PROJECT PROPOSAL

GhostWriter

Team 21

Lucas Bruder
Matt Mercedes
Jake Stephens

February 12th 2016

Table of Contents

Project Description

Our project, called GhostWriter, is an electromechanical system to generate written content by use of existing drawing tools. The goal of our project is to create a system capable of outputting physical drawings from a given digital input. This system would allow a user to submit an image file to our system, and have a machine reproduce the image onto paper with a user provided drawing utensil such as a pen, pencil, or marker. Furthermore, we also plan to have our system receive physical input via an attached camera. This addition will allow our system to interact with potential human stimuli, and extend the possible use cases of the system beyond existing solutions such as playing games, copying handwriting, detecting errors in reproduction, etc.

Requirements

Explicit Requirements

- System shall be able to recreate pictures and write out text via a provided drawing utensil
- System shall be able to detect drawing surface and size, so as to produce an accurate reproduction given the space provided
- System shall be able to receive feedback via camera/light and act on it

Stretch Goals

- System should be able to communicate wirelessly between motors and microcontroller.
- System should be able to analyze and respond to human input, such as playing the game Tic-Tac-Toe
- System should be able to draw on vertical surfaces
- System should have a mobile or web application that allows transfer or upload of drawing date and notify user of system status

Coolness Requirements

- Recreate an image in a reasonable amount of time for a bystander to stay and witness
- Have a slim form factor that allows for travel between surfaces and locations

Architecture

As detailed in Figure 1, a user interacts with the system by sending a file to a processing platform. The platform will then convert the file to a format understood by the motor controllers to draw. The motor controllers will command the motors to draw on the drawing medium (paper, whiteboard, chalk, etc.). When the motors are done drawing, the processing platform will notify the user it's done and the user can now get their drawing, a perfectly drawn smiley face.

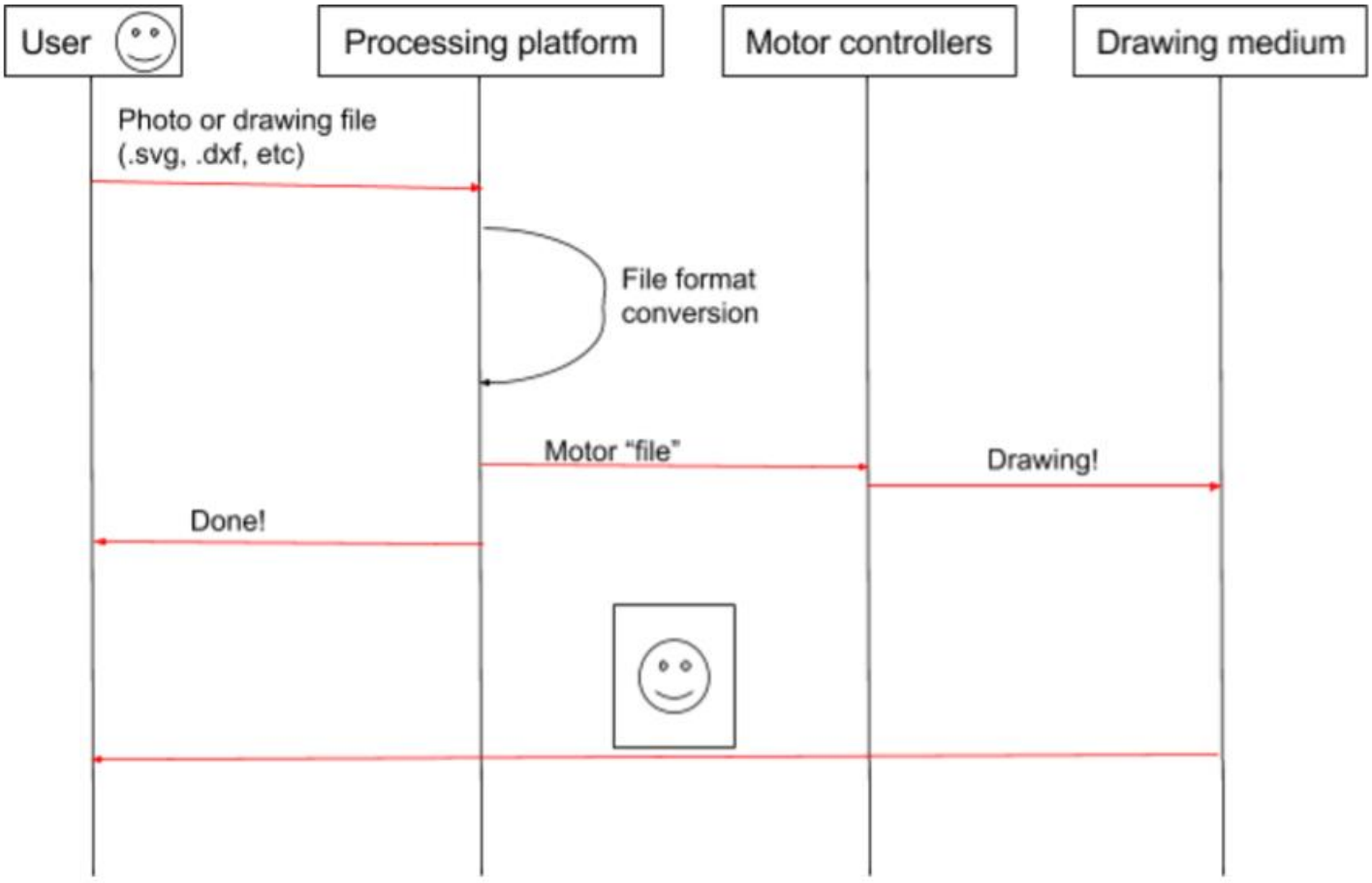

Figure 1

Design Trade-offs

Frame

Choice 1

The majority of x-y plotters use a standard square frame as depicted in Figure 2. The benefits of this frame include simplicity, ruggedness, and ease of use. Each motor is only responsible for one axis and that greatly simplifies reliability and trajectory control. However, the frame is a fixed size and is big and heavy. If one wanted to increase the size of their workspace, they would have to buy several new components including extruded aluminum, belts, and several other key components to the operation of the plotter.

Choice 2

Our second choice for frame design is a design similar to the ZarPlotter[1]; We have sketched out a picture of what the frame looks like in Figure 3. This design features four motors in each corner (pictured in black) with strings running to a tool in the workspace responsible for drawing. This design is more complicated, could be more prone to reliability issues, and complicates the trajectory control. However, this design can be used on any surface whether it be horizontal or vertical and can also be used on workspaces with different areas. Also, when drawing, it looks much cooler for a bystander.

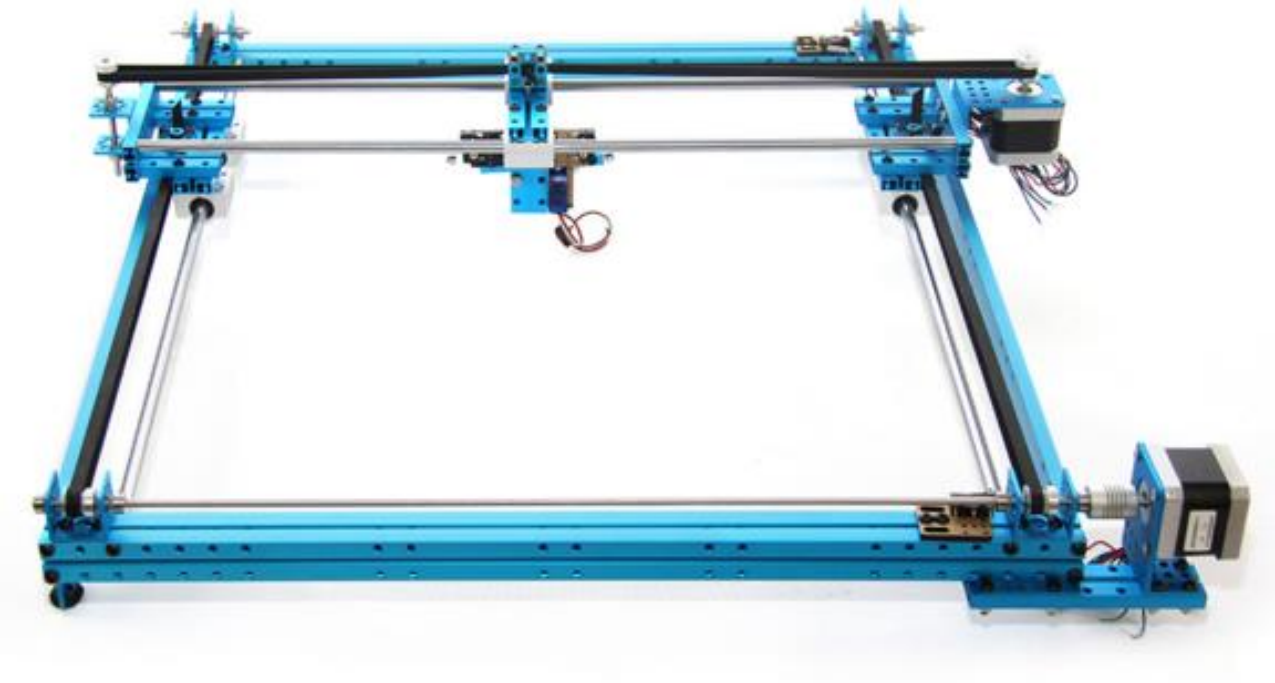

Figure 2

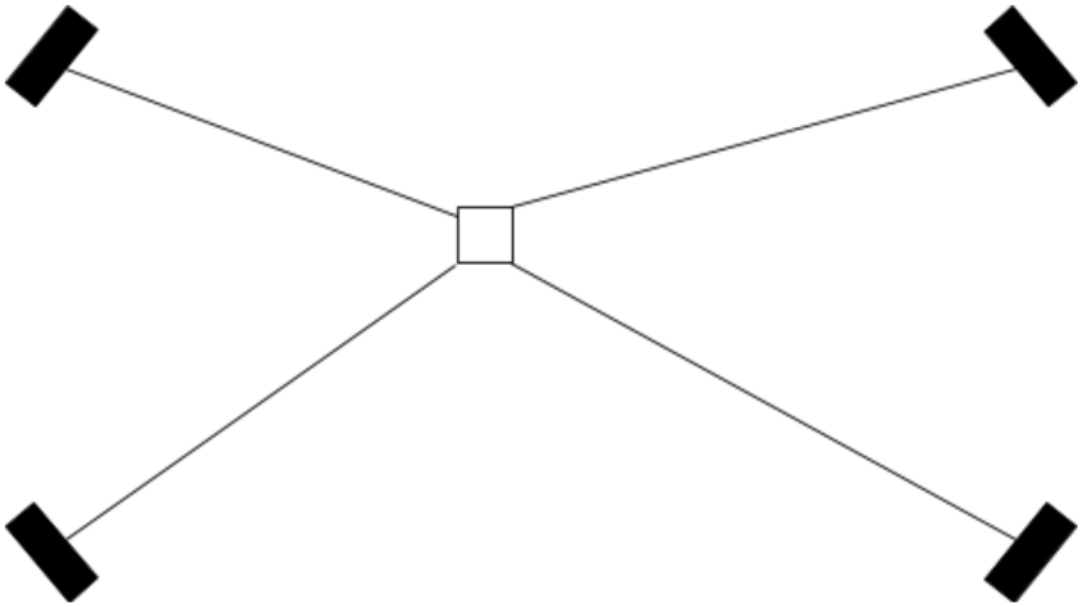

Figure 3

Decision

Since one of our design and coolness requirements is being able to use the product on a workspace of any size, we have decided to go with choice 2 for the frame. However, if this frame layout is too complicated, our backup plan is choice 1, which utilizes many of the same components found in choice 2.

Processing Platform

There are various processing platforms available that would have enough power for what we need. We decided to narrow it down to two processing platforms to choose between.

Choice 1

Our first choice of processing platform is the Raspberry Pi. There are several different models to choose from ranging from $5 to $30, single core to quad core. There is a huge community support and finding example code and libraries would be extremely easy.

Choice 2

Our second choice for a suitable processing platform is the Beaglebone Black with a price point at around $50. The capabilities of the Beaglebone Black are much less than the capabilities of the Raspberry Pi with a lower clock speed and lower amount of GPIO pins.

Decision

Because of the bigger user support, more options, and cheaper price point, we have decided to use a Raspberry Pi as our main processing platform.

Other Decisions

There are a few other design decisions that need to be made including stepper motors, stepper motor drivers, microprocessor, and method of attachment. These design decisions will be evaluated more in depth as the semester progresses and additional information becomes available.

System Overview

Description

Overall System Design

Our overall system depiction is pictured in Figure 4. The Raspberry Pi will be the main processing unit, performing the translation from one file format to a format that the microcontroller can understand. Once this new format is computed, it will send the data to the main microcontroller which parses the received data and is responsible for controlling the four separate stepper motor drivers, which in turn control the stepper motors.

Stepper Motor Driver Subsystem

Our stepper motor driver subsystem is depicted in Figure 5. We will most likely be using the A4988 chip, but the interface for a stepper motor is pretty standard across motor controllers. In the top right of the figure, you can see that V_{mot} is attached to the motor power supply. This power supply must be able to handle relatively high amperage, since stepper motors can draw at or more than 2 amps. Next, there are ports 2B, 2A, 1A, and 1B, which lead to the stepper motor itself. These lines are responsible for switching the motor phases on and off to ensure the motor steps the correct number of times, and they must be hooked up in specific motor to ensure it functions properly. V_{dd} is a line used to determine what voltage logic level to use. Most importantly, we have the step and direction line. The step line is pulsed to let the controller take a step, and the direction line is usually set high for one direction and set low to go the opposite direction.

Camera Subsystem

The camera subsystem is less complex and therefore there is no need of a diagram explaining the connection, since it will most likely be through a simple USB cord. There aren't any current x-y plotters on the market that provide feedback, so we plan to use a camera as feedback to where the plotter lies in its workspace. Although getting the camera functioning is not our main objective, we plan on using it to provide feedback as to where the plotter is on the x-y plane as a simple camera stream. If time allows, this subsystem would develop into an application utilizing computer vision that would allow interactive applications like playing the computer against tic-tac-toe or solving and finishing mazes.

Depiction

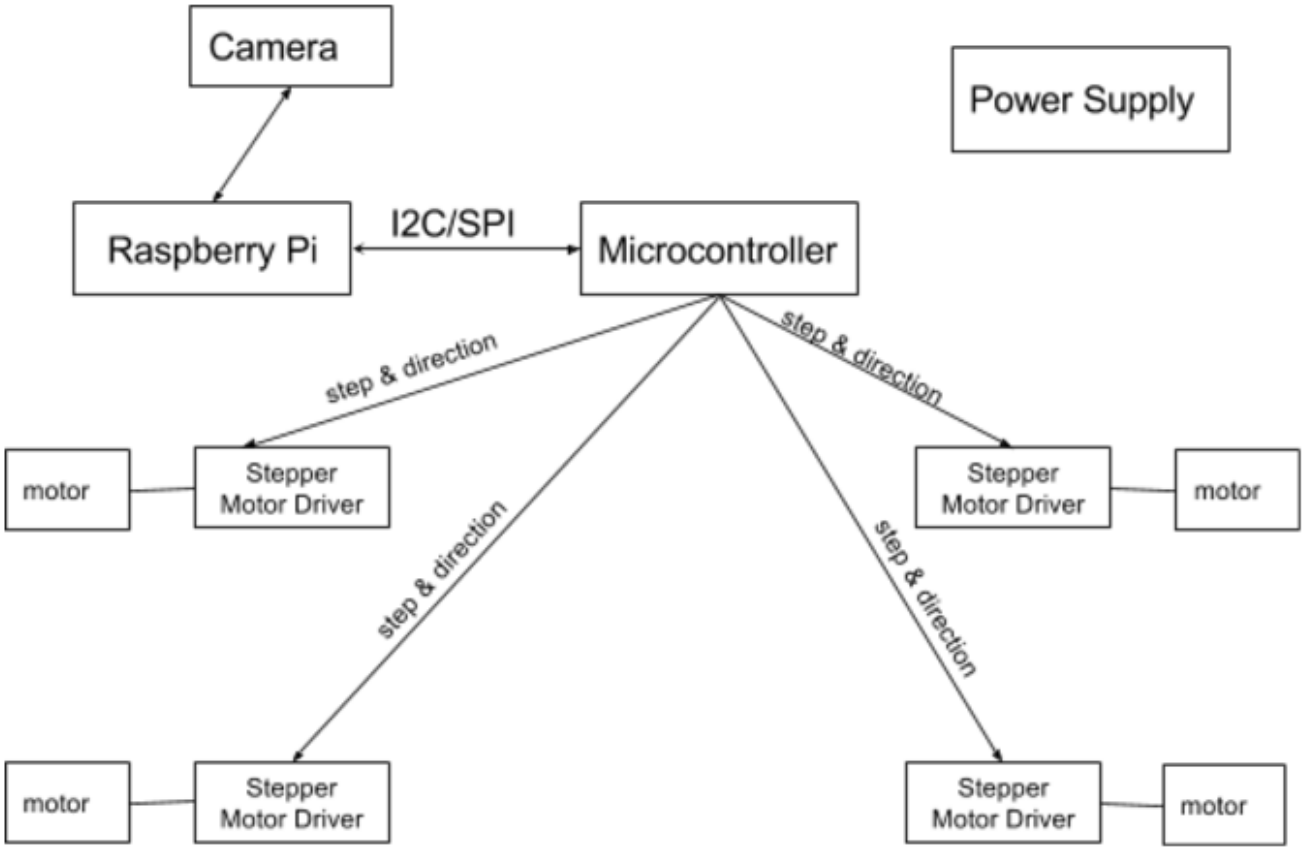

Figure 4

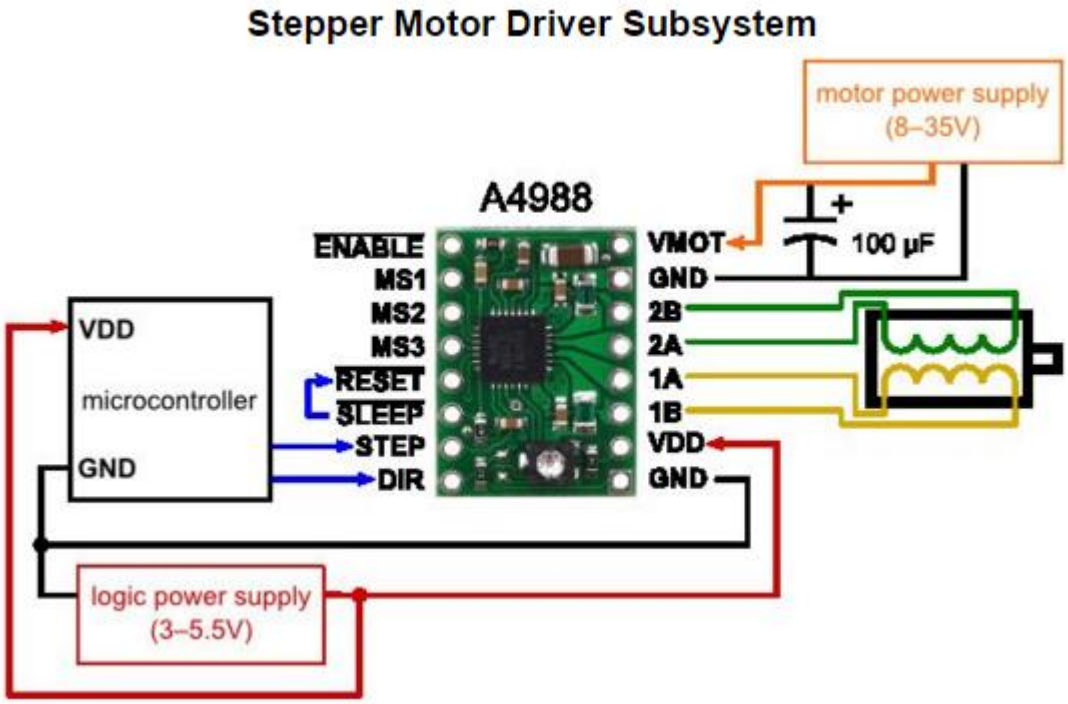
Stepper Motor Driver Subsystem
motor power supply
(8–35V)
A4988
100 µF
ENABLE
MS1
MS2
MS3
RESET
SLEEP
STEP
DIR
VMOT
GND
2B
2A
1A
1B
VDD
GND
VDD
microcontroller
GND
logic power supply
(3–5.5V)

Figure 5

Project Management

Schedule

- We plan to have a group meetings three times a week
- Meetings are scheduled after class on Mondays and Tuesdays from 1:30pm to 3:00pm
- Additional meeting on Saturday from 2:00pm to 5:00pm
- Schedule is outlined in Figure 6

Week 1	Feb 15	Feb 22	Feb 29	March 7
Goals	· Create Website · Begin to work on acquiring necessary materials · Begin formalizing an electrical design plan	· Set up stepper motors and pulley to achieve basic 2D movement · Create a simulation to plan out plotting	· Assemble all parts so that we have all the hardware setup the way we need it · Finalize testing code for basic drawing functions · Begin planning for camera usage	Spring Break

Week 2	March 14	March 21	March 28	April 4
Goals	· Continue with finalization of hardware · Attach camera and confirm functionality · Begin testing code with actual hardware	· Finalize basic components · System able to create a drawing · Start adding additional features such as camera and more human interaction	· Continue debugging any issues that come up and adding additional features · Finish website up to checkpoint 2 requirements	· Add features and fix any issues that arise

Week 3	April 11	April 18	April 23	April 30
Goals	· Add features and fix any issues	· Add features and fix any issues	· Add features and fix any issues · Finalize Website and video	· Prepare for final demo · Begin Writing Final Report

Figure 6

Responsibilities

Jake Stephens

Will be responsible for working on designing the circuit composing the drawing hand that holds the tool to impress the design onto the drawing surface and implementing the ability to raise and lower the drawing utensil on command.

Matt Mercedes

Responsible for writing the code that takes a digital image format and translating it into a set of instructions for the stepper motors to follow. Will also be responsible for both creating and maintaining an updated version of our website that depicts our project's progress.

Lucas Bruder

Will be responsible for assembling the four motors and accompanying pulley systems along with their respective drivers. Will also be the main person for maintaining the overall hardware design of our project.

Budget

Raspberry Pi	$5
Microcontroller	$5
Stepper Motor Drivers (4)	$20
Stepper Motors (4)	$80
Physical Equipment (pulleys, string, etc.)	$100
4’ x 8’ Plywood	$15
Small Digital Camera	$30
5v power supply (10A minimum)	$30
XY Plotter Kit (If necessary)	$300
Total	$585

Risk Management

In terms of risk based on the schedule we have assigned enough time so that we have plenty of time to debug the system. Based on our estimates, we should be done with what we expect to accomplish by the 4th or 5th week. After that we will debug and provide ourselves with plenty of time to fix any important issues or add additional features. If we are unable to get the system working in the way we intend, then we are prepared to purchase an x-y plotter for a mechanical base and then implement our additional features on top of it to create our new product. Thus, we will not have to worry about not being able to complete the basic functionality of our project and instead focus on the additional features that similar products have not yet created.

Related Work

DryPlt

DryPlt[2] mostly focused on integrating with common whiteboards. DryPlt tried to use several different techniques to locate the position of the pen on the whiteboard. However, it was never confirmed if it successfully implemented the ability to recreate images on a white board. Additionally, there is no input from the physical world during, or after the recreating of a given image.

XY Plotter

The XY Plotter[3] only has basic fixed-size drawing functionality. One cannot change the size of the area that will be drawn on and is restricted to the given layout of the plotter. In addition, there is no input from the physical word or feedback provided to the system. There is also no ability for the plotter to continue from a previously left off position or respond to potential interruptions during the drawing process.

ZarScribe

The ZarScribe[4] is fixed in place and is unable to change the size of the area that it is working on. Additionally, there is no input or feedback from its surroundings just like many of the other plotters. Also, the controller of this plotter is big and clunky and must be wired up to all the other parts of the system. It is also very new and thus does not have much documentation so there is no info on how it actually works or what systems are in place to allow it to recreate a given image. In addition, since it is very new it is not currently available in the consumer market for purchase, nor is there a date for when it will become available.

Basic Plotters

Basic plotters do not have many features and are fixed to printing the images they are given with no further input. They commonly function as simple printers and thus do not provide too much functionality beyond basic reproduction functions.

References

[1] http://www.zarplotter.com/

[2] https://www.ece.cmu.edu/~ece549/spring15/team15/website

[3] http://www.makeblock.cc/xy-plotter-robot-kit/

[4] http://www.zarplotter.com/