International Journal of Engineering Applied Sciences and Technology, 2017
Vol. 2, Issue 8, ISSN No. 2455-2143, Pages 21-28
Published Online December 2017 in IJEAST (http://www.ijeast.com)

ROBUST CONTROL ALGORITHMS FOR HAND-ON VEHICLE ROUTING

M.N.SP. Therianos
Dept. of Automation Engineering,
PUAS, Athens, Greece

C. Drosos
Dept. of Automation Engineering,
PUAS, Athens, Greece

M. Papoutsidakis
Dept. of Automation Engineering,
PUAS, Athens, Greece

D. Tseles
Dept. of Automation Engineering,
PUAS, Athens, Greece

***Abstract*— In this paper, there is a study of basic Line Following algorithms for autonomous Lego Mindstorms vehicles. Initially a study of existing Line Following algorithms will take place in order to determine their field of application and at the same time outline their merits and disadvantages through case studies. Next, these algorithms will be implemented and applied to an autonomous vehicle that uses a single light sensor for line following. The use of a single light sensor, although it imposes a number of constraints and difficulties related to accuracy, it reduces the cost of the autonomous vehicle while at the same time examines whether it is possible to implement fast and accurate line following robots that relay on a single light sensor. Therefore mentioned algorithms will be implemented using the LabView for Mindstorms integrated programming environment (IDE). The implemented algorithms will be tested using regulated and unregulated motors. Regulated motors are self-correcting motors where self-corrections are realized by an embedded in the motor PID controller which regulated the power delivered to the motors. Unregulated motors on the other hand allow the developer to fine control the motors' behavior according to real-time data obtained from them and thus provide the developer to surgically control the motion of the autonomous vehicle. Based on the results that will be obtained a comparison of the two approaches will take place. This thesis effectively is divided in two parts: The first one addresses the study of line following algorithms for single line sensor autonomous vehicles. The second one is concerned with experimentation, where the algorithms will be tested and the results obtained compared in order to determine the advantages and disadvantages related to them. These results will aid the engineer-developer to evaluate and access the obtained results when selecting a line following algorithm for a task in hand.**

Keywords— **line following algorithms, autonomous vehicles, tuning, oscillation, odometry**

I. INTRODUCTION

Line following autonomous vehicles are used in various applications such as warehouse management, restaurant management, healthcare management systems, amongst others [1,2,3]. Line following, although at first seems a straight forward problem to solve, is a complex situation that encompasses a number of external factors that affect it. For instance, the surface on which the robot moves, the color of the line to be followed, the height of the light sensor from the surface that the robot moves etc., all play a significant role in achieving precise line following while maintaining acceptable rates of velocity. The velocity of an autonomous vehicle, hereafter AV, is a key aspect in AV management, since the time needed for a task to complete is proportional to the cost of the task. Thus, in the same time space, a fast moving AV completes more tasks than a slow moving one.

In this paper, we address the issue of maintaining high velocity for an AV that uses a single light sensor. The use of a single light sensor was based on an attempt to minimize the cost of the AV itself.

First, we provide a background on AVs and line following. The latter is based on the theory of automated control systems and in particular controllers. We then proceed, to examine in detail various different types of controllers presenting their benefits and drawbacks for the task in hand. Next, we provide enhancements on one particular controller, the PID controller, by introducing aspects of odometry in the controller's loop that ensure that velocity will not be sacrificed for precision. Finally, we compare our approach against a standard PID controller configuration in order to show our contribution to the field.

II. BACKGROUND

AVs are mainly used in hazardous environments such as nuclear reactors, as well as for tedious and/or reparative tasks

International Journal of Engineering Applied Sciences and Technology, 2017
Vol. 2, Issue 8, ISSN No. 2455-2143, Pages 21-28
Published Online December 2017 in IJEAST (http://www.ijeast.com)

such as medicine delivery to patients, warehouse keeping or even floor cleaning. To do so they have to follow a pre-determined well-defined course. Furthermore, they have to interact with their environment, acquiring data that will be used in their navigation and movement process. In doing so, they engage in three discrete tasks the results of which are computationally combined so that the AV completes in a predetermined spatio-temporal framework its activity(ies).

2.1 Odometry

Odometry is the use of data from motion sensors to estimate or even calculate change in position over time[4]. Used on their own such data are not entirely reliable because they themselves rely on the interaction of the mechanical parts of the motors. However, when used in conjunction with other navigational data they can provide a means to enhance the precision of any corrective action.

2.2 Perception of the surrounding space

AVs need to "know" the space in which they operate. Know refers to the process of acquiring environmental specific data of their surrounding space, evaluating them and using them in order to aid or enhance their navigational process. Such data include, but are not limited to, obstacle recognition, landmark recognition [5], arrival of an object to be stored in a warehouse.

2.3 Combined computational space

The term combined computational space refers to the algorithmic combination of data retrieved form odometry and the AV's surrounding space in order to established a well formed virtual path on movement that matches the one that the AV has to follow in order to achieve its task in hand.

2.4 Line following

In most applications an AV has to follow a line from point A to point B and in between these points has to perform some tasks like placing books on shelves or administering medicine to patients. The space in which the AV has to move is very well defined in terms of colored lines that the AV has to follow (Figure 1). For instance, in a hospital ward the AV follows a line from the nurses' station or booth to a particular patient and issue a certain medicine. We are not concerned on how the medicine is issued but only how the AV is going to travel from the nurses' booth to the patient without spilling the medicine. Bear in mind that issuing medicines is a time constrained problem. In other words different medicines have to be issued to different patients in a ward in a given time.

Figure 1 Line following AV

Line following in general has to take into account a number of different contributing factors:

- The sampling frequency of the light sensor
- The light sensor's quality and technical specifications
- The AV's desired velocity
- The sensor's height in reference to the line to be followed
- The shape and width of the line that the Av has to follow
- The color and material of the line
- The color and material of the surface on which the AV navigates.

2.4.1 Sensor sampling frequency

In many cases the sampling frequency of a sensor is such that certain values have to be discarded in order to obtain a reliable representation of the AV's world. If too many values are used to determine the combined computational space, the need for corrective adjustments increases with little significance to the process as a whole due to the emerging oscillation related to consecutive adjustments. This in itself affects the operational velocity of the AV.

If an AV oscillates, the period of the oscillation hinders the AV's overall velocity. A large period implies significant decrease of the AV's velocity since the AV movement is a broken line (zigzag) rather than a straight line. The amount of sample to be discarded is closely related to the AV's velocity. The higher the larger the number of samples that have to be discarded.

2.4.2 Sensor's quality and technical specifications

Different light sensors produce different results for a particular line and point on it. This is due to manufacturing issues. Manufacturing issues have to be taken into account during sensor calibration. Apart from manufacturing related issues sensor values may vary from sensor to sensor due to the height of the sensor from the line it follows. Experimentation has

International Journal of Engineering Applied Sciences and Technology, 2017
Vol. 2, Issue 8, ISSN No. 2455-2143, Pages 21-28
Published Online December 2017 in IJEAST (http://www.ijeast.com)

shown that a height between 0.5cm and 1.3cm, depending on the width of the line to be followed, produce reliable sample.
Another issue that is has to be taken under consideration is the vibration of the sensor due to the AV' movement and possible oscillation. If the sensor is extended far from the AV it is prone to vibration. Vibration will result in variable heights of the sensor during the AV's movement. This, in turn, will result in unreliable sensor reading. Consequently, the sensor has to be places as closely as possible to the AV and should be fastened to the AV as tightly as possible to eliminate any vibration. Shock absorbers are not to be considered as they do not work well given the weight of the sensor.

2.4.3 AV's desired velocity
The velocity that the AV has to move is fundamental to the precision of its movement. Firstly, as the velocity increases the sensor samples (given a fixed sampling rate) become less important. That is, not all sample are needed in producing the required adjustment in position, if one is needed.
The geometric nature of the path to be followed plays a very important role. A path that is made of short straights followed by sharp turns cannot be navigated solely by line following at high speeds. Do we reduce speed? The answer to this question is twofold. If speed is not of the essence then the answer is yes. However if speed is important then the answer is no and odometry data have to compensate for the inability to follow the path solely on light sensor readings. To sum it up, we need to follow a path at maximum velocity (whichever that is) without oscillation relying as much as possible on light sensor data and introducing odometry data whenever the need occurs i.e. sensor data in order to be reliable will result in velocity reduction.

2.4.4 Sensor height and position
Although, we mentioned that the light sensor has to be placed at a certain height form the line that forms the path to be followed we did not explain the reason as to why this is important.
Usually line following AV using light sensors operate the in reflected light intensity (RLI) mode. The concept behind RLI is that a light beam is emitted towards a surface and then the light returned due to reflection is measured. A bright surface will reflect most of the emitted light, while a darker surface will absorb some of the emitted light, thus reflecting light at a lower intensity.
Apart from manufacturing issues, the sensor's height form the reflective surface is key to reliable measurements.
If the radius of the light spot emitted from the sensor is small, then although the intensity of the reflected light depicts precisely the color of the surface, the surface area in question is small and hence may be rendered "out of interest". On the other hand, if the radius is big then although a larger are is covered, the intensity of the reflected light is not as reliable because light is affected by its surrounding environment (ambient light).
Sensor height is not the only issue related to the physical position of the light sensor. The exact position of the sensor related to the AV is as well an important issue. The sensor has to be placed on either sides of the AV so that the horizontal axis that runs through the center of the vehicle is parallel to the followed path. In order to ensure that this is the case the light sensor has to be placed in front of either wheel.

2.4.5 Shape and width of the line forming the path
A thin line may cause more navigational problems that a thick one. The thinner the line is the easier is to overshoot it. By overshooting we refer to the condition that due to an acute adjustment the light sensor crossed and bypasses the line a circumstance in which the AV can no longer navigate. Thick path lines are best for line following as it becomes almost impossible to raise the aforementioned condition.

2.4.6. Path line and surrounding surface color.
The lighter the color of the path line the harder it becomes to follow it. One must ensure that the contrast between the path line's color and that of the surrounding surface is such that a clear distinction between the two is possible at all times irrespective of ambient light and shadows.

2.5 Lines followed by lines
A path is a series of geometric shapes with a starting point and an ending point. Consequently a path is formed from straight lines, curves, turns, broken lines and intersections. Depending of its constituent parts a path requires different handling in order to achieve maximum reliability while maintaining high velocity levels.In this section we will examine each of a path's constituent parts in an attempt to show the issues associated with them.

2.5.1 Straight lines
Straight lines are the most straight forward geometric shapes for an AV to follow. Given that they are well formed, of an acceptable width, and of color that ensures a high contrast to the surrounding surface, straight line following is dependent to a good light sensor calibration and an accurate threshold estimation. A threshold in our case is the resulting reflected light intensity when the light sensor spot is half over the path line and half over the surrounding surface. As long as the light sensor spot abides to this condition the AV will move forward or backward without any deviation from the straight line. In order to follow a straight line fast one has to eliminate oscillation as oscillations results in overshooting as depicted in the following figure (**Error! Reference source not found.**2):

International Journal of Engineering Applied Sciences and Technology, 2017
Vol. 2, Issue 8, ISSN No. 2455-2143, Pages 21-28
Published Online December 2017 in IJEAST (http://www.ijeast.com)

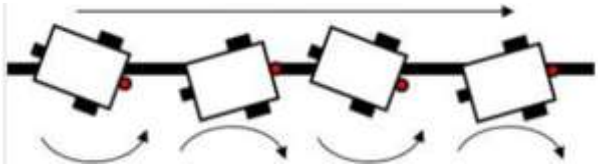


Figure 2 Oscillation during line following

2.5.2 Curves

Curves, as opposed to straight lines, are not so straightforward to follow. Since the AV tends to move on a straight line, a curve has to be tackled by consecutive adjustments in a manner that a curve is straightened. Oscillation to some extent will occur but should be kept to a minimal as far as its period is concerned. Figure 3 depicts the aforementioned issue

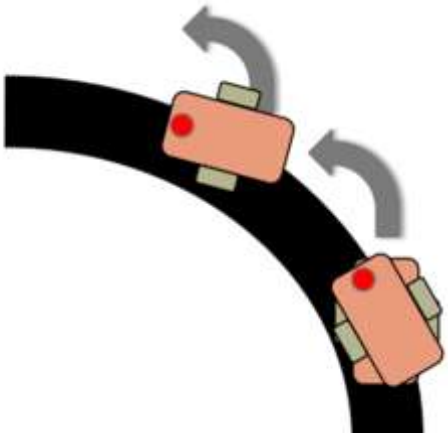


Figure 3 Curve related oscillation

Although smooth curves can be, to some degree, addressed as straight lines, sharp ones may need to be approached using a combination of light sensor data and odometry data.

2.5.3 Turns

Turns are perhaps the most difficult geometric shape an AV can encounter. Usually Av do not possess a steering system and as a result the only means for turning they possess is pivot and spin turns. These two types of turning differ significantly in their underlying concept. Pivot turns are achieved by stopping one motor and moving forward the other resulting in a shift in the AV's position. On the other hand, spin turns ensure no shift in the position of the AV. Spin turns are by far faster that pivot turns both in terms of velocity and resulting position. Figure 4 diagrammatically depicts the difference between the two types of turns.

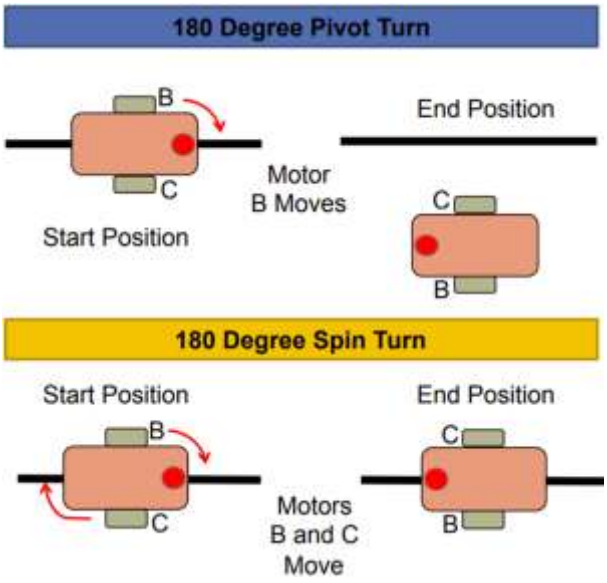


Figure 4 Pivot vs Spin turns

2.5.4 Broken lines

Broken lines can be viewed as consecutive straight lines joined in an angle. The angle itself can be wide, right or sharp. Be that as it may, broken lines are handled as straight lines and turns

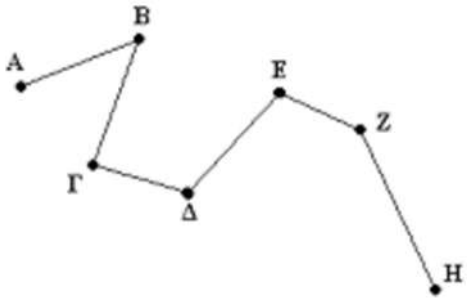


Figure 5 Broken line

2.5.4 Intersections

Intersections, due to their inherent geometric complexity, have to be handled using odometry data, since a line follower can detect them but cannot decide what to do with them unless specifically instructed. Specific instructions can only be given at specific points on the plane formed by the intersecting lines.

International Journal of Engineering Applied Sciences and Technology, 2017
Vol. 2, Issue 8, ISSN No. 2455-2143, Pages 21-28
Published Online December 2017 in IJEAST (http://www.ijeast.com)

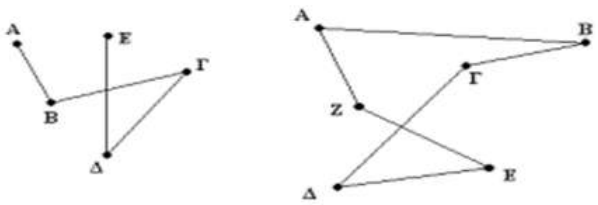


Figure 6 Intersections

III. ALGORITHMS

As mentioned previously, in this work, we examine algorithms based on the theory of automated control systems, and in particular controllers. Such algorithms operate on data received from the controller's environment, such the intensity of reflected light, and corrective measures according to the values received in order to achieve the task in hand. The complexity of such algorithms depends entirely on what they control and how they do so.

3.1 On/Off Controller

On/Off controllers are the simplest kind of controllers as their operation is based on Boolean logic. If the value that controls the controller is above a threshold the controller is turned on otherwise it remains dormant.

In our case if the light sensor read black the AV is drifting, more and more, towards the black line, and must turn to the right – assuming a configuration where the light sensor is placed on the right hand side of the AV. If the light sensor reads white then the AV is moving far away from the black line in the opposite direction and thus has to turn to left. On.Off controllers are suitable for straight line following but the inherent mode of operation generated large period oscillation. Figure 7 the implementation of an On/Off controller using LabView.

The drawbacks associated with this approach are related to:

1. The large period of oscillation
2. The low velocity due to 1
3. The inability to handle curves, turns, broken lines and intersection
4. The zigzag movement of the AV

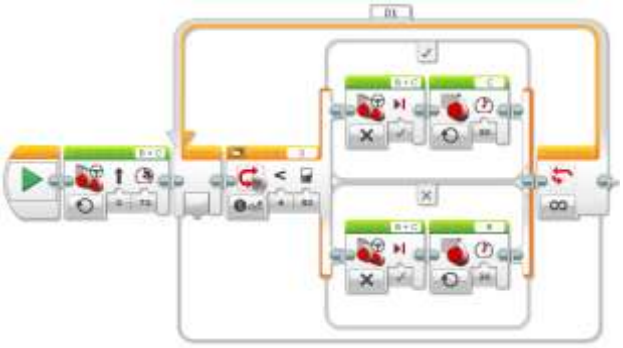


Figure 7 Implementation of an On/Off Controller

3.2 P-Controller

A P-controller or proportional controller is a linear control system with feedback where a correction is enforced on the control variable. This correction is proportional to the difference of the threshold and the value obtained from the light sensor. Such algorithms are best suited wherever the value we obtain (in our case the RLI value) changes rapidly over time. The controller's output is proportional to the error that occurs between the threshold and the measurement. In other word the controller's output is the product of the error times the proportional gain.

$$P_{out} = K_p e(t) + p0$$

where:

P_{out} is the controller's output

K_p is the proportional gain

$e(t)$ is the error that occurs from the difference of the threshold minus the current measurement

$p0$ is the controller's output when the error is 0

A serious drawback of this algorithm is its inability to eliminate the residual error.

Figure 8 Implementation of a P-Controller

Another issue associated with P-controllers is that they cannot possess knowledge of accumulated error. That in itself is considered as a limitation that hinders the ability of a p-controller to command a moving vehicle on a complex path. Error accumulation is significant because it incorporates all the errors that occurred during the Av's movement up to now. In as such, it can provide useful information regarding the AV's motion to this point and can be used to avoid extreme error variations in the future. Nevertheless a p-controller can:

1. Eliminate oscillation
2. Guarantee high velocity
3. Ensure straight line following at high speed
4. Reduced speed curve following

Thus it becomes obvious to the reader that it a significant improvement to its On/Off counterpart. However it cannot achieve:

1. Right angle turning
2. Broken line following

International Journal of Engineering Applied Sciences and Technology, 2017
Vol. 2, Issue 8, ISSN No. 2455-2143, Pages 21-28
Published Online December 2017 in IJEAST (http://www.ijeast.com)

3. Intersection handling

Due to these drawbacks we had to examine a variant of the P-controller, namely the PI controller

3.3 PI Controller

P-controllers rely on instantaneous data and possess no memory related to previous errors or previous error accumulation. The latter can be achieved through the use of a PI controller. Pi controllers use the notion of integration in order to obtain knowledge related to errors that occurred in the past. Time plays a significant role in this as shown by the following formula

$$K_p\Delta + K_I \int \Delta \, dt$$

Δ is the difference between the threshold and the sensor's value. Ki is the integration gain. Although this algorithm complements in this aspect the previous one it is not recommended for usage in situations where the situation or the real data change rapidly, and for this reason alone it will not be studied in depth. Furthermore, it can be considered as a subset of the PID algorithm presented in next section. However, for completion reasons alone, we provide the controller's block diagram

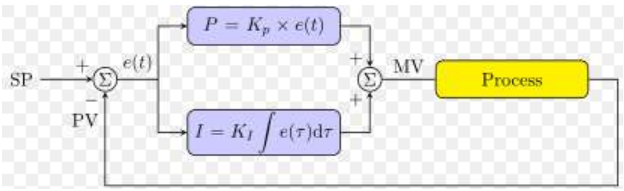


Figure 9 PI Controller

3.4 PID Controller

A PID algorithm could be considered as the flag ship of one variable feedback loop automated control systems controller. Not only it posses memory but also it provides for "error look ahead". Since the main contribution of this work is based on enhancements on the PID controller, in this section we will discuss to some detail how a PID controller operates with respect to line following.

Consider the AV depicted in Figure 10. The red circle represents the light spot emitted by the light sensor. It is placed on ahead of the AV on its right hand side. Motors B and C are motion motors. As one can see the sensor's spot is half over the black line (line to be followed) and half over the line's surrounding space.

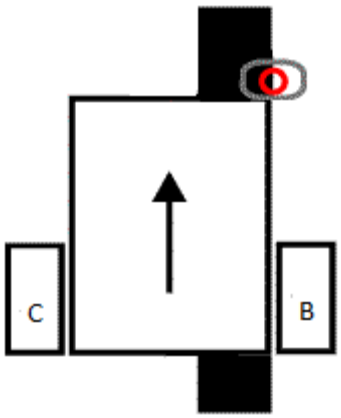


Figure 10 AV following a straight line

Assuming a threshold of 45 (RLI) when the sensor is as describe above we can safelly assume that a left turn occurs when the value of RLI is less than 45 and a right turn occurs when the value of RLI is greter than 45 (see figure 11)

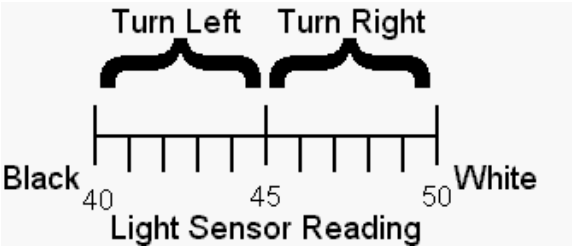


Figure 11 Sensor value interpretation

These assumptions will guarantee smooth line following of a straight line even when travelling at high speed. What happens when a turn is encountered. Smooth turn are handled in an acceptable manner. Sharp one they leave a lot to be desired. Let us now divide our range of values into three discrete areas. One for turning left, one for going straight and one for turning right as shown in figure 12

This approach is an improvement on the previous one. Adding a state (forward movement) betters the behaviour of the AV. Adding more states will fine grain the AV's movement altogether. Having said that the question that arises is how many such states are needed in order to achieve the smoothest possible movement on a complex path.

International Journal of Engineering Applied Sciences and Technology, 2017
Vol. 2, Issue 8, ISSN No. 2455-2143, Pages 21-28
Published Online December 2017 in IJEAST (http://www.ijeast.com)

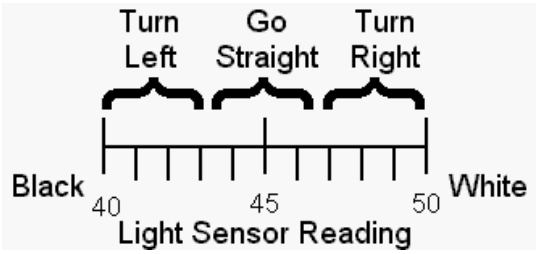
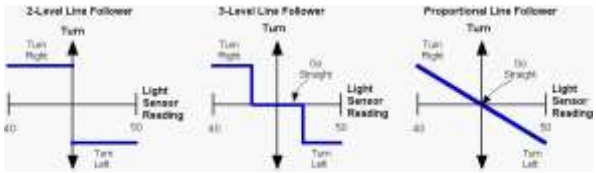


Figure 12 Accommodating for turns and straight line movement

The answer lies in error normalisation. We need to find the means to normalise the error instead of adding more and more states.Error normalisation is achieved via a P-Controler as we demonstrated above. Diagrammatically, this can be shown as follows:



So far what we have is nothing more that a P-Controller. Let us provide our controller with some memory. An integral is the accumulation of a value over time. Adding an integral part in the existing P-Controller we produce a controller, Pi controller, that posses the knowledge of its previous behaviour. Thus calculatng an adjustment in the AV's movement becomes

Turn = Kp*(error) + Ki*(integral)

Kp*(error) is the P-Controller part

Ki*(integral) is the memory

The integral is defined by

integral=integral _ error

Ki is the integral gain.

What remains now is to provide our controller with some kind of look ahead capability. We would like to be able to for see what is next in terms of what will be the next error that we will encounter. Although it seems farfetched it is not as hard as it sounds. The addition of a derivative accommodates for this need. We can predict the future with respect to errors by assuming that the next change in error will be the same as the previous one. In other words the values of the previous error and that of the next error are the same. This implies that the next error will be the the present error plus the change between the previous two readings of the light sensor. The change in error between two consecutive points or better instances is called derivative. The derivative can be considered as the slope of a line.

Hence, the value of the new error is as follows

(error) – (last error)

where error is the threshold minus the sensor's reading.

As a result, turning adjustmets for our AV are now calculated as follows:

Turn = Kp*(error) + Ki *(integral) + Kd*(derivative)

Bellow we provide the implementation of a PID controller

Figure 13 Error calculation

Figure 14 Inegral and derivative calculation

Figure 15 Steering calculation

Figure 16 PID main loop

IV. ENCHANGED PID

Although a PID controller is suitable for most line following situations there are some cases where the error changes are such that the AV overshoots or takes considerable time to settle. To accommodate for these two issues we altered the PID's main loop be adding a parametric time delay. The delay time is calculated according to the value of the foreseeable error so that noise is taken out of the equation. A beneficial side effect of the time delay proved to be the rejection of light sensor values that would not contribute in significant course adjustment. On the contrary they would "confuse" the controller as it would be unable to process them effectively.

We then proceeded in providing the controller's main loop with data related to odometry. Odometry data can, under certain circumstance, be used to calculate the exact position of a moving object. Adding odometry data to a PID levitates the burden of having to understand an intersection. Once we know that the Av is on the intersection, we know it because we travelled so far (specific distance) we can take the necessary

actions to follow the direction we have to. For instance if the medicine has to be delivered to patient X whose bed is on the left branch of the intersection we know that we have to turn left and then follow the line to the patient's bed.

From experimentation we concluded that enhancing a PID loop with time delay and odometry data not only made it operate properly on right angles and intersections, but also contributed in maintaining high speed while doing so. Overall we calculated a 20%-25% increase in the operational speed of our enhanced PID for a given path that incorporated straight lines, curves, acute as well as smooth angles and intersection.

Fine tuning of our enhance PID is an area that requires further investigation. Although the **Ziegler-Nichols** tuning method still applies in our case it does not accommodate for odometry data. Adaptive tuning is also an area that can be researched. By adaptive tuning we refer to the process that the controller's tuning is incorporated into its main loop and depends on values within its own computational space. Fro instance, variable gains could be introduces that will be calculated according to the outcome of previous errors, derivatives and integrals. The area of research, is our belief, is open for further investigation and experimentation.

V. CONCLUSIONS

In this paper we address the issue of line following using a single light sensor autonomous vehicle. Key to our approach was to maintain the Av's velocity as high as possible. First, we addressed issues related to line following such as straight line, curve, turn, broken line following as well as intersection handling, We then proceeded by examining common practices to line following. In doing so we reviewed existing algorithms, all from the field of automated control systems, pinpointing their limitations with respect to the task in hand. On/Off controllers were proved inadequate in following a complex path as did P-controllers. PI and PD controllers resolved some of the issues that their simpler counterparts were unable to do so but were limited to straight lines and curves. The PID controller proved to be a significant improvement over the other alternatives, limited however by sharp turns and intersections. In order to resolve the inherent limitations of the PID controller, we presented an enhancement of the PID controller. This enhancement concerns two aspects of the controller. The first one has to do with the controller's main loop. By adding a time delay loop we ensured that a number of sensor readings were not taken in account in the computational space of the controller thus eliminating noise and redundant information. The second one, an innovation by itself, was the introduction of odometry data into the controller's loop hence creating a two variable PID controller. The innovation is that the controller's loop exhibits an "intelligence" related to when to use odometry data. When the absolute value of the error increases significantly odometry data are used to compensate. As a result we implemented a PID controller able to follow a complex path made of sharp curves, sharp turns and inpersections.

ACKNOWLEDGEMENT

All authors would like to express their gratitude to the Post-Graduate Program of Studies "Automation of Productions and Services" of PUAS, for the financial support to undertake this research project.

VI. REFERENCE

[1] Deepak Punetha, Neeraj Kumar, Vartika Mehta, Development and Applications of Line Following Robot Based Health Care Management System, International Journal of Advanced Research in Computer Engineering & Technology (IJARCET), Vol.2, No.8, 2013, pp. 2446-2450.

[2] Harshit Kaur Chawla , Pallavi Rana , Priyasi, An RFID based warehouse robot, International Journal of Technical Research and Applications, Special Issue 39 (KCCEMSR), March 2016, pp. 91-93,

[3] Pakdaman, M.; Sanaatiyan, M.M., Design and Implementation of Line Follower Robot, Computer and Electrical Engineering, 2009. ICCEE '09, Vol.2, No., pp.585-590, 2009.

[4] https://en.wikipedia.org/wiki/Odometry

[5] Hatem Nasr and Bir Bhanu, Landmark recognition for autonomous mobile robots, Proceedings. IEEE International Conference on Robotics and Automation1988, pp. 1218-1223

[6] M. Zafri Baharuddin, Izham Z. Abidin, S.Sulaiman Kaja Mohideen, Yap Keem Siah, Jeffrey Tan Too Chuan, Analysis of Line Sensor Configuration for the Advanced Line Follower Robot, http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.523.6444&rep=rep1&type=pdf. 2005

[7] Gregory Dudek and Michael Jenkin, Computational Principles of Mobile Robotics, Cambridge University Press, 2010.