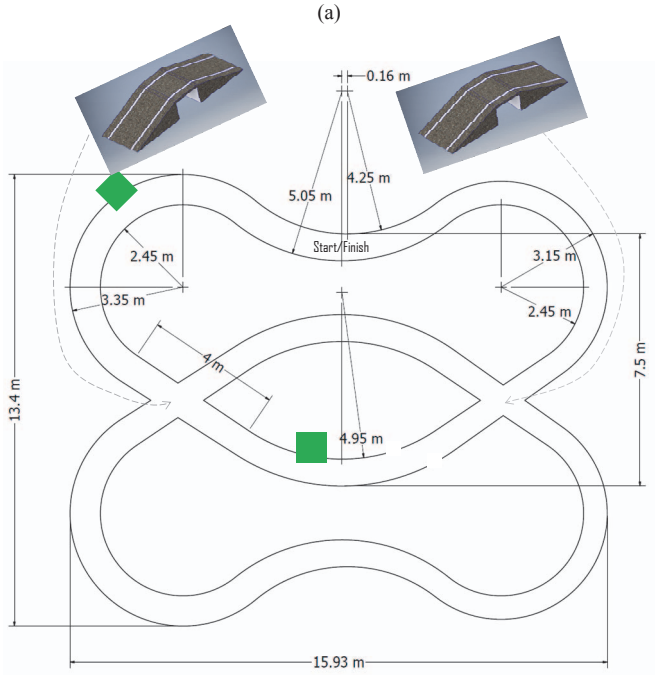
B. *Tracks*

The challenge consists of two rounds with different tracks. In round one, the track (See Fig. 2.a) consists of curved road with two obstacles appear in the road. The degree of curve is strong enough to make part of the lane lines (right or left) disappear in the image acquired from camera.

In round two, the organizer requires the car going thru a harder track (See Fig. 2.b) where it has a bridge in its sight and go underneath of another bridge (in addition to the other obstacles as in the round one). In addition to that, some parts of the track contain missing lane lines so that the model car



(a)



(b)

Fig. 2. Structure of tracks in: (a) Round 1; (b) Round 2. Green and Red squares denote obstacles. In round 2, two bridges have been added to the track. In one lap, the car goes underneath a bridge and goes uphill another one.

cannot infer the lane directly from the image. The degree of difficulty of the lane detection and the other becomes more significant in this case since there are several situations where the model car has no vision ahead. (To be addressed in the Section III). Furthermore, the constraint between high-speed to follow the track uphill and the low-speed to go downhill safely is hard to satisfy.

III. Methods

Given the hardware requirements from the organizer, the actual tracks and the practical issues with the computational platform, we propose a sustainable solution to efficiently and automatically control the car. The solution consists of four parts: estimate center of lane, infer the obstacles, overcoming the bridge and control the car; Each of which will be addressed in details in the following sub-sections. Basically, to control the car accurately, we need to infer the information about the surrounding environment by using sensors (in our case they are camera and sonar detector). In particular, from the images

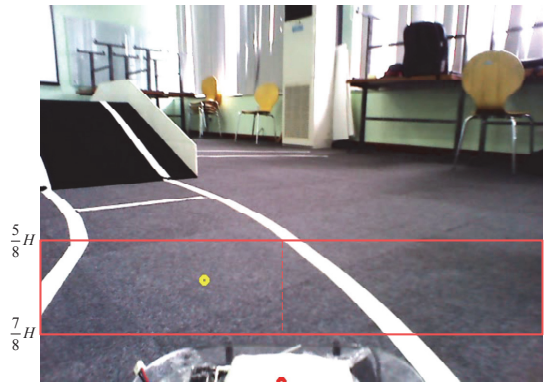


Fig. 3. Red box denotes the area in which the center of lane is determined. The dashed line separates the region of interest (ROI) into two parts: left region of interest (L-ROI) and right region of interest (R-ROI).

acquired by sensors, we need to measure the center of lane from which the car will be controlled accordingly.

A. Center of lane estimation

In our car, we get the frames with the size of 640×480 from camera. To determine the center of lane (from which the car will be controlled accordingly), we propose to use 1/4 of the image with the region starting from $\left[\left(0, \frac{5H}{8}\right), \left(W, \frac{7H}{8}\right)\right]$ (as shown in Fig. 3), where W and H is the width and height of the frame, respectively. The reason of using that region is that the lower part is covered by part of the car while the upper part usually does not contain reliable lane and/or lane lines.

From that region of interest, we separate it into two sub region of interests called left region of interest (L-ROI) and right region of interest (R-ROI).

The left lane (in fact, it is the largest contour surrounding the left lane line) is derived from the L-ROI where it goes thru the following process: (i) threshold the L-ROI to get the binary image (using an adaptive thresholding method); (ii) locate the contours in the binarized image; (iii) filter out the contours having the length less than a threshold value (those contours are considered as noise); (iv) get the contour having longest contour and has the height higher than 1/3 of the height of the L-ROI. (See Table 1 for description of the algorithm).

Similarly, the right lane line (the largest contour surrounding the right lane line) is also indicated by the procedure described in Table 1.

The left lane line extracted above is then used to find the lane line direction. It is a Left-to-Right lane line if the lane line directs to the right direction and It is a Left-to-Left if the lane line directs to the left direction.

Similarly, for the right lane, we also find its directly, it is either Right-to-Left or Right-to-Right. (See Fig. 4). (This classification can be performed easily by comparing the right-most top and right-most bottom pixel in the lane line's contour - See the designated label in Fig. 4(a)). Note that, there are cases where no lane line appears in the L-ROI or R-ROI.

Table 1. Determine the largest contour in a region of interest

Input: region of interest (left of right) **r**.
Output: largest contour in **r**.
Algorithm
Step 1: Threshold **r** to get binary image **b**.
Step 2: Find contours (denoted by **c'**) in image **b**.
Step 3: Filter out contours having size < τ and denote the result by **c**
Step 4:
    for each contour c_i in **c**
        if height(c_i)>height(**r**)/3
            $c_{max} = c_i$;
    end for
Step 5: Return c_{max}.

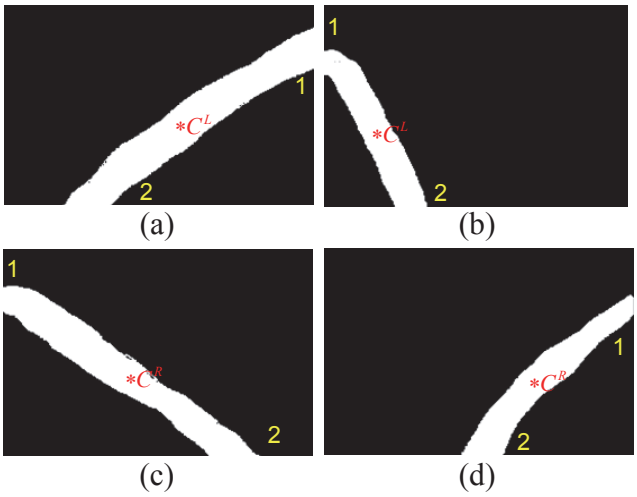


Fig. 4. Classify the lane line into different types. For left region of interest: (a) Left-to-Right lane line; (b) Left-to-Left lane line. For right region of interest: (c) Right-to-Left lane line; (d) Right-to-Right lane line. The label 1 and 2 denotes the position in the lane line from which its relative x-values comparison is used to determine the direction. C^L and C^R denotes the center of the left lane line and the right lane line, respectively.

As an intermediate step, we also calculate the center of left lane line and center of right lane line, denoted as C^L and C^R, respectively (as shown in Fig. 4) by measuring its mean values.

Given the direction of lane lines in the L-ROI and R-ROI, and the center C^L of left lane line in L-ROI and the center C^R of the right lane line in R-ROI, the calculation for actual center of lane is given in Table 2. In Table 2, (x_L^*, y_L^*) denotes the coordinate of right most pixel in the left lane line (in L-ROI), and (x_R^*, y_R^*) denotes the coordinate of left most pixel in the right lane line (in R-ROI). W denotes the width of road in previous frame. (Due to limited space, we omit the illustration for deriving the algorithm in Table 2)

B. Obstacle Avoidance

In both rounds of the competition, the organizer sets up hard obstacles, which were made from concrete-like materials, to increase the degree of difficulty as well as to model the real world scenario. Once hit the obstacle, the car stops moving (it also lightly damages the sensors and the chassis of model car). The obstacle is covered with dark-green and dark-red color, which is somewhat similar to the background of the track.

Table 2. Estimate center of lane

Input: Center of lane line C^L in L-ROI and C^R in R-ROI, direction of lane line (dir(***l***)) in L-ROI and dir(***r***) in R-ROI.

Output: Center of lane $\hat{C}$.

Algorithm:

if $\exists$ lane line ***l*** in L-ROI and $\exists$ lane line ***r*** in R-ROI
  if dir(***l***) = Left-to-Right **and** dir(***r***) = Right-to-Right
    and $|\, y_L^* - y_R^* \,| < \tau^*$

$$\hat{C}_x = C_x^L + \frac{W}{2}; \hat{C}_y = C_y^L$$

  end if
  else if dir(***l***) = Left-to-Left **and** dir(***r***) = Right-to-Left
    and $|\, y_L^* - y_R^* \,| < \tau^*$

$$\hat{C}_x = C_x^R - \frac{W}{2}; \hat{C}_y = C_y^R$$

  end if
  else

$$\hat{C}_x = \frac{1}{2}\left(C_x^L + C_x^R\right); \hat{C}_y = \frac{1}{2}\left(C_y^L + C_y^R\right);$$

$$W = C_x^R - C_x^L$$

  end if
else if $\exists$ lane line ***l*** in L-ROI **and** $\nexists$ lane line ***r*** in R-ROI
  if dir(***l***) = Left-to-Left

$$\hat{C}_x = C_x^L - \frac{W}{2}; \hat{C}_y = C_y^L$$

  else

$$\hat{C}_x = C_x^L + \frac{W}{2}; \hat{C}_y = C_y^L$$

  end if
else if $\nexists$ lane line ***l*** in L-ROI **and** $\exists$ lane line ***r*** in R-ROI
  if dir(***r***) = Right-to-Right

$$\hat{C}_x = C_x^R + \frac{W}{2}; \hat{C}_y = C_y^R$$

  else

$$\hat{C}_x = C_x^R - \frac{W}{2}; \hat{C}_y = C_y^R$$

  end if
else */*There are no lane lines in both L-ROI and R-ROI*/*
  */*Use the center of lane estimated from previous frame*/*
end if

To navigate the car to avoid the obstacles, there are two steps involved: (1) obstacles detection and extraction; (2) re-calculate the center of lane to avoid the collision with the obstacle.

In Step (1), the region of interest (in which we extract the information about the obstacle) is indicated (which is similar to the ROI in Fig. 3). In the ROI, we calculate its contours from the edge map which was found by using Canny edge detector. The contour meets the following three conditions will be the obstacle: (i) it is a hole contour; (ii) its area is larger than a designated value (a threshold); (iii) the ratio of contour height to contour width is in between a designated range (since the obstacle has cube shape).

In addition to extracting the obstacle from the frame, we also find whether it belongs to left or the right lane by comparing the center of obstacle with the center of left lane line and center of right lane line. The obstacle will belong to the closer one.

In Step (2), the center of lane is updated according to the following rule. Denote the bottom left and bottom right of the obstacle by O^L and O^R, respectively. Then the center of lane is updated as in Table 3 and illustrated in Fig. 5.

One can consider using an adaptive thresholding method and treat the obstacle as the lane line. In that case, the computation time can be reduced. One can also consider using a segmentation-based method to segment image and derive the obstacles from the segmented image. In that case, highly accurate segmentation method such as mean-shift [6] can be used. However, the limited computing power of the embedded board Jetson TK1 prohibits the use of such method. In our try, with the current configuration in the board, it can process 3 frames per second and is not suited for real time processing.

One can also consider using sonar for detecting obstacle. However, due to the low rate of sonar as well as its noisy signal, it is not applicable to real-time processing.

C. Going Across the Bridge

In the second round of competition, the organizer set up a bridge having the structure as shown in Fig. 2(b) and Figs. 6(a), (b). To ease the requirement, a horizontal line crossing the road is painted on the track so that the car can recognize the starting point of bridge. To computationally recognize it, we use Hough transform to determine the line crossing the road.

After determining the starting point of the bridge (via the horizontal line), then comes the control of the car. The control process is divided into three phases: (1) go uphill; (2) go in flat road; (3) go downhill.

The difficult constraint here is that the speed of the car needs to be high enough to go uphill, then it needs to be reasonably low enough to stably go in the flat road and go downhill safely. And finally, it needs to be high again in the flat road. Otherwise, it will end up with missing direction in the flat road (phase 2), and eventually rollover if going downhill with high speed.

In our solution, once having the signature of the starting of the bridge, the motor will be given high pulses to go uphill (phase 1). Since the length of the uphill road is given beforehand, we can indicate (by counting the pulses in the movement motor) the moment the car reaches the peak of hill precisely. Around the peak, the camera cannot see the lane lines (as illustrated in Fig. 6 (c)). To handle that difficulty, we use a lane stabilizing program to accept the new lane center only if the new lane center is not far from the previous lane centers.

When the car reaches the peak of hill, we move to phase 2. In phase 2, the speed of car is decreased. When passing the flat road in the top of bridge (Figs. 6(a) and (b)), then comes phase 3 and the speed will be decreased further. Otherwise, the car will be ended up with a rollover. The moment of decreasing the

Table 3. Updating the center of lane to avoid obstacle

If obstacle occupies the left lane line

$$\hat{C}_x = \frac{1}{2}\left(C_x^R + O_x^R\right); \hat{C}_y = \frac{1}{2}\left(C_y^R + O_y^R\right)$$

else /*obstacle occupies the right lane line*/

$$\hat{C}_x = \frac{1}{2}\left(C_x^L + O_x^L\right); \hat{C}_y = \frac{1}{2}\left(C_y^L + O_y^L\right)$$

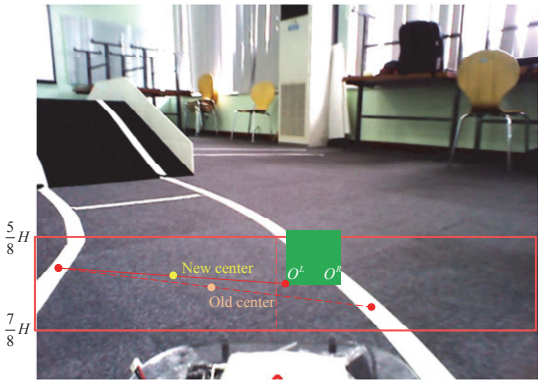


Fig. 5. Illustration of updating the center of lane when there is obstacle (green box) on the road.

speed is given when it finishes phase 2. To accurately determine the staring of the going downhill, we also measure the actual length of travelled distance in phase 2.

Once finishing the going downhill part, the speed of the car is increased again so that it can achieve the best timing performance. The ending of the bridge is also indicated by using the same principle as the one for determining the peak of the hill in phase 1.

It is important to note that, for a bridge with abrupt intersecting point (as shown in Figs. 6(a) and (b)), it is not the case where the faster the better since a high speed will end up with a rollover of the car. Meanwhile, a low speed might not enough for the car to go uphill.

D. Control the car to follow the center of lane

To control the car to follow the road smoothly in high speed, we need to address the following two issues: (i) How to control the rotary actuator (angular motor); and (ii) How to control the movement motor. It should be optimized so that the car reaches high speed in flat and straight road, decreases its speed in curved road (or when obstacle shows up in the road), varies its speed smoothly when going uphill or downhill of the bridge.

Control the rotary actuator (angular motor)

To efficiently and smoothly control the angular motor of the car, we use a fuzzy controller as shown in Fig. 7. The fuzzy controller takes the real angle β measured from the center of lane as the input and generates the θ angle (that will be use to control the angular motor) as the output.

Constructing the fuzzy controller [7] consists of the following steps: (1) Define the language; (2) Build the set of fuzzy rules; (3) Fuzzying the plain input into fuzzied values by using membership functions; (4) Evaluate the rules in rule set; (5) Combining the results in each rule; (6) De-fuzzing. Due to the limited page space, we do not show the details of each step here.

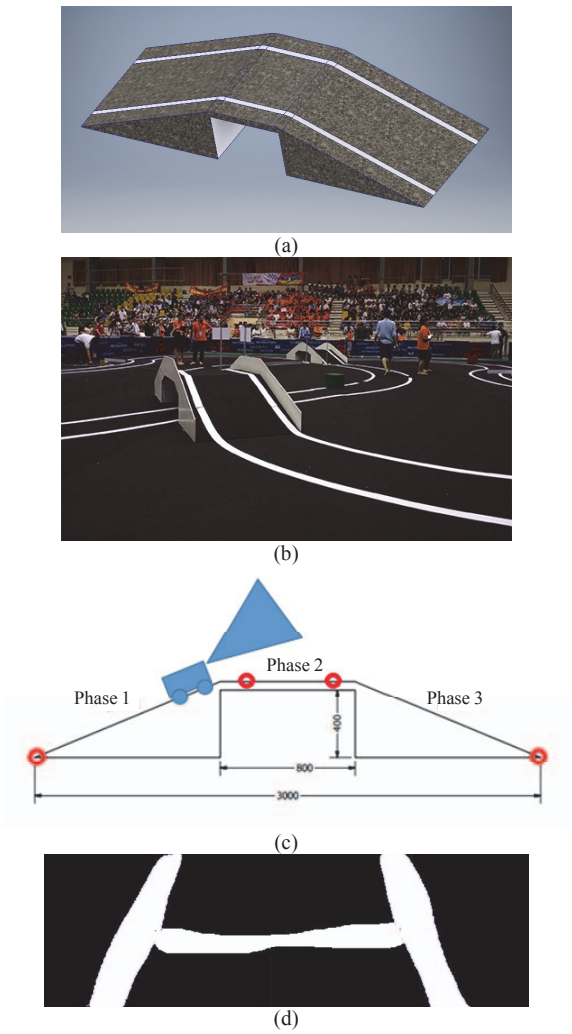


Fig. 6. Structure of the bridge using in the round 2 of competition: (a) visualized bridge, (b) actual road with a bridge; (c) controlling the car is divided into three phases; (d) determining the starting point of the bridge by detecting a horizontal lane line.

Control the movement motor

To ensure the car go with high speed in both straight and curved lane, we build the controller that infer the speed from the angular of the car and the predefined speed (set speed). A PID controller is used for that purpose. [8]

IV. Experimental Results

The proposed lane detection algorithm was implemented in OpenCV and deployed to the Jetson TK1. We performed several tests in our own track, in a test track provided by the organizer, and the real tracks in the competition. In our tests, the car was able to accurately determine the center of lane and smoothly navigate along tracks. It avoided the obstacle placed dynamicly in the track (the obstacle was placed to occupy 40% of the lane) (See Fig. 8(a)). In the test case for going uphill thru the bridge, the car successfully went thru it (See Figs. 8(e)-(h)).

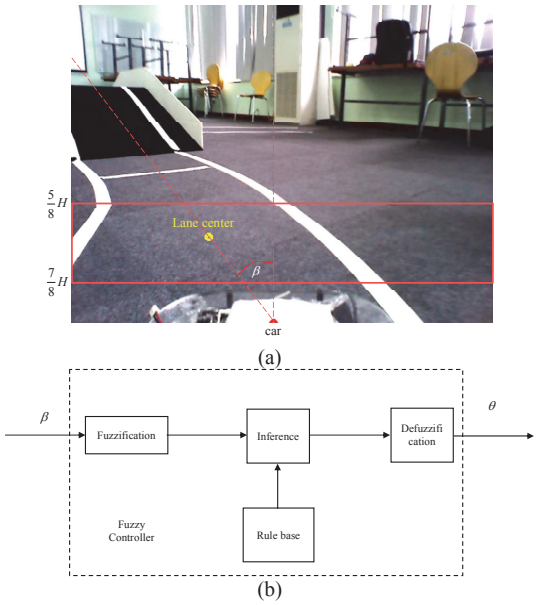


Fig. 7. (a) Real angle β is measured from the visual data; (b)Construction of fuzzy controller to get θ angle.

In Round 1 of the competition where there are two obstacles on the road (See Fig. 2(a)), our car has managed to complete 34.7m track in 13 seconds with the average speed of 2.67m per second. In Round 2, it completed two laps (See Fig. 2(b)) in 32 seconds. The running time in Round 2 is longer than Round 1 due to the fact that, in addition to two obstacles, the track in Round 2 has additional missing lane lines and a bridge to go thru.

The representative images from Round 2 of the competition are showed in Fig. 9 where our car accurately avoids the obstacle and stably goes thru the bridge.

V. Discussions and Conclusions

When participating the competition, we have faced several tough decisions. It is whether to use machine learning methods, such as linear regression method or neural network, to infer the lane center. We turned down that possibility since we could not collect enough data to train the model to work properly. The organizer gave access to the track few hours (for both rounds) prior to the racing time. The manual control of the car does not work well as the automatic counterpart, therefore we could not collect good data set for training. The inference time using machine learning is higher than the analytic method.

In addition to that, the use of concrete-like obstacle (in both rounds) prohibits us in trying with machine learning technique. Errors in estimating the center of lane resulted in damages in the model car and its sensors. (Due to such damages, we had to replace the chassis of the model car right before the racing time).

The other issue in the competition is the stability of the camera and its degradation during the testing period. When reaching the final round, the camera gets blur and contain severe noise. Finally, while the tests were performed mainly on daylight, the actual competition was organized in indoor and at

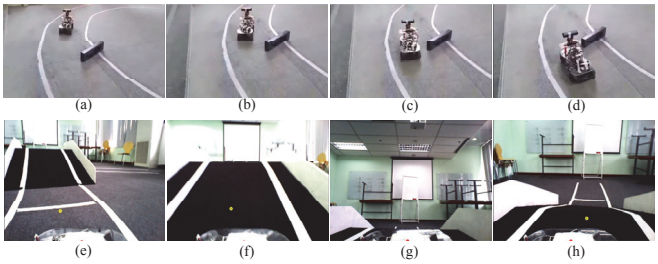


Fig. 8. Samples from video recorded in outdoor, daylight condition where the car avoids obstacle smoothly (a)-(d); Indoor condition where the car goes thru the bridge (e)-(h). Yellow dot in (e)-(h) denotes the estimated center of lane. Images in (a)-(d) were captured from https://www.dropbox.com/s/ht41v39w2ibf4yl which was recorded three weeks prior to the competition. Images in (e)-(f) were captured from https://www.dropbox.com/s/shi3717nxvuuz19

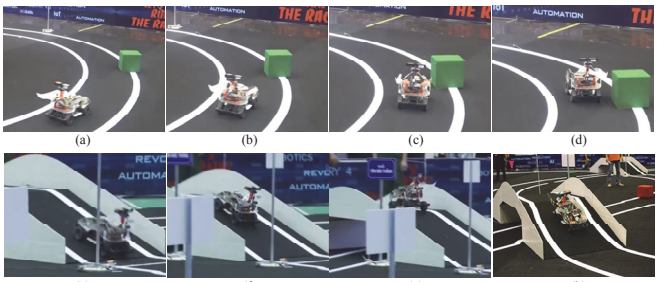


Fig. 9. Samples from video recorded in the competition. First row contains images when the car avoids obstacle and Second row contains images when the car goes thru the bridge. Images in (a)-(g) were captured from https://www.dropbox.com/s/kj4i8q90isx8mau

night. The actual lighting (with several color light sources) is very different from the training and makes the obstacles look similar to the background of the track and results in troubled lane center.

In conclusion, we have proposed a rapid lane detection algorithm for real time processing with limited computing resource. The method works based on analyzing the real world situations and performs the calculation accordingly. Through extensive tests, it has been proven to accurately and robustly estimate the lane and the center of lane in severe conditions of missing lane lines, obstacles and bridge. The combination of lane detection with our controllers helps us to reach the first place in the "Digital Race - Driverless Car" competition.

References

[1] T. Jiang, S. Petrovic, U. Ayyer, A. Tolani, and S. Husain, "Self-driving cars: Disruptive or incremental?" *Applied Innovation Review*, no. 1, pp. 3–22, 2015.

[2] FPT Corporation, *Digital race*, https://cuocduaso.fpt.com.vn, 2017.

[3] A.B. Hillel, R. Lerner, D. Levi and G. Raz, "Recent progress in road and lane detection: a survey," *Machine Vision and Applications*, 2015.

[4] S. Sivaraman and M.M. Trivedi, "Looking at vehicles on the road: a survey of vision-based vehicle detection, tracking, and behavior analysis", *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 4, pp. 1773-1795, 2012.

[5] NVIDIA, Jetson TK1 embedded development kit, http://www.nvidia.com/object/jetson-tk1-embedded-dev-kit.html.

[6] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 24, no. 5, pp. 603–619, 2002.

[7] T. Hagglund, *PID Controllers: Theory, Design and Tuning,* ISA: The Instrumentation, Systems, and Automation Society, 1995.

[8] K.M. Passino and S. Yurkovich, *Fuzzy Control*, Addison Wesley, 1997.