Print Monitoring

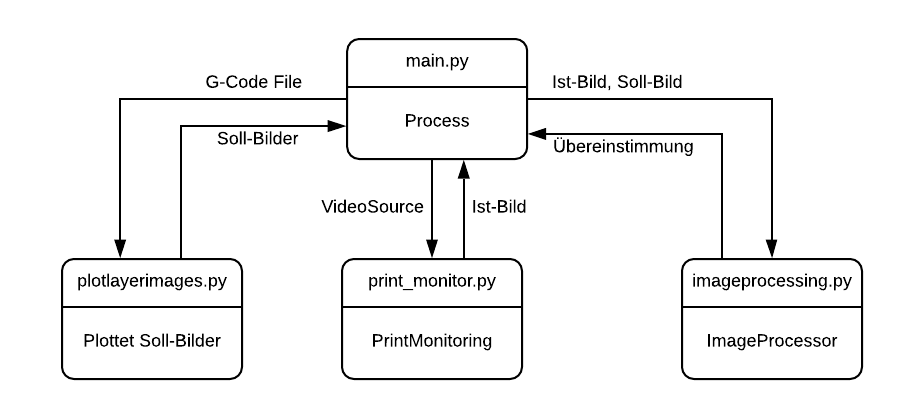
# Ausgangslage

Die meisten FDM-3D-Drucker besitzen keine Fehlerüberwachung, auch wenn einen Druckfehler auftritt, wird weiter gedruckt und somit Filament verschwendet oder der Drucker kann beschädigt werden. Ziel des Projektes ist es die Qualität eines FDM-3D-Drucks durch eine Kamera live zu überwachen, um so den Druck beim Auftreten eines Fehlers automatisiert pausieren oder abbrechen zu können.

# Ziel

Bevor ein Druck gestartet wird, soll für jeden Layer des Drucks ein Soll-Bild geplottet werden. Während dem Drucken wird dann nach jedem gedruckten Layer ein Ist-Bild aufgenommen. Aus dem Ist- und Soll-Bild wird dann eine prozentuale Übereinstimmung berechnet. Anhand dieser Übereinstimmung kann dann entschieden werden, ob der Auftrag korrekt gedruckt wird. Grundsätzlich soll das Tool Livebilder einer Webcam verarbeiten. Im Rahmen der Projektarbeit sollen aber auch das Verwenden von zuvor aufgenommenen Videos möglich sein. Die Kommunikation mit dem Drucker (starten, stoppen einen Auftrags) ist nicht Teil dieser Arbeit und wird in einem zweiten Schritt implementiert. Das fertige Tool soll schlussendlich im MakerStudio der FHNW Brugg-Windisch eingesetzt werden.

# Umsetzung



Das Tool wurde mittels Python umgesetzt. Um eine Überwachung zu Starten oder eine neue Kamera zu konfigurieren wurde in der main.py ein Command-Line-Interface enzwickelt. Wie das CLI verwendet wird und welche Packages installiert werden müssen wird in dem separaten Dokument «readme.pdf» erklärt.

## Plotten der Soll-Bilder

Um ein 3D-Modell drucken zu können wird die Datei (Normalerweise .stl) von einem Slicer-Programm (Beispielsweise Cura) in einen GCode verarbeitet. Die GCode-Datei besteht aus einer simplen Sammlung von X, Y und Z Koordinaten, welche vom Drucker schrittweise angefahren werden sollen, bis schlussendlich das komplette Modell gedruckt ist.

In der Datei «plotlayerimages.py» wird dieser Gcode gelesen, die X und Y Koordinaten werden mittels MatPlotLib Schicht für Schicht geplottet und als Bild abgespeichert. Somit entsteht eine Sammlung von Soll-Bildern. Für das Plotten des GCodes wurde eine Library von *zhangyaqi1989* verwendet (<https://github.com/zhangyaqi1989/Gcode-Reader>)

## Aufnehmen der Ist-Bilder

Sobald sämtliche Soll-Bilder generiert wurden, wird der GCode am Drucker übergeben und der Druckauftrag wird gestartet. In dem Script «print\_monitor.py» wird der Drucker nun überwacht, um zu erkennen, wann einen Layer fertig gedruckt wurde. Das Überwachen des Druckes funktioniert folgendermassen:

Der GCode wird so generiert, dass der Kopf des Druckers nach jedem gedruckten Layer an seine Ausgangsposition fährt. Diese Ausgangsposition muss nach dem Positionieren der Kamera einmalig manuell auf dem Bild ausgewählt werden. Siehe Dokument «Anleitung» Kapitel «Konfiguration»

Für jedes Video-Frame werden folgende Schritte ausgeführt:

1. Konvertieren des Bildes in den **HSV-Bereich**
2. Mittels OpenCV einen **Color-Treshhold** auf das Bild anwenden. Um den Druckkopf freizustellen. Danach ein **Errode** und **Dilate** um das Binärbild zu bereinigen.
3. Das berechnete Binärbild wird mit den Binärbildern der letzten drei Frames **And-Verknüpft.** So können weitere Fehler eliminiert werden und eine höhere Stabilität wird erreicht.
4. Es wird überprüft, ob sich die BoundingBox der Binärbildes innerhalb oder ausserhalb des definierten Ausgangspositions-Bereichs befindet.
5. Die gewonnene Information wird in eine StateMachine gespiesen, welche dann Aussage über den Status des Druckers macht. Die möglichen Drucker-Stadien sind:
   1. STARTING
   2. LAYER\_PRINTING
   3. LAYER\_FINISHED
   4. PRINT\_FINISHED

Sobald in den Status LAYER\_FINISHED gewechselt wird, wird das aktuelle Frame als IST-Bild an das main.py übergeben, wo es dann mit dem passenden Soll-Bild verglichen wird.

## Berechnen der Übereinstimmung

Aus den ersten beiden Schritten wurde ein Soll und Ist-Bild für einen gedruckten Layer generiert.

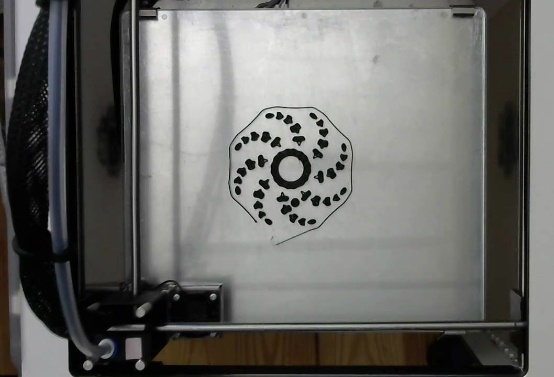
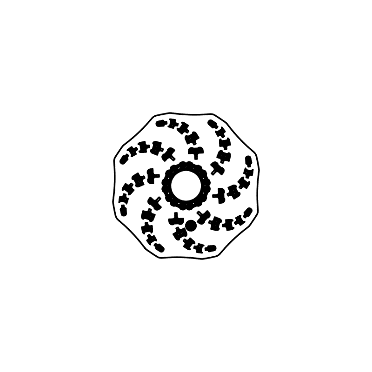


Abbildung 1: Soll-Bild Abbildung 2 Ist-Bild

In einem nächsten Schritt werden diese beiden Bilder verglichen und eine prozentuale Übereinstimmung wird berechnet. Dies geschieht in der Datei imageprocessing.py mit der Funktion getMatch(..)

### Freistellen des Druckes

Zuerst muss der Druck auf dem Ist-Bild freigestellt werden. Dies wurde folgendermassen umgesetzt:

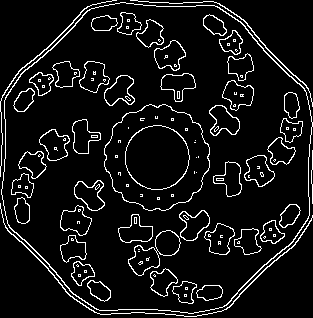
Bevor der Druck gestartet wird, wird durch das PrintMonitoring ein Bild des leeren Druckers angefertigt. Dieses Bild wird als Maske für das Ist-Bild verwendet werden. Dafür wird ein Canny-Edge-Dedector auf das Bild angewendet. Die Berechneten Kanten werden dann mittels einem Dilate vergrössert. (Code in imageprocessing.getMask(…))

Sobald die Maske erstellt ist, wird ein Canny-Edge-Dedector auf das Ist-Bild angewendet. Danach werden die Kanten des Ist-Bildes maskiert und es entsteht ein freigestelltes Bild der Kanten des Druckes.

### Berechnen der Übereinstimmung

In einem weiteren Schritt werden die Kanten des Soll-Bildes detektiert und das Bild wird auf sein BoundingRect zugeschnitten. Das zugeschnittene Soll-Bild wird nun als Template für das OpenCV Templatematching verwendet. Der Templatematching Algorithmus funktioniert folgendermassen: Das Template wird Pixel für Pixel über das maskierte Ist-Bild geschoben, wobei nach jedem Pixel die Überlappung zwischen Template und Ist-Bild berechnet wird. Als Resultat wird die maximale Überlappung, sowie die Koordinaten der maximalen Überlappung, ausgegeben. Verschiedene Versuche ergaben, dass der Algorithmus zuverlässiger funktioniert, wenn er lediglich auf die Kanten des Druckes angewendet wird, anstatt auf das ganze Binärbild von Druck und Template. (<https://docs.opencv.org/2.4/modules/imgproc/doc/object_detection.html?highlight=matchtemplate#cv2.matchTemplate>)

Abbildung 3 Template (links) und gefundene Position und Grösse in Ist-Bild (rechts)



Um das TemplateMatching skalierungsinvariant und somit stabiler umzusetzen, wird der Algorithmus für verschiedene Skalierungen des Template durchgeführt. Als Resultat wird der grösste Match aller Durchgänge verwendet. (<https://www.pyimagesearch.com/2015/01/26/multi-scale-template-matching-using-python-opencv/>)

Somit ist nun die exakte Position und Grösse des Druckes im Ist-Bild bekannt.

Mit dieser Information wird nun das Soll-Bild auf die Grösse des Ist-Bildes gebracht, und der Druck innerhalb des Bildes wird so positioniert, dass er sich an derselben Position wie der Druck auf dem Ist-Bild befindet. Das Resultierende Soll-Bild ist in Abbildung 4 dargestellt.

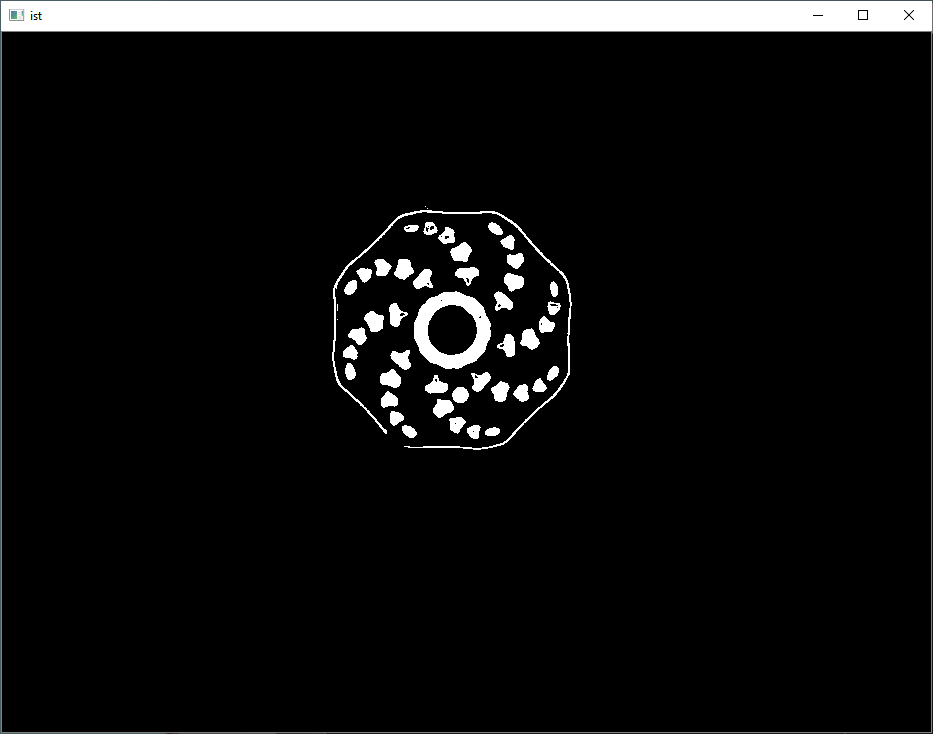
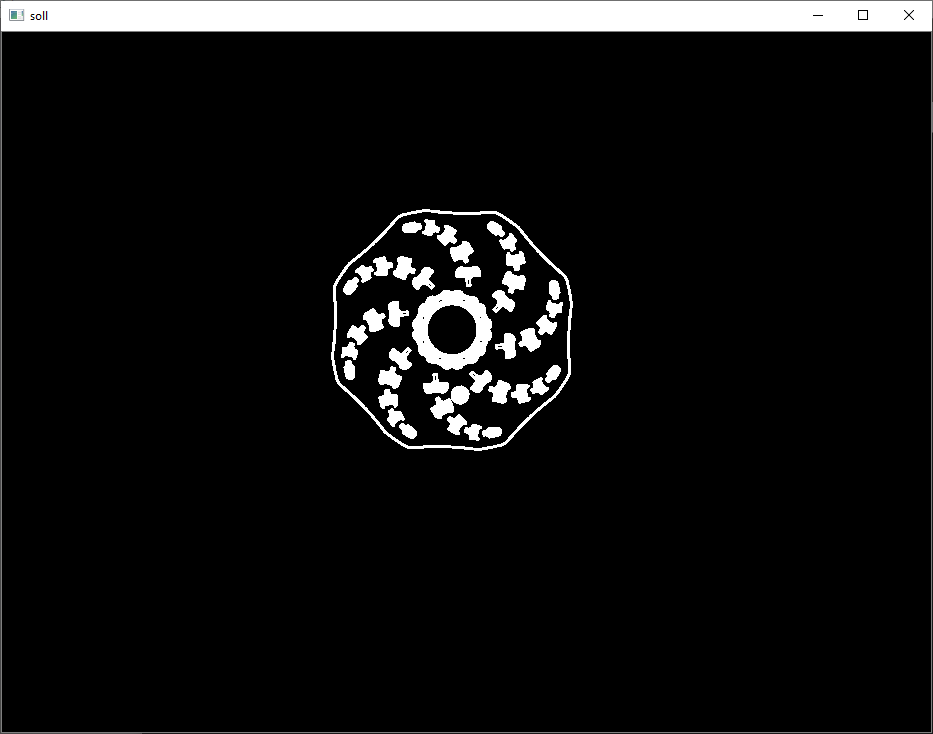


Abbildung 4 Resultierendes Soll-Bild mit Druck an Position von Druck in Ist-Bild

Abbildung 5 Ist-Bild

Zur Berechnung der Übereinstimmung werden die beiden binären Ist- und Soll-Bilder mit einem bitweisen XOR verglichen. Daraus entsteht ein Differenzbild (Abbildung 6), woraus sich schlussendlich die prozentuale Übereinstimmung berechnen.



Abbildung 6 Differenzbild

## Konfiguration

Um das Tool für eine neue Kamera oder eine neue Videodatei verwenden zu können, muss dies einmalig manuell konfiguriert werden. Die ROI des Druckers und die ROI des Druckkopfes müssen ausgewählt werden. Die Konfiguration wird danach in der Datei settings.ini abgelegt. Genaue Instruktionen befinden sich in der Datei *readme.pdf*.

## Hardware

**Raspberry**

**Kamera**

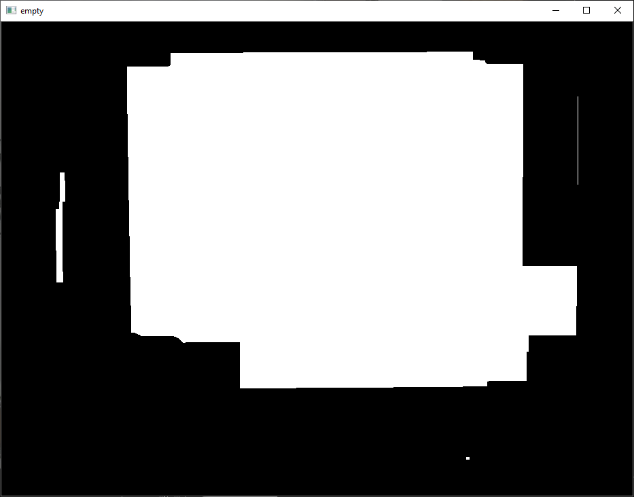
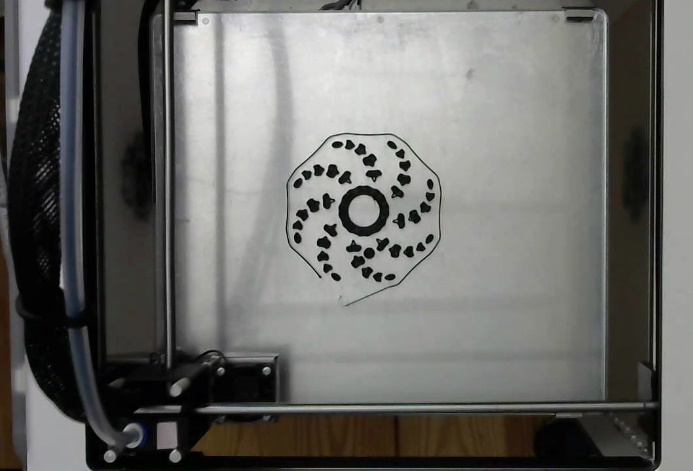
**Beleuchtung**

## Tests

|  |  |  |  |
| --- | --- | --- | --- |
| TestNr 1.  VideoSource: testprint4.mp4  Gcode: Octpus\_PreProcessed.gcode | | | |
| LayerNr. | Ist | Soll | Resultat |
| 1 |  |  | 0.929  -> OK |
| 2 |  |  | 0.901  -> OK |
| 3 |  |  | 0.884  -> OK |
| 4 |  |  | 0.871  -> OK |
| 5 |  |  | 0.851  -> OK |
| 6 |  |  | 0.846  -> Druck wird abgebrochen |
| Test erfolgreich.  Ab Mitte des 2. Layers war die Düse des Druckers verstopft und es konnte kein Material mehr aufgetragen werden. Der Druckfehler nahm mit jedem Layer zu und erreichte den Grenzwert von 85%, wo der Druck abgebrochen wurde. | | | |

# Fazit

Die Projektarbeit wurde stabil und zuverlässig umgesetzt. Häufige Druckfehler, wie zum Beispiel eine verstopfte Düse, zu wenig Filament oder fehlende Stützstrukturen werden erkannt. Somit kann das Tool in einer produktiven Umgebung eingesetzt werden, um den Verschleiss der Drucker und das Verschwenden von Filament zu verkleinern.



Druckfehler, welche auf der Draufsicht des Druckes nicht sichtbar sind, können aber nicht erkannt werden. Das Tool könnte deshalb noch mit einer zweiten Kamera, welche seitlich auf den Druck gerichtet ist, erweitert werden, um eine komplette Qualitätsüberwachung zu garantieren.

Zudem kann nicht die komplette Druckplatte überwacht werden. Aufgrund der gewählten Methode zur Freistellung des Druckes, wird die maximale Grösse des Druckes um zirka 5 – 10 cm kleiner. In Abbildung 7 ist der berduckbare Bereich hell dargestellt.

Abbildung 7 Maskierter Druckbereich

Zudem funktioniert das Freistellen nur bei dunkeln Filamenten, welche ausreichend Kontrast zu der Druckplatte haben, genügend stabil. Um weitere Farben zu unterstützen, müssten die Grenzwerte des verwendete Edge-Dedectors und der Treshholds dynamischer gewählt werden.

# Abbildungsverzeichnis

[Abbildung 1: Soll-Bild Abbildung 2 Ist-Bild 3](#_Toc42503808)

[Abbildung 3 Template (links) und gefundene Position und Grösse in Ist-Bild (rechts) 4](file:///C:\Users\Manuel\Documents\repos\bverproject\Dokumentation.docx#_Toc42503809)

[Abbildung 4 Resultierendes Soll-Bild mit Druck an Position von Druck in Ist-Bild 4](file:///C:\Users\Manuel\Documents\repos\bverproject\Dokumentation.docx#_Toc42503810)

[Abbildung 5 Ist-Bild 4](file:///C:\Users\Manuel\Documents\repos\bverproject\Dokumentation.docx#_Toc42503811)

[Abbildung 6 Differenzbild 5](#_Toc42503812)

# Quellen

Open Source Computer Vision Library, Itseez, 2015, https://github.com/itseez/opencv