[entry]none/global/

Path Finding

- as used in Rescuing Robots -

Project Report
1-ED3

Aalborg University
Electronics and Computer Engineering

We used LaTeXfor typesetting this report, Code::Blocks for prototyping the code and IAR-Workbench for programming the microcrontroller.

Electronics and Computer Engineering
Aalborg University
http://www.aau.dk

AALBORG UNIVERSITY
STUDENT REPORT

Title:
Path Finding

Theme:
Analog Instrumentation

Project Period:
Fall Semester 2017

Project Group:
1-ED3

Participant(s):
Daniel Frederik Busemann
Razvan-Vlad Bucur
Troels Ulstrup Klein

Supervisor(s):
Akbar Hussain

Copies: 1

Page Numbers: 29

Date of Completion:
December 6, 2017

Abstract:

This project is about path finding on a grid based map. To test our theoretical work, we decided to build a small four wheeled robot. Therefore we needed to think of a way to move the robot, a way to observe our surroundings and a way to manage the collected data.

Contents

Todo list

Preface

This report was made by three students from Aalborg University Esbjerg attending the Electronics and Computer Engineering course, with the purpose of completing the P3 project in the third semester. From this point on, every mention of we refers to the three co-authors listed below.

Aalborg University, December 6, 2017

Daniel Frederik Busemann
<dbusem16@student.aau.dk>

Razvan-Vlad Bucur
<rbucur16@student.aau.dk>

Troels Ulstrup Klein
<tklein11@student.aau.dk>

Chapter 1

Introduction

In this project we want to talk about path finding algorithms, with the main focus of building an example implementation on a small scale.

We expect the reader to have a basic understanding of math, programming and simple physics. But will explain the applied topics.

how to reference to another chapter: Read more about path finding in Chapter 5.

1.1 Examples

You can also have examples in your document such as in example 1.1.

Example 1.1 (An Example of an Example)
Here is an example with some math

$$0 = \exp(i\pi) + 1 \,. \tag{1.1}$$

You can adjust the colour and the line width in the `macros.tex` file.

1.2 How Does Sections, Subsections, and Subsections Look?

Well, like this

1.2.1 This is a Subsection

and this

This is a Subsubsection

and this.

A Paragraph You can also use paragraph titles which look like this.

A Subparagraph Moreover, you can also use subparagraph titles which look like this. They have a small indentation as opposed to the paragraph titles.

Is it possible to add a subsubparagraph?

I think that a summary of this exciting chapter should be added.

Chapter 2

Movement

For our robot to be able to show the results of path finding, it needed to be able to move. BlaBla

We decided to move only along a simple 2D grid-like structure, therefore wheels were the easiest solution.

we need to make this flow nicer into the next section

2.1 Motors

A stepper motor is a motor that moves one step at a time, with its step defined by a step angle.

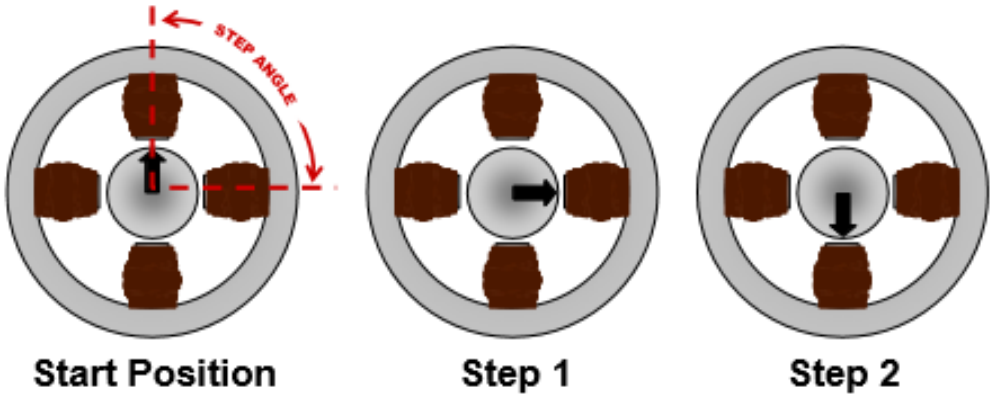


Figure 2.1: Step Angle

Figure 2.1 represents a stepper motor that requires 4 steps to complete a 360 degrees rotation. This determines the step angle to be 90 degrees. The main components of a stepper motor are represented in the image below, and they consist of stators, windings(phases), and rotor. The part that moves, is the rotor, which can be magnetized or not, depending on the type of stepper motor.

draw arrows on the figure instead?

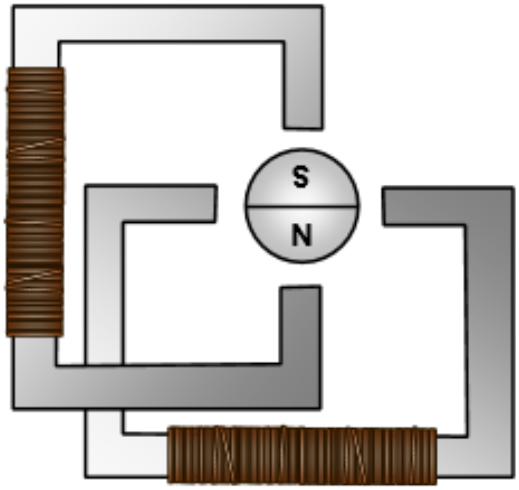


Figure 2.2: Main Components

By applying a voltage across one of the windings, current will start flowing through it. By using the right-hand rule, the direction of the magnetic flux can be determined. This is represented in figure 2.3.

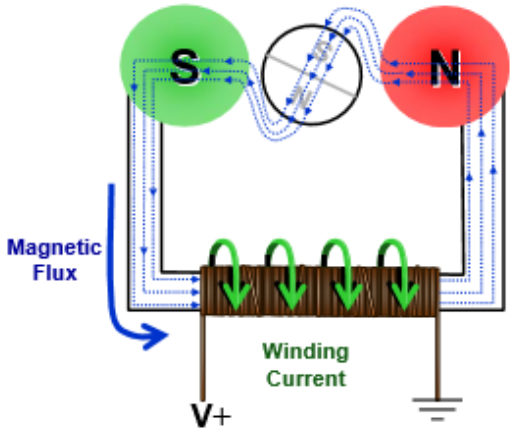


Figure 2.3: Direction of Magnetic Flux

The flux will want to travel through the path that has the least resistance. This determines the rotor to change its position to minimize resistance. This is shown in figure 2.4.

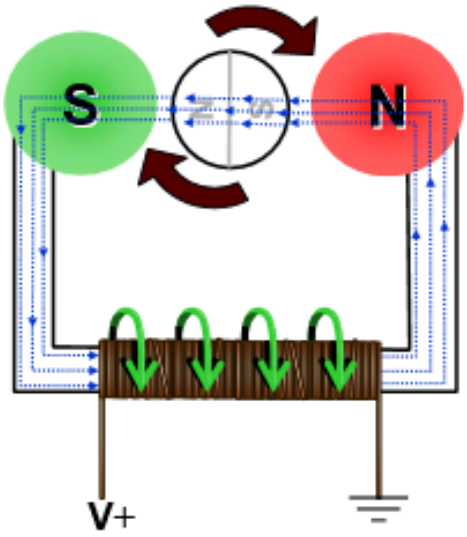
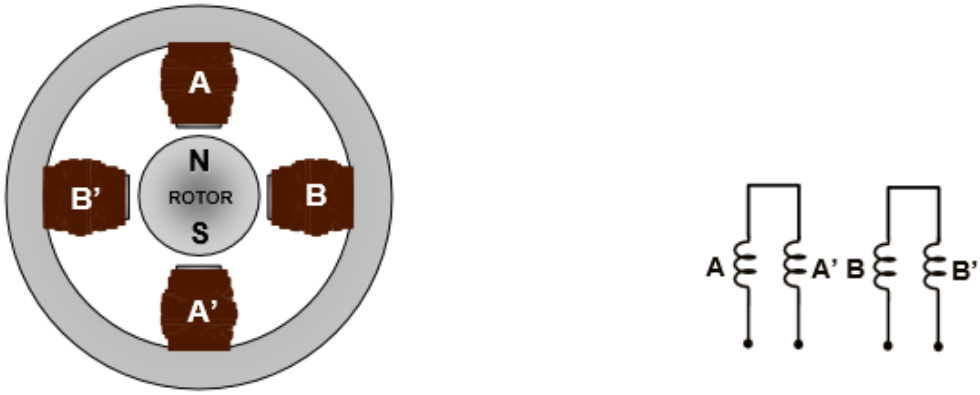


Figure 2.4: Direction of Magnetic Flux 2

2.1.1 Types of Stepper Motors

Permanent Magnet Motor

This type of stepper motor has a magnetized rotor. Each winding, will be subdivided into two, to better understand how to motor functions. Figure 2.5 represents the windings, and how they are distributed inside a stepper motor.

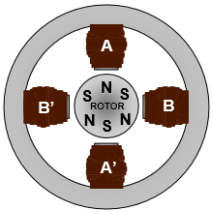


(a) Rotor **(b)** Winding

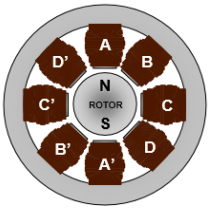
Figure 2.5: Basic structure of a motor

The resolution of the motor can be improved in two ways, either by increasing the number of pole pairs in the rotor itself, or by increasing the number of phases

as shown in figure 2.6.



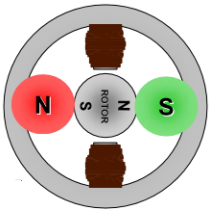
(a) Increased Pole Pairs



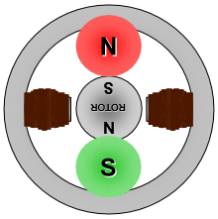
(b) Increased Number of Winding

Figure 2.6: Increased resolution

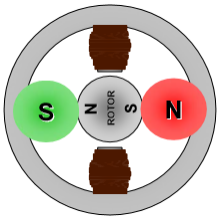
To rotate the motor, simply apply a voltage across the windings in a sequence. A full rotation is shown in the images below, with the corresponding phase energized.



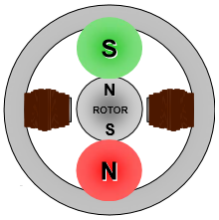
(a) 1st Step



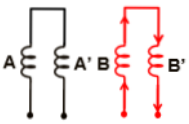
(b) 2nd Step



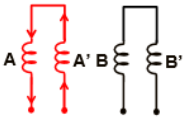
(c) 3rd Step



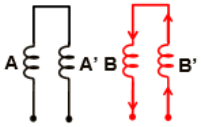
(d) 4th Step



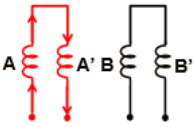
(e) 1st Step Winding



(f) 2nd Step Winding



(g) 3rd Step Winding

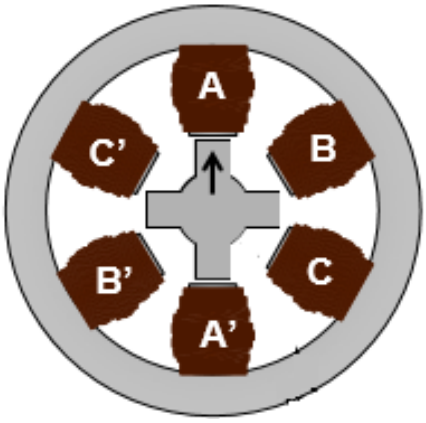


(h) 4th Winding

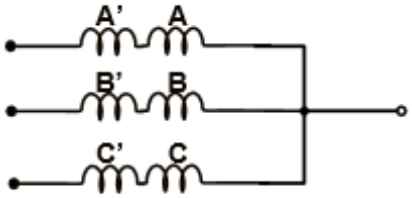
Variable Reluctance Motor

This type of motor, uses a rotor that is not magnetized, and has a number of teeth as seen in the image below. The windings are configured differently, as depicted in the second figure, all having a common voltage source but with each end being separate. They usually have 3 or 5 windings. Greater precision can be achieved by adding more teeth to the rotor.

To spin the motor, each winding is energized one at a time, and the rotor rotates in such a way, to minimize reluctance. Some of the differences, between this type

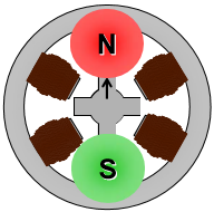


(a) Non Magnetized Rotor

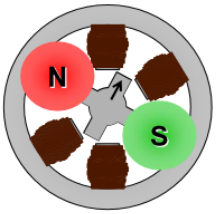


(b) Windings

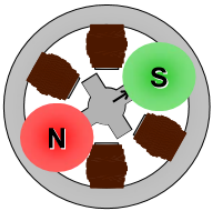
of stepper motor and the permanent magnet motor, are that, in order to spin the motor in a direction, the windings have to be energized in a reverse sequence, as depicted in the images below. In addition, the step angle is actually half of the one of a permanent magnet motor with the same number of windings is.



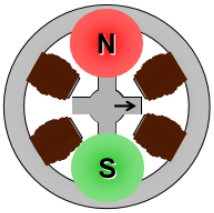
(a) 1st Step



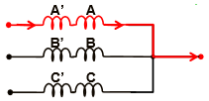
(b) 2nd Step



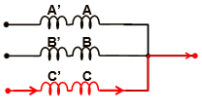
(c) 3rd Step



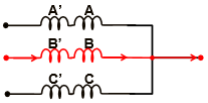
(d) 4th Step



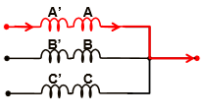
(a) 1st Step Winding



(b) 2nd Step Winding



(c) 3rd Step Winding



(d) 4th Winding

Hybrid Stepper Motor

Hybrid stepper motors borrow characteristics both the previous ones. The picture below shows the two of the main components of the hybrid stepper motor. On the left side, the stator can be seen consisting of 8 poles. On the right side, is represented the rotor. The rotor consists of two sets of teeth, corresponding for the two poles, north and south, respectively.

It is important to notice two additional things. The first, is that the teeth on the

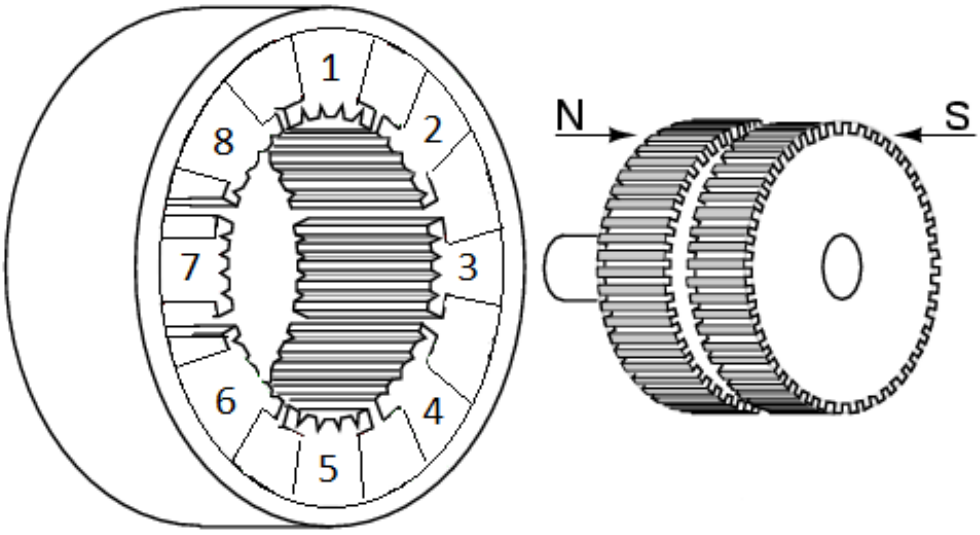
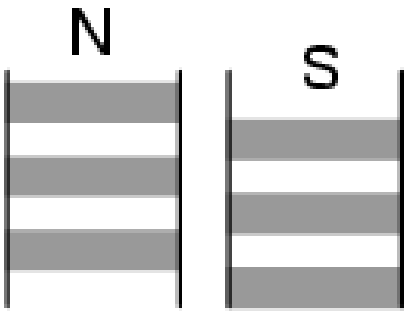
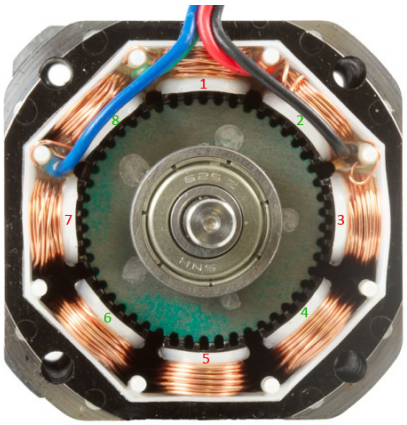


Figure 2.7: Stator And Rotor

rotor are not aligned but are interleaved, as shown in the picture below. The second, is the placement of the stator teeth in respect to those of the rotor, represented in the second picture.



(a) Interleaved Teeth



(b) Stepper Motor Inside

It can be observed in the second picture, the windings with numbers 1 and 5 are completely aligned with the teeth of the rotor. Windings number 3 and 7 are completely unaligned, while the others are half aligned. This results in higher precision and higher torque offered by the hybrid stepper motor, depending on the stepping method used.

The picture below represents the front view of the hybrid stepper motor to-

gether with the configuration of the two windings.

(a) Front View Stepper Motor **(b)** Winding Configuration

It is important to notice that, even though the motor has only two windings, each individual winding energizes 4 stator poles.

The pictures below represent the way this motor operates.

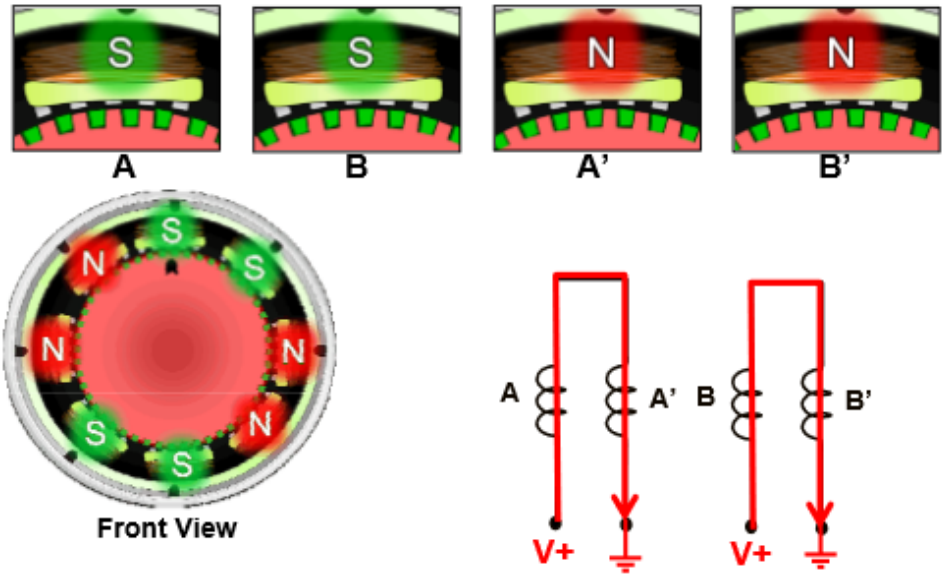


Figure 2.8: Something

By applying a voltage to both windings, the current flow can be controlled, thereby controlling the polarity of each stator pole, thus controlling the direction of the motor. Notice that, initially, poles A and A' are completely aligned, and poles B and B' are half aligned.

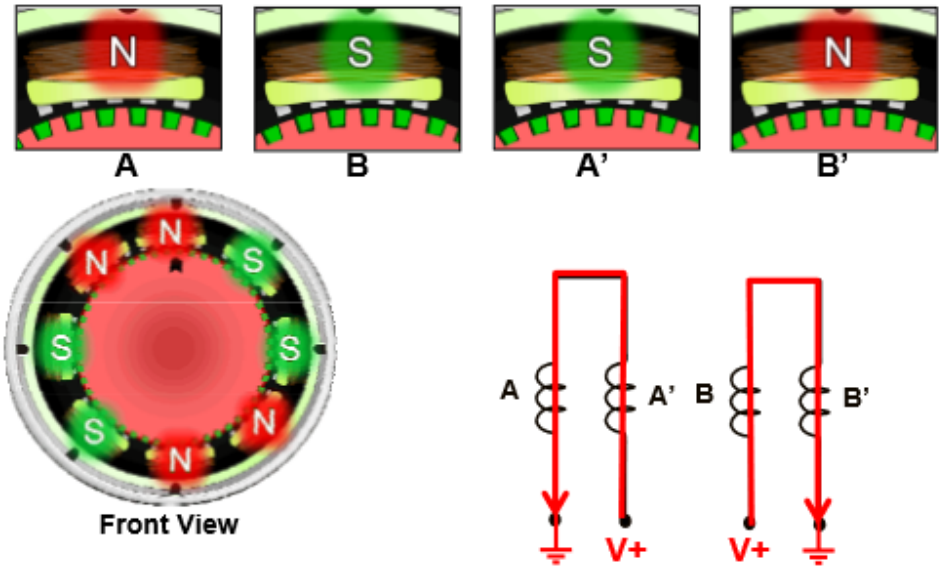


Figure 2.9: Something

Next step involves changing the direction of the current in winding A by applying a voltage at the other end of the winding. Even though only the current in winding A has been changed, all stator poles are aligned differently. Poles A and A' are now half aligned, and poles B and B' are completely aligned.

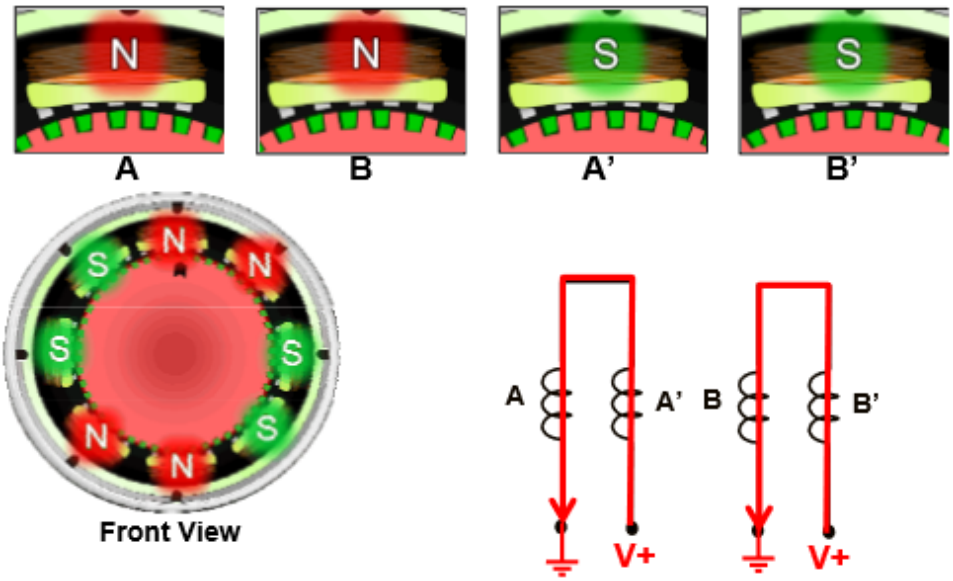


Figure 2.10: Something

Now, changing the direction of the current in winding B, changes the polarity of the stator poles B and B', again, determining a change in the alignment of all

stator poles. A and A' are now completely aligned, and stator poles B and B' are half aligned. The positions of the stator poles now correspond to those of the first step.

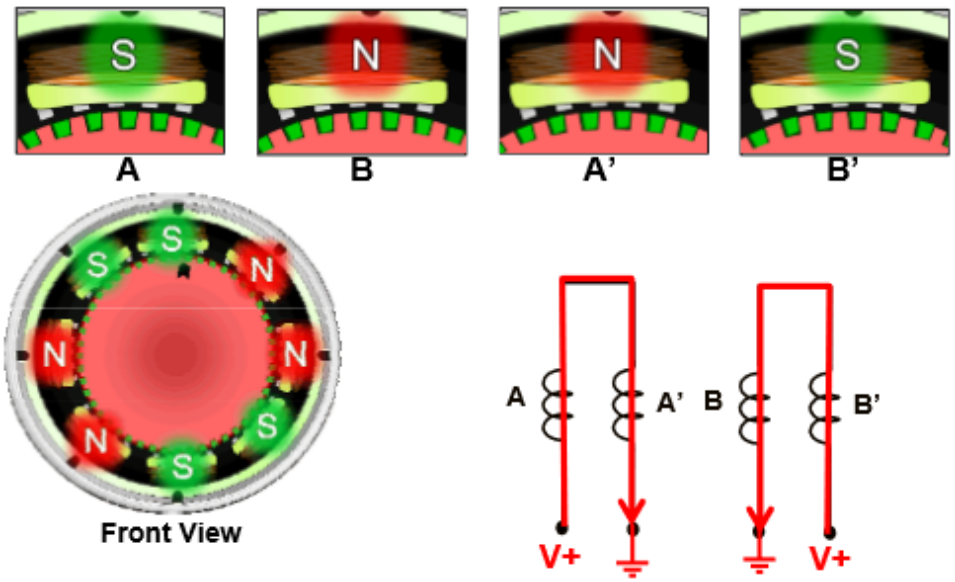


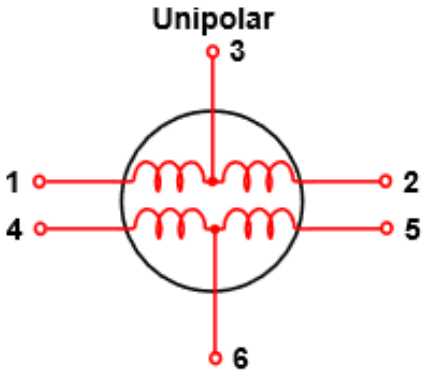
Figure 2.11: Something

Finally, again changing the direction of the current in winding A, determines the rotor to move another step. Notice the alignment of the stator poles. A and A' are half aligned, while B and B' are fully aligned. By changing the direction of the current in winding B, the motor arrives in the initial state, thus repeating the sequence.

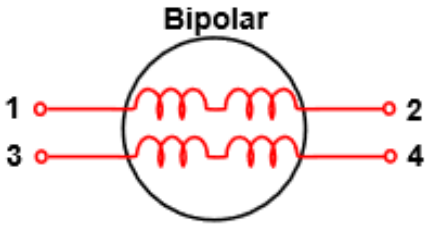
2.1.2 Unipolar And Bipolar Stepper Motors

Another classification of stepper motors, is depending on the way the windings are configurated. Even though, nowadays, almost every stepper is both. Meaning that unipolar and bipolar, are rather modes in which the stepper motor can be driven. Exception being, stepper motors which have only four wires coming out of them, corresponding to bipolar stepper motors.

The pictures below represent the configuration of the windings in both unipolar and bipolar stepper motors.



(a) Bipolar



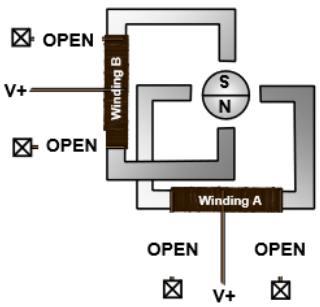
(b) Unipolar

Figure 2.12: Something

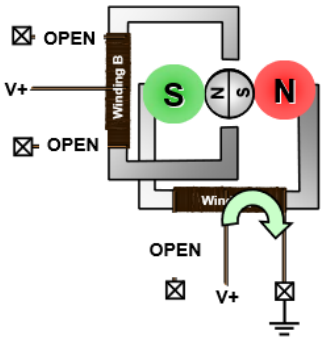
Unipolar stepper motors allow current flow in only one direction through the winding, so a center wire has been added, that provides a voltage and determines the stator poles do either be of north or south polarity. Unipolar stepper motors are widely used in applications that require high torque at high speed.

Bipolar stepper motors, allow current to flow in both directions through the windings, so the need of a center wire to provide a voltage disappears. However, the circuit needed to drive a bipolar stepper motor is more complicated. Bipolar stepper motors driving boards exist to make it easier for a person to program the motor. Bipolar stepper motors are used in applications that require high torque at low speeds.

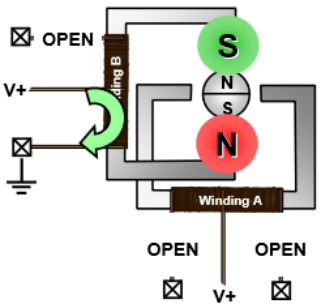
The pictures below are intended to better explain the operating method of both types of stepper motors.



(a) 1st Step



(b) 2nd Step

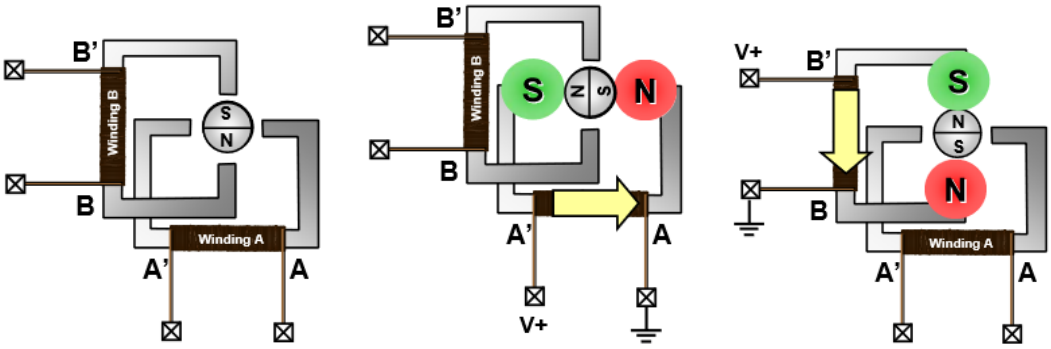


(c) 3rd Step

(a) 4th Step **(b)** 5th Step

Figure 2.13: Spinning Steps

As seen in the pictures above, each step only uses half of the winding, determining the polarity of the stator poles. The voltage is always applied at the same wire two wires.



(a) 1st Step **(b)** 2nd Step **(c)** 3rd Step

(a) 4th Step **(b)** 5th Step

Figure 2.14: Spinning Steps

Note that the bipolar configuration allows the current to flow in both directions, but the voltage and ground continuously switch positions. This makes bipolar stepper motors a bit more complicated to drive, but as previously stated, motor driver boards simplify the task.

2.2 Wheels

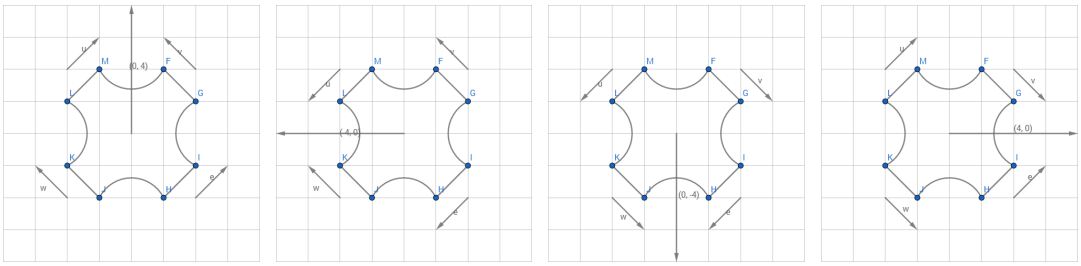
We wanted the robot to be capable of moving in eight directions from every position. One solution for this would have been for the robot to turn every time it needed to change direction, but since our goal was to test and time our path finding algorithm, this was highly impractical. One of our group members had worked with small wheel-based robots before, and remembered omni-wheels. Omni-wheels

(a) Standard Wheel **(b)** Omni-Wheel

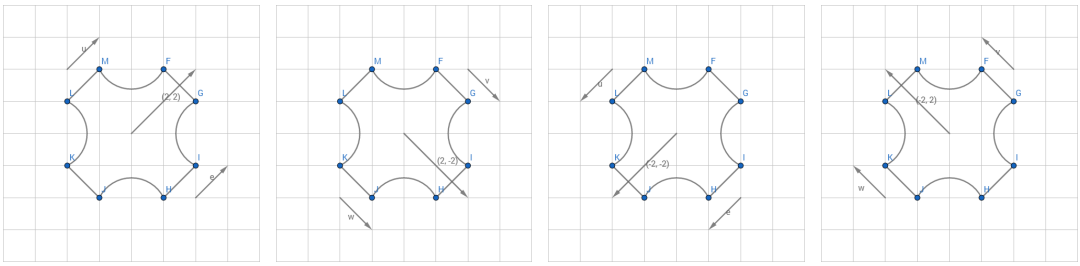
Figure 2.15: different wheels

are wheels, whose contact area consists of smaller wheels. Those smaller wheels are able to spin freely, with very little force needed. If mounted in two pairs, with the axes crossing at a 90 degree angle, one pair controls all forces along the x-axis, and the other one along the y-axis.

find degree symbol



(a) North **(b)** East **(c)** South **(d)** West



(e) North-East **(f)** South-East **(g)** South-West **(h)** North-West

Figure 2.16: forces from multiple wheels added together

2.3 Direction Control

To control the direction of the robot, we had to control which wheels turn how many degrees.

One option for this would have been to control each motor individually, for this we would have needed four individually controllable stepper motors, and four times four control outputs from our microcontroller. This solution also meant precise timing between the four motors was needed, in order to not generate any rotation.

We decided to look for a solution using less output pins, and without the danger of rotation.

By analysing the requirements for our directional control, we figured out that we only need 8 configurations of motor pairs.

Direction	Pair A	Pair B
North	forward	forward
East	forward	backward
South	backward	backward
West	backward	forward
North-East	forward	off
South-East	off	backward
South-West	backward	off
North-West	off	forward

Direction	Aflip	Aon	Bflip	Bon
North	0	1	0	1
East	0	1	1	1
South	1	1	1	1
West	1	1	0	1
North-East	0	1	1	0
South-East	1	0	0	1
South-West	1	1	1	0
North-West	1	0	1	1

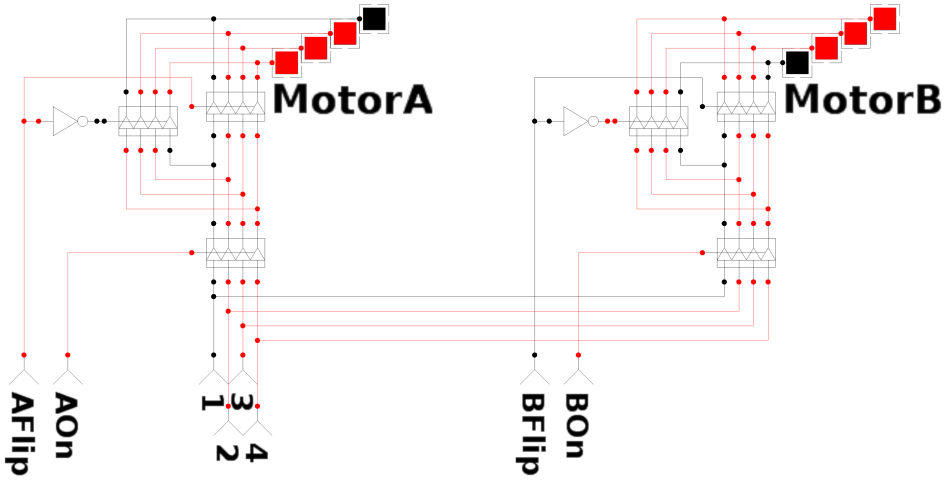


Figure 2.17: motor control circuit

We decided to use 6 tri-state buffers as shown in figure 2.17.

Chapter 3

Maphandling

Pathfinding is generally the process of finding a path from a given starting point ('A') to a given destination ('B'), on a given map.

3.1 Section

Like with many others, is the first step in Dijkstra's algorithm to reduce the map to the necessary minimum. After this reduction, the map only consists of *nodes* and *edges*. An edge connects two nodes together and has one integer *travel cost*. In this integer is stored how much it costs to traverse along that edge, measured in the metric that should get optimized (in our case distance).

Chapter 4

Scan

Pathfinding is generally the process of finding a path from a given starting point ('A') to a given destination ('B'), on a given map.

4.1 Section

Like with many others, is the first step in Dijkstra's algorithm to reduce the map to the necessary minimum. After this reduction, the map only consists of *nodes* and *edges*. An edge connects two nodes together and has one integer *travel cost*. In this integer is stored how much it costs to traverse along that edge, measured in the metric that should get optimized (in our case distance).

Chapter 5

Pathfinding

Pathfinding is generally the process of finding a path from a given starting point ('A') to a given destination ('B'), on a given map.

There are different approaches to find the best path, and different qualifiers for 'best'.

We chose to define 'best' as 'shortest time', because in a real world application time is the most crucial resource in rescue. In other applications 'best' could also mean shortest distance, least expensive (toll roads), most convenient or any number of other qualifiers.

Since our robot has approximately equal movement speed in all used directions, the shortest distance path will be very close to the shortest time path. Since our robot only operates on a very limited map, the shortest distance was also easiest to evaluate.

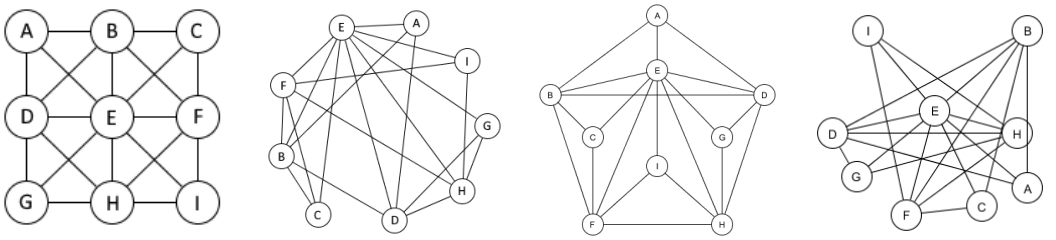
We chose to start implementing Dijkstra's shortest path algorithm, since it is fairly simple to understand and can be used as a baseline for better, more complicated algorithms, like A*.

5.1 Dijkstra

Like with many others, is the first step in Dijkstra's algorithm to reduce the map to the necessary minimum. After this reduction, the map only consists of *nodes* and *edges*. An edge connects two nodes together and has one integer *travel cost*. In this integer is stored how much it costs to traverse along that edge, measured in the metric that should get optimized (in our case distance).

A node has a *name*, an integer *travel cost* and a reference to another node *parent*. The name is used as an identifier, travel cost sums the travelling costs of all edges along the current shortest path and parent refers to what node is the previous in that path.

[Pound2017]



(a) grid graph **(b)** circular graph **(c)** tree graph **(d)** unstructured graph

Figure 5.1: different representations of the same graph

Dijkstra's approach now is to look at the facts we already know: The shortest path from A to A and the costs to get to any of A's neighbours.

Those values then get added to and stored inside the neighbours' travel cost with A as the parent. Now the starting node gets marked as done and the node with the lowest travel cost gets selected as the starting point for the next round of calculations

do we want this next paragraph?

Since the only known travel cost before runtime is the starting node (being 0), this is always where calculations start.

This algorithm can be implemented recursively, with the starting point being the node with the lowest value and the breaking condition of the starting pint being equal to the destination.

It can also be implemented to loop a given amount of times, iterating through all existing nodes.

5.2 Pathfinding on a grid

Pathfinding on a grid is slightly different to pathfinding on a regular map, because all nodes tend to have the same amount of neighbours, and all edges have the same or similar costs.

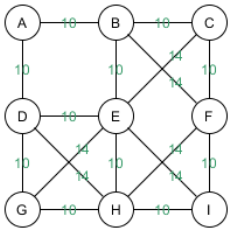


Figure 5.2: 3x3 grid with edge costs

5.3 Our implementation

does this belong into sec:map-handle?

For our grid we chose to allow vertical, horizontal and diagonal movement, giving us 8 possible directions to move in from every node. We decided to store those eight directions in one single byte, with the least significant nibble (LSN) corresponding to the four main directions (N,E,S,W), and the MSN corresponding to NE, SE, SW and NW.

N	E	S	W	NE	SE	SW	NW	byte
0	0	0	0	0	0	0	1	0x01
0	0	0	0	0	0	1	0	0x02
0	0	0	0	0	0	1	1	0x03
0	0	0	0	1	1	1	0	0x
1	0	1	0	1	0	1	0	0x
1	0	1	0	1	0	1	0	0x
1	0	1	0	1	0	1	0	0x
1	0	1	0	1	0	1	0	0x

[Madsen2010], **[Oetiker2010]** and **[Mittelbach2005]**.

fix citation stuff

Chapter 6

Conclusion

In case you have questions, comments, suggestions or have found a bug, please do not hesitate to contact me. You can find my contact details below.

Jesper Kjær Nielsen
jkn@es.aau.dk
http://kom.aau.dk/~jkn
Fredrik Bajers Vej 7
9220 Aalborg Ø

Appendix A

Appendix A name

Here is the first appendix