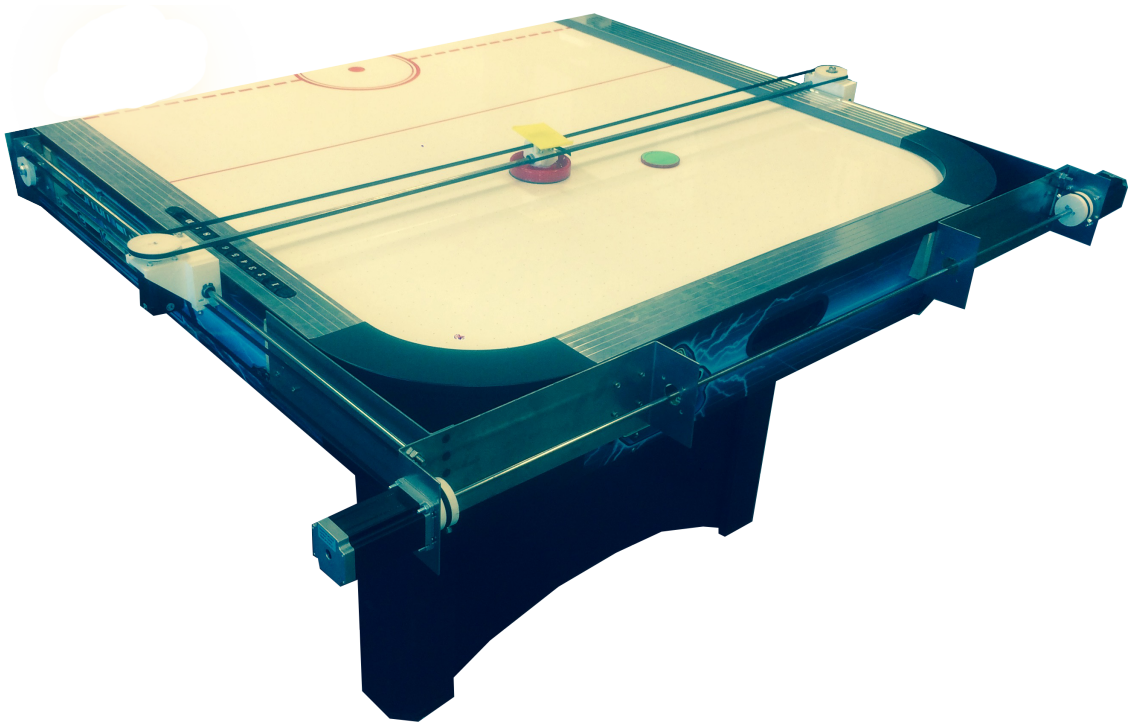


Autonomous Air Hockey

SSYX02-16-11

Bachelor thesis at the Department of Signals and Systems

Martin Ansgar, Hanna Berggren, Pontus Borg,
Rasmus Damberg, Mattias Gudjonsson & Lukas Mared

Department of Signals and Systems
Chalmers University of Technology
Gothenburg, Sweden 2016

Abstract

This reports describes the development and construction of a autonomous mechatronic air hockey robot. The robot shall be able to play well against both humans and other robots while following the regular rules of the game.

The device constructed consists of a camera above the table for puck detection, a system of stepper motors, linear bearings and metal rods for movement and several Arduino microcontrollers for computations. Everything is held together by 3D-printed plastic mounts and attached to the table by custom water cut steel mounts. The microcontrollers are programmed using C.

The result is a device that can detect the puck, calcuates its speed and predicts trajectory and moves the paddle accordingly to intercept. The robot can defend the goal against pucks incoming at a speed of up to 7 m/s and can shoot a stationary puck back at a speed of 1.7 m/s. The robot can not perform attack on its own, it can only defend.

Sammanfattning

Den här rapporten beskriver utvecklandet och konstruktionen av en autonom mekatronisk robot vars uppgift är att spela air-hockey. Kraven som ställs på roboten är att den ska klara av att spela väl mot både andra robotar och människor, utan att bryta mot spelets regler.

Konstruktionen består av en kamera som är placerad ovanför bordet för att upptäcka pucken, ett system av stegmotorer, linjärkullager och stålstänger för rörelse samt ett antal Arduinos för beräkningar. Konstruktionen hålls ihop av 3D-printade plastfästen och sitter fast i bordet med hjälp av vattenskurna stålfästen. Mikrokontrollerna programmeras med språket C.

Resultatet är en konstruktion som kan hitta en puck, beräkna dess hastighet och förutser dess bana och förflyttar klubban därefter för att returnera pucken. Roboten klarar av att försvara målet mot inkommande puckar i hastigheter upp till 7 m/s och kan skjuta tillbaka en stillastående puck med en hastighet upp till 1.7 m/s. Roboten kan inte genomföra attacker på egen hand, utan endast försvara sig.

Acknowledgement

We would like to thank the following persons for making this project possible,

Hakan Köroğlu, for being our supervisor during the project, providing feedback and help with problems we faced.

Jonas Fredriksson, for providing hardware and supporting with knowledge on microcontrollers and automation.

The Department of Signals and Systems at Chalmers University of Technology for providing the resources enabling us to finish the project.

Calle Carlsson, for the feedback on our report.

Göran Stigler and Jan Bragée for the help in the Prototype laboratory.

The other air-hockey bachelor thesis group SSYX-02-16-82, for motivating us to build a better robot than theirs.

Contents

1

Introduction

Ever since the first machine was created, people have attempted to construct machines or robots to work for them. This is not only because it is cheaper to do so, but also because machines can sometimes perform certain tasks better than humans can. Today, robots have already replaced humans in many lines of work.

There are also many games or similar activities where the robot can use its enormous computing power and clever algorithms to defeat a human player. The idea was made famous by IBM in 1997 when their computer Deep Blue won against the reigning champion Garry Kasparov in a game of chess [1]. The amount of games in which this can happen is increasing every day, with no sign of change. The game of air hockey was invented in the 70s in USA and quickly became very popular in arcade halls around the world. The game consists of a low-friction rectangular surface, seen in Figure 1.1, created by many tiny holes through which air is blowing. A small round disc, called the puck, is then placed on top of it. Thanks to the air, the puck can glide almost frictionless on the surface. On both short sides of the surface, there are goals. In order to win the game, the player must use a round paddle to shoot the puck in the opponent's goal, while defending his or her own goal. The game hence requires two people to play.

Figure 1.1: A professional match on a standard air hockey table [2].

In this project, an autonomous air hockey-playing robot is constructed. At a first glance, this task may seem fairly easy. With only two dimensions to work with in motion control and a relatively simple objective, i.e. to score goals against the opponent and to protect one's own goal. What makes it difficult is that the game is very fast-paced, resulting in that the robot must be able to react and move the paddle to the correct position within a fraction of a second, where only the slightest error in positioning may result in a lost game. The construction of the device is a challenge for the designers because the robot has to be built not only so that it matches the requirements for speed and accuracy, but also program it to react correctly and to the many different situations that can occur.

All of the members of this project have studied at Chalmers University of Technology for more than two years and have taken courses that are related to similar engineering problems. Nevertheless most of the courses just focus on isolated problems that are restricted in scope. On the other hand, this is a more elaborate design and implementation project which includes several problems from different areas. This project is therefore considered as a good opportunity to put to use what the group members have learned so far to acquire new knowledge and skills as needed for the realization of the required tasks.

1.1 Purpose

The purpose of this project is to develop an autonomous mechatronic device which can play air hockey. It should be able to locate the puck on the table and use this information to move the paddle and defend the goal, while also attempting to score goals of its own.

1.2 Problem Description

The construction of such a robot has already been attempted in a bachelor's project in 2015 [3]. In that project, the robot could only defend its own goal, lacking the ability to be offensive and actually try to score goals itself. In this project, the knowledge gathered from that project is used to create a more solid solution. Through this, the goal is to further improve the playing performance of the robot, mostly by modifying the areas that were previously insufficient.

For this to be possible, the robot needs a specific hardware setup. It must be able to move in all planar directions at high speed. It is also necessary that it can change its direction almost instantly when the puck's trajectory or speed changes. All this has to be achieved together with the ability to move to a given location with a desired accuracy.

The vision system must be able to detect the puck with good accuracy at all times, regardless of where the puck is on the playing area. It needs a sufficient frame rate to be able to determine the puck's velocity and trajectory even at very high speeds. It also has to be able to deliver this information repeatedly to the other involved components.

The software must be able to receive information from the vision system. After that, the optimal playing strategy has to be determined for each incoming puck. Finally, the correct signals must be sent in order to move the paddle correctly, while preparing for the next move.

The robot should play well against not only a human, but also other robots. Since there are two groups assigned with this task, a competition is planned between the two groups. The purpose of this competition is to further motivate the groups and to create something new and interesting for Chalmers University of Technology to put on display.

1.3 Requirements and Goals

For the project to be considered complete, the following requirements should all be met. While there is not any need to reach any of the goals listed, the robot is designed with them in mind.

1.3.1 Requirements

The requirements are based on the results that the previous project achieved. The different numbers have all been increased slightly, since this project is considered as an improvement of that project.

- (R 1) The robot must be able to reach the puck with the paddle at any point on its own half of the table.
- (R 2) The construction must not reach over to the other group's half of the table.
- (R 3) The robot should be able to hit a majority of all pucks approaching at a speed of 2 m/s.
- (R 4) The robot should be able to defend the goal from a majority of all pucks approaching at 6 m/s.
- (R 5) Given optimal circumstances, the robot must be able to shoot the puck with a speed of at least 2 m/s.

With these requirements in mind, the defense would be strong but the offensive capabilities are of a lower quality, making it so that the robot rarely will be able to score goals of its own.

1.3.2 Goals

The goals are based on average game play between humans. While it is possible to hit the puck at a speed of almost 30 m/s [4], testing of puck speed for this project revealed that the average puck speed in regular play is much lower than this.

- (G 1) The robot should be able to hit a majority of all pucks approaching at a speed of 4 m/s.
- (G 2) The robot should be able to defend the goal from a majority of all pucks approaching at a speed of 9 m/s.
- (G 3) Given optimal circumstances, the robot must be able to shoot the puck with a speed of at least 4 m/s.
- (G 4) The system should be easy to start, without any configuration or calibration needed.

A robot that can meet all these goals would be a tough opponent for any average player not very familiar with the game. Even experienced players would have a hard time to score goals, since the defensive play is prioritized during the construction.

1.4 Boundaries

Due to limited time and financial budget, the following boundaries have been introduced to constrain the depth and scope of the project:

- The project has a budget of 5000 kr for anything needed.
- The prototype is designed specifically for the dimensions of the table, and therefore no additional effort is put to facilitate the possibility for mass production.
- Since the table is meant to be stationary, the prototype is not designed to be mobile.
- The puck localization device is constructed to work under optimal conditions, with no consideration for outside factors such as low light or other distortions.
- Human interaction is required to retrieve the puck after scoring a goal and to put it back to the table.
- The color of the puck need to differ from the paddle in order for the vision system to detect these objects.
- Since there are two groups that are using the same table to mount their robots on, the length of the prototype may not exceed half the length of the table.
- To power the microcontrollers and the motors, external power sources are to be used to feed the correct voltage.

2

System Description

The system consists of a combination of hardware and software components. An overview of how they are connected and interact with each other is presented below.

The paddle used to play is able to move along a steel rod through linear bearings, held together by 3D-printed plastic mounts. The movement in the sideways direction is generated by a small motor attached to one of the mounts and is transferred using cogwheels and rubber belts, seen in Figure 2.1. All this is then able to move forward and backwards through a similar system of steel rods and rubber belts. Since more weight needs to be moved in this direction, two more powerful motors are used. Between these motors, a metal shaft is mounted to synchronize the movement.

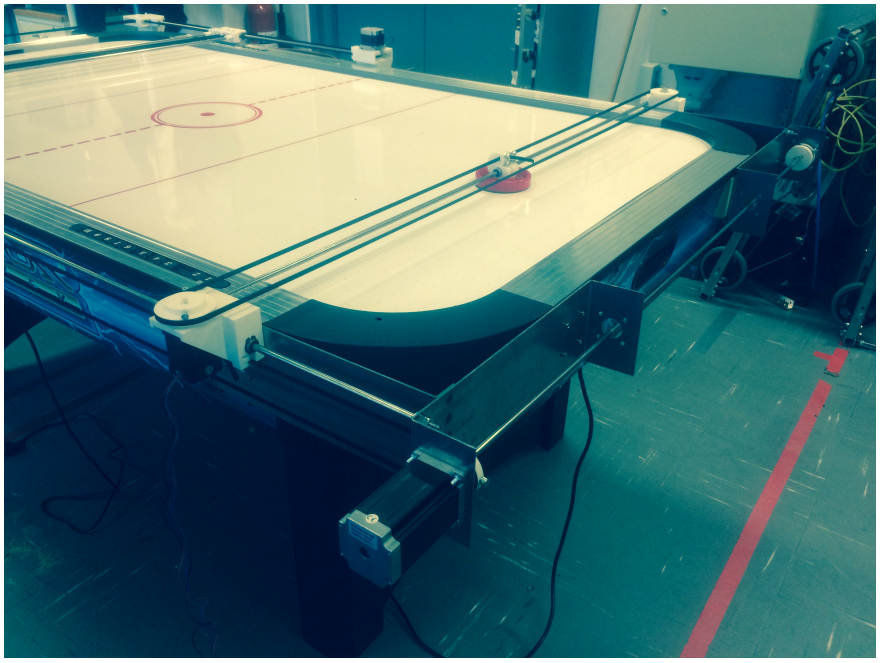


Figure 2.1: Overview of the mechanical system.

A camera is mounted on a stand directly above the table at a fixed height. The camera is connected to an Arduino microcontroller, which is responsible for the necessary computations. This Arduino is acting like a master unit, which controls two other Arduino units that control the forward movement.

The camera detects where the puck is, in which direction it is going and what speed it has. Based on this, the microcontroller determines the optimal strategy for returning the puck. The paddle is then moved according to this by the motors, which receive their power from the external power supplies. The master Arduino provides the right information to the two slaves, which control the stepper motors. An overview of the system can be seen in Figure 2.2.

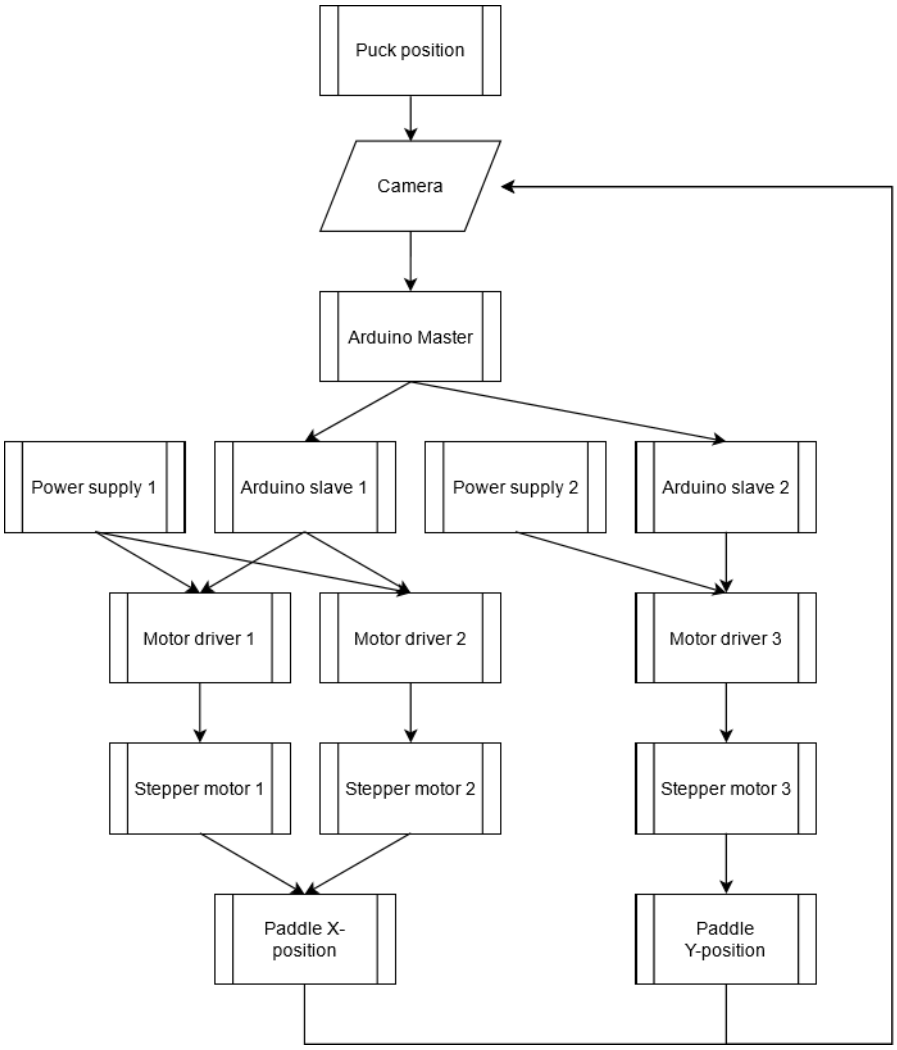


Figure 2.2: Overall view of the system.

3

Mechanical Construction and Hardware

The hardware parts of the construction are chosen and designed with respect to certain demands. The manufacturing process and the motivations about why each component is suitable for the specific task is presented.

To save time, the construction is based on the design from the previous project [3]. That project's main issue was to shoot the puck with a sufficient speed. To achieve higher speeds while attacking, a more powerful motor setup is used, while also reducing the construction's weight through various optimizations in the mechanical design.

3.1 Table

This main focus of this project is on the mechanical device and the programming. Therefore the air hockey table used is bought from a store instead of being built. The table is used by both groups, which further increases the need for a very sturdy and high quality table. The table needs to withstand the forces from both robots moving around simultaneously, while not shaking and disturbing the mechanical systems.

The table used is the Buffalo Terminator [5]. It has a playing area of 198x106 *cm*, a height of 82 *cm* and is quite heavy. With these characteristics it is suitable for both robots to move around without any shaking effect. The sides of the table are made out of wood, which makes it easy to attach the mechanical system to it.

3.2 Camera and Camera Stand

The camera used is the Pixycam CMUcam5 [6]. It has several significant advantages over its competitors, both from a technological and an economical perspective. The device was inherited from a previous project, so it did not have to be considered in the budget. It also comes with an automatic color detection system, thus eliminating the need to write a program for this purpose and thereby saving time. In order to get maximum field of view, the camera is equipped with a fish-eye lens. It also has a frame rate of 50 FPS while not sacrificing too much resolution, 320x200 $pixels$, to achieve it. With this frame rate and a puck moving at the goal speed of 9 m/s, the puck moves 18 cm between two frames. Considering the total length of the table, the frame rate is sufficient for the project, giving the robot enough time to react. The camera is used not only to see the puck, but also to determine where the paddle is.

The camera's aspect ratio does not perfectly match the table's aspect ratio. Hence in order to see the entire table, the camera is placed in such a way that it sees slightly further than the playing field on the long sides of the table. While this sacrifices accuracy, it gives the robot more time to react to incoming pucks. Since some of the pixels are outside of the table surface only pixels 315x190 are used to calculate the size of the average pixel.

To calculate the average size of a pixel on the table the following equations were used:

$$Pixel\ LengthX = \frac{106cm}{190pixels} = 0.56\ cm/pixel \tag{3.1}$$

$$Pixel\ LengthY = \frac{198cm}{315pixels} = 0.63\ cm/pixel \tag{3.2}$$

The reason that the average length of the pixels are different in the x- and y-directions is because of the camera's fish-eye lens.

The camera is mounted on a camera stand above the table. The stand is shared with the other group, with both cameras mounted 130 cm above the playing area, roughly 2 m from the ground. The camera stand is constructed separately from the table so that the camera image is not disturbed by the robot's movement.

3.3 Motors and Drivers

Stepper motors are used as actuators of the mechanical unit. This is because they are easily controlled, with very precise movements. Stepper motors also have high torque when operating at low speeds, which is beneficial in this case. The previous project encountered problems with the offensive capability of the robot because the motor responsible the forward motion was not powerful enough. To make sure this was not repeated, some calculations were made to decide which motors to use.

The maximum power needed from the motors was calculated based on the objective to defend the goal from all pucks approaching at a speed of 9 m/s and the maximum distance that the paddle would need to travel. The weight of all the moving parts was estimated and the radius of the cogwheels were chosen. In order to perform the power calculation, the motor torque at the desired maximum velocity needs to be calculated. The formulas that are used in these calculations are provided below.

To calculate the minimum torque the motors need to deliver at the maximum angular velocity the equation for acceleration torque was used:

$$T_{acc} = J_{tot} \cdot \frac{d\omega}{dt} \tag{3.3}$$

where T_{acc} is the minimum torque needed to achieve the goal to defende goal from all pucks moving at 9 m/s and $\frac{d\omega}{dt}$ is the assumed constant angular acceleration of the motors. Where the maximum angular velocity that the paddle needs to be able to reach is given by:

$$\omega = \frac{v_{max}}{r} \tag{3.4}$$

where r is the radius of the cogwheel connected to the motor, and v_{max} is the maximum velocity which can be calculated through

$$v_{max} = 2 \cdot \frac{d}{t} \tag{3.5}$$

where d is the maximum distance that the paddle needs to travel, in this case half of the table's width. The time available to move is defined as t seconds. It is the time that it takes for the puck to reach the paddle after two frames have been analyzed by the camera, while the puck is traveling at 9 m/s. This formula is derived by assuming constant acceleration of the paddle. Thereafter the total moment of inertia J_{tot} needs to be calculated, which is calculated through the following equation:

$$J_{tot} = J_{motor} \cdot n_{motors} + J_{load} + J_{cogwheel} + J_{belt} \tag{3.6}$$

Where J_{motor} is the motor's internal moment of inertia which is found in the motor specification attached in appendix, and n_{motor} is the amount of motors used. To calculate J_{load} the formula for a point mass is used:

$$J_{load} = m_{load} \cdot r^2 \tag{3.7}$$

In this expression, m_{load} is the mass that needs to be moved, while r represents the radius of the cogwheel. The moment of inertia of the belt is calculated in a similar fashion:

$$J_{belt} = m_{belt} \cdot r^2 \tag{3.8}$$

For the cogwheels, the moment of inertia is calculated as if they were solid cylinders, as follows:

$$J_{cogwheel} = \frac{1}{2} \cdot n_{cogwheels} \cdot m_{cogwheel} \cdot r^2 \tag{3.9}$$

where $m_{cogwheel}$ is the mass of a single cogwheel and $n_{cogwheels}$ is the number of cogwheels. These equations are then computed for the movement in both the forward and sideways direction. To get a more even distribution of force, two motors are used for the forward motion, one on each side. Evaluating these equations resulted in the minimum torque needed for the sideways motion to approximately 0.35 Nm, and for the forward motion approximately 1.04 Nm per motor.

The motor used for the sideways motion is a Nema17 17HS24-2104S stepper motor, combined with a ST-7128 driver. This motor delivers approximately 0.6 Nm, which is enough to meet the goals. The motors used for the forward motion are two Nema23 23HS41-1804S stepper motors with combined with two M542T stepper motor drivers. These motors deliver approximately 2.0 Nm each, also sufficient to achieve the goals. Both types of motors can be seen in Figure 3.1.

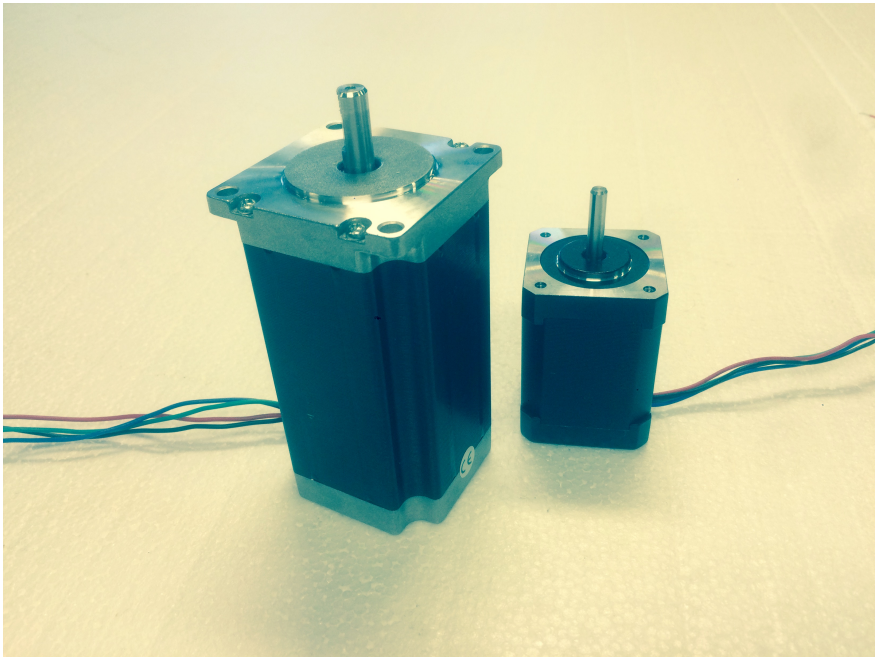


Figure 3.1: The Nema23 and Nema17 stepper motors.

The motors are controlled by using half steps. This means that each step results in a 0.9° turn, which equals to 400 *steps/revolution*. By knowing the radius of the cogwheels, an exact distance can be calculated per step. Through this, it is calculated that the robot moves 0.48 mm per step in the y-axis, and 0.41 mm

per step in the x-axis. In Figure 3.2, the distance moved with regards to steps is visualized.

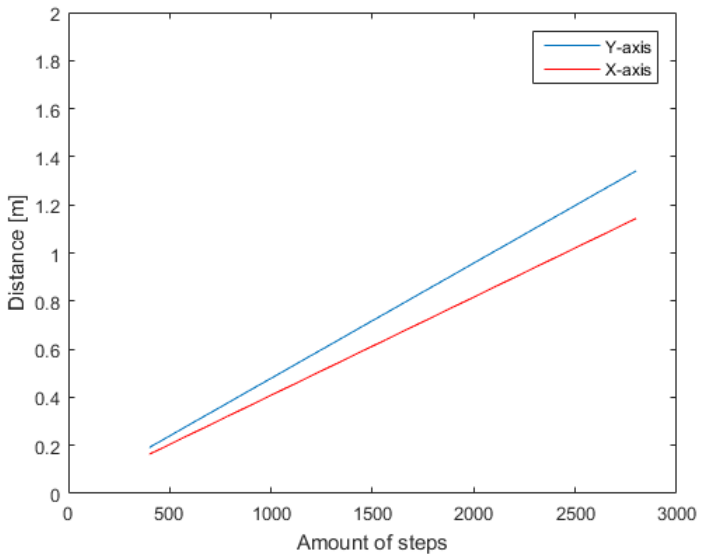


Figure 3.2: Distance moved in x-, and y-axis, with regards to the amount of steps.

In order to function properly, the drivers need to be fed with 24 V DC for the small motor and 48 V DC for the large ones. External power supplies connected to the power grid are used to handle this.

3.4 Cogwheels and Belts

In order to transform the rotational force of the motors to horizontal force to move the paddle, a combination of cogwheels and rubber belts are used. The cogwheels are custom-designed based on the belt's profile and printed in a 3D-printer. Their sizes were chosen to match the torque of the motors, as the cogwheel diameter affects the speed and strength of the horizontal movement.

The belts used are the GT2 Timing Belts commonly used in 3D printers. The belts are made of rubber with a threaded core, which makes them not stretchable. They also have a specific profile, seen in Figure 3.3, designed to prevent slips. The belts are chosen as small as 6 mm wide, since wider ones would only increase friction and costs.

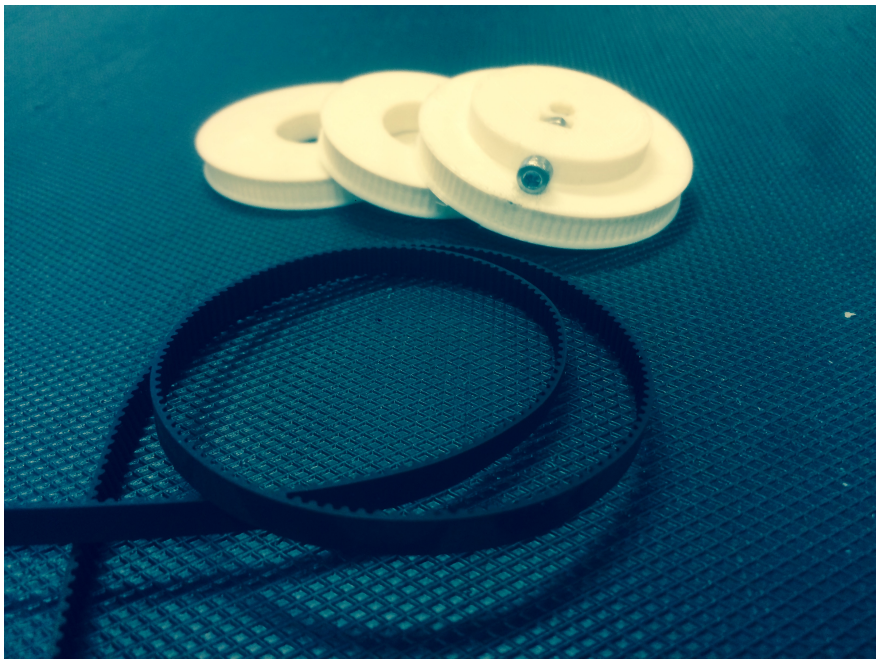


Figure 3.3: The custom designed 3D-printed cogwheels and the GT2 Timing Belt.

3.5 Microcontroller Setup

The Arduino platform is used to handle all computations and data processing. Since a single Arduino unit has a limited computation power, three units are used for this project. An Arduino Mega is used as the master unit, which receives information from the camera. Two Arduino Uno units are used as slave devices. These are responsible for movement in x- and y-direction respectively. All of them are connected to a 5 V power source and operates with a clock speed of 16 MHz. When programmed, the units are connected to a computer via an USB-cable and when a game of air hockey takes place the master Arduino executes a program, while the two slaves executes another program.

3.6 Mounts

Several custom-made mounts are designed to hold the mechanical construction. Stationary steel mounts are designed to anchor the entire construction to the table, while plastic 3D-printed mounts are used for the moving parts, as this reduces the total weight.

3.6.1 Steel

The sheet steel parts hold the rods, the two large motors and the cogwheels needed for the forward motion. A solid construction is needed to avoid movements, as the cogwheels and belts are very sensitive. Steel is stiff and has a high strength which makes it suitable for this purpose.

The steel plates are symmetric to make the manufacturing easier. These are cut using water jets for increased precision and detail accuracy. Extra support for the shaft between the motors was designed in case the shaft would not be stiff enough. This is currently unused, as this turned out not to be the case.

3.6.2 Plastic

All details with more a complex geometry have been 3D-printed in plastic. The 3D printing process makes it easy to manufacture complex designs with high accuracy, which is needed for the smaller moving parts. While not as robust as steel, the plastic is still able to withstand the forces created by the robot. In total, three different mounts are printed. Two are used for the sliding motion and are shown in Figure 3.4. The last mount is used to hold the paddle in place and is shown in Figure 3.5

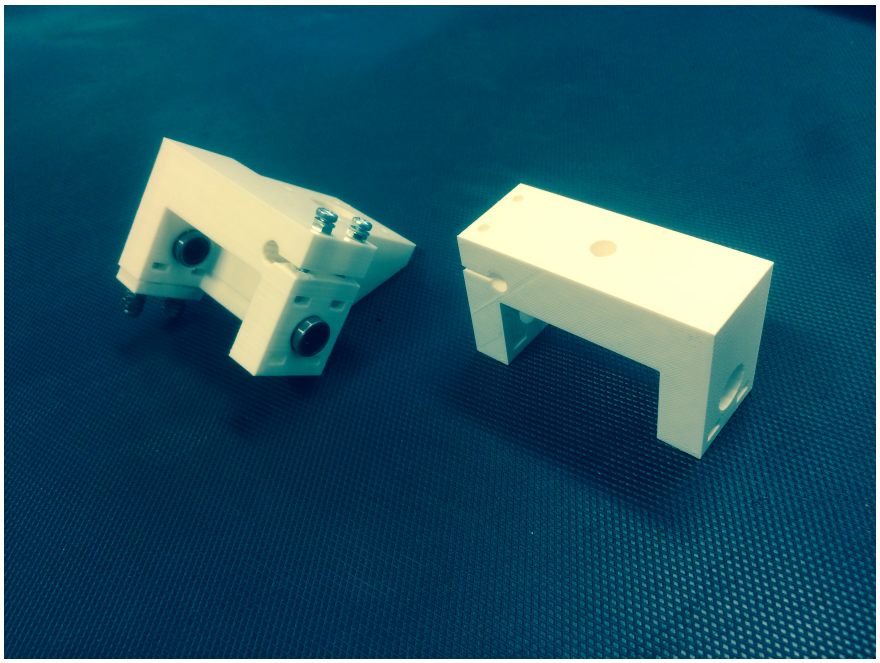


Figure 3.4: The custom-designed 3D-printed mounts for the forward sliding motion.

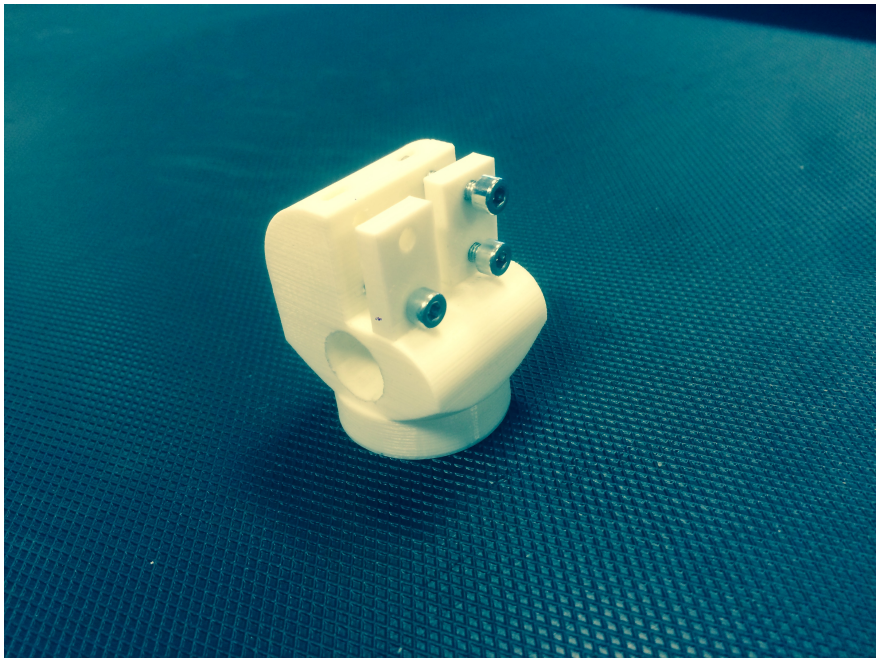


Figure 3.5: The mount that holds the modified paddle, ready for bearings to be mounted on it.

3.7 Steel Rods

The mechanical system consists of a total of four steel rods that are used for the construction's movement. A rod is placed on each side of the table for the paddle sled to move on. A third longer rod is then mounted across the table, which the paddle itself slides along. The fourth rod is placed between the motors as a shaft to synchronize the forward motion. The rods have a diameter of 8 mm.

When choosing what material to use, several different materials were considered. The rods have to be stiff and durable, while still weighing as little as possible. Carbon fiber rods were considered, but they are not durable enough for long term use. Based on similar projects, steel was chosen instead. [7]

3.8 Bearings

It is very important that there is as low friction as possible when the robot is moving the paddle around to decrease the force needed. To achieve this, different bearings are used. Traditional radial bearings are used for the cogwheels rotation and linear bearings are used for the parts' mounts along the rods. The linear bearings were chosen over other alternatives, such as linear guides, mainly due to the much lower cost and the ability to reuse the ones from last year.

Both the radial bearings, model 608 ZZ, and the linear bearings, model LM8UU, have an inner diameter of 8 mm .

4

Software Design

The software design decides how the robot should react to all the different things that might occur during a game. All of the code is written in the programming language C, adapted for Arduino. The system process for a general air hockey play is brief explained and shown in Figure 4.1.

The system starts by gathering information from the camera about the puck's and paddle's position twice. This information is transferred to the master Arduino, which adjusts the coordinates that are distorted due to the fish-eye lens. Thereafter when two positions are known it calculates the velocity and direction which is used to analyze if the puck is moving towards or away from the goal. If the direction is away from the goal the system updates the paddle position through feedback from the camera. However, if the puck is moving towards the goal the system decides for a strategy based on the velocity.

When a strategy has been chosen the system calculates the impact position, i.e where the paddle should be placed for the best possible puck interception. This information is consequently transferred to the two slave devices that controls the forward and sideways motion, hereafter referred to as movement in the x- and y-axis. The slave devices then starts an iterative process of moving the paddle to the correct position. The paddle stays in this position until the puck has been hit, resulting in a changed puck direction. When this direction changes, the system uses the camera for feedback control to return the paddle to the starting position in the middle. This entire computation takes between 24 to 45 ms, depending mostly on the amount of new position information needed to calculate the velocity, which is provided every 20 ms.

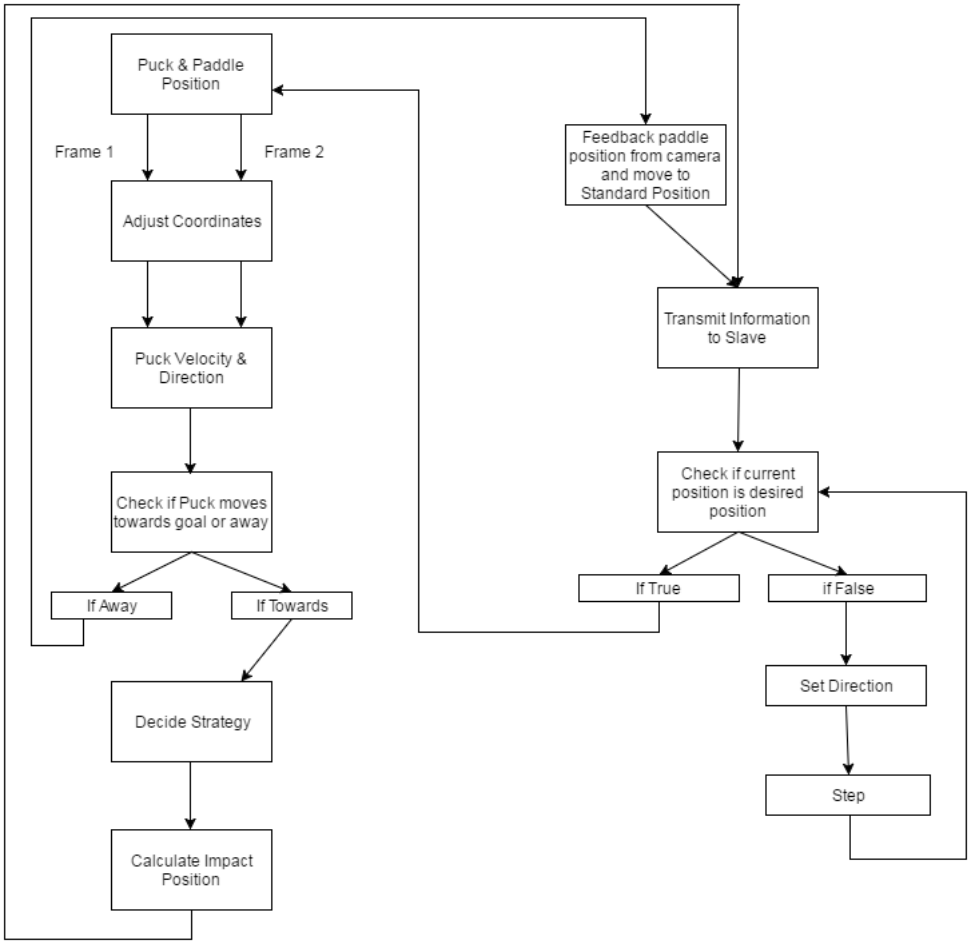


Figure 4.1: Flowchart of the software

4.1 Strategy

For the robot to play well, it must have a strategy concerning how to act for all the different cases that might occur during a game. The strategy is divided into two main cases, one being offensive play and the other being defensive. The robot decides its playing mode based on a number of parameters, the most important one of which is the speed of the incoming puck. When the speed of the puck is high, the robot is only focusing on defending the goal, while a puck with low speed results in the robot actively trying to hit it for an attack.

Between shots, the paddle is placed in a standard position right in front of the goal. This position was chosen after observing several games of professional air hockey, where this spot is used as the standard defense position [8]. After every shot, the paddle is returned to this spot to be ready for the next attack.

4.1.1 Standard Play

When the puck is not moving very fast or very slow towards the robot, it uses the standard play. This strategy is a combination of the defensive and offensive strategies, where the robot first positions itself to defend the puck and then move forward to hit it. Defensive action is still prioritized, as this makes the robot harder to beat.

4.1.2 Defensive Play

When the incoming puck's speed is high enough, the robot no longer tries to make a counterattack. Instead it only moves sideways to defend the goal. To block any shot heading for the goal, the paddle only needs to be moved a few centimeters. This allows for blocking of pucks incoming at a higher speed, further improving the playing performance.

To achieve this, the robot positions the paddle in the standard position and only intercepts pucks heading directly for the goal. It does not attempt to move forward at all.

4.1.3 Offensive Play

If the puck is approaching with little to no speed, the robot then attempts a full offensive move. This means that it attempts to hit the puck as fast as possible at an angle, which would make the puck very difficult to block for the opponent. The robot calculates where the puck enters its half of the table, positions itself accordingly and then charges the puck when it is in reach.

4.2 Speed, Trajectory and Impact Calculations

For the robot to be able to play properly, it must be able to predict where the puck is going and when it is going to be there. For this reason, the puck's speed and trajectory need to be calculated. This data is then used to calculate an impact position where the robot intercepts the puck. The impact position differs if the robot engages defense or offensive mode.

4.2.1 Speed

The speed of the puck is estimated by comparing two frames and then calculating how much the coordinates of the puck have changed. The time between frames is not constant for every iteration, so the Arduino measures this using it's internal clock.

$$v_{puck} = \frac{\sqrt{\Delta x^2 + \Delta y^2}}{\Delta t} \tag{4.1}$$

where x and y are the coordinates given by the camera and Δt is the time between frames, measured by the Arduino.

4.2.2 Trajectory

The trajectory is calculated in a similar fashion. Two frames are compared and the difference for both coordinates is analyzed. The result is a variable that represents at which angle compared to the center the puck is traveling in.

$$k = \frac{\Delta y}{\Delta x} \tag{4.2}$$

where k is the slope, i.e the angle compared to the center line of the table.

When the puck hits the wall, it is assumed that no energy is lost and that the angle compared to the wall is the same after the impact as it was before.

4.2.3 Defensive Impact Position

The impact position, x_{impact} and y_{impact}, is needed for the robot to know where it must be to intercept the puck. In the defensive mode this is attained by using the slope in equation (4.2), the puck's position in the x- and y-axis and the paddle's position in the x-axis. . y_{impact} is estimated as follows:

$$y_{impact} = y_{puck} - k \cdot (x_{puck} - x_{paddle}) \tag{4.3}$$

The impact position in the x-axis, x_{impact}, is equal to the current position of the paddle in the x-axis, x_{paddle}, which is given by the camera.

4.2.4 Offensive Impact Position

In offensive mode, the robot also needs to know where to intercept the puck, but it tries to be in movement when doing so instead of just blocking. Therefore, the puck's trajectory, speed and position in the x-direction needs to be measured, as well as knowing the paddle's acceleration and position in the x-direction. Since the system is operating under constant acceleration, the acceleration is always known. To calculate the point of interception, the acceleration and speed needs to be integrated to represent coordinates.

$$\int v_x \, dt + x_{puck} = v_x t_x + x_{puck} \tag{4.4}$$

Where v_x is the velocity of the puck in x, and t_x is time.

$$\iint a_x \, dt + x_{paddle} = \frac{a_x t_x^2}{2} + x_{paddle} \tag{4.5}$$

Where a_x is the acceleration of the paddle in x. The equation is then solved for t_x:

$$\begin{aligned} v_x t_x + (x_{puck} - x_{paddle}) = \frac{a_x^2 t_x}{2} \Rightarrow \frac{a_x^2 t_x}{2} - v_x t_x - \Delta x = 0 \\ \Rightarrow t_x = \frac{v_x}{a_x} \pm \sqrt{\frac{v_x^2}{a_x^2} + 2 a_x \Delta x} \end{aligned} \tag{4.6}$$

In this equation, v_x is always negative since the calculation is computed only when the puck is moving towards the robot, resulting in only one possible solution. This provides the greatest time that it takes to reach the point of interception, i.e. where the system reaches the greatest speed in the x-direction and still intercepts the puck. The point of interception in the y-direction is calculated in the same way as in equation 4.3 where x_{paddle} is the interception point in x-direction.

4.3 Communication Between the Microcontrollers

There are several ways to communicate between Arduinos, where I^2C and serial communication are the most common. The difference between the two interfaces is mainly how they are connected, where the I^2C can interconnect up to 120 other Arduinos with two wires [9], whereas the serial communication is limited to the amount of serial inputs and outputs of the Arduinos [10]. To avoid limitation in the communication, the I^2C interface is used.

4.4 Acceleration Profile

When operating a stepper motor, there is a risk that the motor begins to stall. This usually happens if the acceleration phase is too fast, or if the initial speed is too high. One way to delay the stalling is to use constant acceleration. If this is used, the motor only stalls if the speed becomes too fast, as the torque reduces with greater speed [11]. This can be avoided by experimentally measuring the speed the motors can achieve before stalling, with an attached load to resemble normal circumstances. After conducting research, a suitable acceleration profile was found for this objective. The following formula is used to calculate the time delay between each step for the stepper motor [11]:

$$c_n = c_0(\sqrt{n+1} - \sqrt{n}) \tag{4.7}$$

In this equation, c_n is the time delay that is used in the n:th step, and c_0 is the initial time delay. Due to limitations in the computational power of the microcontrollers, the double square root calculation is not fast enough. Instead, by using a Taylor series approximation, a new formula was derived as follows [11]:

$$c_n = c_{n-1} - \frac{2c_{n-1}}{4n+1} \tag{4.8}$$

This expression can be used efficiently in the microcontroller to determine the step delay, as it has an approximate calculation time of 20 μs. This equation also gives a high versatility of control, since the main factor affecting the acceleration is the initial time delay as can be seen in Figure 4.2.

The acceleration is constant if the initial time delay is constant, as presented in Figure 4.3.

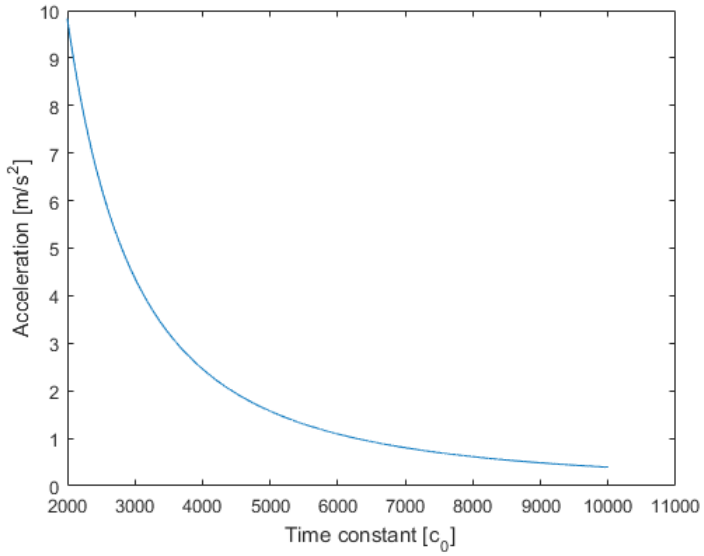


Figure 4.2: Graph illustrating how the acceleration of the system depends on the initial time delay (c_0), with a constant amount of steps.

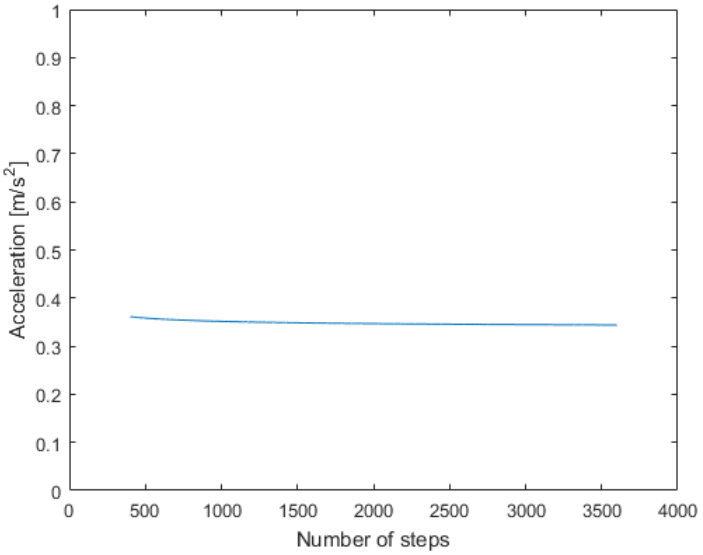


Figure 4.3: Graph illustrating how the acceleration of the system depends on the amount of iterations, with an initial time delay (c_0) of 10000.

4.5 Vision System

To be able to see the entire table, a fish-eye lens is used. The downside of such lens is that the image received is distorted. This distortion is limited in the center, but very notable in the corners. To compensate for this distortion, a mathematical formula that changes the distorted coordinates to more accurately match the actual coordinates is used.

First, a MATLAB application called Camera Calibration [12], [13] was used. This application calculates parameters for how the lens distorts the image, visualized in Figure 4.4. [14].

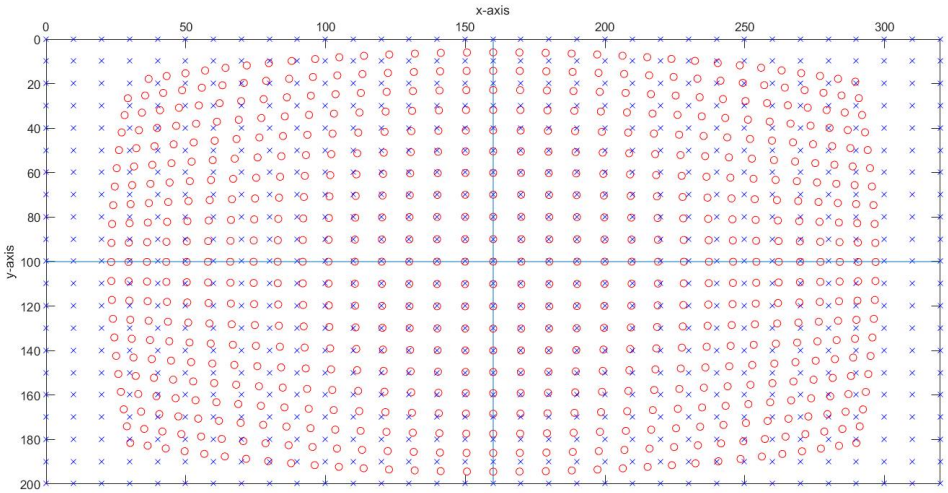


Figure 4.4: Image distortion. Red circles represent what the camera sees and blue crosses represent reality.

The camera distortion parameters are then analyzed by the iterative MATLAB process Undistort Points[15], resulting in a matrix that contains how much each coordinate is affected by distortion, visualized in Figure 4.5 and Figure 4.6.

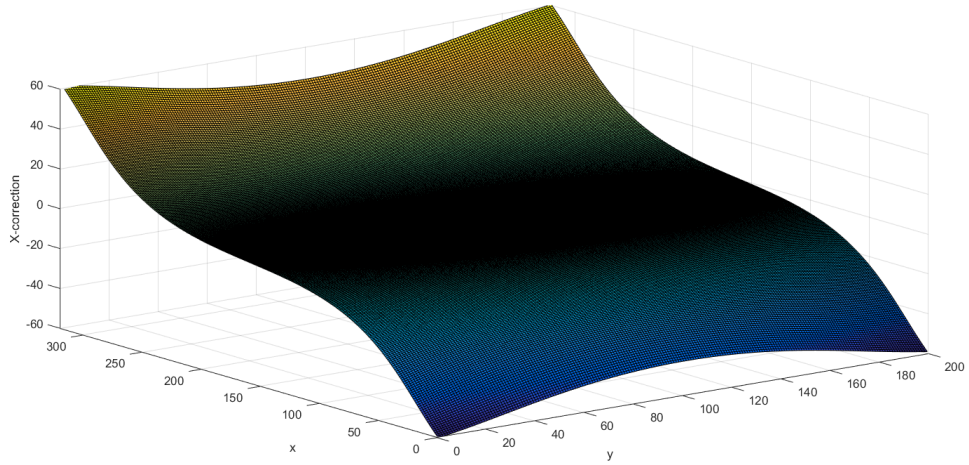


Figure 4.5: Correction in x-direction [pixels]

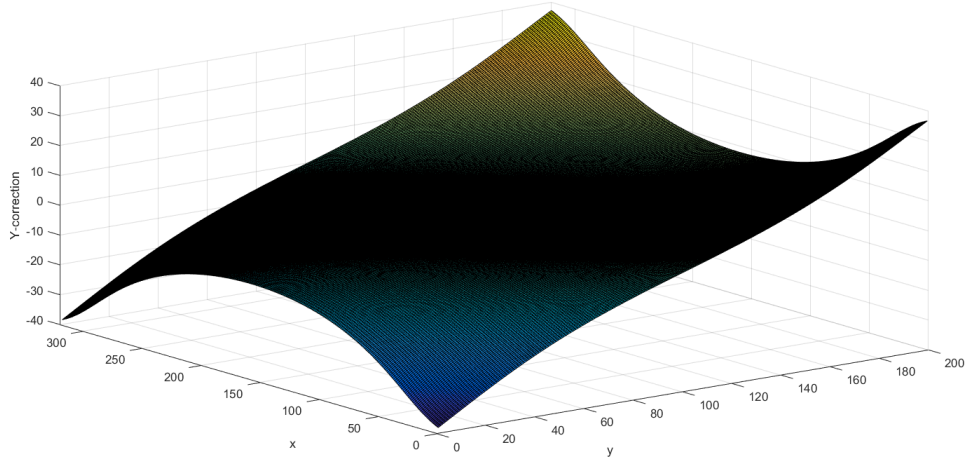


Figure 4.6: Correction in y-direction [pixels]

To efficiently implement this in the microcontroller, the matrices must be approximated by formulas. The MATLAB function Curve Fit[16] is used for this, resulting in the formulas below:

$$Y_D(x_d, y_d) = p_{00} + p_{10} \cdot x + p_{01} \cdot y + p_{20} \cdot x^2 + p_{11} \cdot x \cdot y + p_{02} \cdot y^2 + p_{21} \cdot x^2 \cdot y + p_{12} \cdot x \cdot y^2 + p_{03} \cdot y^3 \tag{4.9}$$

$$X_D(x_d) = a_1 \cdot x_d^3 + a_2 \cdot x_d^2 + a_3 \cdot x_d + a_4 \tag{4.10}$$

where a and p are different constants also given by the function. The correction in the x-direction only slightly depend on the y-coordinate. To make this equation

simpler, it is assumed that it only depends on the x-coordinate instead. This is not the case with the correction in the y-direction, where the full formula instead must be used. When these formulas are applied, the distortion is very minimal, seen in Figure 4.7. Even with this implemented, there is still a slight distortion in the corners of the image. Most of these points are not on the actual table though, so they can be ignored.

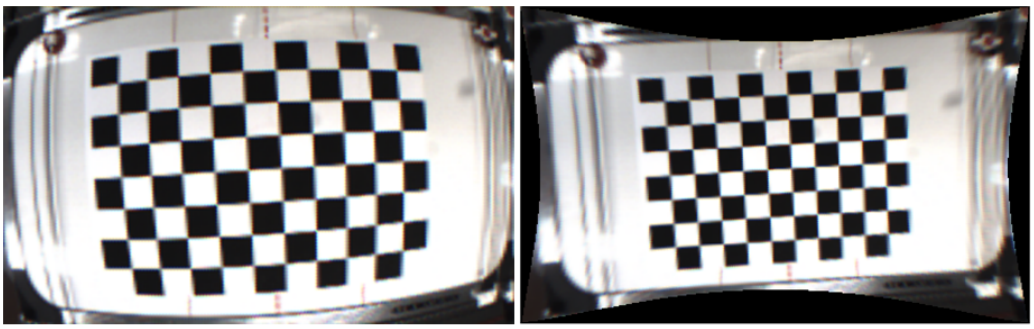


Figure 4.7: Distorted and undistorted image

5

Verification

To meet the requirements set in the beginning of the project, the system needs to undergo a verification process that measures the performance of the system. The results of the verification process gives a brief understanding of how successful the project is.

5.1 Verification Process

To verify the first requirement (R 1), the puck is placed at arbitrary spots repeatedly. If the robot is able to move the paddle to the spot where the puck is located, this goal is considered achieved.

The second requirement (R 2) is verified through measuring how far the system stretches along the table. To meet this requirement it must not reach beyond 99 cm, since this is the length of half the table.

The goal (G 4) is deemed passed if the system can be moved, started and then function properly without any re-calibrations or configurations.

The requirements and goals ((R 3), (R 4), (G 1), (G 2)) are verified analogously. They must all be able to pass the requirement 75% of the shots or more at or above the required speed to pass. The tests are repeated at least 15 times. Since the shots and measurements are performed by humans, the amount of iterations needs to be high to rule out errors as much as possible.

The fifth requirement (R 5) and the third goal (G 3) are verified by placing the puck in front of the paddle and then hit the puck with full speed.

6

Results

Each requirement and goal is tested through the verification process. Presented below is how they relate to the results conceived.

(R 1) Requirement met. The robot is able to reach the puck with the paddle at any point on its own half of the table.

(R 2) Requirement met. The construction does not reach over to the other group's half of the table.

(R 3) Requirement not met. The robot is not able to hit pucks approaching at any speed.

(R 4) Requirement met. The robot is able to defend the goal from a majority of all pucks approaching at 7 m/s, which is above the required 6 m/s.

(R 5) Requirement not met. Given optimal circumstances, the robot is able to shoot the puck at a speed of 1.7 m/s, which is below the required 2 m/s.

(G 1) Goal not met. The robot is not able to hit pucks approaching at any speed.

(G 2) Goal not met. The robot is able to defend the goal from a majority of all pucks approaching at 7 m/s, which is below the targeted 9 m/s

(G 3) Goal not met. Given optimal circumstances, the robot is able to shoot the puck at a speed of 1.7 m/s, which is below the targeted 4 m/s.

(G 4) Goal not met. The vision system needs to be calibrated before start.

7

Discussion

In this section, the results and why the project turned out as it did are discussed. Alternative solutions and possible improvements for future iterations of the project are also included. Part of the discussion concerns some of the problems that were encountered.

7.1 Performance Evaluation

Even though the robot lacks the software for offensive capabilities, it still performs well. The major goal for the project was to construct a robot that can defend as well as the previous project, but also be a better player mainly by being able to play more offensively. Based on the achieved puck speed when the robot is attacking a stationary puck, it is a success. This together with the fact that the robot can reach the puck on its entire side (R 1) and that the construction does not occupy more space than allowed (R 2), the mechanical construction is considered to be sufficient.

The requirement (R 3), to be able to hit the puck with a speed of at least 4 m/s was not met. This is because the torque drops significantly at higher speeds, making the motor calculations inaccurate. In retrospect, the requirement and goals regarding offensive speeds were likely set too high, as achieving these would require significantly larger motors than the already large ones used, or a completely different mechanical concept.

While the goal (G 2), to be able to defend against all pucks incoming at speeds below 9 m/s was not reached, the requirement for defense, (R 4) was passed. The current maximum puck velocity of 7 m/s is still to be considered as a decent playing capability, especially against opponents with limited offensive capabilities, such as other robots. For pucks moving faster than 7 m/s, the Pixy camera sometimes has trouble to keep track of the puck, resulting in strange values being sent to the micro-controllers. This in turn makes the robot move the paddle to the incorrect position, missing the puck.

Since the system needs to be calibrated before it can be used, goal (G 5) is not reached. This is because the coordinates need to be adjusted based on minor camera movements from the camera stand being moved between uses. This is because a decision was taken to not mount the camera stand to the table and instead have

it standing next to it. While making the system more difficult to start, it is still considered as a good decision since the table's movement can not affect the camera in any way.

The Arduino setup turned out to not only provide more computation power, but also an increased programming difficulty. To tackle this, research on the setup and how it is programming should have been started earlier, as this would reveal the difficulty, allowing for more time to be spent on it early on.

7.2 Future Improvements

There are several things that could be improved for future iterations of this project. Only minor improvements would be necessary for the hardware, with the majority of improvements being of the software side.

Even with the robot's limited use, the ball bearings have already caused tears in the steel rods. More durable steel rods and linear bearings of a higher quality would increase the lifetime of the robot, while also reducing friction.

The current camera could be replaced by one with higher resolution and a faster frame rate. A custom-designed image detection algorithm could also be developed to gain full control of the camera, thereby increasing its performance.

A faster device for computations than the Arduino platform could be used. More processor threads and higher clock speed would allow for more extensive programming without sacrificing performance, giving the programmers more time to develop the strategies without worrying about optimization.

8

Conclusion

The purpose of this project was to construct an autonomous mechatronic device that could locate the puck and play accordingly, both offensively and defensively.

The result is a robot that can detect the puck and move the paddle to all locations on the robot's side of the table. The robot can defend the goal from most pucks incoming at speeds below 8 m/s. The robot's mechanical design allows for offensive plays as well, but the programming for such maneuvers is not yet implemented.

Even though most of the requirements set up are met, the purpose can not be considered fulfilled. Due to the lacking offensive capabilities, the robot can not score goals of it's own, which was a necessity for the purpose to be fulfilled. However, not much more work would be needed to implement the code for this statement to change. While not currently being a very good air hockey player, the implementation of the offensive algorithm would likely make the robot into a decent opponent.

Bibliography

[1] IBM100 - Deep Blue. (2016). [online] Www-03.ibm.com. Available at: http://www-03.ibm.com/ibm/history/ibm100/us/en/icons/deepblue/ [Accessed 2016-04-13]

[2] Commons.wikimedia.org. (2005). File:AirHockeyTable.JPG - Wikimedia Commons. [online] Available at: https://commons.wikimedia.org/wiki/File:AirHockeyTable.JPG [Accessed 15 May 2016].

[3] Bergendal, E., Ganelius, M., Gustafsson, J., Hellberg, N., Hesselgren, J. and Karlsson, J. Självspelande air-hockeyspel, examensarbete, Chalmers Tekniska Högskola, Institutionen för Signaler System, 2015.

[4] Ogawa, M., Shimizu, S., Kadogawa, T., Hashizume, T., Kudoh, S., Suehiro, T., Sato, Y., and Ikeuchi, K., "Development of air hockey robot improving with the human players", in IECON 2011 - 37th Annual Conference of IEEE Industrial Electronics, 7-10 Nov. 2011, Melbourne, VIC, Australia; https://www.engineeringvillage.com/share/document.url?mid=inspec_10655dd135e97ac522M5f772061377553database=ins

[5] Samleisure.co.uk. (2016). BUFFALO AIR HOCKEY 7ft TERMINATOR - SAM Leisure. [online] Available at: http://www.samleisure.co.uk/products-page/air-hockey-tables/buffalo-air-hockey-7ft-terminator/ [Accessed 15 May 2016].

[6] Charmed Labs. (2016). Pixy (CMUcam5). [online] Available at: http://charmedlabs.com/default/pixy-cmucam5/ [Accessed 2016-05-13].

[7] Hackster.io. (2016). Air Hockey Robot. [online] Available at: https://www.hackster.io/windowsiot/air-hockey-robot-7d7a24 [Accessed 2016-05-13].

[8] YouTube, (2016). Professional Air Hockey player Cory Dzbinski VS Blade Brown - 2012 Air Hockey World Championship. [online] Available at: https://www.youtube.com/watch?v=8No_vUUyC_U [Accessed 2016-03-04].

[9] "Wire", *Arduino*, https://www.arduino.cc/en/Reference/Wire [Accessed 2016-05-16]

[10] "Serial", *Arduino*, https://www.arduino.cc/en/Reference/serial [Accessed 2016-05-16]

[11] ATMEL - AVR446: Linear speed control of stepper motor. (2006). [Online] Www.atmel.com Available at: http://www.atmel.com/images/doc8017.pdf [Accessed 2016-04-13]

[12] Se.mathworks.com. (2016). Camera Calibration with MATLAB - MATLAB Simulink Video. [online] Available at: http://se.mathworks.com/videos/camera-calibration-with-matlab-81233.html [Accessed 16 May 2016].

[13] Se.mathworks.com. (2016). Description of Camera Calibration application in MATLAB. [online] Available at: http://se.mathworks.com/help/vision/ug/single-camera-calibrator-app.html [Accessed 16 May 2016].

[14] Se.mathworks.com. (2016). Camera Calibration Application in Matlab - parameters received. [online] Available at: http://se.mathworks.com/help/vision/ref/cameraparameters-class.html [Accessed 16 May 2016].

[15] Se.mathworks.com. (2016). Matlab function - use camera parameters to approximate undistorted points. [online] Available at: http://se.mathworks.com/help/vision/ref/undistortpoints.html [Accessed 16 May 2016].

[16] Se.mathworks.com. (2016). Matlab function - approximation of curves/surfaces. [online] Available at: http://se.mathworks.com/help/curvefit/fit.htmlbto2vuv-11 [Accessed 16 May 2016].

A

Budget

The budget for everything needed in the project was 5000 kr. The total expenses actually add up to 6900 kr. Nevertheless, as a result of a common decision with the other group and project supervisor/examiner, a new air-hockey table was bought, with partial funding from the Department of Signals and Systems. The remaining cost was split between the two project groups using it.

There is some money left in the budget. This is because some was reserved for unexpected purchases, which turned out never to be necessary.

Part:	**Cost:** [SEK]	**Planned cost:**	Total: [SEK]
Air hockey table	1500	0	1500
Stepper motors	841	1000	2341
Drivers	974	500	3215
Miscellaneous:	0	1500	3215
Belt	271	0	3486
Power supplies	650	0	4136

Table A.1: Cost of the purchased parts as well as the planned costs

B

Motor Specifications

Nema 17 Stepper Motor

Rev: A
Date:

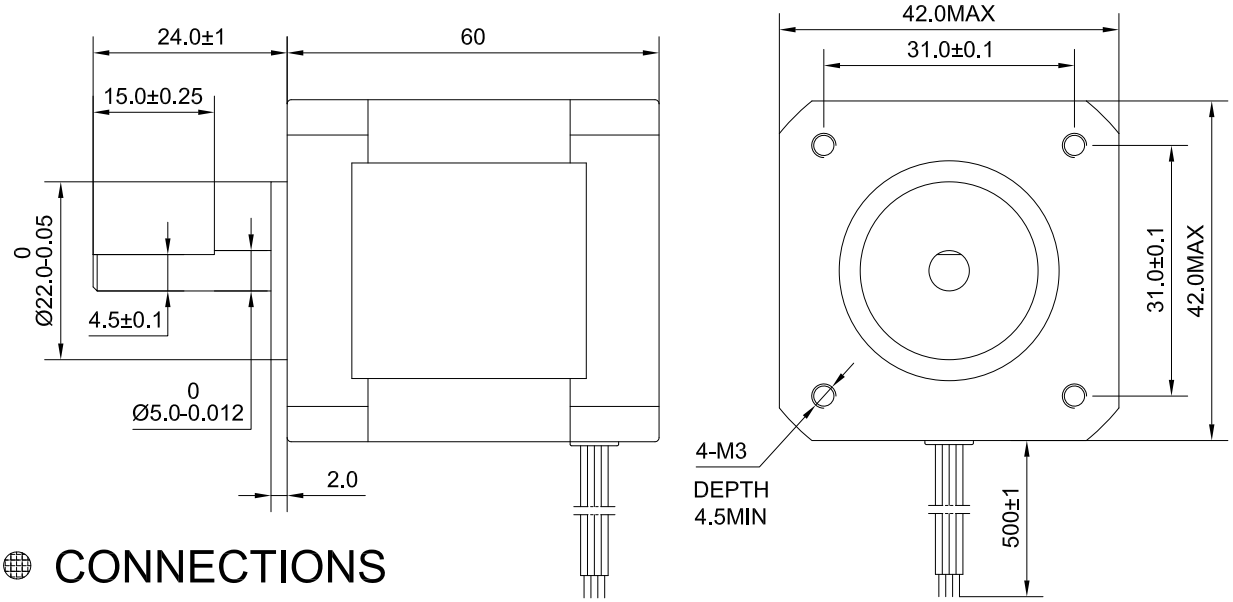
COMMON RATINGS

Step angle :	1.8°	Dielectric strength :	500VAC
Positional accuracy :	±5%	Insulation resistance :	100Mohm(500VDC)
Number of Phase :	2	Ambient Temperature :	-10℃~50℃
Temperature rise :	80℃MAX	Insulation class :	B
Rotor Inertia :	82 gcm^2	Weight :	0.5 Kg

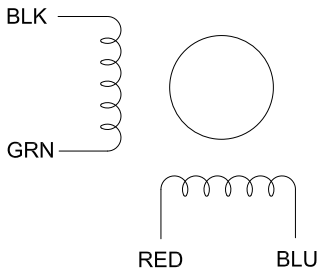
SPECIFICATIONS

Holding Torque (2 phases on) Kg.cm	Rated Current/Phase (Amps DC)	Phase Resistance (ohms) ±10%	Voltage Current/Phase (V DC)	Phase Inductance (mH)±20%(1KHz) Typical
6.5	2.1	1.6	3.36	3.0

DIMENSIONS unit=mm



CONNECTIONS



						17HS24-2104S		技术规格书
标记	处数	分区	更改文件号	签名	日 期			
设计			标准化			阶段标记	重量	比例
审核								www.OMC-StepperOnline.com
工艺			批准			共 张	第 张	

Nema 23 Stepper Motor

Rev: A
Date:

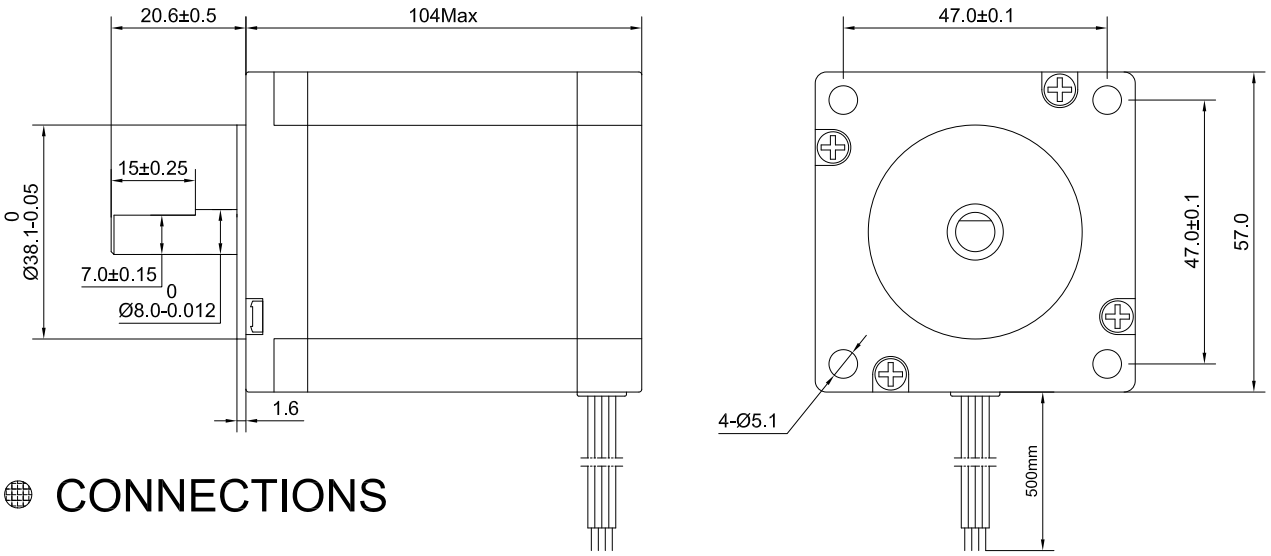
COMMON RATINGS

Step angle :	1.8°	Dielectric strength :	500VAC
Positional accuracy :	±5%	Insulation resistance :	100Mohm(500VDC)
Number of Phase :	2	Ambient Temperature :	-10℃~50℃
Temperature rise :	80℃MAX	Insulation class :	B
Rotor Inertia :	680gcm^2	Weight :	1.25Kg

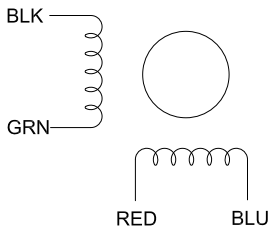
SPECIFICATIONS

Holding Torque (2 phases on) Nm	Rated Current/Phase (Amps DC)	Phase Resistance (ohms) ±10%	Voltage Current/Phase (V DC)	Phase Inductance (mH)±20%(1KHz) Typical
2.4	1.8	2.75	4.95	17.0

DIMENSIONS unit=mm



CONNECTIONS



						23HS41-1804S		技术规格书
标记	处数	分区	更改文件号	签名	日 期			
设计			标准化			阶段标记	重量	比例
审核						共 张	第 张	www.OMC-StepperOnline.com
工艺			批准					

C

Measured Puck Speed

The measured puck speed for requirement (R 4).

Speed m/s	Success or Fail
6.7	Success
8.4	Success
7.3	Fail
6.4	Success
6.4	Success
7	Success
7.3	Fail
7.8	Fail
6.4	Success
7.6	Success
7	Success
7.3	Fail
6.4	Success
6.2	Success
6.6	Success
7.3	Success
7.3	Success
6.7	Fail
6.2	Success
6.2	Success
7.3	Success
6.2	Success
6.2	Success