Quality Assurance in Additive Manufacturing Through Mobile Computing

Sam Hurd(✉), Carmen Camp, and Jules White

Vanderbilt University, Nashville, TN 37235, USA
{sam.p.hurd,carmen.camp}@vanderbilt.edu, jules@dre.vanderbilt.edu

Abstract. The increase in use of consumer 3D printers for in-home or small business manufacturing may signal the start of an additive manufacturing revolution, but unfortunately these printers are often error prone. In order to remedy the time and materials lost when a failed print continues on a low-end 3D printer, a cost-effective method is needed to monitor the quality of a print and stop it when an error occurs. This paper presents an approach to using a commodity smartphone and computer vision to perform quality assurance on selected layers of a 3D print. Our results indicate that a commodity mobile device using our technique is capable of accurately detecting printing errors and then effectively determining whether or not a print should continue.

Keywords: Consumer 3d printers · Cost-effective · Quality assurance · Computer vision

1 Introduction

Emerging Trends and Challenges. Additive Manufacturing (AM) is the process of constructing a 3D object from a model design file by joining together solid materials [12]. A 3D model of the object is created and then translated into a series of instructions, such as GCode, which dictate the movements of an AM machine to construct the physical object layer by layer [1]. For example, a Fused Filament Fabrication 3D printer (FFF) works by moving a heated nozzle that extrudes thin layers of plastic to build a 3D object layer by layer as shown in Fig. 1.

AM machines are being used increasingly as their capabilities develop. They minimize waste and can create lighter, yet sturdy, parts more quickly for some applications. Additionally, they provide a method of building objects with complex interiors, a challenging task that few machines can accomplish accurately. For example, General Electric is creating a new fuel nozzle for aircraft engines, and these parts are being created using AM machines to make the fuel nozzles lighter, simpler to craft and more durable [2]. In addition to creating useful parts for aircrafts, AM machines are also beginning to be used for medical purposes. For example, researchers are beginning to create organs through the AM of human tissue [4] which would allow for failed organs to be replaced. As AM machines continue to develop, their potential continues to grow.

S. Sigg et al. (Eds.): MobiCASE 2015, LNICST 162, pp. 203–220, 2015.
DOI: 10.1007/978-3-319-29003-4_12

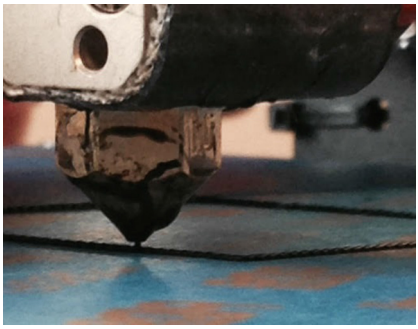


Fig. 1. The heated nozzle extruding filament to produce the sides of a square.

Recently, inexpensive AM machines have become more available allowing consumers to bring manufacturing into the home [3]. While printing is still not fast as it can take over an hour to print even a small box, consumer 3D printing is increasingly becoming a viable option for the future of small-scale, in-home manufacturing. There is a large and growing "maker" community that is using consumer AM machines to design and print complex parts at home and in small businesses.

Open Problem ⇒ Low-cost Quality Assurance for 3D Printers. Low-cost AM machines that are used by consumers for in-home or similar small scale printing are error-prone. With current consumer-focused devices, when an error occurs, the machine continues working wasting time and materials. Additionally, errors that occur early may not be detectable on the finished product and could compromise the structural integrity of the final object, especially in machines that are lower-cost, but also not necessarily professional or commercial grade [10].

Quality assurance and detection for 3D printing is still in early stages, and while some processes have been developed for industrial uses [5] or for analyzing the instructions' data points for correctness before the print [6], there are no automated mid-print quality control processes for low-end or home scale 3D printers. This paper focuses on addressing this gap in low-cost quality assurance systems.

Solution Approach ⇒ Mid-print Quality Assurance with a Commodity Mobile Device. To address the print quality detection problem with 3D printing, we created a cost effective mid-print quality assurance process using a mobile device that can easily be utilized alongside commercial FFF printers to detect when the printer has made an error through the use of image analysis. This solution uses a mobile device mounted above a 3D printer to capture images of selective layers after their completion, perform computer-vision based analysis of the layers, and if an error has occurred, stop the print and notify the user. The approach relies on automated pre-print analysis of the 3D model to predict what each layer should look like when printed and rewriting of the GCode sent the printer to insert quality assurance checks in the print instructions.

In Sect. 5, we present empirical data that we have gathered from experiments with a MakerGear M2 printer showing that our solution is able to effectively identify errors through a smartphone or tablet's camera with a high level of accuracy. These experiments centered on examining the accuracy of smartphone-based visual quality control for AM. This paper provides the following contributions to the study of using mobile computing for mid-print quality control in AM:

- We provide a method for extracting a 2D representation of what layer should be printed and where this layer should be printed.
- We provide an image based analysis of the printed object that can pinpoint errors through use of a mobile device's camera.
- We present empirical data comparing the accuracy and speed of two different image analysis approaches for AM quality assurance.
- Overall, we provide a mobile device-based architecture for quality control on consumer AM machines.

The remainder of the paper is organized as follows: Sect. 2 describes typical problems that arise during a 3D print, which we use as a motivating example throughout the paper; Sect. 3 describes the challenges of creating an effective mid-print quality assurance process; Sect. 4 presents our technique for analyzing a manufactured object mid-print and determining when an error has occurred; Sect. 5 presents empirical results demonstrating the ability of our process to effectively distinguish between failed prints and good prints; Sect. 6 compares our work with related research; and Sect. 7 presents concluding remarks.

2 Motivating Example

The need for a quality assurance process can easily be demonstrated using a commercial 3D printer using Polylactic Acid (PLA) as the printing filament [17]. Several types of errors can occur when printing with this setup and watching the print to manually catch errors is time consuming. To print a very basic one-inch cube takes approximately two hours, resulting in much lost time and effort if the print must be redone.

Figure 2 illustrates two errors that are common in commercial 3D printing. When printing a curved area, the PLA sometimes does not effectively stick to the previous layer and creates a chord between two points on the curve. Additionally, present in the figure is an internal error caused by the filament not properly sticking to the printerbed and being pushed around by the nozzle as it continues to print. 3D printing is becoming increasingly utilized in areas from manufacturing aircraft parts to printing organs, but unfortunately errors still occur often during prints. It is important that an effective error detection mechanism is developed to locate the error, save materials and prevent structurally unsound objects from being printed.

3 Challenges

In order to develop a mobile image-based approach to quality control, a number of key challenges must be overcome:

Fig. 2. The beginning of a failed print. Internal error caused by PLA not sticking to the printerbed, and external error caused by PLA not sticking to the previous layer.

3.1 Challenge 1: Generating a 2D Representation of a Print Layer

In order to effectively detect errors in a 3D printed object, the monitoring device must know what the object should look like at different stages of the manufacturing process. That is, the imaging process must be able to predict what the 3D model will look like in the physical world when printed on a specific 3D printer's bed with the chosen printing materials. For example, different plastics may be varying colors and printer beds vary in design and color. However, the image analysis must be able to predict what a layer is expected to look like while taking into consideration each of these factors. Figure 2 illustrates internal and external errors in printing, but without the monitoring device knowing what is supposed to be printed, it would not be able to identify this error. This challenge appears in all prints. With the extensive variety of objects available to print, it is impossible to assume internal gaps or thin filaments around the outline are errors.

3.2 Challenge 2: Time Synchronization with the Printer

During printing, the printerbed moves constantly along two axises as filament is extruded to build the object. Because the object's position relative to the monitoring device is constantly changing, it is difficult to know where the object will be at any given time to begin the quality analysis. To perform effective quality analysis, it is crucial that the monitoring device and the printer are time-synchronized so that the monitoring device runs the quality analysis check at the right time.

The challenge of synchronization manifests itself in every print as the movement of the printer bed occurs during every print to build the object. If the printer and the monitoring device are not in sync, error could be detected simply because the object is not where the monitoring device expects it to be due to printerbed movement.

3.3 Challenge 3: Accurate Positioning of the Monitoring Device

Effective quality analysis of a 3D print relies on the print monitoring device knowing where the print should occur. If the monitoring device is not aligned perfectly with the printer bed, a good print could be deemed erroneous simply because the print is not where the device believes it should be. Errors identified through the incorrect positioning of the monitoring device manifest themselves in various manners. If the device is too close or too far from the print, it may believe the print is an incorrect size. If the device is tilted or not in line with the printer bed, it may believe the print is warped or printed askew.

3.4 Challenge 4: Identifying When a Print has Failed

Identifying where an error has occured in the print is only one step towards generating effective quality assurance. All prints may contain some level of error due to natural process variation. An effective monitoring solutions needs to be able to differentiate between prints that have an acceptable amount of error and prints that should be stopped.

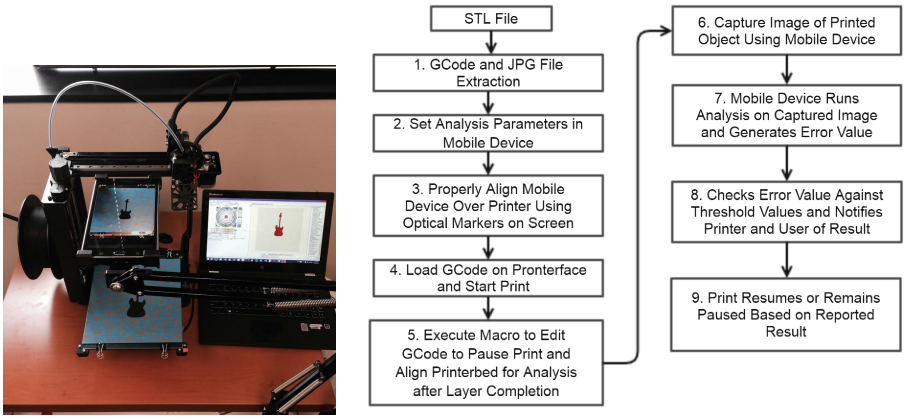
A failed print from commercial 3D printers is typically easily detectable to the human eye since we can see large internal gaps or filament outside the area the object should be printed in. While it is easy for humans to visually process whether a print is good or bad, it is difficult for a monitoring device to know how much error is too much. Section 4 describes how we address this challenge by running experiments to arrive at accepted threshold values.

4 Print Quality Assurance with a Mobile Device

The solution we propose is a process for detecting the print quality using a mobile device. Developers can utilize our approach to remotely monitor the quality of a 3D print, and the process can autonomously stop a failed print. The basic setup for the approach is shown in Fig. 3a. A small stand is used to position a mobile device's camera over the printer bed. As the printer constructs the 3D object, time-synchronized quality control checks are performed to verify the integrity of printed layers. The overall architecture is based on 4 components: (1) the 3D printer, (2) the host computer that is controlling the printer by sending GCode commands over USB, (3) a mobile device performing the visual quality assurance using an Android app, and (4) a back-end server that coordinates communicating between the host computer and the mobile device.

The workflow of the approach is shown in Fig. 3b. The initial step in the process is to generate 2D representations of the layers that are being checked for quality. The set of 2D reference images, $r_{it} \in R$, is produced by analyzing the 3D model, which is typically an STL file, at a monotonically increasing series of time steps S_t, and producing a 2D rendering of the expected top-down perspective on the model at each time step s_{ti}:

$$\forall s_{ti} \in S_t, r_{ti} = r(M, P, S_t)$$



(a) Mobile device positioned above printerbed for analysis.

(b) Step-by-step description of analysis process.

Fig. 3. The effectiveness of the analysis relies on device positioning and a nine step process.

A set of process parameters, P, provides information about the printer and materials being used, such as the color of the plastic filament. With these R 2D reference images, the mobile device will be able to compare the printed layers to what the printed layers should look like and then determine if they are acceptable. To implement $fx(M, P, S_t)$ and extract a 2D image from a 3D file, we used a transcoder provided by the Batik library. During step 1 of Fig. 3b, this transcoder turns a SVG file into a 2D format, such as PNG or JPEG [7]. These files are used for comparison to the actual imagery of the printed layers later in the process.

The most common standard for controlling AM equipment is GCode. In order to print an object, a 3D model, M, needs to be converted into a series of "slices" or layers that the printer is instructed to print:

$$G = slice(M, P)$$

These successive layers are represented as a series of GCode instructions, G, that tell the printer how to move the print head, how much material to extrude, etc. to produce the layer. The combined GCode for all of the layers is sent to the printer to produce the final object. The slicing process also relies on the process parameters, P, in order to calculate appropriate GCode for the print.

Another parameter that must be determined is the offset, $o_i \in O$, where the i_th layer will be printed on the printer bed. The same 3D object can be printed at different offsets on the printer bed and this must be accounted for in the visual quality control process. To determine the print offset information, an extraction

function is applied to the GCode that finds the offset of the layer printed at time s_{ti}:

$$o_i = o(G, P, s_{ti})$$

The next step in the process is to setup the mobile device, which is not assumed to be permanently attached to the printer, so that it can capture imagery of the print. An important step in this process is aligning the mobile device's camera with the print bed so that accurate image analysis can be performed. We added a white rectangle to the camera viewfinder fragment used in our Android quality control application. This white rectangle's dimensions are proportional to the dimensions of the printerbed, so during step 3 of Fig. 3b the user can adjust the position of the mobile device using a stand like the one pictured in Fig. 3a until the printerbed lies inside the rectangle when the printerbed is in the resting position.

To solve the time synchronization issue of coordinating the printer and the mobile device imaging, we apply a program transformation to the GCode, G, to generate a modified set of instructions that include specific synchronization points. The modified GCode, G', stops the printer at specific points in time, s_{ti}, and moves the printer bed to specific coordinates for imaging and waits for feedback from the image analysis process (described later) that the print should proceed. The transformed GCode is produced via a program transformation function that takes the original GCode and a series of synchronization time steps as input:

$$G' = \omega(G, S_t)$$

At each of these synchronization points, the printer moves the printer bed into positioning for imaging and the host computer controlling the printer sends a message to the back-end server in the cloud to send a push notification to the mobile device to begin the imaging process. The mobile device captures an image, I_t, of the printer bed and then runs one of the image analysis algorithms, $\delta(R, I, L)$, described in Sect. 4.1, to calculate an error value, e:

$$e = \delta(r_{it}, I_t, o_i, P)$$

$$\beta = b(M, o_i, I_t)$$

If e is above a configurable threshold, β, then the device sends a message back to the back-end server to notify the host computer to stop the print. Optionally, the back-end server also can send a notification to the user's mobile device (one not being used for the quality control) to allow for a decision on whether or not to continue the print. If the print is stopped, the host computer sends terminating GCode instructions to the printer to end the print.

4.1 Algorithm to Discover Misprinted PLA

To begin addressing the challenge of identifying when a print should fail, which is discussed in Sect. 3.4, we developed two algorithms to implement $\delta(r_{it}, I_t, o_i, P)$ that would be able to find and highlight error in an image of the 3D printed

object during step 7 of Fig. 3b. The first algorithm we used to implement δ involves image subtraction and the second algorithm we use involves searching a single picture. These two algorithms use a similar process at the end to arrive at the number of erroneous pixels, and they both depend on using black PLA and a blue or dark gold background.

Image Subtraction. The first algorithm to detect errors in a print involves an image difference method, which is applied following the completion of a layer that should be analyzed for error. The error detection by image subtraction involves two input images $I_{t-x}(o_i, P)$ and $I_t(o_i, P)$ taken x seconds apart where x is the print time up to the point the analysis is run. A classical image subtraction is performed to identify where PLA has been printed:

$$I_d(o_i, P) = |I_{t-x}(o_i, P) - I_t(o_i, P)|$$

The obtained image, $I_d(s)$, is a light color where the object has been printed while the remainder of the printerbed is dark. We can then compare the light spot in this image to r_{it}, the corresponding 2D slice, to produce the image I_f:

$$I_f = \rho(I_d, r_{it})$$

This function iterates through I_d and inserts the slice r_{it} into the image by changing the appropriate pixels in I_d to white. At the conclusion of this function, where the object should be located is entirely white, while error appears as a light color and the background remains white. As this function runs, another function to determine internal error, e_y is performed concurrently. This function analyzes all pixels p_{ab} in I_d that should contain PLA according to slice r_{it}. If this pixel is dark and therefore does not contain PLA, it is perceived as erroneous. The function to determine the number of internal errorneous pixels can be defined as:

$$s_y = \phi(I_d, r_{it}, p_{ab})$$

After these methods have finished running, the mobile device iterates through image I_f. Since the background is dark and where the object should be is white, the device can use a function s_x to find external error by searching all pixels p_{cd} in the image for light, but not white pixels. This function to find the number of external erroneous pixels can be defined as:

$$s_x = \upsilon(I_f, r_{it}, p_{cd})$$

The total number of erroneous pixels, S, can then be defined as:

$$S = s_x + s_y$$

Image Searching. The second algorithm to detect errors in printed objects is a simple image searching algorithm. This algorithm initially inserts the correlating slice, r_{it}, into the image $I_t(o_i, P)$ by iterating through the pixels of the image

and changing pixels where PLA should be printed to white. This function creates a new image, I_f:

$$I_f = \varphi(I_t, r_{it})$$

While this algorithm runs, the mobile device concurrently searches for internal error, s_y by searching each pixel p_{ab} that should have PLA according to r_{it} but does not. This error is identified by checking if the pixel p_{ab} is black before changing it to white per $\varphi(I_t, r_{it})$:

$$s_y = \psi(I_t, r_{it}, p_{ab})$$

After $\varphi(I_t, r_{it})$ has completed, a new function iterates through I_f and searches the image for external error s_x. Since the printerbed is blue, the allocated location for the object is now white, and the PLA is black, we can search each pixel p_{cd} of the image and count the number of black pixels as those indicate PLA outside of the acceptable area:

$$s_x = \varrho(I_f, r_{it}, pcd)$$

The total number of erroneous pixels, S, can therefore be defined as:

$$S = s_x + s_y$$

While the image searching algorithm is similar to the subtraction algorithm, the subtraction algorithm provides the benefit of the background being very dark if not black. When simply searching the image of the object, a color similar to the color of the PLA may be present in the image and cause the mobile device to falsely detect that as error.

4.2 Identifying Failed Prints

As discussed in Sect. 3.4, a necessity for effective quality assurance is determining a threshold value for β when the print should fail. In Sect. 4.1 we began to address this challenge by describing two processes to discover the number of pixels that contain error in an image of the printed object. The mobile device calculates the error value e using the number of pixels that contain error and the total number of pixels in the slice r_{it}:

$$e = \delta(r_{it}, I_t, o_i, P) = \frac{S}{\sum p_{ab}}$$

While calculating this error value is a crucial first step towards determining the quality of a print, this value alone is not enough. The device needs to know what error values should be passing and what error values should be failing, and then it can compare the calculated error value of a print to these reference values β to determine the quality of the print during step 8 of Fig. 3b.

By visually determining whether or not a print should pass and recording the calculated error value as well as the analysis parameters used, we could

experimentally discover threshold values that marked the line between a passing print and a failing print. After discovering these values and reporting them to the mobile device, a simple comparison between the calculated error e for any given print and the appropriate reference value β based on the analysis parameters can determine whether or not the print should continue.

5 Empirical Results

5.1 Experimental Platform

An important consideration in this research was the real-world performance of a mobile device in detecting errors on 3D prints. We conducted a series of experiments to compare the performance, in terms of accuracy and speed, of both image analysis approaches. We also provide data on how effective functions for calculating the error threshold, β, can be determined.

To conduct the experiments, we used a Samsung Galaxy Tab 3 running Android 4.4.2. The device has 8.0 GB of ROM and 1.0 GB of RAM and features a 3.0 megapixel camera. The screen is 7 in. and 1280 pixels by 800 pixels [18]. To hold the device over the printerbed, we used a modified desk lamp.

The 3D printing device we used for making the objects is a Makerbot M2 printing with black PLA. The nozzle was set to 200°C and the printerbed temperature was set to 70°C for all of the tests. The glass of the printerbed was also covered with blue painters tape that has gold lettering to easily contrast with the black PLA.

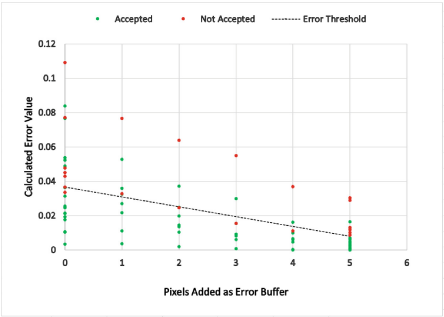
5.2 Experiment 1: Image Subtraction Analysis

First, we tested our algorithm that uses image subtraction on various prints of differing qualities to experimentally discover threshold error values that indicate when an object has too much error. Discovering these values allows us to solve Challenge 6 presented in Sect. 3.4 as we can compare a calculated error value to our threshold error values to determine the quality of a printed object. The data collected in this experiment will also show the ability of the subtraction algorithm to consistently produce similar error values for similar quality prints.

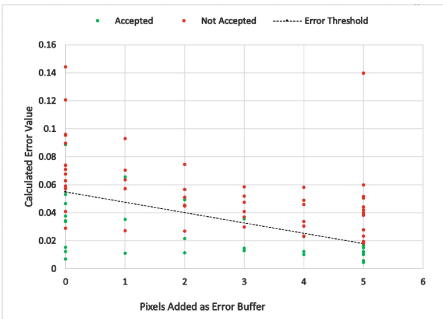
Hypothesis: Threshold Error Values. Our hypothesis was that there is a threshold error value for each permutation of subtraction analysis parameters that can be used to distinguish between passing and failing prints.

Experiment 1 Results. To experimentally discover the various error threshold values for different parameters, we collected data by executing the image subtraction algorithm on objects we printed and recording the calculated error value as well as whether or not the print should pass. Figure 4a illustrates the results of varying the buffer parameter while keeping the internal search parameter false. While there is not a clear division between accepted prints and not accepted prints, we were able to determine a threshold line, indicated by the dotted line

on the graph, that most accurately divides the two. When performing subtraction analysis not searching for internal error, we found that $error = -0.0057x + .0365$, where x is the buffer value, is a good indication of print quality with error values below that line passing and error values above that line not passing.



(a) Only searching for external error.



(b) Searching for internal and external error.

Fig. 4. Passing and failing prints when using the subtraction algorithm

Similarly, we collected data to discover the threshold error values when running the subtraction algorithm, processing the image for internal error and varying the error buffer. Figure 4b shows the results of these tests. Once again, there is not a clear division between accepted prints and not accepted prints, but we are able to determine a threshold line, the dashed line on the graph, that most accurately divides the two. When performing subtraction analysis and processing for internal error, we found that $error = -0.0074x + .055$, where x is the buffer value, is a good indication of the print quality. Prints with error values above this line are not accepted, and prints with error values below this line are passing.

From this experiment we discovered that we can use our subtraction algorithm to distinguish between good and bad prints with some accuracy. Table 1 contains various percent error values that signify that the algorithm is most successful when a 5 pixel buffer is added. This buffer allows our algorithm to be more robust to account for positioning errors discussed in Sect. 3.3. Additionally, Table 1 also indicates that false positives are not very likely when using this algorithm meaning that prints that should fail typically do fail, which indicates that this algorithm effectively stops failed prints.

5.3 Experiment 2: Image Searching Analysis

Next, we tested our algorithm that uses an image searching process to find error in printed objects to experimentally discover threshold error values that indicate

Table 1. Percent errors when using the subtraction algorithm with at least 25 samples for each calculation.

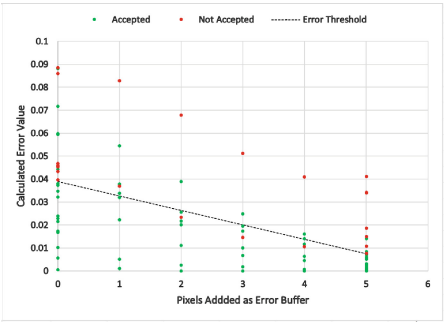
0 Error Pixels	
False negative	0.214286
False positive	0.125
Overall	0.173077
5 Error Pixels	
False negative	0.074074
False positive	0
Overall	0.040816
Average	
False negative	0.178947
False positive	0.101266
Overall	0.156069

when an object has too much error. We are also able to compare the results of using this algorithm to the results of using the image subtraction algorithm discussed in Sect. 5.2 to see which algorithm does a better job identifying the quality of a print. The data collected in this experiment will also help to solve Challenge 6 presented in Sect. 3.4 as we are experimentally finding values that can help us determine the quality of a print.

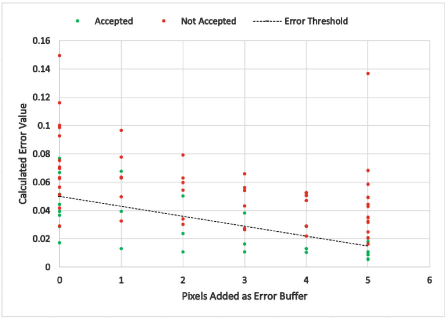
Hypothesis: Error Threshold Values. Our hypothesis was that there is a threshold error value for each permutation of searching analysis parameters that can be used to distinguish between passing and failing prints and that the results of using the searching analysis will be similar to results when using the subtraction analysis.

Experiment 2 Results. To experimentally discover the various error threshold values for different parameters when using the image searching algorithm, we printed out various objects and ran the image searching algorithm on them. By varying the parameters of the search - whether or not to search inside the object for internal error and adding an error buffer - we were able to collect data that helps us to determine the threshold values that indicate an erroneous print. Initially we ran tests where the image searching algorithm did not search for internal error, and we only varied the error buffer parameter, and by recording whether or not a print should pass as well as the calculated error value, we were able to generate the graph pictured in Fig. 5a. While accepted prints and not accepted prints are not clearly divided from each other, we were able to determine a threshold line, indicated by the dotted line n the graph, that can be used to most accurately predict the quality of a print. When running the image searching algorithm and not searching for internal error, we found that $error = -0.0063x + 0.039$ where x is the buffer value, is a good indication of

print quality with error values below that line passing and error values above that line not passing.



(a) Only searching for external error.



(b) Searching for internal and external error.

Fig. 5. Passing and failing prints when using the searching algorithm

After collecting the appropriate data for figuring out threshold values when not searching objects for internal error, we then conducted the same tests, but this time we did search for internal error. Figure 5b illustrates the results of searching the internal and external areas of the object for error using the image searching algorithm and varying the error buffer. Once again, the accepted prints and not accepted prints are not clearly separate, but we can find a threshold line - the dotted line in the graph - that most accurately defines an appropriate boundary. We found that $error = -0.007x + 0.05$, where x is the error buffer, is a good indication of print quality where error values above this line are not accepted and error values below this line are accepted

From this experiment, we did discover threshold values that can be used with some accuracy to differentiate between prints that should be accepted and prints that should not be accepted. Table 2 contains various percent error values that help show the accuracy of this method. Similarly to the results found in Experiment 1 in Sect. 5.2, using the image searching algorithm produces relatively low percent error especially in terms of false positives meaning that this algorithm only rarely continues a print that is of low quality. These results also indicate that both the image subtraction analysis and the image searching analysis are valid ways to detect errors in a print.

5.4 Experiment 3: Correlations Between Size and Analyzation Speed

Finally, we determined the cost of running the analysis program and whether the searching or subtraction method was faster. Each time a test is run over the

Table 2. Percent errors when using the searching algorithm with at least 25 samples for each calculation.

0 Error Pixels	
False negative	0.285714286
False positive	0.125
Overall	0.2115
5 Error Pixels	
False negative	0.148148148
False positive	0
Overall	.08
Average	
False negative	0.242105
False positive	0.088608
Overall	0.201149

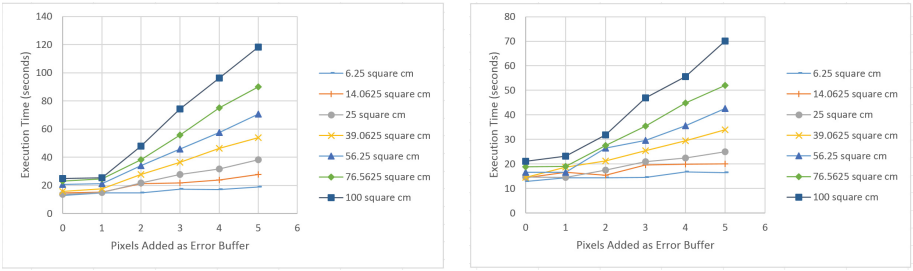
printed object, x amount of time is added to the process, with x being dependent upon the size of the object printed and the parameters for error allowance entered by the user.

Hypothesis: Time Increase with Added Size and Error Pixels. We hypothesized that through running these experiments, we would find that more error pixels and larger objects result in a slowing down of the analysis.

Experiment 3 Results. There are three main components that consume time:

1. Moving from the final location of printing to the home position required to run analysis provides a constant, baseline amount of time that does not change with variation of the parameters. The average time during our experiments was 4.718 s, with a range from 4.24 s to 5.16 s.
2. Actually running the analysis may vary based on the size of the object, and the number of error pixels added to the outside of the object. The graphs shown in Fig. 6a and b detail results of the time taken to perform search and subtraction, with varying size and parameters. Our results indicate that, from a time standpoint, simply using the searching algorithm is better, especially for larger objects.
3. Lastly, returning to the print after analysis provides another constant time. The average time calculated during our experiments was 9.145 s with a range from 7.94 s to 10.56 s.

Adding each of the formerly stated three process steps together returns the total time cost of the entire procedure. Figure 6a and b show the distinct correlation between the upward trends of both the search and subtraction methods based upon increasing numbers of error pixels that must be searched during the analyses. Comparing the two graphs also leads to the conclusion that the image



(a) Subtraction Algorithm. (b) Searching Algorithm.

Fig. 6. Algorithm execution times based on object size and error buffer.

searching algorithm is faster than the image subtraction algorithm, especially on larger prints and when the error buffer increases.

5.5 Analysis of Results

The data we collected during testing indicates that our process can effectively and consistently identify error location in a print. Additionally, using this identification, it can then identify with fairly high accuracy whether or not a print should continue.

We experimented with two different algorithms, and the results of those experiments indicate that the image searching algorithm is a better algorithm than the image subtraction algorithm. While comparison of the results from Experiment 1 in Sect. 5.2 and Experiment 2 in Sect. 5.3 leads to the conclusion that the two algorithms produce similar results in determining the quality of a print, Experiment 3 in Sect. 5.4 indicates that the image searching algorithm is a faster method.

In all experiments we printed out objects of different shapes and sizes to run the tests. While the different sizes had a large impact on execution time as outlined in Sect. 5.4, the size of the object seemingly had very little bearing on the calculated error value. Results from Experiments 1 and 2 in Sects. 5.2 and 5.3 show that all sizes of objects have the same error threshold value that can be used to determine the quality of a print.

We completely solved the problem because we were able to develop an easy to use process that accurately identified 3D printing errors. While some prints during testing stopped when they could have continued, very few failed prints continued, and this was the goal of our work. We are able to stop failed prints with our work.

Our data also shows that we are better than existing approaches. Most quality assurance processes are either industrial solutions or analyze finished product, but we were able to design a cost-effective mid-print quality assurance process.

While our solution works in many instances, there are places that it does not perform as well. As the print approaches its last layers, it sometimes creates a shadow and this shadow is sometimes perceived as error by the mobile device. Additionally, it is time consuming to perform analysis, and the break in printing that occurs during the analysis can occasionally cause improper extrusion when the device returns to printing.

6 Related Work

Although three dimensional printers have been available since the late 20th century, little research has been done to ensure the quality of a product while printing [13].

Though the taxonomy of our research is nebulous on account of the many uses and possibilities for additive manufacturing, it is clear that all research utilizing three dimensional printers face the same issue of security and needing a guarantee that precious time and resources will not be wasted in making a faulty object. Other examples of similar work number very few, revealing the necessity for a sort of quality assurance when printing [14]. Developers have begun to integrate sensors into their additive manufacturing processes to maintain constant, perfect conditions. Others detect structural insecurities in designs and fix them prior to printing. Areas such as those in the medical industry troubleshoot until theyve built a quality product, then design it with renewing chemicals to repair itself should any damage occur after its been created.

Monitoring Manufacturing Conditions During Print. Sigma Labs [5] patented a program in spring 2014 that balances the amount of energy entering the powder layer of an additive manufacturing machine to maintain a constant, ideal temperature for the product. While their controlling sensor monitors the object and fabricating conditions throughout the process, our analyzation runs after a certain time interval or number of layers, depending upon user input. Instead of controlling situational parameters, we test the freshly manufactured hardware to determine whether or not it is a quality product and thus whether the build should be terminated.

Structural Security in Design. Benes [8] collaborated with Adobes Advanced Technology Labs to develop a software that detects weaknesses in initial STL-type files that are sent to printers. Instead of spending time and resources to fabricate large, layered designs, their program identifies weak and structurally unsound areas on the design template, then fixes these frailties so that the printer is sent a stronger, more durable construction.

Self Renewing Materials Post-construction. Lewis and White et al. [9] began researching printable materials to self-healing organs or vessels made of tissue similar to that found naturally in the human body [19]. Their approach to mending damaged or flawed printed products is to implant chemicals within them to detect a change in the structure and release healing cells when erosion occurs. Though our project detects flaws during the building process to create

an ideal result, it is not specific to a need such as medical implantation, and thus does not require a self-mending feature.

Optical Assessment of Print with Robot. The most similar design to ours appeared recently from Alcona in the form of a multi-axis robot attached to a sensor [20]. This setup requires the purchase of the robot and sensor, and is not mobile as ours is; however, the design allows a live, constantly updating look at the quality of a print throughout the printing process. While printers are not updated as to whether the print is performing well or not if they are not present and observing the setup, it is indeed a powerful tool to quickly and accurately provide quality assurance without pausing the print after a certain number of layers or amount of time as ours does.

7 Concluding Remarks

It is challenging to determine the quality of a 3D printed object. This paper describes how we were able to use a mobile device and computer vision to identify errors in a print and then distinguish between prints that should be continued and prints that should stop.

The following are lessons learned from our efforts thus far:

- By adding an error buffer, we created a robust analysis process that could accurately identify printing errors and then determine the quality of a print even when the mobile device's positioning is slightly off.
- In future work, we plan to develop a faster algorithm for determining error to reduce the time cost of using the analysis during a print.

This research has been supported in part by the National Science Foundation and Department of Homeland Security through Grant #CNS1446303.

References

1. Gibson, I., Rosen, D., Stucker, B.: Additive Manufacturing Technologies: 3D Printing, Rapid Prototyping, and Direct Digital Manufacturing, 2nd edn. Springer, New York (2015)
2. 3D Printing Creates New Parts for Aircraft Engines. GE Global Research, Web. 13 July 2015
3. Rega, S.: How 3D Printing Will Revolutionize Our World. Business Insider. Business Insider Inc., 22 August 2014, Web. 27 July 2015
4. Mironov, V., Boland, T., Trusk, T., Forgacs, G., Markwald, R.R.: Organ printing: computer-aided jet-based 3D tissue engineering. Trends Biotechnol. **21**(4), 157–161 (2003)
5. Sigma Labs, Inc.: Announces Patent Filing of Unique Sensor Invention That Helps Both Process Development and Quality Assurance in Additive Manufacting of Metal Components. Sigma Labs Inc. , 25 March 2014, Web. 10 July 2015
6. Millsaps, B.B.: Trinckle 3Ds Free Error Detection Service: No More Misprints!. 3DPrintcom, 17 October 2014, Web. 10 July 2015

7. DeWeese, T., Hardy, V.: Introduction to the Batik Project [PDF document]. http://old.koalateam.com/ftp/batik/apacheCon.pdf
8. Walton, Z.: Purdue University Professor Fixes Major Flaw In 3D Printing - WebProNews. WebProNews. 19 September 2012, Web. 21 July 2015
9. Groopman, J.: Print Thyself. New Yorker, 24 November 2014, Web. 7 July 2015
10. Cel Robox 3d Printer Review. IT Pro, Web. 29 July 2015
11. FAQ: CubeX 3D Printer. Cubify, Web. 28 July 2015
12. Gibson, I., Stucker, B., Rosen, D.: Additive Manufacturing Technologies, 2nd edn. Springer, New York (2015)
13. History of 3D Printing: 3D Printing Industry. May 2015, Web. 29 July 2015
14. Molitch-Hou, M.: EOS Partnership Signals 3D Printing Quality Assurance for Aerospace. 3D Printing Industry, 21 January 2015, Web. 29 July 2015
15. Parse: Parse, Web. 29 July 2015
16. PrintRun: Pronterface, Web. 29 July 2015
17. Royte, E.: Corn Plastic to the Rescue. Smithsonian. August 2006, Web. 29 July 2015
18. Samsung Lays Out Which Devices Will Get Android 4.4.2 KitKat. Android Central, Web. 29 July 2015
19. Soft Tissue Repair and Healing Review. Electrotherapy, Web. 29 July 2015
20. Woodcock, J.: 3D Metrology Robot for Automated Optical Quality Assurance. TCT, 27 July 2015, Web. 30 July 2015