Ruprecht-Karls-Universität Heidelberg
Institut für Informatik
Optimierung in Robotik und Biomechanik

Bachelorarbeit

RepRap 3D-Drucker – Ein Open Source-Projekt für Hardware

Name: Joachim Schleicher
Matrikelnummer: 2604291
Betreuer: Katja Mombaur und Khai-Long Ho Hoang
Datum der Abgabe: 5. April 2012

Ich versichere, dass ich diese Bachelor-Arbeit selbstständig verfasst und nur die angegebenen Quellen und Hilfsmittel verwendet habe.

Heidelberg, den 5. April 2012 .....

Zusammenfassung

Rapid Prototyping bezeichnet die schnelle Umsetzung eines 3D-Modells von einem Computerentwurf (CAD) in ein physikalisches Objekt. Eine einfache und günstige Möglichkeit dazu bietet der 3D-Drucker *RepRap*, der 2008 von A. Bowyer vorgestellt wurde. RepRap ist ein selbst-replizierender rapid-prototyper, da er seine Kunststoffteile selbst drucken kann. So lässt sich mit einem Reprap eine Kopie des Roboters herstellen, den der Benutzer direkt mit Standardschrauben und Elektronik zusammenbauen kann.
Das Konzept der *Open Source* stellt dabei die freie Verbreitung des Druckers sicher. Alle Teile und Software stehen unter einer freien Lizenz, die sicherstellt, dass auch Verbesserungen wieder frei verfügbar sind.
Im Rahmen dieser Arbeit wurde ein 3D-Drucker vom Typ RepRap Mendel aus seinen Einzelteilen zusammengebaut und kalibriert. Die Genauigkeit liegt im Sub-Millimeter-Bereich und passende Software ist in der Lage, beliebige CAD-Modelle in für den Drucker geeignete GCode-Befehle umzusetzen. Ein Mikrocontroller übernimmt die Schrittmotoransteuerung und Sensorauswertung auf Hardware-Level. Als Anwendung haben wir eigene Teile für den Einsatz mit Robotern aus der Arbeitsgruppe entworfen und gedruckt.

Abstract

Rapid Prototyping is a technique to quickly build real physical objects from CAD models. RepRap is the replicating rapid prototyper presented by Bowyer in 2008. It utilizes Fused Filament Fabrication and prints plastics objects in 3D. A large fraction of its own parts can be built by the robot itself allowing the user to copy the machine's essential parts and print spare parts for the machine.
The concept of Open Source allows a free distribution and requires improvements to be published under an open license as well. Thus several thousand copies of RepRap were built during the last four years. The community evolves rapidly and steadily improves the design.
During this thesis we built an RepRap Mendel from its individual parts and tested the capabilities of the printer. Some calibration work is necessary but overall the machine works pretty well. A microcontroller is used to control stepper motors and read sensor values on hardware level. As a usability demonstration additional parts for robots available in the lab were modelled and printed with RepRap Mendel.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

Zum Rapid Prototyping eröffnet ein 3D-Drucker neue Möglichkeiten: Computer Aided Design (CAD)-Entwürfe können innerhalb kürzester Zeit als reale Objekte getestet werden. Diese Technik eröffnet nicht nur für Architekten und Planer neue Visualisierungsmöglichkeiten, sondern lässt sich auch sehr gut für Robotikbauteile aus Kunststoff einsetzen. Der 3D-Drucker RepRap Mendel ist ein Roboter, der einen Teil seiner Bestandteile selbst reproduzieren kann. Er besteht aus Mechanik- und Elektronik-Komponenten, die im Robotikbereich geläufig sind. Der Antrieb wird mit Schrittmotoren realisiert, die von einem Mikrocontroller-Board gesteuert werden. Ein Computer schickt low-level Druckkommandos über eine serielle Schnittstelle an den „Drucker“. Eine stabile mechanische Rahmenkonstruktion bildet die Voraussetzung für reproduzierbare Druckergebnisse.

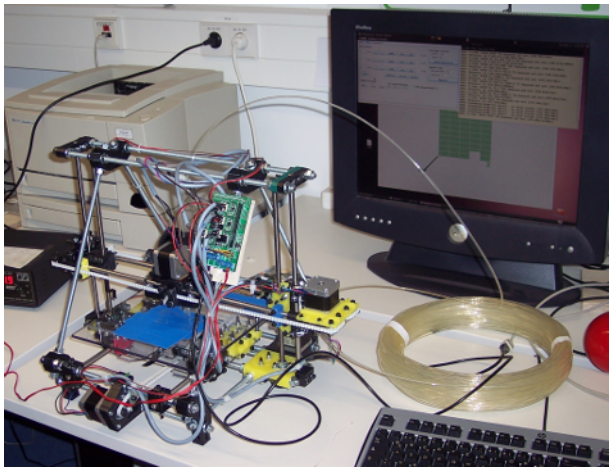


Abbildung 1.1: RepRap Mendel bei einem ersten Druck. Im Hintergrund zeigt der Bildschirm die Java Host Software.

Open Source konnte sich in den letzten Jahrzehnten für namhafte Software-Projekte etablieren. Schnelle Änderbarkeit sowie die Garantie, dass auch geänderte Versionen stets frei verfügbar sind, bilden die Grundlage dafür. Verschiedene Projekte erweiterten mittlerweile die Prinzipien von OpenSource auf Hardware und stellen sowohl Elektronik-Schaltpläne als auch CAD-Zeichnungen aller Teile unter eine freie Lizenz. Das schnelle Wachstum des Replicating Rapid-prototyper (RepRap)-Projekts ist ein Beispiel, wie Entwicklungen auch im Hardware-Bereich von einer offenen Lizenz profitieren können. de Bruijn (2010) hat die Auswirkungen und Entwicklung innerhalb dieses Projekts systematisch untersucht und deutlich höhere Wachstumsraten ermittelt, als für geschlossene Industrieentwicklungen üblich sind.
Im Rahmen dieser Bachelorarbeit wurde ein RepRap Mendel 3D-Drucker aus Einzelteilen aufgebaut, angesteuert und getestet (vgl. Foto in Abbildung 1.1). Dazu zählt sowohl

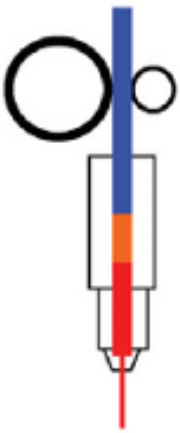


Abbildung 1.2: Bei der *Fused-Filament-Fabrication* wird ein Filament im Extruder geschmolzen und verbindet sich so mit vorher gedruckten Lagen des zu druckenden Objekts.

die elektrische Verbindung und elektronische Ansteuerung von Schrittmotoren und Extruder als auch die Software-Schnittstellen zum PC. Verschiedene Software-Projekte implementieren die Host- und Mikrocontroller-Funktionalität, sodass wir die für unseren Zweck am besten geeignete Lösung aussuchen konnten. Sämtliche verwendeten Pakete stehen unter einer freien Lizenz, sodass Änderungen am Quellcode schnell und unproblematisch möglich waren.

Fused-Filament-Fabrication wird vom RepRap Mendel angewandt, da es eine sehr einfache Rapid-Prototyping-Lösung ist. Das Rohmaterial wird als Filament zugeführt und bei hoher Temperatur im Druckkopf geschmolzen. Das gewünschte Objekt lässt sich Ebene für Ebene produzieren und wächst so dreidimensional in die Höhe. Dieses Prinzip ist in Jones et al. (2011) beschrieben und in Abbildung 1.2 veranschaulicht.

Im Bereich professioneller Rapid-Prototyping-Maschinen sind auch andere Fertigungsmethoden üblich. Beim Lasersintern wird ein Pulver schichtweise aufgebracht und mit einem Laser ausgehärtet. Zum Laminated Object Manufactoring werden Papier- oder Kunststoffschichten aufeinander laminiert und die Konturen jeder Ebene ausgeschnitten. Bei der Entwicklung des RepRap lag das Hauptaugenmerk jedoch auf einer einfachen und kostengünstigen Variante (Sells, 2009). So scheiden Laser-basierte Methoden schnell aus und es zeigte sich im Laufe der Entwicklungen, dass ein 3D-Drucker basierend auf Fused-Filament-Fabrication deutlich günstiger hergestellt werden kann, als bisher auf dem Markt verfügbare Modelle. Die Kosten für einen RepRap Mendel liegen bei unter 700 Euro, während beispielsweise ein *Dimension 1200es* 3D-Drucker – der nach demselben Grundprinzip arbeitet – bei einem Listenpreis von 18 500 Euro liegt. *Bits from Bytes* und *Makerbot Industries* bieten inzwischen erschwingliche Drucker an, die auf den Prinzipien des RepRap basieren und verwendet werden können, um Teile für einen RepRap zu drucken.

Die vorliegende Arbeit beschreibt zunächst die Grundideen der mechanischen Konstruk-

tion, um danach auf die vorgesehene Elektronik einzugehen. Die Inbetriebnahme umfasste das Flashen der Firmware in den Mikrocontroller, nachdem typische Parameter und Optionen zur Host-Kommunikation an unseren Aufbau angepasst waren. Der darauf folgende Teil behandelt Kalibration sowie den Workflow vom 3D-CAD-Modell zum gedruckten Kunststoff-Objekt. Dazu haben wir beispielhaft einen Greifer für einen in der Arbeitsgruppe verwendeten Roboterarm entworfen, der ein Wasserglas aufnehmen und balancieren kann. Im Anhang sind der vollständige Quellcode dieses Greifers und die Konfigurationseinstellungen für Skeinforge tabellarisch dargestellt. Kleine Anpassungen an der Firmware konnten upstream eingebracht werden und sind ebenfalls im Anhang aufgelistet.

2 Aufbau des RepRap Mendel

Die Konstruktion des RepRap Mendel ist beschrieben im RepRap-Wiki (Bowyer et al., 2012b) und steht unter der freien GNU General Public License. Sämtliche Kunststoffteile können von einem RepRap 3D-Drucker selbst gedruckt werden. Wir entschieden uns aber für die Bestellung gegossener Kunststoffteile, aus denen wir den RepRap Mendel zusammengebaut haben (Krautwasser und Wenger, 2011–2012). Der Druck verschiedener Ersatzteile hat sich anschließend an eine exakte Kalibration als problemlos möglich erwiesen.
Um später die Funktionsweise der Software erläutern zu können, möchten wir zunächst den Hardwareaufbau im Überblick beschreiben. Dieser Abschnitt folgt dabei den Bauplänen aus dem RepRap-Wiki sowie dem RepRap Mechanik-Repository (Bowyer et al., 2012b,a).

2.1 Mechanischer Aufbau

Um das aufgeschmolzene Kunststofffilament präzise verteilen zu können, muss der Druckkopf gegenüber dem Werkstück dreidimensional positionierbar sein. Der RepRap Mendel besteht aus einer stabilen Rahmenkonstruktion aus M8 Gewindestangen mit einem in y-Richtung beweglichen Druckbett. Darauf wird das Werkstück extrudiert. Der Extruder mit angebautem Hotend ist in x-Richtung beweglich und die gesamte x-Achse kann relativ zum Rahmen auf Gewindestangen in z-Richtung präzise positioniert werden. Dieses Prinzip ist in Abbildung 2.1 veranschaulicht.
Die erforderlichen Linearlager werden durch günstige Kugellager realisiert, die auf zweckmäßig geformte Kunststoffteile geschraubt werden. Durch eine Anordnung von zwei mal drei Kugellagern ergibt sich ein Linearlager, das durch eine Stellschraube justiert werden kann, sodass es spielfrei und gleichzeitig leicht läuft.

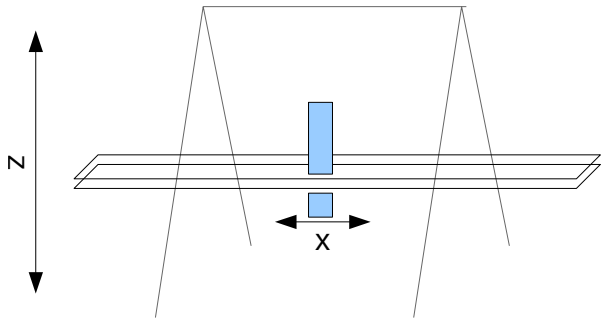


Abbildung 2.1: Funktions-Skizze der mechanischen Grundkonstruktion. Das Druckbett bewegt sich entlang der y-Achse senkrecht zur Zeichenebene.

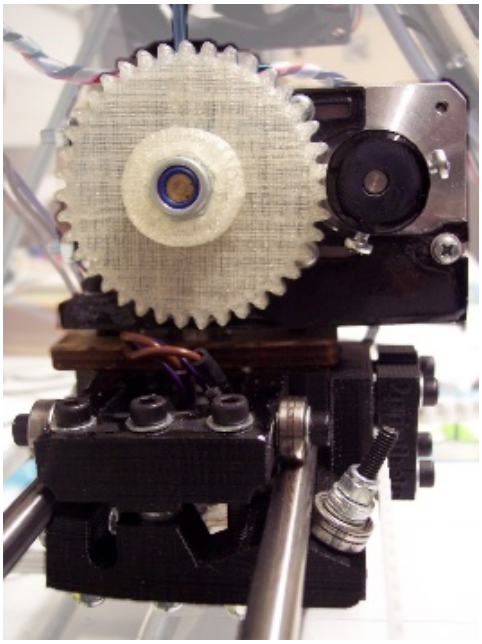
Die gedruckten oder gegossenen Kunststoffteile werden also zunächst zu Baugruppen verschraubt. Gleichzeitig werden die Kugellager angeschraubt. Eine detaillierte Anleitung ist im RepRap-Wiki verfügbar. Je nach Fertigungstoleranzen müssen einige Kunststoffteile mit der Feile nachbearbeitet oder Löcher nachgebohrt werden. So erhält man Lager- und Motorblöcke für alle beweglichen Teile, die auf 8mm-Rundstahl laufen.
Zuletzt werden diese Baugruppen im Rahmen montiert und justiert. Die Motoren und Sensoren sind mit einer Steuerungsplatine verkabelt, wofür es wiederum mehrere quelloffene Varianten gibt. Wir entschieden uns für eine Ein-Prozessor-Lösung namens „RAMPS“, die auf der Arduino-Plattform aufsetzt und A4988-basierte StepStick-Schrittmotortreiber verwendet. Die Verkabelung ist gut dokumentiert und aufgrund des quelloffenen Platinenlayouts auch vollständig nachvollziehbar. Zum Anschluss sind gebräuchliche Stift-/Buchsenleisten vorhanden; Spezialstecker sind nicht notwendig.

2.2 Kalibration der Schrittmotoren

Die eingesetzten NEMA-17 Schrittmotoren haben eine Auflösung von 1,8° und lassen sich von den StepStick-Treibern mit 1/16-Mikroschritten ansteuern. Die x- und y-Achse werden direkt mit 5mm-Zahnriemen angetrieben; bei Antriebszahnrädern mit acht Zähnen ergibt sich eine theoretische Auflösung von $\frac{8 \cdot 5\text{mm/Umdrehung}}{16 \cdot 200\text{Schritte/Umdrehung}} = 0{,}0125\,\text{mm/Schritt}$. Umgekehrt bewegt sich der Motor um 80 Mikro-Schritte, um den Druckkopf einen Millimeter weit zu bewegen.
In z-Richtung erfolgt der Antrieb über M8-Gewindestangen (Steigung 1,25 mm). Diese bewegen die x-Achse um 1,25 mm pro Umdrehung. Zum parallelen Antrieb beider Gewindestangen sind diese mit einem Zahnriemen verbunden, der gegenüber dem Motor eine Übersetzung von 20/8 realisiert. Der Umrechnungsfaktor von Schritten in Millimeter ergibt sich damit zu:

$$\frac{16 \cdot 200\,\text{Schritte/Umdrehung} \cdot \frac{20}{8}}{1{,}25\,\text{mm/Umdrehung}} = 6400\,\text{Schritte/mm}.$$

Etwas schwieriger gestaltet sich die Kalibration des Extruders. Der Extruder wird ebenfalls von einem Schrittmotor angetrieben, der über ein Getriebe eine spezielle Vorschubschraube bedient (Abbildung 2.2(a)). Das Kunststofffilament wird von einem Kugellager gegen diese Schraube gepresst. Die PC-Software kann die Distanz des einzuziehenden oder zu extrudierenden Kunststoffes berechnen. Für polylactic acid (PLA) bleibt das Volumen erhalten – auch nachdem es erhitzt und extrudiert wurde – sodass es inzwischen üblich ist, die Länge des eingezogenen Rohmaterials (in Millimeter) zu messen. Wir haben also den Kalibrationsfaktor zwischen Schrittmotor-Mikroschritten und Filament-Distanz zu ermitteln. Um einen groben Anhaltspunkt zu haben, starten wir mit einer Überschlagsrechnung, um den Wert anschließend empirisch zu überprüfen und zu präzisieren.
Die Filamentvorschubschraube hat einen Durchmesser von ungefähr $6 \pm 0{,}5$ mm (siehe Abbildung 2.2(b)). Bei einer Getriebeübersetzung von 39/11 erhalten wir einen Kalibra-



(a) Getriebeübersetzung des *Wades Extruder*. Rechts befindet sich der Motor, das Filament wird von der linken Schraube von oben nach unten bewegt.

(b) Filament-Vorschubschraube mit großem Zahnrad

Abbildung 2.2: Wades Extruder

tionsfaktor von

$$\frac{16 \cdot 200 \cdot 39/11\,\text{Schritte}}{\pi \cdot 6\,\text{mm}} \approx (600 \pm 50)\,\text{Schritte/mm}.$$

Als Mittelwert mehrerer Messungen ergibt sich ein Faktor von 558 Schritte/mm.
Die Ansteuerung der Schrittmotoren erfolgt über StepStick-Boards, die eine A4988 Treiberstufe verwenden. So ist eine einfache Kontrolle der Schrittmotoren möglich. Der Mikrocontroller gibt lediglich die Richtung vor und bei anliegendem Enable-Signal werden Schrittimpulse direkt von den Motoren umgesetzt. Die Einstellung der maximalen Stromstärke ist mittels eines Mini-Potentiometers möglich. Das Pinout der StepStick-Boards ist in Abbildung 2.3 dargestellt.

2.3 Sensorik

Als Sensoren werden nur drei optische Endstops verwendet, um die Anfangspositionen der Achsen zu markieren. So ist eine reproduzierbare Startposition zu Beginn eines Druckauftrags möglich. Daneben sind ein oder mehrere Temperatursensoren vorgesehen, um die Temperatur des Extruders und des optionalen Druckbetts zu regeln.
Weitere Positions- oder Qualitätssensorik ist nicht eingeplant, sodass der Controller sich darauf verlassen muss, dass die Aktoren sich wie vorgegeben verhalten. Bei unseren

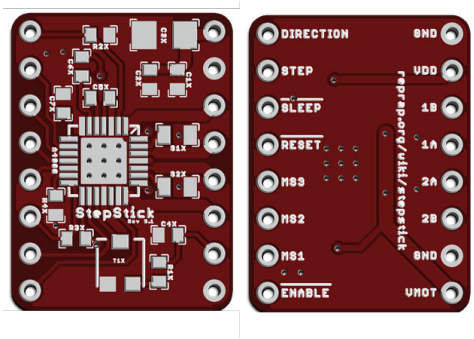


Abbildung 2.3: StepStick Schrittmotor-Treiber Board Layout. Für eine einfache Ansteuerung müssen vom Mikrocontroller nur die Pins „Step“ und „Direction“ gesteuert werden.

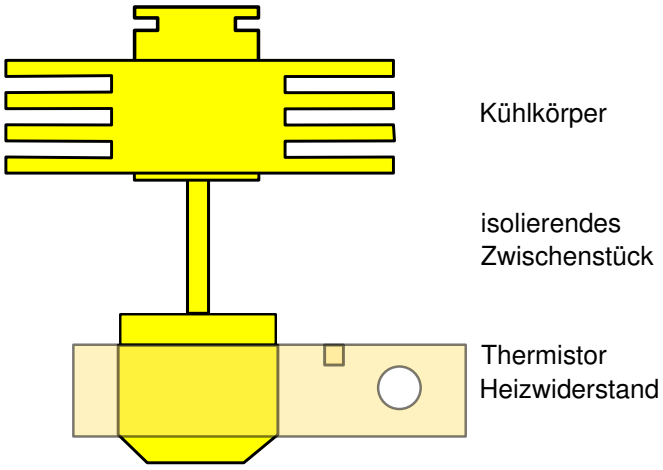


Abbildung 2.4: Schematischer Aufbau des Hotends Version 4.0 von Arcol.hu. Das Filament wird von oben vom Extruder eingeführt und schmilzt nur am unteren Ende im geheizten circa 10 mm langen Bereich.

Tests stellten wir fest, dass die Schrittmotoren durch die aus Kugellager aufgebauten Linearlager in der Tat sehr reproduzierbar funktionieren. Lediglich zur automatischen Fehlererkennung wären weitere Sensoren denkbar. Es können sich beispielsweise Schrauben mit der Zeit lösen und zu einem Ausfall einer der Bewegungsachsen führen, ohne dass der Controller dies feststellen kann. Zum längeren unbeaufsichtigten Betrieb ist der 3D-Drucker – auch wegen der Unzuverlässigkeit des Extruders – momentan nicht geeignet.

2.4 Hotend

Bei dem verwendeten Hotend Version 4.0 von arcol handelt es sich um eine noch sehr neue Entwicklung. Durch das dünne Zwischenstück schmilzt der Kunststoff nur am

untersten Ende auf einer Länge von circa 10 mm (bei aktiver Kühlung mit einem PC-Lüfter). Bei einem Testbetrieb ohne Lüfter verklebte das Filament weit oben, woraufhin ein weiterer Vorschub unmöglich war. Für solche Fälle lässt sich das Hotend komplett zerlegen und reinigen.
Probleme hatten wir vor allem mit dem sehr dünnwandigen Zwischenstück, das zur Isolierung dienen soll. Um die Wärmeleitung vom heißen zum kalten Ende zu minimieren, ist dieses Teil mechanisch sehr dünn gefertigt, was zu einer Bruchstelle bei zu festem Eindrehen beim Zusammenbau oder bei einer Kollision des Hotends mit zuvor gedrucktem Kunststoff führt.
Die Druckqualität des Hotends hat uns dagegen positiv überrascht; die mit 0,5 mm Durchmessser sehr feine Düse verteilt den Kunststoff präzise und zuverlässig selbst bei hoher Druckgeschwindigkeit (bis 40 mm/sec in x- und y-Richtung).

3 Software

Die Verarbeitung eines 3D-Modells umfasst drei Stufen:

1. Modellierung mit einer CAD-Software (OpenSCAD, Blender, ...)
2. Umwandlung in Maschinenbefehle (Druckbewegungen)
3. Übertragen der Bewegungsbefehle vom PC zum Drucker (Ausführung)

Für den Entwurf des 3D-Modells existieren zahlreiche freie und proprietäre Softwarelösungen. Mit allen gängigen CAD-Programmen lässt sich die Zeichnung als Dreiecksgitter im Surface Tessellation Language (STL)-Format exportieren. Wir werden daher im Folgenden nur auf Schritte zwei und drei näher eingehen und die Berechnung von druckbaren Ebenen sowie die Maschinenansteuerung (Firmware) beschreiben. Das Kapitel 4.3 zeigt dann einen Kunststoff-Greifer als Anwendungsbeispiel und betrachtet auch dessen Konstruktion beispielhaft mit der 3D-CAD Software OpenSCAD.
Als Befehlsformat zur Maschinensteuerung wird die Beschreibung mittels GCode-Befehlen verwendet. Diese Kommandos sind im CNC-Bereich üblich und wurden für den Einsatz im 3D-Drucker entsprechend ergänzt. Die vollständige Liste gültiger Befehle befindet sich im RepRap-Wiki[1]. Diese Befehle werden auf unterster Ebene von einem Mikrocontroller verarbeitet, der die Motorsteuerung und Sensorauswertung übernimmt.

3.1 Firmware: Schrittmotorsteuerung und Sensorauswertung

Auf der Arduino-Plattform steht ein ATmega 1280 Mikrocontroller zur Verfügung, um die Schrittmotoransteuerung, Sensorauswertung und Kommunikation mit dem Computer zu übernehmen. Wir entschieden uns für die Firmware „Sprinter“ (Yanev, 2012), die für 3D-Drucker mit verschiedenen Elektronikvarianten einsetzbar ist.
Einige freie Parameter müssen vor der Kompilation angepasst werden. Dies sind unter anderem die Umrechnungsfaktoren von Schrittmotorschritten in Millimeter (siehe Kapitel 2.2) sowie eine Thermistortabelle zur Umrechnung der gemessenen Spannung in eine Temperatur. Für verschiedene Typen von Wärmewiderständen stehen solche Tabellen mit der Firmware schon zur Verfügung, sodass lediglich der Typ ausgewählt werden musste. Die Drehrichtung der Schrittmotoren lässt sich ebenfalls in der Konfiguration ändern sowie zusätzliche optische Maximums-Endstops aktivieren.
Die Firmware interpretiert einen großen Teil des GCode-Befehlssatzes, der auf Bowyer et al. (2012b) vorgestellt ist. Die Befehle erlauben eine Angabe von Position und Geschwindigkeit, mit der sich der Druckkopf bewegen soll. Für CNC-Fräsen ist diese Kommunikation etabliert; von der RepRap-Community wurden manche Befehle zum

[1] Liste der GCodes: http://reprap.org/wiki/G-code

3D-Drucken erweitert oder ergänzt. Dagegen entfallen die Rotationsfreiheitsgrade einer computergesteuerten Fräsmaschine beim 3D-Drucker. Die wichtigsten Befehle haben wir im Anhang in Tabelle A.1 zusammen gestellt.

Listing 3.1 zeigt beispielhaft eine Routine zum Setzen der Temperatur, bevor der eigentliche Druck beginnt. Jede Zeile beginnt dabei mit einem Bewegungsbefehl (G##) oder Parameterbefehl (M##). Anschließend folgen optionale Parameter. Kommentare, die auf ein Semikolon folgen werden von der aktuellen Version der Firmware ignoriert (ein dazu notwendiger Bugfix ist im Anhang A.3 aufgelistet). Zusätzlich kann eine Zeilennummer und XOR-Checksumme übertragen werden, um Übertragungsfehler auszuschließen. Fehlerhaft empfangene Zeilen kann der Controller vom PC neu anfordern (re-send).

```
M104 S195 ; set temperature
M106 ; fan on
G28 Y ; home all
G28 Z
G28 X
G1 F1200 ; set feedrate fast
G1 X90 Z20. ; move head up
M105 ; read temperature
M109 S195 ; wait for temperature
G92 E0
G1 F72 ; set feedrate for extruder
G1 E19.2 ; extrude for several seconds
```

Listing 3.1: Einfaches GCode-Beispiel zum Vorheizen des Druckkopfes und Test des Filamentvorschubs

Alle Befehle überträgt der Computer im Klartext über die serielle Verbindung an den Controller. Nach jeder Zeile quittiert der Mikrocontroller die Nachricht mit einer entsprechenden Antwort. Der dadurch entstehende Overhead verzögert die Kommunikation zwar, führt aber zu einer leichten Lesbarkeit und ermöglicht eine manuelle Steuerung per direkter Befehlseingabe über ein serielles Terminal.

3.2 Druckersteuerung: Live-Kontrolle und Übertragen von GCode-Dateien

Die Übertragung von einzelnen Befehlen kann direkt von einem seriellen Terminal aus gestartet werden. Um die Funktionen der Firmware zu testen, starteten wir mit *minicom*. Für den zuverlässigen Betrieb des Druckers ist jedoch eine komfortablere Software notwendig, die auch auf zurück gemeldete Übertragungsfehler automatisch reagieren kann. Eine grafische Benutzeroberfläche vereinfacht darüber hinaus die Bedienung. Dafür existieren aktuell mehrere gute Lösungen, die sich im Funktionsumfang unterscheiden und von denen drei hier kurz vorgestellt werden sollen. Ein direkter Vergleich ist in Tabelle 3.1 zusammen gefasst.

	Live-Steuerung	GCode-Datei senden	Druck von SD-Karte	Pause und manuelle Steuerung	Abschätzung der Druckdauer	Umwandlung STL in GCode
Skeinforge http://fabmetheus.crsndoo.com/	–	(Ja)	–	–	grob	Ja
Terminal (Minicom, gcdump)	Ja	Ja	manuell	–	–	
RepRapHost http://github.com/reprap/release	Ja	Ja	–	–	Ja	Ja
ReplicatorG http://replicat.org/	Ja ohne Extruder	Ja	?	–	–	–
Printrun Pronterface http://github.com/kliment/Printrun	Ja	Ja	Ja	Ja	Ja	–

Tabelle 3.1: Verschiedene Software-Lösungen im Vergleich

Java Host Die ursprüngliche Host-Software von Bowyer et al. (2011) (Abbildung 3.1) ist in Java geschrieben und erlaubt sowohl die Umwandlung von STL-Oberflächen in GCode als auch den Druck dieser erzeugten Datei. Daneben steht eine Live-Steuerung zur Verfügung, die allerdings noch nicht auf die volumetrische Extruder-Konvention eingestellt ist, sodass die Geschwindigkeitseinstellungen angepasst werden müssen. Die Interpretation der Firmware-Antworten ist sehr strikt, sodass am Sprinter-Code mehrere Anpassungen notwendig werden [2]. Positiv hervorzuheben ist der Debug-Modus der alle Befehle ausgibt, die seriell versendet werden. Außerdem enthält das RepRap-Wiki eine gute Einführung und Installationshinweise.

ReplicatorG ReplicatorG ist ebenfalls in Java geschrieben und erlaubt zusätzlich eine Drehung und Skalierung von Objekten, die man als STL-Datei lädt. Die Ausgabe von GCode an den Drucker kann unterbrochen und fortgesetzt werden, allerdings funktionierte bei unseren Tests während der Pause die Livesteuerung nicht. Die Umwandlung von STL-Dateien in GCode greift auf Skeinforge zurück; allerdings werden nur wenige Optionen abgefragt, sodass der Benutzer schneller ein druckbares Ergebnis erhält.

Printrun: Pronterface Die Python-Software Printrun enthält eine generische Druckersteuerung, die über eine Kommandozeile (Pronsole) oder über eine grafische Oberfläche (Pronterface) basierend auf wxGtk zur Verfügung gestellt wird. Die Software stammt von Kliment – wie auch die Firmware Sprinter; so ist eine reibungslose Kommunikation mit den im Mikrocontroller implementierten Funktionen möglich. Neben einer intuitiven

[2] https://github.com/kliment/Sprinter/pull/135 fasst die notwendigen Anpassungen zusammen

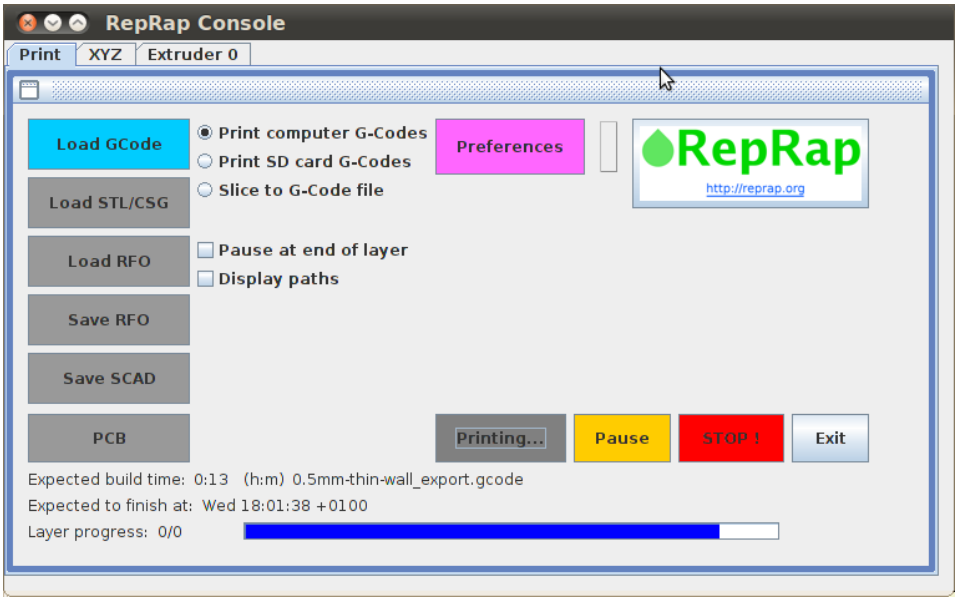


Abbildung 3.1: Java Host Konsole zur Auswahl einer STL- oder GCode-Datei.

Livesteuerung wird zusätzlich der Zugriff auf die SD-Karte unterstützt. Abbildung 3.2 zeigt einen Screenshot der Benutzeroberfläche von Pronterface.

3.3 Skeinforge: Vom 3D-Modell zu Druckbewegungen

Die Java-Host-Software von Bowyer et al. (2011) erlaubt die Umwandlung von STL-Oberflächen in Maschinenbefehle. Die dortigen Einstellungsmöglichkeiten sind jedoch schlecht dokumentiert. Da sich einige unserer Testobjekte mit der Host-Software nicht laden ließen, entschieden wir uns schnell für Skeinforge (Perez, 2012). Die in Python geschriebene Toolchain erlaubt die Anpassung vieler einzelner Parameter, die alle in einer HTML-Dokumentation erläutert sind. Die graphische Benutzeroberfläche ist gewöhnungsbedürftig, da keine Unterscheidung zwischen wichtigen und sekundären Optionen gemacht wird.
Die Toolchain Skeinforge ist eine Sammlung von Python-Skripten, die nacheinander ausgeführt werden. Zunächst wird der Umriss berechnet, dann gibt es Optionen zum Auffüllen im Inneren des Druckobjektes. Danach kann der Code von optionalen Modulen wie 'Speed', 'Skin' oder 'Lash' korrigiert werden, um die bestmögliche Genauigkeit und größtmögliche Geschwindigkeit im Druck zu erreichen. Allerdings sind die Module nicht vollständig unabhängig, sodass manche Parameter nur relevant werden, wenn ein anderes Modul aktiv ist (z.B. wird die maximale z-Geschwindigkeit unter 'Speed' nur berücksichtigt, wenn das Modul 'Limit' aktiv ist).
Um ein erstes Druckergebnis zu erhalten, reichen die Default-Einstellungen leider nicht aus. Tabelle 3.2 gibt einen Überblick, welche Parameter auf jeden Fall überprüft werden sollten. Die zentralen Parameter *Durchmesser des Filaments* im Modul *Carve* und *Höhe der Ebenen* im Modul *Dimensions* müssen eingestellt werden. Verwendet man

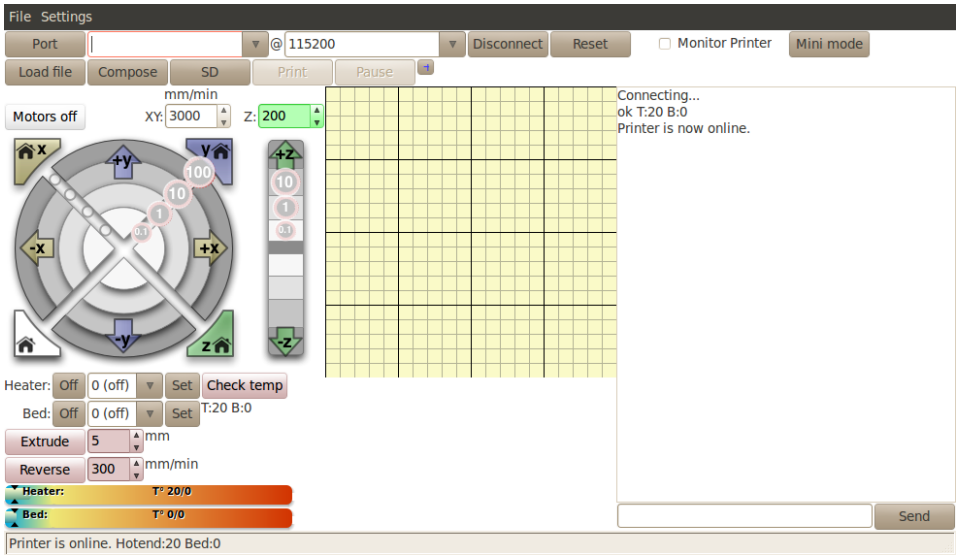


Abbildung 3.2: Printrun: Pronterface Benutzeroberfläche

die volumetrische Extruderkonvention (und misst die Länge des eingezogenen Filaments statt der Länge des extrudierten Strings), sind die Parameter Feedrate und Flowrate jeweils auf denselben Wert zu setzen. Die Standardeinstellungen berücksichtigen diese Vereinfachung leider nicht. Die Parameter *Perimeter Width over Thickness* und *Infill Width over Thickness* sind ebenfalls von Bedeutung[3]; beide Parameter sollten im Bereich von 1,2–1,8 liegen. Beide Parameter werden von Skeinforge zur Ermittlung des zu extrudierenden Volumens herangezogen.
Das Filament verlässt die Düse des Hotends in Zylinderform mit einem Durchmesser von 0,5 mm. Der abgelegte Kunststoff-Strang wird jedoch oval verformt und hat nur noch die als Ebenenhöhe eingestellte Höhe und eine entsprechend größere Breite. Dieser Asymmetriefaktor wird für den Umfang des Objekts als *Perimeter Width over Thickness* angegeben. Zur Kalibration eignet sich ein dünnwandiges Objekt; letzterer Faktor wird so lange reduziert, bis gerade noch eine geschlossene Wand entsteht. Für die Füllung steht ein eigener Parameter *Infill Width over Thickness* zur Verfügung. Bei vollständiger Füllung eines Quaders kann dieser Faktor empirisch so lange erhöht werden, bis der Kunststoff gerade eine glatte Oberfläche bildet und noch nicht über die Oberfläche überquillt.
Nach den ersten erfolgreich gedruckten Objekten bietet sich der Druck von weiteren Kalibrationsobjekten [4] an, um genauere Werte sowie die richtige Drucktemperatur festzustellen. Die gedruckten Resultate, die der Mendel 3D-Drucker bauen kann, möchten wir im nächsten Kapitel betrachten.

[3] Startwerte haben wir mit Hilfe des Skripts von `http://makerblock.com/profilemaker/` ermittelt
[4] `http://reprap.org/wiki/Calibration` gibt hilfreiche Tipps zur Kalibration

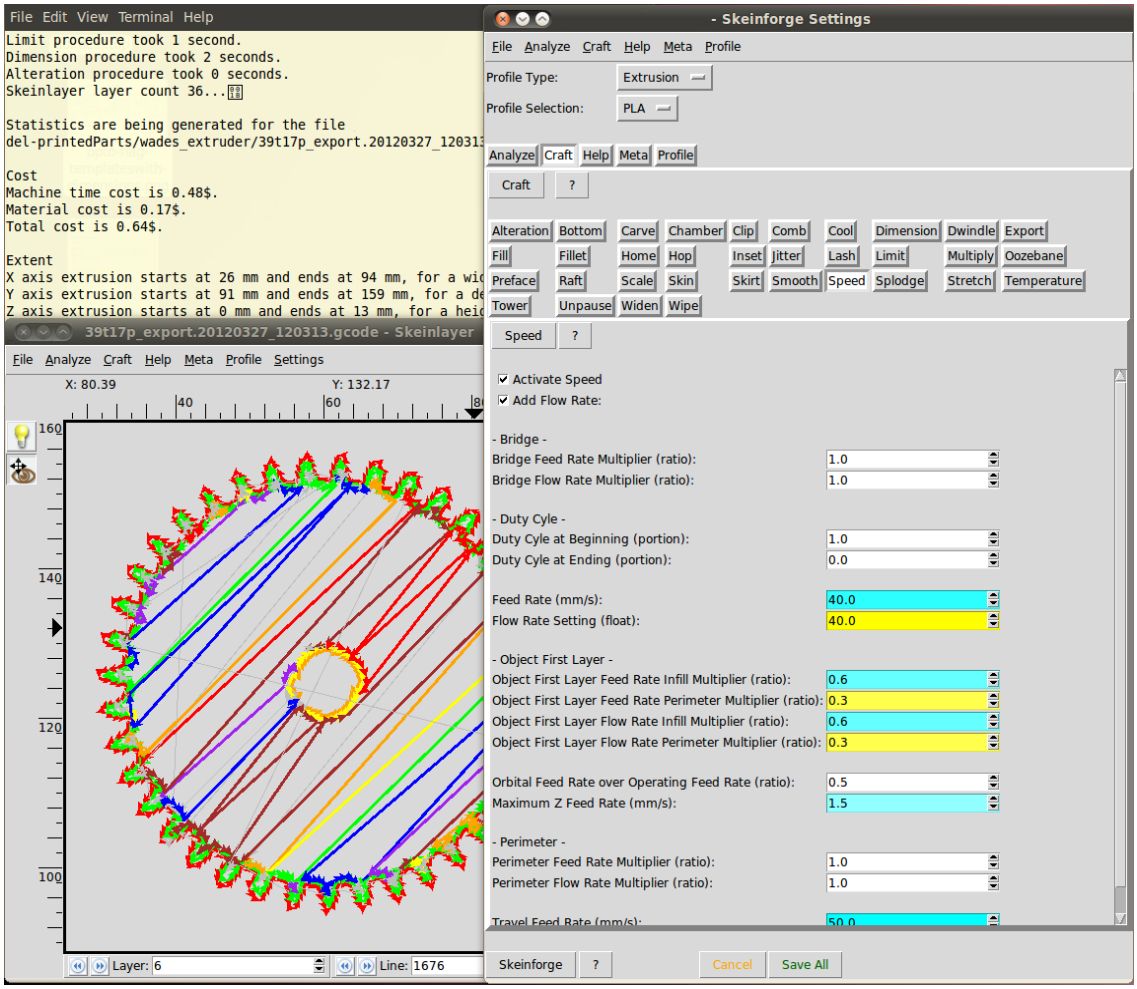


Abbildung 3.3: Die grafische Oberfläche von Skeinforge erlaubt eine einfache Parametereingabe. Die generierten Druckbewegungen werden als Linienfolge dargestellt und eine Textausgabe gibt statistische Informationen zum Druck.

Modul	Parameter	Wert	Beschreibung
Dimension	Filament Diameter	2.9	Durchmesser des Rohmaterials
Carve	Layer height	0.36	z-Auflösung
Carve	Edge width over Height	1.38	Linienbreite
Inset	Infill width over Thickness	1.38	Breite für Füllung
Temperature	xxx Temperature	185-220	verschiedene Hotend-Temperaturen
Fill	Infill Solidity	0.0-1.0	Plastikanteil im Innern des Objekts
Multiply	Center X,Y	60,100	Mittelpunkt auf dem Druckbett
Speed	Feed Rate, Flow Rate	30-50	Geschwindigkeit des Druckkopfes
Cool	Minimum Layer Time	8-20	Mindestdruckdauer pro Ebene

Tabelle 3.2: Parameter in Skeinforge: Die wichtigsten materialabhängigen Grundeinstellungen (oben) sowie häufige Feineinstellungen je nach gedrucktem Objekt (unten).

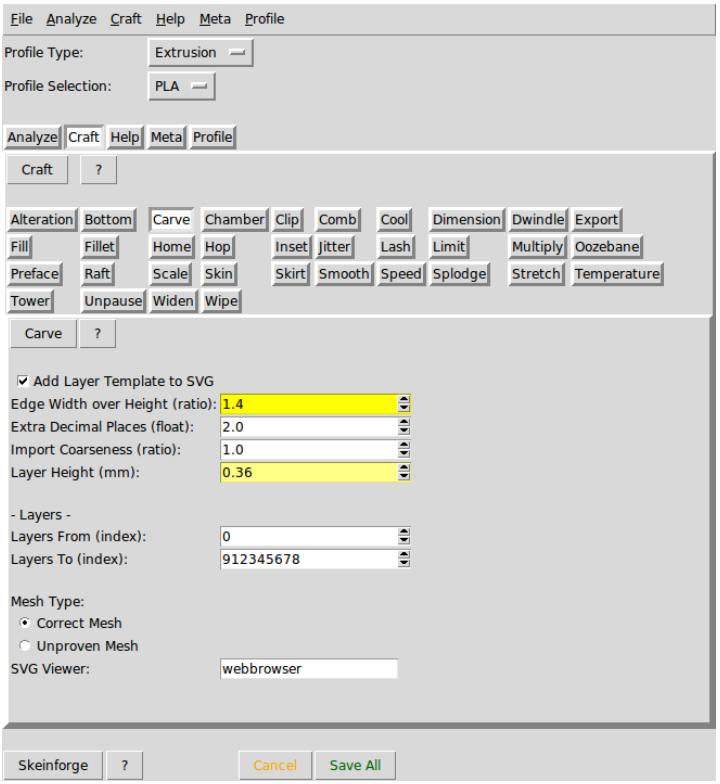


Abbildung 3.4: Das Modul *carve* enthält eine der Grundeinstellungen: Die Ebenenhöhe bestimmt die z-Auflösung des gedruckten Objekts. Weitere wichtige Einstellungen sind in Tabelle 3.2 zusammengestellt.

4 Anwendungen und Erweiterungen für unseren RepRap Mendel

Mit der nun bekannten Software und funktionierender Kommunikation zwischen PC und Mikrocontroller können einfache dreidimensionale Objekte gedruckt werden; allerdings treten im Detail noch verschiedene Artefakte auf. Einige Einstellungen lassen sich kalibrieren, sodass dann kompliziertere Objekte ausprobiert und schließlich eigene Designs entworfen werden können.

4.1 Inbetriebnahme und Kalibration

Nachdem wir die Bewegung der einzelnen Achsen, die Temperatursensoren und Heizung sowie die optischen Endstops schon einzeln getestet haben, ist die Inbetriebnahme des Druckers nicht mehr schwer. Ein Foto des ersten Quaders ist in Abbildung 4.1 zu sehen. Es ist jedoch offensichtlich, dass einige Parameter noch angepasst werden können.

Hotend-Temperatur Wie oben schon angedeutet ist die Temperatur ein kritischer Parameter während des Druckprozesses. Die Düse muss heiß genug sein, um das Filament ohne Widerstand zu extrudieren; andererseits schmelzen bei zu hoher Temperatur die vorher gedruckten Ebenen wieder auf und sorgen für ein „verschmiertes“ Druckergebnis. Außerdem ist anzumerken, dass mit dem verwendeten arcol.hu-Hotend ein Lüfter zwingend notwendig ist, wenn mit PLA gedruckt werden soll. Erste Versuche ohne aktive Kühlung führten stets zum Schmelzen des Filaments weit oben im kalten Teil des Hotends (siehe schematische Abbildung 2.4 auf Seite 12). Die Kühlrippen des Hotends müssen also stets einem Luftstrom ausgesetzt sein. Der im Rahmen befestigte Compu-

Abbildung 4.1: Hohlquader gedruckt mit Default-Einstellungen der Java-Host-Software (links) und mit kalibrierten Parametern in Skeinforge (mitte und rechts)

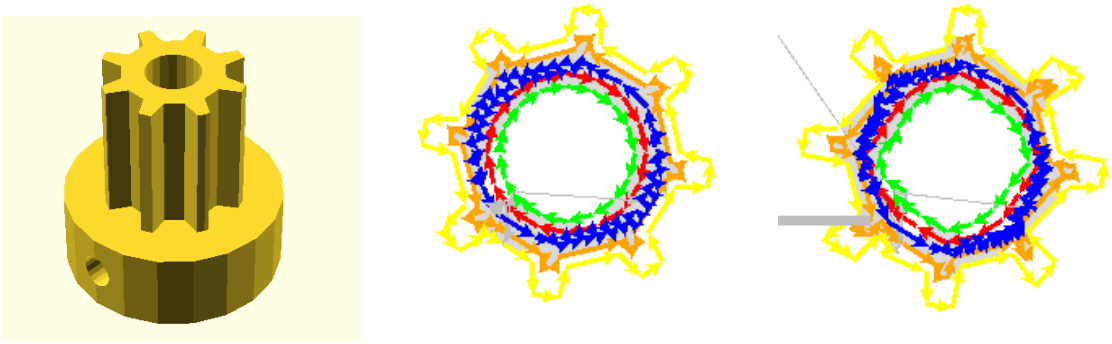
terlüfter bewirkt außerdem ein schnelleres Abkühlen des gedruckten Kunststoffes und erhöht damit die Druckqualität.

Kunststofffeder für z-Endstop Als eine der ersten Erweiterungen haben wir eine Feder für die Halterung des z-Endstops nachgedruckt, die nicht im gekauften Set enthalten war. Im Originalbauplan ist diese Feder enthalten, damit der optische Endstop mit nur einer Schraube eingestellt werden kann. Dies ist eine sinnvolle Erweiterung und erleichtert die Einstellung des Druckkopfabstands zum Druckbett. In der ersten Ebene sollte der Abstand möglichst klein sein, damit der Kunststoff gut an der Oberfläche haften bleibt. Zur Einstellung legt man ein gefaltetes Blatt Papier zwischen Druckkopf und Druckbett. Am minimalen z-Anschlag sollte der Abstand so klein ein, dass ein deutlicher Widerstand zu spüren ist, sich das Papier aber noch bewegen lässt.

Abkühlen Nach einigen Experimenten auf verschiedenen Untergrundmaterialien stellte sich heraus, dass die Temperatur für die unterste Ebene deutlich höher gewählt werden kann, um einen guten Kontakt mit dem Klebeband herzustellen. Der Kunststoff wird so dünnflüssiger und legt sich besser auf den Untergrund. Ab der folgenden Ebene kann die Temperatur niedriger gewählt werden, damit der gedruckte Kunststoff schnell fest wird. Unsere für PLA verwendeten Temperaturen sind unter den Skeinforge-Einstellungen in Tabelle A.2 (Seite 33) zusammengefasst.
Das Skeinforge-Modul *cool* muss aktiviert werden, um eine Minimalzeit pro Ebene vorzugeben. Bei sehr kleinen Objekten wird der Druckkopf dann um das Objekt herum bewegt, bis der Kunststoff fest geworden ist. Alternativ kann die Geschwindigkeit des Drucks verlangsamt werden, sodass eine Ebene mindestens 10-12 Sekunden dauert.

Verhindern von Fäden Auch ohne Vorschub tropft die Düse des Druckkopfes gewöhnlich weiter. Dies führt zu Fäden im gedruckten Resultat oder kann während *cool*-Phasen dazu führen, dass Kunststoff „verloren geht“, der außerhalb tropft und im gedruckten Objekt dann fehlt. Um diese Fäden zu vermeiden helfen zwei Einstellungen, die abhängig von der verwendeten Farbe und Konsistenz des Filaments angepasst werden können: Im Modul *dimension* konfiguriert man den Extruder, sodass dieser das Filament zum Beginn eines solchen Leerlaufs zurück zieht. So sinkt der Druck im Hotend und es können nur noch kleine Reste nachtropfen. Eine zusätzliche Einstellungsmöglichkeit bietet das Modul *Oozebane*, das die Geschwindigkeit kurz vor dem Absetzen verlangsamt. Die dort vorhandene Funktion zum langsamen Start führte bei uns zu Artefakten in der ersten Druckebene, sodass wir nur die Funktionalität des langsamen Stops verwenden. Im Zusammenspiel von *Oozebane* und *Retraction* lassen sich Fäden im gedruckten Ergebnis fast vollständig vermeiden.

Umkehrspiel Der Zahnriemenantrieb erlaubt zusammen mit den gut gelagerten Achsen eine Reproduzierbarkeit der Position über viele Stunden. Allerdings werden die Zahnriemen nur zwischen zwei Kunststoffteile eingespannt und so bleibt auch bei gut gespannten Riemen ein Umkehrspiel bestehen. Das Umkehrspiel in den Zahnriemen ist



(a) 3D-Modell (b) Skeinforge-Export ohne Lash (c) Skeinforge-Export mit Kompensation des Umkehrspiels

Abbildung 4.2: Zahnrad für verwendeten 5mm-Zahnriemen. Beim Druck der Zähne fällt ein Umkehrspiel im Antriebsriemen besonders deutlich auf.

deutlich größer als erreichbare Präzision der Schrittmotoren mit Mikroschritten von 0,0125 mm/Schritt, sodass eine Berücksichtigung in der Software sinnvoll ist. Allerdings ist das Umkehrspiel schwierig zu messen, sodass wir ein kleines Zahnrad als Kalibrationsobjekt verwendet haben (vgl. Abbildung 4.2). Um die Zahnräder für den Riemenantrieb zu drucken, bewegt sich der Kopf bei jedem Zahn hin und her, sodass hier das Umkehrspiel besonders ins Gewicht fällt. Anschließend lässt sich die Asymmetrie in x-/y-Richtung bestimmen und durch ein einberechnetes Umkehrspiel von bis zu 0,3 mm kompensieren.

4.2 Objekte von Thingiverse

Die Webseite „Thingiverse“ (Smith und Pettis) enthält eine Vielzahl von Objekten, die unter einer freien Lizenz zur Verfügung gestellt werden. Es handelt sich um Reparatur- oder Ergänzungsteile für 3D-Drucker oder um eigenständige Designs, die als STL sowie teilweise als parametrisches Modell ausgetauscht werden.
Um die Fähigkeiten des 3D-Druckers zu testen, bieten sich solche fertigen Objekte ideal an. Im Folgenden sind einige Druckergebnisse abgebildet. Der Druck einer Hälfte der Nautilus-Schnecke dauert gut eine Stunde (Abbildung 4.3).
Ein Abbrechen des Filaments oberhalb des Extruders war anfangs problematisch und lässt sich durch Verwendung eines „Filament Guide“ verhindern. Nach einem Tipp vom Lieferant unseres Mendel-Bausatzes drucken wir eine Führungsöse, die in Abbildung 4.4 dargestellt ist (Krautwasser und Wenger, 2011–2012).
Schließlich ersetzen wir die provisorische Lüfteraufhängung aus Draht durch gewinkelte Kunststoffklemmen am Rahmen. Die Winkel sind mit der freien 3D-Grafiksoftware Blender[1] modelliert, sodass sie mit Schrauben an Klemmen befestigt werden können,

[1] http://www.blender.org/

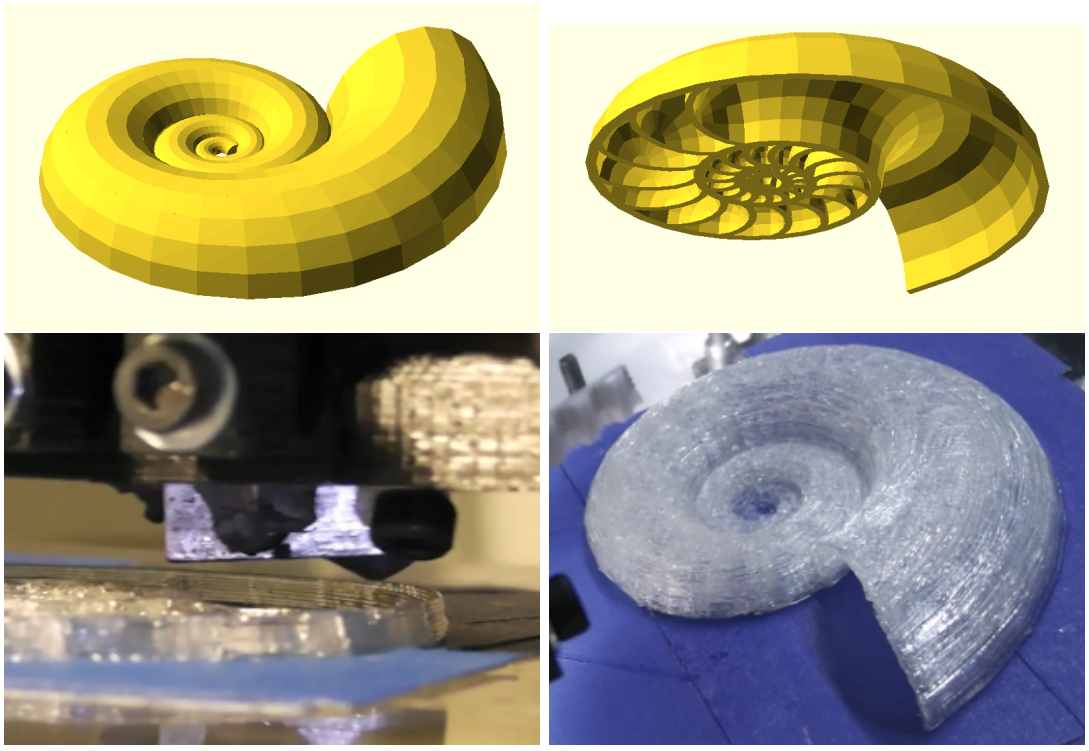


Abbildung 4.3: Nautilus-Schnecke von polymaker im Modell (oben) und ausgedruckt (unten), siehe http://www.thingiverse.com/thing:13829.

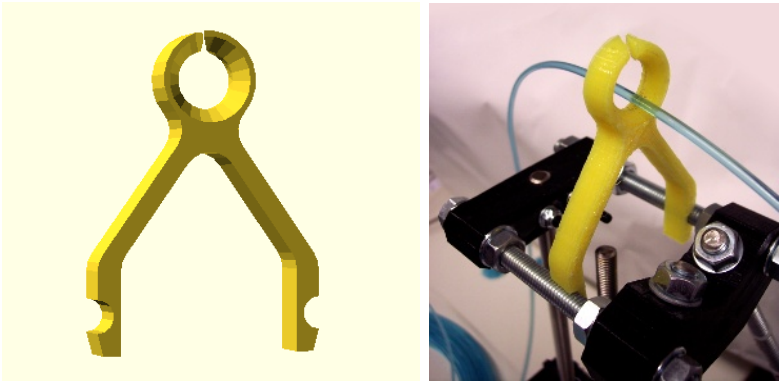
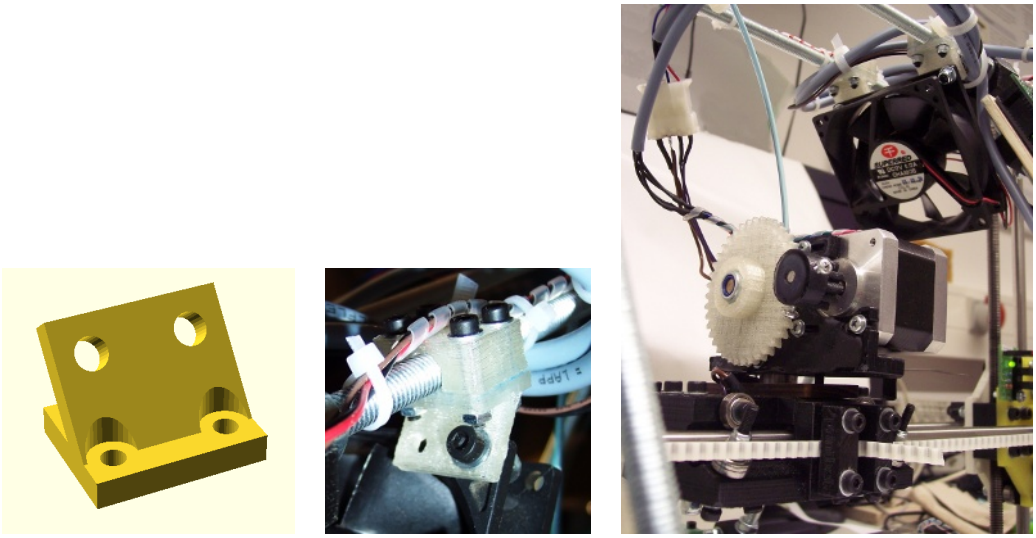


Abbildung 4.4: Filament-Guide von *toomanyplugs*, siehe http://www.thingiverse.com/thing:11521.



(a) Modell

(b) gedruckt und befestigt am Rahmen

(c) Lüfter im Hintergrund; im Vordergrund ist der Extruder zu sehen.

Abbildung 4.5: Selbst entwickelter Winkel zur Befestigung des Lüfters am Rahmen des RepRap.

wie sie auch für die Befestigung der Elektronik-Boards verwendet werden (Abbildung 4.5).

4.3 In OpenSCAD zum Kunststoff-Greifer

Am Beispiel des Kunststoff-Greifers, der im Rahmen dieser Arbeit entworfen und gedruckt wurde, lässt sich die komplette Toolchain des 3D-Drucks anschaulich zusammenfassen.
Ein Greifarm-Roboter ist mit parallel schließenden „Fingern“ ausgestattet und soll nun in der Lage sein, ein mit Wasser gefülltes Glas zu balancieren. Die dafür nötige Halterung soll an den Fingern befestigt werden – nach Möglichkeit, ohne diese zu beschädigen oder dauerhaft zu verändern. Dafür bietet sich das Rapid-Prototyping mit unserem 3D-Drucker an und wir haben in enger Zusammenarbeit mit Benjamin Reh ein Kunststoffteil entworfen und gedruckt. Zunächst musste das Glas und die vorhandenen Finger modelliert werden; wir entschieden uns dazu für die Software OpenSCAD (siehe Listing 4.1). OpenSCAD erlaubt eine textuelle Beschreibung des Objekts, wobei Primitive für Würfel, Kugeln und Zylinder zur Verfügung stehen. Verschiedene Objekte können transformiert und mit boole'schen Operatoren kombiniert werden. Das Ergebnis wird dreidimensional in einem Fenster dargestellt, sodass ein instantanes Feedback von der beschriebenen Form zu einer räumlichen Darstellung besteht.

```
module glass() {
  color([0.3,0.3,1])
  translate([width/2,35,15/2+10])  union() {
```

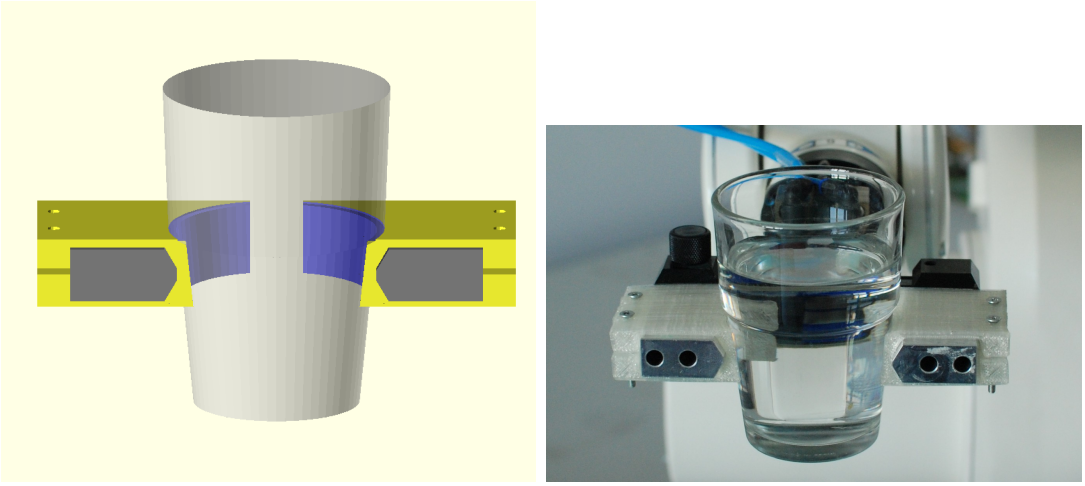
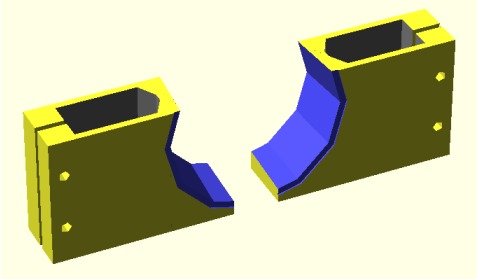


Abbildung 4.6: Glas-Greifer im Modell (links) und gedrucktes Resultat am Roboterarm (rechts).

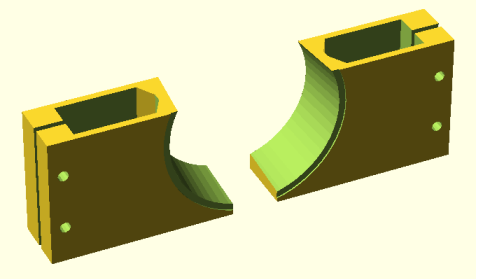
```
        cylinder(h = 44, r2 = 68.5/2, r1 = 65/2, center = false);
30      translate([0,0,-53+0.1]) {
          cylinder(h = 53, r2 = 60.5/2, r1 = 50/2, center = false);
        }
      }
    }
```

Listing 4.1: Glas modelliert in OpenSCAD

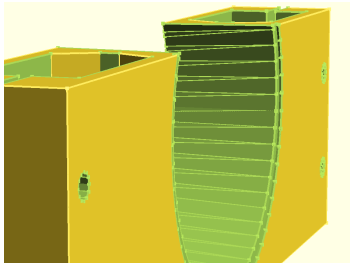
Nachdem das Glas ausgemessen und modelliert war, sollte der Kunststoff-Greifer entworfen werden. Das zu druckende Teil muss nun so auf die Finger des Roboterarms passen, dass bei geöffneten Fingern das Glas aufgenommen werden kann. Im geschlossenen Zustand soll das Glas fest gehalten werden. Zur Befestigung der Greifer am Finger haben wir eine Klemmvorrichtung angebracht, die mit M3-Schrauben gespannt werden kann. So kann die Halterung schnell an den vorhandenen Fingern befestigt werden, ohne zusätzliche Löcher in die Finger bohren zu müssen. Die fertigen Teile sind in Abbildung 4.6 dargestellt; der zugehörige OpenSCAD-Code ist im Anhang in Listing A.1 (Seite 37) abgedruckt.
Die Oberfläche der entworfenen Teile wird nun mit Dreiecken approximiert, die im Austauschformat STL exportiert werden. Dazu lassen sich Kreise in Polygone mit sinnvollen Seitenlängen umwandeln, um einerseits den Kreis gut zu approximieren und andererseits die Dateigröße nicht unnötig aufzublähen (Parameter: `$fa`, `$fs`, siehe Abbildung 4.7). Das Dateiformat STL wird von allen gängigen CAD-Programmen unterstützt und besteht aus einer ASCII-kodierten Liste von Dreiecken. Da jedes Dreieck an mindestens drei andere Dreiecke grenzt, werden alle Eckpunkte mehrfach abgespeichert, was die Dateigröße deutlich erhöht. Trotzdem dient das STL-Format als Quasi-Standard zum Austausch von 3D-Objekt-Informationen.
Die STL-Beschreibung wird im nächsten Schritt mit einem Slicing-Tool geladen und



(a) Schnelleres Rendering mit einem Kreissegment pro 30° (`$fa`=30)

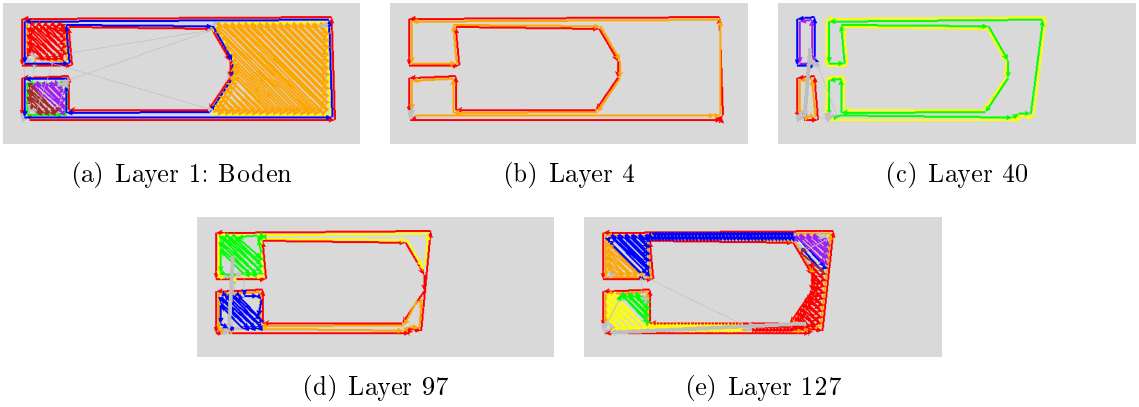


(b) Fertige Version (`$fa`=6)



(c) Fertige Version mit eingezeichneten Kanten

Abbildung 4.7: Beim Export von Kreisbögen aus OpenSCAD in eine STL-Oberfläche ist eine Approximation durch Polygonzüge nötig. Die Genauigkeit ist ein Kompromiss zwischen Dateigröße und Genauigkeit der Approximation.



(a) Layer 1: Boden (b) Layer 4 (c) Layer 40

(d) Layer 97 (e) Layer 127

Abbildung 4.8: Einige Ebenen des Skeinforge-Exports. Die Oberfläche wird ebenenweise betrachtet und als GCode ausgegeben. Dargestellt ist der Pfad, den der Druckkopf zurück legt.

zum Druck in GCode umgewandelt. Wir haben dafür die oben beschriebene Python-Sammlung Skeinforge verwendet. Einige Ebenen sind beispielhaft in Abbildung 4.8 gezeigt; wir haben das Objekt nicht gefüllt (Infill Solidity = 0), um einen schnellen Druck zu ermöglichen. Die Stabilität des fertigen Teils wurde dadurch nicht negativ beeinflusst. Als Beispiel des produzierten GCodes haben wir einen Ausschnitt der ersten Ebene in Listing 4.2 abgedruckt. Wir haben dazu die beiden Hälften separat behandelt und einzeln gedruckt.

```
G28
M106
M104 S220.0
G1 X25.268 Y136.162 Z0.36 F2400.0
G1 F798.0
G1 E5.0
G1 F2400.0
G1 X27.434 Y138.402 Z0.36 F720.0 E5.087
G1 X94.978 Y138.042 Z0.36 F720.0 E6.9725
G1 X95.084 Y114.136 Z0.36 F720.0 E7.6398
G1 X93.516 Y111.577 Z0.36 F720.0 E7.7236
G1 X26.96 Y111.577 Z0.36 F720.0 E9.5814
...
```

Listing 4.2: Ausschnitt aus dem von Skeinforge exportierten GCode zur Beschreibung der ersten Ebene des Glas-Greifers

Dieser GCode wird von der Firmware auf der Maschine direkt in Motorbewegungen umgesetzt, sodass das fertige Teil nach gut einer Stunde Druckzeit getestet werden konnte. Bis wir sinnvolle Werte für alle Toleranzen und Abstände ermittelt hatten, waren noch einige Iterationen notwendig; doch die schnelle Änderbarkeit stellt gerade ein wichtiges Feature des Rapid-Prototyping dar.

5 Zusammenfassung und Ausblick

Der Zusammenbau des 3D-Druckers RepRap Mendel ist innerhalb weniger Wochen mit Hilfe der im RepRap-Wiki verfügbaren Anleitungen möglich. Die anschließende Kalibration anhand von Testobjekten erhöht die Genauigkeit bis in den Sub-Millimeter-Bereich. Kleinste Strukturen von 0,5 mm können mit einer Positionsgenauigkeit von einigen Zehntelmillimetern gedruckt werden. Die fertigen Teile können in der Regel mit der Feile nachbearbeitet oder Löcher mit einem passenden Bohrer geweitet werden.
Zur Ansteuerung des Druckers haben wir verschiedene Softwarelösungen und deren Zusammenspiel getestet und die für unsere Ansprüche passenden Programme ausgewählt. Die Kalibrationsoptionen und Parameter erfordern ein detailliertes Verständnis des Druckvorgangs; in einer tabellarischen Aufstellung haben wir die wichtigsten Optionen zusammengefasst.
Als Erweiterung hat uns Kliment einen SD-Kartenleser zur Verfügung gestellt – der Autor der Firmware Sprinter. Damit kann man GCode-Dateien direkt auf eine Speicherkarte laden, sodass der PC nur noch das Kommando zum Start des Drucks geben muss und dann abgeschaltet werden kann. Der Anschluss an den Mikrocontroller erfolgt über die SPI-Schnittstelle; die Firmware hat die Funktionalität zum Lesen von FAT32-SDHC-Karten bereits implementiert.
Viele OpenSource-Projekte werden inzwischen gemeinschaftlich entwickelt und beispielsweise auf Github[1] gehostet. Jedermann kann Änderungen am Code vornehmen und so einen eigenen „Fork“ weiterentwickeln. Änderungen werden dann als Pull-Request dem Original-Autor vorgelegt und „upstream“ integriert. Dies ermöglicht eine schnelle und stabile Softwareentwicklung, auch wenn es sich wie beim RepRap-Projekt eher um eine Hobby-Aktivität handelt und keine Programmierer fest angestellt sind.
Auch zur Modellierung von dreidimensionalen Objekten existieren OpenSource-Lösungen wie Blender und OpenSCAD. Damit entwarfen wir eigene Teile zur Verbesserung des 3D-Druckers und Fertigung eines Kunststoff-Greifers, die am Greifarm-Roboter der Arbeitsgruppe eingesetzt werden. Entsprechend ist ein Einsatz des Druckers im Labor auch für den Druck von Teilen für künftige Projekte möglich.
Momentan besteht noch Verbesserungsbedarf, was die Zuverlässigkeit des Druckers angeht. Zusätzliche Endstops könnten einfach eingebaut werden, um zu erkennen, wenn sich ein Schrittmotor über das Softwarelimit hinausbewegt. Eine Sensorik für den Extruder zur Überwachung des Filamentvorschubs oder zur Inspektion des Druckergebnisses dürfte dagegen deutlich schwieriger sein. Momentan ist es zweckmäßig, den Drucker regelmäßig zu kontrollieren und zu beaufsichtigen. Es ist zu erwarten, dass Verbesserungen und Entwicklungen aus der weltweiten RepRap-Community auch die Zuverlässigkeit erhöhen werden. Die Präzision ist momentan schon sehr beachtlich, wenn man die Anschaffungskosten mit denen kommerzieller 3D-Drucker vergleicht.

[1] https://github.com

Auch die Replikation aller eigenen Teile auf unserem Drucker steht momentan noch aus. Es wäre prinzipiell möglich, die Kunststoffteile eines RepRap Mendel oder Prusa nachzudrucken. Allerdings liegt die geschätzte Druckzeit dafür bei insgesamt mehr als 20 Stunden. Dies war im Rahmen dieser Bachelorarbeit leider nicht mehr machbar.
Die Entwicklung des RepRap-Projekts ist insgesamt sehr beachtlich. Der erste Drucker, der seine Kunststoffteile selbst replizieren konnte wurde 2008 von Bowyer an der Universität Bath gebaut. Seither wurden mehrere tausend Kopien des 3D-Druckers weltweit nachgebaut (Jones et al., 2011). Und der Bausatz wird immer einfacher und günstiger und begeistert mehr und mehr Fans fürs Rapid Prototyping.

A Anhang

A.1 Skeinforge-Einstellungen

Tabelle A.2: Skeinforge-Einstellungen aus allen aktivierten Modulen. Nicht aufgeführte Module sind deaktiviert. Nur wenige dieser Parameter müssen tatsächlich angepasst werden; siehe dazu Tabelle 3.2.

Parameter-Name	Wert
carve	
Add Layer Template to SVG	True
Edge Width over Height (ratio):	1.38
Extra Decimal Places (float):	2.0
Import Coarseness (ratio):	1.0
Layer Height (mm):	0.36
Layers From (index):	0
Layers To (index):	912345678
Correct Mesh	True
Unproven Mesh	False
SVG Viewer:	webbrowser
preface	
Set Positioning to Absolute	True
Set Units to Millimeters	True
Start at Home	True
Turn Extruder Off at Shut Down	False
Turn Extruder Off at Start Up	False
inset	
Add Custom Code for Temperature Reading	True
Infill in Direction of Bridge	False
Infill Width over Thickness (ratio):	1.38
Ascending Area	True
Descending Area	False
Overlap Removal Width over Perimeter Width (ratio):	0.6
Turn Extruder Heater Off at Shut Down	True
Volume Fraction (ratio):	0.82
fill	
Activate Fill	True
Diaphragm Period (layers):	100
Diaphragm Thickness (layers):	0
Extra Shells on Alternating Solid Layer (layers):	2
Extra Shells on Base (layers):	1
Extra Shells on Sparse Layer (layers):	1
Grid Circle Separation over Perimeter Width (ratio):	0.2
Grid Extra Overlap (ratio):	0.1
Grid Junction Separation Band Height (layers):	10
Grid Junction Separation over Octogon Radius At End (ratio):	0.0
Grid Junction Separation over Octogon Radius At Middle (ratio):	0.0

Tabelle A.2: Skeinforge-Einstellungen (Fortsetzung)

Parameter-Name	Wert
Infill Begin Rotation (degrees):	45.0
Infill Begin Rotation Repeat (layers):	1
Infill Odd Layer Extra Rotation (degrees):	90.0
Grid Circular	False
Grid Hexagonal	False
Grid Rectangular	False
Line	True
Infill Perimeter Overlap (ratio):	0.15
Infill Solidity (ratio):	0.2
Sharpest Angle (degrees):	60.0
Solid Surface Thickness (layers):	2
Lower Left	True
Nearest	False
Surrounding Angle (degrees):	60.0
Infill > Loops > Perimeter	False
Infill > Perimeter > Loops	False
Loops > Infill > Perimeter	False
Loops > Perimeter > Infill	False
Perimeter > Infill > Loops	False
Perimeter > Loops > Infill	True
multiply	
Activate Multiply	True
Center X (mm):	60.0
Center Y (mm):	100.0
Number of Columns (integer):	1
Number of Rows (integer):	1
Reverse Sequence every Odd Layer	False
Separation over Perimeter Width (ratio):	12.0
speed	
Activate Speed	True
Add Flow Rate:	True
Bridge Feed Rate Multiplier (ratio):	1.0
Bridge Flow Rate Multiplier (ratio):	1.0
Duty Cyle at Beginning (portion):	1.0
Duty Cyle at Ending (portion):	0.0
Feed Rate (mm/s):	40.0
Flow Rate Setting (float):	40.0
Object First Layer Feed Rate Infill Multiplier (ratio):	0.6
Object First Layer Feed Rate Perimeter Multiplier (ratio):	0.3
Object First Layer Flow Rate Infill Multiplier (ratio):	0.6
Object First Layer Flow Rate Perimeter Multiplier (ratio):	0.3
Orbital Feed Rate over Operating Feed Rate (ratio):	0.5
Maximum Z Feed Rate (mm/s):	1.5
Perimeter Feed Rate Multiplier (ratio):	1.0
Perimeter Flow Rate Multiplier (ratio):	1.0
Travel Feed Rate (mm/s):	50.0
temperature	
Activate Temperature	True
Cooling Rate (Celcius/second):	1.5

Tabelle A.2: Skeinforge-Einstellungen (Fortsetzung)

Parameter-Name	Wert
Heating Rate (Celcius/second):	3.0
Base Temperature (Celcius):	215.0
Interface Temperature (Celcius):	215.0
Object First Layer Infill Temperature (Celcius):	215.0
Object First Layer Perimeter Temperature (Celcius):	220.0
Object Next Layers Temperature (Celcius):	195.0
Support Layers Temperature (Celcius):	223.0
Supported Layers Temperature (Celcius):	223.0
raft	
Activate Raft	True
Add Raft, Elevate Nozzle, Orbit:	True
Base Feed Rate Multiplier (ratio):	1.0
Base Flow Rate Multiplier (ratio):	1.0
Base Infill Density (ratio):	0.5
Base Layer Thickness over Layer Thickness:	2.0
Base Layers (integer):	0
Base Nozzle Lift over Base Layer Thickness (ratio):	0.4
Initial Circling:	False
Infill Overhang over Extrusion Width (ratio):	0.05
Interface Feed Rate Multiplier (ratio):	1.0
Interface Flow Rate Multiplier (ratio):	1.0
Interface Infill Density (ratio):	0.5
Interface Layer Thickness over Layer Thickness:	1.0
Interface Layers (integer):	0
Interface Nozzle Lift over Interface Layer Thickness (ratio):	0.45
Name of Support End File:	support_end.gcode
Name of Support Start File:	support_start.gcode
Operating Nozzle Lift over Layer Thickness (ratio):	0.5
Raft Additional Margin over Length (Raft Margin (mm):	3.0
Support Cross Hatch	False
Support Flow Rate over Operating Flow Rate (ratio):	1.0
Support Gap over Perimeter Extrusion Width (ratio):	1.0
None	False
Empty Layers Only	False
Everywhere	False
Exterior Only	True
Support Minimum Angle (degrees):	60.0
skirt	
Activate Skirt	True
Convex:	True
Gap over Perimeter Width (ratio):	5.0
Layers To (index):	1
jitter	
Activate Jitter	True
Jitter Over Perimeter Width (ratio):	10.0
clip	
Activate Clip	True
Clip Over Perimeter Width (ratio):	0.2
Maximum Connection Distance Over Perimeter Width (ratio):	10.0

Tabelle A.2: Skeinforge-Einstellungen (Fortsetzung)

Parameter-Name	Wert
stretch	
Activate Stretch	True
Cross Limit Distance Over Perimeter Width (ratio):	5.0
Loop Stretch Over Perimeter Width (ratio):	0.11
Path Stretch Over Perimeter Width (ratio):	0.0
Perimeter Inside Stretch Over Perimeter Width (ratio):	0.32
Perimeter Outside Stretch Over Perimeter Width (ratio):	0.1
Stretch From Distance Over Perimeter Width (ratio):	2.0
cool	
Activate Cool	True
Bridge Cool (Celcius):	1.0
Orbit	True
Slow Down	False
Maximum Cool (Celcius):	2.0
Minimum Layer Time (seconds):	8.0
Minimum Orbital Radius (millimeters):	10.0
Name of Cool End File:	cool_end.gcode
Name of Cool Start File:	cool_start.gcode
Orbital Outset (millimeters):	2.0
Turn Fan On at Beginning	True
Turn Fan Off at Ending	True
oozebane	
Activate Oozebane	True
After Startup Distance (millimeters):	1.2
Early Shutdown Distance (millimeters):	1.2
Early Startup Distance Constant (millimeters):	20.0
Early Startup Maximum Distance (millimeters):	1.2
First Early Startup Distance (millimeters):	25.0
Minimum Distance for Early Startup (millimeters):	1000.0
Minimum Distance for Early Shutdown (millimeters):	0.0
Slowdown Startup Steps (positive integer):	3
lash	
Activate Lash	False
X Backlash (mm):	0.15
Y Backlash (mm):	0.08
limit	
Activate Limit	True
Maximum Initial Feed Rate (mm/s):	40.0
dimension	
Activate Dimension	True
Absolute Extrusion Distance	True
Relative Extrusion Distance	False
Extruder Retraction Speed (mm/s):	13.3
Filament Diameter (mm):	2.9
Filament Packing Density (ratio):	0.97
Maximum E Value before Reset (float):	91234.0
Minimum Travel for Retraction (millimeters):	1.0
Retract Within Island	False
Retraction Distance (millimeters):	5.0

Tabelle A.2: Skeinforge-Einstellungen (Fortsetzung)

Parameter-Name	Wert
Restart Extra Distance (millimeters):	0.0
alteration	
Activate Alteration	True
Name of End File:	end.gcode
Name of Start File:	start.gcode
Remove Redundant Mcode	True
Replace Variable with Setting	True

A.2 OpenSCAD-Modell des Glas-Greifers

```
// Modell für Greif-Aufsatz für ein Glas (Greifarm)
// (c) Benjamin Reh und Joachim Schleicher, 2012

// Maße:
width=120;
bohrung=3.8; // Durchmesser!

// Auflösung:
$fa=6;    // Kreis-Sampling in Grad
$fs=0.5; // min. Länge eines Kreisbogens

// Settings:
export = false;

module finger() {
  color([0.5,0.5,0.5])
  difference(){
    translate([-2,-1,-0.5]) cube([32,52,16]);
    union (){
      translate([26,-2,-1]) rotate(a=30, v=[0,1,0]) cube([30,50+5,15]);
      translate([26,-2,16]) rotate(a=60, v=[0,1,0]) cube([30,50+5,15]);
    }
  }
}

module glass() {
  color([0.3,0.3,1])
  translate([width/2,35,15/2+10]) union() {
    cylinder(h = 44, r2 = 68.5/2, r1 = 65/2, center = false);
    translate([0,0,-53+0.1]) {
      cylinder(h = 53, r2 = 60.5/2, r1 = 50/2, center = false);
    }
  }
}

module piece() {
  color([1,1,0.2])
  difference () {
```

Befehl,	Beispiel-Parameter	Beschreibung
G1	X100 Z0.36 F450	Linear interpolierte Bewegung (X,Y,Z, E, F)
G28		Home (optional nur angegebene Achse)
G90		Absolute Positionierung (Standard)
G91		relative Bewegung
G92	E0	Überschreiben der aktuellen Position
M104	S190	Setzen der Extruder-Temperatur
M105		Abfragen der Temperatur
M106	S128	Lüfter an (Parameter PWM-Wert)
M107		Lüfter aus
M109	S190	Setzen und Warten auf Extruder-Temperatur
M114		aktuelle Position ausgeben

Tabelle A.1: Übersicht über die wichtigsten verwendeten GCode-Befehle. Für eine vollständige Liste der Kommandos siehe http://reprap.org/wiki/G-code und in der Dokumentation zur Firmware (Yanev, 2012).

```
    translate([2-14,2,-3])   cube([width/2+4,46,21]);

    union () { // das soll nicht gedruckt werden:
      finger();
      glass();
      translate([-12.5,0,7])   cube([20,50,2]); // Aussparung zum Klemmen
      // Und zwei Bohrungen:
      translate([-7,15,9])
        cylinder(h=26, r1=bohrung/2,r2=bohrung/2,center = true);
      translate([-7,35,9])
        cylinder(h=26, r1=bohrung/2,r2=bohrung/2,center = true);
    }
  }
}

module half() {
  piece();
  if(!export) finger();
}

//---- Main ----
rotate(90,[1,0,0]) // aufrecht stellen zum Druck
{
  half(); // eine Hälfte
  translate([width,0,0])   mirror ([1,0,0]){
    half(); // die andere Hälfte
  }
  if(!export) % glass();   // und das Glas
}
```

Listing A.1: OpenSCAD-Modell des Glas-Greifers

A.3 Eigener Quelltext

An der Mikrocontroller-Firmware konnten mehrere kleine Verbesserungen Upstream eingebracht werden (Listing A.2). Der Quellcode befindet sich auf http://github.com/kliment/Sprinter.

```
d30c319 compile experimental using the Makefile
d85e549 fix comment mode.
39bd33a compile experimental using the Makefile
77c3dd8 fix indentation
591b4fa correct dependencies in Makefile
```

Listing A.2: Verbesserungen an der Mikrocontroller-Firmware Sprinter

Glossar

Arduino Mikrocontroller-Board und dazu passende Entwicklungsumgebung. Das Design steht unter der freien Creative Commons Lizenz cc-by-sa 2.5. 10, 15

CAD Computer Aided Design. Entwurf und Modellierung mit Hilfe des Computers. 5, 15, 28

Extruder Mechanik, die für den Vorschub des Kunststofffilaments verantwortlich ist und das geschmolzene Rohmaterial durch eine dünne Düse an der Spitze des Hot Ends presst. 6

Filament Kunststoff-Draht, der als Rohmaterial im 3D-Drucker eingesetzt wird. Üblicherweise mit einem Durchmesser von 3 mm. 6

Hotend Heizung zum Aufschmelzen des thermoplastischen Kunststoffes. Am unteren Ende befindet sich eine feine Düse, die als Druckkopf dient. 9

Mikrocontroller Prozessor mit begrenztem Instruktionssatz (RISC) aber direktem Zugriff auf I/O-Pins. In der Robotik werden Mikrocontroller oft für Steuerungsaufgaben und das Auslesen von Sensoren verwendet. Die hier verwendeten ATmega Prozessoren lassen sich in C/C++ programmieren und haben mehrere Kilobyte Programm- und Arbeitsspeicher. 5, 6, 11, 12, 15, 16, 23

PLA Polymilchsäure ist ein Thermoplast. Der Bio-Kunststoff schmilzt bei circa 160°C und eignet sich bei 190°C sehr gut für den 3D-Drucker RepRap Mendel. 10, 23, 24

RepRap *Rep*licating *Rap*id-prototyper. 3D-Drucker, der einen großen Teil der eigenen Bestandteile replizieren kann. 5, 6

STL Surface Tessellation Language oder Standard Tessellation Language. Beschreibung der Oberfläche eines dreidimensionalen Objekts durch aneinander grenzende Dreiecke. 15, 17, 18, 25, 28

Literatur- und Softwareverzeichnis

Adrian Bowyer et al. RepRap Host software, 2011. URL https://github.com/reprap/release. Lizenz: GNU LGPL v2.1.

Adrian Bowyer et al. RepRap Mendel Mechanik-Repository, 2012a. URL https://github.com/reprap/mendel. Lizenz: GNU GPL v2 or later, Abgerufen am 19. März 2012.

Adrian Bowyer et al. RepRap Wiki, 2012b. URL http://www.reprap.org/. Lizenz: GNU GPL v2 or later, Abgerufen am 3. April 2012.

Erik de Bruijn. On the viability of the open source development model for the design of physical objects. Master's thesis, University of Tilburg, Niederlande, 2010. URL http://thesis.erikdebruijn.nl/master/MScThesis-ErikDeBruijn-2010.pdf.

Rhys Jones, Patrick Haufe, Edward Sells, Pejman Iravani, Vik Olliver, Chris Palmer, und Adrian Bowyer. Reprap – the replicating rapid prototyper. *Robotica*, 29(Special Issue 01):177–191, 2011. doi: 10.1017/S026357471000069X.

Holger Krautwasser und Dominik Wenger. 2PrintBeta, 2011–2012. URL http://www.2printbeta.de/.

Enrique Perez. Skeinforge – Toolchain zur Umwandlung eines 3D-Modells in GCode, March 2012. URL http://fabmetheus.crsndoo.com/. Lizenz: GNU AGPL v3, Version vom 15. März 2012.

Edward Anthony Sells. *Towards a Self-Manufacturing Rapid Prototyping Machine*. PhD thesis, University of Bath, 2009. URL http://opus.bath.ac.uk/20452/.

Zach Smith und Bre Pettis. Thingiverse – digital designs for physical objects. URL http://www.thingiverse.com/. Abgerufen am 3. April 2012.

Kliment Yanev. Sprinter – Firmware for RepRap printers and similar devices, 2012. URL https://github.com/kliment/Sprinter. Lizenz: GNU GPL v3 or later, Abgerufen am 3. April 2012.