Handwritten text recognition printer

Gizmo

Design Engineering 2

Contents

Abstract:

The task was to utilize mechatronics and computing to create a gizmo which affords a meaningful interaction between itself and the user. The proposed idea is a handwritten text recognising machine which prints the image associated to the users chosen word. This was done in the form of a multi-layered circular structure including a raspberry pi, where the dimensions are within the requirements of: 250 mm x 250 mm x 350 mm (width, depth, height).

Nomenclature and abbreviations:

All words/phrases with a * attached at the end can be described here.

API	*Application programming Interface*
string	*collection of letters in any order – eg. 'sldjf'*
Half step	*a bipolar stepper motor can have multiple resolutions, in this case half step allows the finest resolution, as in halves the possible resolution of full step*
step	*The minimum rotation between two electromagnetic positions in a bipolar motor*
class	*Controls the input and output of each layer in a neural network*
PIL	*Python imaging library*
JavaScript	*Another programming language and platform such as python*

Background:

Research:

The initial concept can be broken down into 3 main parts: 1. The recognition or input of a word from the user, 2. The processing of the word and turning it into a printable output, and 3. The printing of the image.

Recognition:

This process contains the initial part of the interaction. The user should be presented with a means to input this data into the gizmo. This interaction should be more than some computer input or typed text as the desired interaction encourages the use of more hands-on methods such as handwriting, speech or touch. There were 2 main methods of achieving this input, either, 1. a predetermined set of words that the user can input, meaning the processing would only include matching the input with existing data. 2. The harder method was to allow any word to be an input, or perhaps not even a word. This meant the processing would search for individual letters and instead of matching data, producing new data which would have to be based on the alphabet. An example for each option could be: 1. Matching the recorded speech to a list of 5 possible words only, or 2. Creating a word/string of letters from the recorded speech.

Matching can be done by finding similarities between two datasets, the input dataset and the existing dataset in the form of an website or app which is for recognising speech, or handwriting but creating the word from input would include a way of analysing each input and matching it to a much larger dataset such as a dictionary. However, the hardest option is to analyse each individual letter, meaning the output can not only be a word but any string of letters in any order. All 3 possibilities can be achieved using different APIs* or extension packages such as OpenCV or TensorFlow.

Processing:

Providing the input is a 'string' of letters, the next stage is to process the string into a printable image. This can be done off an imaging website link (URL). This stage can be further broken down into 1. Searching for an image using the inputted string, 2. Downloading the image, 3. Processing the image into something that can be printed. Steps 1 and 2 include using the internet to search and locate an image, and step 3 can be done locally in the computer. Researching image retrieving from the internet provided a variety of ways to download an image. The main idea, however, is to use "urlib.request" which will allow an image to be found as well as downloaded. This will likely be the method used to go from input from word recognition, to image processing. Stage 3 requires joined thinking with the printing of the image, specifically, time, resolution and size. These 3 things are correlated as it will greatly affect the outcome of the user interaction. The desired outcome from the entire processing stage is to have a printable format of an image completely ready for the final printing stage.

Printing:

To print an image, there are multiple things to consider such as: what printing mechanism to use, and what format the printing mechanism accepts. As the image processing stage will be done through python3, the printing mechanism will be run off python. There are many ways of reading an image in python and giving the mechanism commands, but to keep the interaction and printing mechanism simpler and more achievable in the time frame, a single coloured image is best. This can be done with a pen and there are many ways of making pen-printers with just a servo and a pen. The main problem is dealing with x and y axis movement, guiding the pen to print in the exact spaces. A possible solution is to use stepper motors which have unlimited rotation unlike servo motors. This can be used in conjunction with a lead screw to convert rotational motion into linear motion. A good example of this mechanism in use is in old DVD/CD writer machines. These machines come equipped with tracks, a laser but most useful of all, a stepper motor with a lead screw already attached. This also reduces the cost of the concept as stepper motors are very expensive especially with a lead screw while an individual DVD/CD writer machine is around £2-5 and contains both the stepper and the screw. However, the stepper extracted is much smaller in size, which means if it were to be used, the printing mechanism would have to be designed around the dimensions of the stepper motor.

Figure 1 – DVD/CD writer machine parts

Parts extracted from the DVD/CD writer, containing 1 stepper motor each, also called the “CD Stepper motor” are shown in figure 1.

Figure 2 shows a clearer photo of how the stepper motor works. The stepper motor has a white component part which holds onto the screw thread using some lubricant, allowing linear motion (circled in red).

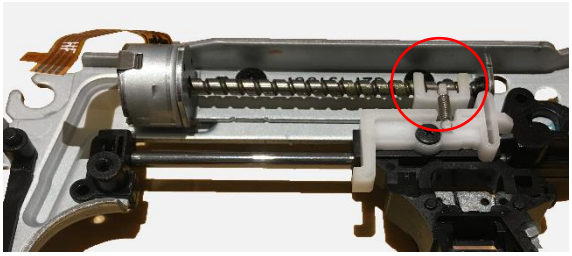


Figure 2 – DVD/CD writer stepper motor

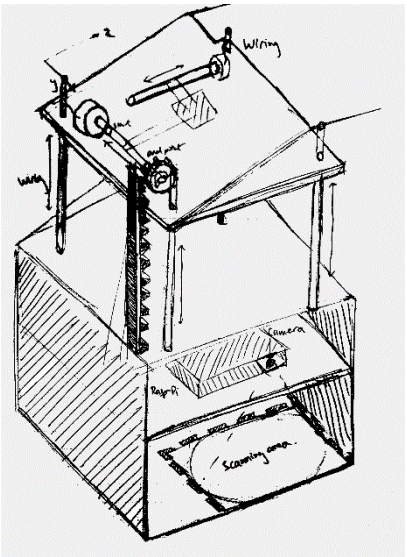
The CD stepper motor is a bi-polar stepper motor which contains 2 separate coils to drive, it will also need a driver chip which there are many options for, such as the A4988 or DRV models.

Inspiration:

There has always been an interest in creating a machine which is able to print, as the mechanisms are tricky and difficult to work with. The concept was created from an idea to create a physical version of google search, a method more direct in getting an image than having to connect a printer to a laptop and providing enough ink to product a very basic depiction of a word/phrase. A creation of a multi-levelled printing machine which is as straight forward to use as the idea sounds was intriguing and ultimately the idea that was decided.

Initial design:

As an initial design, sketches were made to visualise the concept structurally and mechanically.



Fiaure 3 – First sketches

Figure 3 shows the first layout of the concept and the 3 stages of recognition, processing and printing all in one machine. The design has selected certain mentioned possibilities in the research stage such as using handwritten text as the main input and such the scanning area at the bottom of the design to start this. In this design, the size of the gizmo is exactly the maximum dimensions of the width and depth stated before, and the top layer which includes the printing mechanism moves up or down to indicate progress of the printing. This provides feedback for the user and makes the interaction more satisfying. The mechanism behind this includes a rack and pinion which is connected to the lead screw of one of the 2 stepper motors. The design also includes 2 stepper motors as it will allow movement in the X-Y plane, which is the surface of the top face.

Figure 4 shows servo schematics and how the printing stage will be done. A servo motor will have a pen attached to a plastic attachment, moving 30 degrees above horizontal when there is nothing to print. The two stepper motors will each move in one direction and a bar (support beam) opposite each stepper

motor will act as support and to keep the servo motor up, and the pen at the same level, allowing the pen to create cleaner marks.

The servo motor will be placed on a platform which is guided by rods attached to each pair of stepper motor and its opposite support beam and as the dimension of the lead screw in the CD stepper motor is 4cm, the maximum printable image will be 4cm x 4cm. the image can be altered to have different resolutions which will depend on how large each pixel the printer will make.

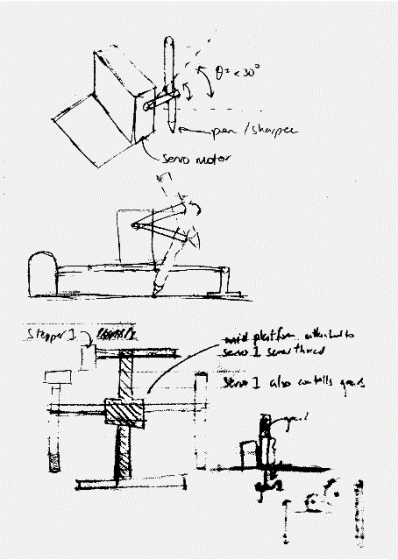


Figure 4 – printing mechanism

Figure 6 shows a clearer image of how the printing stage will look like including wiring which will go downwards.

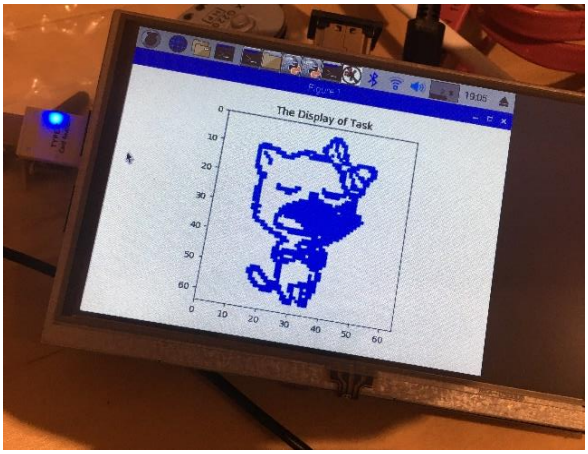


Figure 5 – Print image

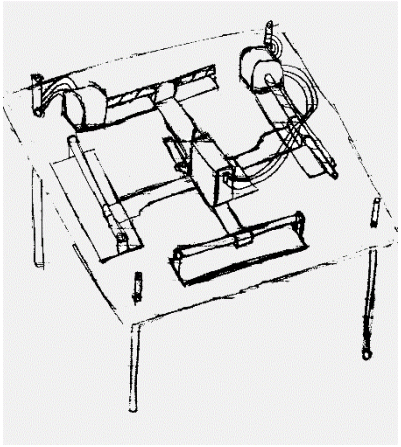


Figure 6 – Print stage

Development:

To check viability for the initial design, prototyping, FEA analysis, and first stage coding was done.

Initial Prototype:

In terms of the form of the gizmo, a more circular outlook for the product was tried as to mimic a feeling of upwards ascension of progress while the user is interacting with the product.

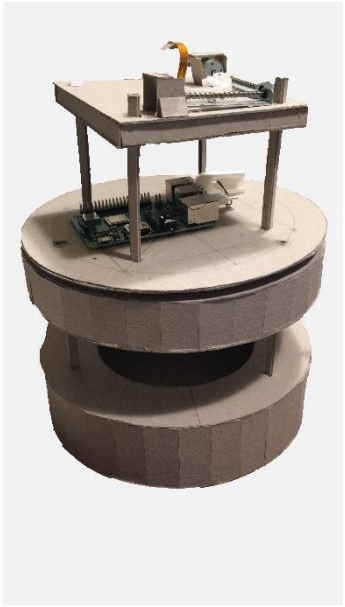


Figure 7 – Prototype

The purpose of this model: Size, space for each layer and printing stage.

The prototype contains 3 separate levels, all interconnected by rods going from layer 1 (bottom) to 2 (middle) and layer 2 to 3 (printing stage). The diameter of layers 1 and 2 are the same and are 250mm. Layer 1 will be used as the input layer where the user will insert a piece of paper with a handwritten word on it. This will be later scanned by a raspberry pi camera on the bottom of layer 2. Inside layer 2, the raspberry pi will be placed, but in figure 7, it is placed above just to indicate the approximate sizes of component parts. Layer 2 will also contain all wiring connecting the raspberry pi and other components such as stepper motor drivers to the moving parts in layer 3.

The input in layer 1 is a piece of paper with dimensions 40mm x 40 mm. once the camera has taken a picture of this piece of paper, the user can flip the paper and move it to layer 3 where the other side will be an image of the requested image. The interaction provides a souvenir as well as a sense of progress as the piece of paper goes through the layers.

The card model aims to show dimension and feasibility of each layer and approximately how much space is required. From the model, layer 1 will require as much space depending on the camera (the likely method to capture the image) breadth and zoom. The second layer will also require space for wiring to layer 3 which in the model, has been given 4cm of height, accounting for the raspberry pi's height and 2cm more, this may be modified later. Layer 3 is modelled as the most open layer, as it requires many wires connecting to its components. Overall this model proved helpful in further developing areas such as the connection between layer 2 and 3, as well as space allocation for all 3 layers.

Printing stage alterations + Engineering Analysis:

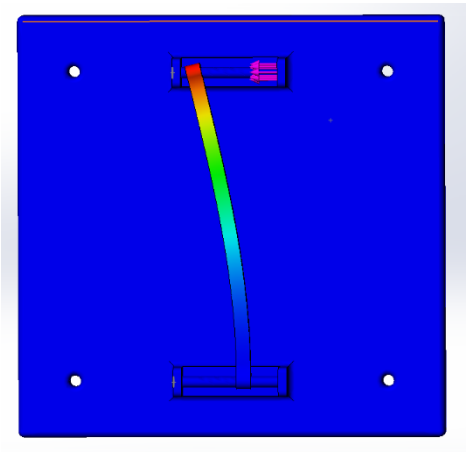


Figure 8 – Top face FEA analysis

Applying a small force mimicking a step in the stepper motor turning and translating rotational movement to linear movement in the rod moving in the X or Y axis. The amount of force which the rod would move can be estimated using the equations:

$$Power = Current * Voltage$$

$$Power = Force * Velocity$$

Maximum Current	*200mA*
Maximum Voltage	*12V*
Maximum Velocity	*4cm/s - 0.04m/s*

These values show the maximum force in the bipolar motor at half step* is 25N. This value has been used to model the FEA analysis shown in figure 8. The figure also shows greater deformation in the top (stepper motor controlled) than the bottom (support beam – slides parallelly with the top stepper motor's actions). As the deformed rod will be much more rigid, it will be at an angle to the vertical rather than bending. This also means that as the rod does not slide on the support beam in the same time as the stepper motor, the rod will be stuck and will not be able to move accurately thus failing by malfunctioning. As a result, a decision to use two stepper motors on either side, removing the need for a support beam has been enforced to remove this asymmetrical displacement.

The top printing platform has also been altered to not move up and down for user feedback. This is due to the same problem of asymmetrical displacement.

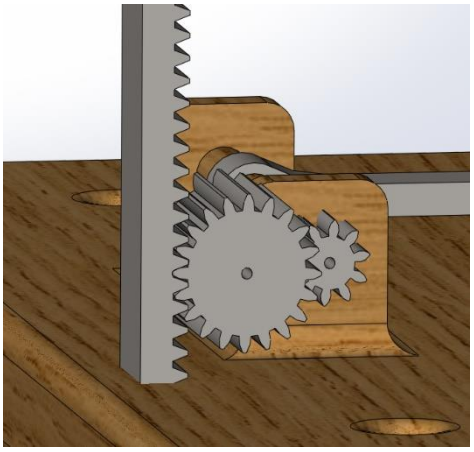


Figure 9 – Top face Gear concept

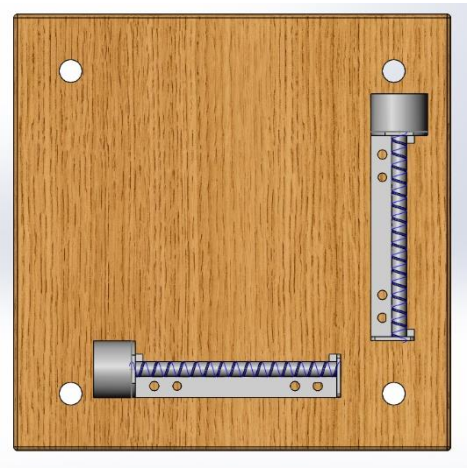
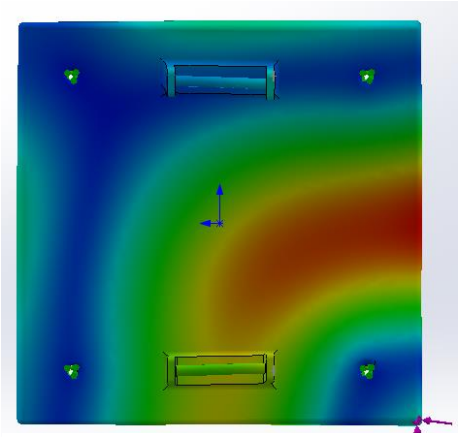


Figure 10 – Top face arrangement

Figure 10 shows the mechanism for the proposed moving printing platform. The stepper motors will be controlling 2 spur gears (ratio 1:2) which will translate to linear movement of the platform. However, as there are only 2 stepper motors and if arranged as shown, creates imbalance for the platform (figure 12), imposing asymmetrical displacement. In figure 11, stress concentration can be found corresponding to the location of the rack and pinion gears. Hence, the moving printing platform has been reduced to a fixed platform on the top of the gizmo. A solution could include 4 sets of rack and pinions, but it would render the platform very clustered and much harder to interact with and therefore removing it is the decision.



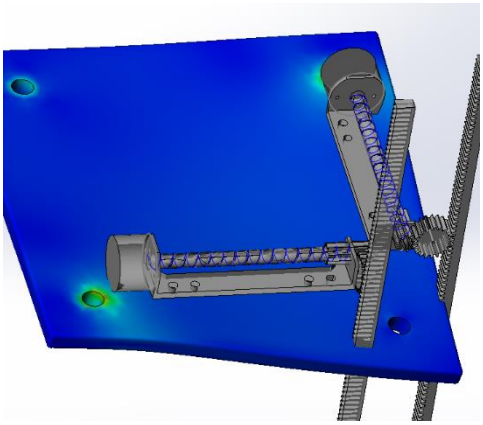


Figure 11, 12 – Top face FEA analysis 2

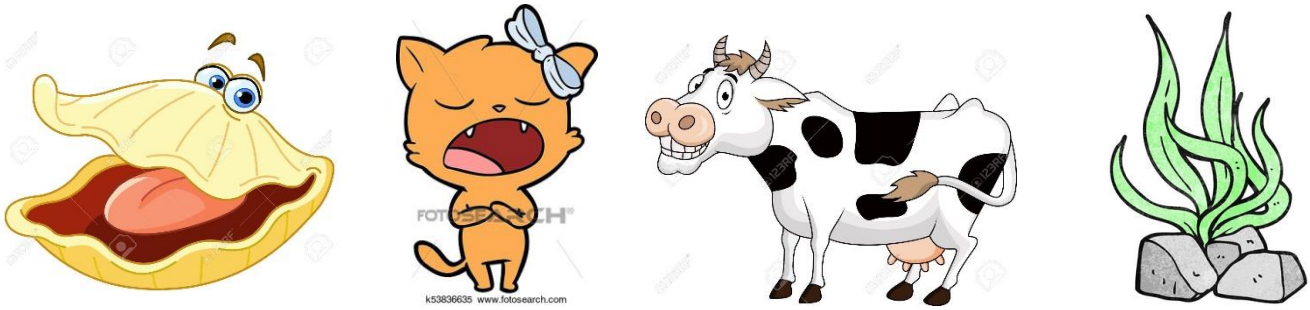
First Stage Coding:

```
#selects a random browser and requests a link for an image
def link_finder(ua,a,url,C,x):
    headers = {"User-Agent": random.choice(ua)}

    req = requests.get(url, headers = headers)
    html = req.content

    soup = bs(html,"lxml")
    images = soup.find_all("div",{"class":"rg_meta"})

    images = [i.text for i in images]
    images = [json.loads(i) for i in images]

    for x in range(100):
        if len(C) == 1:
            break
        Link1 = images[x]
        if Link1["ity"] == 'jpg':
            L = Link1["ou"]
            print(L)
            C.append("finished")
    return L
```

```
#Uses the link to download the image directly
def downloader(image_url):
    file_name = a
    full_file_name = str(file_name) + '.jpg'
    print(full_file_name)
    print(image_url)
#    os.system("wget -O {0} {1}".format(full_file_name, image_url))
    while True:
        try:
            urllib.request.urlretrieve(image_url,full_file_name)
            break

        #initial testing showed some searches showed problems
        except OSError:
            print("Im sorry, if we do this, it will be copyright infringement")
    return full_file_name
```

Figure 13, 14 – initial Image Processing

Initially there was a focus on image processing, more focusing on searching for an image to print and processing it to allow printing. Above, 2 functions are shown, “link_finder” locates the URL of an image, using a range of web browsers such as Mozilla Firefox, google chrome, safari or others, to give the URL retriever options in case one browser doesn’t work. The full list of options is shown in lines 18-20 in file: Adjust2. “link_finder” utilizes the library beautifulsoup4 which is shorthanded to bs as in line 36. This is a very commonly used library in python to retrieve information from web pages. The library was first released in early 2014 and has been updated constantly, it is also perfect for grabbing URLs efficiently. Json.load(i) in line 40 is also used to covert javascript*, in this case object notation, into python which extracts “i” into URL format. After a URL for an image is established, “urlib.request” can be used as mentioned before, to download the requested image. This requires the command “urlretrieve” from the library “request” and is the most straightforward method to download something from a website.



Left to right; Figures 15 -18 – “oyster”, “singing cat”, “cow”, “seaweed” from image URL download

Figures 15-17 show examples of testing the image retrieval code. From trial and error, only searching for the input such as “cow” or “dog” can give very complex images, and so in the search for the image, the word “cartoon” is added before the input to narrow the search to more simplistic images which could be printed easier.

Subsystems testing:

Handwriting recognition:

Keras:

Keras is a python library which tackles machine learning and includes neural networks and is extensively used for recognising new datasets such as handwriting from trained datasets of which contains words or phrases, accustoming the neural network to a handwriting or letter type. The Keras library was explored as a basis to learning about machine learning and how to use this to predict different peoples handwriting, however, there were a few very time-consuming processes to learn and do which ultimately rendered doing it implausible: A neural network contains many layers of training and within each layer, the training data has to go through many batches and epochs which directly correlate to how good the model is at predicting handwriting. Another task would be finding training data suitable for the task, ideally completely created for this model, however this would involve taking thousands of photos of different text as well understanding what text to take pictures of.

As a result, an existing piece of code has been taken from GitHub called: SimpleHTR.

SimpleHTR:

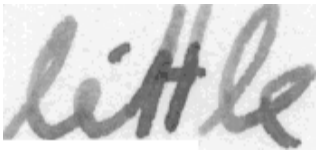


Figure 19 – training sets

SimpleHTR is trained based on datasets and photos which look like figure 19, which has each word in a confined box and clear black and white separation. It uses TensorFlow to create a neural network recognising images such as figure 19 into "little". The accuracy of the model is around 90% however it varies greatly depending on the style of handwriting and how neatly it is written. Clearly written individual letters have an accuracy of up to 95%. This is very accurate given that the training data is predetermined and is not tailored. The existing training data also includes cursive handwriting although not to an extreme extent, in aims to incorporate more styles. The speed of this code tested on the pi is around 90 seconds from input of the image to the output of text which is ideal.

The version of python that is being used on the raspberry pi 3B+ is 3.7.3 but the latest version of TensorFlow only was released for version of python up to 3.5, meaning that to run TensorFlow on the 3B+ will require some method of downgrading the existing version of python. However, the method that was chosen was to install two different versions of python and to run code which called python3.5 for TensorFlow and python3.7 for anything else.

Image Processing:

Processing the downloaded image is required to make give the printing mechanism and stepper motors an input. To do this, a matrix indicating which areas needed ink was the chosen method as it required the least number of nested loops of code. Printing using a matrix has a few points of consideration, namely size of matrix or quality of image, printing method, and finally what is being printed. The quality of print that is desired will greatly affect the amount of printing time. Figure 20 shows 3 different displayed matrixes of different print qualities, and the number below each matrix indicates the amount of 'pixels' which have ink in them. The highest quality print would be 1458 filled pixels and if each pixel takes 0.5 seconds to fill (which would be fast), then the total time for printing would be 12 minutes 9 seconds. This high-quality image has a resolution of 140x140 pixels and to fit that within a 40mm x 40mm would mean each pixel is 285 x 285 microns large which a 0.5mm ball point pen would take up a total of 4 pixels space in one marking. If the ball point pen's area is the minimum size per pixel, then the minimum pixel size would be:

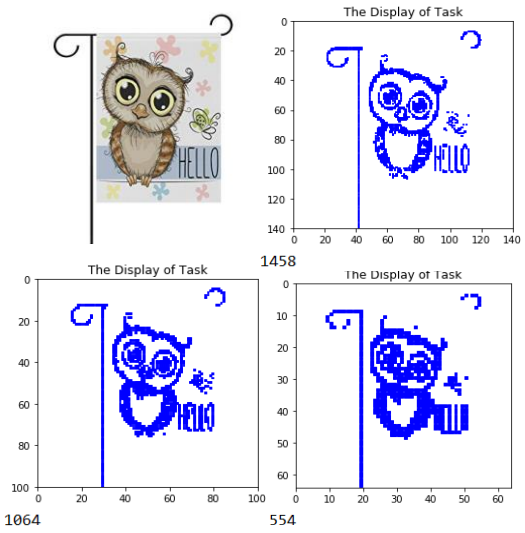


Figure 20 – matrix quality consideration

$$paper\ length\ (4cm) = \ point\ length * max\ number\ pixels$$

$$\frac{40mm}{0.5} = max\ number\ pixels = 80$$

As 80 is the maximum pixel size, the resolution would still have dropped below the middle pixel size, 1064. Resolution of the 1064 matrix is 100x100 is still 0.4 x 0.4 mm per pixel. The chosen resolution is 64 x 64 pixels, this leaves enough space for each mark to show a small gap in between, firstly preventing the chance of overlap between two pixels and secondly to provide the stepper motor with enough distance to do one minimum 'step*'.

DRV 8825:

Controlling the stepper motor required a driver which there are many options for, the chosen driver was the DRV 8825. As there are 4 stepper motors, the normal approach is to use 4 separate drivers to control each individual stepper motor. The 4 stepper motors are paired up to function as X and Y direction motors, meaning each opposite pair of motors can be controlled using the same code and therefore output pins on the raspberry pi. The DRV 8825 can receive voltages between 9V and 12V which means it is very dangerous for the pi, as any contact to the pins of the pi will immediately

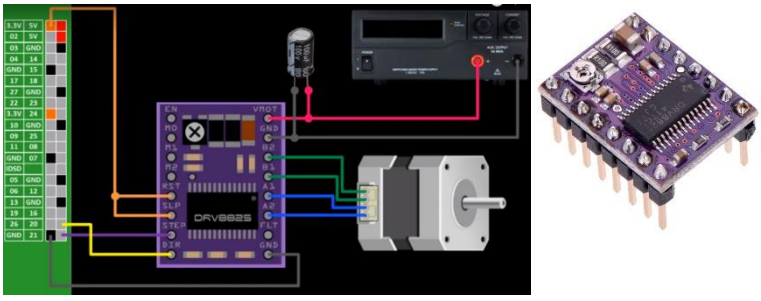


Figure 21 – DRV 8825 schematic + photo

fry the pi. The driver also has a variable output controlled by rotating a screw shown in the top left of the DRV 8825. This controls how much current gets passed to the output pins A1 A2 B1 B2. Normally the CD stepper motor only requires 250mA to run, however giving it the bare minimum can cause malfunctions so 275-300mA is ideal. As each opposite pair of steppers are going to be controlled by one driver, the maximum output current can be doubled to around 550 – 600mA to fully control both stepper motors. The stepper motors can also be controlled to different degrees of detail, regarding step, ranging from full step to 1/32 of a step. The main use of smaller steps is to increase the accuracy of positioning however another use is to create a smoother action and rotation.

Linking the matrix to the stepper motors:

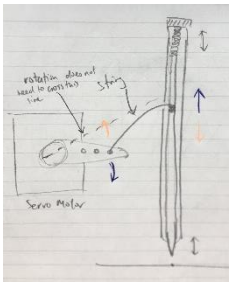


Figure 23 – printing Trial 2 Sketch

To link the matrix to the stepper motor, the code is simply uploaded onto the raspberry pi and for each pixel, the stepper motors move according to a set method of printing. The printing pattern is shown in figure 22. This arrangement saves 2 seconds / row of printing, in the case of 64 pixels, 128 seconds or around 2 minutes. The initial proposed idea of printing shown in figure 4 has been tested and clearly shows attaching a pen directly to the servo motor to rotate will draw lines instead of draw dots, as to draw dots, the pen must be placed very precisely. Another problem is that as the pen swings, the paper will also move, causing an inaccurate print. Trial 2 included an upright method of printing, attaching the ink cartridge to a string, converting the rotational element to vertical motion, shown in figure 23, the string is attached to the inside of the pen through a small hole in the side.

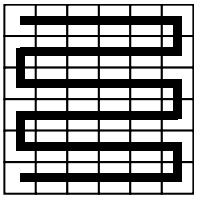


Figure 22 – printing method

Final Design:

Total Weight	Total Cost	Time per interaction
1.719kg	£125.58	4 minutes 30 seconds

The final design consists of a 3-staged circular-layered structure which brings the input of a piece of paper with handwritten text through a process and ultimately prints an image on the other side of the paper. The estimated time of the interaction is split into 1 minute on TensorFlow handwriting recognition, then 1 minute of image retrieval and finally 2 minutes 30 seconds of printing. The printing mechanism has been sped up to approximately 3.5 pixels per second, which would print the 64 x 64-pixel resolution image in figure 20 in 2 minutes 38 seconds. The structure is rendered in figure 24.

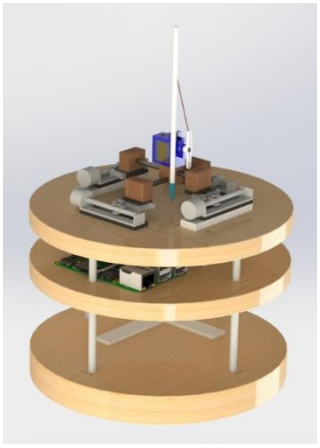


Figure 24 – Final design

Circuitry:

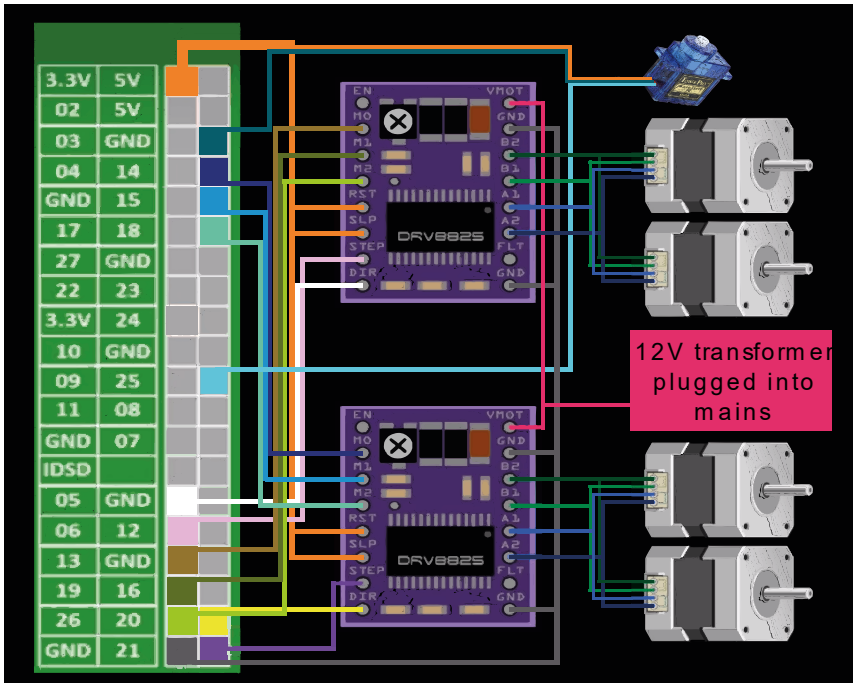


Figure 25 – Total Circuitry

Figure 25 shows all connections between the raspberry pi and all servo and stepper motors with their respective DRV 8825s. The gizmo requires 2 wall supply sockets, one to power the pi individually and one to power the drivers, this is to prevent overlap and frying the pi. Each driver requires 5 independent pins excluding ground and 3.3V pins. As there are only 2 drivers, the number of pins needed on the pi have been halved.

2 extra LEDs (pins 2 and 3) and a button (pin 4) have been placed on the edge of layer 2 for an initial start button as well as feedback from flashing LEDs. The button runs a first code file which subsequently calls the rest of the code in sequence. The feedback LEDs have a sequence which has been iterated through user feedback. As

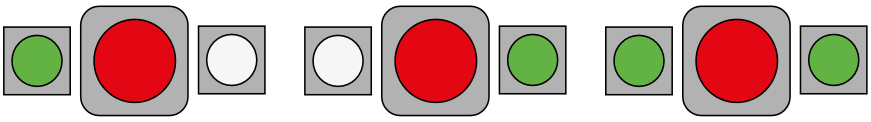


Figure 26 – Light flashing sequence

there are three main steps in the interaction: Recognition, Processing, Printing, the light sequence can be split into 6 sections, pre-recognition, post-recognition, pre-processing, and so on. In figure 26, each LED is next to the large red button in the middle. Each 'Pre' step will be a flashing light which is selected in green, and each 'post' step will be a 3 second stable light of the same selected light. Initially the user will place a piece of paper in the specified slot in layer 1 and press the button once, this activates the interaction.

Pre-Recognition:

The initial input is a 4cm x 4cm piece of paper with its top left corner slotted into layer 1's allocated slot (slightly shown in figure 24). Afterwards, a raspberry pi-camera is used to take a photo of the paper. However, the image will be very dark due to layer 2 covering layer 1. An initial use of lights was considered but rejected as it would include more wiring than there already is. Therefore, image processing was chosen as the method to convert the image to a form like figure 19. Figure 27 shows the transformation to get the desired output.

Figure 27 – Image processing of user input

From left to right, the procedure was: manipulating brightness and contrast, removing green belt (colour removal/replacement) and then a threshold. All these steps have been done by understanding what each pixel is, in terms of its matrix and manipulating it into something that is desired, in this case white and black.

Recognition:

The overall accuracy of the recognition stage is around 60%, even though this doesn't seem much, the number exceeds expectation as it more than 20 different handwriting styles from different people. This would be an area to consider developing if revisited. With more knowledge around neural networks and how to build one for this purpose, the output could see up to 20% increased accuracy, if trained properly with multiple classes* and the correct data within each class.

Processing:

The most successful part of the interaction was bringing the output of the recognition stage to image printing and moving the servos in the right time and to the right location. All of this was done in python and was successful due to the accuracy that can be achieved from coding and image manipulation using the PIL* library. The final printing image was a 4096-pixel resolution image (64 x 64) and has an average printing time between 2 – 3 minutes depending how the threshold was done, as shown in figure 20.

Printing:

The printed image has a good quality due to the accuracy of the stepper motors as well as the image resolution.

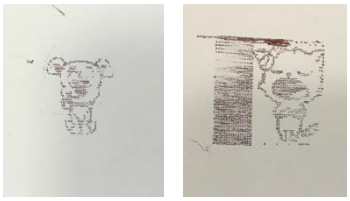


Figure 28 – printed images

Figure 28 shows two examples of printed images each only 2cm x 2cm to minimise printing time, first of a cartoon dog and the second of a singing cat (also shown in figure 16). However the printing mechanism which uses a servo motor to pull a string attached to the heart of the pen's ink, moving it up and down (figure 23), is not as efficient as desired, the ball point pen loses its ink after one or two prints, which then requires replacing and resetting the string attachment. This will be further discussed in the analysis section.

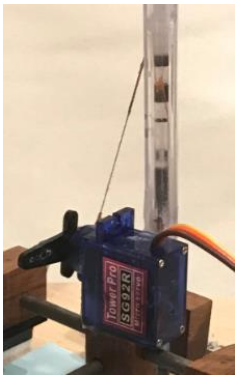


Figure 29 – printing mechanism

The parts connecting the opposite pair of stepper motors is a laser-cut part made of acrylic. This sits well on top of a white connecting part which slides on the lead screw, affording horizontal movement. On top of the acrylic part, sits a wooden block with a one ended hole, used to place a mild steel rod to connect the twin connecting system for the opposite stepper motor. One pair of stepper motors are higher than the second pair, so the interconnecting mild

steel rods do not overlap, a joint is created connecting the two rods with holes perpendicular to each other. This centre platform is for the servo motor to sit on, and in turn to control the pen's movements.

The pen itself has been deconstructed for this purpose, with the spring replaced at the back end, and a small hole drilled into the side of the pen, about halfway down so a piece of string can attach the servo motor's mini rotating arm to the ink cartridge. As the servo rotates one direction, it pulls the string and the pen vertically. Rotating the other direction will release the pen from its charged state due to the spring above it, allowing enough force for a mark to be made on the paper.

Stepper motor connector:

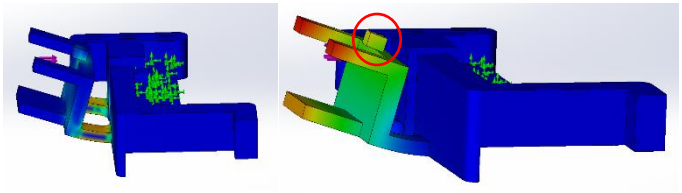


Figure 30, 31 – stepper motor connector FEA analysis

A likely part which will fail over time is the stepper motor connector. There is a spring which connects the front of the attachment to the back end shown in figure 32 where the two connected parts are highlighted in red circles. This causes a deflection of about 3mm upwards, and to model this, a force has been applied to the block shown in figure 31 in the red circle of 3.5N, which shows a modelled deflection of 2.89mm. the spring constant of the connecting spring can therefore be calculated using Hooke's law (within elastic limits):

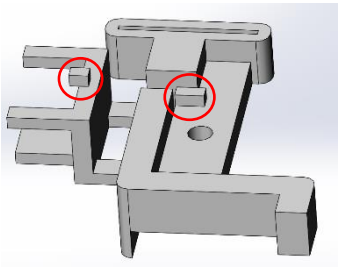


Figure 32 – spring connection

$$F = k * x \qquad 3.5 = k * \frac{2.89}{1000} \qquad k = 1211\ N/m$$

To analyse frequency of use failure, a second situation of the attachment area bending downwards will be modelled again by 3.5N of force and is shown in figure 33. The results of 1,000,000 cycles from this deflection results in no damage. This is likely because the maximum stress still causes deflection well within the elastic limit.

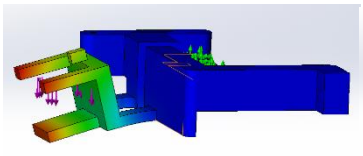


Figure 33 – situation 2

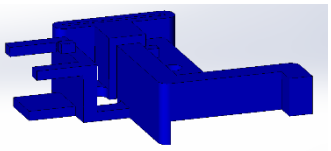


Figure 34 – fatigue failure

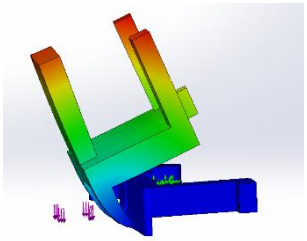


Figure 35 – buckling failure

From buckling analysis, the mode of failure will result in a deformation similar to figure 35. The elongation of the attaching part is due to the material being stretched more than it is deformed. The force required for buckling is 86.6N which is very unlikely any force of that magnitude will affect this part.

Conclusion and Design Analysis:

Overall, the gizmo works well and for the most part consistently, a lot has been learnt especially in coding, incorporating libraries to accomplish tasks in multiple ways such as image processing and neural networks (mainly understanding the outlying structure for building one). The software side of the gizmo is definitely a success.

In terms of the physical design, the 3 layered structure is as desired however layer 2 and the circuitry is messier than expected and would be best if it was to be contained in some way. The initial intended outcome was to show a "processing" layer which will include wires and circuitry, for the user to imagine the machine working to find an image to print. However, this does not affect the gizmo in any way besides aesthetically, which may cause the user to find the interaction slightly less engaging.

Resource allocation:

The area with the most amount of time spent is choosing and building the printing mechanism. This area requires a lot of precision, working with multiple parts which attach indirectly to the stepper lead screw. Just the XY stage itself took 4 iterations to create, which is definitely an area which can be improved. To line up the printing rods for the servo motor platform, 3D printed blocks could have been a better method for precision, however, would be more time consuming and harder to adjust and remove material compared to mahogany wood.

A major cost was the raspberry-pis. Initially, the pi and the DRV 8825s were powered by the same wall supply with two step-down transformers, but this method was risky as the wires had a large chance of intersecting and frying the pi. Each pi costs around £33 – 34. A total of 3 pis were fried, before deciding two wall supplies would save more future potential cost.

Design process:

The design process was split into 3 stages. 1. Ideation and concept practicality research, 2. Prototyping, testing, final design, 3. Final high-fidelity model, polishing up. Phase 1 took around 1-1.5 months as many ideas were exchanged and many possible solutions to the initial concept were provided through research. Entering phase 2, an initial design was tested with critical criteria which therefore eliminated parts of the design such as the moving printing platform which took around 2 months. Once the final mechanisms and methods were clear, the timeline entered phase 3 which was the final 2-2.5 months of the project. Here the building of the final model took some iterations such as the parts controlling the movement translated by the stepper motor. However, overall the tasks were split quite evenly over a 6-month span.

Areas of improvements:

The largest obstruction is the printing mechanism, not producing prints as easy as expected as of an ink issue. The printing mechanism does work, however only for a couple prints. Another method such as replacing the servo motor with a DC motor and attaching a snail cam which will provide a drop where a mark could be made could be better. A jelpen would also serve better than a ball point pen as it doesn't require lateral movement for it to deposit ink.

References:

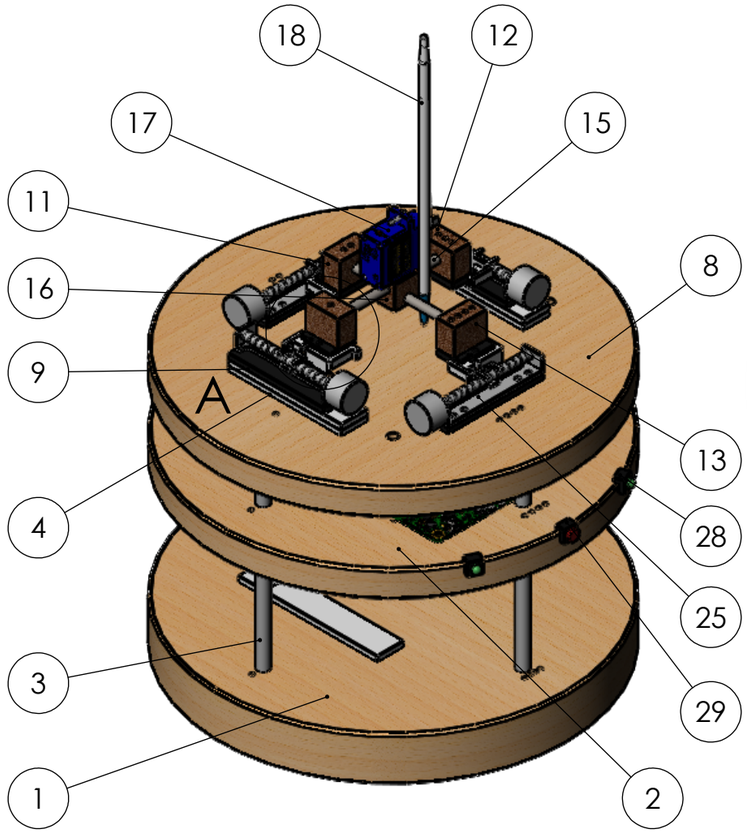
1. Taking old DVD writers apart for Stepper motors and rail, maybe laser:
    a. https://www.instructables.com/id/Disassembling-a-CDDVD-reader-and-reusing-its-parts/
2. Video of how to wire stepper motor from DVD writer
    a. https://www.youtube.com/watch?v=cwwYBgH-w8A
3. Python handwriting recogniser method 1: requires a lot of datasets for letters a-z
    a. https://github.com/snazrul1/PyRevolution/blob/master/Puzzles/Handwriting_Recognition.ipynb
    b. https://github.com/githubharald/SimpleHTR
4. Google searching
    a. https://docs.python.org/2/glossary.html#term-2to3
    b. https://stackoverflow.com/questions/2792650/import-error-no-module-name-urllib2
    c. https://www.youtube.com/watch?v=MaWm1VpWj1A
    d. https://pythonprogramminglanguage.com/get-all-image-links-from-webpage/
5. Installing packages onto the pi:
    a. https://linuxconfig.org/how-to-install-python3-beautiful-soup-environment-on-debian-linux
    b. https://www.raspberrypi.org/learning/visualising-sorting-with-python/lesson-1/plan/
    c. https://stackoverflow.com/questions/37604289/tkinter-tclerror-no-display-name-and-no-display-environment-variable
6. Controlling and using pi
    a. https://www.youtube.com/watch?v=LUbhPKBL_IU&t=922s
    b. https://www.electronicshub.org/raspberry-pi-servo-motor-interface-tutorial/
    c. https://www.rototron.info/raspberry-pi-stepper-motor-tutorial/
    d. https://www.youtube.com/watch?v=RznKVRTFkBY&list=PLZbbT5o_s2xrwRnXk_yCPtnqqo4_u2YGL
    e. https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_gui/py_image_display/py_image_display.html
7. Downloading python3.5
    a. https://www.youtube.com/watch?v=QTBRBr6o1Ms
    b. https://www.liquidweb.com/kb/install-pip-windows/

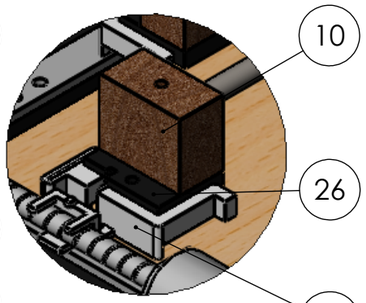
Images Sources:

Figure 15	https://thumbs.dreamstime.com/z/oyster-cartoon-19731763.jpg
Figure 16	https://previews.123rf.com/images/lineartestpilot/lineartestpilot1802/lineartestpilot180214616/94972941-cartoon-singing-cat.jpg
Figure 17	https://previews.123rf.com/images/idesign2000/idesign20001205/idesign2000120500017/13446445-funny-cow-cartoon.jpg
Figure 18	https://st2.depositphotos.com/1742172/9624/v/950/depositphotos_96246944-stock-illustration-freehand-cartoon-seaweed.jpg
Figure 21	https://www.robotshop.com/uk/drv8825-stepper-motor-driver-header-pins-soldered.html?gclid=Cj0KCQjw-tXlBRDWARIsAGYQAmc6ST1M4W8sNdjLMViMZOK0ka7YuFw8WvwGYblxFTzQxnbise7L_hIaAqjlEALw_wcB

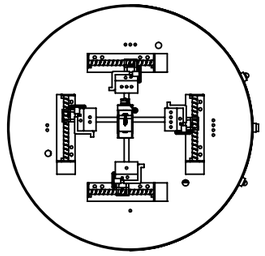
GitHub:

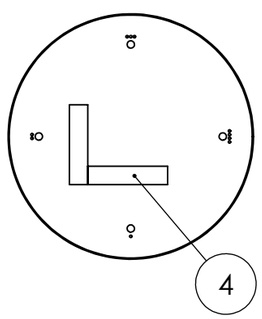
https://github.com/Elvislee123/HandwritingRecognitionPrinter



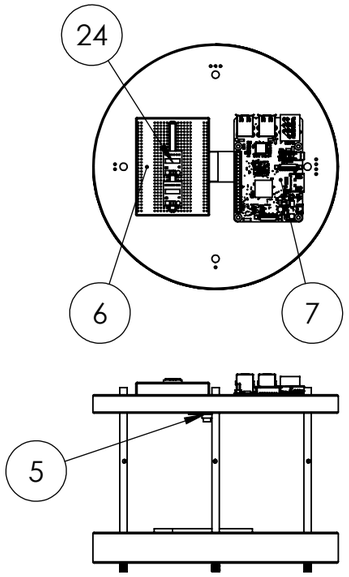


DETAIL A
SCALE 1 : 1

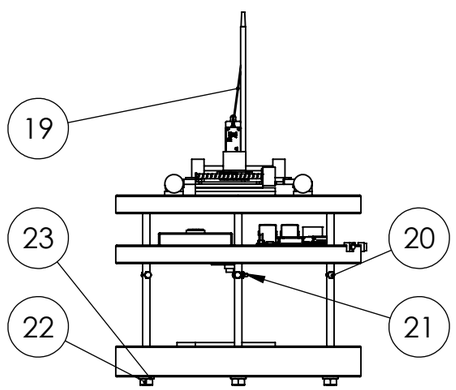




Layer 1



Layer 2



Layer 3

ITEM NO.	PART NUMBER	DESCRIPTION	MATERIAL	QTY.
1	layer 1	Bottom input layer	Oak	1
2	Layer 2	Midle processing layer	Beech	1
3	main rod	Support pillars	Mild Steel	4
4	long acrylic tab	Multi-purpose for elevation	Acrylic	6
5	Pi-Camera 5M	Standard part		1
6	breadboard	Standard part		1
7	Raspberry Pi 3B+	Standard part		1
8	layer 3	Top Printing Layer	Beech	1
9	short acrylic tab	platform stepper motors	Acrylic	4
10	printing block1	1 dot for orientation	Mahogany	1
11	printing block2	2 dot for orientation	Mahogany	1
12	printing block3	3 dot for orientation	Mahogany	1
13	printing block4	4 dot for orientation	Mahogany	1
14	stepper motor connector	Further explored	Polypropylene	4
15	printing rods	short(6.5cm) 4mm rods	Mild Steel	2
16	servo platform	guided by printing rods	Mahogany	1
17	SG90 - Micro Servo 9g - Tower Pro	Standard part		1
18	Pen	Standard ball point pen		1
19	string		Nylon	1
20	BS EN 24014 - M2.5 x 16 x 11-C	standard M2.5 bolt		4
21	BS EN 24032 - M2.5 - W - N	M2.5 Nut		4
22	BS EN 24032 - M6 - W - N	M6 Nut		4
23	Bright washer BS 4320 - M6 (Form A)	M6 Washer		4
24	DRV 8825	Stepper motor driver		2
25	Stepper Motor	Standard part		4
26	stepper motor connector 2	second connecting part	Acrylic	4
27	Wires	for circuitry figure 23		47
28	light	Feedback routine		2
29	button	interaction initiation		1

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS
SURFACE FINISH:
TOLERANCES:
LINEAR:
±0.1mm
ANGULAR:
± 15°

Total physical parts | Total parts (CAD model)

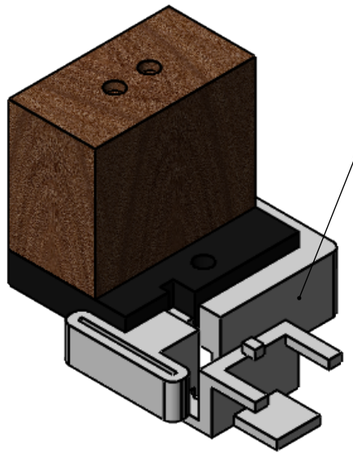
TITLE:
General Assembly

Total Cost

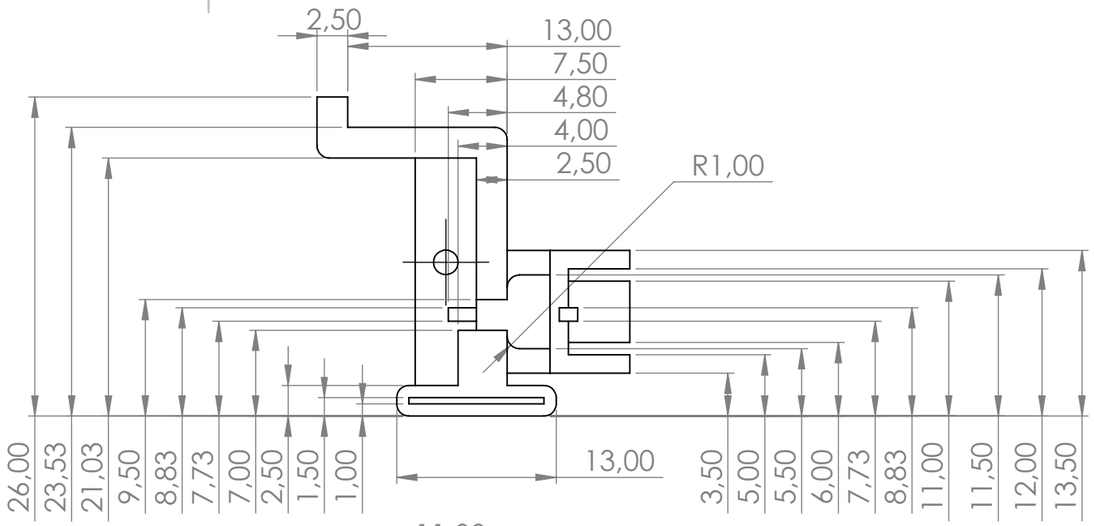
Drawing type Assembly Drawing | A3

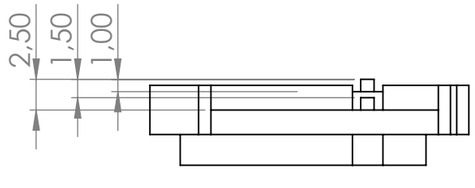
WEIGHT:

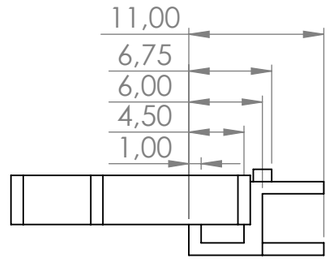
SCALE:1:5

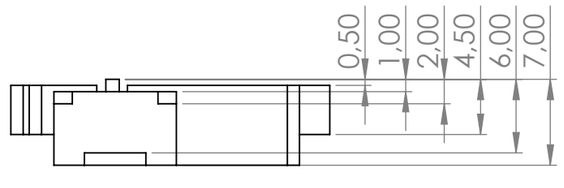


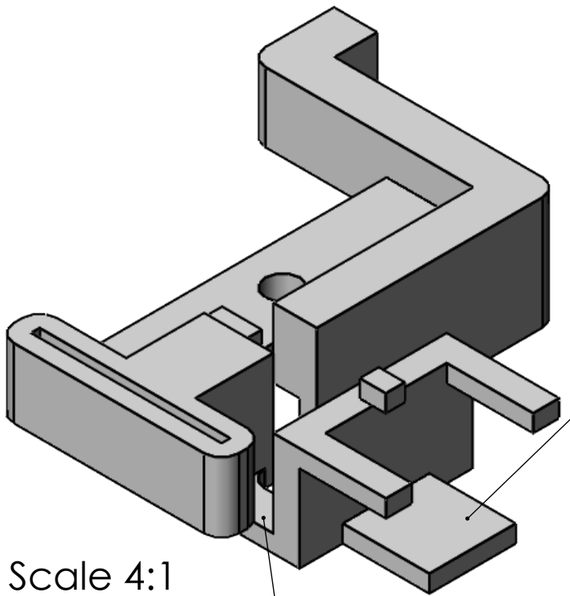
Part 14 from General Assembly drawing





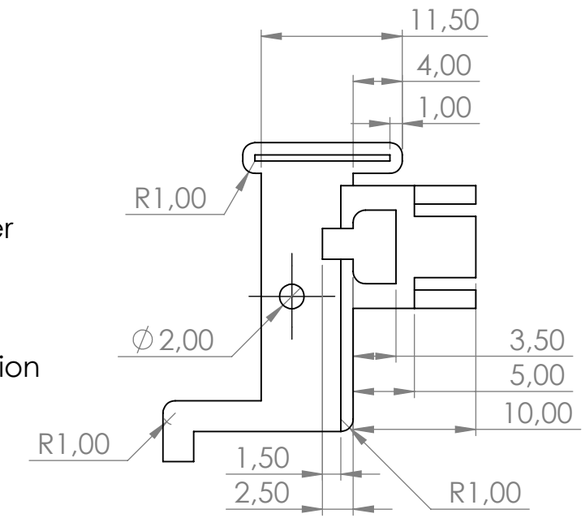






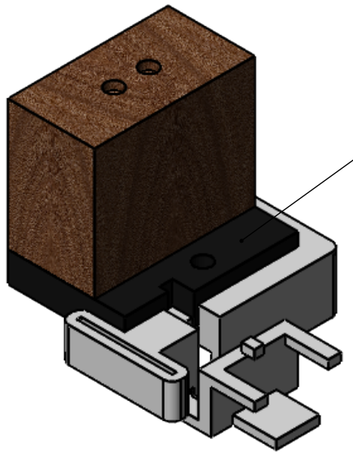
Scale 4:1

Front latches onto stepper motor lead screw, the applied lubrication helps it slide along the screw instead of rotating with it translating rotational motion to horizontal movement

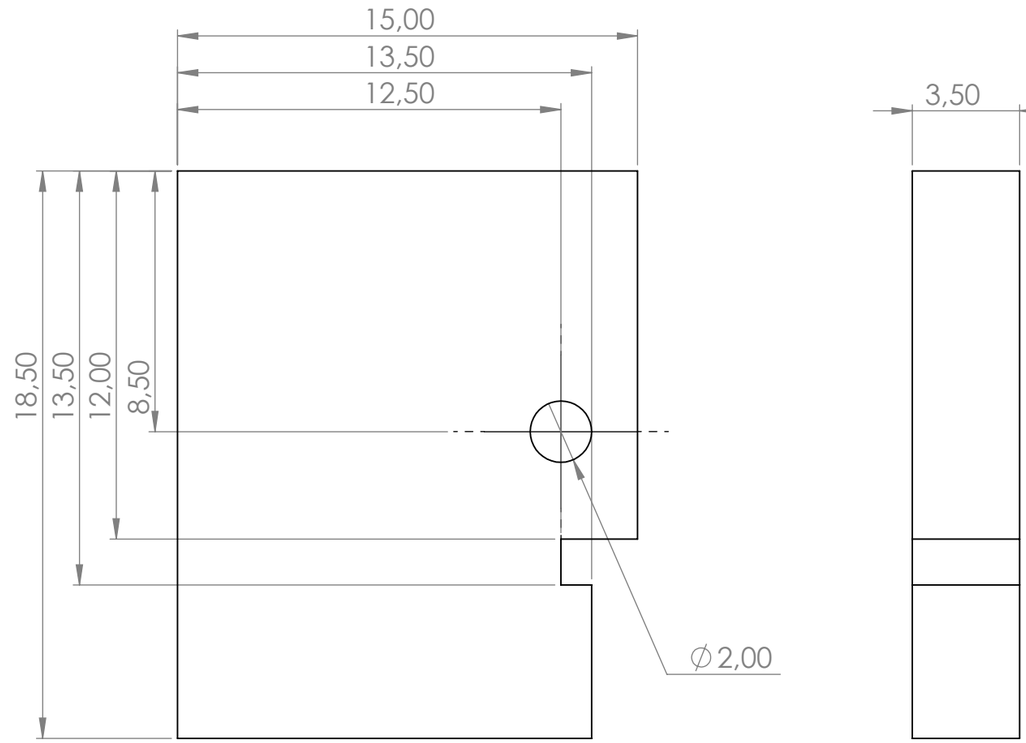


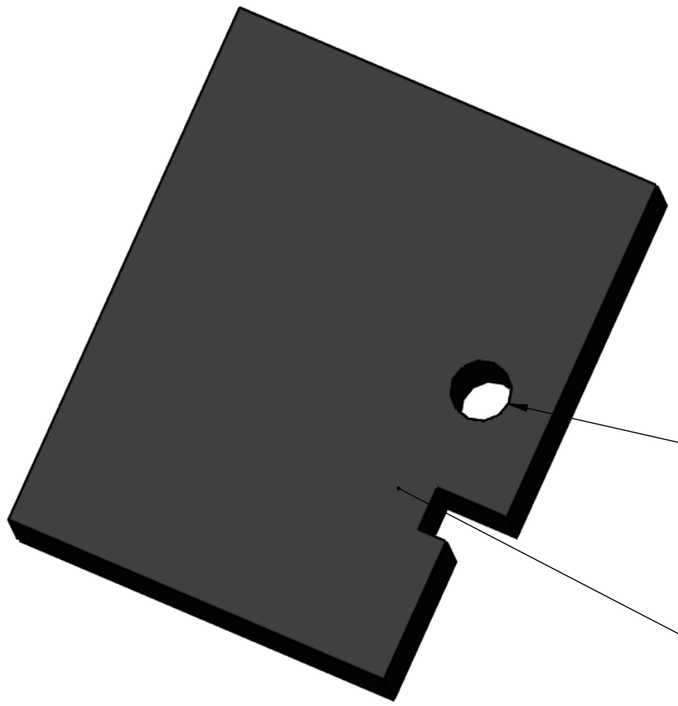
The attachment is connected by a thin flexible ledge which can flex upwards or downwards, allowing more flexibility for moving parts within the system

UNLESS OTHERWISE SPECIFIED: DIMENSIONS ARE IN MILLIMETERS SURFACE FINISH: TOLERANCES: LINEAR: ±0.1mm ANGULAR: ± 15°	TITLE: Stepper motor connector	
	Document Type: Part Drawing	
MATERIAL: Polypropylene	SCALE:2:1	A3 SHEET 1 OF 3



Part 26 from General Assembly drawing

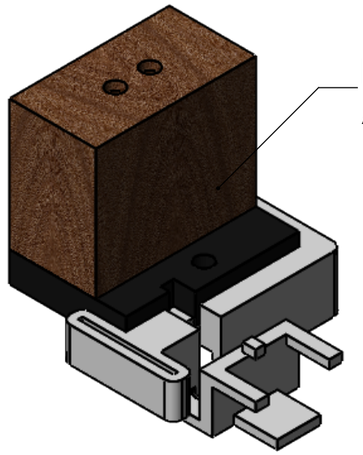




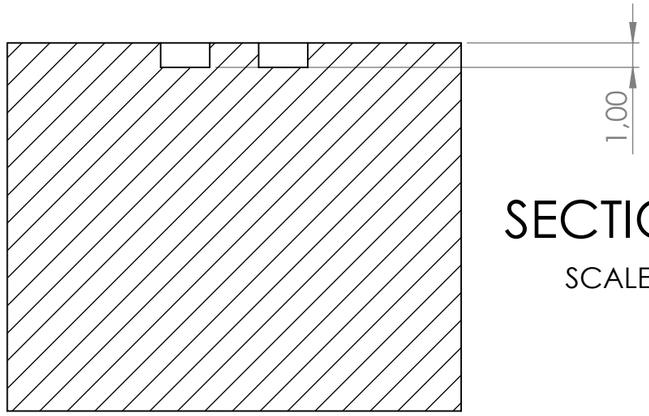
M2 screw wlil also aid attaching this part to "Connector 1"

Second connecting part which attaches onto "stepper motor connnector 1 shown in top left corner.

UNLESS OTHERWISE SPECIFIED: DIMENSIONS ARE IN MILLIMETERS SURFACE FINISH: TOLERANCES: LINEAR: ±0.1mm ANGULAR: ± 15°	TITLE: Stepper motor connector 2	
	Document Type: Part Drawing	
MATERIAL: Acrylic	SCALE:5:1	A3 SHEET 2 OF 3

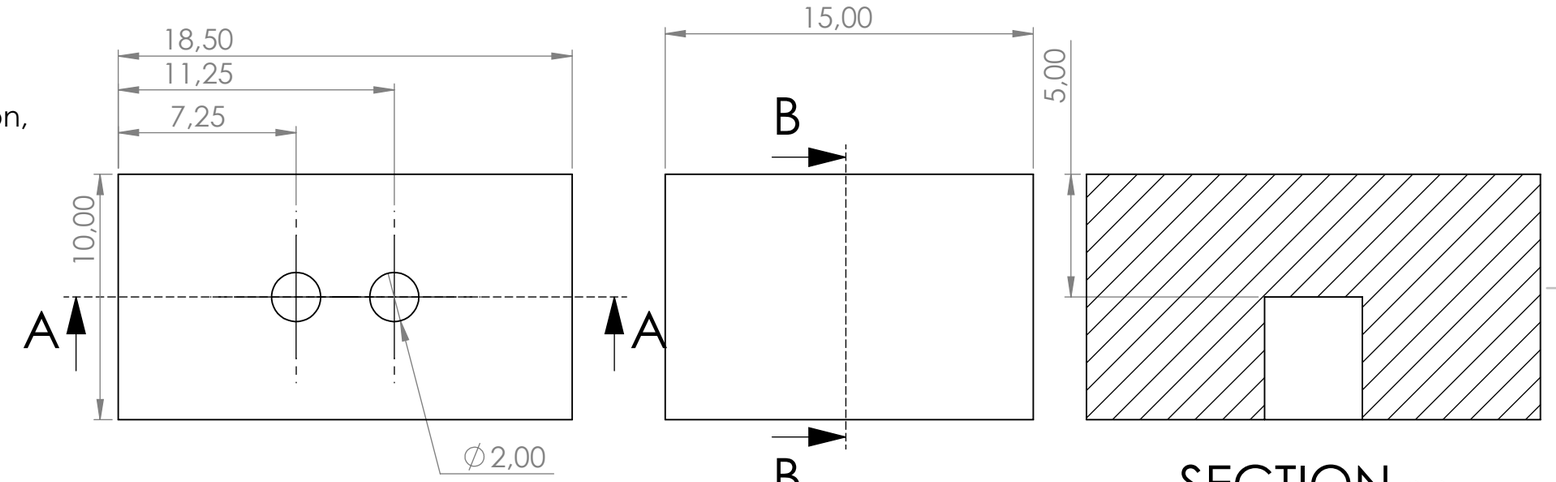


Part 10 from General Assembly drawing



SECTION A-A

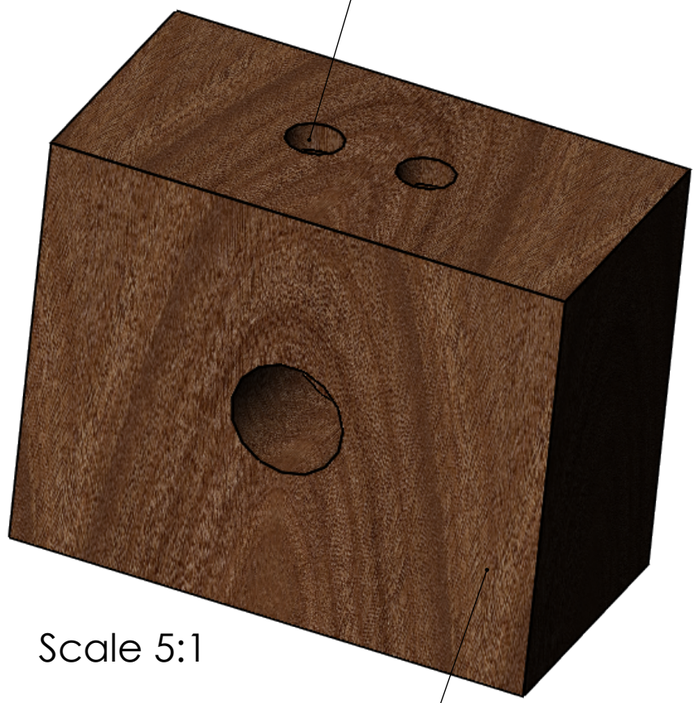
SCALE 4 : 1



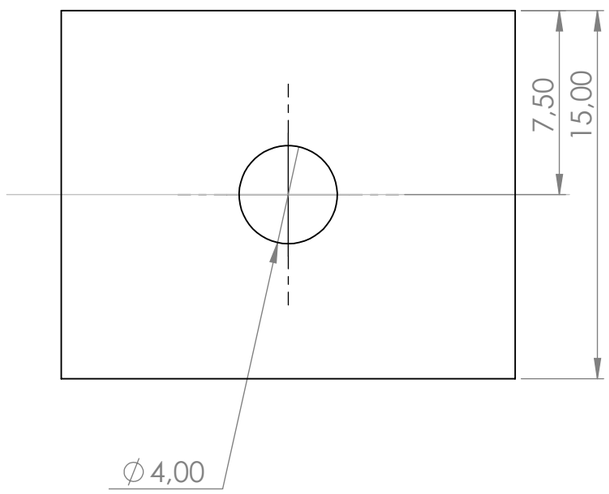
SECTION B-B

SCALE 4 : 1

Indicators for orientation, easier to identify and assemble



Scale 5:1



Attached either end of "printing rods" to allow accurate X-Y movement.

UNLESS OTHERWISE SPECIFIED: DIMENSIONS ARE IN MILLIMETERS SURFACE FINISH: TOLERANCES: LINEAR: ±0.1mm ANGULAR: ± 15°	TITLE: Printing Block 2	
	Document Type: Part Drawing	
MATERIAL: Mahogany	SCALE:4:1	A3 SHEET 3 OF 3