System Design Project - Individual Report 4

Biser Hong - s0840625 - Group 12

1 Introduction

This report documentsl the major changes and improvements made since Milestone 3 and the updates on previously set goals. These include restructuring the code base, implementing a generalized collision detection algorithm, adding additional drawing functionality and extending the prediction capabilities started just before Milestone 3.

2 Simulator updates

2.1 Collision detector

Having had troubles managing all collisions by passing objects around to all other objects, I designed a central collision detection mechanism. Objects that need support for collisions are registered with a collision detector and all relevant parties are notified in the case of collision. The collision detector requests each object to send their shape on every update cycle. For every corner of that shape a test is made to establish whether it falls within any of the other registered objects' shape. If this is the case, a collision packet, which includes the type of collision and the side of the shape that is closest to the other shape's corner, is constructed and sent to each object participating in the collision.

2.2 Improved drawing

As the strategy was becoming more and more complex, efficient debugging methods needed to be employed, especially in the case of finding logical errors in the code, which are usually harder to detect. Since any strategy involves fine tuning and certain thresholds are inevitable, a simple but useful transparent oval is drawn whenever there is a need to visualize, for example, the area around a point to which the robot needs to move. This ensures calculations are always correct and strategy is proceeding as it should (see Figure 1).

3 Movement prediction

Having laid out the basis for a prediction algorithm at the end of Milestone 2, I generalized it to be used for predicting any point, after which Marc extended it and fixed several inaccuracies. The major problem was that the angle of the fitted line through the history of positions we store was not enough to determine the direction of the movement. This was fixed by taking into account the x-positions of the previous points stored in a buffer. The distance of the predicted point depends on the speed of the movement, which is calculated by using the distance between the each of the previous points. To ensure accuracy, we plot our predictions as well.

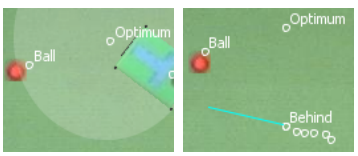

Figure 1: Threshold visualization and movement prediction