POLITECNICO DI TORINO

Collegio di Ingegneria Informatica, del Cinema e Meccatronica

Tesi di Laurea Magistrale

DESIGN AND DEVELOPMENT OF A SYSTEM TO DETECT DEFECTS IN 3D PRINTING THROUGH ANALYSIS OF LAYER IMAGES

Relatore:
Prof. LUCA IULIANO
Corelatori:
Prof. PAOLO MINETOLA
MANUELA GALATI

Candidato:
MANKIRAT SINGH KHANDPUR

October 2019

1 Acknowledgments

First and foremost I am highly grateful and indebted to my supervisor, Prof. LUCA IULIANO who was kind enough to let me pursue the thesis at Politecnico Di Torino for my master's studies.

I would also like to extend my gratitude to Prof. PAOLO MINETOLA and MANUELA GALATI for supporting and guiding me all along. Their motivation and encouragement were like the beacon of light, enlightening my path with wisdom and knowledge. It would not be an exaggeration to say, that without their support I would not have been able to complete my thesis.

Last but not least, I am thankful to my family for being the constant source of emotional support and being there for me always. My mother's love has always helped me to overcome the difficulties. My brother's wise words have guided me to the right path whenever I strayed away from my goal. I would also like to remember my late father who I am sure is looking down upon me from the skies. I hope I have made him proud.

2 Abstract

3D Printing is a process for making a physical object from a three-dimensional digital model, typically by laying down many successive thin layers of a material. It brings a digital object (its CAD representation) into its physical form by adding layer by layer of materials[1]. Like any other process, no matter how perfect we try to make them, defects do creep in. As such, defects like missing filament, abnormal printing etc. could be present in our product. This thesis describes the idea of finding defects during the ongoing process of 3D printing on each layer. The goal is achieved by developing a system to be attached to any open 3D printer with minimal interference on its functionality or parameters. The first step is to analyze the given 3D printer to extract the working parameters mainly start-up conditions and stopping conditions. The work is divided into four major sections which include creating a reference image from Gcode, modifying Gcode file according to the image acquisition system, followed by acquisition of layer images from actual 3D print and finally comparing the reference image and layer image in order to find any possible defects. In this study, the Matlab codes are developed to generate the image and to analyse with layer image. And the integration of the system has also been proposed. Using these concepts the results were analyzed and are mentioned in the following chapters.

Keywords: 3D printing, Matlab, Gcode, Image acquisition, Image processing.

Contents

List of Figures

List of Tables

3 Introduction

3.1 Overview

3-D printing is an additive manufacturing (AM) technique for fabricating a wide range of structures and complex geometries from three-dimensional (3D) model data. The process consists of printing successive layers of materials that are formed on top of each other. This technology has been developed by Charles Hull in 1986 in a process known as stereolithography (SLA), which was followed by subsequent developments such as powder bed fusion, fused deposition modelling (FDM), inkjet printing and contour crafting (CC)[2].

3.2 Fused Deposition Modeling

Fused Deposition Modeling (FDM) is an additive manufacturing process that belongs to the material extrusion family. In FDM, an object is built by selectively depositing melted material in a pre-determined path layer-by-layer. The materials used are thermoplastic polymers and used in a filament form[3]. For this thesis we will be working on FDM type printers.

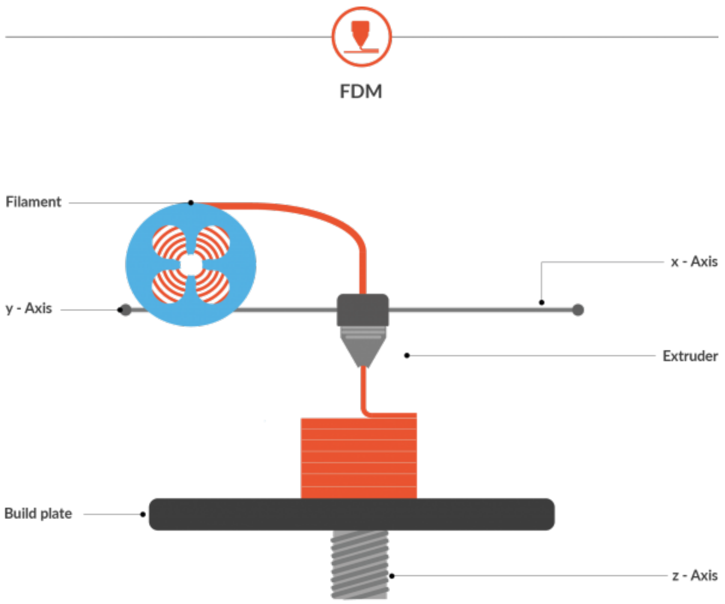


Figure 1: FDM process [4]

This FDM printing process mainly consists of five major steps:

1. CAD modelling, which can be an exact replica or scale models.
2. Tesselation of the 3D model.
3. Orientation and slicing the model into different layers.
4. Selecting material and setting the printing parameters for desired finish and strength.
5. Printing and post-processing for desired surface finish.

The main physical parameter which we set during step 3 and 4 are as follows:

- Layer thickness
- Infill pattern
- Infill Density
- Printing Temperature
- Extrusion flow rate
- Build plate temperature
- Print speed
- Travel speed
- Support structure density
- Cooling fan speed

All these parameters conclude in determining print quality and printing time. Apart from these, many parameters which directly affect print quality are as follows

- Filament exposure to direct moisture
- Printer maintenance
- Ambient temperature
- Printer bed adhesion and cleanliness

The main characteristics of the FDM process are summarized in Table 1

Materials	Thermoplastics (PLA, ABS, PETG, PC, PEI etc)
Dimensional accuracy	$\pm0.5\%(lowerlimit \pm 0.5mm)$ $(desktop)$ $\pm 0.15\%(lowerlimit \pm 0.2mm)$ $(industrial)$
Typical build size(mm)	$200x200x200(desktop)$ $1000x1000x1000(industrial)$
Max. Resolution	50 to 400 microns
Support	Not always required(dissolvable available)

Table 1: Fused Deposition Modeling (FDM) [3]

3.3 Advantages of FDM 3D printing

The main advantages of the FDM process are:

- It is a very economical process for producing customized thermoplastic parts and other functional prototypes.
- It is one of the most used technology in the prototyping industry.
- This technology has one of the shortest lead time.
- Due to its small machine size and low capital cost, it is widely used.

3.4 Limitations of FDM 3D printing

The main drawbacks of the FDM process are:

- FDM has the lowest accuracy and resolution compared to other methods of additive manufacturing.
- In many cases post-processing is required to obtain smooth surfaces.
- The layer adhesion mechanism makes FDM parts inherently anisotropic[3].

3.5 Common Defects

In this section some of the common defects are discussed which are found with FDM 3D printing:

- **No Extrusion**

Printer does not extrude plastic at the beginning of the print which could happen because either extruder was not primed before beginning the print or nozzle starts too close to the bed or the filament has stripped against the drive gear or the extruder is clogged.

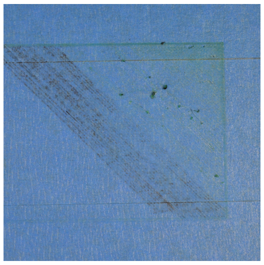


Figure 2: No Extrusion [5]

- **Not Sticking to the Bed**

The first layer does not stick to the bed and the print quickly fails which could happen because either build platform is not level or nozzle starts too far away from bed or first layer is printing too fast.

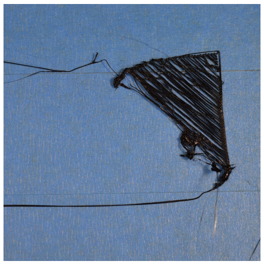


Figure 3: Not Sticking to the Bed [5]

- **Under-Extrusion**

Printer does not extrude enough plastic which leads to gaps between perimeters and infill, The reasons could be either because of low feed rate or incorrect filament diameter.

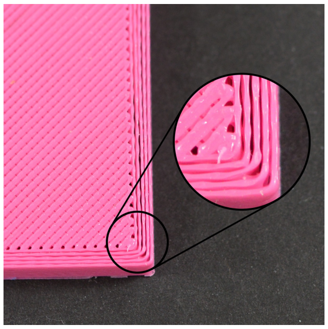


Figure 4: Under-Extrusion [5]

- **Over-Extrusion**

Printer extrudes too much plastic which make prints looks very messy, The reasons could be either because of high feed rate or incorrect filament diameter.

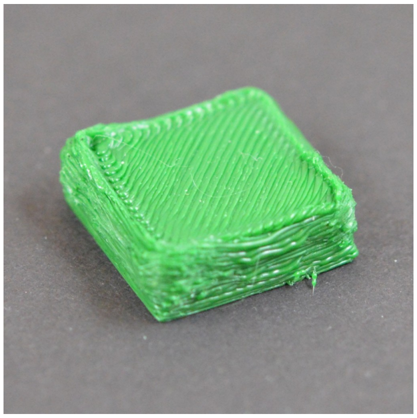


Figure 5: Over-Extrusion [5]

- **Layer Shifting**

Layers are misaligned and shift relative to one another, The reasons could be either because of printer head is moving too fast or part is not properly sticking to the bed.

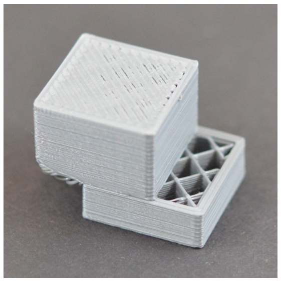


Figure 6: Layer Shifting [5]

- **Warping**

Warping of large parts, particularly with high temperature materials such as ABS, The reason could be non uniform temperature of part during printing.

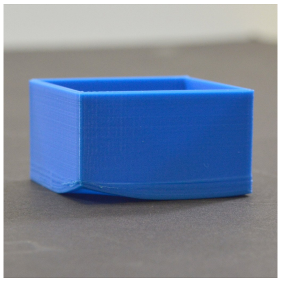


Figure 7: Warping [5]

4 State of the Art

4.1 Current State

In a common 3D printer, there are three-axis namely X, Y, and Z. In most of the printers, the printing bed is fixed on the Z-axis which relates only to vertical movement. And the extruder is mounted on the printer head which can move in direction with X and Y only. To limit the axis movement about 4-6 limit switches placed along the three-axis. Different types of modifications and features can be added which will be explained in the following section.

4.1.1 Dual extruder or changeable printer head

Dual extruder on printer is added to enable dual extrusion of two different kinds of filaments which can be used to print plastic from one extruder and printing supports which enable us for easy removal either by force or by dissolving it. Another way is to use different colour filaments to make multi-colour parts. Also in some cases, the printer head is changeable to make 3d printer also to function as milling or laser engraving machine.

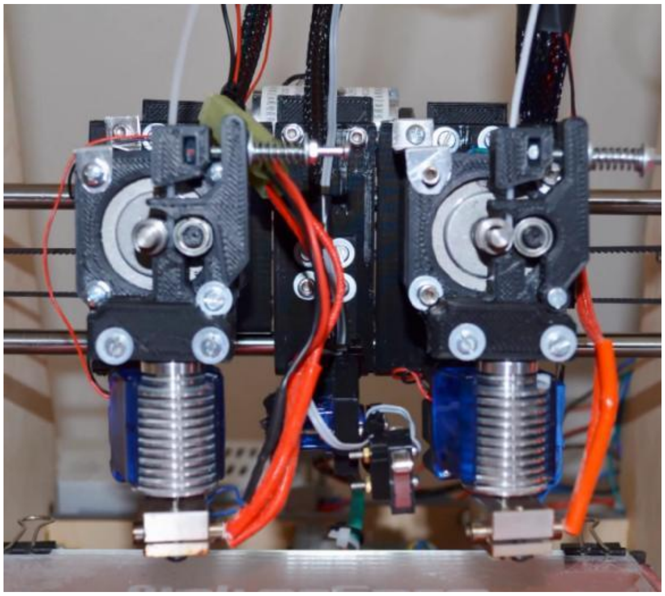


Figure 8: Dual extruder [6]

4.1.2 Heatable Bed

Many printers are now equipped with a bed which can be heated to ensure starting layer adhesion and also to make uniform temperature along the height of the object from bottom to top while printing.

4.1.3 Stepper motor position for extruder

Stepper motor used for pushing the filament is generally positioned to two different positions. Direct Extruders[7] mechanisms are those in which the extruder motor is directly mounted on the extruder body, providing smooth motion of filament.

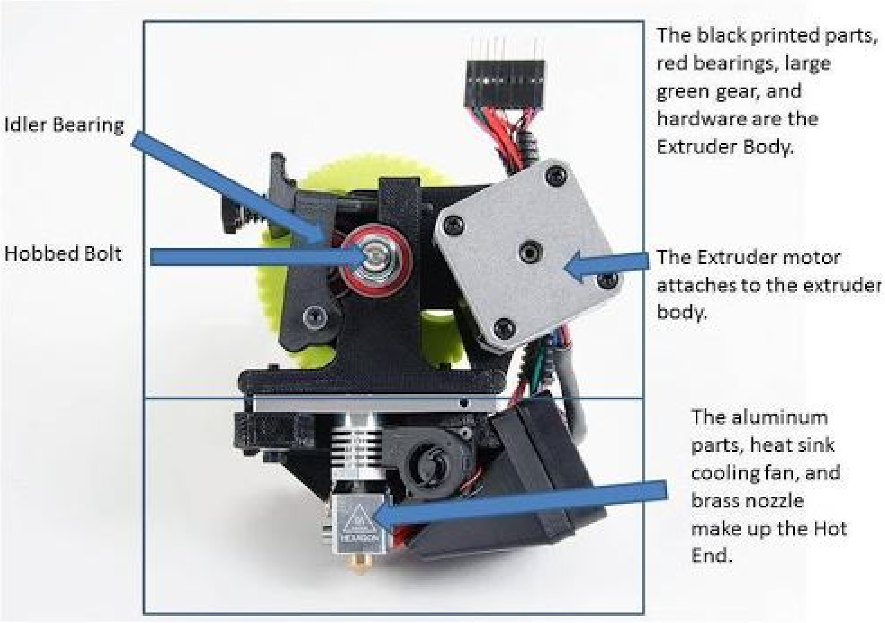

The black printed parts, red bearings, large green gear, and hardware are the Extruder Body.
Idler Bearing
Hobbed Bolt
The Extruder motor attaches to the extruder body.
The aluminum parts, heat sink cooling fan, and brass nozzle make up the Hot End.

Figure 9: Direct Extruders [7]

Whereas in the case of Bowden Extruders[7] the stepper motor is attached to the frame of the printer, this provides low inertia of printer head leading to higher printing acceleration.

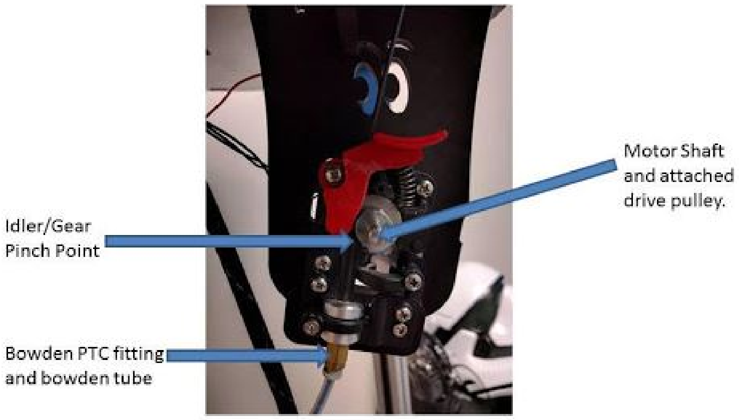


Figure 10: Bowden Extruders [7]

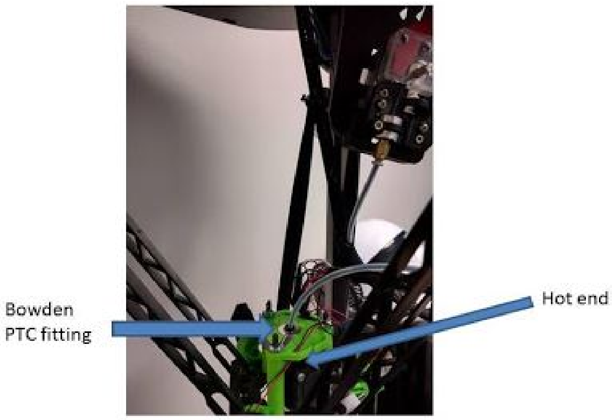


Figure 11: Bowden Extruders Stepper Motor [7]

4.1.4 Bed with Auto-Leveling

Some printers head are equipped with auto bed levelling feature which performs a task that makes the bed always levelled according to the print axis by probing different parts of the bed.

4.2 Process Flow

In order to understand the process of 3D printing following flowchart can be used to define major stages.

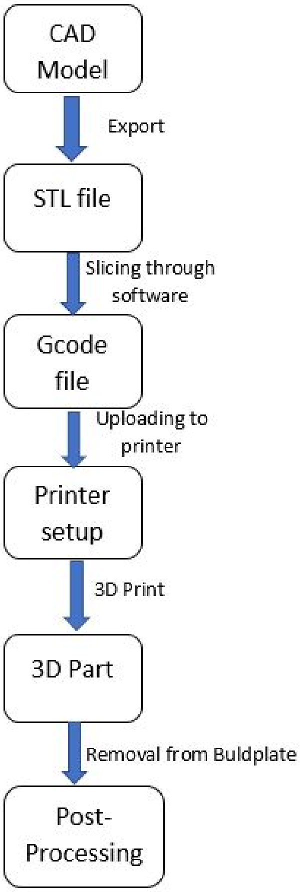


Figure 12: FDM Normal Process Flowchart

4.3 Slicing

Slicing is the process in which the slicing software based on the user layer thickness slices the STL model into a model with small layer thickness. STL files are standard triangulation files which are used to describe the surfaces to be built by the 3D printing. Each triangle in the STL file is described by the three vectors indicating the outward side of the triangle. Slicing is generally done in the Z direction which is the direction of the printing.

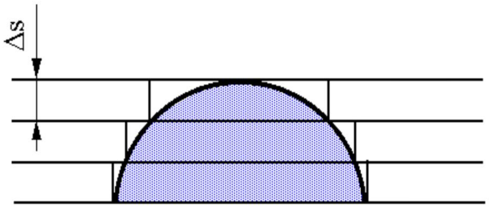


Figure 13: Slicing (A) [8]

The orientation of part on the printing bed generally describe the path in which the part is sliced. It also depends on the nature of support and the build platform.

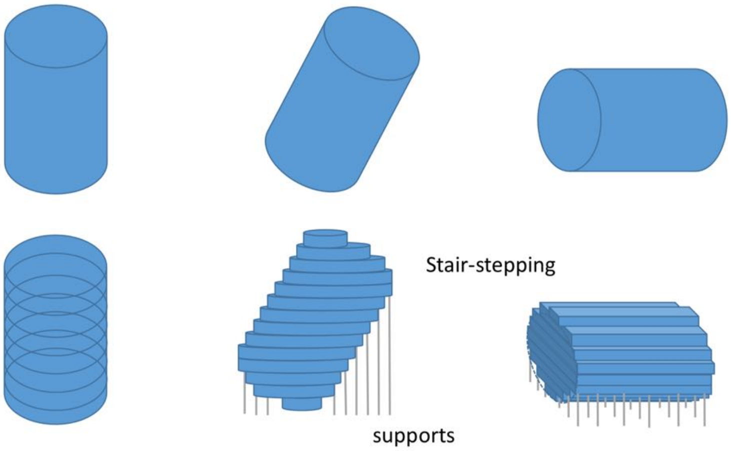


Figure 14: Slicing (B) [8]

4.4 Slicer / Slicing software

The slicer cuts the CAD model into horizontal layers based on the settings set and calculates how much material the printer will need to extrude and how long it will take to do it. All of this information is then bundled up into a Gcode file which is sent to the printer. Also, slicer settings do impact the quality of your print so it's important to have the right software and settings to get the best quality print possible[9]. For this thesis, the slicing software used was CURA 2.4.0 [10]. The software is of open-source license by Ultimaker.

The main advantages of Cura 2.4.0 are:

- It has seamless integration with CAD software like Solid Works.
- Multiple Compatible file types: STL, OBJ.
- Open-source and easy to use software with multiple plug-ins available online.
- It offers a wide range of custom settings which can be used to improve print quality.

4.5 Post-Processing

FDM 3D prints have a rough surface finish, the layer lines are clearly visible and more often these parts require post-processing. This step can also lead to dimensional inaccuracy if properly not accounted for. [11]

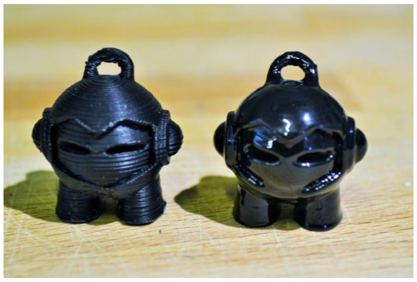


Figure 15: Post-Processing [11]

This post-processing step can be categorised in mainly two categories

- Support removal
- Surface finish

4.5.1 Support removal

Support removal technique basically depends upon the type of support generated during the printing process.
Insoluble supports: insoluble material can be PLA, Nylon, PC which are generally the same material used for the print model. The removal of this support is done by hand or by pliers by force. In some printing this removal can sometimes leave rough surface finish.

Soluble supports: Soluble material can be HIPS (generally used for ABS) and PVA (generally used for PLA) which can be dissolved in a chemical solution.

4.5.2 Surface Finish

To obtain the desired surface finish following methods can be adopted

- Sanding
- Vapour smoothing
- Priming and Painting
- Polishing

4.6 Advancement in Additive Manufacturing

Additive manufacturing has been identified as one of the technologies for the current fourth industrial revolution. The latter is based on the connection between physical and digital systems, which involves the use of advanced sensors and the collection of a large amount of information and data (big data) on which to perform complex analyses. The results of these analyses make it possible to make decisions and adaptations in real-time, even remotely via the internet connection. With a view to lean production, these new cyber-physical systems can allow greater flexibility of production in small lots at large-scale costs and a better quality deriving from the reduction of waste, thanks to sensors that monitor production in real-time.The flexibility granted by modern additive manufacturing systems is unique, because it does not require the use of tools and equipment and allows the simultaneous production of components of different geometry, in the same work volume and directly starting from the virtual model (CAD). Added to these advantages are those of greater design freedom, deriving from the possibility of reproducing even very intricate shapes without the geometric complexity leading to a significant decrease in costs. Faced with fewer geometric constraints, the creativity of designers and designers can focus on the functionality of the component or on weight reduction, defining innovative forms that cannot be achieved using traditional technologies. The high costs of systems and materials constitute an obstacle to the adoption of additive technologies. It is found that a greater diffusion of this technology has taken in the racing, aerospace and biomedical sectors, where the cost/benefit ratio is advantageous. Sensors, signal processing algorithms and data storage strategies are under active development, but the additive process can already be considered as the most monitored and controllable among the various production processes. Nowadays, post-production control and inspection procedures significantly affect the time required to produce a component. In the near future, reliable layer-by-layer piece testing during manufacturing will increase productivity and reduce waste-related losses. The additive manufacturing process of a defective part can be interrupted in between for saving of material, energy and resources.

Therefore there is a need for a system enables the user for more controllability for this manufacturing technique.

5 3D Printer

5.1 Fabtotum Replicator

The printer on which the experiments were performed is an open printer named as “FABTOTUM PERSONAL FABRICATOR” by company “FABTOTUM”. [12]

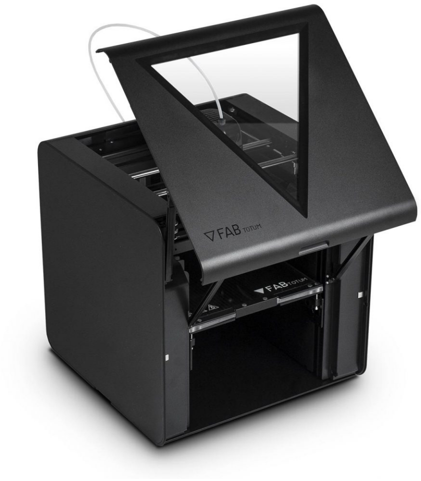


Figure 16: Printer[12]

This printer is featured with a heated bed as shown in figure 17, which will be used to keep the model evenly heated.

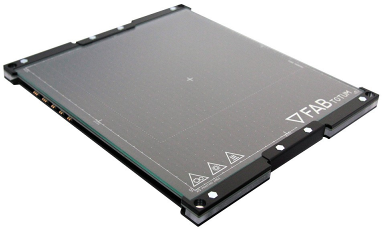


Figure 17: Printer Bed [12]

5.2 Printer head

The Printing Head is a removable extruder, just like any extruder of a 3D printer, it is composed by an heatsink and a melting chamber. The heatsink (also called "cold end") is where the filament is kept solid until in the melting chamber. On the hot end side, the most important part is the nozzle: this last component heats up and deposit the material in thin layers. Here there will be a thermistor and a resistance: the first one will detect the temperature and send the value to the board to record it, the second one will increase or decrease the input value in order to get the needed one when printing.

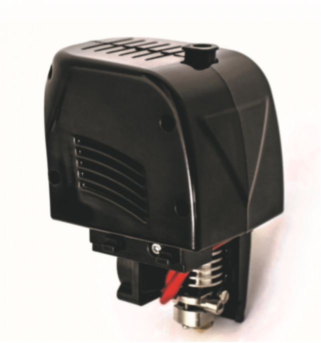


Figure 18: Printer Head [12]

A wide range of filaments can be used, from standard PLA and ABS to flexible (such as TPU), from Nylon to exotic ones such as wood fill and special filled filaments [12]. Also, the Printing Head features exchangeable nozzles, serviceable hotend and more importantly a smooth and detailed surface finish. Different nozzles are shown in figure 19.

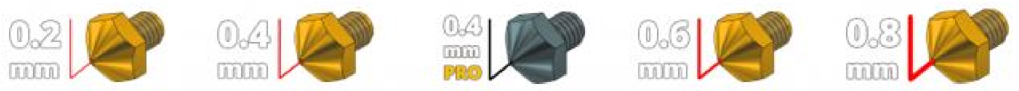


Figure 19: Different nozzles [12]

This printer head can be detached and replaced with other attachments like milling or scanning attachment as shown in figure 20.

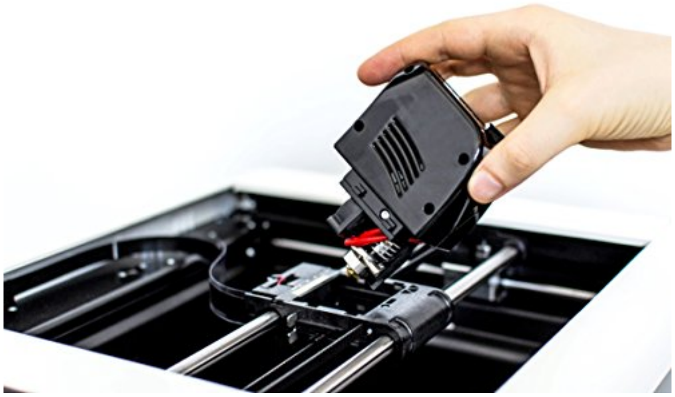


Figure 20: Removable Printer Head [12]

5.3 Other features

This printer can also perform task like milling and scanning by interchanging the printer head with suitable attachment shown in figure 21.

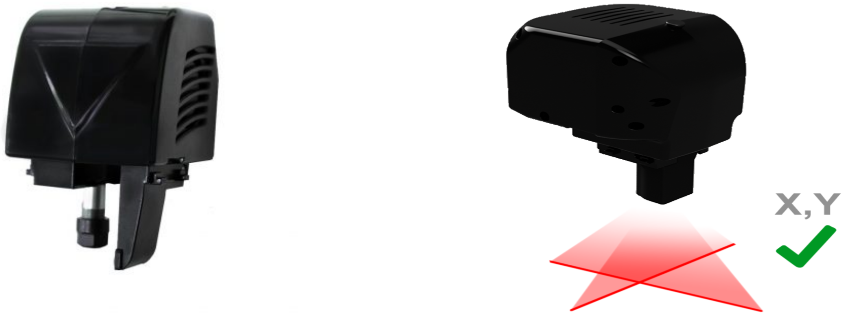


Figure 21: Heads for different functionality

5.4 System Description

5.4.1 Overview

- Classification: FABtotum PRO
- Head: Exchangeable modular heads.
- Capabilities: Hybrid Manufacturing, Digital Acquisition
- I/O: USB Host, Wifi B/g/n, Ethernet
- Control: Standalone Network 3D Printer
- Memory: 16GB onboard
- Size: 366x366x366mm
- Weight <12kg
- Illumination: LED RGB Ambient light

5.4.2 Software, Firmware

- License: Open Source (CC BY-NC-SA 3.0)
- Interface: FABUI, open-source web panel no 3D printing software needed
- FAB UI Compatibility: PC, Mac, tablet, smartphones (responsive web interface)
- Mods and Plugins: Built-in plug-in installer and updater tool
- Updates: automatic with internet connectivity (updates are not mandatory)

5.4.3 3D printing

- Volume: max 214x236x242mm
- Print Z layer height: 50,100 micrometres
- Auto bed Leveling: Assisted, Automated

- Z layer increments: 47 micrometres
- Print to size-ratio: 25%
- Printing Speeds: up to 150mm/s
- Speeds (rapids): up to 400mm/s
- Filament end autopause
- Cooling: Fan/Turbine twin cooling system
- Heated Bed: Hi-quality Glass Heated Bed (Double-faced platform)

5.4.4 Safety

- Fabrication Area: Enclosed
- Safety switches: Front Panel door
- Overvoltage protection: built-in
- Homing Endstops: On all 3 Axis
- Safety Endstops: On all 3 Axis
- Safety Config: Customizable from the FAB UI

5.4.5 Subtractive Mode (milling, engraving)

- Milling volume: 214x236x[milling bit height] mm
- Milling Motor: 200W Brushless motor, automatic RPM correction
- Fan cooling: Twin fan cooling system
- Working Plane: structural aluminium plate, with fixtures (double-face platform)
- Illumination: Working area LED ambient

6 Objective

6.1 Description

To enable more controllability to the user as discussed in section 4.6 , a system is needed to be developed, where its role is to help the user in the identification of defects. This system functionalities can be categorized into different sections as

- Image from Gcode
- Image Acquisition system
- Gcode modifier
- Image processing

Several modifications were made to process flow explained in figure 12 as shown in figure 22.

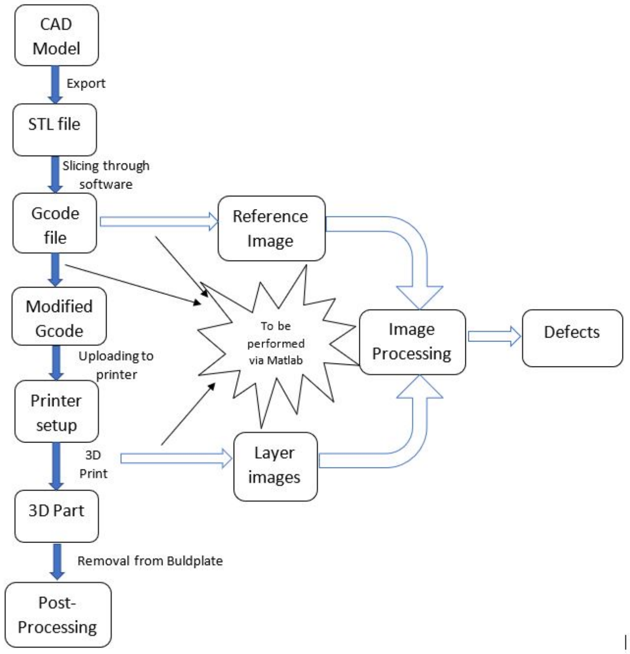


Figure 22: Modified Process Flow chart

6.2 Image from Gcode

6.2.1 Description

The objective of this section is to read each command from the Gcode file and identify the part of code which is used for print action. These commands are used further to make changes to the blank image (All-white pixeled image) for each layer pixel by pixel. This leads to the mapping of all print movements of extruder to be performed by extruder. This section is further explained in section 7.

6.3 Image Acquisition system

6.3.1 Description

This system provides acquisition of images of each layer with minimum modifications done to any Generic 3D printer.

6.3.2 System components

The system comprises of the following parts

- Camera
- Circuit
- Switch mounting

With all these parts integrated into the printer, we are able to determine the timings of image capture to avoid any obstacle in front of the camera while capturing the images. This section is further explained in section 8.

6.4 Gcode modifier

6.4.1 Description

The objective of this section is to make modification to the Gcode file received from the slicer software in order to incorporate our image acquisition system by adding a fixed number of lines of command with variable parameters used to set the movement of the printer head for acquisition. This section is further explained in section 9.

6.5 Image processing

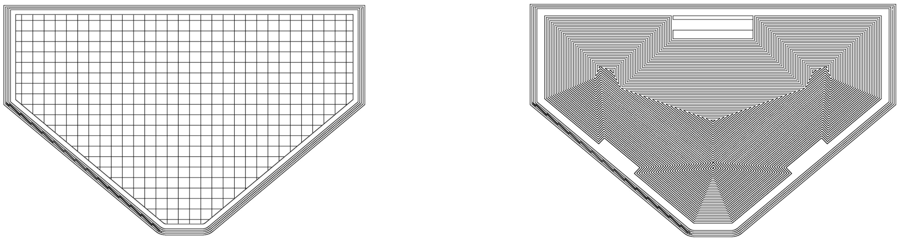
6.5.1 Description

After receiving the images from actual print layers we compare it with the reference images to find possible defects in print. This is achieved by comparing all pixels of the images Also to keep minimal camera distortion the camera calibration is done via Matlab camera calibrator app[13]. This section will be explained later in section 10.

7 Image from Gcode

7.1 Description

For generating layer images a reference image is made with all-white pixels. Then Matlab script requires reference image parameters and name of the file as input. After defining them, we search for the first relevant position in X and Y direction. These coordinates will be used for current position coordinates for the first cycle of the loop only. Then we proceed to the next command line in Gcode file, if the code does not represent code for movement we go to next line in command. Otherwise, if the command is for movement we set the coordinates as the target position. After defining the two positions we check whether the command was for printing or not. If the command was for printing we load the values of the position in two variables which are starting point of the line and the final point of the line then we map the corresponding line on the reference image. This process is repeated until the end of the layer is reached after which we publish the image of the layer and reset the reference image again to all-white pixels image. This process is repeated for all layers until the end of the file.
The sample model used for demonstration is the same model used for sensor mounting attachment explained in section 8.3. The reference image generated from the Gcode is shown in figure 23a and figure 23b with their respective layer image.



(a) Layer 1 from Gcode

(b) Layer 33 from Gcode

Figure 23: Layer Images from Gcode

These images are further used as reference which will be used in section 10.

7.2 Matlab Script

```
clc
close all
clear all
%% Defining Image parameters
im_size=12500;
im=255*ones(im_size);
imG0=im;
imG1=im;
pointscalingF=50;
interpolation=1000;
roundfact=2;
box_size=[60,70];                            % [X,Y]
box_pos=[108,38];                            % LIKE PRINTER [X,Y]
corner_pts=[box_pos(1,2),box_pos(1,1)];
corner_pts_scaled=pointscalingF*corner_pts;
%%
nameoffile='FPF_switch_base.txt';     %name of the file
text1=fileread(nameoffile);
strG0='G0';
% strG1='G1';
G0=strfind(text1,strG0);              %string to find
z=length(text1);
space=' ';
strLAYER=';LAY';
%%  catching first G0 line for position
for i=1:1000
    if text1(1,G0(1,1)+i)==newline
        endline=G0(1,1)+i;
        break
    end
end
line=text1(1,G0(1,1):endline);
line_size=length(line);
%%%%%%%%%%%%%%%%%%%%%%%%% for X %%%%%%%%%%%%%%%%%%%%%

for j=1:line_size
        if line(1,j)=='X'
            L_X=j;
            break
        end
end

for j=1:line_size                              %for X
    if line(1,L_X+j)==space
        L_spaceX=L_X+j;
        break
    end
    if line(1,L_X+j)==newline
        L_spaceX=L_X+j;
        break
    end

end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


%%%%%%%%%%%%%%%%% for Y %%%%%%%%%%%%%%%%%%%%%
for j=1:line_size
        if line(1,j)=='Y'
            L_Y=j;
            break
        end
end

for j=1:line_size                              %for Y
    if line(1,L_Y+j)==space
        L_spaceY=L_Y+j;
        break
    end
    if line(1,L_Y+j)==newline
        L_spaceY=L_Y+j;
        break
    end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Pos_X_start=line(1,L_X+1:L_spaceX-1);
curr_X=str2num(Pos_X_start);

Pos_Y_start=line(1,L_Y+1:L_spaceY-1) ;
curr_Y=str2num(Pos_Y_start);

%% Initializing min\max position
max_X=0;
```

```
max_Y=0;
min_X=230;
min_Y=230;


%% catching next G0 or G1 for position
layerIDX=1;                          %layer index initialised to 1
while endline+1<z                    % exit if end of file
    max_X=max(max_X,curr_X);
    max_Y=max(max_Y,curr_Y);
    min_X=min(min_X,curr_X);
    min_Y=min(min_Y,curr_Y);
    startline=endline+1;

    % To get Gcode line one by one
    for i=1:1000
        if text1(1,startline+i)==newline
            endline=startline+i;
            break
        end
        if startline+i==z
            endline=startline+i;
            break
        end
    end
    line=text1(1,startline:endline)
    line_size=length(line);
    % checking for any positional value in line
    m=contains(line,'X');
    n=contains(line,'Y');
    %%%%%%%%%%%%%%%%%%%%%%%%% for X %%%%%%%%%%%%%%%%%%%%%
    if m==1
        %finding position of X in line
        for j=1:line_size
            if line(1,j)=='X'
                L_X=j;
                break
            end
        end
        %finding last digit of X-position array
        for j=1:line_size
            if line(1,L_X+j)==space
                L_spaceX=L_X+j;
                break
            end
            if line(1,L_X+j)==newline
                L_spaceX=L_X+j;
                break
            end
            if line(1,L_X+j)=='+'
                L_spaceX=L_X+j;
                break
            end
        end
    end


    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


    %%%%%%%%%%%%%%%%% for Y %%%%%%%%%%%%%%%%%%%%%
    if n==1
        %finding position of Y in line
        for j=1:line_size
                if line(1,j)=='Y'
                    L_Y=j;
                    break
                end
        end
        %finding last digit of Y-position array
        for j=1:line_size
            if line(1,L_Y+j)==space
                L_spaceY=L_Y+j;
                break
            end
            if line(1,L_Y+j)==newline
                L_spaceY=L_Y+j;
                break
            end
            if line(1,L_Y+j)=='+'
                L_spaceY=L_Y+j;
                break
            end
        end
    end
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    if m==0
        target_X=curr_X;
    else
```

```
        Pos_X_start=line(1,L_X+1:L_spaceX-1);
        [target_X,tf]=str2num(Pos_X_start);
        if tf==0
            target_X=curr_X;
        end


    end
    if n==0
        target_Y=curr_Y;
    else
        Pos_Y_start=line(1,L_Y+1:L_spaceY-1);
        [target_Y,tf]=str2num(Pos_Y_start);
        if tf==0
            target_Y=curr_Y;
        end

    end

    % Writing to image if print command
    if line(1,1:2)=='G1'
        start_posX=curr_X;
        start_posY=curr_Y;
        final_posX=target_X;
        final_posY=target_Y;
        % Scaling all points
        start_posX_scaled=round(start_posX,roundfact)*...
                                            pointscalingF;
        start_posY_scaled=round(start_posY,roundfact)*...
                                            pointscalingF;
        final_posX_scaled=round(final_posX,roundfact)*...
                                            pointscalingF;
        final_posY_scaled=round(final_posY,roundfact)*...
                                            pointscalingF;
        % Forming lines
        path_x=round(linspace(start_posX_scaled,...
                             final_posX_scaled,interpolation));
        path_y=round(linspace(start_posY_scaled,...
                             final_posY_scaled,interpolation));
        linethickness=1; %belongs to [0,inf] for line thickness
        for i=1:length(path_x)
            imG1(path_x(1,i)-linethickness:path_x(1,i)+...
                    linethickness,path_y(1,i)-linethickness:...
                    path_y(1,i)+linethickness)...
                    =zeros(2*linethickness+1);
        end
    end

    % checking if end of layer register image
    cmp=strcmp(line(1,1:4),strLAYER);

    if cmp==1
        %cropping image to desired size
        imG1_cropped=imcrop(imG1,[corner_pts_scaled(1,1),...
                    corner_pts_scaled(1,2),pointscalingF*...
                    box_size(1,2)-1,pointscalingF*...
                    box_size(1,1)-1]);
        fname=['Layer_from_gcode' num2str(layerIDX) '.jpg'];
        imwrite(imG1_cropped,fname,'jpg');
        layerIDX=layerIDX+1;
        imG1=im;
    end
    % preparing variables for next layer
    curr_X=target_X;
    curr_Y=target_Y;
end
box.size=box_size;
box.pos=box_pos;
save box
msgbox('All layers Done');
```

8 Image Acquisition system

8.1 Camera

Figure 24: Camera

The USB camera shown in figure 24 was used to capture images of the layer. And in order to maintain the focus to the uppermost layer, the camera used has manually adjustable focus and exposure settings.

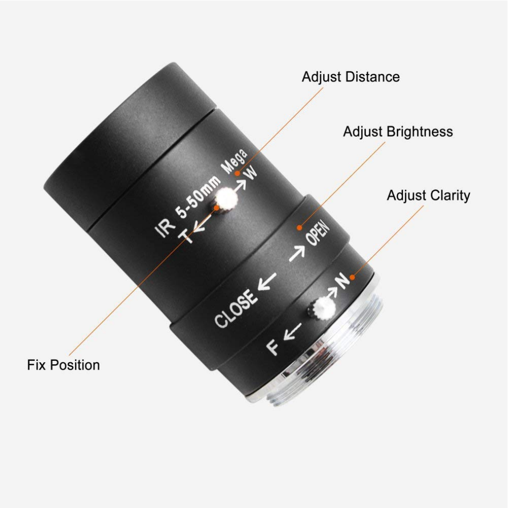


Figure 25: Camera Adjustment

Other Camera Details are given in table

Model number	SV-USBFHD06H-SFV
Item weight	331 g
Lens size	1 / 2.9 inches
Max. Resolution	1080P
Megapixel	2.0
Compression format	H264 / MJPEG / YUV2 (YUYV)
S / N ratio	42 dB
Dynamic range	86 dB
Sensitivity	5.0 V / lux-sec @ 550nm
Type of shot	electronic shutter / exposure frame
Operating temperature	-20 to 85 ℃

Table 2: Camera Description

8.2 Circuit

8.2.1 Objective

The purpose of this electrical circuit is to define the moment at which the camera should click photos and also stop the image acquisition session manually.
Circuit comprises of the following components

1. Arduino Uno.

2. Breadboard.

3. Mini-Breadboard.

4. Two resistors of 10 kohm.

5. A Red LED for indication.

6. Two tactile sensors.

7. Jumper wires.

- **Arduino Uno**

The Arduino board is used to capture the two digital signals, received from the two tactile sensors. Also, an LED is introduced to indicate input received by the tactile sensor which is attached to the printer.

- **Resistors**

The resistors are connected to the ground in such a way, so as to avoid any floating value that the Arduino will receive.

- **Sensors**

One of the sensor‘s function is to manually send the signal to the image acquisition system to stop the cyclic process explained.
The other sensor function is used to send the signal to the system indicate when to click and store the image.

8.2.2 Circuit Diagram

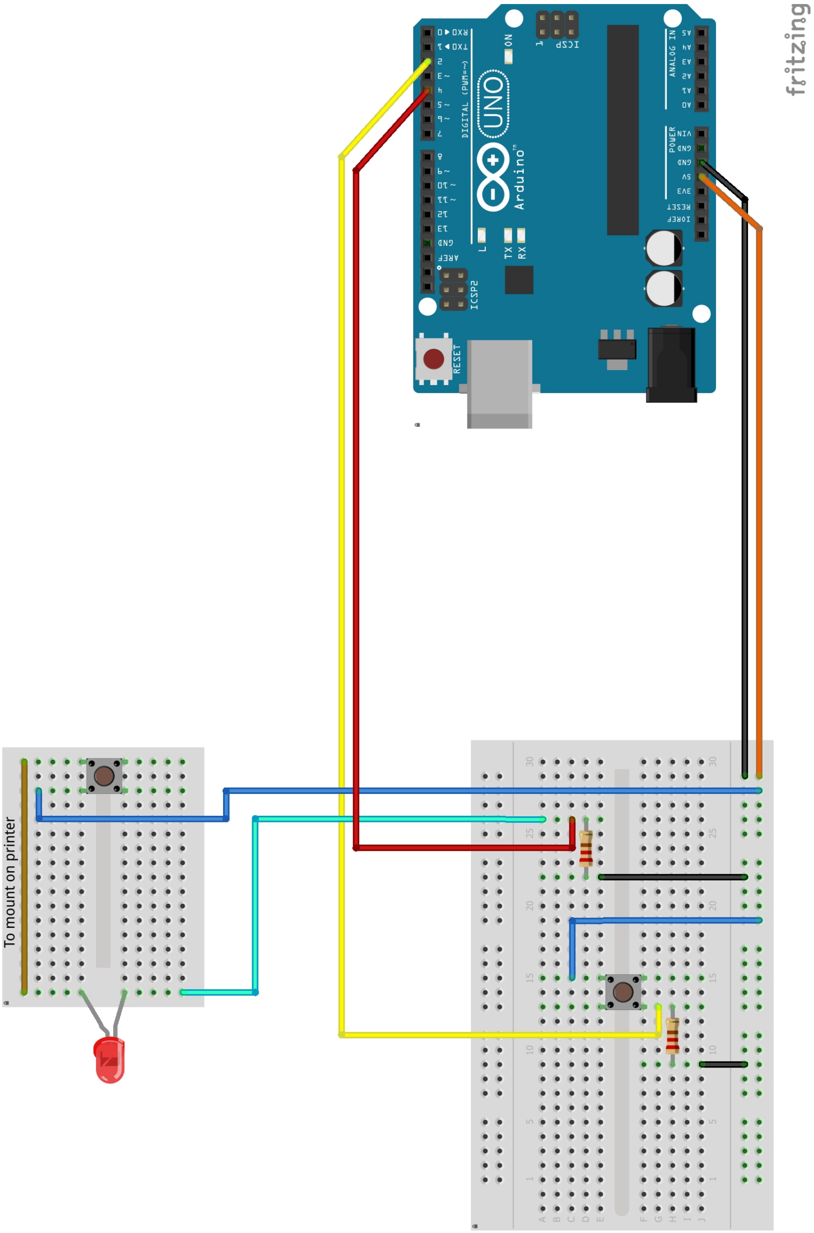


Figure 26: Circuit Diagram

8.3 Sensor mounting

In order to acquire a signal from the printer when the printer head moves to the capture position, and also to accomplish minimum clearance between the moving part, a 3D object switch mounting is first designed in Solidworks and it is printed to first prototype shown as follows.

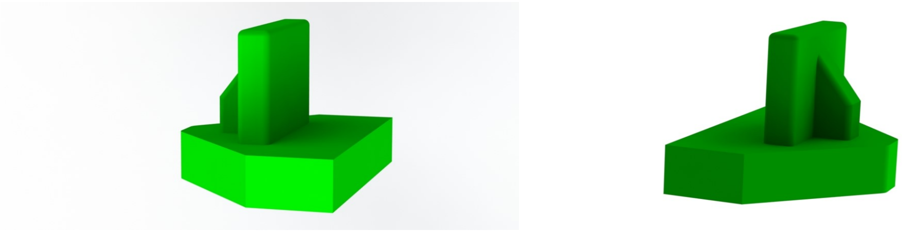


Figure 27: Sensor mounting CAD model

Printer setting used as shown in following table

Layer Height	0.15 mm
Wall Thickness	0.6 mm
Top/Bottom Thickness	0.8 mm
Infill Density	65 %
Printing Temperature	230 ℃
Build Plate Temperature	80 ℃
Print Speed	65 mm/s

Table 3: Printer Setting for Sensor Mounting

8.4 Implementation

This Image Acquisition system is attached to the printer in the following manner as shown in the figure 28.

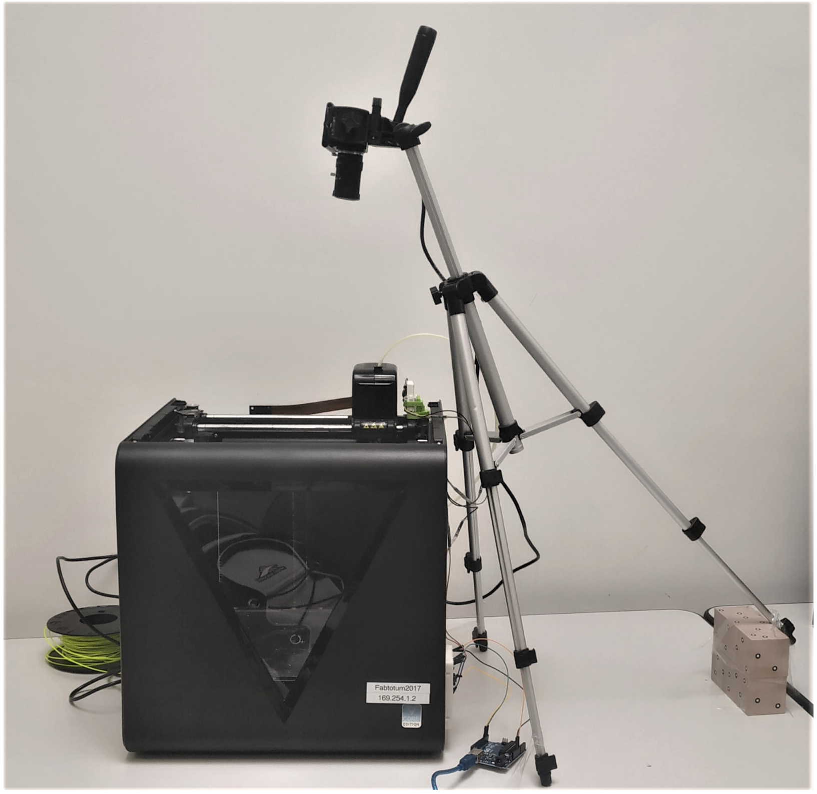


Figure 28: Image Acquisition system

One of the two breadboards is attached to the side of the printer in the following manner as shown in figure 29.
And the other for the sensor is attached to its mounting design and fabricated earlier in section 30.

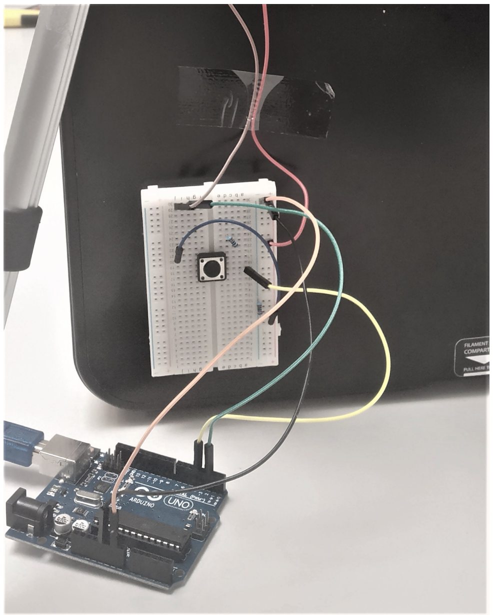


Figure 29: Assembly 1

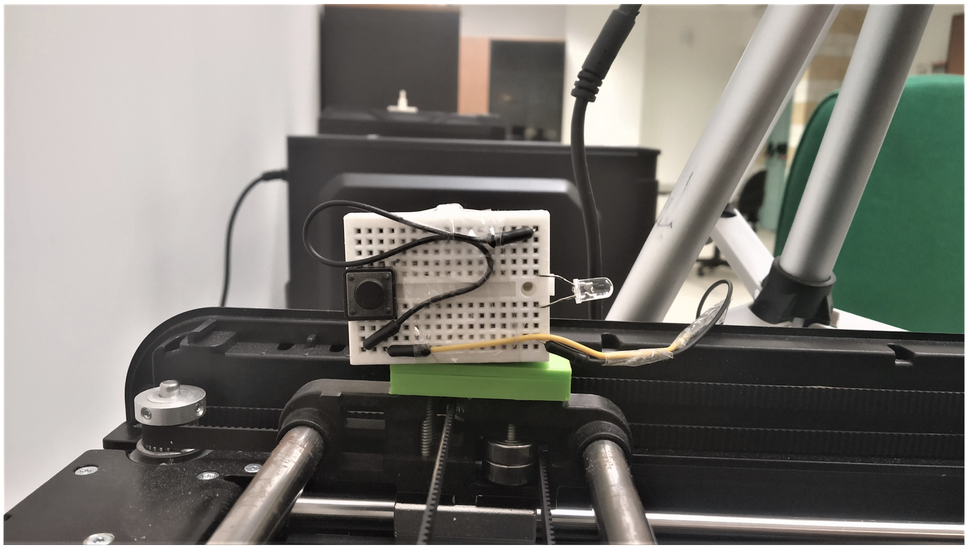


Figure 30: Assembly 2

8.5 Matlab Script to Capture Images

```
%% Clear
imaqreset
clc
clear all
close all
%% Arduino setup
a=arduino();
%% Camera setup and preview
webcamlist
cam=webcam(1);
cam.Resolution= '800x600';
preview(cam);
pause(5);
%% End preview
clear cam;
%% configuring camera for recording
vid = videoinput('winvideo', 2, 'MJPG_1920x1080');
src = getselectedsource(vid);
vid.FramesPerTrigger = 1;
image_ROI=getsnapshot(vid);
figure(2), imshow(image_ROI);
close figure 2;
pause(5);
%% Capturing Images
i=1
while readDigitalPin(a,'D4')==false
    if readDigitalPin(a,'D2')==true
        img=getsnapshot(vid);
        fname=['Layer_' num2str(i) '.jpg'];
        imwrite(img,fname,'jpg');
        i=i+1
        pause(8);           % for 8+1 sec delay
    end
end

%% ending operation
delete(vid);
clear all
close all
```

9 Gcode modifier

9.1 Descrption

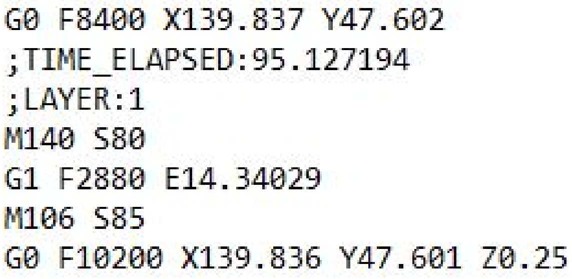
In order to modify Gcode, we require to define bed size, offset, retraction and name of the file to the Matlab script. After that we search for the string “LAYER:” and record its position. This search is performed at each layer as the position for next string keeps on changing at every loop cycle. Then from the position of the string we search the file in reverse direction to find the last position value of X, Y and E and record them. These values are used to compose part of varying Gcode which is needed to be inserted. This varying Gcode is different for every layer. Now the Gcode is ready to be inserted to the original file. After this step, we again start with new search of the string “LAYER:” in the modified file with next layer to be targeted. If all layers has been updated then we stop the loop cycle.

9.2 Varying Gcode

The varying Gcode depends upon the last position of the printer head coordinates before layer change code has arrived. The text shown as “[xx]” is the numerical value changing which depends on layer. The syntax of Gcode is as follows

```
M25;PAUSE_PRINT
G1 E[xx] F3600;RETRACT_X_mm
GO X[xx] Y[xx] F1000.00;FOR_CAM_SWITCH_USE_ABSOLUTE_ONLY
G4 S8;WAIT_X_SEC %enter delay here
G0 X[xx] Y[xx] F1000;GO_TO_RETURN_POSITION
G1 E[xx] F3600;UNRETRACT_X_mm
M82;
M24;resume_SD_print
```

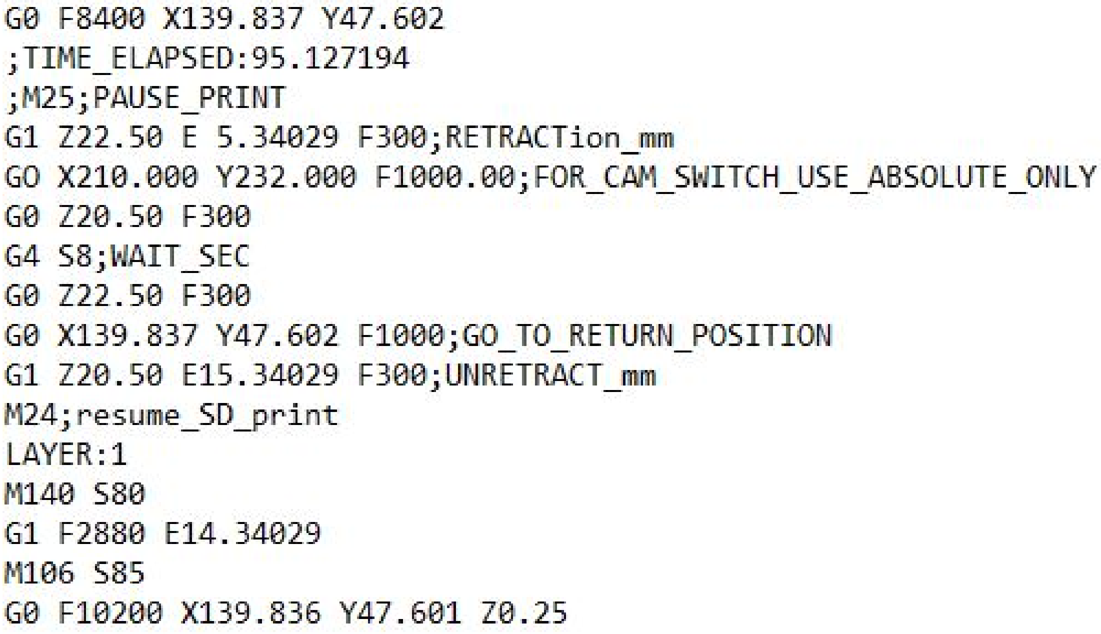
The implementation at each layer can be observed in the following figure 31 which was from original Gcode received from slicer software



```
G0 F8400 X139.837 Y47.602
;TIME_ELAPSED:95.127194
;LAYER:1
M140 S80
G1 F2880 E14.34029
M106 S85
G0 F10200 X139.836 Y47.601 Z0.25
```

Figure 31: Orignal Gcode

And the modified Gcode after processing it with this program is shown in figure 32.



```
G0 F8400 X139.837 Y47.602
;TIME_ELAPSED:95.127194
;M25;PAUSE_PRINT
G1 Z22.50 E 5.34029 F300;RETRACTion_mm
GO X210.000 Y232.000 F1000.00;FOR_CAM_SWITCH_USE_ABSOLUTE_ONLY
G0 Z20.50 F300
G4 S8;WAIT_SEC
G0 Z22.50 F300
G0 X139.837 Y47.602 F1000;GO_TO_RETURN_POSITION
G1 Z20.50 E15.34029 F300;UNRETRACT_mm
M24;resume_SD_print
LAYER:1
M140 S80
G1 F2880 E14.34029
M106 S85
G0 F10200 X139.836 Y47.601 Z0.25
```

Figure 32: Modified Gcode

9.3 Matlab Script

```
clear all
close all
clc
%%
nameoffile='FPF_switch_base.txt';     %name of the file
BED_Size=[216,232];                   %in mm [X,Y]
offset=0;                             %in mm
% for capture position
Xlimit=BED_Size(1)-offset;            %in abs
Ylimit=BED_Size(2)-offset;            %in abs
text1=fileread(nameoffile);
strLAYER='LAYER:';
r=strfind(text1,strLAYER);            %string to find
line_end=text1(r(1)+7);
z=length(r);
space=' ';
%% Searching for Layer
for i=1:z-1                           %neglecting layer:0
    strLAYER_Num=compose(strLAYER+"%d"+line_end,i);
    x1=strfind(text1,strLAYER_Num);   %string to find
    % for storing positional value
    for j=1:1000
        if text1(1,x1(1)-j)=='X'
            L_X=x1(1)-j;
            break
        end
    end

    for j=1:1000                      %for X
        if text1(1,L_X+j)==space
            L_spaceX=L_X+j;
            break
        end
        if text1(1,L_X+j)==line_end
            L_spaceX=L_X+j;
            break
        end
    end

    for j=1:1000
        if text1(1,x1(1)-j)=='Y'
            L_Y=x1(1)-j;
            break
        end

    end

    for j=1:1000                      %for Y
        if text1(1,L_Y+j)==space
            L_spaceY=L_Y+j;
            break
        end
        if text1(1,L_Y+j)==line_end
            L_spaceY=L_Y+j;
            break
        end
    end
    t=1
    for j=1:1000
        if text1(1,x1(1)-j)=='E'
            if t==4
                L_E=x1(1)-j;
            break
            end
            t=t+1;
        end

    end

    for j=1:1000                      %for E
        if text1(1,L_E+j)==space
            L_spaceE=L_Y+j;
            break
        end
        if text1(1,L_E+j)==line_end
            L_spaceE=L_E+j;
            break
        end
    end
    % converting string to number
    Pos_X=text1(1,L_X+1:L_spaceX-1);  %e.g '113.71' for X113.71
    curr_X=str2num(Pos_X);
    move_X=Xlimit;
    Pos_Y=text1(1,L_Y+1:L_spaceY-1) ;
    curr_Y=str2num(Pos_Y);
    move_Y=Ylimit;
```

```
    Pos_E=text1(1,L_E+1:L_spaceE-1);  %e.g '113.71' for X113.71
    curr_E=str2num(Pos_E);
    retract=curr_E-2;                 %setting retraction to 2mm
    unretract=curr_E;

    % composing text to be added
    Section_Text1="M25;PAUSE_PRINT\nG1 E";
    Section_Text2=" F3600;RETRACT_X_mm\nGO X";
    Section_Text3=" Y";
    Section_Text4=" F1000.00;FOR_CAM_SWITCH_USE_ABSOLUTE_ONLY...
                    \nG4 S8;WAIT_X_SEC\nG0 X";  %enter delay here
    Section_Text6=" Y";
    Section_Text7=" F1000;GO_TO__RETURN_POSITION\nG1 E";
    Section_Text8=" F3600;UNRETRACT_X_mm\nM82;...
                                    \nM24;resume_SD_print\n";
    % Merging all sections
    Insert_Text = compose(Section_Text1+"%7.5f"...
                         +Section_Text3+"%3.3f"...
                         +Section_Text4+"%3.3f"...
                         +Section_Text6+"%3.3f"...
                         +Section_Text7+"%7.5f"...
                         +Section_Text8,retract,...
                         move_X,move_Y,curr_X,...
                         curr_Y,unretract);
    Add_Before=compose("LAYER:%d",i)
    new_text1 = insertBefore(text1,x1,Insert_Text);
    fid = fopen(nameoffile,'wt');
    fprintf(fid, new_text1');
    fclose(fid);
    text1=fileread(nameoffile);            %name of the file
    r=strfind(text1,strLAYER);

end
msgbox('All layers Done');
```

10 Image processing

10.1 Description

The concept is to compare the two images of each layer, one acquired from the image acquisition system and the other generated from the Gcode. In order to make these images compare ready certain modification is needed to be done before actual comparison which is explained in the following section.

10.2 Image Correction

10.2.1 Calibration setup

The image acquired from the image acquisition system is raw containing distortion from the camera. Also to imply world coordinates to the image, calibration is needed to be done.

This is done by a checkerboard pattern which is created and then printed on an A4 sheet of paper as shown in figure 33. The dimension is set 1cm x 1cm also the checkerboard is kept at a distance of 2 cm from the corner of the paper.

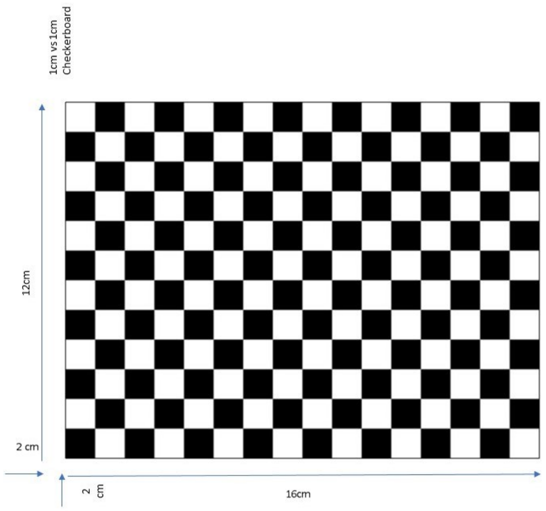


Figure 33: Checkerboard page

Before proceeding to the next step a model is printed on the printer to know the coordinates of the calibration sheet. The model is shown in figure 34.

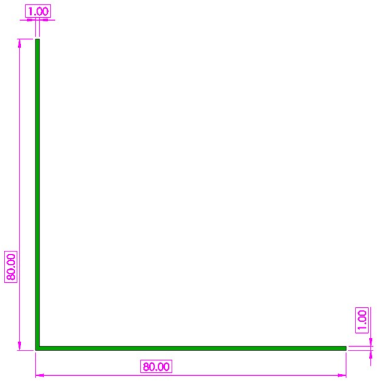
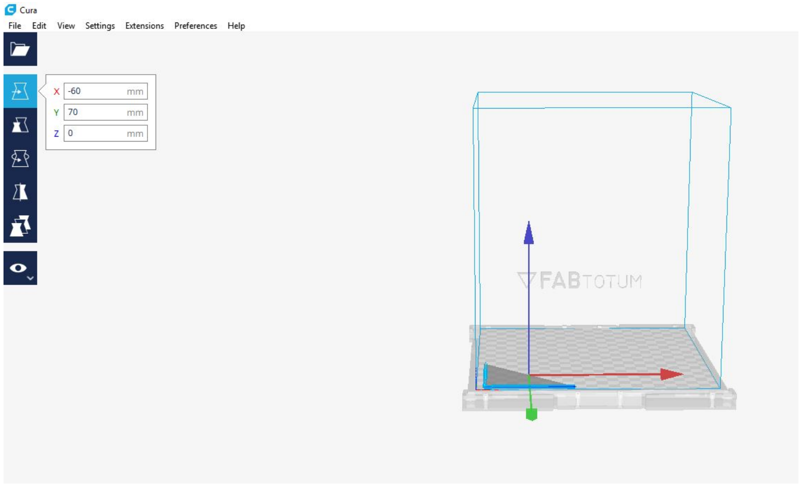


Figure 34: Caliberation Model

In order to set a predefined position of the calibration model. The position set in the slicer software is shown in figure 35.



X -60 mm
Y 70 mm
Z 0 mm

Figure 35: Position in Cura software

The printer setting used for this print is as follows

Layer Height	0.15 mm
Wall Thickness	0.6 mm
Top/Bottom Thickness	0.8 mm
Infill Density	65 %
Printing Temperature	230 ℃
Build Plate Temperature	80 ℃
Print Speed	65 mm/s

Table 4: Printer Setting for Calibration Model

Now without removing this print from the bed, checkerboard paper is then pasted on the bed which will be used to calibrate the images for distortion correction. This paper is also used to capture world coordinates. The implementation is shown in the following figure 36.

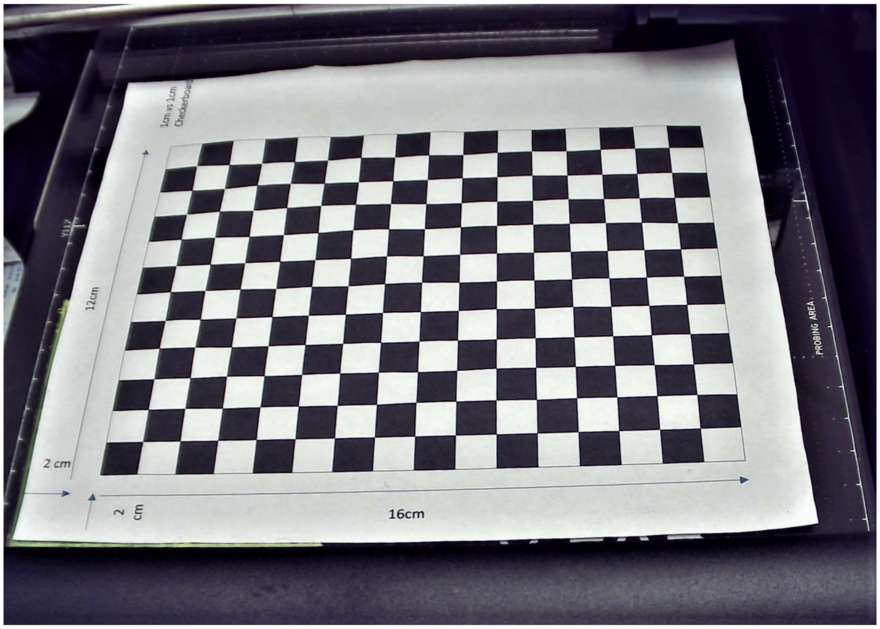


Figure 36: Image from Camera

A set of images (around 20 images) of this scenario is captured with different orientation, position and with different zoom from the camera. These images are then uploaded to Camera Calibrator Application mentioned earlier in section 6.5. From this application we compute and export camera parameters, which will be used to undistort the images. The effect of the above process can be observed in figure 37.

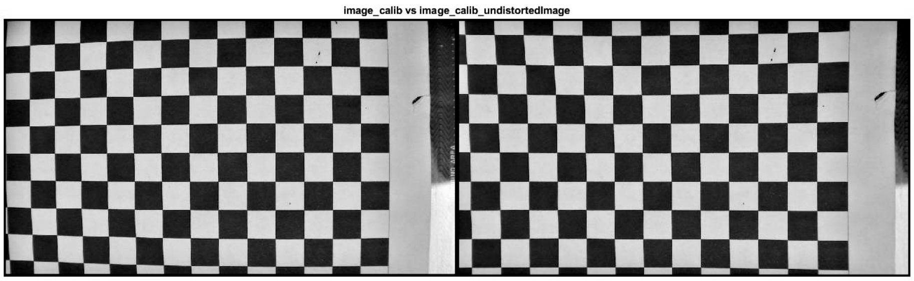


Figure 37: Distortion Correction

The paper can now be removed along with the printed model. And now the printer is ready to print any print job keeping in mind that the image acquisition system remains unchanged and active throughout the print job.

10.2.2 Layer Image setup

On completion of the print job and acquiring the layer images from it. The process of undistorting all the layer images should be followed. After which the next step is to set up the layer images according to our desired region of interest and cropping the undesired part. This is done by exploiting the in-built Matlab function “impixelregion” to acquire the pixel coordinates. This is first applied to the computer image containing the checkerboard as shown in the figure 38.

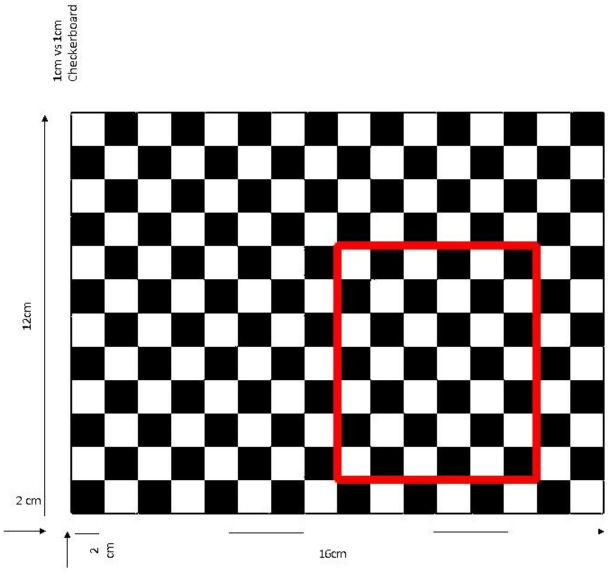


Figure 38: Selecting Region Of Interest (ROI)

In order to extract the image shown in figure 39. At this step, the size of the image is also recorded in order to maintain all the image size.

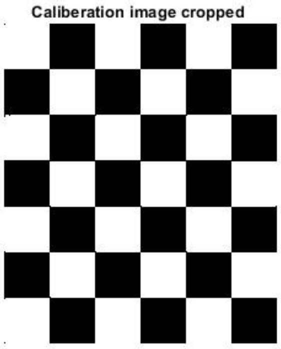


Figure 39: Region Of Interest (ROI)

And then the same relevant position coordinates are extracted from the image containing the checkerboard page pasted to the printer bed. These two images are shown in figure 40.

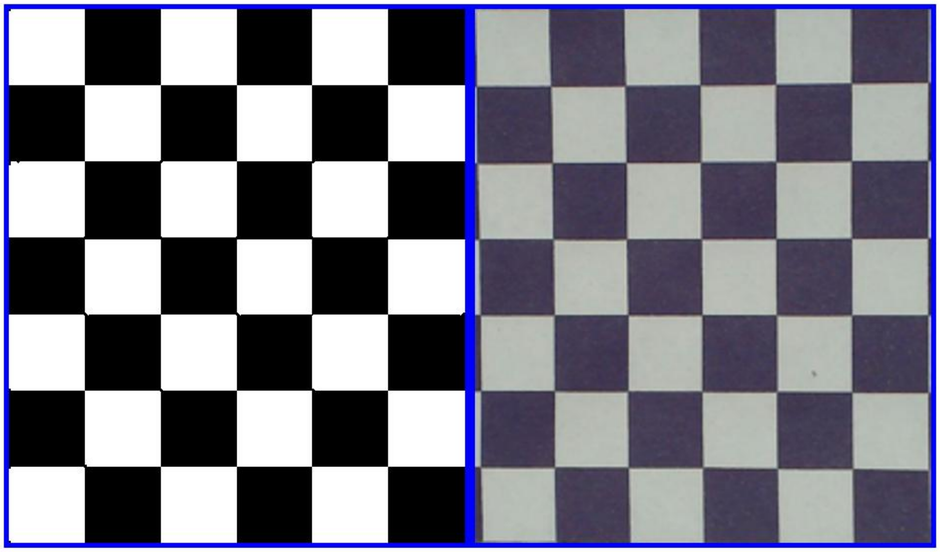


Figure 40: Comparing Computer and print bed images

Since the camera remains stationary all these points correspond to the same position of the bed for all layer images as well, therefore all layer images are also cropped and adjusted accordingly. Then these images are ready to extract the transformation vector to make these to images related to each other. For this, we extract again all the coordinates of the same relevant position in both the images in, particularly the box corners. After applying the transformation vector on calibration image and computer image we obtain the related image shown in figure 41. This vector is applied also on to each layer images to make it compare ready.

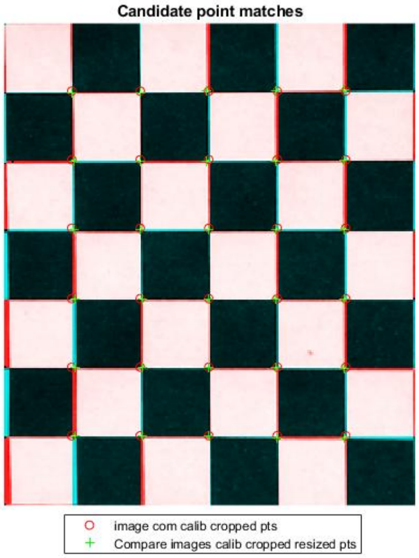


Figure 41: Calibration Image points

Applying this transformation to layer image results in the following figure 42.

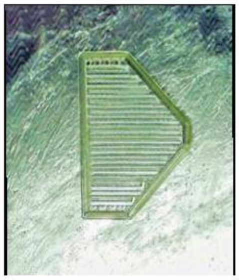


Figure 42: Transformed Layer Image

10.2.3 Reference Image setup

The reference images before comparison are also needed to be adjusted accordingly to the ready to compare layer images which will be done in the following sections. The images generated from the section 7 are imported into the workspace. A sample image of Layer 1 is shown in figure 43.

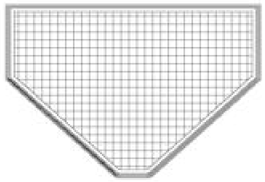


Figure 43: Reference Image

Adjusting and transformation

The images received from the Gcode is scaled-down and then relevant points on images which correspond to the same points as displayed in figure 41 are selected and shown in figure 44.

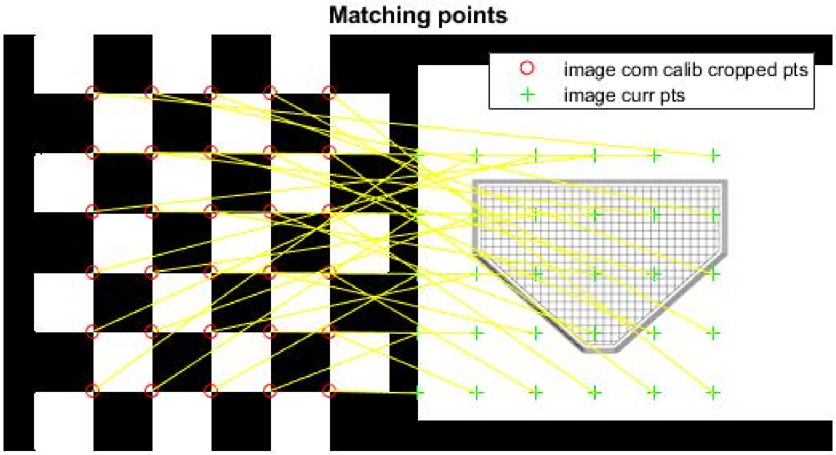


Figure 44: Reference Image points

After which the image obtained is shown in figure 45.

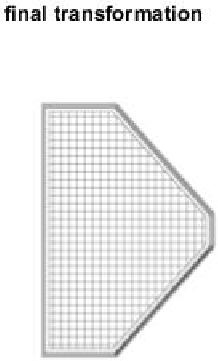


Figure 45: Transformed Reference Image

10.3 Processing Images

10.3.1 Correcting non-uniform illumination of Layer images

Due to Non-uniform lightening conditions, as shown in figure 42, there is a need for correction of the images. This is done by first applying local filters on different region of the image based on their intensity and forming a Background image as shown in figure 46.

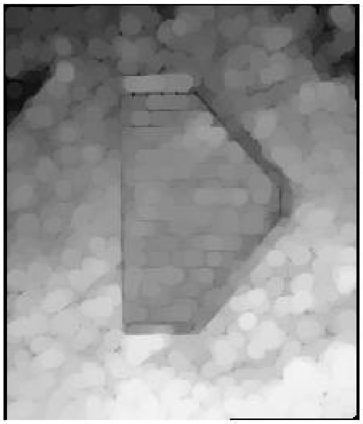
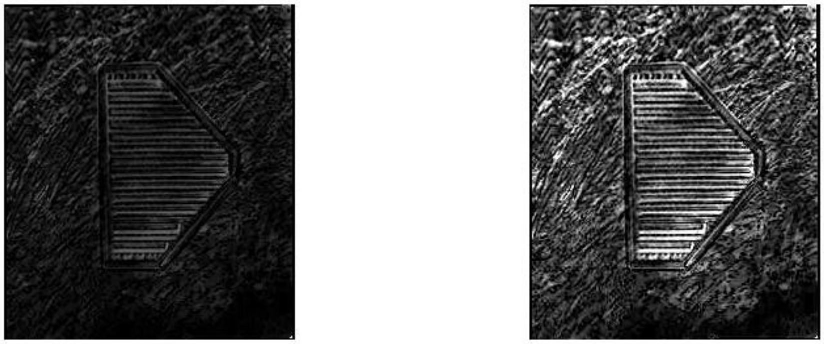


Figure 46: Background image

This background image is then subtracted which results in the image as shown in figure 47a. This procedure also makes the image of low contrast, therefore, a contrast adjustment is needed to be done as shown in figure 47b.



(a) Uniform Illumination

(b) Contrast Correction

Figure 47: Illumination Correction

10.3.2 Filter

To focus the error detection algorithm to the relevant region of the image, A filter is designed in such a way that it pixels all the undesired region of the image to black color. Since the printing of parts always performed with varying shapes and sizes, therefore it is necessary for the filter to be a dynamic filter which adapts accordingly. Where the edge of filter is set to be 4 pixels wider than the desired area. For example, the filter for figure 45 is shown in figure 48.

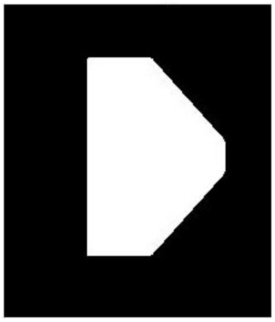
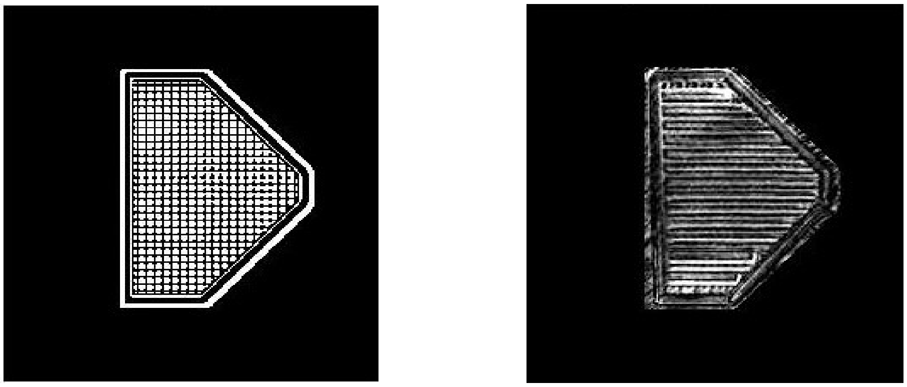


Figure 48: Filter

The effect of the filter can be noticed in both the reference image and layer image as shown in figure 49a and in figure 49b respectively.



(a) Reference Image

(b) Layer Image

Figure 49: Effect of Filter

10.3.3 Reducing Haze

This section is to reduce the amount of haziness in the images. This is done by first inverting the image before haze removal and then inverting the image back again as shown in the following figures.

(a) Inverted Image

(b) Image without haze

Figure 50: Reducing Haze

10.4 Matlab Script

```
%%
close all
clear all
clc

%% Computer Calliberation Image
image_com_calib_dir='D:\mechatronics\Thesis...
                        \image_analysis\New_folder3\base\';
image_com_calib = dir([image_com_calib_dir '*.jpg']);
file_name_com_calib=image_com_calib.name;
image_com_calib=imread([ image_com_calib_dir...
                file_name_com_calib]); %importing to workspace
image_com_calib=imrotate(image_com_calib,90);       %adjusting
image_com_calib=rgb2gray(image_com_calib);
image_com_calib=imbinarize(image_com_calib);
figure, imshow(image_com_calib);
hold on
title(file_name_com_calib);

%% Correcting Distortion
load('cameraParams.mat');            %loading camera parameters
mkdir('undistorted_images');
image_layer_dir='D:\mechatronics\Thesis...
                            \image_analysis\New_folder3\layer\';
image_layer = dir([image_layer_dir '*.jpg']);
[i,~]=size(image_layer);

for k=1:i
    s1="Layer_";
    s2=".jpg";
    image_layer_name=char(compose(s1+"%d"+s2,k));
    image_curr=imread([ image_layer_dir image_layer_name]);
    [image_layer_undistortedImage,newOrigin] = ...
                        undistortImage(image_curr, cameraParams);
     figure,montage({image_curr,image_layer_undistortedImage}...
                        ,'BackgroundColor','blue','BorderSize',5)
     title('image\_calib vs image\_calib\_undistortedImage')
    fname=['Undistorted_layer' num2str(k) '.jpg'];
    FILENAME = ['D:\mechatronics\Thesis\image_analysis...
                                    \undistorted_images\', fname];
    imwrite(image_layer_undistortedImage,FILENAME,'jpg');
```

```
end


%% Loading caliberation image of bed and correcting distortion
image_calib_dir='D:\mechatronics\Thesis\image_analysis...
                                     \New_folder3\calib_images\';
image_calib = dir([image_calib_dir '*.jpg']);
[i,~]=size(image_calib);
file_name_calib=image_calib(i).name;
image_calib=imread([ image_calib_dir file_name_calib]);
[image_calib_undistortedImage,~] = ...
                        undistortImage(image_calib, cameraParams);
figure,montage({image_calib,image_calib_undistortedImage},...
                        'BackgroundColor','blue','BorderSize',5)
title('image\_calib vs image\_calib\_undistortedImage')
fname=['Undistorted_calib' num2str(i) '.jpg'];
FILENAME = ['D:\mechatronics\Thesis\image_analysis...
                                    \undistorted_images\', fname];
imwrite(image_calib_undistortedImage,FILENAME,'jpg');
figure
c=imfuse(image_layer_undistortedImage,image_calib_undistortedImage);
imshow(c);

load('box.mat');    %loading box paramters from image from Gcode
%% Selecting Region of Interest
point1=[374,678];    % need to change for every new print setup
point2=[595,678];
point3=[595,418];
point4=[374,418];
figure(1)
hold on
plot(point1(1):point2(1),point1(2):point2(2),'*r');
plot(point3(1):point2(1),point3(2):point2(2),'*r');
plot(point4(1):point3(1),point4(2):point3(2),'*r');
plot(point4(1):point1(1),point4(2):point1(2),'*r');
hold off
image_com_calib_cropped=imcrop(image_com_calib,...
                [point4(2)-7,point4(1)+8,point3(1)-point4(1),...
                                        point1(2)-point4(2)-1]);
figure,imshow(image_com_calib_cropped)
fname='cropped_com_calib.jpg';
FILENAME = ['D:\mechatronics\Thesis\image_analysis...
                    \New_folder3\reference_images\com\', fname];
imwrite(image_com_calib_cropped,FILENAME,'jpg');
```

```
[im_com_width,im_com_height]=size(image_com_calib_cropped);

%% Cropping the Calib wrt computer image

Undistorted_images_dir='D:\mechatronics\Thesis...
                            \image_analysis\undistorted_images\';
Undistorted_images = dir([Undistorted_images_dir '*.jpg']);
[i,~]=size(Undistorted_images);
s1="Undistorted_calib8";
s2=".jpg";
Compare_images_name=char(compose(s1+s2));
Compare_images_calib=imread([ Undistorted_images_dir...
                                            Compare_images_name]);
figure,imshow(Compare_images_calib)
%impixelregion
hold on
point1=[657,1051];      % need to change for every new print setup
point2=[1407,1051];
point3=[1407,191];
point4=[657,191];
plot(point1(1):point2(1),point1(2):point2(2),'*r');
plot(point3(1):point2(1),point3(2):point2(2),'*r');
plot(point4(1):point3(1),point4(2):point3(2),'*r');
plot(point4(1):point1(1),point4(2):point1(2),'*r');
hold off
Compare_images_calib_cropped=imcrop(Compare_images_calib,...
                    [point4(1),point4(2),point3(1)-point4(1),...
                                            point1(2)-point4(2)]);
figure,imshow(Compare_images_calib_cropped)

[im_width,im_height]=size(Compare_images_calib_cropped);
Compare_images_calib_cropped_resized =...
                        imresize(Compare_images_calib_cropped,...
                                [im_com_width im_com_height]);...
                                %resizing according to computer image

figure
cz=imfuse(image_com_calib_cropped,...
                            Compare_images_calib_cropped_resized);
imshow(cz);
figure, montage({image_com_calib_cropped,...
                        Compare_images_calib_cropped_resized},...
                            'BackgroundColor','blue','BorderSize',5)

% figure
% imshow(image_com_calib_cropped)
% impixelregion                                      %for points
% figure
% imshow(Compare_images_calib_cropped_resized)
% impixelregion

%% Secting points for caliberation
% For computer image
image_com_calib_cropped_pts...
                        =[37,37;74,37;111,37;148,37;185,37;...
                        37,74;74,74;111,74;148,74;185,74;...
                        37,111;74,111;111,111;148,111;185,111;...
                        37,149;74,149;111,149;148,149;185,149;...
                        37,186;74,186;111,186;148,186;185,186;...
                        37,223;74,223;111,223;148,223;185,223;];
% For Calib  image
Compare_images_calib_cropped_resized_pts=...
                        [37,38;74,38;110,37;147,37;184,36;...
                        38,75;74,75;110,75;147,75;184,74;...
                        38,112;74,112;111,112;147,112;184,112;...
                        39,149;75,149;111,149;147,149;184,149;...
                        39,186;75,186;111,186;147,186;184,187;...
                        39,223;75,223;111,223;148,224;184,224;];


% Matching points
figure; ax = axes;
showMatchedFeatures(image_com_calib_cropped,...
                        Compare_images_calib_cropped_resized,...
                        image_com_calib_cropped_pts,...
                        Compare_images_calib_cropped_resized_pts...
                        ,'Parent',ax);
title(ax, 'Candidate point matches');
legend(ax, 'image com calib cropped pts',...
                        'Compare images calib cropped resized pts');

% Estimating Transformation pararmeters
tform = estimateGeometricTransform(image_com_calib_cropped_pts,...
                Compare_images_calib_cropped_resized_pts,'affine');

% Displaying result
outputView = imref2d(size(image_com_calib_cropped));
Compare_ready_image_calib =...
            imwarp(Compare_images_calib_cropped_resized,tform,...
                                        'OutputView',outputView);
```

```
figure; imshow(Compare_ready_image_calib);
title('final transformation');

Compare_ready_image_calib1=rgb2gray(Compare_ready_image_calib);
Compare_ready_image_calib2=imbinarize(Compare_ready_image_calib1);
figure; imshow(Compare_ready_image_calib2);

C = imfuse(Compare_ready_image_calib2,image_com_calib_cropped,...
                                                    'falsecolor');
figure,imshow(C)

%Registering Image
fname='Compare_images_calib.jpg';
FILENAME = ['D:\mechatronics\Thesis\image_analysis\New_folder3...
                        \reference_images\Compare_images\', fname];
imwrite(Compare_ready_image_calib,FILENAME,'jpg');

%% Applying the same Transformation parameter to all layer images
a=i-1;                                          %for 1 calib image
s1="Undistorted_layer";
s2=".jpg";
for k=1:a
    Compare_images_name=char(compose(s1+'%d'+s2,k));
    Compare_images_layer=imread([ Undistorted_images_dir...
                                            Compare_images_name]);
    Compare_images_layer_cropped=imcrop(Compare_images_layer...
                                        ,[point4(1),point4(2),...
                                            point3(1)-point4(1),...
                                            point1(2)-point4(2)]);
%     figure,imshow(Compare_images_layer_cropped)
    Compare_images_layer_cropped_resized =...
                        imresize(Compare_images_layer_cropped,...
                                    [im_com_width im_com_height]);
%    figure; ax = axes;
%     showMatchedFeatures(image_com_calib_cropped,...
                        Compare_images_layer_cropped_resized,...
                        image_com_calib_cropped_pts,...
                        Compare_images_calib_cropped_resized_pts,...
                        'Parent',ax);
%     title(ax, 'Candidate point matches');
%     legend(ax, 'image com calib cropped pts',...
                        'Compare images layer cropped resized pts');
    Compare_ready_image_layer = ...
                    imwarp(Compare_images_layer_cropped_resized,...
                                    tform,'OutputView',outputView);
```

```
%Registering Image
     fname=['Compare_layer' num2str(k) '.jpg'];
    FILENAME = ['D:\mechatronics\Thesis\image_analysis...
            \New_folder3\reference_images\Compare_images\', fname];
    imwrite(Compare_ready_image_layer,FILENAME,'jpg');

end


%% Selecting the same relavent points on Image from Gcode
image_ref_dir='D:\mechatronics\Thesis\image_analysis\New_folder3...
                                \reference_images\image_from_Gcode\';
image_ref = dir([image_ref_dir '*.jpg']);
[ii,~]=size(image_ref);
n_box=[6,7];
%close all
for kk=1:ii
    s1="Layer_from_gcode";
    s2=".jpg";
    image_ref_name=char(compose(s1+"%d"+s2,kk));
    image_curr=imread([ image_ref_dir image_ref_name]);
    image_curr=imresize(image_curr,0.072);       %scale
    figure,imshow(image_curr)
     image_curr_pts=[98,118;98,128;98,138;98,148;98,158;...
                     88,118;88,128;88,138;88,148;88,158;...
                     78,118;78,128;78,138;78,148;78,158;...
                     68,118;68,128;68,138;68,148;68,158;...
                     58,118;58,128;58,138;58,148;58,158;...
                     48,118;48,128;48,138;48,148;48,158];
     image_curr_scaled_pts=(image_curr_pts-[38,108]+[-10.5,8])*3.6;

     figure,imshow(image_curr);
     figure; ax = axes;
     showMatchedFeatures(image_com_calib_cropped,image_curr,...
                image_com_calib_cropped_pts,image_curr_scaled_pts,...
                                            'montage','Parent',ax);
     title(ax, 'Matching points');
     legend(ax, 'image com calib cropped pts','image curr pts');

     tform1 = estimateGeometricTransform(image_com_calib_cropped_pts,...
                                image_curr_scaled_pts,'similarity');

     Compare_image_curr = imwarp(image_curr,...
                                tform1,'OutputView',outputView);
     Compare_image_curr=imrotate(Compare_image_curr,180);
     Compare_image_curr(1:260,1:15)=...
                        255*ones(260,15);%to maintain image edges
     Compare_image_curr(223:260,1:222)=255*ones(38,222);
     Compare_image_curr(1:260,201:222)=255*ones(260,22);
     figure; imshow(Compare_image_curr);
     title('final transformation');
     iii=size(Compare_image_curr);
      for aa=1:iii(1)
          for bb=1:iii(2)

              if Compare_image_curr(aa,bb)<=226
                  Compare_image_curr(aa,bb)=0;
              else
                  Compare_image_curr(aa,bb)=255;
              end
          end
      end

      figure; imshow(Compare_image_curr);

% Registering Image
    fname=['Reference_layer' num2str(kk) '.jpg'];
    FILENAME = ['D:\mechatronics\Thesis\image_analysis...
                        \New_folder3\reference_images...
                        \final_reference_image\', fname];
    imwrite(Compare_image_curr,FILENAME,'jpg');

end

%% Comparing Refence and layer images
m=a;
image_number=1; % Select target layer

close all
mkdir('Compared_images');
Compare_images_dir='D:\mechatronics\Thesis...
            \image_analysis\New_folder3\reference_images...
                                        \Compare_images\';
Compare_images = dir([Compare_images_dir '*.jpg']);

[ii,~]=size(Compare_images);
ii=ii-1;
l1="Compare_layer";
```

```
l2=".jpg";

Final_ref_images_dir='D:\mechatronics\Thesis...
            \image_analysis\New_folder3\reference_images...
                                \final_reference_image\';
Final_ref_images = dir([Final_ref_images_dir '*.jpg']);

[a,~]=size(Final_ref_images);
r1="Reference_layer";
r2=".jpg";

%loading images
Compare_ref_images_name=...
                    char(compose(r1+"%d"+r2,image_number));
Compare_ref_images=imread([ Final_ref_images_dir...
                                Compare_ref_images_name]);

Compare_images_name=char(compose(l1+"%d"+l2,...
                                            image_number));
Compare_layer=imread([ Compare_images_dir...
                                    Compare_images_name]);
figure,imshow(Compare_layer);
Compare_layer=imrotate(Compare_layer,-1); % rotate
Compare_layer=Compare_layer(3:262,3:224,:); %adjusting size
Compare_layer_final=Compare_layer;

Compare_layer=rgb2gray(Compare_layer);
figure,imshow(Compare_layer);

%% For Background Correction
se = strel('disk',5);
background = imopen(Compare_layer,se);
figure,imshow(background)
I2 = Compare_layer - background;
figure,imshow(I2)

 Compare_layer = imadjust(I2);
figure,imshow(Compare_layer)

%%%%%%%%%%%%%%%%%%%% Filter Design %%%%%%%%%%%%%%%%%%

filter1=zeros(iii);
gap=3;

% for left side
```

```
for aa=1:iii(1)
    for bb=1:iii(2)-gap
        if Compare_ref_images(aa,bb+gap)>=150
            filter1(aa,bb)=1;
        else
            break
        end
    end
end
%for right side
for aa=1:iii(1)
    for bb=1:iii(2)-gap

        if Compare_ref_images(aa,iii(2)+1-bb-gap)>=150
            filter1(aa,iii(2)+1-bb)=1;
        else
            break
        end
    end
end

% for top side
for bb=1:iii(2)
    for aa=1:iii(1)-gap
        if filter1(aa+gap,bb)==1 && Compare_ref_images...
                                         (aa+gap,bb)>=150
            filter1(aa,bb)=1;
        elseif filter1(aa+gap,bb)==0 && ...
                        Compare_ref_images(aa+gap,bb)>=150
            for mm=1:gap+1
                if filter1(aa+mm,bb)==1
                    filter1(aa+mm-1,bb)=0;
                else
                    break
                end
            end
            break
        elseif Compare_ref_images(aa+gap,bb)<=150
            for mm=1:gap+1
                if Compare_ref_images(aa+mm,bb)>=150
                    filter1(aa+mm-1,bb)=0;
                else
                    break
                end
            end
            break
        end
    end
end
%for bottom side
for bb=1:iii(2)
    for aa=1:iii(1)-gap
        if filter1(iii(1)+1-aa-gap,bb)==1 &&...
                    Compare_ref_images(iii(1)+1-aa-gap,bb)>=150
            filter1(iii(1)+1-aa,bb)=1;
        elseif filter1(iii(1)+1-aa-gap,bb)==0 &&...
                    Compare_ref_images(iii(1)+1-aa-gap,bb)>=150
            for mm=1:gap+1
                if filter1(iii(1)+2-aa-mm,bb)==1
                    filter1(iii(1)+2-aa-mm,bb)=0;
                else
                    break
                end
            end
            break
        elseif Compare_ref_images(iii(1)+1-aa-gap,bb)<=150
            for mm=1:gap+1
                if Compare_ref_images(iii(1)+2-aa-mm,bb)>=150
                    filter1(iii(1)+2-aa-mm,bb)=0;
                else
                    break
                end
            end
            break
        end
    end
end
filter1=~filter1;
figure,imshow(filter1),title('Filter for background')
filter1=uint8(filter1);
%%%%%%%%%%%%%%%%%%%% Applying Filter %%%%%%%%%%%%%%%%%%%%%%%%%

Compare_ref_images1=Compare_ref_images.*filter1;
figure,imshow(Compare_ref_images1),...
                            title('Reference image after filter')

Compare_layer=Compare_layer.*filter1;
figure,imshow(Compare_layer),title('Layer image after filter');

% Adjusting contrast and removing haze
```

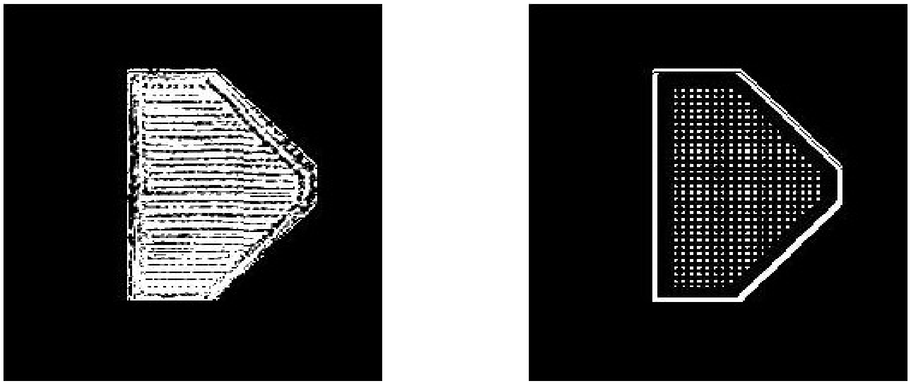
```
Compare_layerInv = imcomplement(Compare_layer);
figure, imshow(Compare_layerInv),title('inverted');
Compare_layer = imreducehaze(Compare_layerInv);
figure, imshow(Compare_layer),title('after haze reduction');

    iii=size(Compare_layer);
for aa=1:iii(1)
    for bb=1:iii(2)
        if Compare_layer(aa,bb)<=124
            Compare_layer(aa,bb)=0;
        elseif Compare_layer(aa,bb)>124 &&...
                                        Compare_layer(aa,bb)<180
            Compare_layer(aa,bb)=140;
        else
            Compare_layer(aa,bb)=255;
        end
    end
end
figure,imshow(Compare_layer),title('Compare layer');
figure,montage({Compare_layer,Compare_ref_images},...
                        'BackgroundColor','blue','BorderSize',5)
C = imfuse(Compare_layer,Compare_ref_images,'falsecolor');
figure,imshow(C)
%% Computing Results
close all
Compare_layer=255-Compare_layer;
figure,imshow(Compare_layer)%,title('Compare Layer');
Compare_ref_images=Compare_ref_images1;
figure,imshow(Compare_ref_images)%,title('Comapare Ref');
result1=zeros(iii); % desired print
for aa=1:iii(1)
    for bb=1:iii(2)
        if Compare_ref_images(aa,bb)<=25 &&...
                                    Compare_layer(aa,bb)>=200
            result1(aa,bb)=Compare_layer(aa,bb);
        else
            result1(aa,bb)=0;
        end
    end
end

figure,imshow(result1),title('desired print');

result2=zeros(iii);  %unwanted print
for aa=1:iii(1)
```

```
    for bb=1:iii(2)
        if Compare_ref_images(aa,bb)>=254 &&...
                                    Compare_layer(aa,bb)>=254
            result2(aa,bb)=Compare_layer(aa,bb);
        else
            result2(aa,bb)=0;
        end
    end
end
figure,imshow(result2),title('unwanted print');

result3=zeros(iii);  %missing print
for aa=1:iii(1)
    for bb=1:iii(2)
        if Compare_ref_images(aa,bb)<=25 &&...
                                    Compare_layer(aa,bb)<=25
            result3(aa,bb)=Compare_layer(aa,bb);
        else
            result3(aa,bb)=255;
        end
    end
end
result3=255-result3;
result3=uint8(result3);
result3=result3.*filter1;
figure,imshow(result3),title('Missing print');
result2=uint8(result2);
result1=uint8(result1);

re(:,:,1)=result3;
re(:,:,2)=result2;
re(:,:,3)=result1;
figure,imshow(re),title('result')
```

11 Results

To extract information from the processed images shown in figure 51a and figure 51b, three different kinds of operators are used. They give the results in the following three ways

- Good Print
- Unwanted Print
- Missing Print



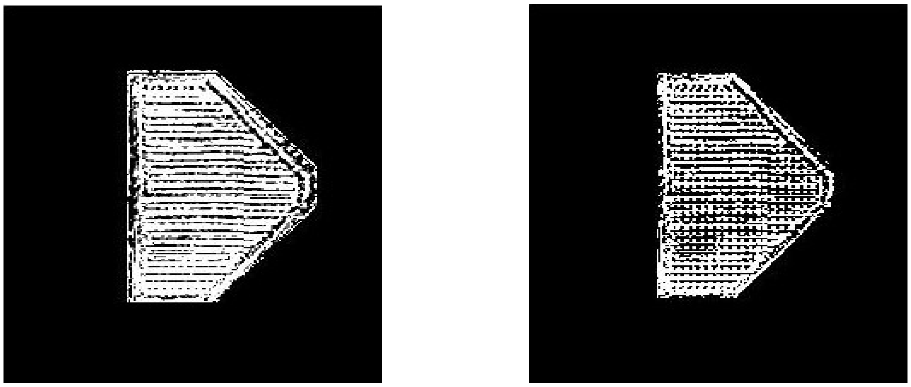
(a) Layer Image

(b) Reference Image

Figure 51: Processed Images

11.1 Good Print

In order to extract regions where the filament should be deposited and termed as ‘Good Print’, a condition in loop is used to search pixel by pixel. Its function is to scan the two images and to find out the desired pixels. For example, the image of ’Good Print’ in comparison with the layer image is shown in figure 52.



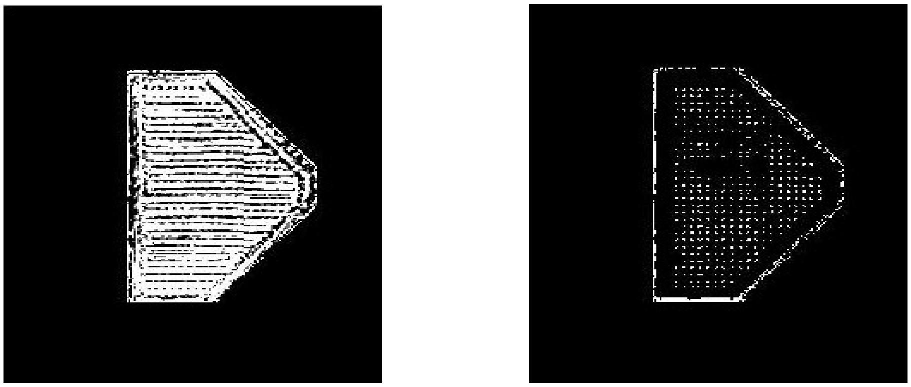
(a) Layer Image

(b) Good Print

Figure 52: Comparison for Good Print

11.2 Unwanted Print

These regions are those where the filament should not be deposited according to the reference image. This image can also depict the value of parameters like line thickness required for the section 7. The ‘Unwanted Print’ image in comparison with the layer image is shown in figure 53.



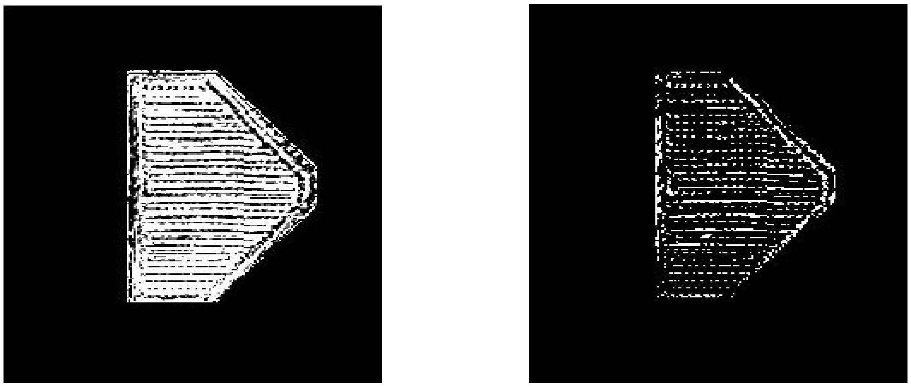
(a) Layer Image

(b) Unwanted Print

Figure 53: Comparison for Unwanted Print

11.3 Missing Print

In some regions, the absence of printed filament can also be noticed. This image is shown in figure 54 along with the layer image.



(a) Layer Image

(b) Missing Print

Figure 54: Comparison for Missing Print

12 Conclusion and Future improvements

12.1 Observations

The effectiveness of this algorithm and validation of all the results is highly dependent upon the following parameters

- Distortion removal
- Filament properties like colour and size
- Calibration
- Ambient lighting
- Adhesive used over bed and cleanliness

12.2 Conclusion

- **Defects and its effect on the components.**

Additive manufacturing or 3D printing is a process of manufacturing an object by using layer on layer addition of the material. Since this process takes place in a different environment and through different processes, we have a number of defects in the process. The common form of defects can be seen in section 3.5. The aforementioned defects make the parts unusable which leads to a loss of time and money as these products are really complicated.

- **Technical advancements and defect detection previously**

The technology for the detection of defects in the additive manufacturing process have been developed. Inspection is generally carried out after fabrication, when the components are already built. Inspection methods include ultrasound, x-ray or even destructive procedures which the parts need to be cut into section or in the form where the part cannot be used ahead. Also in this kind of approach, the inspection cannot be done for all the parts processed particularly in the case when they are highly compact and the geometries are very complex. This approach also requires machines and apparatus which are costly and a trained person to operate them all the time.

- **The idea presented in this thesis**

The purpose of this thesis is to come over the idea of physical inspection and detecting defects through inspection of cross-section by cutting the part. That is to make a system which is automated once set up and finds detects at each layer of deposition of material while the process is going on. The defect detect mechanism chosen is to take images of the layer by layer, match with the actual sliced images overlap them and then see the areas of defects, for example the area where the material was not deposited or the one where unwanted material deposition occurred.. The images give information about the process quality and also helps in determining factors like strength, reliability etc. Hence the objective of this is achieved.

12.3 Future Improvements

Future improvements can be applied to the proposed methodology. Some suggestions could be:

- Algorithm based on lines rather than pixels.
- Using Machine learning algorithm to determine type of defects.
- Calibration free system.
- Improvement on user interface.
- Application on mobile phones in realtime.
- Active control mechanism for defect correction.

References

[1] https://3dprintingindustry.com/3d-printing-basics-free-beginners-guide#01-basics

[2] Syed A.M. Tofail , Elias P. Koumoulos , Amit Bandyopadhyay , Susmita Bose , Lisa O'Donoghue,Costas Charitidis (2018) "Additive manufacturing: scientific and technological challenges, market uptake and opportunities" https://www.sciencedirect.com/science/article/pii/S1369702117301773

[3] https://www.3dhubs.com/knowledge-base/introduction-fdm-3d-printing#what

[4] http://fab.academany.org/2018/labs/fablabkochi/students/suhail-p/week6.html

[5] https://www.simplify3d.com/support/print-quality-troubleshooting/

[6] https://3dprint.com/26590/dual-parking-extruder/

[7] https://www.matterhackers.com/articles/extruders-101:-a-crash-course-on-an-essential-component-of-your-3d-printer

[8] Lecture slides about Additive Manufacturing, Prof. Paolo Minetola, Politecnico di Torino.

[9] https://all3dp.com/3d-slicer-settings-beginners-8-things-need-know/

[10] https://ultimaker.com/software/ultimaker-cura

[11] https://manufactur3dmag.com/8-techniques-for-post-processing-of-fdm-3d-printed-parts/?fbclid=IwAR37AscUxTRsoz-yoKlEQs4PrS_RRcEaUZ4lA6hjnidU8Ecrcx-2BC5tobo

[12] https://opentotum.github.io/fabtotum.com-backup/index.html

[13] https://in.mathworks.com/help/vision/ref/cameracalibrator-app.html