UNIVERSITY OF SOUTHERN DENMARK

COURSE PROJECT

ROBOTICS 5TH SEMESTER - FALL 2018

# Marble finding robot


Alexander Tubæk Rasmussen
alras16@student.sdu.dk

Kenni Nielsen
kenil16@student.sdu.dk

Marcus Enghoff Hesselberg Grove
grov16@student.sdu.dk

# 1 Abstract

# Contents

# 2 Introduction

## 2.1 Problem statement

### *2.1.1 Problems*

The following problems are stated to better describe the focus points throughout the project. XXXX

1. How can an algorithm, which can find marbles in a closed map, be designed?
2. How can this algorithm be optimized, so the robot can find the ball faster, based on previous trails?
3. How can marbles and obstacles be detected from lidar data?
4. How can marbles and obstacles be detected from camera data?
5. How can a Fuzzy-control algorithm be constructed?
6. How can computer vision be used to detect marbles and obstacles, and to construct the map?

XXXX

### *2.1.2 Limitations*

The robot must be able to find marbles in a closed map and optimize itself based on previous trails. Thus the following limitations to the algorithms is listed:

1. To detect marbles and obstacles.
2. To construct a map.
3. To optimize for the shortest distance and time.

QT Creator, Fuzzy Lite, Gazebo and Matlab will be used throughout the project.

## 2.2 Readers guide

# 3 Design

## 3.1 Environment

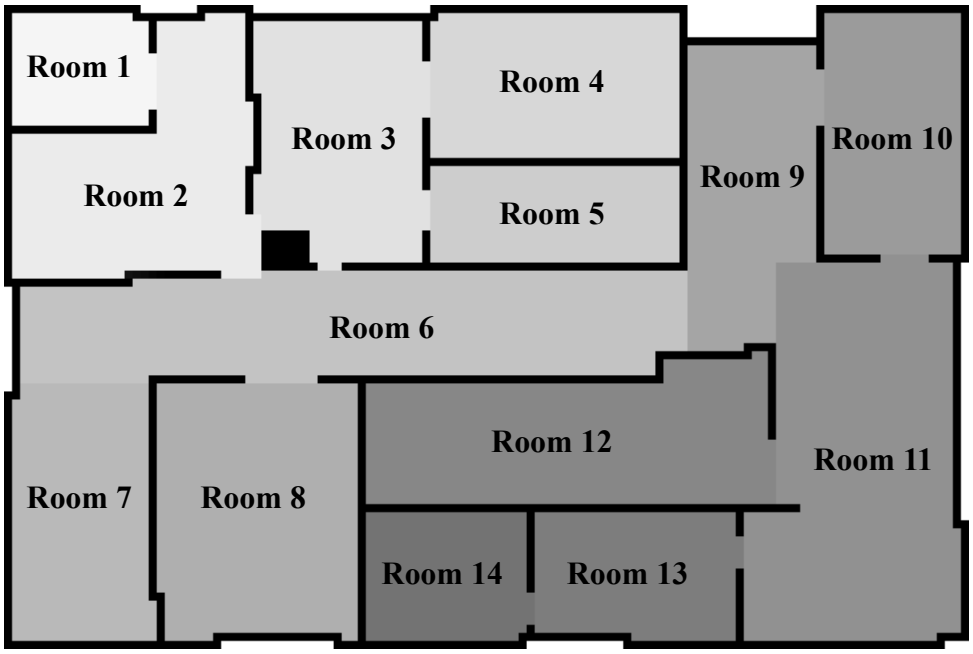


*Figure 1: Illustration of how the world "bigworld" have been divided onto rooms*

## 3.2 Line detection

### *3.2.1 Line Parameters*

First let $Z = \{z_1, \ldots, z_n\}$ be a dataset of points, where each point is represented in cartesian coordinates $z_i = \langle x_i, y_i \rangle$. From this dataset, one can detect lines by looking at the linear relation between its x and y-coordinate, which is represented as follows:

$$y = ax + b$$

where a is the slope of the straight line and b is the y-axis intersection. Using this line representation, one can have some complications detecting vertical lines, because a→∞. To avoid this, one can represent a line in the plane by its normal form, using the following formula:

$$l = \langle r, \alpha \rangle$$

where r is the distance from the origin to the closest point on the line and α is the angle between the x-axis and the plane normal. By using $z_i = \langle x_i, y_i \rangle$ as points on the line $l = \langle r, \alpha \rangle$ (see figure xx) leads to the expression:

$$r = x \cdot \cos(\alpha) + y \cdot \sin(\alpha) \tag{3.1}$$

Missing figure

The points are converted into polar coordinates, changing the line presentation to the following:

$$r = \rho \cos(\theta) \cos(\alpha) + \rho \sin(\theta) \sin(\alpha) \tag{3.2}$$

$$r = \rho \cos(\theta - \alpha) \tag{3.3}$$

### 3.2.2 *Orthogonal distance from point to line*

The shortest distance from a given point $\langle x_i, y_i \rangle$ to a line $l = \langle r, \alpha \rangle$, denoted by $d_i$, is computed by drawing a line $l_i$, that goes through this point and are parallel to the line $l$ (see figure yy). This line $l_i$ is given by the following formula:

$$r_i = x_i \cdot \cos(\alpha) + y_i \cdot \sin(\alpha) \tag{3.4}$$

Missing figure

The separation between those two parallel lines is the difference $d_i = r_i - r$, because both lines have the same $\alpha$. The desired distance is also called the orthogonal distance and is determined as follows:

$$d_i = x_i \cdot \cos(\alpha) + y_i \cdot \sin(\alpha) - r \tag{3.5}$$

This only applies if we assume that there is noise on the measurements.

Likewise, the points in the formula for the orthogonal distance can be converted into polar coordinates, changing the line presentation to the following:

$$d_i = \rho_i \cos(\theta_i) \cos(\alpha) + \rho_i \sin(\theta_i) \sin(\alpha_i) - r \tag{3.6}$$

$$d_i = \rho_i \cos(\theta_i - \alpha) - r \tag{3.7}$$

The line parameter r is determined using least square line fitting, while α is determined using the total least square problem.

### 3.2.3 *Least square line fitting*

If you want to fit a straight line to a dataset with n measurements having errors, you need to consider each measurement to be equally uncertain. This means that you can sum the square of all measurement point errors together to determine an overall fit between the line and all the measurements:

$$S = \sum_{i=1}^{n} d_i^2 \tag{3.8}$$

$$S = \sum_{i=1}^{n} \left(\rho_i \cos(\theta_i - \alpha) - r\right)^2 \tag{3.9}$$

It is considered to be an unweighted least squares solution, because no distinction is made from among the measurements. A condition for minimizing S is done by solving the nonlinear equation system with respect to the line parameters ($r$ and $\alpha$).

$$\frac{\partial S}{\partial r} = 0 \qquad \frac{\partial S}{\partial \alpha} = 0$$

By solving the nonlinear equation system, an expression for these two line parameters are determined to the following:

$$\alpha = \frac{1}{2} \arctan \left( \frac{\sum_{i=1}^{n} w_i \rho_i^2 \sin(2\theta_i) - \frac{2}{\sum_{i=1}^{n} w_i} \cdot \sum_{i=1}^{n} \sum_{j=1}^{n} w_i w_j \rho_i \rho_j \cos(\theta_i) \sin(\theta_j)}{\sum_{i=1}^{n} w_i \rho_i^2 \cos(2\theta_i) - \frac{1}{\sum_{i=1}^{n} w_i} \cdot \sum_{i=1}^{n} \sum_{j=1}^{n} w_i w_j \rho_i \rho_j \cos(\theta_i + \theta_j)} \right) \tag{3.10}$$

$$r = \frac{\sum_{i=1}^{n} w_i \rho_i \cos(\theta_i - \alpha)}{\sum_{i=1}^{n} w_i} \tag{3.11}$$

### *3.2.4 Total least square line fitting*

The problem of fitting a straight line to a dataset with n measurements having errors, can be determined using the following sum:

$$\chi^2\left(l, z_1, ..., z_n\right) = \sum_{i=1}^{n}\left[\frac{\left(x_k - X_k\right)^2}{u_{x,k}^2} + \frac{\left(y_k - Y_k\right)^2}{u_{y,k}^2}\right] \tag{3.12}$$

where $(x_k, y_k)$ are the points coordinates with corresponding uncertainties $(u_{x,k}, u_{y,k})$ and $(X_k, Y_k)$ denote its corresponding point of the straight line $l$. In case of fitting the best line, minimizes the expression for $\chi^2$ by setting $u_{x,k} = u_{y,k} = \sigma$ and $k = 1, \ldots, n$. This reduces the problem to the so-called total least square problem and minimizing is equal to minimizing the orthogonal distance of the measurements to the fitting line. Therefore, in the case of fitting the best line minimizes the expression in the equation above.

$$\chi^2\left(l; Z\right) = \sum_{i=1}^{n}\frac{d_i^2}{\sigma^2} \tag{3.13}$$

$$= \sum_{i=1}^{n}\frac{\left(x_i\cos(\alpha) + y_i\sin\ (\alpha) - r\right)^2}{\sigma^2} \tag{3.14}$$

$$= \frac{1}{\sigma^2}\cdot\sum_{i=1}^{n}\left(x_i\cos(\alpha) + y_i\sin\ (\alpha) - r\right)^2 \tag{3.15}$$

A condition for minimizing $\chi^2$ is done by solving the nonlinear equation system with respect to each of the two line parameters ($r$ and $\alpha$).

$$\frac{\partial\chi^2}{\partial r} = 0 \qquad\qquad \frac{\partial\chi^2}{\partial\alpha} = 0 \tag{3.16}$$

By solving the nonlinear equation system, an expression for these two line parameters are determined to the following:

$$\alpha = \frac{1}{2}\arctan\left(\frac{-2\sum_{i=1}^{n}\left[\left(x_i - \overline{X}\right) - \left(y_i - \overline{Y}\right)\right]}{\sum_{i=1}^{n}\left[\left(x_i - \overline{X}\right)^2 - \left(y_i - \overline{Y}\right)^2\right]}\right) \tag{3.17}$$

$$r = \overline{X}\cos(\alpha) + \overline{Y}\sin(\alpha) \tag{3.18}$$

## 3.3 Marble detection

## 3.4 Tangent bug

## 3.5 Search strategy

## 3.6 Q-learning

Here is something on Q-learning bla bla bla

***Pseudocode: Q-learning***

Algorithm parameters: step size: $\alpha \in$ [0,1], small $\epsilon > 0$
Initialise $Q(s, a)$, for all $s \in S^+$, $a \in A(s)$, arbitrarily except that $Q(terminal, \dot{)} = 0$
Loop for each episode:
    Initialise $S$
    Loop for each step of episode:
        Choose $A$ from $S$ using policy derived from $Q(e.g., \epsilon - greedy)$
        Take action $A$, observe $R$, $S^{'}$
        $Q(S, A) \leftarrow Q(S, A) + \alpha \left[ R + \gamma \max_a Q(S', a) - Q(S, A) \right]$
        $S \leftarrow S^{'}$
    until $S$ is terminal

# 4 Implementation

## 4.1 Line detection

## 4.2 Marble detection

## 4.3 Tangent bug

## 4.4 Search strategy

## 4.5 Q-learning

# 5 Discussion

# 6 Conclusion

# Appendices

## A Tests

### A.1 Basic test

This test is based on value iteration

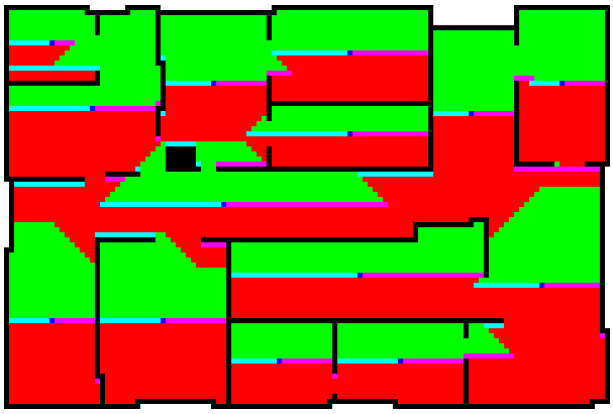

*Figure 2:* $dt = 0.1$