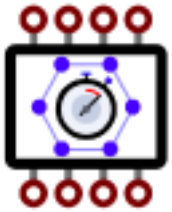


Adaptive Real-Time Systems Laboratory (ARTS-Lab)

Camera-in-the-Loop System for Additive Manufacturing

EMCH 427-428 – 2021-2022

Team Members: Malichi Flemming, Braden Priddy, Tyler Owens, Saif Wilkes-Davis

Industry Sponsor: Dr. Austin Downey

Department of Mechanical Engineering
University of South Carolina
Columbia, SC 29201

Table of Contents

I. Executive Summary *4*

II. Background *6*

III. Customer Needs *7*

3.1 Introduction 7

3.2 Methodology 7

3.3 Interpretation of Needs 8

3.4 Prioritization of Needs 10

IV. Specifications *14*

4.1 Introduction 14

4.2 Metrics Table 15

4.3 Metrics Matrix 16

4.4 Outcomes 16

V. Functional Concept *18*

5.1 Introduction 18

5.2 Concept Combinations 20

5.3 Research performed 27

VI. Concept Selection *32*

6.1 Introduction 32

6.2 Concept Screening 33

6.3 Concept Ranking 34

6.4 Final Concept 34

VII. Product Architecture *36*

7.1 Introduction 36

7.2 Product Layout 36

7.3 Interactions 37

7.4 Geometric Layout 38

VIII. Engineering Analysis *40*

8.1 Introduction 40

8.2 Analyses Performed 40

8.3 Conclusions/Recommendations 47

IX. Manufacturability 48

9.1.1 Design for X 48

9.2 Build instructions 54

X. Economic Analysis 66

10.2 Costs, Savings, and Revenues 66

10.3 Analysis 68

10.4 Break even analysis 69

10.5 Profitability analysis 70

10.4 Summary 71

XI. Product Testing 74

11.2 Test Plan 74

11.4 Conclusions/Recommendations 83

XII. Final Design 84

XIII. Conclusion 86

XIV. Appendix 88

14.1 References 88

I. Executive Summary

The Adaptive Real-Time Systems Laboratory is a research group at the University of South Carolina that does interdisciplinary research. They want a product that does monitors and collects data on a selective laser melting printer for research of a Ph.D. candidate.

The needs of The Adaptive Real-Time Systems Laboratory were determined by doing several interviews with Austin Downey and Yangzhou Fu and touring the printer, and after detailed analysis, it was determined that light spectrum, speed of data acquisition system, speed of code execution, detecting melt pool and quantifying melt pool were critical to satisfying those needs. It was furthermore determined that a product to satisfy their requirements would need to be able to process 200 frames per second, be portable, have a user-friendly repository of work, and able to accurately and reliably generate contours with area calculations. It was thus determined that the product mission should be the ARTS-Lab at the University of South Carolina needs a vision system that can accurately track, calculate the brightness, and area of the melting sputters during the SLM (Selective Laser Melting) process while functioning in real-time.

Given the above requirements, multiple concepts were researched and considered, as described in this report. The ultimate concept selected to best satisfy The Adaptive Real-Time Systems Laboratory's needs was a FLIR camera with a bandpass filter in conjunction with a laptop for storing and displaying data and Speedgoat controller as a real time target for Simulink code to process the images. Details related to concept selection and subsequent product architecture are supplied.

Speed of computational system, visibility of melt pool, and the accurate translation of brightness to contours with area and locations are critical to functionality, and detailed engineering analyses

are provided, demonstrating how expected light spectrum and heat transfer of the from the laser to the PLA housing were modeled, under conditions of heat generated by the laser, and the design was optimized.

Parameters included the camera's execution speed, the melt pool's light spectrum accuracy, and the code's processing speed are high priorities for production of the product. Therefore, the design was optimized for these parameters which consisted of measuring how pixel size affected processing time and analyzing individual functions in the code.

The prototype was tested by assessing different image processing methods to optimize speed and the melting pool's true area, displaying parameters versus not displaying parameters, and frames per second versus the size of melting sputter under conditions ranging from optimal to non-optimal.

Economic analysis of the scope of expenses associated with development of the prototype indicate that, for a $24,253.40 investment, The Adaptive Real-Time Systems Laboratory will recover all costs within the first unit sold and start earning profit at the same time.

II. Background

The Adaptive Real-Time Systems Laboratory is an interdisciplinary research laboratory at the University of South Carolina. It is focused on developing tools and solutions that enable systems (e.g., mechanical systems, aerospace structures, and civil infrastructure) to adapt to their environments in real-time. It was created by and is led by Dr. Austin Downey, a professor at the University of South Carolina in the Mechanical Engineering Department.

Digital twins will be a major factor for manufacturing in Industry 4.0 to increase reliability by allowing real-time corrections to a process. The ARTS-Laboratory, in partnership with the Metal 3D Printing Laboratory operated by Dr. Lang Yuan, is working to reduce waste by increasing the accuracy and repeatability of the Selective Laser Melting 3D printer at the McNair Center.

As part of the digital twin for this project, a computer vision system will be used to monitor the performance characteristics of the printing process. With this data, Yangzhou Fu, a Ph.D. candidate under Dr. Downey, will be able to apply a machine learning process to better the additive manufacturing process with an in-situ system.

III. Customer Needs

3.1 Introduction

Determining customer needs accurately is critical to have a well-defined scope and a successful project. Dr. Austin Downey of the ARTS Laboratory at the University of South Carolina needs an in-situ monitoring system of a selective laser melting printer. With the guidance of Yanzhou Fu, a Ph.D. student at the ARTS Laboratory, a computer vision system was created. This will be used and tested in conjunction with the printer found in the McNair Center at the University of South Carolina that is under the purview of Dr. Lang Yuan. One of the goals mentioned by Dr. Downey was for the project to be useable by other research groups and hopefully industry groups. Ultimately, the vision system needs to track the sputters, also known as molten ejecta, created by the rapid heat of the metal powder. This project is designed to be one half of a digital twin to do real-time corrections to SLM processes to improve the reliability to make it a more accepted and viable practice.

3.2 Methodology

Dr. Downey and Mr. Fu were interviewed to understand what they wanted from the project. This was done over the span of a few weeks to see if the goal would change based on the input. Additionally, research was performed by watching Georgia Institute of Technology videos to help understand what problem was being approached since this was a new concept. Additionally, research was done to determine how the printer worked so it could be determined where the issue in printing most likely was. The McNair center was visited with Mr. Fu to see the printer and its

current setup. Afterward, discussions were had to determine what needs were most relevant via a voting system.

3.3 Interpretation of Needs

Table 1: Original client needs

Needs Group	Needs	Needs #
Camera/Vision System	The system needs to be able to house the existing cameras and other future potential cameras	1
	The camera mount must be stable.	2
	The camera system must have a consistent field of vision.	3
	The cameras must have appropriate frames per second to keep up the SLM's scanner speed.	4
	The cameras must have a proper light spectrum to view the part and the melt pool while the printing is occurring.	5
	The camera must be protected	6
DAQ/ Controller	The data acquisition system must be able to interface with available USB cameras as well as other commonly used USB camera systems.	7
	The data acquisition system must be compatible with a real time system or fast enough to keep up with the cameras.	8
	The data acquisition system must be cost effective.	9
	The data acquisition system must be self-contained and portable.	10
Code	The code must be executed fast enough	12
	The code must be able to find the melt pool and outline the print area.	13
	The code must be able to quantify melting pool area and sputters.	14
	The code must track the x,y position of the melt pool	15
	The code must have adequate documentation for future users.	16
	The code must be able to display important parameters to the end user.	17
Future Use/ Documentation	The system must come from existing equipment or 3D printed parts.	18
	The system must have a repository for explaining how the system works.	19
	The system must have a friendly GUI.	20
	The documentation must be easy to use from technically literate people.	21

Computer vision systems are made of several components such as image sensors, lens, vision processor, communication interface, and lighting. The vision system consists of a camera and a

custom holder to fulfill the image sensor and lens requirements. The data acquisition system consists of a speed goat controller that handles the vision processing and communication aspects of the design. Lastly, the lighting is created by the laser, melt pool, and plasma, and is filtered using a bandpass filter. Each of the elements is broken out by subcomponent.

Needs Explanation:

1. The system needs to be able to house the existing cameras and other future potential cameras - To reduce cost, the use of existing equipment is a consideration.
2. The camera mount must be stable - To avoid vibrations from people walking nearby, the system needs to be held in place.
3. The camera system must have a consistent field of vision - The camera needs to see the same area in a repeatable fashion
4. The cameras must have appropriate frames per second to keep up the SLM's scanner speed – The system needs to process frames at a set interval such that the distance the laser moves does not exceed a critical dimension length
5. The cameras must have a proper light spectrum to view the part and the melt pool while the printing is occurring – The camera must be able to see relevant aspects of the printing process
6. The camera must be protected - The system must be able to be reused for multiple other projects
7. The data acquisition system must be able to interface with available USB cameras as well as other commonly used USB camera systems – Dr. Downey has a FLIR USB camera that would be a preference for the system
8. The data acquisition system must be compatible with a real time system or fast enough to keep up with the cameras - The camera needs to see enough information fast enough to give accurate data on the melt process
9. The data acquisition system must be cost effective – Camera systems already exist, this system needs to save the ARTS laboratory money
10. The data acquisition system must be self-contained and portable – This product is a temporary addition to the SLM printer. It must be able to be removed and placed back again.
11. The code must be executed fast enough - The code must not prevent the camera system from achieving an appropriate speed
12. The code must be able to find the melt pool and outline the print area - The system needs a parameter to track for predictions later. By finding the melt pool and outlining it, it is possible to judge the consistency of the melting process and find deviations.
13. The code must be able to quantify melting pool area and sputters - The sputters, also known as molten ejecta, are a major source of defects in the printing process. Tracking their existence and size may make it possible to predict and eliminate them.

14. The code must track the x,y position of the melt pool - The code must be able to determine the location of the pool for reference of size later to improve the accuracy of the data by correcting locations relative to the camera
15. The code must have adequate documentation for future users – CAD and code must be available
16. The code must be able to display important parameters to the end-user - The end-user should be able to track the performance and correlate it to the printed quality.
17. The system must come from existing equipment or 3D printed parts – To save cost in a one-off prototype, the sponsor needs the part to be inexpensively made and customizable using resources available in the ARTS laboratory
18. The system must have a repository for explaining how the system works – To make future work easier, code, instruction, CAD files, and other documents relevant to the project.
19. The system must have a friendly GUI – The end-user needs to be able to assess what is happening
20. The documentation must be easy to use for technically literate people- The end documentation must be accessible to those likely to use it

3.4 Prioritization of Needs

To determine the most important needs, a voting system [Table 2] was implemented. The most important of the needs were given a four, the next most important given a two, marginally important given a one, and the least important needs are given a zero. The total that must be spent by each member is forty-eight.

Table 2: Point interpretation

Score	Interpretation
0	Minimal Needed
1	Marginally Important Need
2	Mostly Needed
4	Most Important
Must Spend	48

Table 3: Affinity diagram

Needs #	Tyler	Malichi	Saif	Braden	Total	Rank
1	2	2	2	1	7	13
2	1	4	2	4	11	8
3	4	2	2	2	10	9

4	2	2	2	1	7	13
5	4	4	4	4	16	1
6	2	2	2	2	8	11
7	4	2	4	4	14	6
8	4	4	4	4	16	1
9	2	2	2	2	8	11
10	0	1	1	1	3	18
12	4	4	4	4	16	1
13	4	4	4	4	16	1
14	4	4	4	4	16	1
15	4	4	4	2	14	6
16	1	1	2	2	6	16
17	2	2	2	4	10	9
18	0	0	0	1	1	20
19	1	1	1	1	4	17
20	2	2	2	1	7	13
21	1	1	0	0	2	19
Individual Total	48	48	48	48	192	

Table 4: Top five results of first vote

Needs #	Tyler	Malichi	Saif	Braden	Total	Rank
5	4	4	4	4	16	1
8	4	4	4	4	16	1
12	4	4	4	4	16	1
13	4	4	4	4	16	1
14	4	4	4	4	16	1

Top Five Needs:

1. The cameras must have a proper light spectrum to view the part and the melt pool while the printing is occurring.
2. The data acquisition system must be compatible with a real-time system or fast enough to keep up with the cameras.
3. The code must be executed fast enough
4. The code must be able to find the melt pool and outline the print area
5. The code must be able to quantify melting pool area and sputters

Issue: While a top five was determined [Table 4], it was determined in the first round of voting [Table 3]. This does not meet the requirements of a multiple voting process. There is a fair

amount of overlap in the client's needs beyond the top five and the top five fail to consider the full scope of the needs of the sponsor. Therefore, additional needs were generated by combining other high-ranking needs and appending them to the top five. The most important needs after modifications, voting, and discussions with the sponsor are seen below.

Vision System

1. The camera system must have a consistent field of vision - To make the product as reliable as possible, it is important that the edges of the images be consistent, so pixels represent the same space.
2. The cameras must have a proper light spectrum to view the part and the melt pool while the printing is occurring – For the system to work optimally, it is important to ignore extraneous sources of light.

Data Acquisition System

3. The data acquisition system must be self-contained and portable – This product is a temporary addition to the SLM printer. It must be able to be removed and placed back again.

Code

4. The code must be executed fast enough. .5 ms (Allows for 200fps: Camera Max 226) - The system needs to track the rapid emissions of molten ejecta created by the laser heating process. The laser has a maximum travel speed of 2 meters per second. That means in one frame, the laser can travel up to 1 mm which may be a critical for tolerances depending on the application.
5. The code must be able to find the melt pool and outline the print area - The system needs a parameter to track for predictions later. By finding the melt pool and outlining it, it is possible to judge the consistency of the melting process and find deviations.
6. The code must be able to quantify melting pool area and sputters - The sputters, also known as molten ejecta, are a major source of defects in the printing process. Tracking their existence and size may make it possible to predict and eliminate them.
7. The code must be able to display important parameters to the end-user - The end-user should be able to track the performance and correlate it to the printed quality.

Future Use/Documentation

8. The system must have a repository for explaining how the system works – To make future work easier, code, instruction, CAD files, and other documents relevant to the project.

3.5 Mission Statement

The ARTS-Lab at the University of South Carolina needs a vision system that can accurately track, calculate brightness, and area of the melting sputters during the SLM (Selective Laser Manufacturing) process while functioning in real-time.

IV. Specifications

4.1 Introduction

After the needs of the client were acquired over the course of several meetings with the sponsor Dr. Downey and his Ph.D. student Yanzhou Fu. These needs went through three sets of modifications as new information was exposed or issues arose. The list of client needs was reduced to only the most vital 8. The needs were finalized after several rounds of voting using scores ranging from 0-2.

After establishing and confirming the needs with the client, a needs metrics matrix was crafted to help quantify the success throughout the project. Each of the client's needs is weighted by importance based on the prior ranking of needs. For the needs metric matrix, a scale of 0, 1, 3, and 9 was used. This was to emphasize items of utmost importance by using an exponentially increasing ranking system adding more variance and preventing mass tying. The scale was in ascending order of importance such that 0 had no importance and 9 had the most.

During this process, research was conducted to help generate specifications. Similar research was found and used to find possible values for comparison. Most of the specifications came from Dr. Downey and Mr. Fu. Beyond just setting specifications for the project to be considered a success, fallback specifications were established to keep into account the wide range of values that would arise as a result of the research-based environment of the project.

4.2 Metrics Table

Table 5: Metric table with target and fallback values

Needs	Units	Target specs	Fallback	Reference
Needs to detect melting pool (brightness)	Brightness	0-255	0-255	3
Needs to detect melting pool (size)	Pixel value	Radius via contour		3
The cameras must have a proper light spectrum	Boolean	True positive	TP/TN	6
Code must execute fast	ms	.5	33	3
Setup needs to be as compact and mobile as possible	cm^3	50 cm^3	100 cm^3	5
The code must be able to find the melt pool and outline the print area.	Boolean	True Positive	TP/TN	5
The code must be able to display important parameters to the end-user	Boolean	True Positive	TP/TN	5
The system must have a repository for explaining how the system works	Boolean	True Positive	TP/TN	

Customer requirements/needs [Table 5] are in the leftmost column followed by the units used to measure the need. Next are the target specs, which are the goals for the values attained. Fallback values are acceptable values but not desired. Four client needs require true or false measurements, so Boolean is the most often used metric for this project.

The idea of using a camera-in-the-loop system for detecting errors during an SLM (Selective Laser Melting) process is a new idea that has been explored mostly by universities and research groups. Because of the uniqueness of the idea, there are very few industry standards around the targets of the project. Most of the target and fallback values are inferences made from other similar articles and discussions with the project sponsors (Downey, Austin, and Yanzhou, Fu). These resources gave a good idea of what the system's target and fallback values should be in order to be successful.

4.3 Metrics Matrix

Table 6: Needs metrics matrix

Improvement direction		--	--	↑	↑	↓	
Units		mm/pixel valu	Yes/NO	Frames/min	Pixels	lbs	
Customer Requirements	IWC	radius and brightnessof the melting pool	Boolean	Processing time	Video resolution	Weight of setup	total
The camera system must have a consistent field of vision.	5	9	9	3	3		24
The cameras must have a proper light spectrum to view the part and the melt pool while the printing is occurring	4	9	9				18
The data acquisition system must be self-contained and portable.	3		3			9	12
The code must execute fast enough	5		3	9	3		15
The code must be able to find the melt pool and outline the print area.	4	9	9	3	9		30
The code must be able to quantify melting pool area and sputters.	5	9	3	9	3		24
The code must be able to display important parameters to the end-user	5		9	3			12
the must system must have a repository for explaining how the system works	3		9				9
Raw Score		162	228	132	81	27	630
Relative Weight %		0.257142857	0.3619	0.20952381	0.128571429	0.042857143	
Rank Order		2	1	3	4	5	

The needs metrics matrix [Table 6] identifies the overall importance of each engineering characteristic and ranks them in order of most important to least important. Out of the five engineering characteristics, the most important has a ranking of one while the least important has a ranking of nine. To determine these rankings, each customer need is assigned an importance weight factor from one to five with five being the most important and one the least important. The table is set up to ascertain how each engineering characteristic affects each customer requirement with values zero, one, three, and nine representing a no, low, medium, and high correlation, respectively. From the results, it is concluded that the Boolean metric ranks the highest, and the weight of the system ranks the lowest.

4.4 Outcomes

The client's needs were taken and evaluated with the corresponding metrics and units using both research and guidance from the project sponsors. After creating the two different tables above, a clearer insight into the product's specific requirements was more readily attainable. Aside from the

Boolean metric, which is very generic, it was found that the two most essential engineering characteristics were “the radius and brightness of the melting pool” and “processing time”. It was also found that the “weight of the system” was the least necessary characteristic. The client requirement that included the most engineering characteristics was the “the code must be able to find the melt pool and outline the print area” with four correlations. The client requirement that included the least engineering characteristics was the “the system must have a repository for explaining how the system works” with one correlation. It was also determined the client's need with the most critical target and fallback specification is the “need to detect melt pool (brightness)” and “code must execute fast” which is appropriate, considering the two needs are the ultimate goal of the project.

V. Functional Concept

5.1 Introduction

The purpose of this chapter is to identify all functions of the vision tracking system. Research was carried out on a variety of sources to develop a list of functions. From this list a functional decomposition diagram was generated. A concept combination table was created by researching and ideating a multitude of ways to execute each function. The Concept Combination Table was then utilized to design five concepts that are applicable solutions for the product. These concepts were analyzed and voted on to narrow down the options. These tools were beneficial for understanding the problems that were faced and to develop a product that fulfils the sponsor's mission by choosing multiple solutions to answer these problems.

5.2 Functional Decomposition

A functional decomposition breaks down the product mission into detailed functions that describe the problems that must be solved to build a successful vision tracking system. The functions must consist of a verb and noun that answers the question, "what must this product do," to fulfill the product mission. Below is the detailed list of functions gathered for the functional decomposition.

List of Functions

1. Locate Appropriate Camera Distance and Placement
2. Move Cameras into Position
3. Stabilize Camera System
4. Protect Camera from Heat, Molten Metal Sputters and Metal Powder
5. Operator determines Best Camera Settings
6. Set Camera Settings for Optimal Image Quality
    - Set Camera to Correct Light Spectrum
    - Set Resolution to Meet Image Quality Needs

- Set the Camera Frames Per Second to Meet Needs of Speed of the Printer

7. Accept External Energy from Power Source
8. Power Controller, Display, and Camera
9. Boot Real Time Operating System Initiating Tracking Program
10. User Inputs Command to Begin Tracking
11. Collect, Transfer, and Store/Buffer Images of Melting Pool from Camera to Data Acquisition System or Computer
12. Process the Images
13. Measure Radius and Luminance of Melting Pool

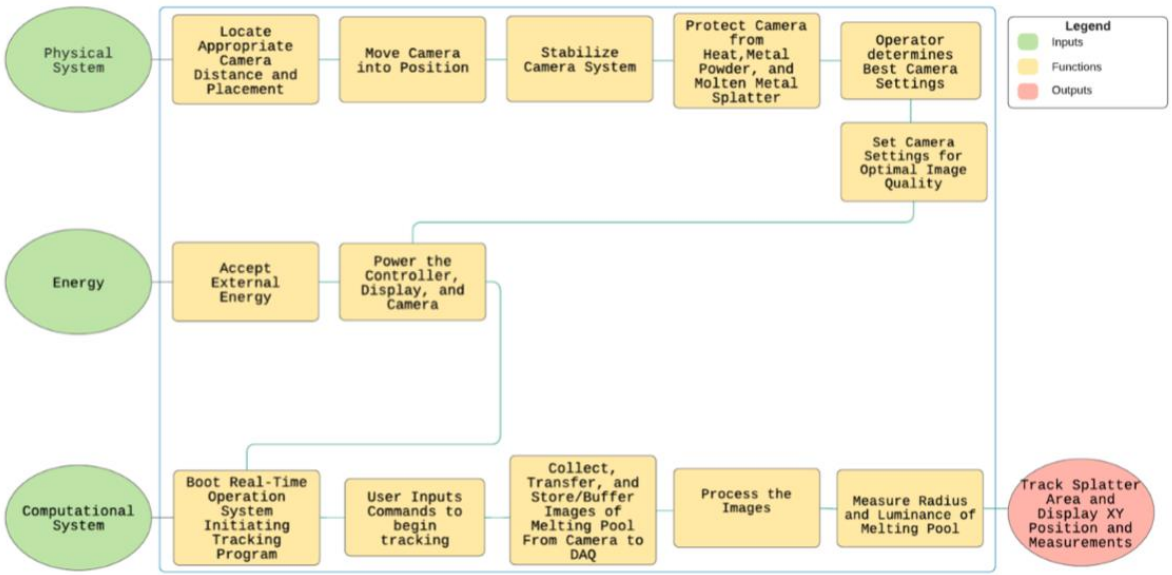


Figure 1: Function decomposition diagram

The diagram [Figure 1] functions as a visual representation of the required actions the working product must implement to produce the desired outcome. The diagram consists of inputs, functions, and outputs, flows primarily from left to right then secondarily from top to bottom, and includes a color-coded legend for clarity. The inputs are the essential ingredients needed to build the vision tracking system and the output(s) is the desired result of the fully functioning product. The completed functional decomposition diagram can then be used to create a concept combination table.

5.2 Concept Combinations

The Concept Combination Table (CCT) is a table that was generated using the functions expressed in the Functional Decomposition Diagram. To formulate creative options, several brainstorming and research sessions took place. In Table 7, each column of the CCT consists of a heading that describes the function, and rows below containing all the feasible options that could carry out the function.

Table 7: Concept Combination Table

Concept Combination Table													
Functions	Locate Appropriate Camera Distance and Placement	Move Camera into Position	Stabilize Camera System	Protect Camera from Heat, Metal Powder, and Molten Metal Splatter	Set Camera Setting for Optimal Image Quality	Accept External Energy	Power the Controller, Display, and Camera	Boot the Real-time System	Collect, Transfer and Store Images of Melting Pool from Camera to DAQ	Process Images	Measure Melting Pool Parameters	Tracking the Melting Pool	Display
Options	Excel File	Manually	3D printed part	O-ring around the camera lens between camera and holder	Focus Dials	Wall outlet	Cords	Manually (start button)	HDD	Python	OpenCV function: calculate intersection()	Heat-seeking camera	Website
	Web tools	Robotic Arm	Wired System	Keep outside of printing area	preset	Battery	USB	Light-sensor activation	SD	C++	Counting the melting pool/splatter pixels	Bright light-seeking camera	TV monitor
	User manuals	Robot dog	Manually holding	Pressurized System	Software	Solar power	Ethernet	Motion activation	SSD	Matlab	Discrete Greens Theorem	Pairing the cords of the 3D printer code with the camera code	Laptop
	Software	Mirrors	Tripod	Fan	Changing lenses	Nuclear	Wireless (built-in battery)	Servo robotic arm	RAM	Assembly	Brightness = pixel level 0-255		Phone app
			robotic hand		Focus Rings				Cloud	OpenCV	Infrared Camera		Audio Speaker w Text to Speech Application
					Polarization filter						Luminances Meter		

Certain functions have only one logical option to fulfill it; however, additional options were added for the sake of brainstorming and generating ideas. Functions such as “Boot the real-time system” and “Move camera into position” would be done by hand. In principle, accumulating these ideas assisted with future design concepts.

5. Concept Generation:

Concepts were generated by assigning one or more options from the Concept Combination Table to each function and assembling the selected options into a viable concept. Each team member assembled four concepts each totaling sixteen concepts, and the top five concepts were chosen by vote. The following figures are illustrations of the chosen concepts along with tables listing the technologies used for each function and brief explanations of how the concepts work.

Table 8: Robot dog concept

Concept 1: Robot Dog Setup													
Functions	**Locate Appropriate Camera Distance and Placement**	**Move Camera into Position**	**Stabilize Camera System**	**Protect Camera from Heat, Metal Powder, and Molten Metal Splatter**	**Set Camera Setting for Optimal Image Quality**	**Accept External Energy**	**Power the Controller, Display, and Camera**	**Boot the Real-time System**	**Collect, Transfer and Store Images of Melting Pool from Camera to DAQ**	**Process Images**	**Measure Melting Pool Parameters**	**Tracking the Melting Pool**	**Display**
Options	Software	Robot Dog/ Robotic Arm	Robotic Hand	O-ring around the camera lens between camera and holder	Software	Solar Power	Wireless (Built-in Battery)	Light Senosr Activation	HDD	C++	OpenCV function: calculate intersection()/ Luminance Meter	Bright light-seeking camera	Audio Speaker w/ Text to Speech Application

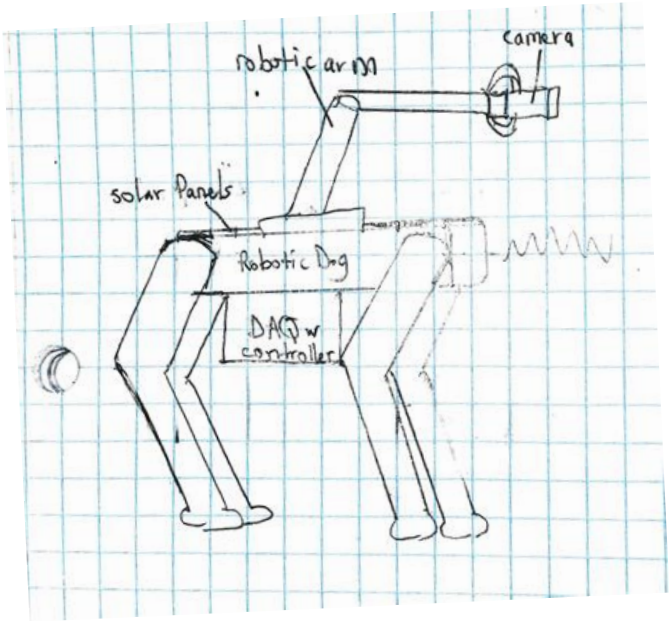


Figure 2: Robot dog concept hand drawing

In Figure 2, the Robot Dog concept uses a Boston Dynamic type robotic dog that is powered by solar energy using solar panels placed along its back. The energy is stored in a built-in battery that also powers the controller and camera. A robotic arm with a hand is attached to the top of the robotic dog to hold, stabilize, and move the camera. A bright light-sensing camera and C ++ software locate where to place the camera, sets camera settings, and processes the images. The OpenCV function, calculate intersection(),a luminance meter records measurement, and all data is stored on a hard drive disk. Desired output data is delivered via audio speaker using text to speech application. Although the robot dog would be a creative idea, it would also would very expensive, and solar power would not be effective indoors. Table 8 shows the technology chosen for each function.

Table 9: Python application on laptop with SSK and heat-seeking camera concept

Concept 2: Laptop w/ SSD and Python Code Setup													
Functions	**Locate Appropriate Camera Distance and Placement**	**Move Camera into Position**	**Stabilize Camera System**	**Protect Camera from Heat, Metal Powder, and Molten Metal Splatter**	**Set Camera Setting for Optimal Image Quality**	**Accept External Energy**	**Power the Controller, Display, and Camera**	**Boot the Real-time System**	**Collect, Transfer and Store Images of Melting Pool from Camera to DAQ**	**Process Images**	**Measure Melting Pool Parameters**	**Tracking the Melting Pool**	**Display**
Options	Web Tools	Manually	3D Printed Mount	O-ring around the camera lens between camera and holder	Focus Rings	Wall Outlet	USB	Manually (start button)	SSD	Python	OpenCV function: calculate intersection()/ Counting the melting pool/splatter pixels	Heat-seeking camera	Laptop

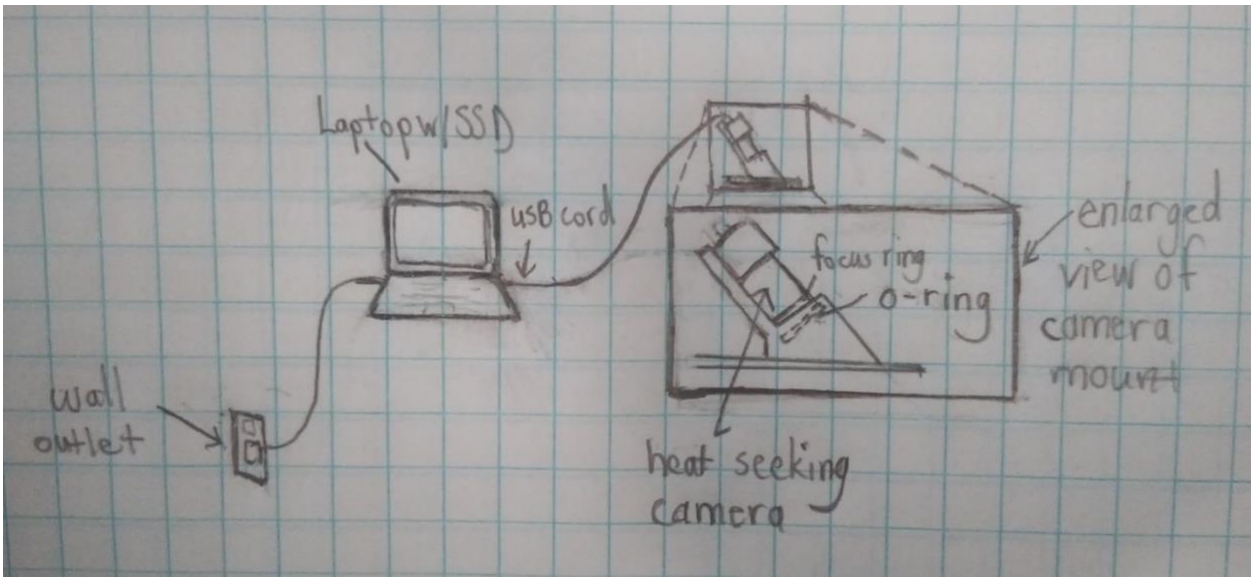


Figure 3: Laptop with SSD and Python concept

In Figure 3, a Python application that tracks the melting pool using a heat-seeking camera will be executed on a laptop with SSD. The laptop is powered by a wall outlet and a USB cord powers the camera and enables the transfer of data from the camera to the laptop. Focus rings are manually added to focus the camera and a 3D printed camera holder with an O-ring stabilizes and protects the camera. The laptop processes the data and displays the measurements on the screen. This concept would be one of the most cost-efficient ideas and is the least complicated which made it an ideal solution. Table 9 shows the technology chosen for each function.

Table 10: Robot arm with camera concept

Concept 3: Robot Arm with Camera													
Functions	**Locate Appropriate Camera Distance and Placement**	**Move Camera into Position**	**Stabilize Camera System**	**Protect Camera from Heat, Metal Powder, and Molten Metal Splatter**	**Set Camera Setting for Optimal Image Quality**	**Accept External Energy**	**Power the Controller, Display, and Camera**	**Boot the Real-time System**	**Collect, Transfer and Store Images of Melting Pool from Camera to DAQ**	**Process Images**	**Measure Melting Pool Parameters**	**Tracking the Melting Pool**	**Display**
Options	Software	Modified SCARA	Robotic Hand with Tilt Compensation	O-ring around the camera lens between camera and 3D printed housing	Software	Outlet	Outlet and 5V USB	Push Button	Native Cloud	Python	OpenCV function: calculate intersection() / Luminance Meter	Bright light-seeking camera	GUI displayed on monitor

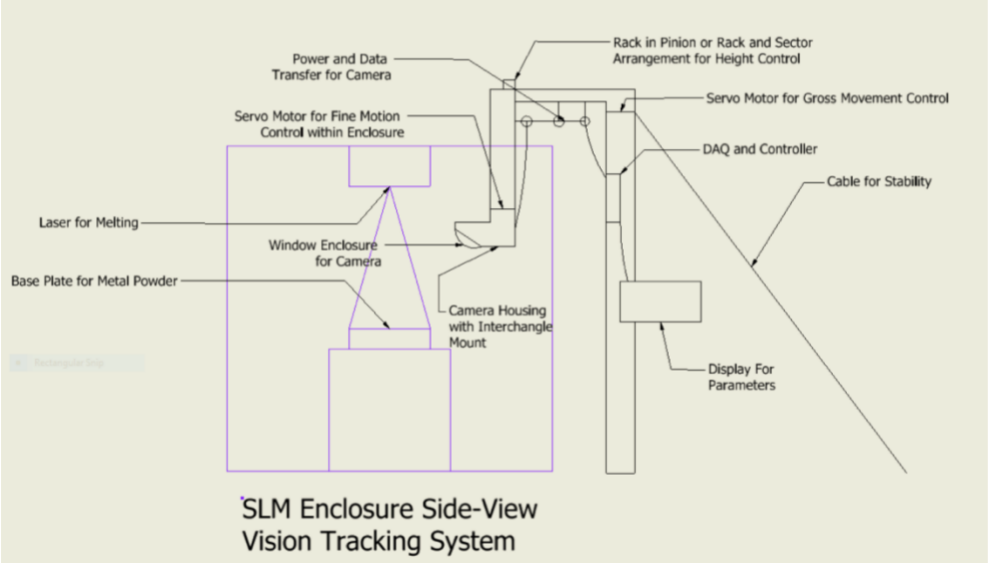


Figure 4: SCARA variant concept

In Figure 4, a robotic arm controls the positions of the camera within the enclosure. The robot arm is designed to lift outside of the enclosure to allow for easy replacement of the cameras as needed. The software takes into consideration the camera lens to determine the height needed for optimal optics. To deal with the camera shaking during movements, a cable is added to absorb some of the vibrations. A seal is formed with an O-ring and a 3D printed camera housing to protect the cameras from harm. Power is supplied to the display, controller, and motors from a 120 Volt outlet. The controller then powers the camera system. Once powered, the

image will be processed in a real-time operating system running Python code. This code tracks luminance to determine the melt pool position and splatter patterns.

This design is good because it can manipulate the camera into better angles than the current mounting setup. The overall setup is inexpensive. The pole could be made of PVC or wood and the motors harvested from equipment at a state surplus. The rest of it could be 3D printed. Unfortunately, the complexity of the design would add more work to the project and add cost to Dr. Downey. Additionally, it would need a secondary controller for the motor system. Table 10 shows the technology chosen for each function.

Table 11:Manual control of system

Concept 4: Manual control of system													
Functions	**Locate Appropriate Camera Distance and Placement**	**Move Camera into Position**	**Stabilize Camera System**	**Protect Camera from Heat, Metal Powder, and Molten Metal Splatter**	**Set Camera Setting for Optimal Image Quality**	**Accept External Energy**	**Power the Controller, Display, and Camera**	**Boot the Real-time System**	**Collect, Transfer and Store Images of Melting Pool from Camera to DAQ**	**Process Images**	**Measure Melting Pool Parameters**	**Tracking the Melting Pool**	**Display**
Options	Excel File	Manually	Manually holding	Fan	Focus Dials	Battery	USB	Manually (start button)	RAM	Matlab	Counting the melting pool/splatter pixels	Heat-seeking camera	TV monitor

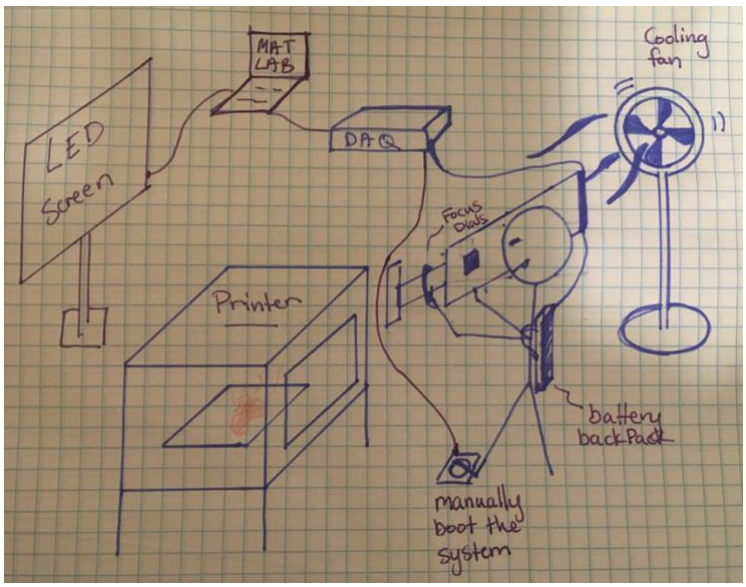


Figure 5: Manual control of system

In Figure 5, the system is controlled primarily by the user. The camera is manually adjusted using focus dials, and the user is holding the camera in the necessary location. The camera is a heat-seeking camera therefore the printer has an open area to let heat out. However, the user and the camera should not get overheated, so a fan is placed in the vicinity for cooling. The user is also wearing a battery pack that is powering the system. Data is displayed on the TV screen. This option is optimal for being able to control the system to the users' liking, however, that means there must always be someone there during use. Table 11 shows the technology chosen for each function.

Table 12: Remotely viewed data

Concept 5: Remotely viewed data													
Functions	**Locate Appropriate Camera Distance and Placement**	**Move Camera into Position**	**Stabilize Camera System**	**Camera from Heat, Metal Powder, and Molten Metal**	**Set Camera Setting for Optimal Image Quality**	**Accept External Energy**	**Power the Controller, Display, and Camera**	**Boot the Real-time System**	**Collect, Transfer and Store Images of Melting Pool from Camera to DAQ**	**Process Images**	**Measure Melting Pool Parameters**	**Tracking the Melting Pool**	**Display**
Options	User manuals	Manually	Wired System	Keep outside printing area	Preset	Solar Powered	USB	Servo robotic arm	SSD	Python	Counting the melting pool/splatter pixels	Heat-seeking camera	Phone App

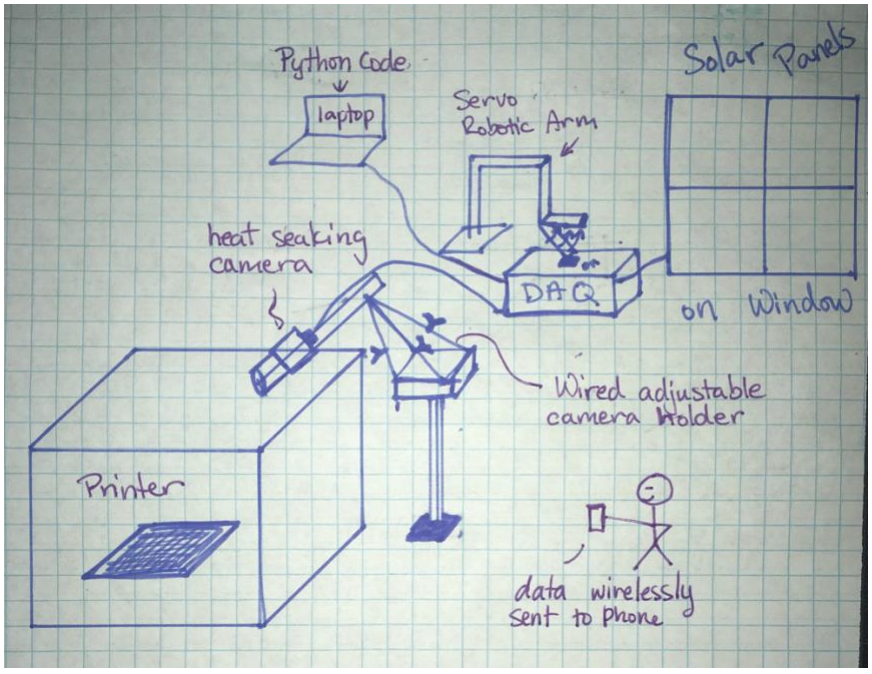


Figure 6: Remotely viewed data

In Figure 6, the camera is set up outside the printing area and held in place using an adjustable wired setup. The system is booted using a remote-controlled servo arm and everything is powered by solar panels attached to a nearby window. The data is configured using python code and then wirelessly sent to the user's phone via a cellphone application. The wired setup is convenient since it can be adjusted to fit the apparatus, however, having data sent wirelessly to a phone app gives opportunities for security violations. Table 12 shows the technology chosen for each function.

5.3 Research performed

Patents were searched for under a couple of websites - Google patents [7] and United States Patent and Trademark Office [8]. Keywords include the following:

- Metal 3-dimensional printer monitoring
- 3-dimensional printer monitoring
- 3-dimensional printer tracking
- Melting pool tracking
- Melting pool monitoring
- Monitoring of laser powder bed
- Selective laser melting
- Melting pool measurement
- Melting pool parameters
- Metal powder bed monitoring
-

There were few relevant patents to consider that were found using the above keywords and that are cited in the appendix of this report. The first being the "Method and System for Monitoring Additive Manufacturing Processes" [9] which is an invention that teaches quality assurance for additive manufacturing using a multi-sensor and a real-time quality system. Similarities include using multiple optical sensors to monitor the powder bed and using a real-time operating system.

Another patent that is related to the project is one titled "Printer monitoring" [10] and is a system that monitors 3D printers using a baseline image to compare to future images taken to

determine the status of the print job and alert the user of any failures. This patent is not, however, used with metal 3D printers, but collecting images to track data is similar to the project.

The last patent that was found that pertained to this project is the invention titled "Procedure and apparatus for in-situ monitoring and feedback control of selective laser powder processing" [11] which monitors and controls the Selective Laser Powder Processing by using a scanning device to follow the laser beam as well as a detector that can capture electromagnetic radiation emitted from movement on the powder surface. It also consists of an optical system to follow the laser beam.

The idea of using a camera-in-the-loop system for detecting errors during an SLM (Selective Laser Melting) process is a new idea that has been explored mostly by other universities and research groups. As such many of the ideas rely heavily upon other journals of other research groups.

The research article "Machine-Learning-Based Monitoring of Laser Powder Bed Fusion" [12] was very constructive for the concept generation. This article is like what is being accomplished in the project and has helped generate quite a few new options for the functions. One idea extracted from this article is the use of angled mirrors to get a top-down view of the metal powder bed [Figure 7].

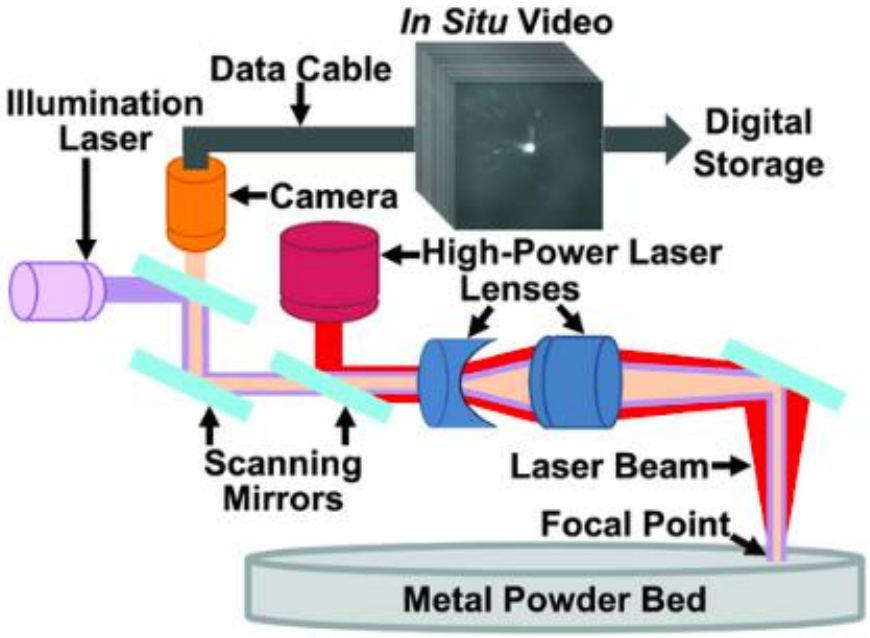


Figure 7: SLM video process

This idea can drastically simplify the project in two ways. With the use of mirrors, there is no concern about varying camera angles and positioning; effectively eliminating two of the functions (1. Locate Appropriate Camera Distance and Placement and 2. Move Cameras into Position).

One important need of the sponsor is the software aspect. The product needs to be able to track the 3D printing process and display the melting pool parameters all in real-time. This can be difficult to code completely from scratch, so other ways to simplify this process were researched. One way that was found was to use the OpenCV library [13]. Open Computer Vision, "is an open-source computer vision and machine learning software library... that provides a real-time optimized Computer Vision library, tools, and hardware [13]." OpenCV helped generate options for two of the functions. The first function "12. Process the Images," is done to extract useful information from an image by modifying it. This can be accomplished almost

exclusively by OpenCV functions. The image can be processed by multiple functions like cvtColor(), GaussianBlur(), threshold(), erode(), and dilate(). All of which help the code find the parameters of the melting pool. The second function OpenCV helped generate options for was, "13. Measure Radius and Luminance of Melting Pool." As shown in Figure 8, OpenCV has a handy function called grab_contours(), which can draw a border around the melting pool.

Figure 8: Contour around the melting pool

Once this boundary is set, the melting pool's parameters (centroid location, area, and radius) can be calculated.

7. Conclusions:

All the designs share the project sponsor's requirement that currently owned cameras were used. This was important because there was a limited budget. Due to the nature of the printer, it was important that the cameras were protected from heat and molten metal splatters. One similarity between designs was that the camera was either outside of the printer enclosure or protected with a casing if it was inside the enclosure.

One design limitation considered was that it needed to minimize the amount of human interaction or manual work because there was not a budget for labor. Furthermore, there was limited knowledge of robotics systems among the team working on the project so avoiding complex solutions was necessary. Additionally, security concerns were considered because of

the potential for confidential images of prototypes being made. The most complex functions to solve were processing images or collecting and transferring images. This was because the coding element for processing the images was successfully achieved in Python but needed to be implemented in real-time.

VI. Concept Selection

6.1 Introduction

Project design is a strenuous and complex process for any engineering project. To alleviate the difficulty of design selection, a variety of brainstorming methods were employed which produced a multitude of designs. The brainstorming process began by listing the sponsor's needs to establish the desired parameters and a budget limit. Then the required functions of the final design were listed, and a Functional Decomposition Diagram was constructed to help illustrate how the design would work. For each of the functions, an extensive number of options were produced. From these options, seven different designs were conceptualized enabling the initiation of the Concept Design Review process. The first part of the process, Concept Screening, involved narrowing down the concepts and deciding which ones could be combined to create an optimal design concept. After narrowing down the seven concepts to three, the second part of the process, Concept Ranking, began. Each of the three concepts were ranked by weighing the sponsor's needs to determine if the concepts could satisfy the design functions.

6.2 Concept Screening

Table 13: Initial concept screening

Criteria	Concepts						
	#1	#2 (reference)	#3	#4	#5	#6	#7
Setup	0	0	-	-	-	+	-
Budget	-	0	-	-	-	+	-
Ease of use	+	0	+	-	0	+	-
Complexity of design	-	0	-	0	-	+	-
Camera placement (accuracy)	+	0	+	+	0	-	+
Data ingestion	0	0	+	0	0	-	0
Ability to display data	-	0	0	0	+	+	0
Ability to protect camera	0	0	+	+	0	-	-
DAQ speed	0	0	0	0	0	-	0
Parameter tracking capablilities	0	0	0	0	0	0	+
Camera type flexibility	-	0	+	+	0	+	+
Camera stability	0	0	0	0	0	-	+
sum +'s	2	0	5	3	1	6	4
sum 0's	6	12	4	6	8	1	3
sum -'s	4	0	3	3	3	5	5
Net Score	-2	0	2	0	-2	1	-1
Rank	7	4	1	3	6	2	5
Continue?	no	yes	combine	no	no	combine	yes

The selection criteria used for the concept screening were created and later updated after meeting with the project sponsor. The criteria were adapted from the sponsor's needs list as well as the concepts that were designed. The concept screening [Table 13] was done by comparing the reference concept, the original design endorsed by the sponsor, with the six novel concept designs. The following symbols were used to rank the designs: a negative sign '-' to represent the design's failure to meet the criteria as sufficiently as the reference design, a positive sign '+' to represent the design successfully exceeding the reference design in meeting the criteria, and a zero '0' to represent the design matching the reference design's ability to meet the criteria. The reference design was given a zero for each of the criteria to provide a base level. The purpose of using this type of concept screening was to rule out any designs that were not actually feasible and to give the team an opportunity to take positive concepts from designs that didn't rank very high as a whole and piece them together to create a new combined design. The outcome of this

concept screening took the team from the original 7 designs to 3 designs that include a combination design.

6.3 Concept Ranking

Table 14: Final concept ranking

Selection Criteria	Weight (%)	Concepts					
		Concept #6 (Mobile design)		Concepts #2 (Reference)		Concept #8 combined (Mobile, python, robot arm)	
		Rating	Weighted Score	Rating	Weighted Score	Rating	Weighted Score
Setup	1.2604	7	0.0882	5	0.0630	2	0.0252
Budget	23.1801	2	0.4636	5	1.1590	3	0.6954
Ease of use	1.7485	7	0.1224	5	0.0874	3	0.0525
Complexity of design	2.4824	7	0.1738	5	0.1241	2	0.0496
Camera placement (accuracy)	7.4347	3	0.2230	5	0.3717	8	0.5948
Data ingestion	12.1424	3	0.3643	5	0.6071	5	0.6071
Ability to display data	3.1587	8	0.2527	5	0.1579	8	0.2527
Ability to protect camera	14.4580	3	0.4337	5	0.7229	5	0.7229
DAQ speed	13.1006	5	0.6550	5	0.6550	5	0.6550
Parameter tracking capablilities	7.3492	3	0.2205	5	0.3675	5	0.3675
Camera type flexibility	2.8655	1	0.0287	5	0.1433	5	0.1433
Camera stability	10.8195	2	0.2164	5	0.5410	6	0.6492
SUM			3.2423		5		4.8151
Rank			3		1		2
Continue?			no		develop		no

The concept ranking [Table 14] took place after determining the weights of the selection criteria, a process that can be seen in the appendix. The criteria were compared to each other and given a score of 1, 3, 5, 7, or 9 with a ranking of 1 having the lowest level of importance and 9 being much more important. Concept #2 (the reference concept) was the final selected concept with the highest ranking. This was mainly due to the low budget available, and the simple setup required. However, because concept #8 ranked so closely with the reference concept, a discussion with the sponsor took place.

6.4 Final Concept

The final concept selected was concept #2, the reference concept, which is a python application that tracks the melting pool splatter using an optical camera and executes on a laptop

with a solid-state drive. Powered from a wall outlet, the laptop is then connected to the heat-seeking camera via USB cable to power the camera and transfer data from the camera to the laptop as seen in Figure 9.

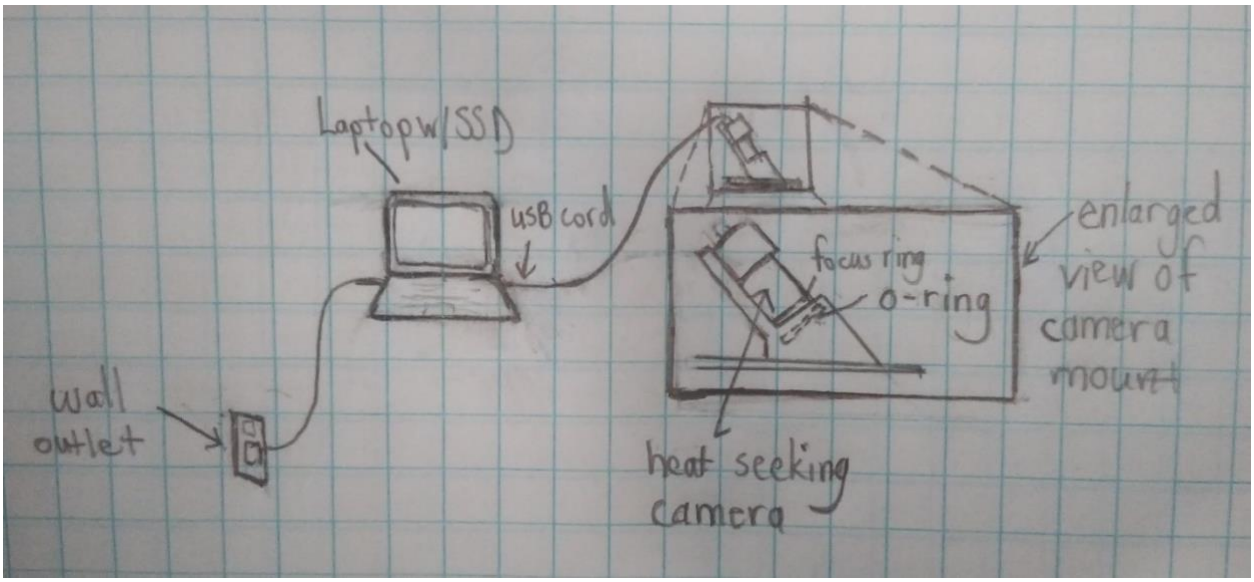


Figure 9: Laptop with Solid State Drive and Python Concept

The camera is held and protected by a 3D-printed camera mount with an O-ring and focused on the printing bed area using focus rings as needed. After the laptop processes the data, the data is displayed on the screen. Challenges with this concept included designing a camera mount that can be adjusted for cameras of various sizes and shapes or capable of holding multiple cameras that can gather data synchronously. Finding a DAQ with the ability to process data from multiple cameras in real-time proved to be challenging as well. However, using different types of cameras simultaneously to find a greater diversity of data could be advantageous. Furthermore, just gathering data in real-time was a difficult task. As the development of the project began, the final design was bound to change. However, these changes were nothing but a small speed bump in the development process due to the work put into the Concept Design Process.

VII. Product Architecture

7.1 Introduction

Creating the project architecture report is a process that is used to break down the functional elements into groups or clusters that can help define the systems of the project. The benefit of this process is it introduced new elements that were not previously considered. Two new clusters were added to the functional decomposition diagram which included the digital and processing aspects of the system.

7.2 Product Layout

The layout below in Figure 10 consists of five clusters total. The five clusters are the Camera Unit, Control Unit, Digital System, C++ Code, and User HMI (Human Machine Interface). The first cluster, the Camera Unit, contains all functions pertaining to the physical setup, positioning, and protection of the camera and camera mount. The second cluster, the Control Unit, contains all functions needed to operate the real-time vision tracking system such as the power supply, real-time operating system, all data flow functions, and tracking code. The third cluster is the Digital System which is embedded within the Control Unit cluster. This cluster demonstrates how the data flows from the camera to be stored, is processed, and ran through the code to obtain the desired splatter measurements. The fourth cluster, the C++ Code, describes how the code will analyze the received data to calculate the area, radius, and brightness of the splatter. This cluster is embedded within the Digital Systems cluster. The last cluster is the User HMI which contains all functions that require human interaction such as manually setting camera settings, inputting commands to the operating system, and receiving output data.

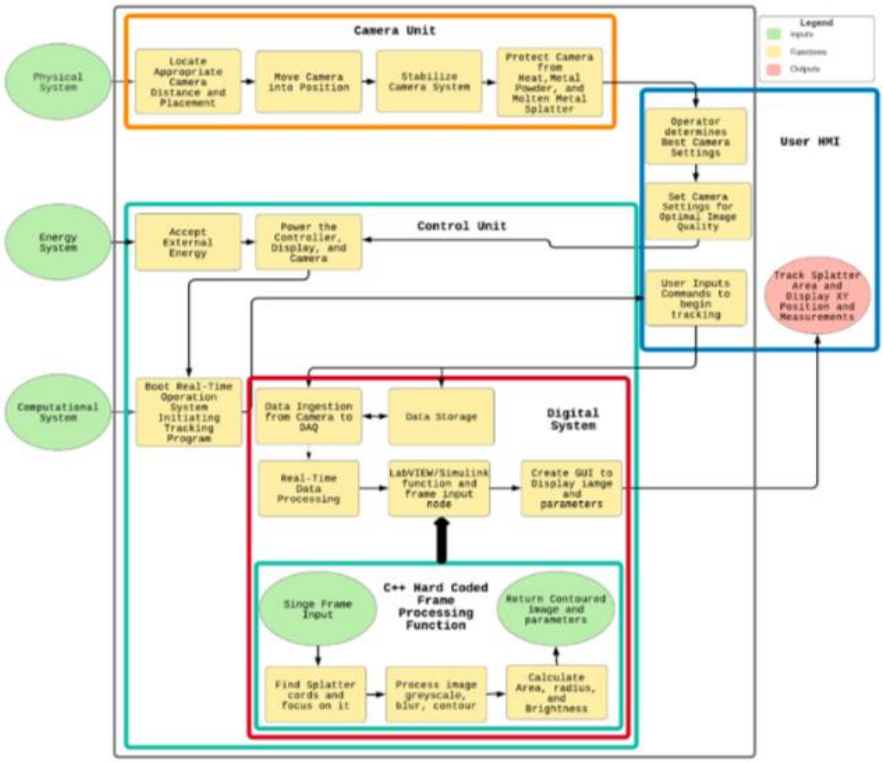


Figure 10: Real-time vision tracking system functional decomposition diagram with clustered functions

7.3 Interactions

Below is a list of interactions or interruptions that could occur while or before the system is running. After the interaction is stated there is a proposed solution to avoid the problems that could occur.

a. The operating system interrupts the process for the code to collect data.
   - Run a real-time operating system to limit the number of OS interrupts

b. Stack overflow
   - More RAM
   - Faster and Deterministic processing
   - Real-time or Stream processing

c. Non-optimal drivers' compatibility could cause slower communication speed or prevent the camera from communicating with the computer system.
   - Build it or find it

d. Vibrations of someone walking near the camera could interrupt the accuracy of the tracking.

- Keep people away
- Add a soft rubber mount to absorb vibrations (Also makes it airtight)

e. Camera distance consistency could cause out-of-focus pictures and an inconsistent field of vision in the area of interest, which will affect the ability to convert pixels into area consistently.
   - Always mount flush

f. Frame losses could result in inaccurate tracking.
   - Better network protocols
   - Better Drivers
   - Better processors
   - More RAM

g. Bandpass lets too many wavelengths in like a laser which won't let the user see the melt pool or other areas of interest.
   - Review bandpass characteristics
   - Buy more

h. The heat bed gives off infrared radiation which could affect picture quality.
   - Avoid thermal cameras
   - Avoid long wavelengths in bandpass filters

i. Code Speed could cause inadequate frames per second which would cause significant loss in data.
   - Faster processor
   - Switching from Python to C++
   - Changing where information is stored and called
   - Change structure and functions used

j. Slow bus speed will overwhelm the system and cause more data loss.
   - Purchase a faster computer of at least PCI 2.0

7.4 Geometric Layout

For the geometric layout, a CAD model [Figure 11] was created to represent how the physical system is laid out. Below, a metal 3D printer is used in conjunction with the already manufactured camera holder in orange that fits securely into the upper window on the printer's frame. The camera is shown in the camera holder and will be sealed with an O-ring. The camera will be connected to the controller, and the controller will be connected to a laptop. In Figure 12, the graphical model is shown with the corresponding data flow.

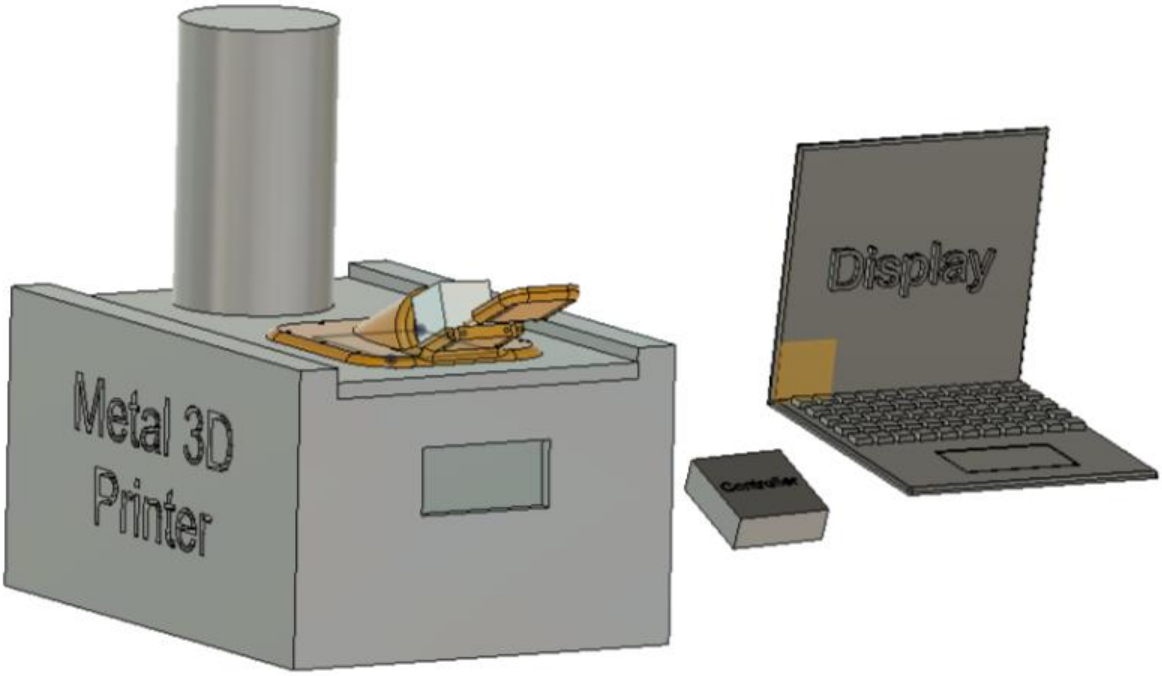


Figure 11: 3D CAD model representing the physical layout of the Real-Time Vision Tracking system setup

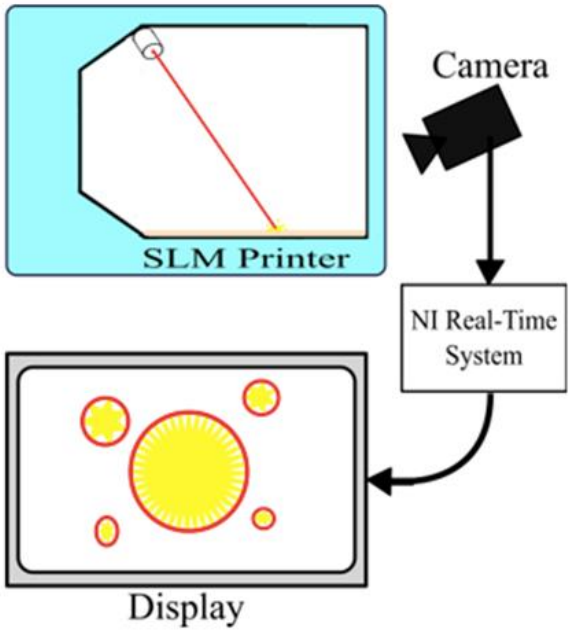


Figure 12: Graphical model representing data flow of the system

VIII. Engineering Analysis

8.1 Introduction

Team Downey was asked to create and implement a vision tracking system for a selective laser melting (SLM) printer in the McNair Center. Ensuring that the system has the fewest errors possible is imperative to allow the sponsor to move forward to the next steps of the project. The issues that have the potential to get in the way of reaching this target include not reaching the real-time goal, inaccurate sputter parameter data, an unclear picture produced from the camera, and a deformed camera housing from excessive heat exposure. The camera housing unit will be evaluated using thermal analysis via Fusion 360 and with the known properties of the PLA used to print the part. Ensuring the real-time goal is reached can be done by evaluating the computation time inside the code. The optical issues are avoided with the proper filter and the angle that the camera sits at. These parameters are calculated using the focal length and field of view equations, and the wavelength selection.

8.2 Analyses Performed

Analysis #1 - Thermal

Analysis performed:

The thermal analysis of the camera housing was performed using Fusion 360 a CAD software. With a laser temperature of 500 °C, the SLM process might seem too extreme to use a PLA 3D printed part as a camera mount. However, the measured temperature at the point of contact between the camera housing and the metal 3D printer averaged only 38 °C when tested. This information was then put into a CAD thermal simulation for analysis.

For thermal analysis of PLA material, the melting point temperature is not a principal factor because deformation of the material begins prior to reaching its melting point. It is the glass transition temperature of PLA, which is the point at which the storage modulus begins to decrease [Figure 13].

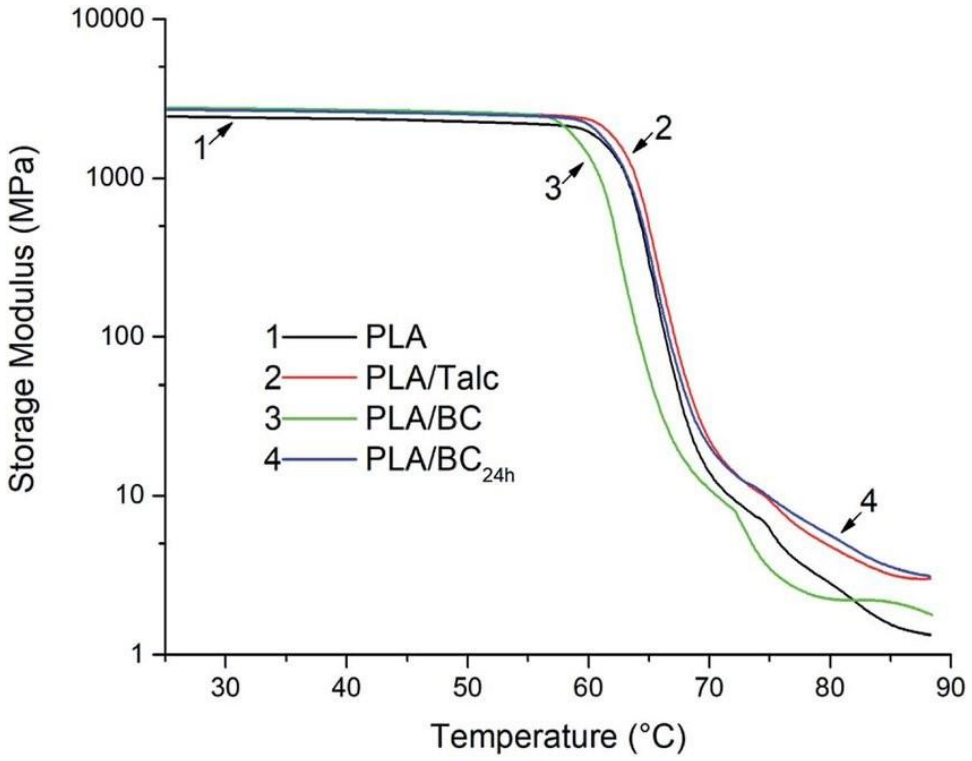


Figure 13: Temperature vs. Storage Modulus of PLA

The storage modulus measures the elastic response of a material. During this phase, the amorphous material begins to lose rigidity as it transitions from a solid and slightly brittle state into a rubbery and glass state. The glass transition temperature, the temperature at which PLA will begin to deform, is between 50°C and 80°C. Therefore, the temperature in contact with the camera mount should stay below 50 °C to ensure the camera will stay in the intended place and undamaged.

Modeling/Analysis results:

Below in Figure 14, the temperature is analyzed throughout the camera housing. With the bottom face of the object receiving an applied thermal temperature of 38°C and the top faces being exposed to room temperature of 20°C, the maximum temperature is 37.8 °C, and the minimum temperature 31.58°C.

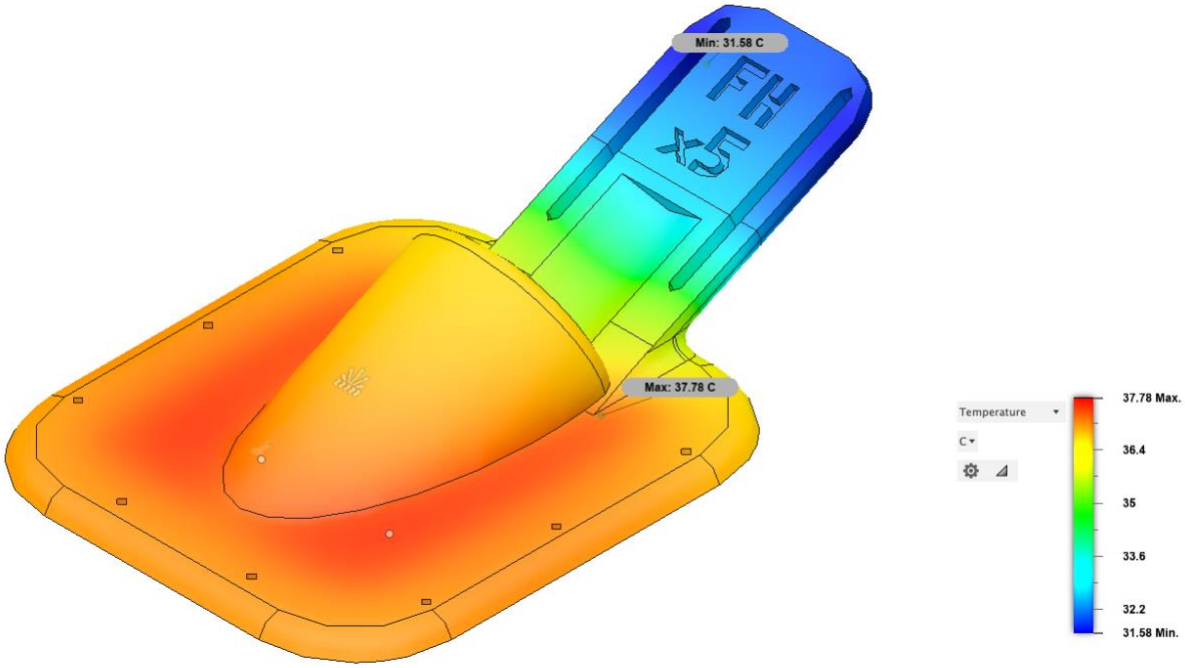


Figure 14: Thermal analysis of camera housing using Fusion 360

Analysis #2 – Computation Time

Analysis performed:

The most essential part of the system is that it must be implemented in real-time. In software terms real-time is defined as, “Real-time programs must guarantee a response within specified time constraints, often referred to as "deadlines".[20]” To accomplish this goal, all the digital parts of the system need to work on the same timing schedule to ensure that there is no buffering in the output display and data collection. There are two types of deadlines in real-time computing, hard and soft. In soft deadlines, if a deadline is missed the system will not fail and the result can still be used. However, the longer the system runs the bigger the buffer becomes. For example, if the end-user were watching the output video feed thirty (30) minutes after the start of the system they could only

see what was happening five (5) minutes ago which would make a soft deadline system not useful. In hard deadlines, the system will fail, and no data can be collected if the deadline is missed. This real-time system adheres to a hard deadline because the input data comes from a video feed that can go on for hours.

$$\frac{30\ FPS}{1\ sec} \approx 30ms \qquad \textit{Equation 1}$$

To ensure the software meets the hard deadline, the computation time of the program is matched to the video input frames per second (fps). [Equation 1] shows the approximated deadline to process each frame as it comes in. To determine if the code meets that deadline, two different timing functions will be used for each language the program is currently coded in. "Linux time" will be used for the C++ version, and "time.process_time()" will be used for the Python version since both find the total user CPU time of the code.

Modeling/Analysis results:

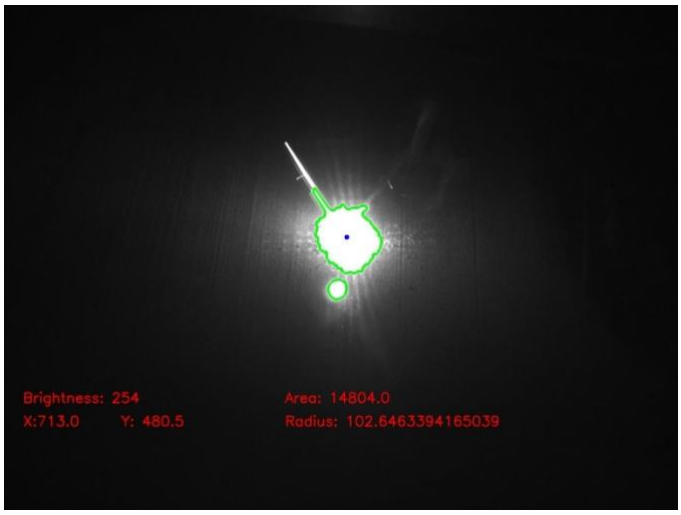


Figure 15: Test Frame

The user CPU time for the C++ program to process one frame of the video was 15 ms which met the hard deadline with ease. The user CPU time for the Python program to process one frame of the video was 28.6 ms. This is cutting it fairly close to the proposed deadline of

30ms and could potentially cause problems in the future. In the test frame used for each program [Figure 15] there are two contours that were found, one for the melting pool and a smaller one for the sputter. If the program processed a frame with multiple sputters, the processing time would increase and potentially lead to a system failure. This factor suggests the Python version will not suffice and implementing the C++ code should be initialized instead.

Analysis #3 - Optics

Analysis Performed – Thin Lens Equation:

Based on the CS-Mount information provided the distance from the sensor to the lens is 12.5mm. This is used in conjunction with the assumed distance to the print to find the focal length. This is an important characteristic for optical properties.

$$\frac{1}{f} = \frac{1}{u} + \frac{1}{v} \quad \textit{Equation 2}$$

Where f is focal length, u is distance from lens to sensor, v is distance from lens to object

Table 15: Focal length calculations

Assumed Distance to Print (mm)	Focal Length (mm)
50	10
75	10.71428571
100	11.11111111
125	11.36363636
150	11.53846154
175	11.66666667
200	11.76470588

Analysis Performed – Field of View Calculations

The calculations are based on the focal length, sensor dimension, and field of view. To find depth of field f-number of the camera in operation is needed which is unknown.

Table 16: Linear field of view calculations

Assumed Distance to Print (mm)	Linear Field of View (Width in mm)	Linear Field of View (Height in mm)
50	24.84	18.63
75	34.776	26.082
100	44.712	33.534
125	54.648	40.986
150	64.584	48.438
175	74.52	55.89
200	84.456	63.342

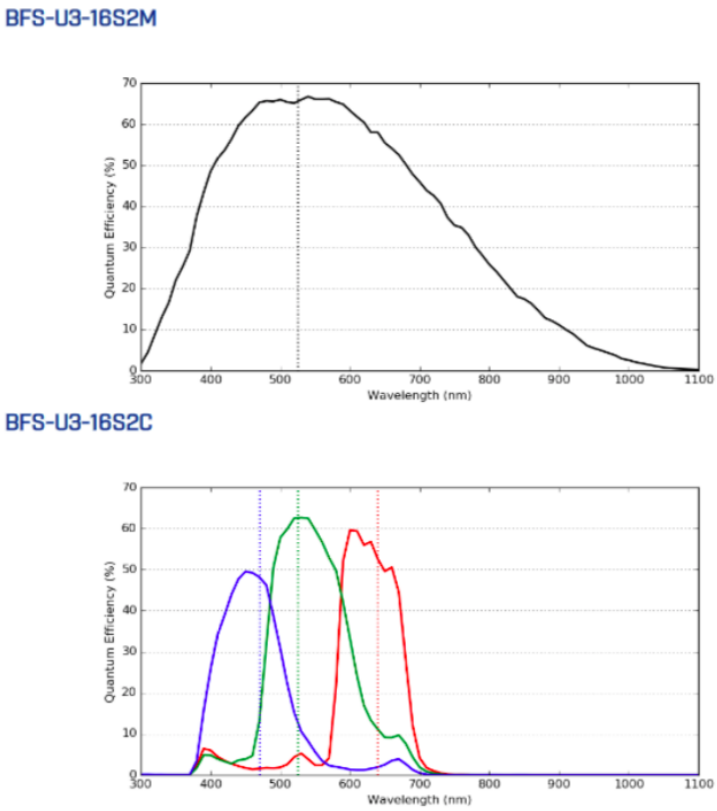


Figure 16: Optical performance of camera based on wavelength

Analysis Performed - Wien's displacement law

Derived from blackbody radiation and its maxwell distribution which is dependent on temperature.

$$wavelength\ peak = \frac{B}{T} \qquad \textit{Equation 3}$$

B – Wien's displacement constant - $2.897771955 \times 10^{-3}$ m·K
T – Temperature

Table 17: Dominate Wavelength at a given temperature

Temperature of Melted Steel (C)	Wavelength (nm)
2000	1274.783
2100	1221.066
2200	1171.693
2300	1126.157
2400	1084.029
2500	1044.939
2600	1008.57
2700	974.6471
2800	942.9322
2900	913.2162
3000	885.316

Note: This ignores the effect of plasma in the imaging process. Also, the typical temperature of the melted steel is around 3000 C.

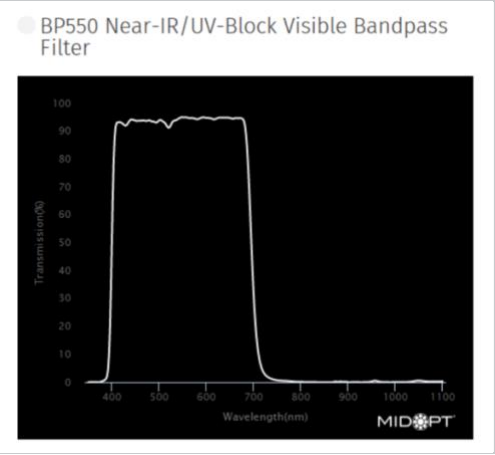


Figure 17: Bandpass filter

This filtering was selected based on a research paper that talked about the wavelength emitted by the laser during the melting process, so it eliminates the noise created by the surrounding area being heated by the melt pool. Additionally, there is an angle when viewing the print so it will experience skewing which will impact the data being collected and analyzed.

8.3 Conclusions/Recommendations

After these analyses were performed it was concluded that the camera housing will not deform due to extreme heat since the analyzed temperature stays below the glass transition temperature. It was concluded that the code would be able to run in real-time and lastly, the camera would be able to be positioned correctly with the right dimensions provided. And, from the preliminary tests Python may be risky to move forward with. Instead, the next step was to work on implementing the C++ code to meet the hard real-time deadline.

IX. Manufacturability

9.1.1 Design for X

Introduction:

After analyzing the materials and functionality of the overall system being produced, the next step is to review the design to safely manufacture and assemble the system for future use. A successful way to analyze the system for refinement is to implement the "Design for Excellence" system known as DFX. Among the given manufacturability, assembly, cost, and safety parameters, other parameters will be recognized during the design refinement. One of the added parameters is an encouraged prerequisite knowledge in the programs used in the system which will aid in the ease of future use. Design for real-time usage is implemented from previously established sponsor needs. Finally, public access is a parameter that was requested through the sponsor and is fulfilled in the new design.

Design for Manufacturability and Assembly:

The system is made of two custom designed pieces to allow visibility into the printing chamber and then a computer vision system. The first designed piece is a replacement for the standard lid that covers the print area and seals the system. The second piece is a housing unit for the camera. In terms of what is needed for each of these parts, the first must allow camera access to the print area without impacting the printing process and the latter must hold the camera stable and have a seal to prevent gas or metal powder from leaking into or out of the print chamber.

To replace the existing lid, a new plate would need to be turned into a lid with a viewing port. Two candidates for material would be steel or aluminum. Because aluminum is cheaper, lighter, easier to machine, and strength of material was not a factor, it was the obvious choice for this task. An alternative to aluminum would be any series of corrosion-resistant steel such as 316

Steel because the melting pool ejecta could solidify onto the lid and cause corrosion attacks resulting in pitting due to dissimilar metals and it is stored in a non-dehumidified room.

For creating the plate, aluminum can be cast, forged, or extruded then milled to specifications. The procurement of the plate was done by the University of South Carolina machine shop and subsequently milled. Please note, most of this was done prior to working on the project.

For the housing unit, any material would have been acceptable if it was strong enough to hold the camera, remain rigid at above room temperatures, cheap to replace and the part can be built quickly with said material. Polylactic Acid (PLA), a thermoplastic, was selected for several reasons. The first was it was readily available in the ARTs Laboratory. The second is that its glass transition temperature is around 60-80 degrees Celsius which is higher than the predicted maximum temperature the housing will be exposed to.

An alternative that would also be inexpensive and still in the realm of 3D printing would be using a lithographic-based resin printer. The resin option would allow for dealing with greater temperatures with the added benefit of greater dimensional accuracy and finish. The biggest drawbacks of this are the toxic fumes from the resin and the availability of this type of printer to the sponsor. This may be a better option should a seal be deemed necessary for the product or if PLA begins to deteriorate from exposure to the metal printing.

There is yet another benefit to working with the 3D printer. As the scope and product get tested, there is flexibility to adjust the design and iterate an enhanced speed via skipping a machinist or ordering an existing part and waiting for shipping. The product needs to have some flexibility to change as the sponsors need. This method best allows the creation of different housing modules to accommodate different cameras down the road should that be needed. There

were several iterations of the camera housing. The angle of the camera was changed several times. To accommodate the change in angle, a support structure was added as the cylinder holding the camera shifted. Additionally, modifications to the slot for the camera were made. One such change was the landing spot for the camera lens to fit against an O-ring.

The remaining items for this project are screws and hardware related to the computer vision system. These are modular pieces that can be interchanged with related products. For example, any USB-3.0 camera could work with the setup. Additionally, more I/O can be added for monitoring things like bed temperature with thermocouples which is another consideration for this project.

The end assembly of the product involves lining up the housing unit with the modified lid and screwing in screws to their corresponding threaded holes. The corners of the print chamber line up with the lid making finding the holes in the print chamber easier with less time spent on the guesswork of positioning. Additionally, for the entire product assembly, wires must be run and connected to their corresponding I/O. There still is a need to investigate the airtightness of the 3d printed housing unit and the aluminum plate.

The last step of getting the product operational is the deployment of the code for the computer vision system. The code is stored on Git-Hub to allow access to it anywhere via the internet. Additionally, all support documentation for the project including the camera housing file is available in the repository to help avoid confusion.

Design for Cost:

To minimize costs, it was necessary to find the best software and hardware that met the needs of the project and was the least expensive. On the hardware side, equipment was needed that was easily transportable, modular, compatible with the software, and fast. An X1 Extreme ThinkPad was chosen for the brains of the system because it’s a durable laptop with plenty of

power and speed. The ThinkPad is very cost-friendly in caparison to its piers with similar performance capabilities. The fast and compact Speedgoat real-time unit system aids in the modularity of the entire system, and its integration with MATLAB/Simulink significantly decreases the cost for data acquisition. The Speedgoat eliminates the need for a controller/chassis system that is far more expensive, larger, and more complex to use for real-time applications. The selected method of creating the housing unit was fused filament fabrication. This was primarily chosen for convenience and cost. The ARTs Laboratory already houses multiple 3D printers. An Ender-3 pro was used, and the geometry of the part allowed for minimal support material to be used saving time and money. An alternative to this choice would be to take an aluminum block/billet. In terms of cost, the block is not expensive ranging from 30-50 USD, however, it requires greater than 3-axis machining so the cost would increase since the part has complex geometries and overhangs of a circular nature. By not going the metal housing route, time and money would be saved. The part prints in under 24 hours without much supervision or intervention. A skilled machinist would have to devote time and energy to replicating the geometries and tolerancing the 3d printer can already achieve. eBay was used to find the hardware needed at the lowest cost. On the software side, MATLAB/Simulink was chosen because it is less expensive than software such as NI's LabVIEW. It's perfect for real-time applications, especially vision-based projects, and is easy to use. For future production, a universal camera housing would be needed but was not necessary for this project.

Design for Safety:

There are multiple safety factors involved in this project, but all are easily controlled by implementing the recommended precautions. The metal 3D printer will include the most safety precautions, and all are listed in the printer's manual (Appendix) which would need to be read

through before use. The system's area includes the proper exhaust and circulation along with the necessary fire extinguishers: A-B-C-extinguishing agents for fires caused by electricity, sand, or D-extinguishing agents for fires of processing materials. Flame retardant gloves, full respiratory mask with category P3 filter, and powder protective glasses are on-site and will be highly recommended in the systems manual. Flame retardant gloves, full respiratory mask with category P3 filter, and powder protective glasses are on-site and will be highly recommended in the systems manual. The design of the camera housing was created with no sharp edges and little to no safety hazards. The electronics will be placed on top of cooling fans to avoid overheating. The milled cover is made of aluminum and is light enough for the average person to be able to handle it with ease.

Design for "Speed":

The purpose of this project is to produce an accurate real-time system that can eventually be used to detect errors in an SLM process. The goal is to hand off this system to the sponsors: Austin Downey and Yanzhou Fu, who will implement a machine learning model that can learn from these errors. For a machine learning model to be able to learn from data a lot of processed data is needed while maintaining a high degree of accuracy.

The most important requirement of the project is the real-time aspect of the system. To reach the real-time deadline, the image processing code needs to be able to process a frame from the video feed before the next frame comes in. Over the course of the last two months, different versions of the same image processing code for C++ and Python were developed and optimized to reach the real-time deadline. This code will then be implemented onto a real-time system (Speedgoat or PXI-8812) with the use of either LABVIEW or Simulink Real-time.

One major drawback to writing the code with C++ is that it is a difficult language to use and implement. There are many little things that can go wrong when setting up OpenCV libraries and this would be required for every user to do. The next design focuses more on the ease-of-use aspect.

Design for "Accessibility":

Another important aspect of the project is accessibility and ease of use. The goal is to make this system as accessible as possible. This is reflected in some of the design choices. The camera frame can be 3d printed, making it accessible to any university or company that wants to implement the system. Simulink Real-time is used instead of LABVIEW. LABVIEW subscription is more costly than a MATLAB subscription. The code will be written in MATLAB because the code needs to be compiled with Simulink in real-time. While working with other coding languages in Simulink is possible. There is an initial setup required by the user before they can implement the code.

Conclusions:

Key refinements in the process were the implementation of MATLAB/Simulink in conjunction with the Speedgoat DAQ. This combination reduced the cost on the hardware end and software end significantly. It also increased the overall speed capabilities of the real-time vision application with integrated real-time modules that are easy to program and implement. The system is easy to assemble, calls for minimal manufacturing and maintenance, efficient, and safe. Ideally, recommended future changes could be made to the camera housing to allow for various camera adaptations and the ability to change the angle of focus.

9.2 Build instructions

Introduction:

This report will provide the necessary instructions for future users to easily set up and operate the system. A detailed list of all materials needed is provided in the appendix along with the links to access the necessary g-code and Python code.

Setting up the code:

The code is the foundation of the project, it is what connects all the components of the system together. The primary need described by the sponsor was to have the system work in real-time (in this case under 30ms). The same frame processing code was written in three different languages (Python, C++, and MATLAB) to see how well each performed and was implemented on a real-time machine.

A. Download the chosen programing language and get acquainted with the platform.
B. Download the chosen program of either Simulink or LabVIEW.
C. Download the supplied Python code from the publicly accessible GitHub account using the following link. GitHub.
D. The utilized Python code can be seen in the screenshot below (Figure 1). Code comments describe how the frame is processed, how the sputter parameters are found, and, can be used to better understand the process.

```python
import IPython as IP
IP.get_ipython().magic('reset -sf')
from time import perf_counter
import cv2
import numpy as np


def process_frame(image):
    # Create a copy of input image to later to layer the contour onto
    OG_image = image.copy()

    # Gray scale image
    gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

    # Blur to remove noise (radius must be OOD)
    image = cv2.GaussianBlur(gray, (5, 5), 0) # 21,21

    # Threshold the image to reveal light regions in the blurred image
    image = cv2.threshold(image, 210, 255, cv2.THRESH_BINARY)[1]

    image = cv2.erode(image, None, iterations=1)
    image = cv2.dilate(image, None, iterations=1)

    # Find Contours
    contours, hierarchy = cv2.findContours(image, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
    im2 = cv2.drawContours(OG_image, contours, contourIdx=-1, color=(0,255,0), thickness=3)
    # cnts = contours[0] if len(contours) == 2 else contours[1]

    # Find the biggest contour
    max_area = 0
    for c in contours:
        area = cv2.contourArea(c)
        # print(area)
        if area > max_area:
            max_area = area
            contour = c

    # Find centroid of the sputter
    M = cv2.moments(contour)
    cX = int(M["m10"] / M["m00"])
    cY = int(M["m01"] / M["m00"])
    cv2.circle(im2, (cX, cY), 5, (255, 0, 0), -1)
    cv2.imshow("Image", im2)

    # Draw contours onto original image
    contour_mask = np.zeros(gray.shape, np.uint8)
    cv2.drawContours(contour_mask, contour, -1, 255, -1) # draw filled contours

    # Calculate brightness
    mean = cv2.mean(gray, mask=contour_mask)

    # Draw centroid onto original image
    ((cX, cY), radius) = cv2.minEnclosingCircle(contour)

    return max_area, radius, cX, cY, mean
```

Figure 18: Screenshot of Python code supplied by GitHub

E. To implement the frame processing code onto the Speed Goat real-time unit, either LABVIEW real-time or Simulink real-time will need to be used. Below (Figure 2) shows the general process to implement the code in Simulink with the code placed into a MATLAB function block.

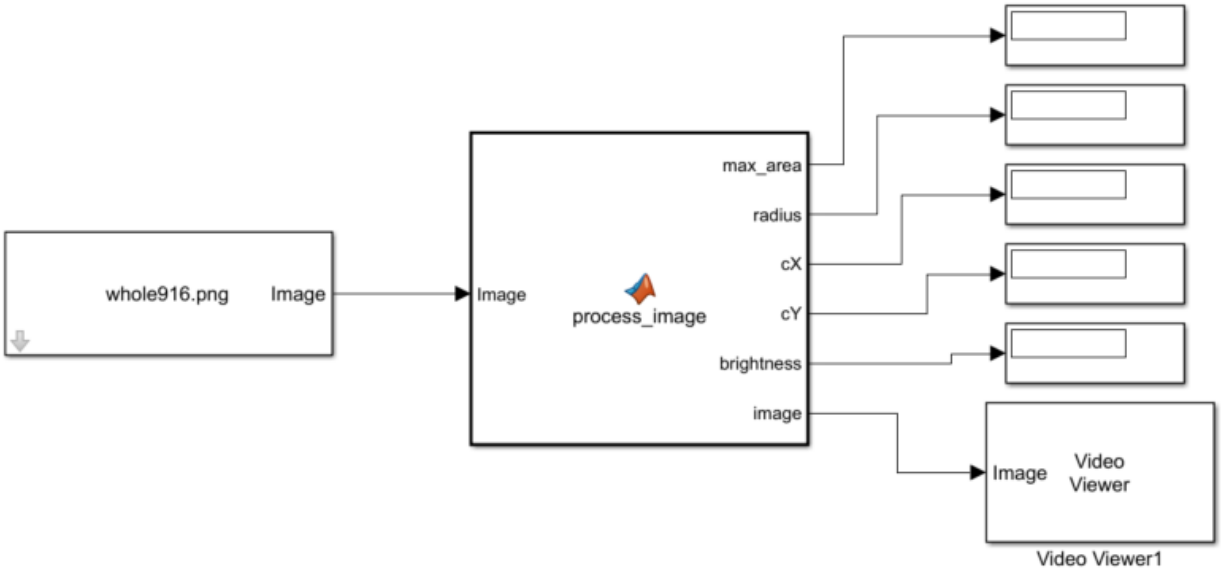


Figure 19: MATLAB function block

F. To completely understand the history and process of this project, there is a general read me in the GitHub repository which describes what the project is about, how it is implemented, and answers to basic troubleshooting problems.
G. Once the code is implemented onto the Speed goat it will be ready to be deployed.
H. The code is set up so that it is user-friendly and can be started with just the touch of any key on the keyboard of the laptop in use.

Preparation of the Ender 3 Creality 3D printer:

A. Clean the printer bed thoroughly by removing any leftover remanence of PLA from any previous prints.
B. The bed should be leveled manually by adjusting the four leveling springs below the bed to acquire equal, paper-thin spacing between the extruding nozzle and the surface of the build plate.
C. The PLA should be loaded into the printer following the instructions from the printer's manual (Appendix).
D. The temperature settings for the build plate and the nozzle should be set at 60° F and 210° F [Figure 20].

Figure 20: Temperature setting for 3D printer

Note: A small test print is highly recommended before attempting to print the camera housing.

Printing the camera housing:

A. Access to the g-code for the camera housing can be found on the public GitHub account for this project using the link below: https://github.com/ARTS-Laboratory/Senior-Design-Project-EMCH427-002-Team-4-Downey/tree/main/G-code
B. The type of filament used to print the camera housing is 1.75 mm diameter MakerBot PLA filament [Figure 21].
C. Place filament onto material rack.

Figure 21: MakerBot PLA filament

D. Preheat the nozzle temperature to approximately 210 degrees.
E. Press the extrusion spring, and feed filament through the extruder until it reaches the nozzle opening.
F. Print the camera mount at 100% speed.
G. Download g-code files from GitHub onto a microSD card.
H. Insert microSD card into microSD card port of 3D printer.
I. Power on 3D printer.
J. Scroll down to the 'Print From TF' command and select.
K. Scroll down to 'camera mount part A' and select to print first section of camera mount.
L. After first section is complete, select 'camera mount part B' to print second section of the camera mount.
M. Remove parts with care and discard any support sections.
N. Using an adhesive, bond the four studs of camera mount part B [Figure 23] into the four holes of camera mount part A [Figure 22].

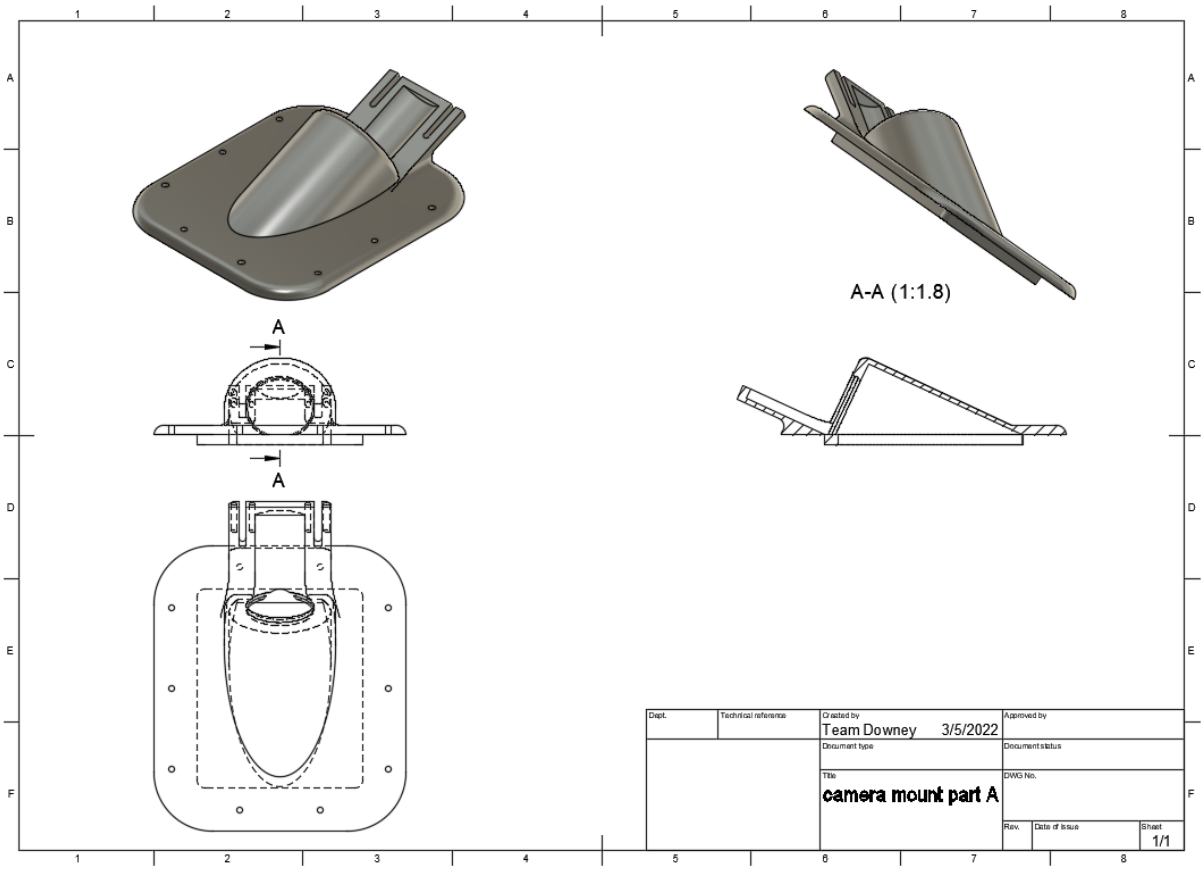


Figure 22: CAD drawing of part A

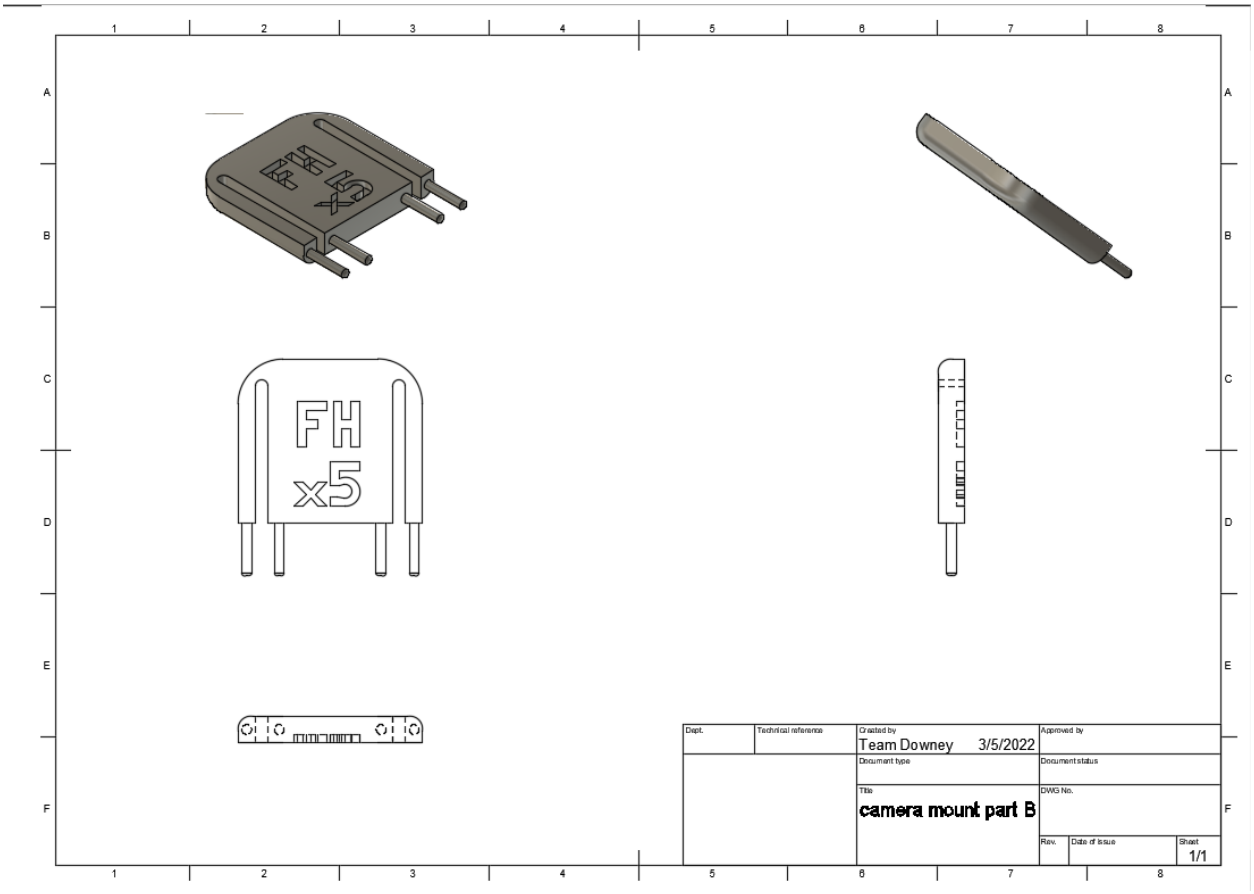


Figure 23: CAD drawing of part B

Camera Settings:

A. Install FLIR Spinnaker support by Image Acquisition Toolbox using the instruction provided at https://www.mathworks.com/matlabcentral/fileexchange/69202-flir-spinnaker-support-by-image-acquisition-toolbox and Spinnaker SDK at https://www.flir.com/products/spinnaker-sdk/
B. Set white balance of the camera using a thick piece of white paper to avoid shadows from bending
C. Enable auto exposure and check the quality of imaging of the printing process. The exposure can be manually controlled to help focus on the desired target more effectively. The camera is under-exposed if the histogram of the light intensity distribution is negatively skewed. The opposite is true for over-exposure.
D. Control the exposure time, gain, and gamma until the brightness of the laser and melting no longer overwhelm the CMOS sensor in the camera.
E. Adjust the image format until it meets the desired area and resolution needed for the zone of interest

Setting up the metal 3D printer with aluminum lid and camera housing:

A. The previously milled and assembled aluminum lid is placed onto the metal 3D printer using the handles to maneuver [Figure 24]. Mount flush with the back wall of the lid holder. It should fit securely in the opening of the 3D printer. The circular opening of the lid should be pointing towards the back of the printer and the square opening towards the front. Close the telescoping housing for the laser to ensure proper placement of the circle.

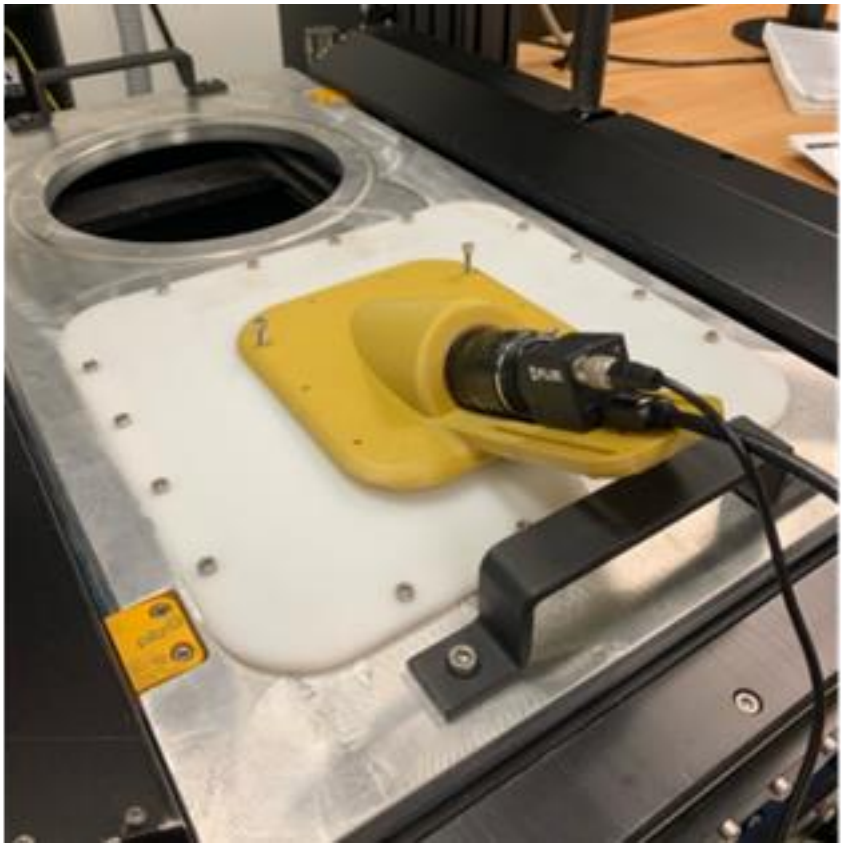


Figure 24: View of aluminum plate with camera housing

B. The camera housing will fit securely into the square opening with the elongated camera rest facing the front of the printer [Figure 24].
C. Secure camera housing corners to the lid using four M4 machine screws [Figure 25] with an Allen wrench.

Figure 25: M4 machine screw

D. O-ring will slide into its built-in compartment in the opening of the camera housing [Figure 26].

Figure 26: O-ring inside camera housing

E. The camera with camera lens already attached should get placed in the camera housing lens first. The camera body will rest on the elongated section of the camera housing and the lens will fit securely against the O-ring. Ensure from the left-hand side FLIR is in the correct orientation. If it is upside down, flip the camera 180 degrees. If it is not visible, rotate 90 degrees and check the orientation.

Connecting all elements:

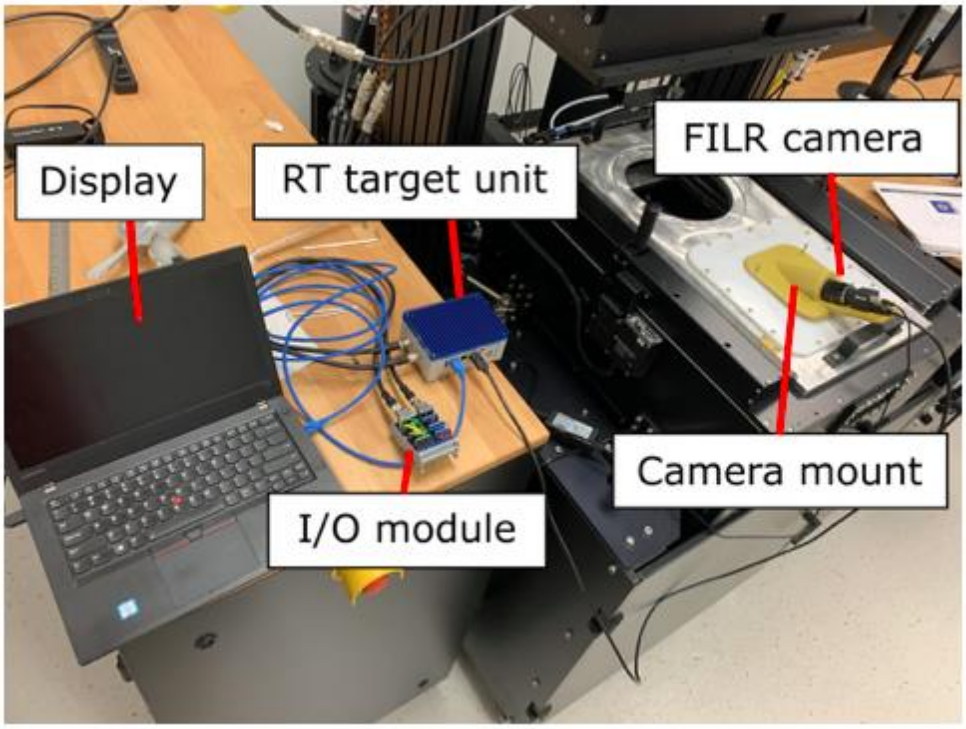


Figure 27: Connections between camera and DAQ

A. The PLA camera mount is first screwed onto the lid of the SLM printer
B. The FILR camera is then secured into the camera mount.
C. The Camera is connected to the Speed Goat real-time target unit via a USB 3.0 cable.
D. The unprocessed frame data is first sent to the I/O module from the data input to translate analog or digital signals to a digital value.
E. Then the digital data is sent back through the data output so the Speed Goat can begin to process the frame data.
F. The frame processing code is hardcoded onto the Speed goat system, so the code does not have to compile each run.
G. The processed frame is then displayed on a ThinkPad laptop connected by an ethernet cable and the sputter parameters are saved to a database on the laptop.
H. The hardware should be near the SLM printer on a sturdy table that won't be in an area that could be in any way disturbed.

Running the system:

A. User will press any key on the keyboard on the display laptop to start the data collection
and the processing of frames
B. A read me found on GitHub using the following link GitHub describes general problems that may occur while code is running. For example, if the code is running

before the SLM process is started there will not be a sputter pool to detect and calculate the parameters, resulting in an error.

i. Troubleshooting the problems while testing code on the Speed Goat real-time unit will be difficult and time-consuming. Speed Goat itself will not display errors. Instead, the error would have to be replicated on the computer using the same inputs that caused the error on the Speed Goat. In order to save time troubleshooting, the code needs to be perfect before implementation on the Speed Goat begins.

C. The display [Figure 28] gives the user the following information:

i. A photo taken using the FLIR camera.
ii. An outline of sputter created.
iii. The brightness in lumens.
iv. The area of outlined splatter.
v. The x and y dimensions.
vi. The radius of sputter outline.

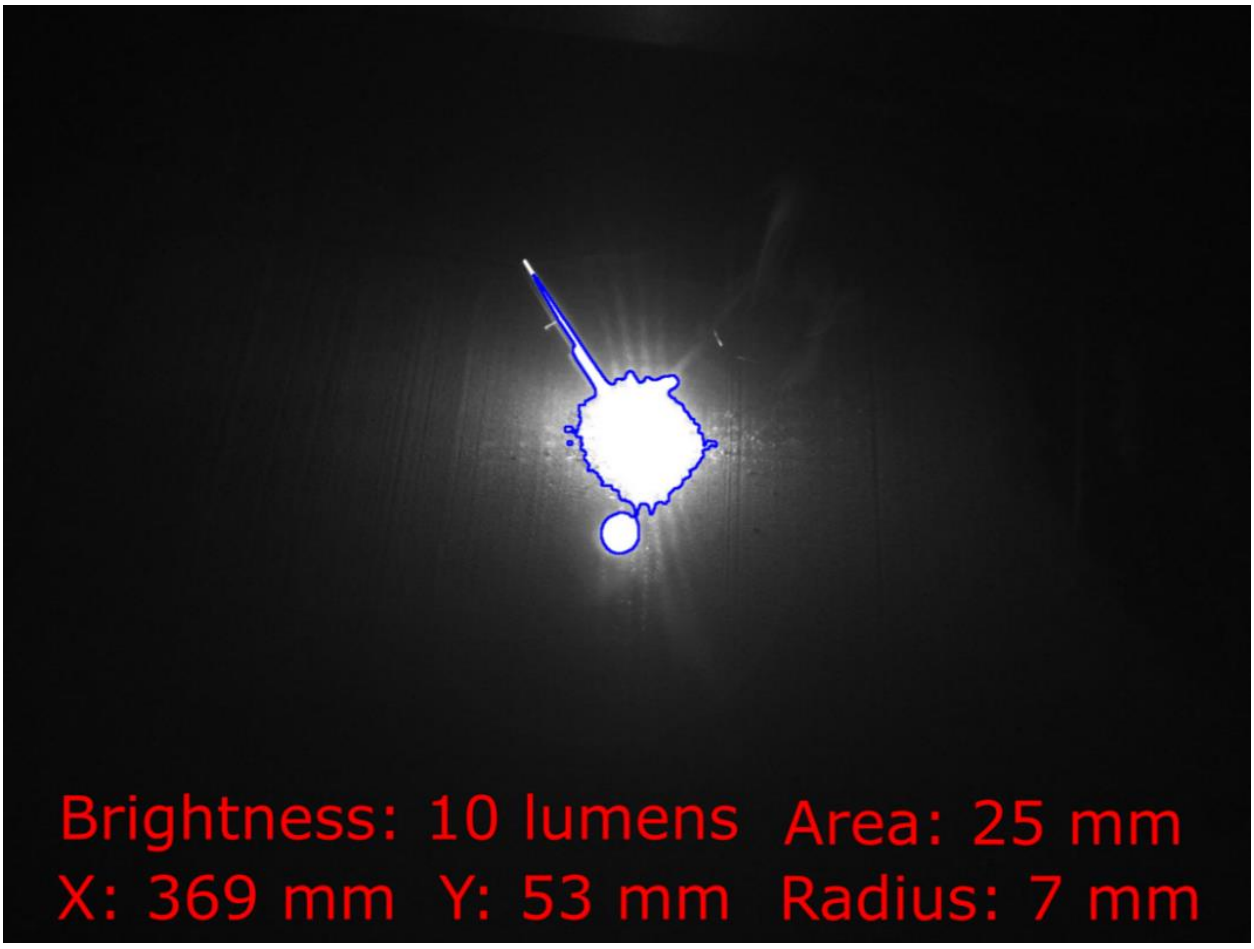


Figure 28:The programmed results display

Maintenance:

A. This system needs to be stored and operated in a low humidity, climate-controlled environment. Left in the heat or elements, PLA (the camera housing's material), degrades becoming brittle and with the potential biodegrading as well. The aluminum lid is exposed to other metals via the powder and molten ejecta in the print chamber, creating the potential for a galvanic cell if exposed to moisture. Electronics, including the camera, are also sensitive to humidity especially if it is high enough for condensation to form.
B. The system should be periodically cleaned.
    i. The lenses of the camera system should only be cleaned with compressed air, avoid canned air, as most cloth materials could accidentally cause scratching.
    ii. In the event of handling the lenses, use untextured nitrile gloves to avoid getting oil or contaminates of the lens.
    iii. The material used to clean the lens should be a cotton swab or a lint free. This would be especially noticeable on the filter lens as abrasions will degrade the filter's ability to block light. Should the filter lens become overly damaged, UV light could begin to degrade the CMOS sensor causing more noise and eventually making it unusable.
    iv. Care should be taken to avoid using solvents to clean the lenses of the camera system as well.
    v. The camera should be stored in a cloth bag inside of a cardboard box.
    vi. Exposure to sunlight should be avoided.
C. Keep the PLA camera housing clean to avoid the presence of the metal powder because it is highly volatile.
    i. Avoid acetone, ethyl acetate, methyl ethyl ketone, tetrahydrofuran and sodium hydroxide as they readily dissolve PLA.
    ii. Using compressed air, avoid canned compressed air.
    iii. If a solvent is needed, Isopropyl alcohol is a preferable option to clean it. The alcohol should evaporate readily and not oxidize the potential metal powder unlike other common cleaning products like bleach. Allow several minutes for the evaporation of isopropanol before reassembly to avoid alcohol vapors in the laser chamber.
D. The aluminum lid should be cleaned periodically to remove any metal powder that may contaminate the surface.
    i. Use a cloth, disposable wipe to remove powder from the surface.
    ii. Take care to use compressed air on threaded holes that cannot be cleaned well with the wipe. It is important to clean these screw holes because they are another source of potential issues.
    iii. Isopropanol can be used if a solvent is needed.
E. Care should be taken to minimize the strain on cables.
    i. Give the cables adequate slack.
    ii. Avoid large bending moments near the connectors.
    iii. Do not jam the connectors if they do not fit, this will result in damage.
    iv. It may be necessary to add a strain relief grommet to increase the longevity of the cables.

v. Cables should be wound into about 4-8 in wide circles depending on their lengths for storage to avoid overstraining.

F. The computer, I/O module, and real-time target for the system should be kept in a climate-controlled environment like an office and not inside of the room where the printer is located. This is to minimize exposure to metal powder during the loading process prior to the print beginning.

G. The computer and real-time target should be cleaned with compressed air after every use in the SLM printer room to avoid the powder from possibly shorting the systems.

X. Economic Analysis

10.1 Introduction:

Economic analyses are necessary for projects that are used for both profit and non-profit purposes. The nature of this project is primarily research-based with almost no startup cost and a heavy emphasis on being available to the general public at no cost. Because of this unique situation, a hypothetical economic analysis was performed with the revenue and lifespan of the product as if it would be used for profit. The estimations that were used for this analysis had been either researched using similar products or discussed with the project sponsor. This economic analysis considered initial start-up costs, labor from the technician operating the system, cost for maintenance, and demand and revenue of the project if sold for profit.

10.2 Costs, Savings, and Revenues

Summary of costs, both up-front and recurring:

All materials that will be needed to run the product are listed below [Table 18]. This does not include the SLM printer or sliding aluminum cover since both are outside the scope of this project. These materials make up most of the cost to build a unit.

Table 18: Bill of materials

Part	Qty	Price ($)
ThinkPad X1 Extreme	1	4,077.98
Ender 3 3D printer	1	251.99
PLA	1 spool	22.99
Simulink software license	1 year	1,300.00
MATLAB software license	1 year	880.00
Camera FLIR blacklfy USB	1	1,887.00
Bandpass filter	1	67.50
O-ring silicone	1	7.92
Speed goat controller	1	9000.00
Machine screws	4	8.00
Total		$17,503.40

Client time:

a. Sponsor -1.5 hrs per week x 30 weeks at $83.00 per hr [30]
b. Consulting Engineer/Technician - 1.5 hrs per week x 30 weeks at $50.00 per hr. [31]

Recurring costs:

a. MATLAB yearly license renewal fees - $880.00 [32]
b. Simulink yearly license renewal fees - $1,300.00 [32]
c. Utilities or non-reusable materials- No use of power that was previously done manually or expendable materials that weren't previously needed. Total cost is listed in Table 19.

Table 19: Upfront and repeating costs

	Cost	Time	Total
Sponsor	$83.00	x 1.5 hrs x 30 wks	$3,735.00
CE/Technician	$50.00	x 1.5 hrs x 30 wks	$2,250.00
MATLAB	$880.00	x 5 yrs	$4,400.00
Simulink	$1,300	x 5 yrs	$6,500.00
Total			$16,885.00

Summary of Savings:

A. Savings on labor:
    i. Technician manually track melting splatter- $66,000 per yr [33]
    ii. Technician manually measure and document melting splatter dimensions and brightness frame by frame- $66,000 per yr [33]
B. Output increase:
    i. Increased output due to accelerated fault detection capability- It would be impossible to determine how much the output would increase because the size and complexity of the part being printed could vary significantly.
C. Maintenance cost savings:
    i. Since the vision system would have no effect on the SLM printer directly, there would not be any maintenance cost savings.
D. Production cost savings:
    i. Although there would be production cost saving, it is impossible to determine due to the variety of parts that could be printed. Total savings are listed in Table3.

Table 20: Product benefits

	Savings	Time	Total
Melting splatter tracker	$66,000.00	x 5 yrs	$330,000.00
Melting splatter measurer	$66,000.00	x 5 yrs	$330,000.00
Total Savings			$660,00.00

10.3 Analysis

Cash Flow Diagram

A cash flow diagram is used to visualize the incoming and outgoing cash throughout the project's life. In Figure 29 it is shown how this project will have an incoming flow of cash after its first year on the market. To analyze the products cash flow further the net product can be found by establishing the revenue, operating costs, gross margin, annual depreciation charge, taxable income, income tax, and net income after taxes [Table 21.]. The depreciation charge can be calculated using the equation 4.

$$Depreciation = \frac{initial\ cost - salvage\ value}{number\ of\ years} \qquad Equation\ 4$$

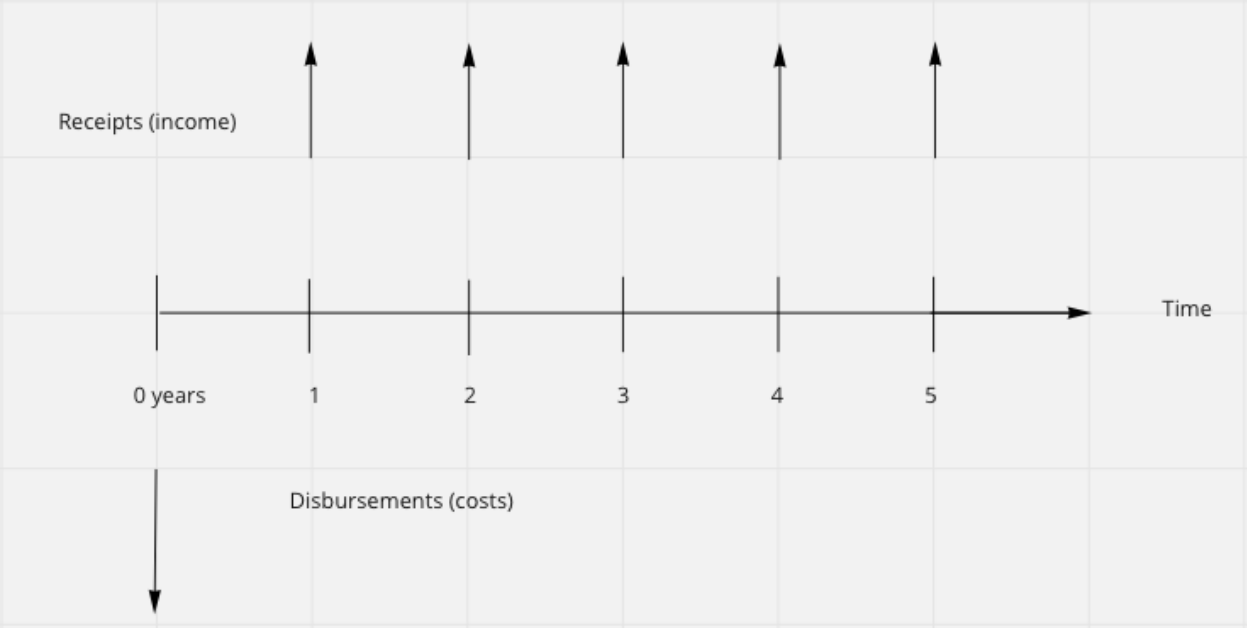


Figure 29: Cash flow diagram

Table 21: Annual cash flow with taxes

Revenue (1 year)	$200,000.00
Operating costs	$44,006
Gross Margin	$155,994
Annual Depreciation charge	$4,382.00
Taxable income	$151,612
Income tax (39%)	$59,129
Net income after taxes	$92,483
Net cash flow (after taxes)	$96,865

10.4 Break even analysis

In order know the point at which the product will start being profitable, a break-even analysis is performed [Table 22]. This project will break even after its first unit is sold which is ideal for a solid business plan. The formula to find the break-even point is below in equation 5.

$$Break\ even\ point = \frac{Fixed\ cost}{Margin\ per\ unit} \qquad \textit{Equation 5}$$

Table 22: Break even analysis

Break Even Analysis	
Startup Costs	$24,253.40
Sale Price Per Unit	$100,000
Units/Year	2
Profit/Unit	$77,997
Profit Margin	78%
Units to break even	1
Days to break even	180
Profit/Year (before taxes)	$155,994

A more visual representation can be seen in graph form [Figure 30] where the grey line represents the total cost over 5 years, the orange line represents cash inflow, and the vertical blue line represents the break-even point which is where the total cost and revenue intersect.

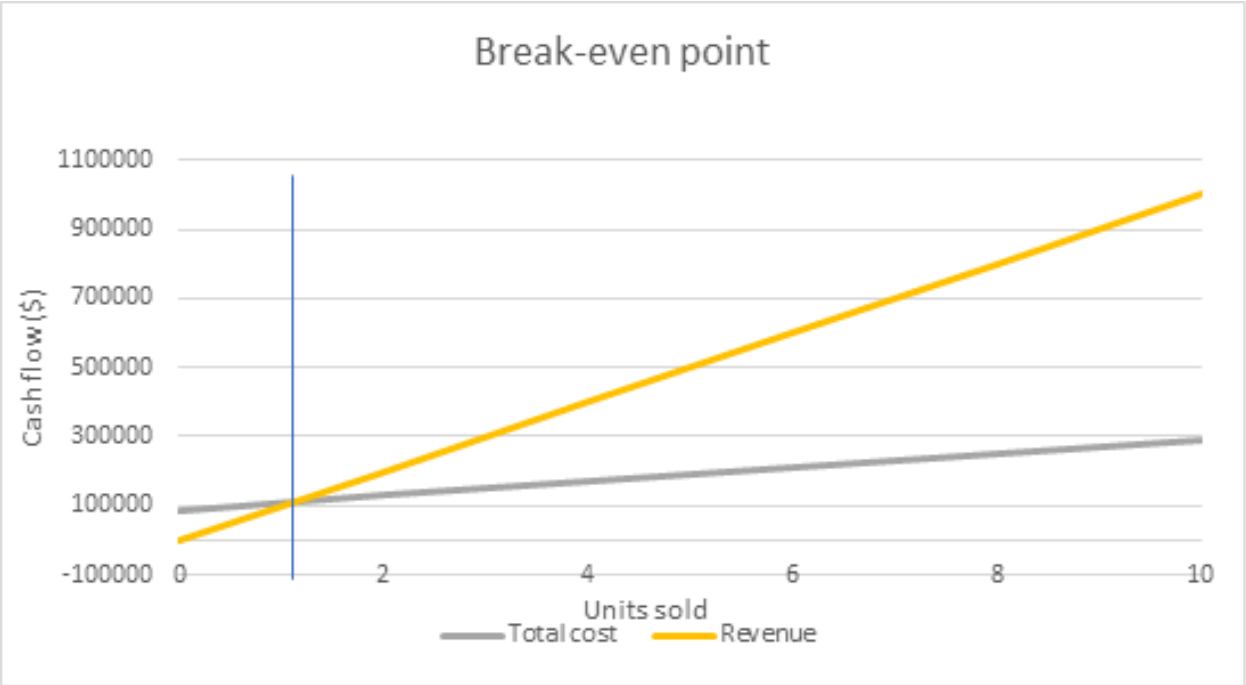


Figure 30: Break-even point graph

10.5 Profitability analysis

To gauge the profitability of a project and the potential return expected, it's important to find the net present value (NPV) or the internal rate of return (IRR). This project's NPV was analyzed using equation 6 below.

$$NPV = \sum_{t=1}^{T} \frac{R_t}{(1+i)^t} \qquad \textit{Equation 6}$$

R_t is the net cash flow, *t* is time of cash flow and *i* is the discount rate. For this analysis the discount rate is assumed to be 10%. This gave a total NPV of $342,942 [Table 23]. The IRR can be found using equation 7.

$$NPV = 0 = \sum_{t=1}^{T} \frac{R_t}{(1+IRR)^t} \qquad \textit{Equation 7}$$

Here the NPV is set to zero and the IRR is calculated. This can be difficult to do manually and is simplified by using the IRR Excel function which is what was done to find the IRR in Table 24. However, it is noted that value given from Excel, 199%, is quite large and this could be due to the large profit margin.

Table 23: Net present value analysis

Net Present Value		
	Project Time (years)	5
	Discount Rate:	10%
Year	Present Value	
0	$-24,253	
1	$88,059	
2	$80,054	
3	$72,776	
4	$66,160	
5	$60,146	
Total Net Present Value (NPW):	$342,942	

Table 24: Internal rate of return (IRR)

Project Time (years)	5
Internal Rate of Return	199%

10.4 Summary

Cost and Profitability

With an estimated revenue of $100,000 per unit sold and two units sold per year by Dr. Downey, the cost of the product, $24,253.40, will be covered fully within the first purchase and be profitable ever year afterwards. It is marketable with projected savings of customers being over half a million dollars per year in labor alone and is one of the first systems available for

monitoring metal 3D printing. Every year, there is an estimated profit of around $155,994 for the project sponsor. In the span of five years, the net present worth of the project will be $342,942.

Additional Benefits

In terms of impact on the market, this is one of the first computer vision systems for metal 3D printing. It will improve the output of metal additive manufacturing by allowing operators to view and stop a print if defects are detected before wasting more time and material continuing a defective print. Eventually, it will be used for real time control of the metal 3D printing process to prevent and avoid defects via a digital twin setup which will make the process more robust, making it a better option for manufacturers. This will allow for new geometries to be implemented that subtractive manufacturing is unable to handle with a similar quality and reliability. Furthermore, with a more robust process, ASME can recognize it and implement standards for those seeking to use it. An example of this would be for heat exchangers in pressure vessels like boiler systems. By having new geometry options, new advances can be made to transfer the heat. This also would apply to computer chips as well. Include discussion of any intangible or less-quantifiable benefits

Computer vision as a means of real time correction will be a major industry overhaul during Industry 4.0. Reducing waste, increasing consistency or quality, and adapting to the manufacturing process are all important to making the best product. 3D printing, in particular metal 3D printing, is plagued with minor inconsistencies which can range from minor visual blobs to internal void defects that severely weaken the materials compared to a wrought metal. The benefit of this project is many folds. First, it will be used in a digital twin setup for U of SC for a PhD Student enhancing his education with an innovative concept that is in its infancy in

now. Additionally, it will spawn multiple papers regarding machine learning and additive manufacturing bolstering the reputation of U of SC.

XI. Product Testing

11.1 Introduction

Product testing is critical to verify the system's functionality and assess its limitations. Data must be acquired through experimentation, processed for error mitigation, and analyzed for results. For this product, the following parameters were evaluated: consistency and speed of the operating system, camera's field of view and placement, most efficient processing method, and accurate measurements. The most rigorously tested parameters were the consistency and speed of the operating system because they fulfill the most essential requirement for real-time image processing.

Since real-time systems are defined by hard deadline times, the system would fail if these deadlines weren't met. Henceforth, most of the testing focused on optimizing the code and the system, as well as testing the limits of the code.

Despite the end product not being functional due to a hardware malfunction of the Speed Goat, the project was a success overall. The code for the system meets the required timing deadline and processes a SLM video on a Speed Goat Real-time Unit. To demonstrate the functionality of the code, a prototype was assembled using a test setup, laptop, and a USB camera. The prototype produced a processed image and a file containing splatter parameter data on a live feed at 30 fps. The last step to completing the product is finding and configuring a Speed Goat compatible camera.

11.2 Test Plan

Due to limited access to the SLM printer laboratory, the construction of a rudimentary model of the SLM printer was required to conduct adequate experimentation. The model was comprised of a cardboard box, the plastic plate where the camera housing is seated, and a

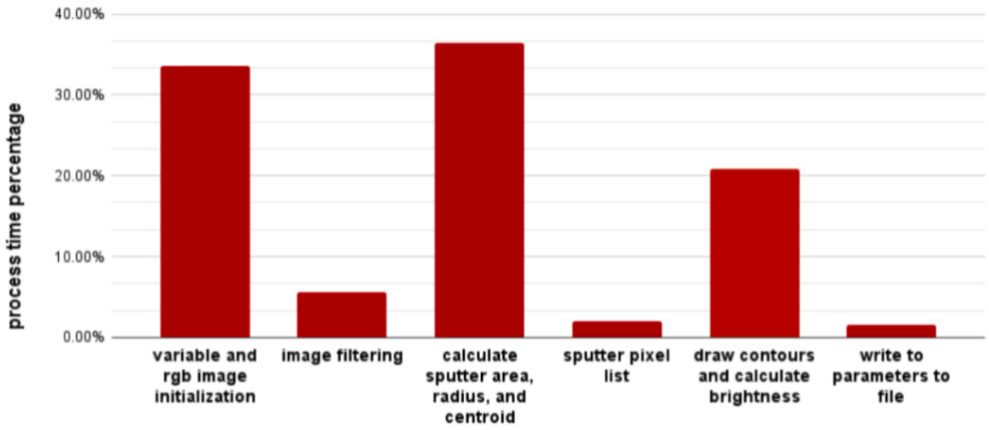
Styrofoam ball to imitate the metal splatter. A square hole was cut out on the top of the box which allows the camera housing to fit securely with the plastic plate and for the camera to view inside the box. A circular hole was cut out at the opposite end of the box as an area for the splatter replica to be suspended from. The inside of the box was painted black, and the splatter replica was painted bright green and illuminated with a flashlight to mimic the light contrast.

Test #1

Testing individual function processing times.

Throughout most of the coding process, only one frame was being processed to see if it met the sponsors' real-time deadline of 30 milliseconds. One way this can be analyzed is looking at individual functions within the code to see which has the greatest effect on processing time [Table 25]. This gives an idea of what to focus on when optimizing. This test was done multiple times throughout the design process. This test is the final function time analysis test of the completed code.

Table 25: Function time analysis of a single frame



Calculating the sputter area, radius, and centroid take up the greatest percentage of the processing time. These calculations are done using the MATLAB function "regionprops", very

little was done to optimize this as calculating the radius and centroid is difficult. The next most expensive function is the “variable and RGB image initialization”, this cannot be optimized further as the minimum number of variables is already used. The last function that takes a large percentage of the processing time is the “draw contours and calculate the brightness”. This can be optimized by no longer drawing the contour around the image, but this eliminates visual aspects of the display which tells the user that the sputter is being found by the code. The reason this function takes so long is that the code must go through the entire sputter pixel list to calculate the brightness.

Test #2

Testing frames per second vs the size of melting sputter.

The code must calculate the brightness and the contours of the main sputter. As the sputter gets bigger the more pixels will have to be gone through to calculate these parameters. For this test, 5 white squares of increasing size show how each affects the processing time [Figure 31]. From this test, the limits of the code were visualized. The code must work at 30 fps and the processing time needs to work under 33 milliseconds.

Figure 31: Five 1440x1080 images of squares

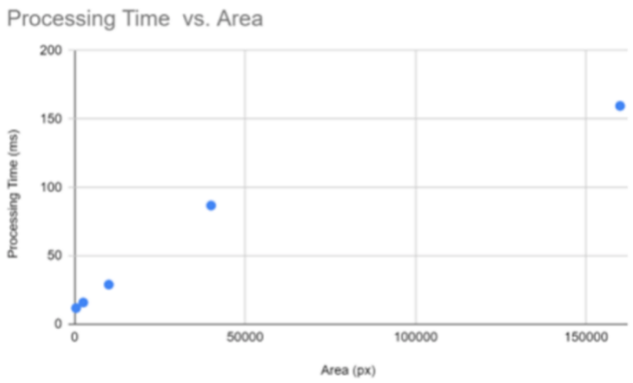


Figure 32: Processing time vs. area

Table 26: Processing time vs. area of squares

	Area	Processing Time
Square 1	400 px	11.81 ms
Square 2	2500 px	15.88 ms
Square 3	10000 px	28.97 ms
Square 4	40000 px	86.78 ms
Square 5	90000 px	159.61 ms

Test #3

Testing different image processing methods to optimize speed and the melting pool's true area.

There are plenty of different image filters provided by OpenCV and MATLAB. Different filters are tried and tested for the various arguments for each function. After the filters are applied, the area of the melting pool and the processing time are calculated. For this test, one base image was chosen as the independent variable to keep the results consistent. The true area of the base image

is calculated and compared to the areas of the filtered images. Each test is run 5 times to acquire an average time.

A couple of important things to note about this test is that the threshold limit for each test is 215. Meaning that any pixel value less than 215 is turned black (0) and all other pixels become white (1). Thresholding is an essential filter to calculate the area and other parameters of the sputter. One other important note is that only the processing time of the filters is taken, not the time it takes to calculate the area of the sputter. This is done to isolate the impact the filters have on the processing time because if one calculated area is bigger than another area that would affect the processing time.

Filters used

- Gaussian Blur: reduces the image noise and the detail of an image.
- Dilate: adds pixels to the boundaries of a binary image.
- Erode: subtracts pixels from the boundaries of a binary image.
- Threshold: simplest way to create a binary image.

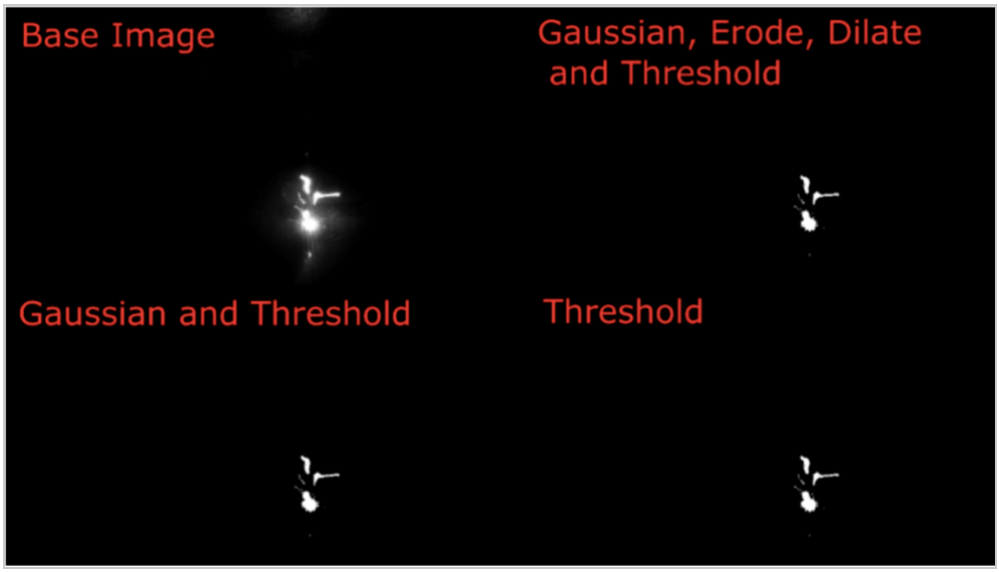


Figure 33: Montage of filtered images

Table 27: Filtered images processing time and area

	Area	Processing Time
Base Image	3237 px	0 ms
Gaussian Blur, Threshold, Dilate, and Erode	3236 px	11.67 ms
Gaussian Blur and Threshold	3239 px	7.08 ms
Threshold	3238 px	2.22 ms

It is seen from the results [Table 27] that filters have a negligible effect on the area of the sputter. There is no loss in any important information from using Gaussian Blur, Dilate, and Erode. For the final model, it was chosen to stay with thresholding.

Test #4

Showing parameters vs not showing parameters

The main focus of the project is meeting the real-time hard deadline required to process frames at 30 fps. If the system is unable to meet this deadline, then the system is useless to the sponsors. There are various ways in which the code can be optimized without affecting the functionality of the end-product. One way in which this is done is by eliminating the visual aspects of the display. This means the code would still save the parameters of the main sputter (X/Y cords, Area, Radius, and Brightness) to a file, but it would not display these parameters in real-time. To test this, there will be three different versions of the code. One in which all the parameters and contours are displayed, a version where only the contoured image is displayed, and a final version where only the preprocessed image is displayed.

For this test the test model [Figure 34] will be utilized to process a live video feed. This live video feed will be able to spit a file with the parameter, with a display window, and an extra Simulink block that will take the fps of the processed video feed. An average fps will be taken from the live feed and will be compared to each test.

Figure 34: Test setup

Figure 35: Live process on the test setup

The above image [Figure 35] is from the base video without any image processing and calculations of the parameters. Due to hardware limitations of the ThinkPad and the UVC camera only a max of 28.28 frames per second can be achieved. The second test returns a

contoured live feed and a ".mat file" at the of the test. There is only a 3 frame drop between the first and second tests. This is not the case when showing the parameters to the user. For the third test both the contour and parameters are shown to the user, but it greatly affects the fps of the live feed. The parameters are already saved to a ".mat file" at the end of the test, so it does not make sense to visually show it to the user if affects the processing time drastically. When the code is implemented onto the real-time unit it can run 60+ fps making it impossible for any user to read the parameters in the live feed.

Test #5

Testing the accuracy of radius measurements to verify that the code's outputs are valid.

By printing out a copy of the splatter image to scale, the radius can be manually measured. A ruler will be used to make the measurement. The manually measured radius data should coincide with the data output from the system. This is needed to determine whether the code is working properly, and that the user will receive useful data. Five tests would have been performed to verify the accuracy of the measurements. Again, because of the hardware malfunction, this test could not be carried out.

Test #6

Testing the portability of the system.

Each component of the system was weighed using a digital bathroom scale. Additionally, the operator of the system documented how each part is picked up and if they meet the requirements in the chart. If the object is less than 51 lbs. and all answers are yes/true, then the object qualifies as an OSHA safe lift. [Table 28] The OSHA requirements are met completely.

Table 28: OSHA test requirements

Object	Weight (lbs)	Able to keep 7 inches away from front of body	No twisting involved in the lifting process	Object located at waist level and directly in front of the person	The object has a handle	The load inside does not shift once lifted.
ThinkPad laptop	2.5	yes	yes	yes	yes	yes
Speed Goat controller	1.8	yes	yes	yes	yes	yes
Camera housing	.1	yes	yes	yes	yes	yes
Camera w/lens	1.5	yes	yes	yes	yes	yes

Test #7

Testing the consistency of the camera placement

To check the constancy of the placement of the camera, fiduciary markers would be placed into the mock setup. The operator will rotate the camera until the markers are aligned with the expected orientation of the system and take a screenshot of the video feed. The operator will then remove the camera and redo the process. Software, WinMerge, will be used to compare the pixels of the two screenshots and consistency will be noted by counting the number of differences calculated by the software. The mock setup was not built with the exact dimensions of the SLM printer and therefore rendered this test invalid.

11.3 Analysis

During this testing process, there was no raw data evaluated. All test results are shown above in the previous section. Because of the malfunctioning hardware and limited access to the SLM printer, there were no measurements made.

11.4 Conclusions/Recommendations

Five tests were performed on this system to check for processing time, frames per second, speed optimization, consistency of parameters, and OSHA compliancy. The results for these tests were satisfactory and met the target values. However, there were two tests that were not performed because of the hardware malfunction prior to beginning the tests and the lack of access to the SLM printer. These would have evaluated the accuracy of measurements being made by the system along with the consistency of the camera's field of view. For future testing, having a fully functional control system and full access to the SLM printer is highly recommended.

XII. Final Design

Team Downey worked with PhD candidate Fu, Yanzhou to create and design a real-time vision system to track parameters of the sputters in an SLM printer located in the McNair Center at the University of South Carolina. Materials needed for the system are listed in Table 29. The final design for the system [Figure 36] is what will be used moving forward with the camera housing design [Figure 37] and the code will be implemented on MATLAB. More testing is needed along with a compatible camera and filter.

Table 29: Bill of materials

Part	Qty	Price ($)
ThinkPad X1 Extreme	1	4,077.98
Ender 3 3D printer	1	251.99
PLA	1 spool	22.99
Simulink software license	1 year	1,300.00
MATLAB software license	1 year	880.00
Camera FLIR blacklfy USB	1	1,887.00
Bandpass filter	1	67.50
O-ring silicone	1	7.92
Speed goat controller	1	9000.00
Machine screws	4	8.00
Total		$17,503.40

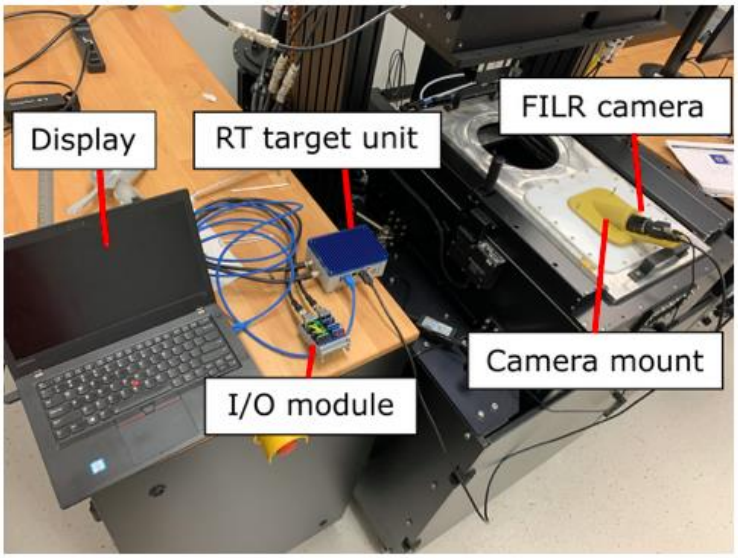


Figure 36: Final design

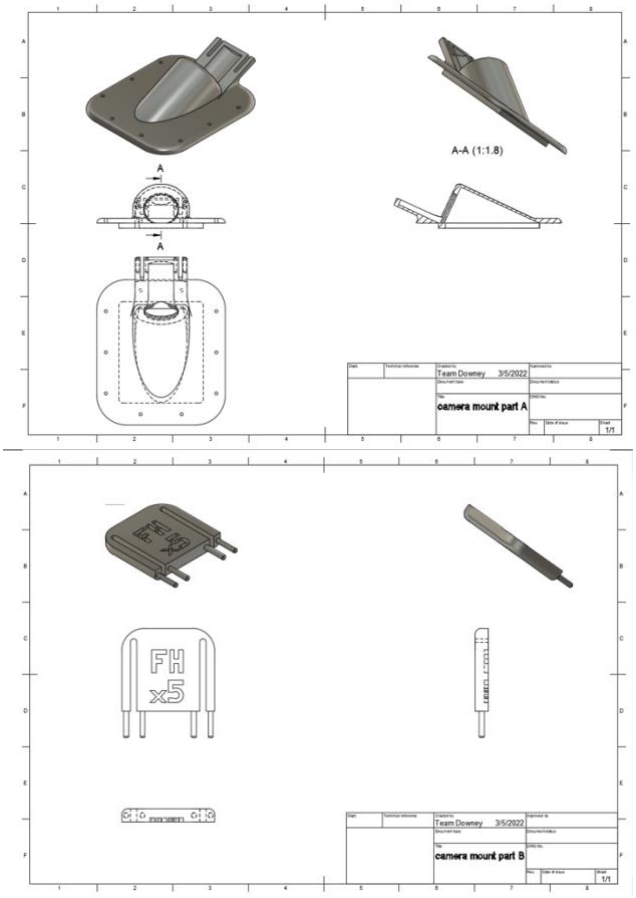


Figure 37: Final camera housing design

XIII. Conclusion

Despite the end product not being functional due to camera compatibility issues, overall, the project was a success. The code for the system can meet the timing deadline given by the sponsors and is able to process an SLM video on a Speed Goat Real-time Unit. To demonstrate the functionality of the code, a prototype was made using a test setup, a laptop, and a USB camera. This prototype can produce a processed frame and file containing sputter parameter data on a live feed at 30 fps. The last step to completing the end product is finding and configuring a Speed Goat compatible camera.

Once the Speed Goat compatible camera is found, the system will need further testing done once it is fully complete. The problem with the system is that it is unable to tell whether the size of the melting sputter is accurate. The sponsors will need to experiment with different camera lenses and thresholding limits to get accurate sputter parameters. More computation speed tests will need to be conducted on the final system. The final system should vastly outperform the prototype performance.

A lot of time was spent trying out different coding languages to see what performed better. Overall doing this was pointless as it would only work on implementing MATLAB version. While this did give a greater understanding of what needs to happen in the code, the time could have been spent better. Time management was a frequent problem for this project. Often, there would be work being done on a problem without fully understanding what that problem was.

This way of working led to many roadblocks, where either the hardware needed to be changed or it needed to start again from nothing. One lesson taken from this project is proper planning is a must.

XIV. Appendix

14.1 References

1. Amini, Mohammadhossein & Chang, Shing. (2018). Process Monitoring of 3D Metal Printing in Industrial Scale. 10.1115/MSEC2018-6332.
2. Clijsters, Stijn, et al. “In Situ Quality Control of the Selective Laser Melting Process Using a High-Speed, Real-Time Melt Pool Monitoring System.” *The International Journal of Advanced Manufacturing Technology*, vol. 75, 2014, pp. 1089–1101.
3. Downey, Austin. *Project Scope: Camera-in-the-Loop System for Additive Manufacturing*. University of South Carolina, 25 Aug. 2021.
4. Downey, Austin, and Yanzhou Fu. “Weekly Meeting 9/24/2021.” 24 Sept. 2021.
5. Flemming, Malichi, et al. “Weekly Meeting 9/10/2021.” 10 Sept. 2021.
6. Fu, Yanzhou. “Discussing the Requirements for the Code.” 23 Sept. 2021.
7. https://patents.google.com
8. https://www.uspto.gov
9. Vivek R. Dave, R. Bruce Madigan, Mark J. Cola, Martin S. Piltch, (2018-0619), “Method and System for Monitoring Additive Manufacturing Processes,” US20160184893A1
10. https://patents.google.com/patent/US20160184893?oq=metal+3d+printer+tracking
11. Gregory A. Peek (2016-12-06), “Printer Monitoring”, US20160283833A1
12. https://patents.google.com/patent/US20160283833?oq=3-
13. dimensional+printer+monitoring
14. Jean-Pierre Kruth, Peter Mercelis, (2009-08-20), “Procedure and Apparatus for In-situ
15. Monitoring and Feedback Control of Selective Laser Powder Processing”,
16. US20090206065A, https://patents.google.com/patent/US20090206065?q=monitoring+of+laser+powder+bed
17. https://onlinelibrary.wiley.com/doi/full/10.1002/admt.201800136?
18. casa_token=CiGQAkBohNkAAAAA%3AwweIp5Q8BWn6dAMl0ncqNZaeaCLheuDcU06z69ZI6K2QFvijpGtvxmn8ljxIm7Crivt_02LpJw_5beTzgA
19. https://opencv.org/about/
20. 1. Downey, Austin. Senior Design Sponsor Meeting (November 12, 2021). In-person.
21. 2. *My USB 3.1 camera does not achieve full frame rate*. Teledyne FLIR. (n.d.). Retrieved November 19, 2021, from https://www.flir.com/support-center/iis/machine-vision/knowledge-base/my-usb-3.1-camera-does-not-achieve-full-frame-rate/.
22. 3. Precision time protocol. (n.d.). Retrieved November 19, 2021, from https://people.cs.rutgers.edu/~pxk/417/notes/ptp.html.
23. 4. *Using the right networking protocol*. NI. (n.d.). Retrieved November 19, 2021, from https://www.ni.com/en-us/innovations/white-papers/10/using-the-right-networking-protocol.html.
24. Ender 3 manual: https://manuals.plus/creality/creality-ender-3-3d-printer-manual
25. Link to download MATLAB: https://www.mathworks.com/products/matlab.html
26. Link to download Simulink: https://www.mathworks.com/products/simulink.html
27. Link to download C++: https://code.visualstudio.com/docs/languages/cpp
28. Link to download Python: https://code.visualstudio.com/docs/languages/cpp

29. Link to download LabVIEW: https://www.ni.com/en-us/shop/labview.html?cid=Paid_Search-7013q000001UgkyAAC-Consideration-GoogleSearch_102713974313&s_kwcid=AL!6304!3!449107487685!b!!g!!%2Bni%20%2Blabview&gclid=CjwKCAiAjoeRBhAJEiwAYY3nDAnGwUk6_IopmtkVKeyH-2ntxA1kRUpAJRdj0g297Wm10hGpNXxUvRoCqagQAvD_BwE
30. https://www.glassdoor.com/Salaries/professor-salary-SRCH_KO0,9.htm
31. https://www.glassdoor.com/Salaries/engineering-consultant-salary-SRCH_KO0,22.htm
32. https://www.mathworks.com/pricing-licensing.html
33. https://www.glassdoor.com/Salaries/laboratory-technician-salary-SRCH_KO0,21.htm

1. Concept selection weight criteria

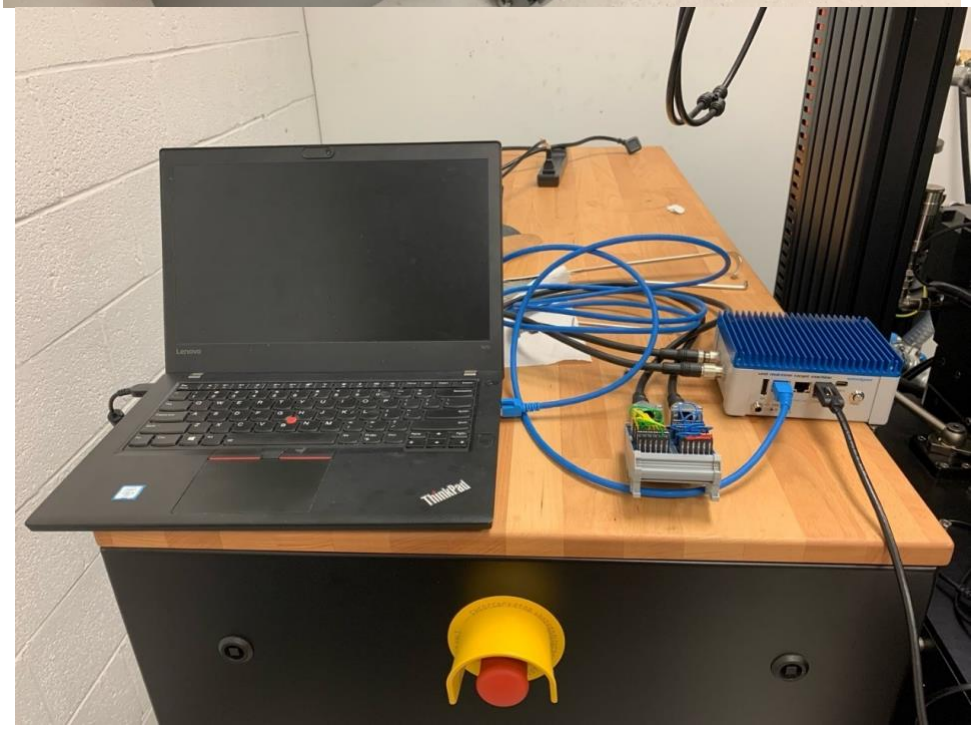
	Setup	Budget	Ease of use	Design complexity	camera placement	data ingestion	display data	Camera protection	DAQ speed	Param. tracking	Camera type flex.	camera stability
Setup	1	0.11111111	0.33333333	0.333333333	0.142857143	0.111111111	0.14285714	0.2	0.14285714	0.142857143	0.333333333	0.2
Budget	9	1	7	9	3	3	5	3	5	7	5	3
Ease of use	3	0.14285714	1	0.333333333	0.2	0.142857143	1	0.111111111	0.11111111	0.142857143	0.333333333	0.142857143
Complexity of design	3	0.11111111	3	1	0.2	0.142857143	0.33333333	0.142857143	0.14285714	0.142857143	3	0.142857143
Camera placement	7	0.33333333	5	5	1	1	3.00	0.333333333	0.33333333	3	1	1
Data ingestion	9	0.33333333	7	7	1	1	9	1	1	7	5	0.333333333
Ability to display data	7	0.2	1	3	0.333333333	0.111111111	1	0.2	0.14285714	0.2	1	0.2
Ability to protect camera	5	0.33333333	9	7	3	1	5	1	1	7	7	3
DAQ speed	7	0.2	9	7	3	1	7	1	1	1	7	3
Parameter tracking	7	0.14285714	7	7	0.333333333	0.142857143	5	0.142857143	1	1	7	0.333333333
Camera type flexibility	3	0.2	3	0.333333333	1	0.2	1	0.142857143	0.14285714	0.142857143	1	0.142857143
Camera stability	5	0.33333333	7	7	1	3	5	0.333333333	0.33333333	3	7	1
Sum	66	3.44126984	59.3333333	54	14.20952381	10.85079365	42.4761905	7.606349206	10.3492063	29.77142857	44.66666667	12.4952381

2. List of all materials needed

Item #	Part	Qty	Website
1	ThinkPad X1 Extreme	1	https://www.lenovo.com/us/en/cart.html
2	3D printer Ender 3	1	https://www.amazon.com/Pro-Removable-Certified-8-66x8-66x9-84in-220x220x250mm/dp/B07GYRQVYV/ref=sr_1_3?keywords=ender+3+pro&qid=1638496927&sr=8-3
3	MakerBot PLA filament	1	https://www.amazon.com/MakerBot-Filament/s?k=MakerBot+Filament
4	Software license for LabVIEW	1	https://www.ni.com/en-us/shop/labview/select-edition.html
5	Software license for Simulink	**1**	https://www.mathworks.com/pricing-licensing.html?prodcode=SL&intendeduse=undefined

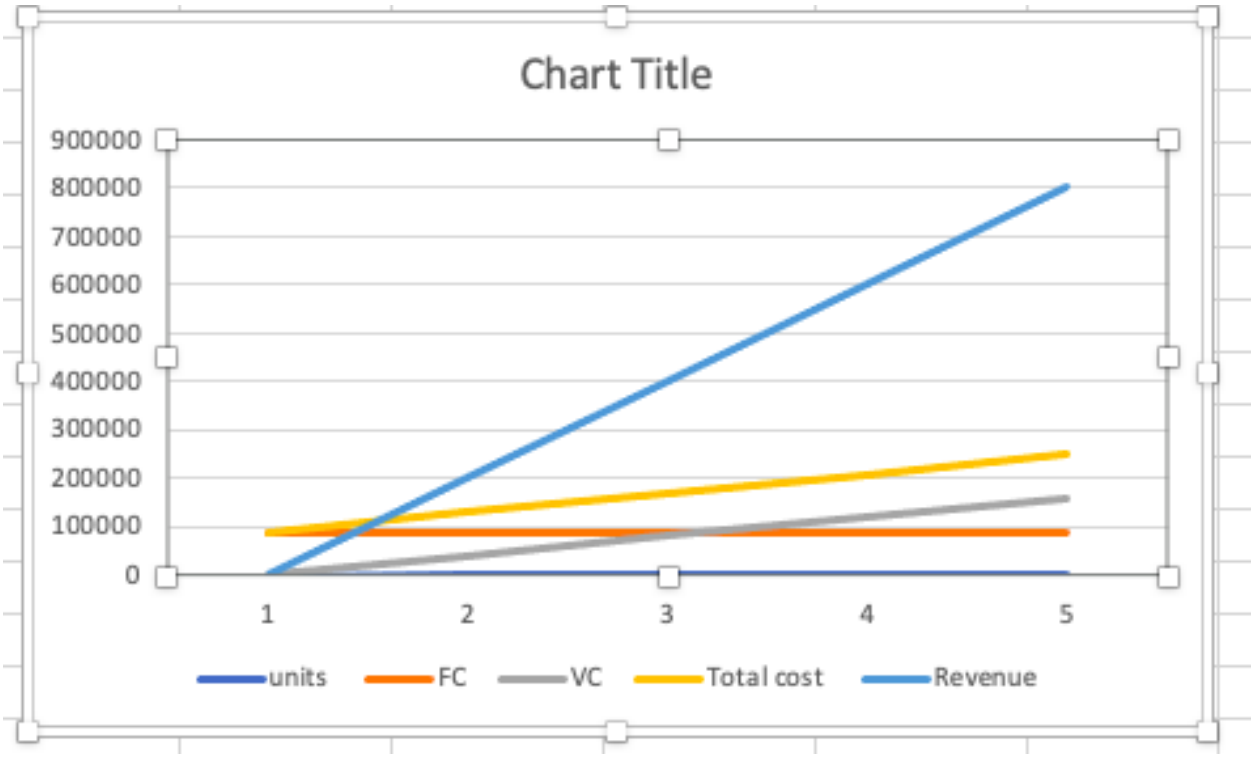
7	Camera FLIR blackfly USB	1	https://www.flir.com/products/blackfly-s-usb3/?model=BFS-U3-122S6C-C&creative=556527044815&keyword=&matchtype=&network=u&device=c&gclid=Cj0KCQiAnaeNBhCUARIsABEee8UtNrFhIgou3_-DeE4ySLAjJf5emQlnNY2VaJhTnl6fnMTkztrV0wsaAhKyEALw_wcB
8	Bandpass filters (BP550 Near-IR/UV-Block Visible Bandpass Filter)	1	https://machinevisiondirect.com/products/midopt-bp550
10	O-ring silicone red 70 shore A (VQM 7) AS568 (#-029) 1/16 x 1.1/2	1	https://www.amazon.com/118-Silicone-Ring-Durometer-Width/dp/B000FN0XSQ
11	PXIE 1062 Chassis	1	https://www.pickeringtest.com/en-us/product/7-slot-lxi-modular-switching-chassis
12	Speed Goat (real-time target unit)	1	https://www.speedgoat.com
13	I/O module	1	https://labjack.com/products/u3?gclid=CjwKCAiAjoeRBhAJEiwAYY3nDDHx1eEP89xTTLIopRgtVFyYNmFiKnDD9qEmPnZu6M9op5fwETUJ2RoChD8QAvD_BwE
14	Size M4 Machine Screw	4	https://www.accu.co.uk/en/cap-head-screws/15874-SSCF-M4-4-A4?google_shopping=1&c=2&gclid=CjwKCAiAjoeRBhAJEiwAYY3nDGdInxp4POws2YVBFKWBa6Rxkc89mtniqu1jGxDSGzIIDZAaoX44bBoC1HgQAvD_BwE

3. Additional hardware setup pictures:


Lenovo
ThinkPad

4. Calculations of break-even analysis:

					VC	19900
					SP	100000
					FC	88250
units	FC	VC	Total cost	Revenue	BEP	1.1017478
0	88250	0	88250	0		
2	88250	39800	128050	200000		
4	88250	79600	167850	400000		
6	88250	119400	207650	600000		
8	88250	159200	247450	800000		
10	88250	199000	287250	1000000		



5. Calculations for depreciation:

Part	Qty	Price ($)	Salvage value
ThinkPad X1 Extreme	1	4,077.98	1,000
Ender 3 3D printer	1	251.99	50
PLA	1 spool	22.99	0
Simulink software license	1 year	1,300.00	0
MATLAB software license	1 year	880	0
Camera FLIR blacklfy USB	1	1,887.00	500
Bandpass filter	1	67.5	0
O-ring silicone	1	7.92	0
Speed goat controller	1	9000	5000
Machine screws	4	8	0
Total		$17,503.40	6,550

Depreciation Cost

$$D = \frac{17,503.40 - 6550}{5 \text{ years}} = \$2190.68 \text{ per unit}$$

Salvage costs were found on Ebay.