

Komplexe Leistung zum Thema: Bau und Erprobung eines 2D-Plotters

Mattis M

8. Juni 2016

Fach: Informatik
voraussichtlich Eingereicht am: 4. April 2016

Inhaltsverzeichnis

| | |
|--|-----------|
| 1 Einleitung | 3 |
| 1.1 Begriffsdefinitionen | 3 |
| 1.2 Zielstellung | 4 |
| 1.3 Hinweise | 4 |
| 2 Aufbau des Plotters | 5 |
| 2.1 Aufbau des ersten Plotters | 5 |
| 2.1.1 Aufbau des Antriebs und des Rahmens | 5 |
| 2.1.2 Aufbau der Elektronik | 7 |
| 2.2 Aufbau des zweiten Plotters (der auf einem 3D-Drucker basiert) | 7 |
| 2.2.1 Mechanischer Aufbau | 8 |
| 3 Erläuterungen zur Software | 8 |
| 3.1 Software des Mikrocontrollers | 8 |
| 3.1.1 Software des ersten Plotters | 8 |
| 3.1.2 Software des zweiten Plotters | 10 |
| 3.2 Steuersoftware des zur Steuerung verwendeten Computers | 11 |
| 4 Weitere Einsatzmöglichkeiten meines und anderer Plotter | 12 |
| 4.1 Einsatzmöglichkeiten meiner Plotter | 12 |
| 4.1.1 Einsatzmöglichkeiten des ersten Plotters | 12 |
| 4.1.2 Einsatzmöglichkeiten des zweiten Plotters | 12 |
| 4.2 Einsatzmöglichkeiten von Plottern (allgemein) | 12 |
| 5 Anhang | 13 |
| 5.1 Bilder | 13 |
| 5.2 Quellenverzeichnis | 22 |

1 Einleitung

Seit dem in den 1990er-Jahren Drucker und Großformatdrucker die Zeichenplotter ersetzt haben, sind diese heute kaum noch in aktiver Verwendung (mehr dazu in Abschnitt 4). Trotzdem sind Plotter nach wie vor bedeutend, da sie die Grundlage für viele Maschinen bilden und außerdem die Bedeutung und den Unterschied zwischen Vektor- und Pixelgrafiken sehr gut veranschaulichen können. Im Gegensatz zu Laser- oder Tintenstahldruckern können sie Vektorgrafiken darstellen, ohne diese vorher in eine Rastergrafik umgewandelt haben zu müssen. [1]

Die Idee zum Bau eines Plotters kam mir, als ich einen alten defekten Scanner zerlegte. Ich überlegte, was ich mit dem noch funktionsfähigen Antrieb, der wie für Scanner üblich aus einem Schrittmotor mit einem Getriebe und einem Zahnräumen bestand machen könnte.

Im Internet findet man zahlreiche Projekte und Projektideen, die erklären, wie man einen Plotter selbst bauen kann, allerdings basiert mein Plotter im wesentlichen auf meinen eigenen Ideen und Überlegungen.

1.1 Begriffsdefinitionen

Plotter: „Ein Plotter (auch Kurvenschreiber) ist ein Ausgabegerät, das Vektorgrafiken darstellt [...], ohne sie vorher in eine Rastergrafik umzurechnen.“ [1]

Vektorgrafik: „Eine Vektorgrafik ist eine Computergrafik, die aus grafischen Primitiven wie Linien, Kreisen, Polygonen oder allgemeinen Kurven (Splines) zusammengesetzt ist.“ [2]

Raster-/Pixelgrafik: Eine Raster- bzw. Pixelgrafik ist eine Computergrafik, „die aus einer rasterförmigen Anordnung von Pixeln (Bildpunkten) besteht“. Das Skalieren einer Rastergrafik führt (im Gegensatz zum skalieren einer Vektorgrafik) zu einem Qualitätsverlust. [3]

Schrittmotor: „Ein Schrittmotor ist ein Synchronmotor, bei dem der Rotor (drehbares Motorteil mit Welle) durch ein gesteuertes, schrittweise rotierendes, elektromagnetisches Feld der Statorspulen (Stator = nicht drehbarer Motorteil) um einen minimalen Winkel (Schritt) oder sein Vielfaches gedreht werden kann.“ [4]

H-Brücke: „Ein Vierquadrantensteller besteht aus einer elektronischen H-Brückenstellung aus vier Halbleiterschaltern, meist aus Transistoren, welche eine Gleichspannung in eine Wechselspannung variabler Frequenz und variabler Pulsbreite umwandeln kann.“ [5]

RepRap: „Der RepRap ist ein 3D-Drucker, der für das Rapid Prototyping und Rapid Manufacturing verwendet werden kann und alle Kunststoffteile seiner Bauteile auch selbst herstellen kann (Autoreplikation).“ [9]

StepStick: Ein StepStick ist ein Schrittmotortreiber (Allegro A4983/A4988) auf einer Platine, welche zur einfachen Verwendung in beispielsweise Plottern, 3D-Druckern (meist) mit Steckkontakten im Standard-Rastermaß () versehen ist. Außerdem sind die Konstruktionsdetails (Platinendaten, Einzelteillisten, ...) unter GPL-Lizenz veröffentlicht und somit frei nutzbar. [vgl. 8]

Arduino (Plattform): „Arduino (seit März 2015 auch Genuino) ist eine aus Soft- und Hardware bestehende Physical-Computing-Plattform. Beide Komponenten sind im Sinne von Open Source quelloffen. Die Hardware besteht aus einem einfachen E/A-Board mit einem Mikrocontroller und analogen und digitalen Ein- und Ausgängen.“ [10]

Mikrocontroller: „Als Mikrocontroller (auch µController, µC, MCU) werden Halbleiterchips bezeichnet, die einen Prozessor und zugleich auch Peripheriefunktionen enthalten.“ [11] Ein Mikrocontroller kann also sowohl Daten verarbeiten als auch mit Elektronik interagieren.

G-CODE: Steuerungssatz für (computergesteuerte) CNC-Maschinen [vgl. 19]

1.2 Zielstellung

Zielstellung dieser komplexen Leistung war es einen zuverlässigen und optisch ansprechenden 2D- Plotter zu bauen, der sowohl einfache geometrische Formen wie Dreiecke, Rechtecke und Kreise als auch gängige mathematische Funktionen in Koordinatensystem darstellen kann.

1.3 Hinweise

Der Quellcode dieses Projekts sowie dieses Dokument befinden sich außerdem in meinem Github-Repository.

2 Aufbau des Plotters

Da ich beim Darstellen von Kreisen und Ellipsen auf komplexe Probleme stieß, die ich nicht lösen konnte, habe ich mich nach Absprache mit meinem Betreuer dazu entschlossen, diese Darstellungen mit meinem 3D-Drucker zu realisieren. Dazu habe ich an der X-Achse den Extruder (Der Teil des 3D-Druckers der den flüssigen Kunststoff extrudiert) durch eine Halterung für einen Stift ersetzt habe. Trotzdem möchte ich auf den mechanischen und elektronischen Aufbau meines ersten Plotters eingehen, um dabei auch die Probleme des mechanisches Aufbaus die letztendlich zu dem Versagen dieses Geräts führten, zu erläutern.

2.1 Aufbau des ersten Plotters

2.1.1 Aufbau des Antriebs und des Rahmens

Um einen ersten Funktionstest durchzuführen, habe ich den Schrittmotor mithilfe eines Mikrocontrollers und einer H-Brücke (Abb. 5.1) angesteuert, um so einen Stift der an dem Schlitten befestigt ist, linear zu bewegen.

Da dieser erste Test erfolgreich war, habe ich dann beschlossen einen Plotter zu bauen. Dafür habe ich zuerst einen 2. teilweise defekten Scanner, der in meiner Schule wegen dieser Defekte aussortiert wurde, zerlegt, um so den Antrieb für die (Y-Achse) zu bekommen. Weil der erste Funktionstest der Schrittmotorsteuerung mit einer H-Brücke erfolgreich war und ich zu diesem Zeitpunkt keinen Grund zur Annahme hatte, dass dies nicht funktionieren würde oder zu Problemen führen könnte, habe ich zwei neue, leistungsfähigere H-Brücken (siehe H-Brücke.jpg) gekauft. Mit diesen habe ich die erste funktionierende Version des Plotters gebaut (wie 5.8, lediglich andere Elektronik).

Zu diesem Zeitpunkt konnte der Plotter bereits problemlos Geraden und Diagonalen zeichnen. Allerdings traten auch die ersten größeren Probleme auf, die ich vorher nicht genau durchdacht hatte: zum einen benötigten die Schrittmotoren mit ihren Getrieben unterschiedlich viele Schritte um die selbe Strecke zurückzulegen, zum anderen wurden sie bei längerem Betrieb so heiß, dass ich Bedenken bekam, ob die Steuerung so günstig konstruiert war.

Die hohen Temperaturen der H-Brücken stellten sich nach lesen des Datenblatts eine „OVERTEMPERATURE PROTECTION“[6, S.1], also einen Überhitzungsschutz besitzen und wie in Abbildung 5.1 ersichtlich einen recht großen Kühlkörper besitzen als unproblematisch dar. Die hohe Temperatur der Motoren, war allerdings doch recht ungünstig, denn die Datenblätter vergleichbarer Schrittmotoren beschreiben „Operating Temp -20°C to +40°C“[6, S.1] , also eine Betriebstemperatur von maximal 40°C. Nach kurzer Internetrecherche wurde mir klar, dass die hohen Temperaturen mit nicht geregelten also zu hohen Strömen zusammenhängen.

Wegen dieser zu hohen, ungeregelten Ströme entschloss ich mich, die bei reprap-3D-Druckern üblichen StepSticks (vgl. Abb. 5.7) [<http://reprap.org/wiki/StepStick>] als Schrittmotortreiber zu verwenden, welche dieses Problem lösten. Außerdem ermöglichen diese eine einfachere und zuverlässiger Ansteuerung sowie weitere

Funktionen wie einen „Schlafmodus“ (mehr dazu in 3.1).

Nachdem ich diese neuen StepSticks erfolgreich eingesetzt hatte konnte ich die Motoren zwar zuverlässig ansteuern, bekam aber für mich unerklärbare Fehler beim Zeichnen von Kreisen bzw. Ellipsen, trotzdem baute ich einen neuen Rahmen der der Anforderung „optisch ansprechend“ genügen sollte und außerdem eventuelle Fehler, die aus einer Instabilität des Rahmens hervorgehen, beheben zu können.

Der Rahmen besteht aus einer hölzernen Grundplatte (L:500mm, B:400mm, H:10mm), um eine möglichst große Stabilität zu erzeugen. Auf dieser sind zwei parallele, glatte, geschmierte Metallachsen ($\varnothing 8\text{mm}$) angebracht, die eine möglichst zuverlässig lineare Bewegung der Y-Achse ohne zu große Ausreißer ermöglichen. Des weiteren befinden sich auf der Grundplatte eine Gabellichtschranke (vgl. Abb. 5.4) und der Schrittmotor (Abb. 5.2) mit Getriebe und Zahnriemen die zur Kalibrierung bzw. Bewegung des Y-Schlittens an welchem die X-Achse befestigt ist, dienen. Außerdem befindet sich auf der Grundplatte der Mikrocontroller mit Schrittmotortreibern (siehe 5.7) sowie die Anschlüsse für die Stromversorgung.

Auf dem X-Schlitten der von den beiden Y-Achsen und dem Y-Zahnriemen in der gewünschten Position gehalten wird, befindet sich der Antrieb der X-Achse (ein Schrittmotor mit Getriebe und Zahnriemen; vgl. Bild 5.3), eine Gabellichtschranke zur Kalibrierung und ein Schlitten, der von einem Servomotor angehoben werden kann und einer unbeschalteten Spule, die als Gewicht zum Anpressen des Stiftes auf das Blatt dient (vgl. 5.11). Am Arm der Spule befindet sich eine Klemme, in der verschiedene gängige Blei- und Filzstifte sowie Kugelschreiber(-minen) und andere Werkzeuge mit zylindrischem oder prismatischem Griff mit einem Durchmesser von maximal 7,5mm innerhalb kürzester Zeit und mit einem Handgriff befestigt werden können (siehe 5.11 rechts).

Auch diese fast finale Konstruktion hatte immer noch einige Probleme. So führte zum Beispiel eine nicht ausreichend exakte Parallelität der Achsen und Achshalterungen zu einem Verkanten des Y-Schlittens, was darin resultierte, dass der Stift nicht in Y-Richtung bewegt werden konnte und manuell in die richtige (nicht verkantete) Position bewegt werden musste. Deswegen musste ich die linke Y-Achse durch eine dünnerne Achse ($\varnothing 5\text{mm}$) ersetzen. Außerdem lagen die Kabel, die zum Y-Schlitten führen und den Servomotor, den X-Schrittmotor sowie die X-Gabellichtschranke steuern und mit Strom versorgen noch lose neben dem Plotter (siehe 5.9). Da ich mittlerweile Besitzer eines 3D-Druckers war, druckte ich eine Kabelkette, um so die Kabel geordnet und wie gefordert „optisch ansprechend“ zu befestigen. Da ich im Internet zwar gute 3D-Modelle von 3d-druckbaren Kabelketten fand, diese allerdings zu große Reibung aufwiesen, als dass sie von den schon ohnehin stark belasteten Y-Schrittmotor bewegt werden konnten, musste ich diese leicht modifizieren.

Trotz all dieser Verbesserungen und ergebnisloser stundenlanger Fehlersuche und Überlegungen nach möglichen Quellen für die unerklärlichen Fehler beim Zeichnen von Kreisen bzw. Ellipsen konnten diese nicht behoben werden. So entschloss ich mich, jetzt wie bereits in der Einführung erwähnt, dazu meinen 3D-Drucker umzubauen um so einen zuverlässigen funktionierenden Plotter zu

haben. (mehr dazu in 2.2)

2.1.2 Aufbau der Elektronik

Ich habe mich bei der Elektronik für eine Arduino bzw. Genduino UNO R3 Mikrocontroller-Board (vgl. 5.6) entschieden, weil es im Internet zahlreiche Anleitungen und Referenzen gibt, die erklären, wie man einen derartigen Mikrocontroller programmiert und wie dieser mit diverser Elektronik (in meinem Fall Schrittmotortreiber und Gabellichtschranken) interagiert. Außerdem war ich bereits im Besitz eines solchen Mikrocontroller und hatte mit diesem auch schon einige einfachere Projekte realisiert, weswegen ich bereits grundlegende Erfahrungen mit dem Mikrocontroller und dessen Programmierung hatte.

Der Arduino UNO bildet also die Steuer- und Computerinteraktionszentrale der Elektronik, da er sowohl die Schrittmotoren steuert, als auch mit dem Computer (bzw. meiner Steuerungssoftware) über USB. Die verwendete Logikspannung beträgt 5V, der Arduino UNO basiert auf einem ATMEV ATmega328-Mikrocontroller und operiert mit einer Taktfrequenz von 16MHz, hat einen Flash-Speicher (für Programme) der 32 KB groß ist und einen 2KB großen SRAM (Arbeitsspeicher).

Die Schaltung ist recht einfach und besteht im wesentlichen aus dem Mikrocontroller, den Schrittmotortreibern (StepSticks basierend auf dem Allegro A4998-Chip), welche wiederum mit den Schrittmotoren und den 12V-Steckernetzteilen verbunden sind. Außerdem wird Stromversorgung der Schrittmotortreiber durch 2 Kondensatoren (2200 μ F und 3300 μ F) stabilisiert. Die Schrittmotortreiber, Kondensatoren sowie Anschlüsse für die Schrittmotoren und Gabellichtschranken und den Servo befinden sich auf einer Erweiterungsplatine mit Lochraster (siehe 5.7), welche auf dem Mikrocontroller-Board steckt. Im Anhang befinden sich auch der Schaltplan (Abb. 5.5) und eine schematische Darstellung der Schaltung (Abb. 5.6).

Liste aller verwendeten Bauteile (der letzten Version) und deren Bedeutung:

- 1 Arduino bzw. Genduino UNO - Mikrocontroller-Board
- 2 StepSticks (basierend auf Allegro A4998 Chip) - Schrittmotortreiber zur Ansteuerung der Schrittmotoren
- 2 Infrarot - Gabellichtschranken mit Vorwiderständen - Enstoperkennung der X- und Y-Achse
- 2 Kondensatoren - Stabilisierung der Versorgungsspannung bei Spitzenlasten
- 2 Schrittmotoren und 1 Servomotor - Bewegung der X-, Y-, und Z-Achse

2.2 Aufbau des zweiten Plotters (der auf einem 3D-Drucker basiert)

Da wie bereits im letzten Abschnitt erwähnt, der Plotter nicht so funktioniert, wie ich es mir vorgestellt hatte bzw. wie es gefordert war, entschloss ich mich,

meinen 3D-Drucker (Prusa i3 Original) so zu modifizieren, dass man ihn als Plotter verwenden kann. Dafür habe ich den Extruder (siehe 5.13) durch eine Halterung für den Stift ersetzt (siehe 5.10). Dadurch ist es mir möglich, die bereits vorhandenen X-, Y- und Z-Achsen-Antriebe sowie deren Steuerungselektronik (RamboMini motherboard) zu nutzen.

2.2.1 Mechanischer Aufbau

Wie in Abbildung 5.13 bereits zu erkennen ist, besteht der 3D-Drucker aus einem sehr stabilen Metallrahmen, außerdem sind die Schrittmotoren (in 5.13 als „stepper“ bezeichnet) wesentlich größer und auch dementsprechend leistungsfähiger. Außerdem ermöglichen die Motoren Geschwindigkeiten von bis zu $200 \frac{mm}{s}$ ($0,2 \frac{m}{s}$), was nicht nur eine einfachere und schnellere Entwicklung der Steuerungssoftware sondern auch wesentlich schnellere Zeichnungen ermöglicht. Die wesentlichen relevanten Unterschiede im mechanischen Aufbau zu meinem ersten Plotter sind schnellere und leistungsfähigere Motoren, sowie eine stabilere und damit zuverlässigere mechanische Konstruktion. Zudem kann es durch die Verwendung von Linearkugellagern (statt der ungelagerten Metallhülsen beim ersten Plotter) nicht zu einem Verkanten des Y-Schlittens kommen. Außerdem wird bei dem zweiten Plotter (im Gegensatz zum ersten) Einer der wenigen Nachteile ist allerdings die geringere Druckgröße von maximal 20×20 cm (statt 30×30 cm), dadurch können keine A4-Blätter genutzt werden.

3 Erläuterungen zur Software

3.1 Software des Mikrocontrollers

3.1.1 Software des ersten Plotters

Die Software für den Mikrocontroller wurde in C++ geschrieben. Das bringt mehrere Vorteile mit sich: Zum einen kann der Code schnell und effizient ausgeführt werden, was besonders bei Mikrocontrollern wegen ihrer begrenzten Rechenleistung und relativ geringen Taktrate vorteilhaft ist, um eine schnelle Ausführung zu ermöglichen. Zum zweiten ist C++ objektorientiert und kann deswegen sehr flexibel und effizient programmiert werden. Zum dritten gibt es bereits sehr viele Bibliotheken, da C++ die vom Arduino-Projekt verwendete Programmiersprache ist, welche auch die einzige mit der offiziellen Arduino-IDE programmierbare Sprache ist.

Zum Entwickeln habe ich anfangs die Arduino-IDE, später Qt Creator genutzt, da dieser durch Features wie dynamischer Vorschläge während des Eingabens und teilweise automatischer Vervollständigung eine wesentlich schnellere und bequemere Entwicklung ermöglicht. Die Fehlersuche ist durch bessere und detailliertere Fehlermeldungen sowie übersichtliche Syntax-Hervorhebung einfacher und schneller möglich.

Da mir leider zum Zeitpunkt des Programmierens die Vorteile des für CNC-Maschinen üblichen und standardisierten G-CODES nicht bekannt waren, ha-

be ich ein eigenes Protokoll für die Kommunikation zwischen Zeichensoftware bzw. Computer und Plotter entwickelt, welches schlussendlich zwar hervorragend funktioniert, aber den Plotter nur exklusiv mit meinem Programm steuerbar macht. Daraus ergeben sich entscheidende Nachteile für z.B. das Zeichnen von Objekten, die von meinem Zeichenprogramm nicht unterstützt werden. Deswegen ist es nicht möglich, diese zu zeichnen, was die Einsatzmöglichkeiten ohne Veränderung der Software stark einschränkt. Trotzdem ist mein Protokoll effizient und ermöglicht eine zuverlässige Ansteuerung des Plotters, wobei dieser auch die meisten Rechenaufgaben übernimmt und so auch eine computerlose also selbstständige Steuerung oder zumindest eine Ansteuerung durch beispielsweise einen zweiten Mikrocontroller möglich wäre. Außerdem ist es auch für Menschen leicht lesbar, da die Befehle nach der üblichen C-Funktionssyntax aufgebaut sind (Befehl(Parameter1, Parameter2, ...)). Auch Fehlerausgaben wie z.B. „ungültiger Befehl“ sind so gestaltet, dass sie für Menschen leicht lesbar sind.

Der Algorithmus für die Interpretation der Befehle sucht zuerst nach Zeichen, die eine Trennung des Befehls in Befehlsname und Parameter ermöglichen („(“, „)“, „,“). Deren Positionen werden gespeichert und anschließend genutzt um mithilfe der substring-Funktion der String-Klasse eine Zwischenspeicherung des Befehlsnamens (als Zeichenkette) und der Parameter (als Zahl) zu ermöglichen. So können diese im letzten Schritt genutzt werden, um die richtige Funktion des Programms mit den notwendigen Argumenten aufzurufen. Dieser Algorithmus ist in der Funktion „processInputCommand“ in der Datei „app.cpp“ im Ordner „PlotR“ implementiert.

Zum zeichnen von Linien in X- und Y-Richtung wird ein einfacher Algorithmus verwendet, der die notwendige Anzahl von Schritten, die der jeweilige Schrittmotor durchführen muss, berechnet und diese mithilfe der „PlotRAxis“-Klasse durchführt.

Nach jeder durchgeführten Bewegung müssen die Motoren in den „Schlafmodus“ geschaltet werden, um nicht zu überhitzten und trotzdem ihre Position beizubehalten.

Anfangs hatte der Plotter auch eine Funktion zum Zeichnen von Diagonalen, welche die beiden Schrittmotoren abwechselnd angesteuert hat und den Motor mit der höheren Auflösung bestimmte Schritte überspringen lassen hat, um die unterschiedlichen Auflösungen zu kompensieren. Jedoch wurde diese entfernt, da sie durch den Algorithmus zum Zeichnen beliebiger Geraden unnötig wurde und der Programmspeicher des Mikrocontroller relativ klein ist.

Der Algorithmus zum Zeichnen dieser beliebigen Geraden basiert auf dem Bresenham-Algorithmus, da dieser eine sehr schnelle Berechnung ermöglicht, auch wenn diese durch die relativ geringe Zeichengeschwindigkeit nicht unbedingt notwendig ist. Zur einfacheren Erklärung nehmen wir an, die zu zeichnende Gerade hat einen Anstieg $0 < m < 1$ und führt von $(x_1|y_1)$ nach $(x_2|y_2)$. Nehmen wir weiterhin an, $dx = x_2 - x_1$ und $dy = y_2 - y_1$. Bei der Annahme von $0 < m < 1$ gilt dann $0 < dy < dx$, für andere Anstiege muss dann nur eine Fallunterscheidung auf Basis der Vorzeichen von dy und dx getroffen werden. Ähnlich wie bei dem Rastern einer mathematischen Funktion (siehe Ellipsenfunktion) gehen wir von

einer schrittweisen Rasterung der x-Werte aus. Dabei wird nach jedem Schritt in X-Richtung ein Schritt in Y-Richtung durchgeführt falls dieser notwendig ist. Diese Notwendigkeit wird mithilfe des Fehlerglieds ermittelt. Das Fehlerglied ergibt sich aus dem Auflösen der Geradengleichung in der Zweipunkteform $y = \frac{dx}{dy} \cdot (x - x_1) + y_1$ in $0 = dx \cdot (y - y_1) - dy \cdot (x - x_1)$, wobei dann die 0 das Fehlerglied e ergibt. Bei einem X-Schritt (von 1) gilt dann $e_{neu} = e_{alt} - dx$. Wenn das neue Fehlerglied e_{neu} jetzt kleiner als 0 ist muss ein Y-Schritt gemacht werden. Nach diesem Schritt ergibt sich das neue Fehlerglied aus $e_{neu} = e_{alt} + dx$, denn $e_{alt} = dx(y - y_1) - dy \cdot (x - x_1)$ und $e_{neu} = dx(y - y_1 + 1) - dy \cdot (x - x_1)$, da ein y-Schritt zurückgelegt wurde. Durch diese starke Vereinfachung ist es mit wenigen Rechenoperationen, von denen nur eine eine Division ist, während alle anderen Additionen und Subtraktionen sind, die von Computern sehr schnell berechnet werden können (je 1 bitweise Operation). Dieser Algorithmus ist so einfach und gleichzeitig Effizient, dass er teilweise auf Hardwareebene in Grafikchips implementiert wird.]

Um auch Kreise darzustellen, muss der Plotter wegen der unterschiedlichen Auflösungen der Schrittmotoren Ellipsen zeichnen. Anfangs habe ich für diese den Midpoint-Algorithmus verwendet, allerdings hatte ich Probleme mit den Fehler-Variablen, da diese zu Problemen geführt haben, weil sie das Maximum der Variable überschritten (Stackoverflow). Deswegen habe ich schlussendlich einen Algorithmus entwickelt, der die Ellipsenfunktion $y^2 = b^2(1 - (x^2/a^2))$ wobei a der Breite und b der Höhe der Ellipse entspricht, in jedem Quadranten zeichnet. Dabei berechnet der Mikrocontroller jeden y-Wert für jeden x-Schritt mit $y = \sqrt{b^2(1 - (x^2/a^2))}$. Die Implementation dieses Algorithmus befindet sich in der Ellipse-Funktion in der Datei „app.cpp“.

Zur Interaktion mit den Schrittmotortreibern habe ich die PlotRAxis3-Klasse implementiert, welche neben der schrittweisen und Positions- Steuerung eines Schrittmotors mit einem StepStick auch weitere Funktionen wie die Regelung dessen Geschwindigkeit, dem Zurücksetzen der Achse zu seiner Ursprungsposition auch das an- und abschalten den Schlafmoduses des StepSticks ermöglicht.

3.1.2 Software des zweiten Plotters

Die Steuerungselektronik des 3D-Druckers wurde beim Umbau zum Plotter nicht verändert. Trotzdem möchte ich kurz erläutern welche Algorithmen der 3D-Drucker/Plotter für Linien und Kreise bzw. Ellipsenbögen verwendet. Dies ist möglich, weil die Firmware des 3D-Druckers (Marlin V2.2.1) quelloffen ist.

Der Algorithmus für das Zeichnen von Ellipsenbögen bzw. Ellipsen (vollständiger Bogen), die bei gleicher Höhe und Breite Kreise sind befindet sich in [19; Z. 7058-7192]. Die Berechnung der Koordinaten erfolgt dabei mithilfe einer „Vector rotation by transformation matrix“ also einer Vektorrotation mit einer Drehmatrix. Dieser Algorithmus basiert auf einem Lösungsvorschlag von Jens Geisler [vgl. 19; Z. 7094]. Der Ansatz dieses Algorithmus ist die Drehmatrix für die Drehung um α (mathematisch positiv) der euklidischen Ebene R^2 die mit $R_\alpha = \begin{pmatrix} \cos \alpha & -\sin \alpha \\ \sin \alpha & \cos \alpha \end{pmatrix}$ bzw. um α (mathematisch negativ)

$R_{-\alpha} = \begin{pmatrix} \cos \alpha & \sin \alpha \\ -\sin \alpha & \cos \alpha \end{pmatrix}$ definiert ist. Wobei R_α bzw. $R_{-\alpha}$ jeweils den gedrehten Vektor beschreiben. Der zu zeichnende Bogen wird aus sehr kleinen Linien (Segmenten) berechnet, die dann näherungsweise eine/n Ellipsen- bzw. Kreis(-bogen) ergeben bei denen jeweils gilt: $\alpha = \frac{\text{Gesamtwinkel}}{\text{Segmentanzahl}}$.

3.2 Steuersoftware des zur Steuerung verwendeten Computers

Die Steuersoftware ist in C++ geschrieben, außerdem wird für die grafische Benutzeroberfläche (GUI) und die Serielle Schnittstelle die Qt-Bibliothek genutzt. Der Hauptvorteil der Qt-Bibliothek ist, dass derselbe Quellcode für Linux, Windows und Mac OS verwendet werden kann. Außerdem ist auch eine Portierung für Android, iOS, Blackberry und WindowsPhone mit relativ geringen Aufwand möglich. Die Bibliothek ist in einer minimal eingeschränkten quelloffenen Version verfügbar, welche für die meisten Anwendungen ausreicht. Für das (G)UI-Design und die Programmierung wurde die QtCreator-IDE genutzt.

Die GUI der Steuersoftware (vgl. 5.14) ist im Wesentlichen selbsterklärend, weshalb ich nur kurz auf sie eingehen werde. Das Hauptfenster hat 4 Tabs: „Terminal/Connection“, „Basic Control“, „Simple Geometry“ und „Functions“.

Im ersten Tab wird sowohl der Port angezeigt an welchem der Plotter angeschlossen als auch die Baudrate, mit der die Verbindung erfolgt. In diesem Tab kann man außerdem Befehle senden und Rückgaben einsehen.

Der „Basic Control“-Tab ermöglicht die einfache Steuerung des Plotters. Hier können beispielsweise die Achsen kalibriert und die Motoren an- oder abgeschaltet werden. Außerdem befinden sich im „move“-Bereich Steuerelemente für das Bewegen der Achsen - beispielsweise dient der „Z↑“-Button, zum Heben der Z-Achse bzw. des Stiftes.

Um geometrische Primitiven einfach zu zeichnen, können im „Simple Geometry“-Tab Linien („Line“), Dreiecke („Triangle“), Quadrate („Square“), Rechtecke („Rectangle“) und Kreise („Circle“) geplottet werden. Neben dieser Auswahl gibt es beim Rechteck noch einen 2. Definitionsmodus, bei dem 2 Punkte definiert werden, wobei der erste Punkt der untere Linke und der zweite Punkt der obere Rechte ist. In der rechten Hälfte befindet sich ein Vorschaufenster, welches jeweils ein Objekt anzeigen kann, das dann mit dem „Plot“-Button vom Plotter geplottet werden kann.

Der Letzte Tab dient zum Darstellen von Funktionen. Die Vorschau erfolgt dynamisch mit der Eingabe der Parameter. Zum Zeichnen der Funktion dient der „Plot“-Button, welcher ein Fenster öffnet. In diesem Fenster kann die Position des Ursprungs der Funktion auf dem Blatt eingegeben werden, außerdem gibt es verschiedene Optionen für das Koordinatensystem wie beispielsweise die Option, die Pfeile abzuschalten. Nach dem bestätigen mit „OK“ wird der G-CODE erzeugt und im ersten Tab angezeigt, aus welchem er dann kopiert und in eine Datei eingefügt werden kann, um diese dann mit einer 3D-Druck Software zu öffnen.

4 Weitere Einsatzmöglichkeiten meines und anderer Plotter

4.1 Einsatzmöglichkeiten meiner Plotter

4.1.1 Einsatzmöglichkeiten des ersten Plotters

Die erste und offensichtlichste Einsatzmöglichkeit ist das Zeichnen von geometrischen Figuren und Funktionen. Dies ist selbstverständlich ohne jegliche Modifikation möglich. Man könnte selbige auch ausschneiden indem man statt dem Stift ein Messer oder einen Schneidlaser an der Stifthalterung befestigt und so einen Schneidplotter baut. Die Methode mit dem Schneidlaser ist hierbei weniger zu empfehlen, da Laser (dieser Leistungsklasse) Sicherheitsvorkehrungen wie beispielsweise Schutzbrillen erfordern.

Außerdem ist es möglich Muster zu zeichnen, welche auf Linien basieren, dies erfordert jedoch entweder geringfügige Änderungen an der Software oder aufwändige Eingaben.

Die offensichtbare Mechanik eines Plotters (und auch eines 3D-Drucker) ist als Anschauungsmaterial für technisches Verständnis vorteilhaft.

4.1.2 Einsatzmöglichkeiten des zweiten Plotters

Dank der G-CODE-Interaktion und sehr zuverlässigen Funktionsfähigkeit sind mit dem zweiten Plotter und zusätzlicher Software sehr viele weitere Einsatzmöglichkeiten denkbar. So könnte man Logos, Symbole oder sogar Bilder, für welche der G-CODE mit der freien Software Inkscape generiert werden kann Zeichen (z.B. mit einem wasserfesten Stift auf den Akkudecke eines Handys). Mithilfe von Inkscape könnte man außerdem den G-CODE für die Leiterbahnen einer Platine erzeugen, welche geätzt werden soll, und diese dann mit einem Permanentstift auf eine leere Platine Zeichnen und diese dann ätzen.

Der einfache Umbau des 3D-Druckers zum Plotter zeigt, dass 3D-Drucker auf den Grundlagen von Plottern aufbauen.

Durch weitere Modifikationen kann man eine Fräse bauen. Auch hier zeigt sich wieder, dass viele moderne Maschinen (besonders CNC-Maschinen) auf den Grundlagen der Plotter aufbauen.

4.2 Einsatzmöglichkeiten von Plottern (allgemein)

Schneidplotter sind nach wie vor aktuelle Maschinen, sie werden beispielsweise bei der Herstellung von Klebefolien genutzt. [18] Zeichenplotter wurden nach und nach durch Laser und Tintenstrahldrucker ersetzt. Wie bereits im vorherigen Abschnitt erwähnt, bilden Plotter die Grundlage für viele Moderne Maschinen wie beispielsweise Fräsen und 3D-Drucker. Auch Tintenstrahldrucker basieren auf den Grundlagen von Plottern, insbesondere Rollenplottern, da bei diesen auch das Blatt an einem Schlitten mit der Zeichen- bzw. Druckvorrichtung vorbei bewegt wird.

Außerdem können Plotter durch Modifikation zum Beispiel für Kunstprojekte verwendet werden. [18] Im weitesten Sinne ließen sich auch Brückenkräne als eine Form von Plotter oder als mit dem Plotter „verwandt“, aber auf jeden Fall mit „plotterähnlich“ bezeichnen, insofern diese von einem Computer gesteuert werden.

5 Anhang

5.1 Bilder

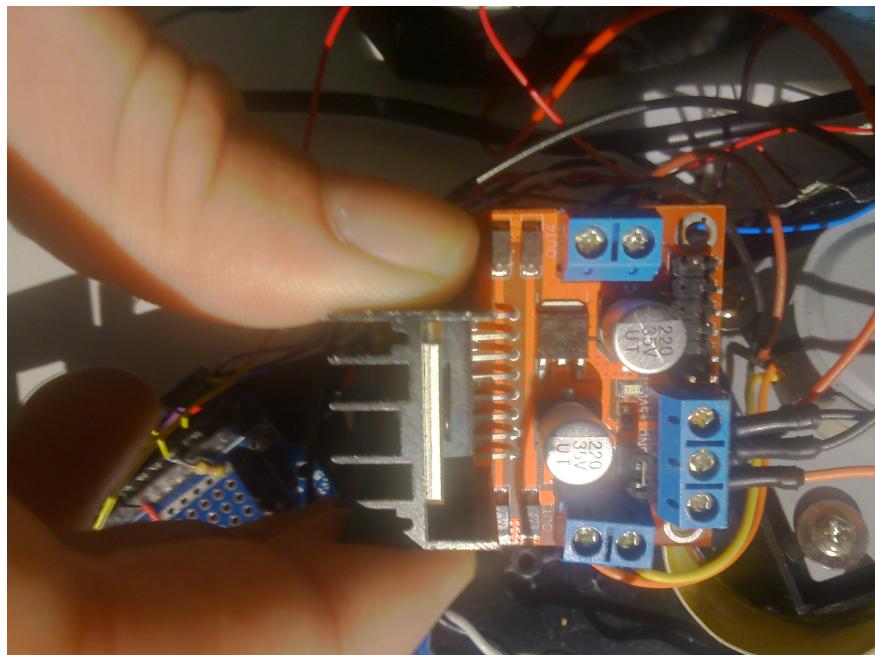


Abb. 5.1. H-Brücke



Abb. 5.2 Y-Schrittmotor

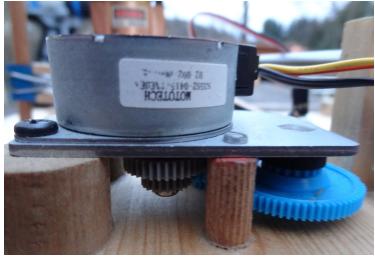


Abb. 5.3 X-Schrittmotor mit Getriebe

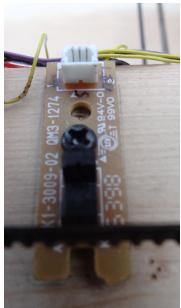


Abb. 5.4 IR-Gabellichtschranke der X-Achse

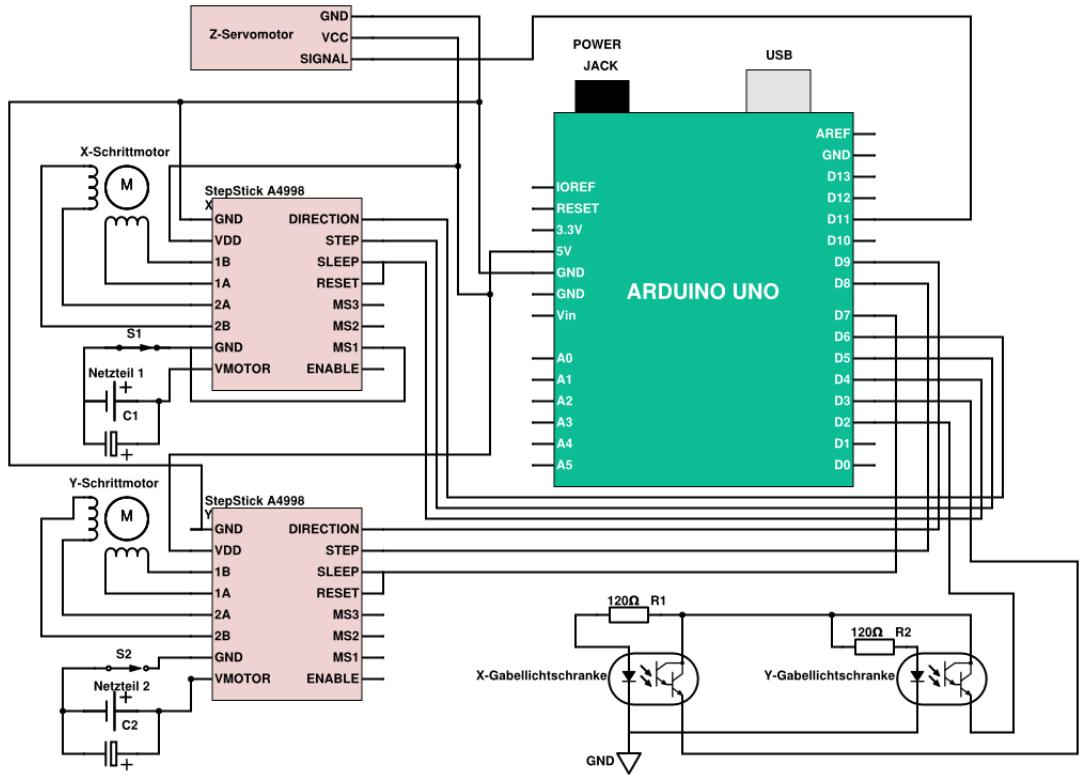
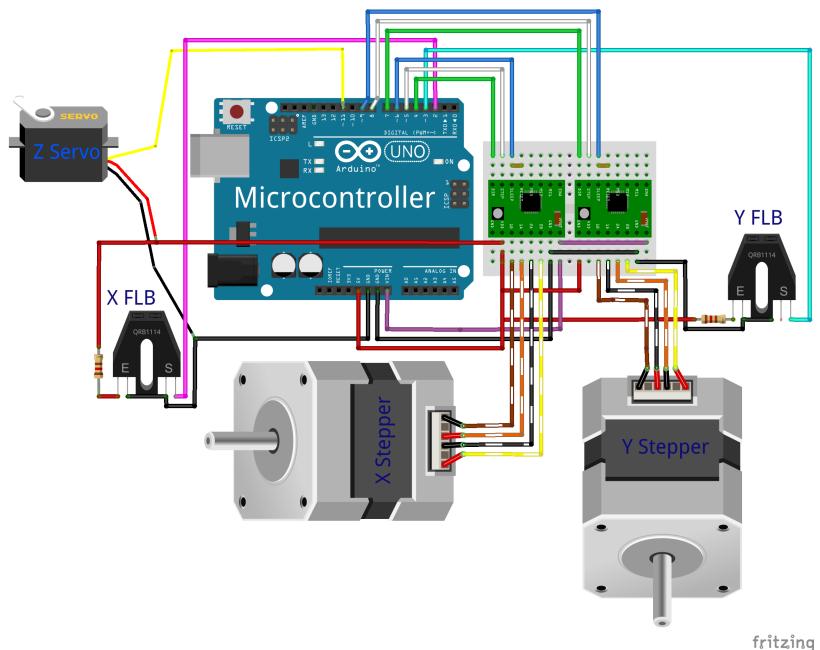


Abb. 5.5 Schaltplan des ersten Plotters



fritzing

Abbildung 5.6 Schematischer Aufbau der Schaltung

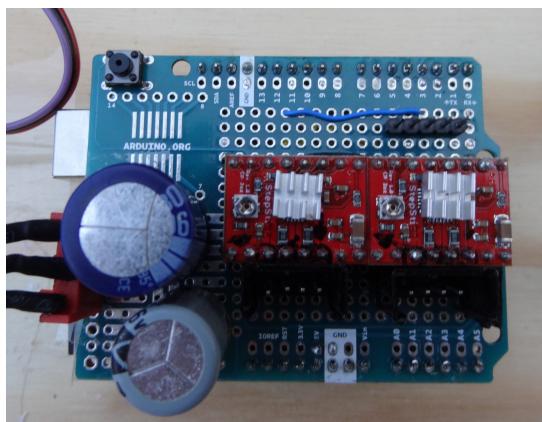


Abb. 5. 7 fertig aufgebautes Steuerungsmodul (auf Mikrocontroller aufgesteckt) mit Schrittmotortreibern (rote Platinen)

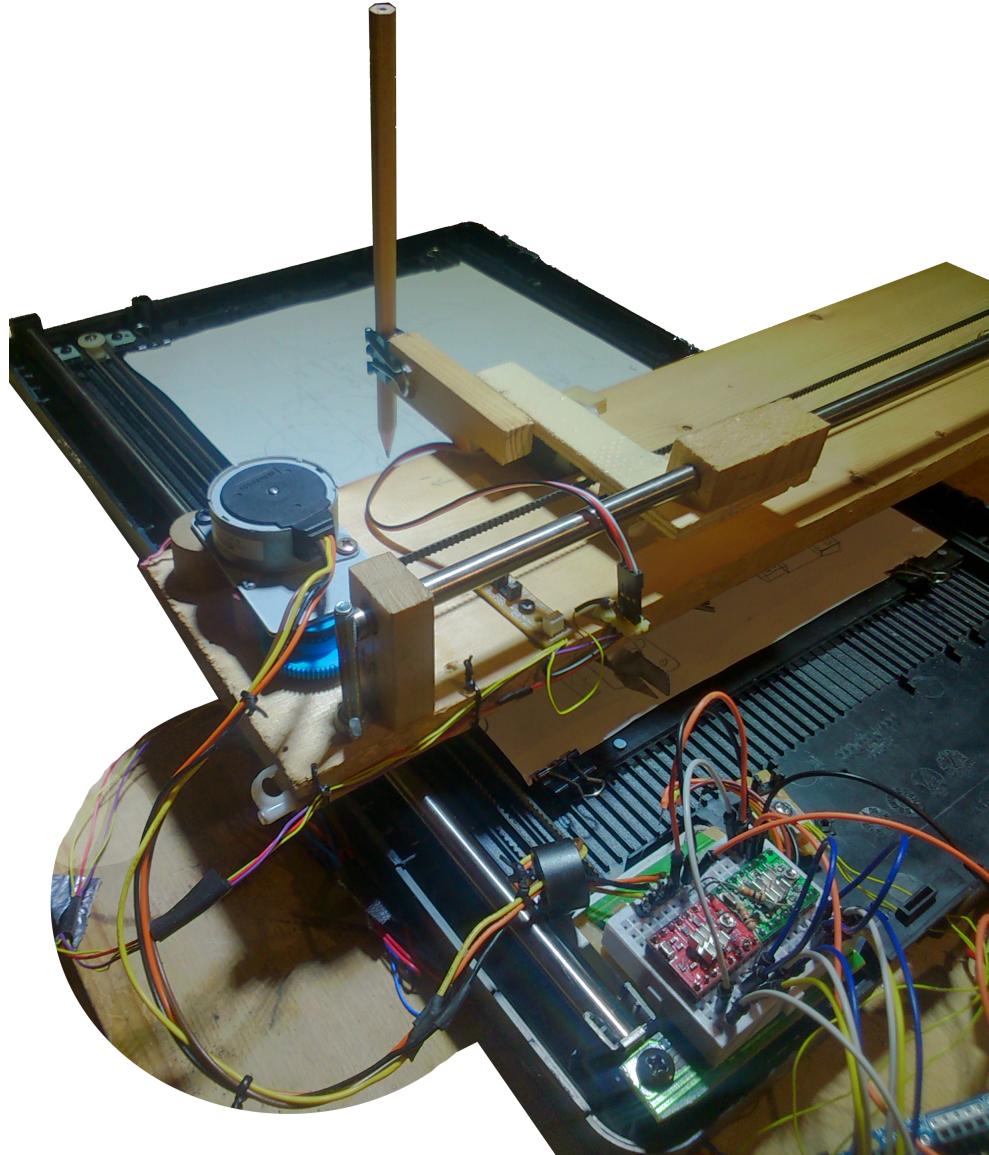


Abb. 5.8 Erster Plotter in der Testphase

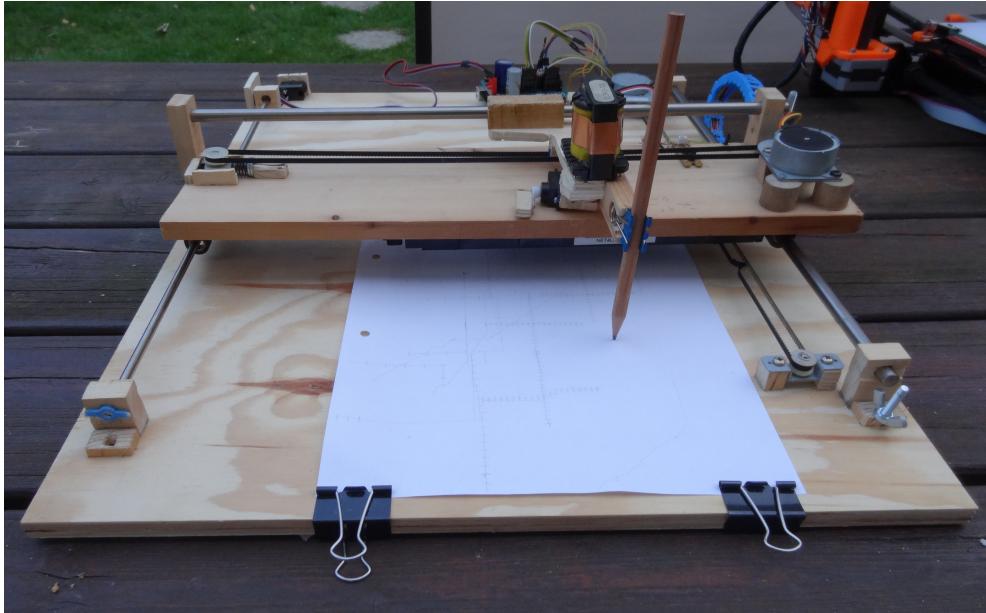


Abb. 5.9 Erster Plotter im aktuellen Status (3.04.2016)

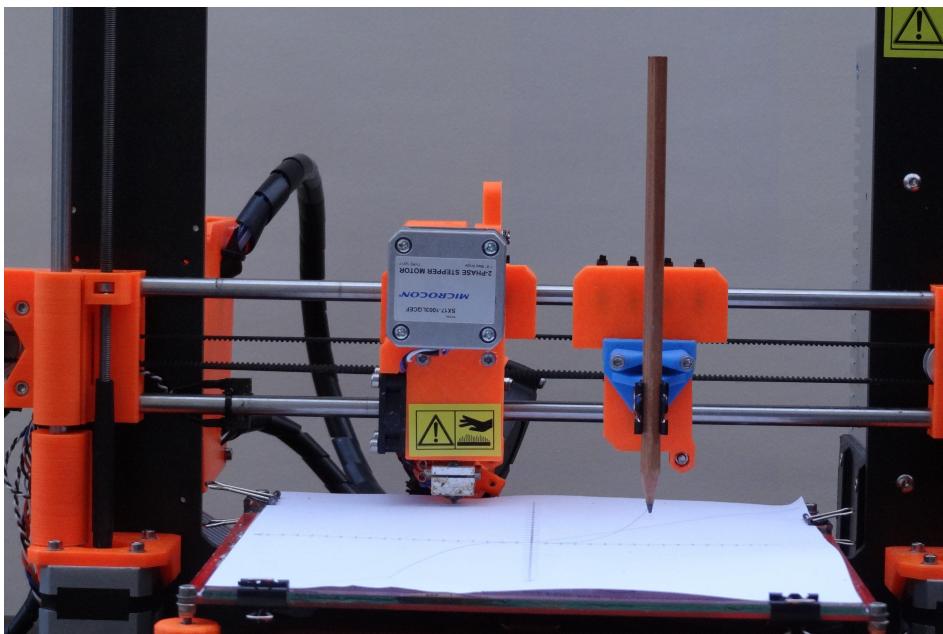


Abb. 5.10 Vergleich von Extruder und Stifthalterung mit Stift (Fotomontage)

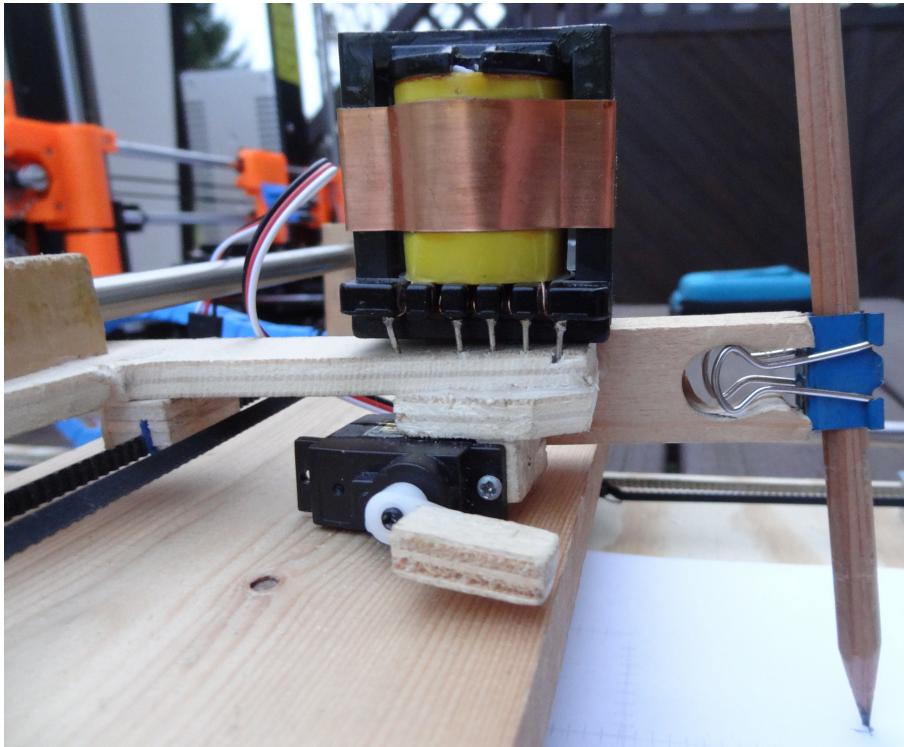


Abb. 5.11 Detailaufnahme des X-Schlittens mit Servomotor, Gewicht und Stift in Stifthalterung

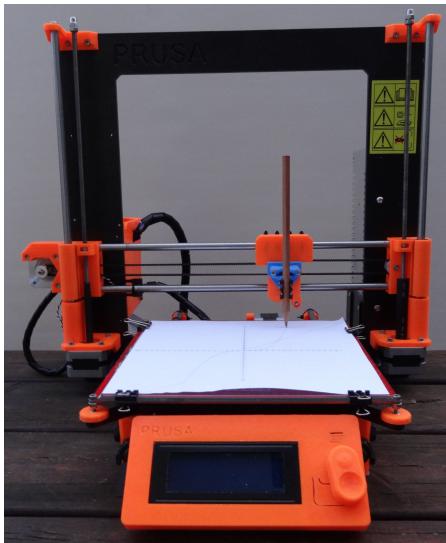


Abb. 5.12 zweiter Plotter

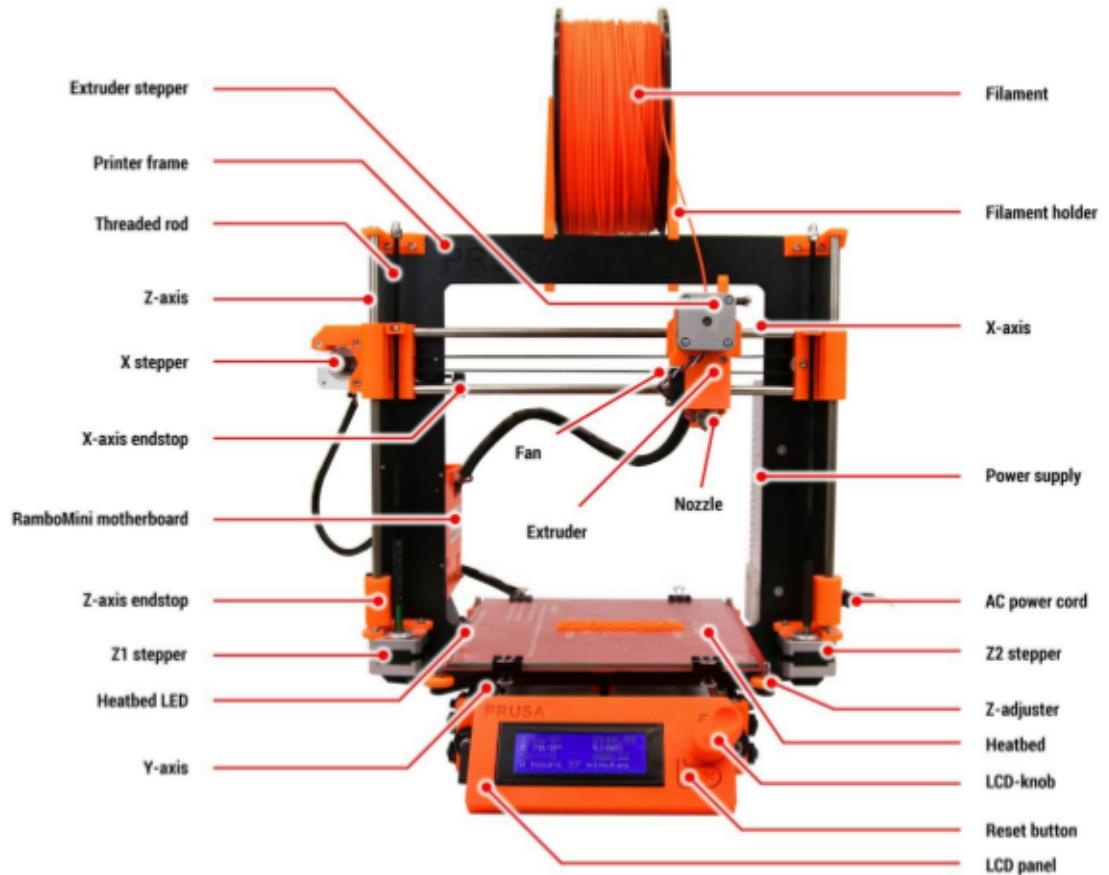


Abb. 5.13 zweiter Plotter (beschriftet) [aus 12]

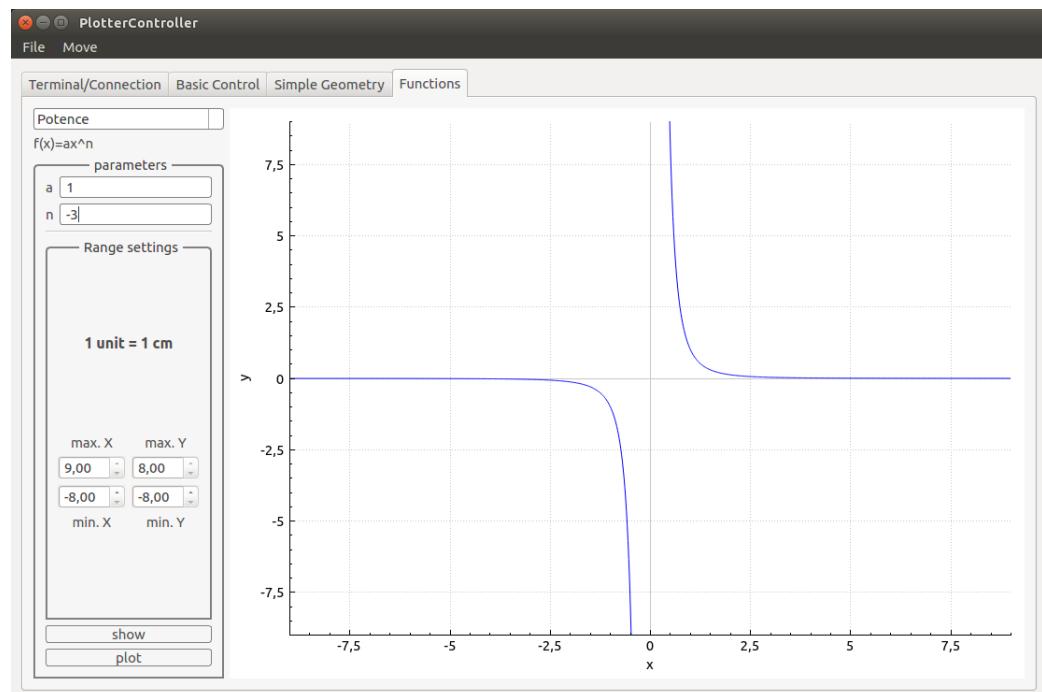


Abb. 5.14 Screenshot des „Funtions“-Tabs der PlotterController-Software

5.2 Quellenverzeichnis

Literatur

- [1] (Wikipedia) mehrere Verfasser. Plotter. in: <http://de.wikipedia.org/wiki/Plotter> zugegriffen: am 2.04.2016 15:52
- [2] (Wikipedia) mehrere Verfasser. Vektorgrafik. in: <http://de.wikipedia.org/wiki/Vektorgrafik> zugegriffen: 2.04.2016 16:54
- [3] (Wikipedia) mehrere Verfasser. Rastergrafik. in: <http://de.wikipedia.org/wiki/Rastergrafik> zugegriffen: 2.04.2016 16:57
- [4] (Wikipedia) mehrere Verfasser. Schrittmotor. in: <http://de.wikipedia.org/wiki/Schrittmotor> zugegriffen: 2.04.2016 17:46
- [5] (Wikipedia) mehrere Verfasser. Vierquadrantensteller in: <http://de.wikipedia.org/wiki/Vierquadrantensteller> zugegriffen: 2.04.2016 19:25
- [6] (ST Microelectronics) Unbekannt. L298 in: http://www.sparkfun.com/datasheets/Robotics/L298_H_Bridge.pdf zugegriffen: 2.04.2016 19:13
- [7] (PBC LINEAR) Unbekannt. Stepper Motor NEMA 17. in: <http://www.pbclinear.com/Download/DataSheet/Stepper-Motor-Support-Document.pdf> zugegriffen: 2.04.2016 19:23
- [8] (Wikipedia) mehrere Verfasser. StepStick. in: <http://reprap.org/wiki/StepStick> zugegriffen: 2.04.2016 19:44
- [9] (Wikipedia) mehrere Verfasser. RepRap. in: <http://de.wikipedia.org/wiki/RepRap> zugegriffen: 2.04.2016 19:49
- [10] (Wikipedia) mehrere Verfasser. Arduino (Plattform) in: http://de.wikipedia.org/wiki/Arduino_%28Plattform%29 zugegriffen: 2.04.2016 19:58
- [11] (Wikipedia) mehrere Verfasser. Mikrocontroller. in: <http://de.wikipedia.org/wiki/Mikrocontroller> zugegriffen: 2.04.2016 20:16
- [12] (Prusa Research s.r.o.) Unbekannt. 3D PRINTING HANDBOOK in: http://prusa3d.com/downloads/manual/prusa3d_manual_175_en.pdf zugegriffen: 3.04.2016 16:32
- [13] (Github) mehrere Verfasser. MarlinFirmware/Marlin in: https://github.com/MarlinFirmware/Marlin/blob/RC/Marlin/Marlin_main.cpp Abgerufen am 3.04.2016 18:04
- [14] (Wikipedia) mehrere Verfasser. Drehmatrix. in: <http://de.wikipedia.org/wiki/Drehmatrix> Abgerufen am 3.04.2016 18:11

- [15] (Wikipedia) mehrere Verfasser. Bresenham-Algorithmus in: <https://de.wikipedia.org/wiki/Bresenham-Algorithmus> Abgerufen am 3.04.2016 19:03
- [16] (Wikipedia) mehrere Verfasser. Geradengleichung. in: <https://de.wikipedia.org/wiki/Geradengleichung> zugegriffen: 3.04.2016 19:08
- [17] Daniel Iglesia. The Draftmasters - I. in: <https://vimeo.com/4611451>. zugegriffen: 3.04.2016 22:30
- [18] (myfolie) Unbekannt. Was sind Klebeschriften. in: <https://www.myfolie.com/was-sind-klebeschriften> zugegriffen: 3.04.2016 23:07
- [19] (Wikipedia) mehrere Verfasser. Computer Numerized Control. in: https://de.wikipedia.org/wiki/Computerized_Numerical_Control#DIN.2FISO-Programmierung_bzw._G-Code zugegriffen: 3.04.2016 23:54