System Design Project Report 4

Group12 - School of Informatics
Seyed Behzad Tabibian

I. Introduction

Following developments of previous milestone, Three major developments are now completed. First, Potential Function was tested on the robot and become fully functional. Second, a new algorithm was developed to deal with situations that ball is near the wall. Third, vision code base that is documented and cleaned up.

II. Path Planning

A. Potential Function

Potential Functions which was extensively discussed in previous reports was tested on robot. Major barriers to use this algorithm was instability in input data from vision stack and also physical constraints of the robot itself. Potential Field algorithm does not take into account robot constraints which in turn makes the control problem specific to nonholonomic robot[1]. Figure 1 shows how different parameters of robot used to calculate linear and angular velocities. Based on various tests, equation 1 is proposed to apply the calculated velocity to the robot.

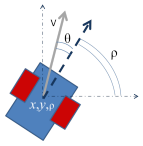

Fig. 1. Simple model of nonholonomic differential drive robot

$$V_{linear} = \left\|v\right\| cos(\theta), V_{angular} = \frac{1}{\left\|v\right\|}\theta \qquad (1)$$

The proposed method has an important property that is it does not force the robot to make sharp turns while it is not too close to the destination point and therefore starts to make sharper turns as it gets closer to destination. Also, angular velocities are controlled with a minimum and maximum velocities so robot never tries to turn faster than some specific threshold.

B. Kicking ball near walls

An algorithm was also implemented to deal with ball near the wall. This work was implemented and completed with Marc Howarth. Developed algorithm is based on the fact that robot can kick the ball toward a wall with specific angle so that when it bounces back, it goes toward opponent goal. Figure 2 depicts how this algorithm works. Using equation

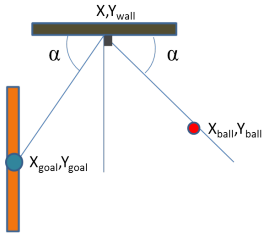

Fig. 2. Position of Different objects near wall. Aim is to calculate X given other positions

2 rebound position on the wall is calculated so that Figure 2 holds.

$$\frac{Y_{ball} - Y_{wall}}{X_{ball} - X} = \frac{Y_{goal} - Y_{wall}}{X_{goal} - X} \qquad (2)$$

$$X = \frac{X_{ball}(Y_{goal} - Y_{wall})}{Y_{ball} + Y_{goal} - 2Y_{wall}} + \qquad (3)$$

$$\frac{X_{goal}(Y_{ball} - Y_{wall})}{Y_{ball} + Y_{goal} - 2Y_{wall}} \qquad (4)$$

In practice this algorithm did not work well because ball is usually moving on the pitch and therefore result may not be accurate but it always moves the ball away from the wall and sends it close to opponent goal.

III. Vision Subsystem

I also focused on documenting and cleaning up the vision code. Code base now contains various utility functions for detecting thresholds, preprocessing algorithms, Machine Learning library and also different methods for calculating orientation of an object. A concise documentation is produced using DoxyGen[2] containing call graphs, description of data structures and functions. Our goal is to publish this API as a platform for next year students.

REFERENCES

[1] R. Siegwart and I. R. Nourbakhsh, *Introduction to Autonomous Mobile Robots*, 1st ed. The MIT Press, 2004.

[2] D. van Heesch, "Doxygen website," 2011. [Online]. Available: http://www.stack.nl/ dimitri/doxygen/