Control of autonomous cars for intelligent transportation system

Mohammad Abdul Qayum[1], Nafiul Alam Siddique[2], Mohammad Abtiqul Haque[3], Abu Saleh Md.Tayeen[4]

[1,2,3,4] Klipsch School of Electrical & Computer Engineering, New Mexico State University.
qayum@nmsu.edu, nafiul@nmsu.edu, atiqul@nmsu.edu, tayeen@nmsu.edu

ABSTRACT
This work concentrates on establishing a technology to develop a unique approach for the integration of intelligent system control into transportation systems. We have implemented an algorithm [1] that controls a car to track a predefined track which integrates a human driver with computer control to increase human performance while reducing reliance on detailed driver attention. Knowledge obtained from the optical tracking system about vehicle position and orientation provides the automatic decision making intelligence needed to follow a virtual vehicle moving on track. The result shows good performance of this model independent algorithm with fairly cheap RC cars.

KEYWORDS: Virtual Vehicle, Autonomous Car, Intelligent Transportation System (ITS)

1. INTRODUCTION

Due to the advancement in technologies, decade old transportation systems and agents (cars, trains etc.) are going towards automation and up-gradation. Intelligent Transportation System (ITS) is a state of art technology that uses motion sensors, computer processing power, communication protocols and human management strategies in collaboration. With implementation of ITS, travellers will get more information to be time-efficient, driver's safety and efficiency will increase, and congestion of ever increasing traffic can be controlled. Our work in this paper aims at developing an autonomous system which can detect collisions and report it to the operator. For this, we have developed an autonomous RC car which receives command from server PC through Zigbee communication protocol. According to received commands the hardware on car drives the car. We have implemented an algorithm which uses virtual vehicle approach to track a predefined path [1]. OptiTrack[2]/Vicon system[3] is used to get the current position and orientation of the car. To test our concept we implemented the algorithm to track a straight line and circle. We achieved significant good results for line following. Due to some constraints, we could not achieve good results for circle, but we were able to validate the concept.

This Paper is organized is as follows- some related works are mentioned in the field of Intelligent Transportation System. Tracking algorithm part tells about the algorithm used and how it tracks a predefined path. Hardware platform explains the details of hardware and other systems used for this project. After that software implementation part explains about programming challenges and finally, the paper is concluded with results and future work.

2. RELATED WORK

Smart Vehicular Transportation Systems by Sandia National Laboratory[4] was built upon established Sandia intelligent systems technology to develop a unique approach for the integration of intelligent system control into the US Highways and urban transportation systems. The Sandia developed concept of the COPILOT controller that integrates a human driver with computer control to increase human performance while reducing reliance on detailed driver attention.

A consortium of 15 European research institutes and private industrial companies have formed the Cyber Cars/Cyber Move program.[5] The programs objective is to create a new option of Intelligent Transport Systems based on road vehicles with fully automated capabilities, or CTS (Cybernetic Transportation System). First Car Platform equipped with robotic technology is RoboCar™ by ZMP Inc[6]. It includes a built-in stereo camera and a cutting edge image recognition module allowing high processing speeds never obtained before. With its various sensors, RoboCar™ allows R&D on vehicles with autonomous motion, Advanced Safety Vehicles, Intelligent Transportation Systems and environment-friendly technologies.

An Example of Intelligent Transportation Systems was built by Nicholas J. Ward and et al. [7] Their work presents a discussion of the possible deleterious effects of automated technology in the transport context in relation to Intelligent. Transport Systems (ITS). The potential deleterious effects of ITS include (a) shifting the driver out-of-the-loop, such that situation awareness and responsiveness to critical events may be impaired, and (b) behavioural adaptation that undermines system effectiveness. The Cots Bots[8] are inexpensive and modular mobile robots built entirely from commercial off-the-shelf components. These robots provide a convenient platform on which to investigate algorithms, cooperation, and distributed sensing in large (> 50) robot networks. Each is equipped with on-board processing, radio communication, and a base platform for mobility. Software is written using TinyOS, an open-source, event-driven operating system for large-scale distributed sensor networks.

3. TRACKING ALGORITHM

We have implemented a model of independent control algorithm for RC car. This algorithm is based on Virtual Vehicle Approach [1]. In this algorithm a desired path is defined by a mathematical equation. A virtual vehicle is assumed on this reference trajectory which is known as reference point and motion of this point is governed by differential equation to give error feedback. This error feedback is used to control RC car which follows this virtual vehicle.

Two main strategies are defined for implementing this algorithm. This strategy only requires control in lateral direction and its longitudinal velocity is kept at a constant value. The advantage of this algorithm is that it is very robust with respect to measurement errors and other disturbances. Therefore it is suitable for RC car model which doesn't have precise control on its steering mechanism. The algorithm can be explained with following figure.

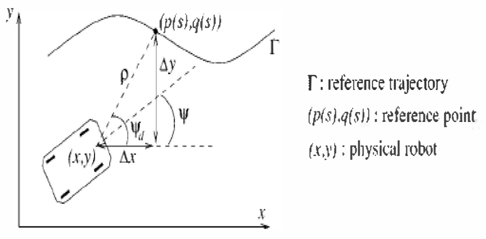

Figure 1: Virtual Vehicle Approach

The desired path is defined using mathematical equation and is shown in figure 1 as Γ. *p(s)* and *q(s)* gives the reference point on the path. In our case we have implemented this for a line and a circle with following parametric equations obtained with OptiTrack sytem:

Line:

$$p(s) = 1391 - 2994s$$
$$q(s) = 1025 - 2116s \quad \textit{where } (0<s<s_f)$$

Circle:

$$p(s) = 1500\cos(s)$$
$$q(s) = 1500\sin(s) \quad \textit{where } (0<s<s_f)$$

The path is taken as $x_d = p(s)$, $y_d = q(s)$; The path starts with $s = 0$, and $s = s_f$ is final point of path. After defining the path with mathematical equation, next step is to perform some geometric calculations. From OptiTrack system, we get the position of RC car as *(x,y)*. As shown in above figure, now we can calculate distance between virtual vehicle and RC car as 'p' and destination angle as ψ_d. The orientation of RC car ψ can get from OptiTrack system. There are main two control objectives of this algorithm and are as follows.

I. $\lim_{t\to\infty} p(t) \leq d_p$

II. $\lim_{t\to\infty} |\psi - \psi_d| \leq d\psi$

The first control objective is to keep the distance between RC car and virtual vehicle less than or equal to d_p, where d_p is the constant look ahead distance between RC car and virtual vehicle, so that they will not overlap each other. And the second control objective keeps the difference between destination angle and car's orientation as minimum as possible.

The equation to move the virtual vehicle on reference trajectory is as defined as:

$$\dot{s} = \frac{1}{\sqrt{p'^2 + q'^2}\cos(\psi_d - \theta_r)}(2v\cos(\psi_d - \psi_v) - \gamma\rho)$$

Where, $\gamma = 2v\cos(\psi_d - \psi)\ d_p = 1/\alpha$

$\theta_r =$ orientation angle of the tangent to the reference curve at 's' and the value of s is updates as,

$$s = s + \dot{s} * \Delta t$$

Following two assumptions are made about reference curve.

1. The bound on the curvatures of the reference curve is sufficiently small. Hence the difference between destination angle and Car's heading should always remain less that 90 degree angle.
2. $2v\cos(\Psi_d\text{-}\Psi) - \Upsilon p \geq \mathcal{E} \geq 0$. This is very important assumption, otherwise value of s_dot will go negative and virtual vehicle will move back instead of going forward. The new concept in this algorithm is that, it uses a virtual vehicle on desired path and our robot will follow it. The velocity of RC car is constant and the speed of virtual vehicle depends on position of RC car. If RC car is far away from the virtual vehicle then it slows down until it approaches near.

Finally, Algorithm can be summarized as follows:

$$\dot{s} = \frac{1}{\sqrt{p'^2 + q'^2}\cos(\psi_d - \theta_r)}(2v\cos(\psi_d - \psi_v) - \gamma\rho)$$
$$\delta_f = -k(\psi - \psi_d)$$
$$v = const$$

Where, 'δ_f' and 'v' controls the lateral and longitudinal velocities respectively.

4. EXPERIMENTAL PLATFORM

The figure 2 shows overall block diagram of the system. The system basically contains two main parts, server and client. Client is nothing but a RC car having all hardware mounted on it. It receives commands from a server PC to follow a certain trajectory. Server and client are connected to each other to form a closed loop by OptiTrack system. According to the information gained from OptiTrack system, server PC runs control algorithm to transmit specific commands to car. Zigbee communication protocol[9] is used for communication between client and server.

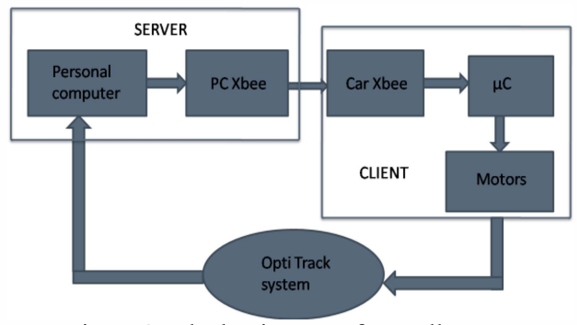

Figure 2: Block Diagram of overall system

4.1 Client Platform

The car (client platform) contains a RF transceiver (Xbee module), which uses Zigbee communication protocol shown in figure 3. A point to point connection is established between two Xbee modules. A server Xbee broadcasts the data packet and client Xbee receives it and passes to the microcontroller for further action. Microcontroller on car gets the packet from Xbee and gives control signals accordingly to motor driver circuit. Basically it receives five main commands with data for each command. Commands are Forward, Backward, Left, Right and Stop. The packet contains speed value for each action except for the Stop command. We have used ATmega 162 microcontroller[10] which is low power AVR 8bit microcontroller having RISC architecture. A basic H-bridge circuit is designed with XN1504 Transistor and S8858 MOSFET.

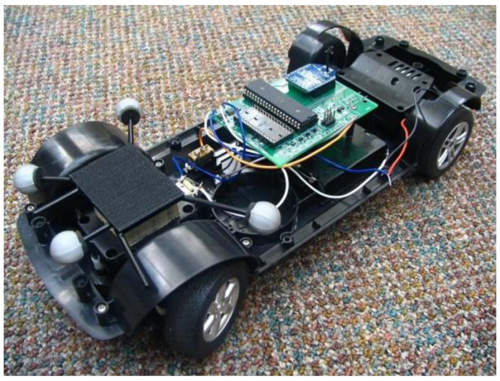
Figure 3: Car Platform with uController and Xbee

4.2 Server Platform

OptiTrack System is a tracking system which gives the location and orientation information of a rigid body. This system consists of total 12 cameras placed around area of working. Optical markers are used to build a rigid body. A rigid body can be built using minimum 3 markers. OptiTrack system camera captures the light reflected by markers and they can locate the markers in work area. All the cameras are connected to computer through a USB port and ARENA software is used to stream all the data of markers. This system is manufactured by Natural Point Inc. In our project we have used 4 markers to create a rigid body and all 4 markers are placed on RC car in an asymmetric way. Vicon System is similar type of system as OptiTrack system. The basic working principle of both systems is same. Vicon system is more accurate and advanced system as compared to OptiTrack system. It can cover more area and also the range of Vicon system cameras is larger. All the cameras are connected to a server and then are also connected to a computer (Server). Vicon Tracker software is used to stream data of rigid bodies.

5. SOFTWARE PLATFORM

To access localization data we used Vicon's sample codes. We modified the access codes according to our needs, as we only need *x, y* and *heading* information of the car. We connect to Vicon server with its IP address. Architecture of the program is as follows: A forward control signal is sent to the micro-controller of the RC car; when the car starts moving, car's localization data (*x, y* and *heading*) are obtained from Vicon/Optitrack system, a Tracking function calculates the steering control by sending RIGHT or LEFT command and duty cycles for turning according to the required rotation for the car, virtual point moves which gives new data to calculate the next step; finally, when the car reaches close to its destination, a STOP command is sent to stop the car. A file is created and required information is stored to analyse the data later. Figure 4 shows the flowchart of the architecture of the program.

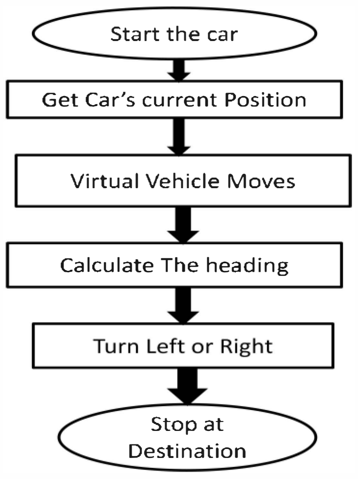

Figure 4: Flow chart of the Tracking Algorithm

As RC car's steering control is not up to the mark, we have to go through lots of trials and errors with FORWARD and LEFT/RIGHT duty cycles. Also, as battery discharges very quickly, we have to always adjust the duty cycles. We also considered the surface as there is no feedback system for the velocity control. As experiment area is small for complicated tracks, we only tested with straight line and circle. For straight lines, we got pretty good result considering the poor control of cheap RC car. But for circle we get marginal data. Possible reasons are curvature is higher than the algorithm to follow, poor steering control of the car and slow response of the RC car's steering.

6. RESULT

The stated algorithm was implemented using a toy RC car. The hardware components used for this mobile platform has already been discussed. Here, some of the observations obtained from the experiments have been analyzed. Two different trajectories were used for evaluation purpose. The first one- line was implemented and executed successfully. But the second one- circle had some issues. These are discussed elaborately in the following sections:

6.1 Straight line as a trajectory

In case of the straight line, the parametric equation of the straight line is shown in section 3. As virtual vehicle moves in a straight line, the RC car follows it. Some of the results are shown below.

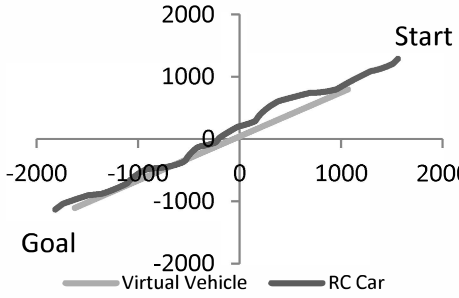

Figure 5: Virtual Car versus Real Car's movement#1 [Both x and y axis in millimetre]

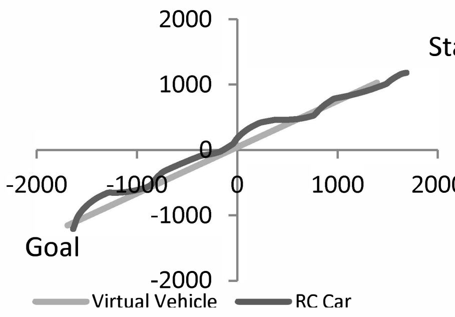

Figure 6: Virtual Car versus Real Car's movement#2

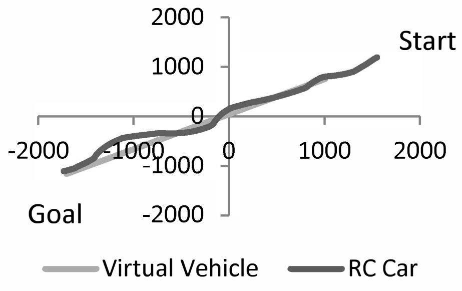

Figure 7: Virtual Car versus Real Car's movement#3

Fig.5 shows the trajectory of the virtual vehicle and the RC car at initial stage. Adjusting the PWM signal for left and right turn and the other constants better results were obtained as shown in Fig.6 and Fig.7. It can be observed that the car is moving more smoothly and closely to the trajectory of the virtual vehicle.

The value of s represents the position of the virtual vehicle. The algorithm start from s=0 and ends at a point $s = s_f$. In this case, the value of s_f is chosen as 1. The increment of the value of s is depicted on the following figure.

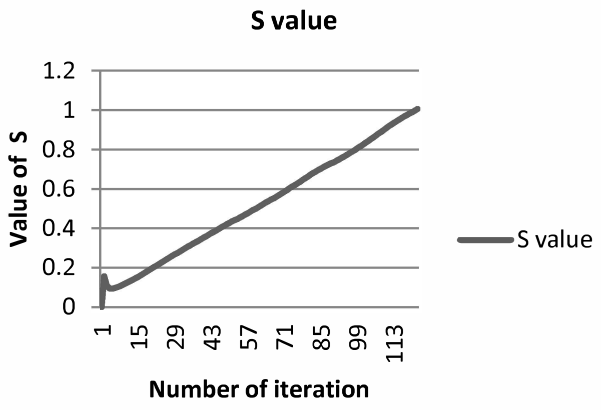

Figure 8: Value of S after each iteration

At first there is a sudden rise in the s value. As the algorithm works in such a way that if the value of s increases suddenly leaving the RC car behind it is going to slow down and coup up with the pace of the RC car. That is why it falls down after a short while and then increases steadily. This is another way to verify that the motion of the virtual vehicle which is governed by a differential equation has been implemented correctly.

6.2 Circle as a trajectory

The RC car was also tested for a circle as a trajectory. The car followed the circle fairly well. The path of the car was not that perfect as the lateral control of the car is not that precise. According to the algorithm the steering was turned left and right depending on the angle between the heading and the desired point on the trajectory of s which is nothing but the δ_f. The turn angle was kept variable, so that it can take turns depending on the angle δ_f. Two different graph plots for the circular trajectories are given in figure 9 and 10. From the graphs it can be concluded that the RC car tried to follow the given trajectory and the deviation is due to the lack of accuracy of the steering control.

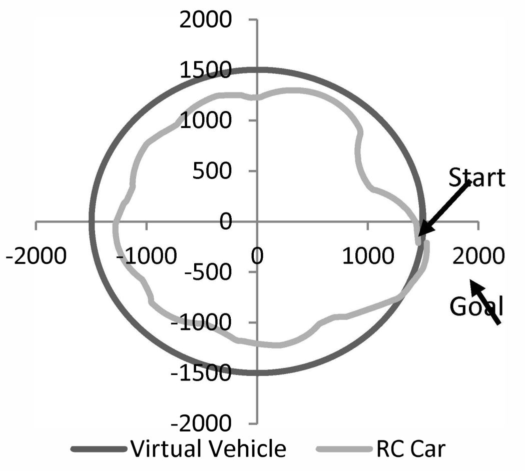

Figure 9: Virtual Car versus Real Car's movement#1

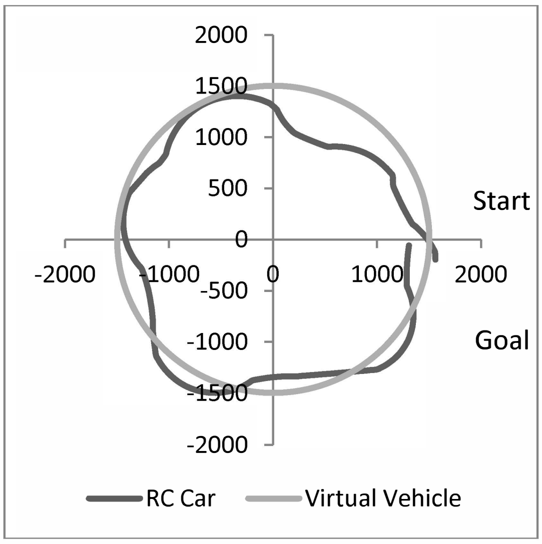

Figure 10: Virtual Car versus Real Car's movement#2

6.3 Result Interpretation

After observing the experiments and analyzing the data, some factors have been determined which affects the motion of the RC car and causes the deviation from a given trajectory. The surface friction has a big effect on the RC car's motion. As it is a fairly cheap toy car, it does not weigh that much. That is why, sometimes if the speed is increased a little bit, the car moves very fast from the starting position and as the steering control is not that precise, and it takes some time to turn, the car might get derailed from the track. The longitudinal speed of the car is kept constant. So the car moves in a constant speed right from the beginning. If the speed is much faster compared to the steering movement response time, then the car moves out of the track. An optimum speed has to be determined to get a smooth tracking. The charge of the battery also affects the ability of tracking. If the battery is fully charged then the speed has to be reduced to keep the car on track as it generates more torque. And as the battery gets discharged, the velocity needs to be increased. The car's steering control does not respond correctly if the instructions are sent in higher speed. There is some time lag due to the calculation and transmission. So the control instructions need to be sent with a little time delay if we want the car to execute every single instruction as mentioned in the previous point there are some delay associated with the calculation and communication. A Xbee transmitter and receiver is used for the communication purpose. Even though the communication speed is quite good enough, some lag is there for calculation and execution of the instruction. After all it can be said that most of the above stated constraints are interrelated. They depend on each other. Considering all these conditions, and optimizing all the possible aspects, best effort was given to obtain an adequate result.

7. FUTURE WORK

In future, the algorithm can be implemented on a car which has a better control system. In that case the movement of the car will be smoother. Also, more complex curves can be implemented. We had developed some complex curves using parametric equations, which will be implemented on the availability of accurate control system for the car. Moreover, velocity can be made variable. Here a constant velocity is used for the motion in the longitudinal direction. Variable velocity can add a totally new dimension to this work. Then the car's velocity can be reduced during taking turns and increased while following a straight line. This will improve the accuracy and make the system more time efficient. Another thing is that car moves in a constant speed right from the starting point. Sometimes it creates problem. Because if the car needs to take a big turn right at the beginning then it needs some time for determining that which way it has to turn and then execute it. By this time if the speed is high, then it is going to miss the desired look-ahead point. So a system which has variable speed (i.e. low speed at starting and increasing gradually) will be more accurate to track the trajectory. Other than this more than one vehicle can be include in the system to follow some given trajectory avoiding collision. Control signal can be broadcasted with individual ID of the vehicles. All the cars will receive the signal but only the one with the broadcasted ID will execute the command. In this way, a system of several mobile platforms can be built which will have better accuracy and efficiency.

8. CONCLUSION

In this work, an algorithm which is based on virtual vehicle approach has been implemented. A radio controlled car is a fairly cheap toy car and so, steering control was not that much precise. The algorithm for tracking the reference trajectory followed here is a robust and model independent algorithm. It provides control over only the lateral direction. As there is no longitudinal control implemented, the algorithm only works locally. The reference trajectory is set by a virtual vehicle, whose motion on a given trajectory is governed by a differential equation which contained error feedback. Thus a closed loop system was formed with the help of the position error. The performance of the algorithm was tested using first the OptiTrack system then in the Vicon system. After implementation, the obtained data was analyzed graphically. On both cases, the car was able to follow the virtual vehicle when it moved in a straight line. Slight deviation of the car was observed from the virtual vehicle's path which can be tolerated as the steering control of the vehicle was not that precise. Using our method to control several cars on a city environment especially at traffic signals, cars can move autonomously in intelligent roads (roads with tracking systems), avoid collisions and bypass possible traffic jams at intersections.

REFERENCES

[1] M. Egerstedt, X. Hu, and A. Stotsky, *"Control of Mobile Platforms Using a Virtual Vehicle Approach"* IEEE Transactions on Automatic Control, Vol. 46, 11, November, 2001

[2[*OptiTrack system*- www.naturalpoint.com/optitrack

[3[*VICON motion system*: http://www.vicon.com/

[4] Charles Q Little, Christopher W Wilson, "*Smart Vehicular Transportation System*", Sandia National Laboratories, United States Department of energy under contract DE-AC04-94AL85000

[5]M. Parent and G. Gallais, "*Intelligent transportation in cities with CTS*" in *Proc. ITS World Congress*, Oct. 2002, pp. 1-5.

[6] Car Robotics Platform RoboCar^{TM} released [Online]. [Accessed: July. 7, 2010]. Available: http://www.zmp.co.jp/data/Press_eng_20090615.pdf

[7] Nicholas J Ward, "*Automation of Task Processes: An Example of Intelligent Transportation System*", Human Factors and Ergonomics in Manufacturing & Service Industries Volume 10, Issue 4, pages 395–408, Autumn (Fall) 2000

[8] *COTSBOT project*- http://cotsbots.org/

[9]P. Kinney, "*ZigBee Technology: Wireless Control that Simply Works.*" ZigBee Alliance, Oct. 2003. [Online]. Available: www.zigbee.org/en/resources/

[10] *ATmega 162 microcontroller* [Online]. Available: www.atmel.com/atmel/acrobat/doc2513.pdf