Der Einstieg in den 3D-Druck [Teil 3]

In den ersten beiden Teilen dieser Blogserie habe ich Ihnen einen ersten Einblick gegeben, wie ein 3D-Drucker aufgebaut ist und welche Tools man benötigt, um kreative Ideen von anderen auf dem eigenen Drucker zu drucken. Grundlage dafür ist in den meisten Fällen eine stl-Datei, die mittels eines 3D-Slicer-Programmes in sogenannten gcode umgewandelt wird.
Auch ich habe viele solcher Dateien runtergeladen und gedruckt, jedoch war ich irgendwann von der Lautstärke des Druckers genervt, gerade wenn die Achsen sich viel bewegen oder viele Richtungsänderungen durchgeführt werden. Im zweiten Teil dieser Serie ging es schon um das Modding bzw. upgraden einer Custom Firmware mittels Ultimaker Cura, in diesem Blog wird der Hauptfokus auf Hardwareverbesserung, Kalibrierung und PID-Tuning bestehen. Zu diesem Zweck nutze ich den Druckserver OctoPrint, welchen ich aber zu einem späteren Zeitpunkt näher beleuchten.
Eine Warnung direkt am Anfang dieser Art von Modifikationen. An einigen Stellen arbeiten Sie in der Nähe von 230V. Diese Spannung ist gefährlich uns sollte niemals unterschätzt werden. Sofern Sie keine Fachausbildung haben oder es sich nicht wirklich zutrauen, können Modifikationen dieser Art teilweise lebensgefährlich sein!

Die Silent stepper motor driver kommen

Nach meiner etwas längeren Recherche, warum mein 3D-Drucker so laut war, waren die verbauten Steppermotortreiber die Quelle des Übels. Um zu verstehen, warum das so ist, habe ich ein sehr einfaches, aber dennoch sehr anschauliches Beispiel gefunden. Stellen Sie sich einfach zwei Polygone vor, eines mit z.B. 6 Seiten und eines mit 10 Seiten, siehe Abbildung 1.

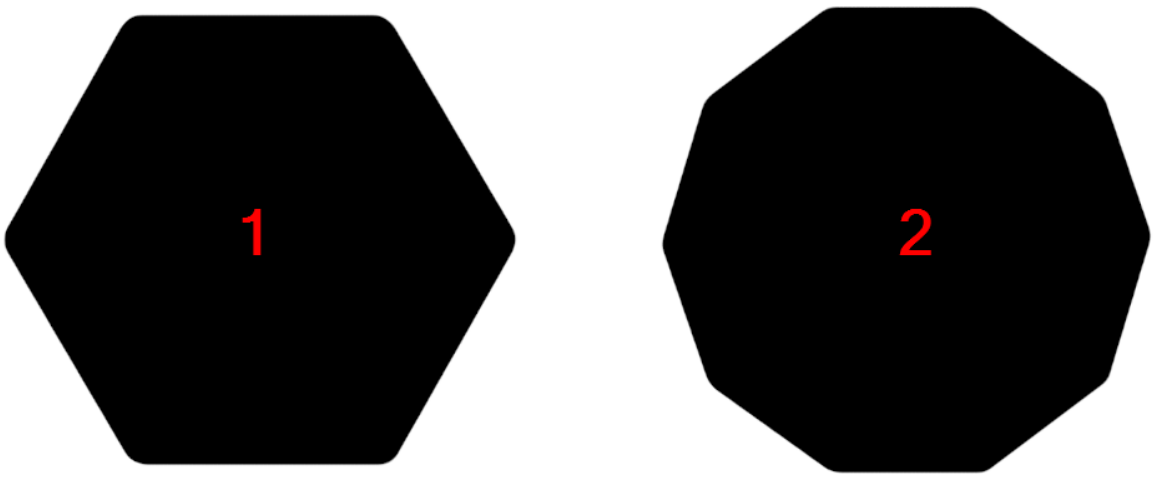

Abbildung 1: Simples Beispiel für Motortreiber

Ein Motortreiber sendet ein Signal an den Motor, wie weit er sich Drehen soll und in welcher Richtung. Bei günstigen Motortreibern, siehe Abbildung 1 rote 1, ist die Auflösung eher gering, was zu einem unsauberen Druckbild führt. Sind die Motortreiber jedoch hochwertiger, siehe Abbildung 1 rote 2, können die Motoren präziser angesteuert werden. Dadurch wird auch gleichzeitig das Druckbild deutlich verbessert und die Temperatur der Motoren kann deutlich reduziert werden. Gerade wenn die Motortreiber eine höhere Auflösung haben und auch qualitativ hochwertiger sind, behebt man gleich drei Probleme:

1. Motoren zu laut
2. Motoren werden zu warm

3. Unsauberes Druckbild

Im Netz finde Sie zum Thema „Silent stepper motor driver“ einen Haufen von Ergebnissen. Je nach Drucker finden Sie meist als präferierte Treiber die TMC2208 oder TMC2209-Varianten. Diese gibt es von vielen Herstellern, doch gerade günstige Motortreiber aus Fernost bringen nicht die erhoffte Verbesserung. Im Gegenteil, meist wird der Drucker zwar leiser, aber das Druckbild wird nicht wirklich besser werden. Gerade bei den TMC 2208 wird mit der Version 2 oder 3 gemarkted, wobei der original TMC2208 es nur bis zu Version 1.2 gibt. Darauf sollte man vor dem Kauf der neuen Treiber achten.
Für meinen Drucker, einen Anycubic i3 Mega S, habe ich mich für die TMC2208 V1.2 von FYSECT entschieden, siehe Abbildung 2.

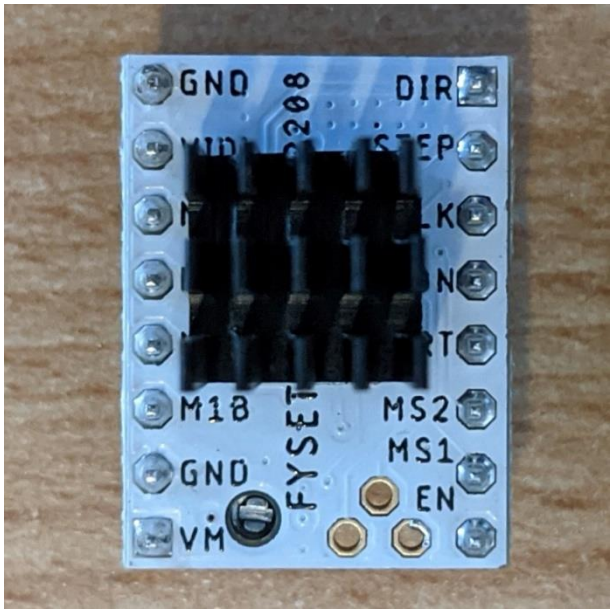

Abbildung 2: TMC2208 V1.2 von FYSECT

Wichtig ist noch anzumerken, dass ich nicht die UART-Version nutze, sondern die Stepper-Variante. Auch hier ist vor dem Kauf zu achten, welche Variante in den Warenkorb landet.

An dieser Stelle wird nun erklärt, wie ich bei einem geöffnetem Gerät Modifikationen durchführe. Es liegen 230V an einigen Stellen der Beschreibung an, was lebensgefährlich ist. Sofern Sie keine Fachkraft oder sich im Umgang mit dieser Spannung verstehen, ist dringend von solchen Modifikationen abzuraten. Fragen Sie stattdessen eine Person vom Fach, diese Modifikationen durchzuführen.

In meinem Drucker sind sogenannte A4988-Schrittmotortreiber verbaut, welche durch die TMC2208 ersetzt werden sollen. Zunächst muss der Boden von meinem Drucker geöffnet werden, was bedeutet diesen am besten auf die Seite zu legen, siehe Abbildung 3. Vorher sollte der Kaltgerätestecker vom Drucker abgezogen werden.

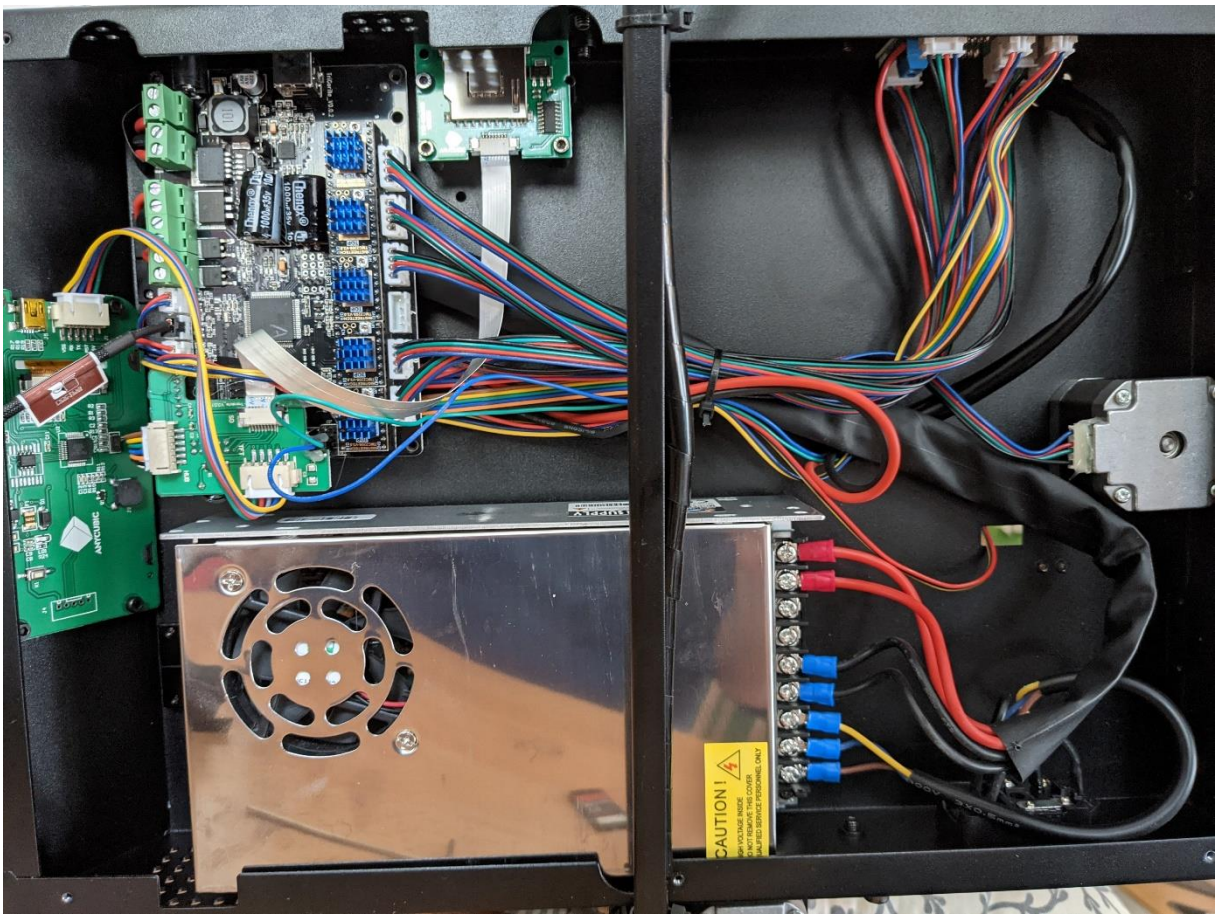

Abbildung 3: Ein Blick in den Drucker

Zunächst sehen wir unten das Netzteil, welches 230V entsprechend in 24 und 12 Volt runterregelt. Wie deutlich zu erkennen, sind die Anschlüsse am Netzteil offen, was diesen Umbau gefährlicher macht. Oben links sehen Sie das Herzstück des Druckers, das Mainboard mit allen Anschlüssen. Links das Display und oben den SD-Kartenleser. Schaut man sich das Mainboard genauer an, sieht man die verbauten Motortreiber recht deutlich, siehe Abbildung 4 rote Umrandung. Vorher habe ich noch den Lüfter zur Kühlung der Treiber abmontiert, damit der Ein- und Ausbau einfacher vonstattengeht.

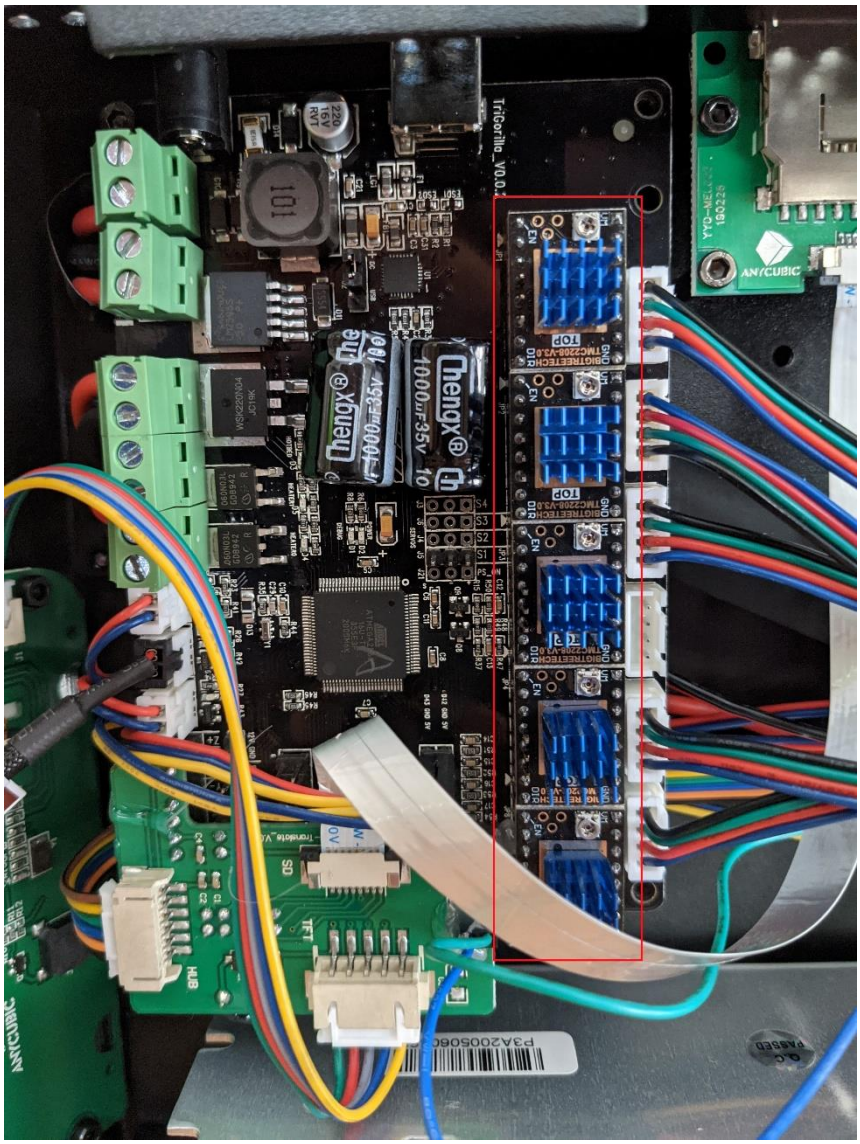

Abbildung 4: Verbaute Steppertreiber

Wundern Sie sich nicht, ich habe kein Originalbild der Motortreiber, da ich für einen Test andere verbaut habe. Prüfen müssen Sie aber, ob die Treiber gesteckt sind oder aber auf das Mainboard gelötet sind, in meinem Fall waren sie gesteckt. Das macht einen Umbau recht einfach und spart Zeit, sowie Ein- und Ausbaumaßnahmen.
Sind die vorherigen Treiber ausgebaut, kann man sich mal die alten und neuen Treiber genauer ansehen, siehe Abbildung 5.

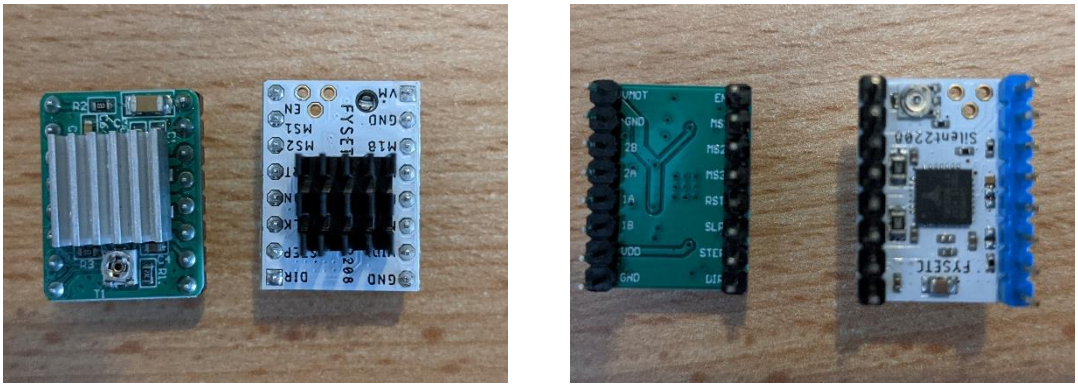

Abbildung 5: Alte und Neue Steppertreiber oben und unten

Allein die verbauten Chips und Elektronikbauteile zeigt, dass der neue Treiber, rechts im Bild, hochfertiger zu sein scheint. Wichtig ist unter anderem auch, welcher Pin für was zuständig ist. Hierbei hilft auch die Beschriftung auf dem Mainboard selbst, siehe Abbildung 6, um die Module korrekt einzubauen.

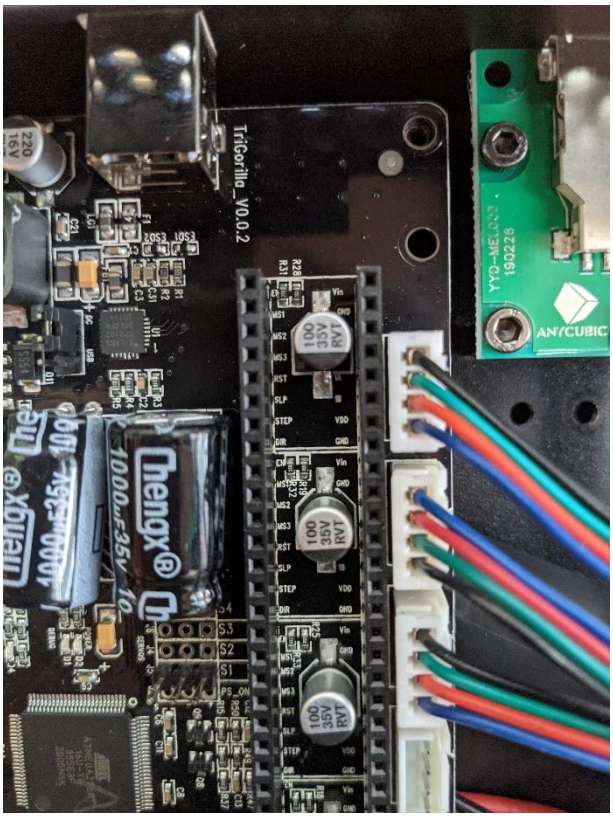

Abbildung 6: Pinbeschriftung auf dem Drucker-Mainboard

Bei den alten Treibern ist die Beschriftung auf der Unterseite, weswegen man spiegelverkehrt denken muss. Bei den neueren Treibern ist die Beschriftung auf der Oberseite und ein Umdenken ist somit nicht nötig. Trotzdem sollte bei dem Umbau genau darauf geachtet werden, dass die Pins auch an der richtigen Stelle stecken, andernfalls riskiert man eine Zerstörung vom Mainboard. In meinem Fall war der Einbau aber sehr schnell und unkompliziert, siehe Abbildung 7.

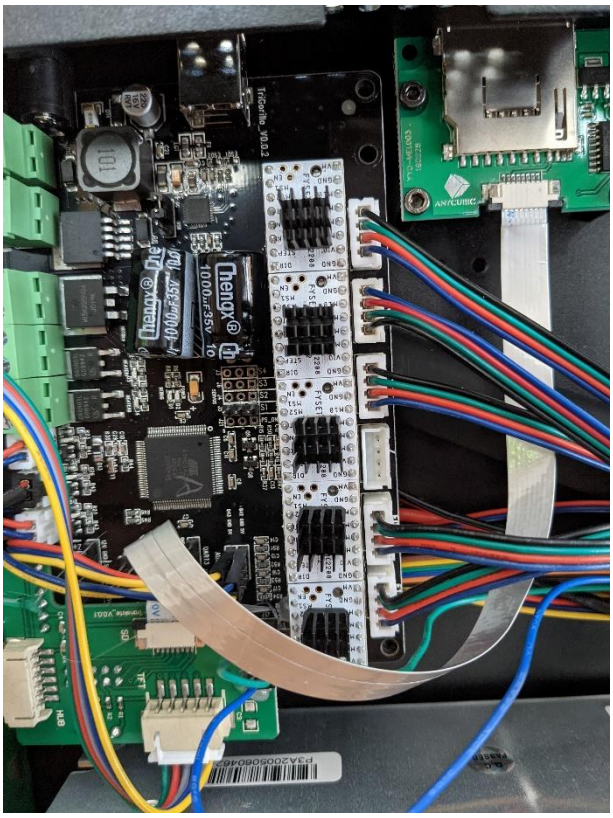

Abbildung 7: Die neuen Steppertreiber auf dem Mainboard

Damit ist der erste und noch recht einfache Teil des Umbaues fertig, wobei Sie durchaus kontrollieren sollten, dass alle Treiber richtig verbaut wurden. Wie schon erwähnt, in meinem Fall hat die Beschriftung auf dem Mainboard sehr geholfen.
Jetzt kommt der gefährliche Teil, da der Kaltgerätestecker wieder an den Drucker angeschlossen werden muss, vorher aber die Kabel außen am Gehäuse zu den Motoren abziehen nicht vergessen, siehe Abbildung 8.

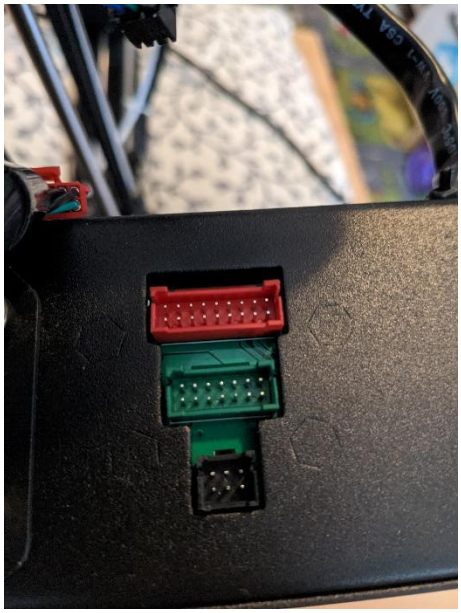

Abbildung 8: Kabel außerhalb vom Drucker abgezogen

Dies muss gemacht werden, da nun die Referenzspannung an jedem einzelnen Steppertreiber eingestellt werden muss. Dies wird über einen beliebigen Masseanschluss am Mainboard und der

Einstellschraube zum Einstellen der Referenzspannung gemessen, siehe Abbildung 9. Zu diesem Zweck muss der Drucker eingeschaltet werden, da sonst keine Spannung am Mainboard anliegt.

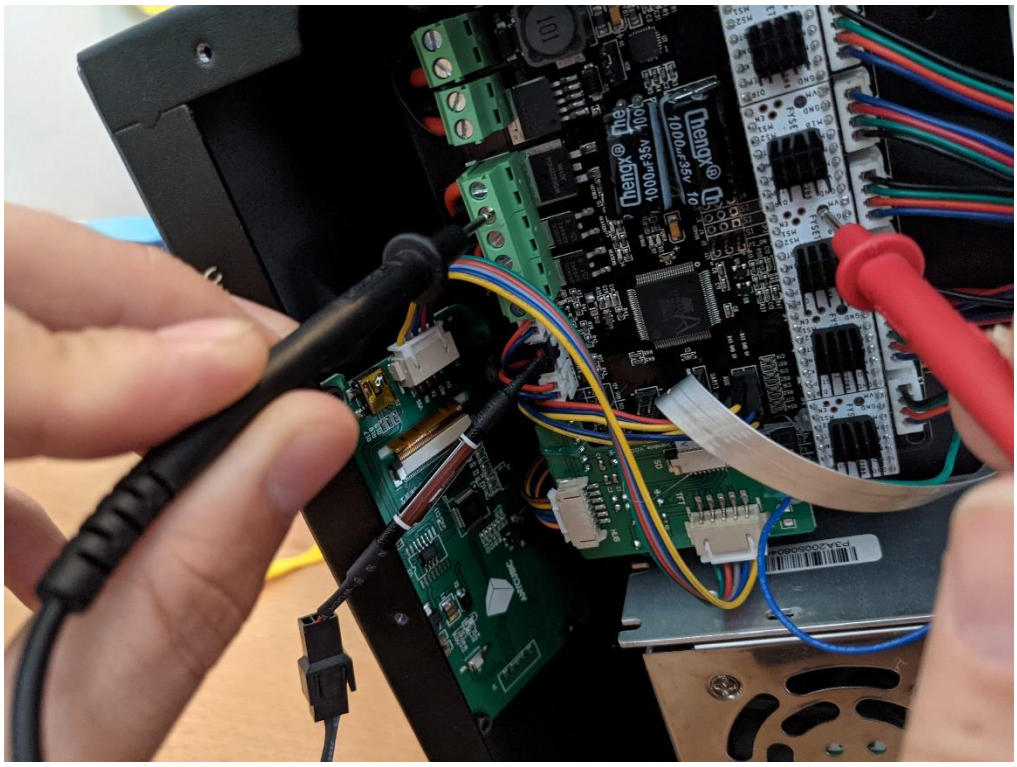

Abbildung 9: Messen der Referenzspannung der Motortreiber

Es ist also Vorsicht geboten, wenn die Referenzspannung an dem Drehpotenziometer eingestellt wird. Bei der Referenzspannung gibt es so viele Meinungen, wie Nutzer in Foren. In meinem Fall habe ich für die X-, Y- und beide Z-Achsen die Spannung auf 1,05V gestellt und den Extruder auf 1,1V. Sollte das Druckbild einen Versatz bei einer Achse aufweisen, so muss die Referenzspannung entsprechend angepasst werden. Auf unter 0,8V, so findet man es in diversen Foren, sollte die Spannung nicht eingestellt werden. Gerade weil das Netzteil offene Anschlüsse hat, achten Sie darauf nicht an die Kabel zu kommen, um einen Stromschlag zu vermeiden oder Bauteile zu zerstören. Zuletzt muss der Drucker wieder zusammengebaut werden, siehe Abbildung 10.

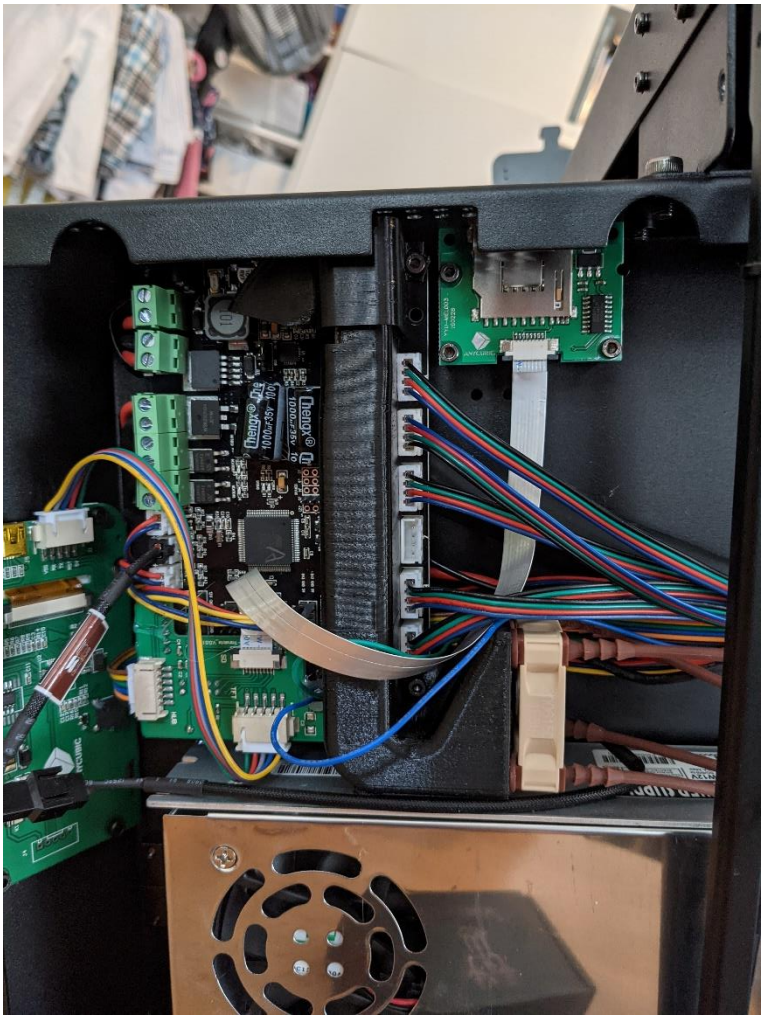

Abbildung 10: Motortreiber unter Lüftungskanal

In meinem Fall habe ich noch vorher einen Lüftungskanal gedruckt und diesen mit einem leiseren 12V-Lüfter in dem Gehäuse, über die Motortreiber, verbaut. Für die Montage des Lüftungskanals werden die Befestigungsschrauben vom Mainboard genutzt. Damit sollten auch die Motortreiber nicht zu heiß werden. Auch hier gibt es diverse Meinungen, ob ein solcher Kanal brauchbar ist oder nicht.
Ist der 3D-Drucker wieder zusammengebaut und alle Stecker angeschlossen, musste in meinem Fall noch die Firmware angepasst werden, da sonst die Motoren in die falsche Richtung drehen würden, sprich die Motorrichtung ist invertiert. Wie schon im zweiten Teil vom Blogbeitrag, nutze ich die angepasste Marlin-Firmware von knutwurst. Damit die Motoren wieder in die richtige Richtung drehen, braucht es die Firmware mit dem Zusatz „_TMC“, siehe Abbildung 11.

Abbildung 11: Auswahl der korrekten Firmware "knutwurst"

Das Aufspielen der neuen Firmware geschieht auf demselben Weg, wie es schon im zweiten Blogbeitrag beschrieben wurde, in meinem Fall mit dem 3D-Slicer-Programm Ultimaker Cura. Der Drucker ist damit nun leiser und ein Problem mit falsch drehenden Motoren ist schon direkt im Vorfeld behoben worden.

OctoPrint als Druckserver

Wie im zweiten Blogbeitrag angekündigt, soll hier OctoPrint kurz vorgestellt werden. OctoPrint ist eine Weboberfläche, welche es erlaubt den Drucker zu steuern und Drucke zu verwalten. Vom Grundgedanken können Sie es sich wie einen Druckerserver vorstellen, wobei Sie auf der Weboberfläche die gcodes hochladen, um diese dann danach auszuwählen für den Druckauftrag. Des Weiteren haben wir Zugriff auf das sogenannte Terminal, welches die direkte Kommunikation mit dem Drucker, via Marlin-Codes, ermöglicht.
OctoPrint wird primär mit einem Image für den Raspberry Pi, dem sogenannten OctoPi-Image, angeboten, jedoch kann es auch auf allen anderen gängigen Betriebssystemen installiert werden. Wie das genau geht, wird auf der entsprechenden Downloadseite von OctoPrint erläutert. Damit Sie OctoPi ohne Einschränkungen nutzen können, empfiehlt die Seite einen Raspberry Pi 3B oder höher. Es gibt aber auch Nutzer in diversen Foren, die über einen erfolgreichen Einsatz mit einem Raspberry Pi 2 oder Zero berichten.
Bevor die Frage aufkommt, OctoPrint ist nicht die einzige Druckerserverlösung für den 3D-Druck. Tatsächlich gibt es auch noch den sogenannten Repetier Server. Diese Software kostet aber Geld um alle Features zu nutzen, bringt aber im Gegensatz zu OctoPrint den Vorteil, dass mehrere 3D-Drucker gleichzeitig verwaltet werden können. Für meine ersten Gehversuche habe ich mich dennoch für OctoPrint entschieden, da ich Ultimaker Cura für das Slicen nutze und OctoPrint dann „nur“ den Druckauftrag managen soll. Für diesen Blogbeitrag ist aber erst einmal das Terminal und die Druckersteuerung wichtig, da wir hier die weiteren Softwareanpassungen durchführen werden.

Das PID-Tuning vom Heatbed und Hotend

An dieser Stelle wird es nun etwas technisch und wir machen einen kleinen Exkurs in die Welt der Steuer- und Regelungstechnik. Wie der Name der Überschrift schon erwähnt, sollen die PID-Regelkreise für das Heatbed und das Hotend angepasst werden. Wer jetzt nicht in der Steuer- und Regelungstechnik zuhause ist, dem will ich das nur kurz erklären.
Ein PID-Regler ist, wie der Name schon vermuten lässt, ein Regelkreis, um einen vorgegebenen Sollwert auch bei Störeinflüssen möglichst gezielt zu halten. In den meisten Lektüren wird diese wie in Abbildung 12 dargestellt.

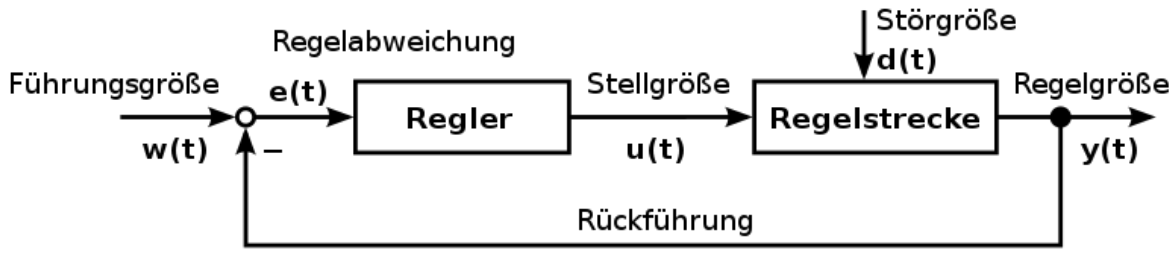

Abbildung 12: PID-Regler, Wikipedia by Mrmw

Im Falle vom 3D-Drucker bedeutet in diesem Fall der Sollwert die Temperatur vom Heatbed und Hotend. Durch die Verfahrbewegung, den Lüfter, Umgebungstemperatur und viele weitere Faktoren, kann die Temperatur stark schwanken, was zu einem unsauberen Druckbild oder ablösen vom Heatbed führen könnte. Mit dem PID-Regler und einem Temperatursensor werden diese Schwankungen erfasst und entsprechende „Gegenmaßnahmen“, um die Temperatur auf dem Sollwert zu halten, durchgeführt. Im Falle vom Heatbed und Hotend wird einfach das MOSFET zum Aufheizen für eine gewisse Zeit ein- und danach wieder ausgeschaltet.
Das Tuning dieser beiden Regelkreise bezieht sich auf die Parameter **P**, **I** und **D**. Der P-Anteil ist ausschließlich der proportionale Anteil der Verstärkung.
Der I-Anteil ist der integrale Anteil im Regelkreis, für die zeitliche Regelabweichung. Der D-Anteil ist der differenziale Anteil, der auf die Änderung des Sollwertes eingeht.
Diese drei Konstanten sind in der Marlin-Firmware vordefiniert, werden aber nicht mit den für den Drucker benötigten Werten übereinstimmen! Zusätzlich bietet mir in diesem Fall erst die Customfirmware „knutwurst“ die Möglichkeit diese Regelkreisparameter anzupassen, da Sie mit der original Firmware nicht veränderbar waren. In beiden Fällen ist wichtig, dass der Drucker die Umgebungstemperatur hat und vorher nicht irgendwie aufgeheizt wurde, andernfalls stimmen die nachher ermittelten Werte aus der Software nicht!

Beginnen wir zunächst mit dem PID-Tuning vom Heatbed. Dazu öffne ich OctoPrint und Stelle eine Verbindung mit meinem Drucker und dem Pi her, siehe Abbildung 13.

Abbildung 13: OctoPrint mit Drucker verbinden

Danach öffne ich das Terminal, siehe Abbildung 14, wobei hier schon einige Informationen angezeigt werden. Die Fülle der Informationen mag im ersten Moment abschrecken, es sind aber nur Informationen, die der Drucker mit dem Raspberry Pi austauschen und visualisiert werden.

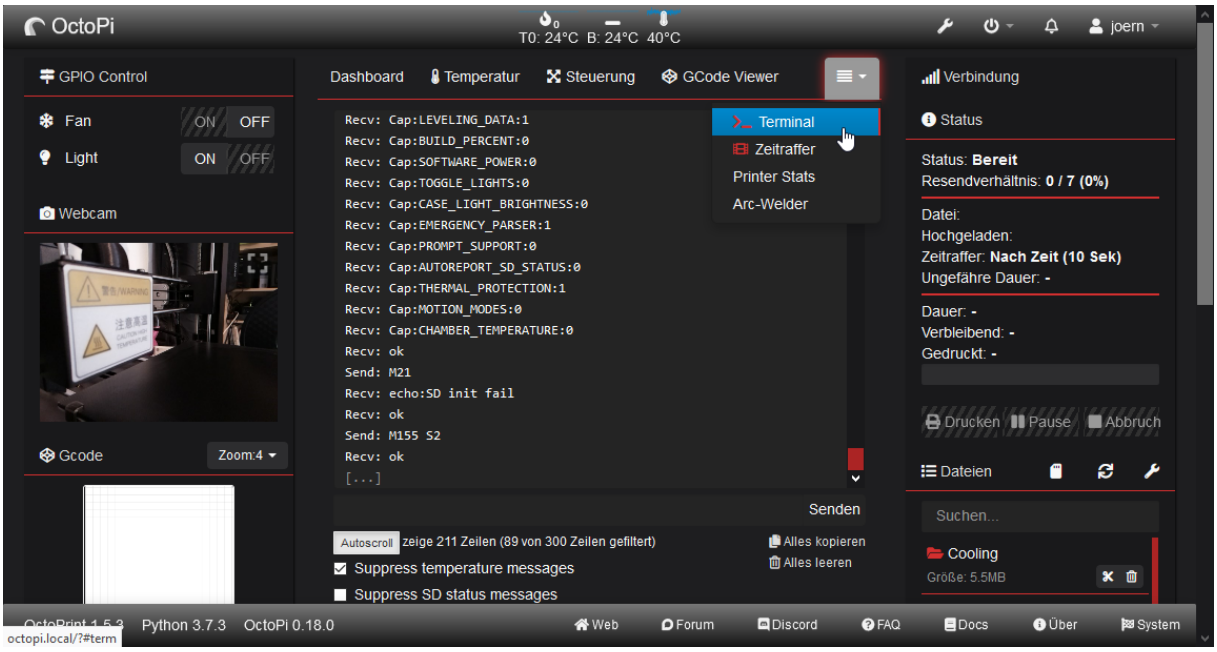

Abbildung 14: Das Terminal öffnen

Danach wird in das Terminal der Befehl aus Code 1 eingegeben. Diesen Befehl mit den Parametern möchte ich noch erklären, damit Sie verstehen was genau der Befehl macht.

```
M303 E-1 S60 C8
```

Code 1: Terminalbefehl für das PID-Kalibrieren des Heatbed

M303 steht für den Befehl das PID-Autotuning durchzuführen. Der Parameter **E-1** steht für das Hetabed. Wichtig an dieser Stelle, da es hier um das PID-Tuning vom Heatbed geht. Der Parameter **S60**, steht für die Solltemperatur 60 Grad, die das Heatbed aufheizen und halten soll. Der letzte Parameter **C8** steht für die Anzahl an Zyklen, die für das PID-Autotuning durchgeführt werden sollen. Dabei wird das Heatbed bewusst wieder abgekühlt und leicht überhitzt, um die genauen Parameter für den PID-Regel zu ermitteln.
Wurde der Befehl am Terminal abgeschickt, passiert anscheinend erst einmal wenig, siehe Abbildung 15, jedoch wird nun das Heatbed auf die Zieltemperatur 60 Grad aufgeheizt.

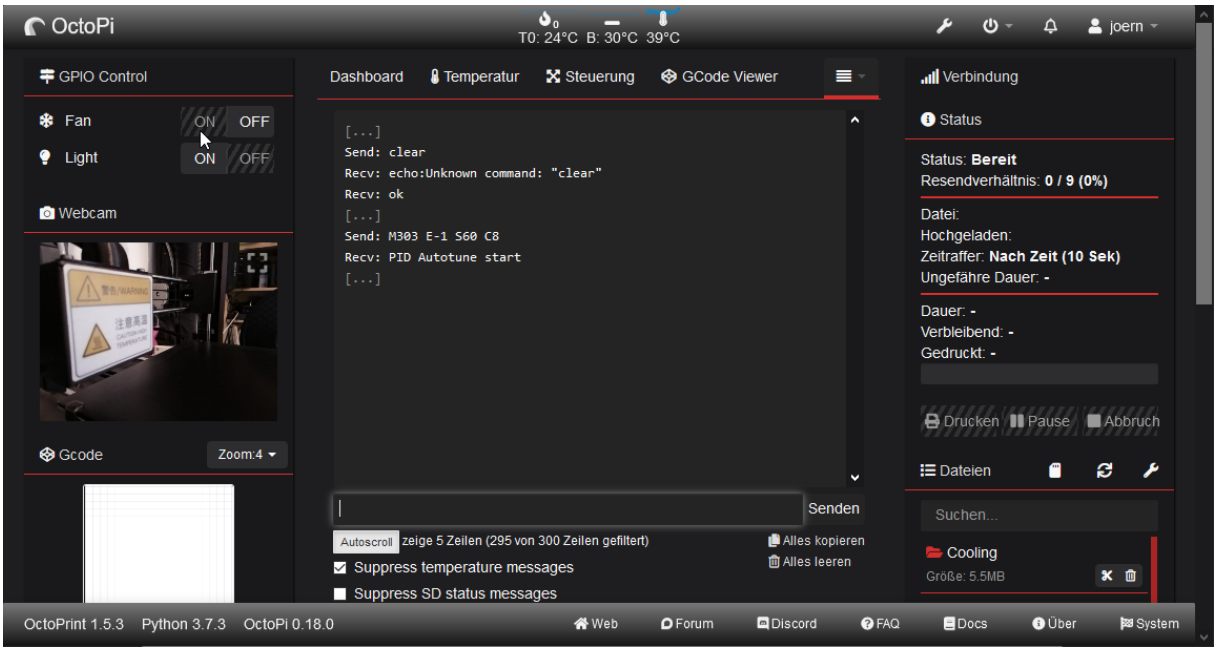

Abbildung 15: PID-Autotuning vom Heatbed gestartet

Nach einer kurzen Zeit wird sich das Terminal mit einigem Text füllen, siehe Abbildung 16, welches unter anderem die neuen Parameter für den PID-Regler ausgibt.

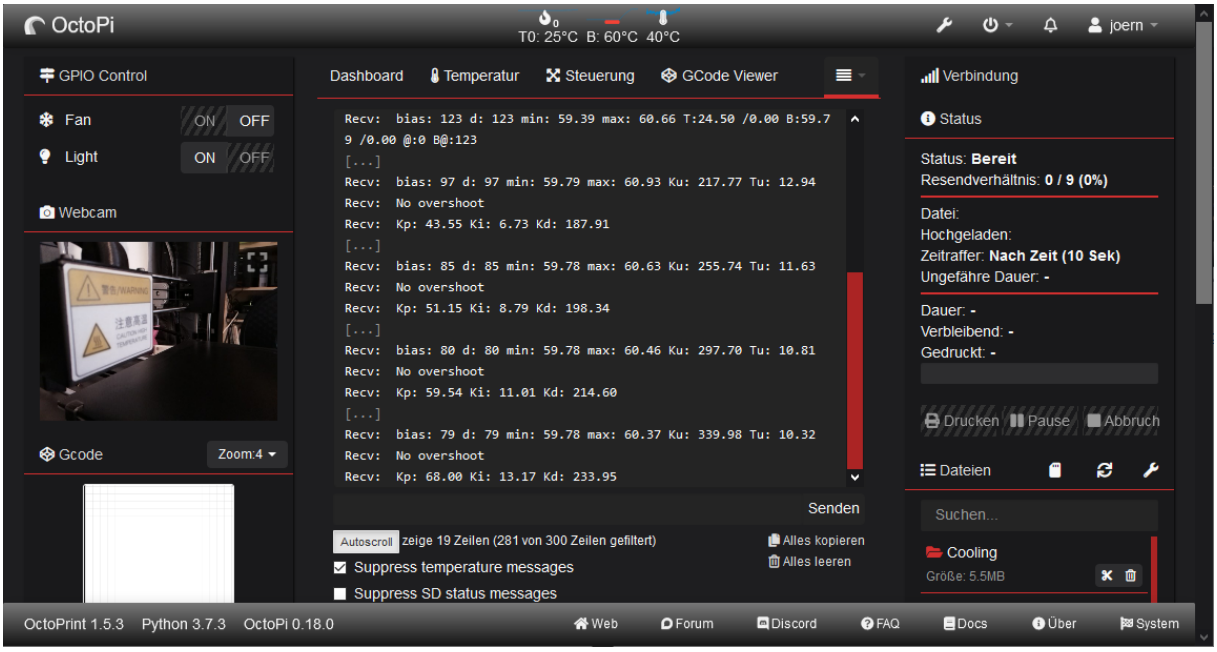

Abbildung 16: Zwischenstand des PID-Autotuning vom Heatbed

Diese Meldungen können erst einmal ignoriert werden, dass es sich nur um einen Zwischenstand handelt! Diese Werte sind die ermittelten Werte des jeweiligen Zyklus, die aber noch nicht die Endparameter darstellen. Wichtig wird es erst, wenn Sie die Zeile wie in Abbildung 17 sehen.

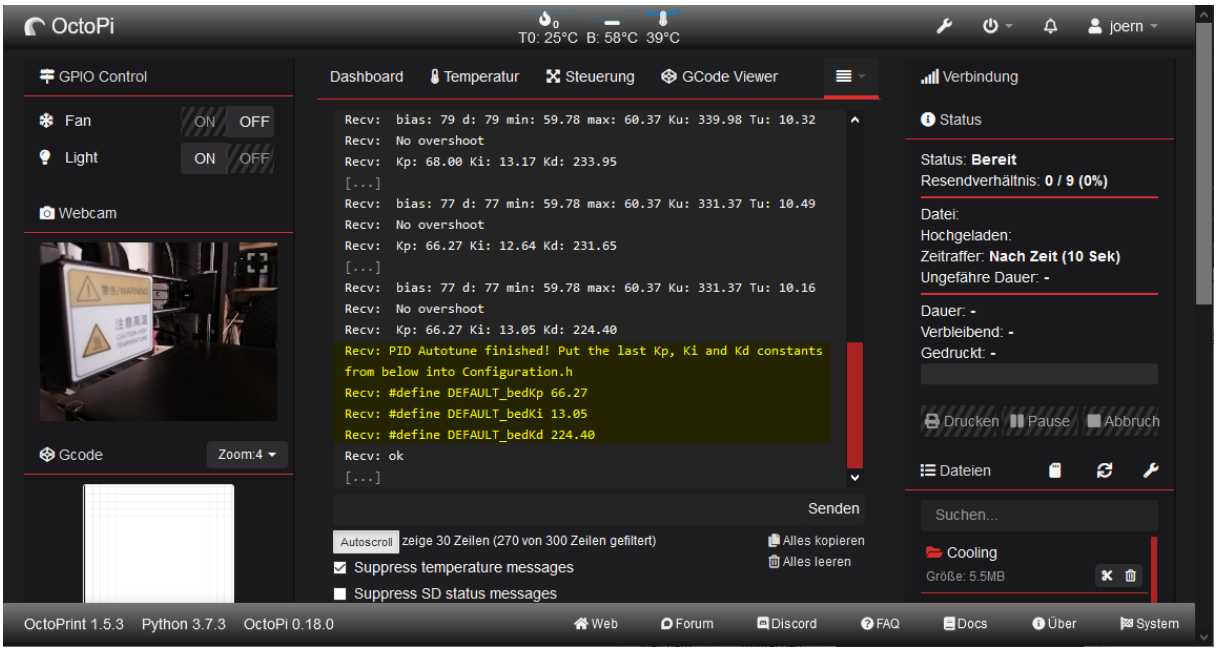

Abbildung 17: Fertiges PID-Autotuning vom Heatbed mit neuen Konstanten

Hier vermeldet das Terminal, dass das PID-Autotuning abgeschlossen ist und die Konstanten für den PID-Regel ermittelt wurden, siehe Abbildung 17 gelbe Markierung. Die Werte werden von den hier angezeigten Werten in Abbildung 17 abweichen, aber das Terminal empfiehlt, diese Werte in der Configuration.h abzuspeichern. Nur dass wir die Firmware nicht neu kompilieren werden, sondern die Werte mittel Terminalbefehl dem Drucker übermitteln. Dazu nutzen wir den Marlinbefehl aus Code 2.

```
M304 P66.27 I13.05 D224.40
```

Code 2: Die neuen Konstanten vom PID-Regler des Heatbeds übermitteln

Auch hier die kurze Erläuterung, damit Sie den Befehl komplett verstehen. M304 ist der Marlinbefehl, um die Werte vom PID-Regler zu setzen. Wichtig hier wird der Parameter erst gesetzt und **nicht** gespeichert! Die Parameter P, I und D repräsentieren die Werte für den Regelkreis und sollten die ausgegebene Werte vom PID-Autotuning haben. Damit es keine Probleme gibt muss statt dem Komma immer den Punkt für die Nachkommastellen verwendet werden! Nach dem eingegebenen Befehl, erscheint nur noch ein kurzes Feedback, dass die Werte übernommen wurden, siehe Abbildung 18.

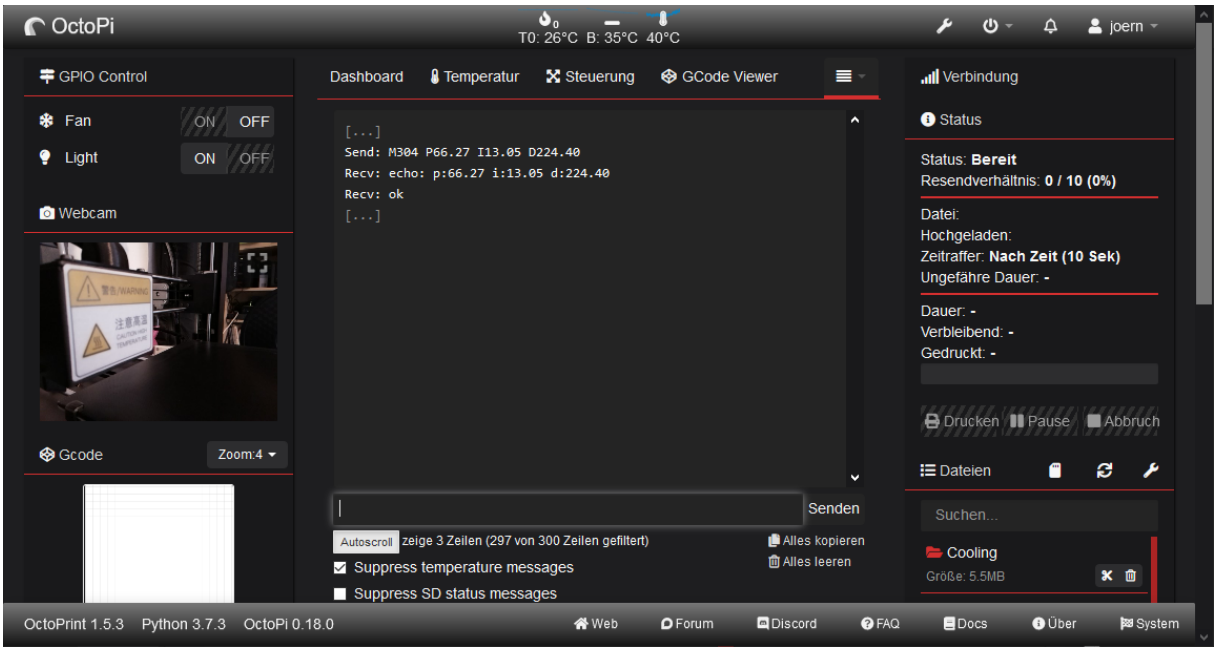

Abbildung 18: Der Drucker hat die neuen PID-Konstanten fürs Heatbed erhalten

Damit sind die neuen Konstanten zwar temporär vorhanden, aber beim Neustart oder Reset via Marlinbefehl auf die EEPROM-Daten, werden wieder die alten Konstanten genutzt. Damit das nicht passiert bzw. die Werte fest hinterlegt sind, speichert man die Werte dauerhaft mit dem Marlinbefehl aus Code 3 im EEPROM vom Drucker.

M500

Code 3: Parameter dauerhaft im EEPROM speichern

M500 ist der Marlinbefehl, um alle Änderungen in das EEPROM zu übertragen, welches geladen wird, wenn der Drucker z.B. neugestartet wird.

Damit bleibt noch das Hotend offen, welches ähnlich von den Parametern ist, wie das Heatbed. Eine kleine Abweichung gibt es aber, wir wollen zusätzlich, dass der Hotendlüfter aktiv ist und als „Störquelle“ auch richtig geregelt wird. Mittels des Marlinbefehls von Code 4, wird der Hotendlüfter auf volle Drehzahl gestellt.

M106

Code 4: Den Hotendlüfter auf volle Drehzahl stellen

Als nächstes soll das PID-Autotuning vom Hotend gestartet werden. Hierzu wird wieder der Marlinbefehl M303 genutzt, jedoch sind die Parameter diesmal leicht anders, siehe Code 5.

M303 E-0 S230 C8

Code 5: PID-Autotuning fürs Hotend starten

In diesem Fall bedeutet **E-0**, dass das Hotend ausgewählt werden soll und **S230**, dass eine Temperatur von 230 Grad der Sollwert ist.
Nun heißt es wieder warten, bis die Zyklen für das PID-Autotuning durchlaufen sind und das Terminal die entsprechend Rückmeldung über das erfolgreiche Beenden anzeigt, siehe Abbildung 19.

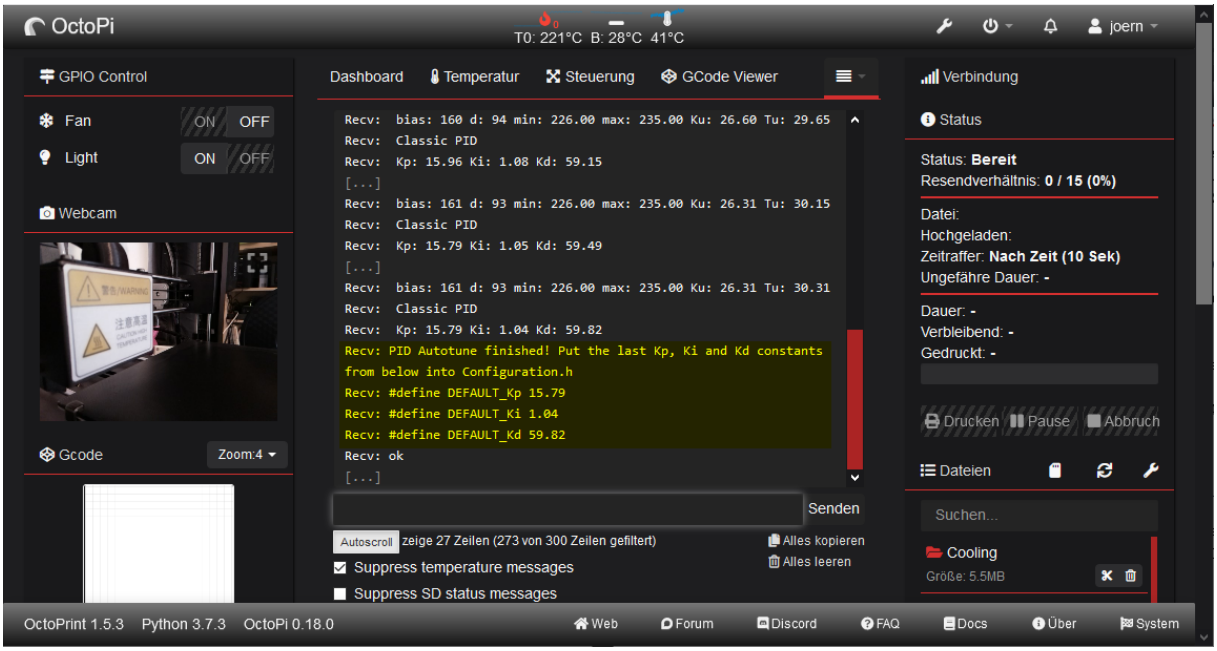

Abbildung 19: PID-Autotuning vom Hotend abgeschlossen

Auch hier sind die Werte für den PID-Regler wieder klar angezeigt, die mittels Code 6 an den Drucker übertragen werden.

```
M301 P15.79 I1.04 D59.82
```

Code 6: PID-Konstante für Hotend setzen

Wichtig ist, dass der Marlinbefehl **M301** genutzt wird, da dieser die PID-Konstanten in den Speicher schreibt. Danach sollten **M500** nicht vergessen werden, damit auch diese PID-Konstanten dauerhaft in den EEPROM vom Drucker gespeichert werden.
Wenn der Lüfter nun nerven sollte, dann führen Sie Code 7 aus, welches die Lüfterdrehzahl wieder auf **0** setzt.

```
M106 S0
```

Code 7: Den Hotendlüfter wieder anschalten

Den Extruder kalibrieren

Der letzte Punkt in diesem Blog und der Verbesserung unsere Druckbildes, widmet sich dem Kalibrieren des Extruders. Der Extruder ist der Teil am Drucker, der das Filament zum Hotend führt. Auch hier wird ein Steppermotor verwendet, den es korrekt einzustellen gilt. Genauer gesagt soll nicht der Steppermotor korrekt eingestellt werden, sondern der Steppermotortreiber, den ich zu Anfang dieses Artikels getauscht habe. Macht man dies nicht, könnte man schnell frustriert sein, da man entweder zu viel oder zu wenig Filament fördert und somit das Druckbild mehr als bescheiden aussieht.
Damit man den Extruder kalibrieren kann, brauche es:

1. Einen Messschieber
2. Einen Stift der gut erkennbar auf dem Filament ist
3. Filament
4. Eine Software mit dem ich explizit den Extruder ansteuern kann, in meinem Fall OctoPrint mit dem Tab Steuerung

Damit Sie auch Kalibrieren können, muss das Hotend auf Betriebstemperatur gebracht werden, da sonst die Firmware vom Drucker den Extruder nicht arbeiten lässt. Gleichzeitig, da kein Filament

verschwendet werden soll, ziehe ich das Filament zurück, sodass das Hotend und der Führungsschlauch kein Material mehr hat.
Im nächsten Schritt nehme ich den Marker und mache einen Punkt für meinen Referenzpunkt, eine Markierung bei 100mm vom Referenzpunkt und einen Punkt bei 120mm vom Referenzpunkt, siehe Abbildung 20.

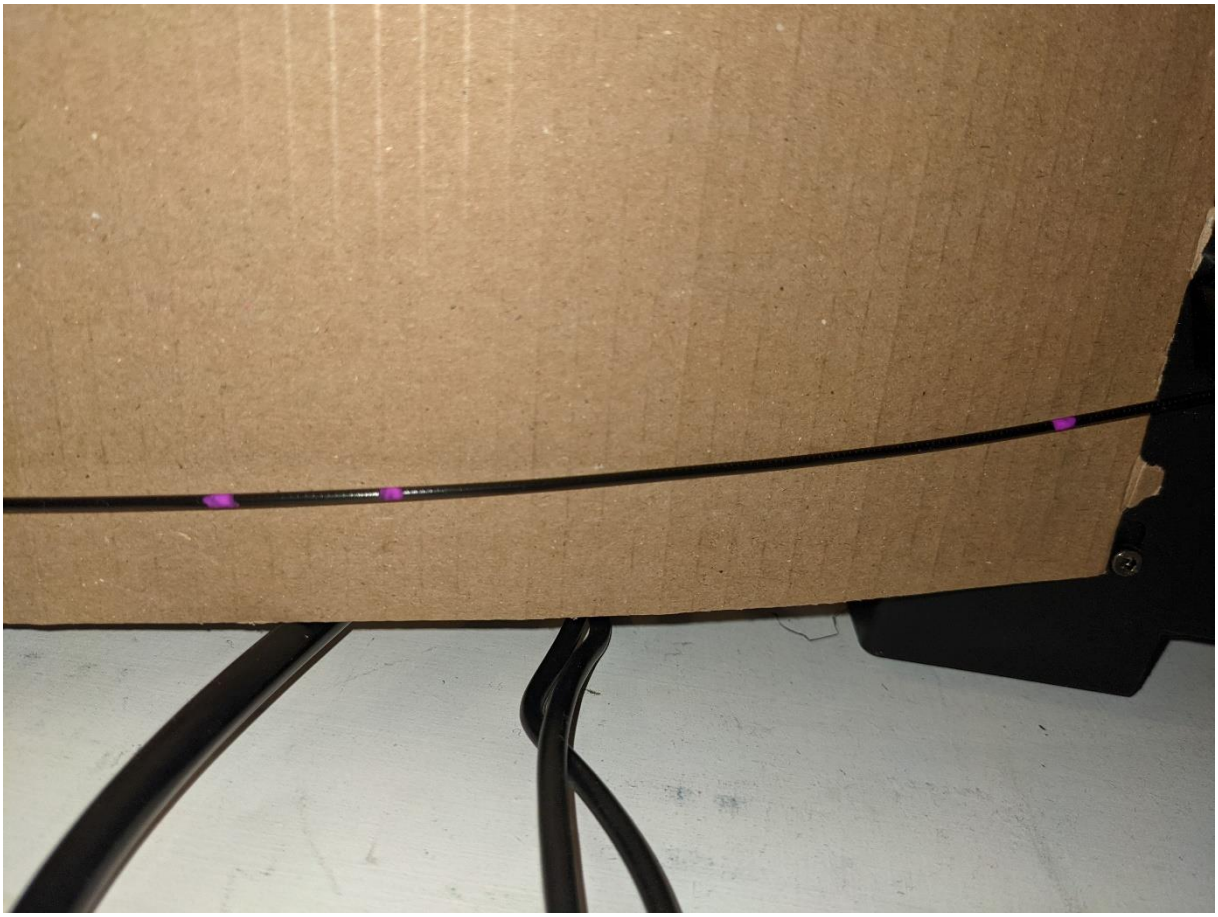

Abbildung 20: Filemant mit Markierung für Abstände

Danach wird der Referenzpunkt so gut wie nur möglich an den Einzug vom Extruder gefahren, siehe Abbildung 21.

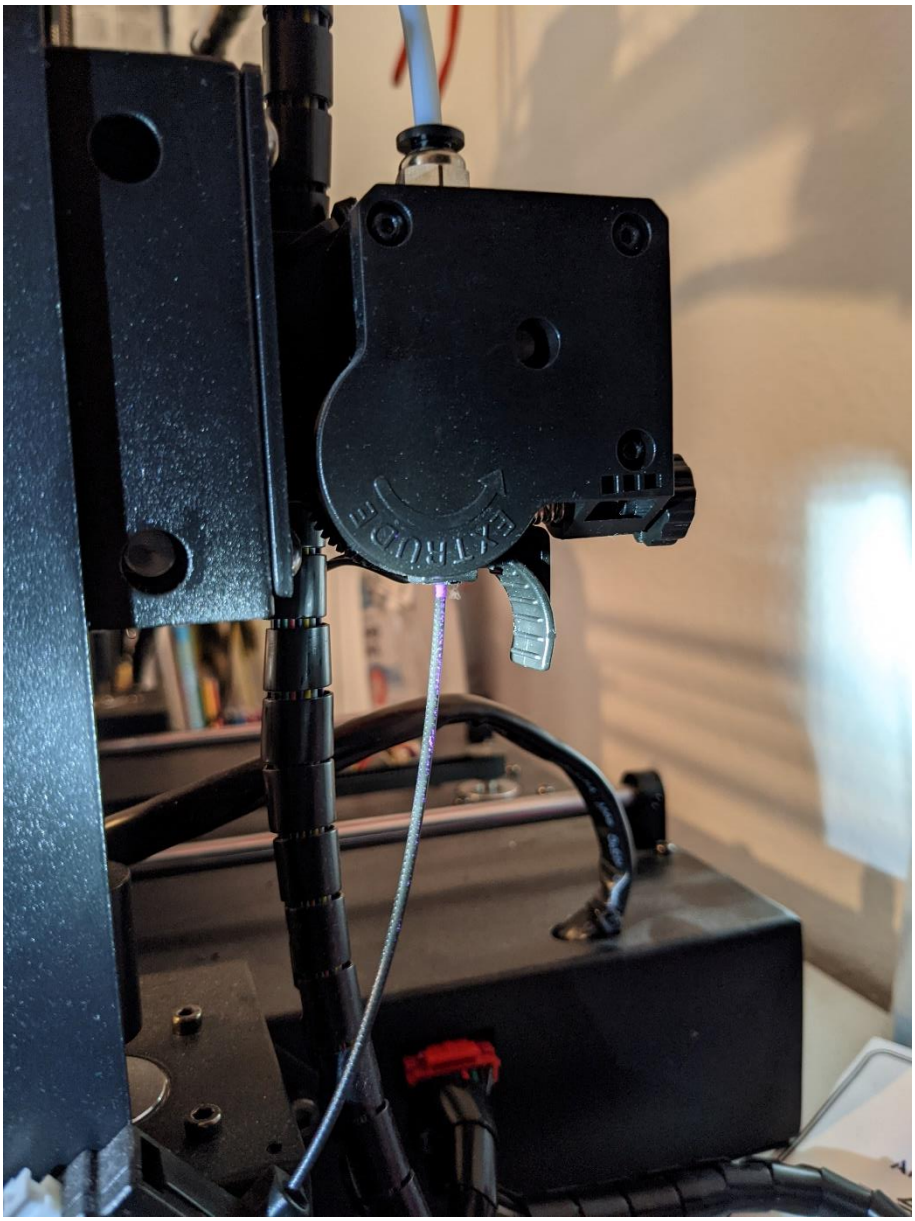

Abbildung 21: Referenzpunkt am Einzug vom Extruder positioniert

Damit das gelingt nutze ich in meinem Fall OctoPrint mit dem Reiter **Steuerung**, der es mir unter anderem erlaubt den Extruder, aber auch die einzelnen Achsen zu bewegen, siehe Abbildung 22. Man kann aber auch das Filament einfach per Hand einschieben, was je nach Extruder leichter oder schwieriger sein kann.

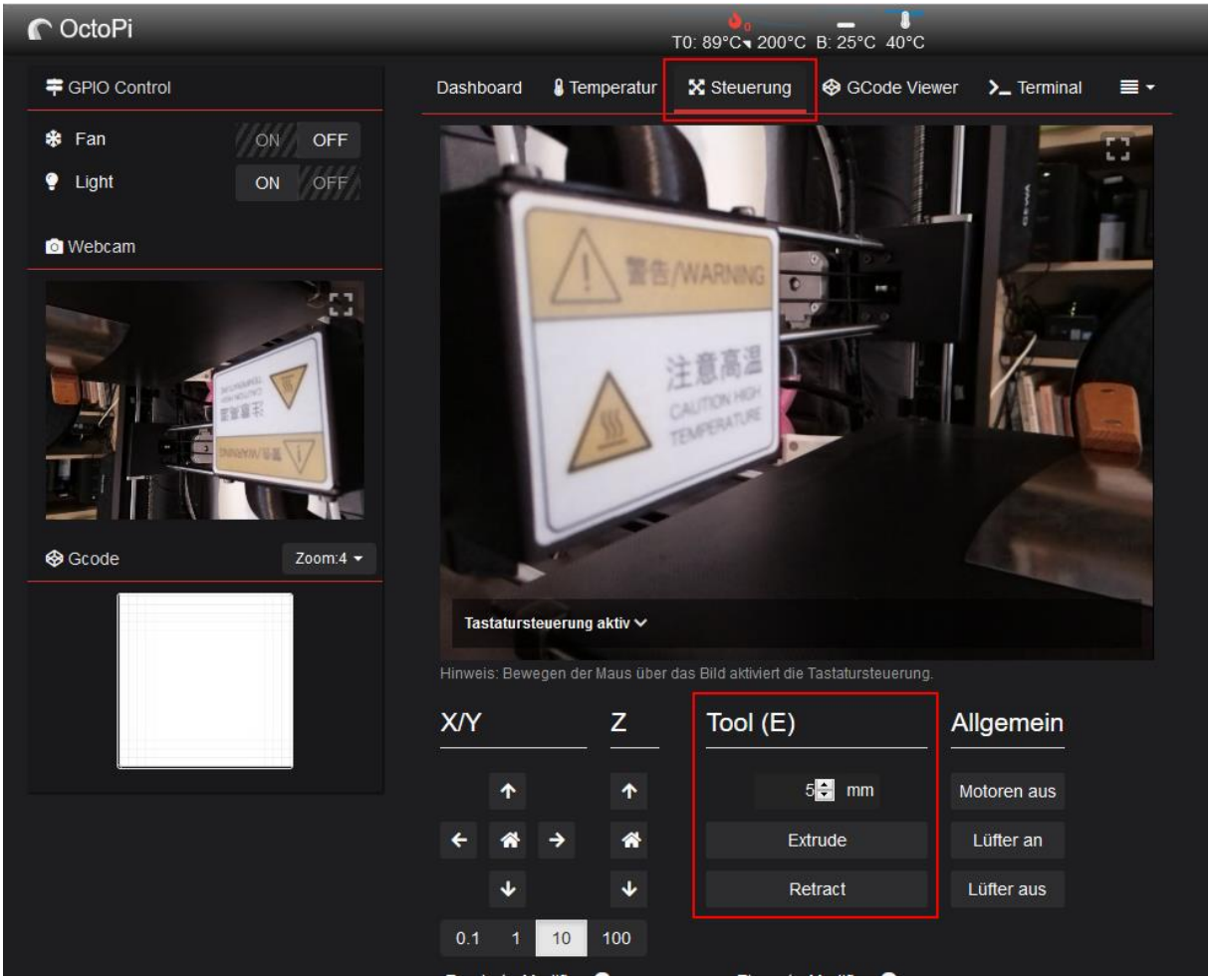

Abbildung 22: Der Tab Steuerung von OctoPrint

Nun lasse ich den Drucker 100mm Filament einziehen und warte darauf, bis der Vorgang beendet ist. Dabei nutze ich in OctoPrint im Tab Steuerung die entsprechenden Steuereinstellungen. Das Einziehen des Filaments sollte nicht lange dauern. Sollte nun die 100mm-Markierung direkt am Anfang des Extruders sein, so ist kein weiterer Schritt notwendig, jedoch wird das in den meisten Fällen nicht passieren. Entweder wird der Einzugsvorgang vor der 100mm-Markierung aufhören oder danach. In meinem Fall hat der Extruder 6,5mm zu viel Filament eingezogen, da von der 120mm-Markierung zum Einlass des Extruders nur noch 13,5mm gemessen wurden, siehe Abbildung 23.

Abbildung 23: Messung wie viel Filament eingezogen wurde

Damit ich nun weiß, wie viele Steps der Drucker aktuell für 100mm annimmt, brauche ich über das Terminal diese Steps vom Drucker. Mit dem Marlinbefehl aus Code 8, liefert mir das Terminal die gewünschte Antwort, siehe Abbildung 24 rote Umrandung.

```
M92
```

Code 8: Die aktuellen Steps für die Achsen und Extruder bekommen

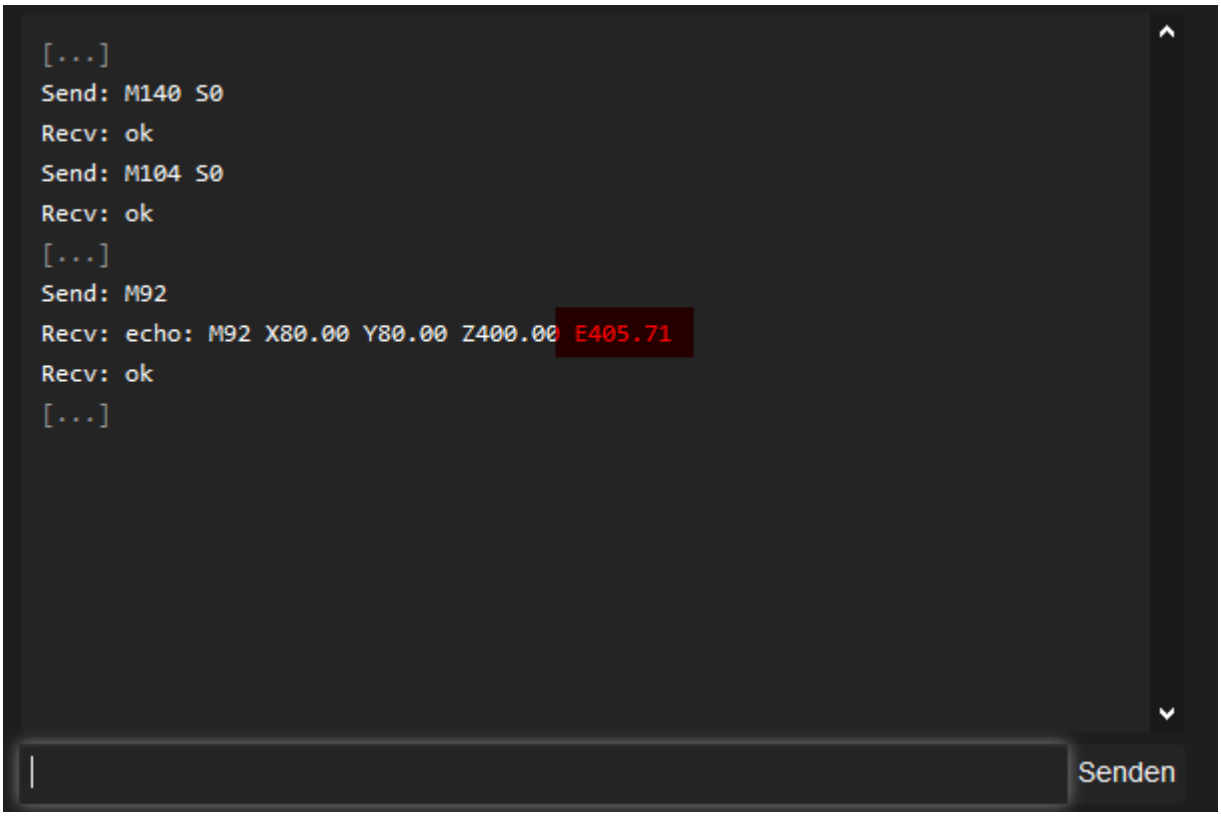

Abbildung 24: Rückgabe für den Befehl M92 im Terminal

Damit ist nun bekannt, dass der Extruder 405.71steps pro 100mm macht. Nun wird es Zeit für ein bisschen rechnen, damit die Extrudersteps auch passen. Die Formel sieht wie folgt aus:

Neuer Extruderstepwert = 100mm/(120mm – gemessener Rest) * Aktuelle Steps

In meinem Fall würde dann die Formel wie folgt aussehen:

Neuer Extruderstepwert = 100mm/(120mm – 13,5) * 405,71
Neuer Extruderstepwert = 380,95 (vorletzte Nachkommastelle gerundet)

Diesen Wert nehme ich und kopiere mir die Ausgabe vom Terminal und korrigiere den Wert für E, sodass mein neuer M92- Marlinbefehl wie in Code 9 aussieht.

```
M92 X80.00 Y80.00 Z400.00 E380.95
```

Code 9: Neue Steps für Extruder übermitteln

Danach noch ein M500 hinterher und die aktuellen Steps sind beim nächsten Neustart noch hinterlegt.

Damit haben wir einen großen Schritt in Richtung perfekten Druck gemacht. Der Drucker ist nun leiser und kann die Achsen präziser ansteuern und der Extruder gibt nun die korrekte Menge an Filament während dem Druck. Damit können wir im Grunde problemlos drucken.
Im nächsten Teil möchte ich OctoPrint ein bisschen näher vorstellen und auch die Installation zeigen, sowie das Thema Mesh Bed Leveling näher erläutern. Im ersten Teil bin ich darauf schon eingegangen, was ich zum einen kurz wiederholen werde, zum anderen eine weitere softwareseitige Methode mit der aktuellen Firmware von knutwurst zeigen möchte. Dies ist eine weitere Tuning-

Maßnahme, welche aber nur bei gekrümmten Heatbeds benötigt wird oder man der Meinung ist, dass der Druck damit noch etwas Verbesserung benötigt. Außerdem möchte ich Ihnen einige Druckvorlagen auf thingiverse.com vorstellen, mit denen Sie leicht sehen können, wie es um die Druckqualität Ihres 3D-Druckers steht.

Dieses und weitere Projekte finden sich auf GitHub unter https://github.com/M3taKn1ght/Blog-Repo.