3D Metal Printer

Final Report

May 2018 - Team 5

Client and Advisor:

Dr. Timothy Bigelow

Team members:

Kevin Oran - Engineering lead, programming and mechanical design

Ben Pieper - Software design and integration

Jett Ptacek - Software design and integration

Rachel Shannon - Sensor integration, wiring/connectors

Caleb Toney - Sensor integration, wiring/connectors

sdmay18-05@iastate.edu

http://sdmay18-05.sd.ece.iastate.edu/

Table of Contents

Project Design

Our team was tasked with designing and creating a Metal 3D Printer. The scope included designing the mechanical systems, safety systems, software to drive the motors, and the user interface. We also have to allow for future addition of non destructive evaluation (NDE) lasers and software that will be done by a future senior design group. Below Figure 1 summarizes these tasks and how they support the client's task.

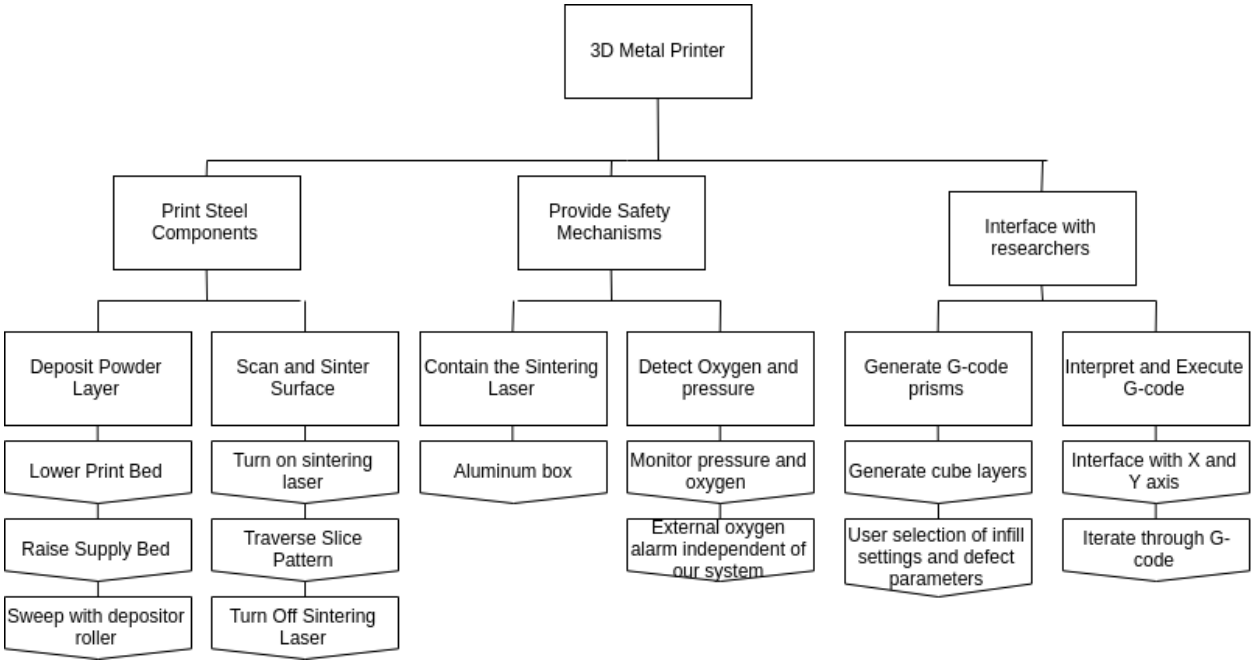


Figure 1. Functional Tree

A quick overview of our solution consists of a mechanical design comprised of several stepper motors to deposit layers of powder onto a print bed and move the melt and evaluation laser heads around. This system is totally enclosed in a sealed chamber which can be evacuated of oxygen gas and replaced with nitrogen. Our solution also contains software which provides the ability print rectangular prisms of various sizes with specified defect areas, monitor sensor readings, and provide critical safety checks of the system.

The primary functionality of our printer is printing of basic shapes using metal powder in a nitrogen environment. Our printer specifically focuses on printing cubes and other square shapes. There is capability to add future shapes and options in the future.

Implementation Details

Mechanical System

Print Beds and Roller

The core mechanical functions of the printer are to scan and sinter a powder surface, deposit another layer on that surface and supply additional material to the depositor for each rail. Part of the design constraints for the printer were to use the Velmex System, which eventually evolved into being required to use the Velmex BiSlide models. Because the print size and focal length requirements, this led to some difficult geometry.

The first main mechanical function is to contain the powder and control the z direction of a print. The team's solution uses two piston-like structures called the powder beds. To constrain the powder beds, the client specified a necessary print volume of 4 cm by 4 cm by 8 cm. The design includes a small buffer to accommodate this in real life. The design of each print bed needed to both contain and translate the steel powder to be used by the printer,the beds also needed to handle high temperatures. To deposit a layer of steel, a roller will push powder from the surface of the supply bed to the surface of the print bed. The final beds are made with a bracket that holds a plastic layer that seals against the print walls above the plastic is the steel print bed, which the powder will sinter to. The beds are operated by smaller velmex slides. The rendering in Figure 2 shows the supply bed and the print bed.

The next feature is the roller. In order to deposit powder from one bed to the next, a roller is used. The roller will push the powder from the supply bed to the printing bed. There is a slot for an overflow container to collect excess powder that overflows from the printing surface. The roller rides on bearings between two rails that prevent powder from spreading out and not flowing into the the print bed. The railings compliment the roller so that if a more complicated roller is needed, it can be pressed against the rails and not depress the printing surface. The depositor roller can be seen in Figure 2 to the right side of the printing machine.

Finally, the mechanical system needs to sweep the surface of the powder with the sintering laser. The team's solution uses a 3-axis slide assembly. The printer's x and y coordinates are controlled by horizontal BiSlides from Velmex. There is also a vertical slide to ensure that the laser height is at the proper distance from the surface of the powder.

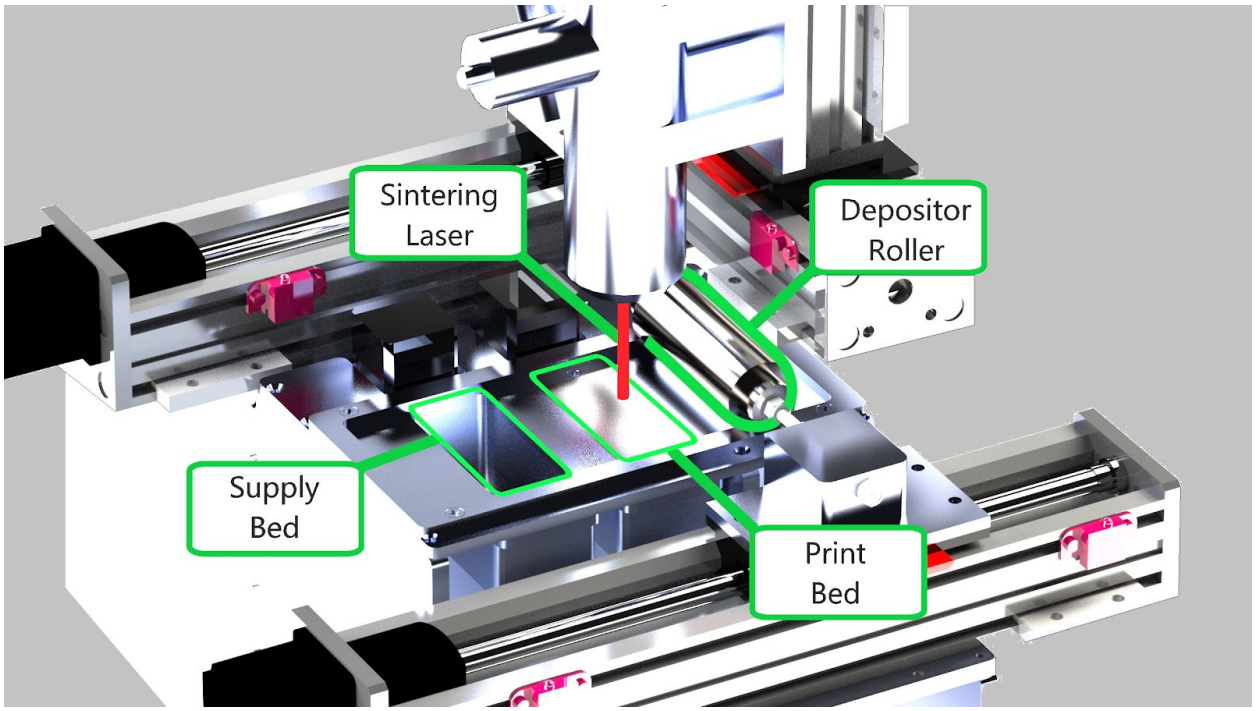


Figure 2. Print Beds, Laser, and Roller Diagram.

Velmex Slides

The team was told by our client that the slides to be used in the printer had to be from Velmex. Velmex produces three main varieties of slides which are all stepper motor driven. In order to operate the machine the slides were studied to select slides which could operate at appropriate. After talking with a Velmex engineer, the team initially selected UniSlides. These slides offered a brushed aluminum finish that is ideal for vacuum conditions, but they were not rated for the same magnitude of load as the other slide types. The original estimates for the weight of the laser head indicated that these slides would work for several functions.

The constraints for the slides later changed, when the client updated the weight estimates of the laser head. Slides had to be re-selected and the machine frame redesigned. The new slides would no longer provide the outgassing prevention the UniSlides had. The team decided the risks posed by the outgassing would not be significant.

The team carried out research on different stepper controllers for the Velmex Slides. The team produced several controller options that would support synchronous operation for less money than Velmex's controllers. However, because the client's research team previously used Velmex's VXM controllers, the client required the team use those controllers again in this design. In total the machine uses four BiSlide models, two XSlides, and 2 motor controllers. Figure 3 shows all the slides side-by-side.

Figure 3. Velmex Slide Product Options

"Velmex Precision Motion-Control And Positioning Equipment." *Velmex Positioning Products*, www.velmex.com/Products/index.html.

Vacuum Chamber

Part of the challenge of selective laser sintering on steel is preventing oxidation of the steel as it is sintered together. Welding techniques usually accomplish this by blowing inert gas over the weld to prevent oxygen getting to the molten metal at its hottest state. This is a bad idea for a machine that deals with powder. In order to accomplish this in industry, professional SLS machines evacuate the air in the printer and fill it with inert gas or nitrogen to prevent oxidation. To achieve this the team needed to design a vacuum chamber to enclose the printer. This constraint brought on several challenges to the design: how to seal the chamber, how to pass electrical signals into the chamber, how to manufacture the chamber and how to pass the fiber optics into the chamber.

As the team looked at how to encapsulate the printer it became clear the chamber would have to address the ship-in-a-bottle problem of how to get the assembled printer inside the chamber. The intent of the original design was to have two halves that would bolt to each other to form a seal. Chamber would then have a removable door giving access to the sintering end of the printer. In addition, the bottom half of this chamber was originally intended to have all electrical ports and the fiber optic ports to ensure that removing the top would be safe and trivial. In order to allow signals and lasers into the chamber, research was done to select electrical feedthroughs and plumbing hardware was selected to permit gas and signals. The

lasers are going to use vacuum rated tubing with specialized gaskets and caps to seal the fiber-optics into the chamber. Another major consideration for the size. The size of the chamber was picked to fit the printer and the fiber optics snaked back to the feed throughs. The client-specified 12 inch bend radius of the fiber optics meant a chamber that was quite a bit bigger than the machinery inside. Fortunately, the team is going to use nitrogen to prevent oxidation, which is a fairly common gas.

The vacuum chamber was manufactured by Sargent Metal Fabricating in Ames, Iowa. The shop cut the aluminum and welded the parts into two large pieces. The vacuum chamber uses rubber gasket material that can handle high temperatures and pressures. The design uses vacuum-sealing washers to help seal the door on. The two halves are fastened with some large bolts that will press the two sides together. A final plate, the door was also produced by Sargent Metal.

After receiving the Vacuum Chamber EH&S felt it was necessary to make changes to enable easier movement of the chamber. The major concern was that the larger top piece of the chamber was too heavy to maneuver without some heavy duty handles. The new arrangement can be seen in figure 4.

Figure 4: Vacuum chamber

Wiring and Vacuum Feedthroughs

A major challenge of the vacuum environment was passing all of the electrical signals through the vacuum chamber wall without compromising the vacuum seal. After much research, our team decided that a Spectite Series WF feedthrough would be the optimal solution to this problem.

The Spectite feedthroughs have holes for each conductor to pass through. The conductors must be Kapton insulated wires in order to make a proper seal. After passing the wires through, the feedthrough must be torqued to around 200 ft-lbs in order to compress the seal around the wires. Then, the feedthrough is screwed into the ¾” NTP threaded hole in our vacuum chamber.

In total, there are 67 wires which need to be passed into our sealed chamber. This includes 10 wires for each motor axis (6 wires for stepper motor, 4 wires for limit switches), and 7 wires for USB signals, I2C communication, and sensor power. We could have cut down on the number of wires required by sharing grounds on the limit switch ground and supply connections, but this would have required additional wire splicing and harnessing inside of the box. Therefore, we chose to simply pass through all of the Velmex related connections, so our Vacuum chamber harness with the feedthroughs is essentially just and extension cord. We specced out and crimped on the same TE Connectivity connectors that come on the Velmex products to make the system truly plug and play.

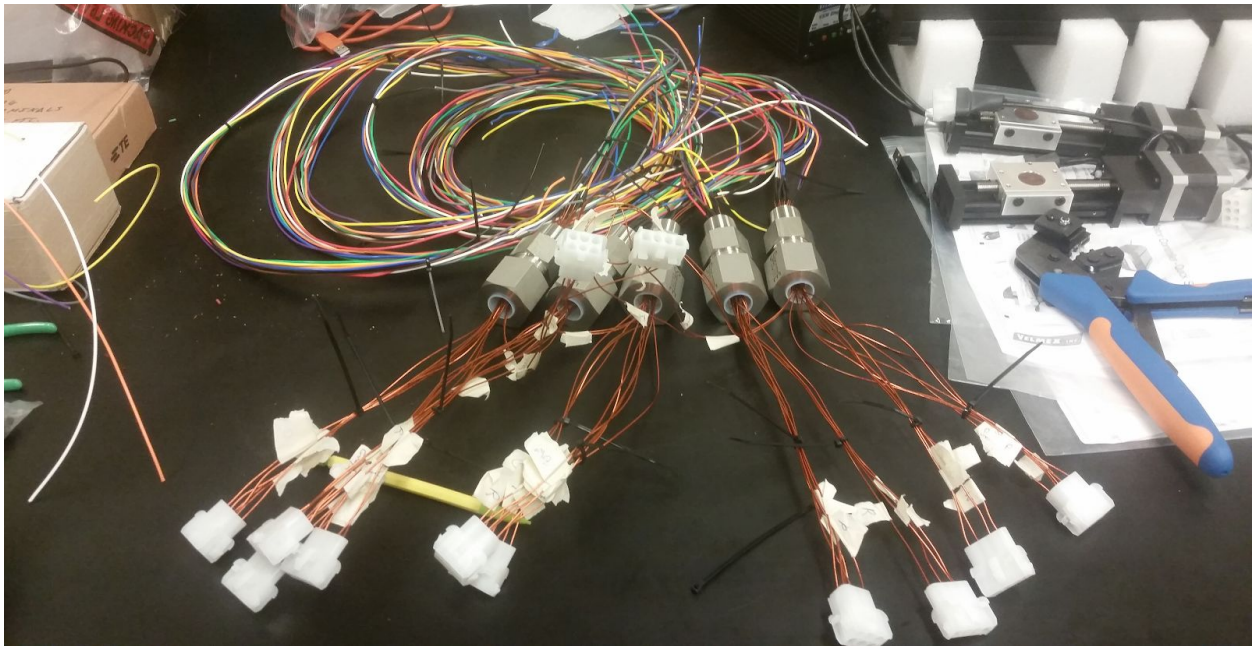


Figure 5: The assembled motor wiring harness.

Sensor System

Oxygen Sensor

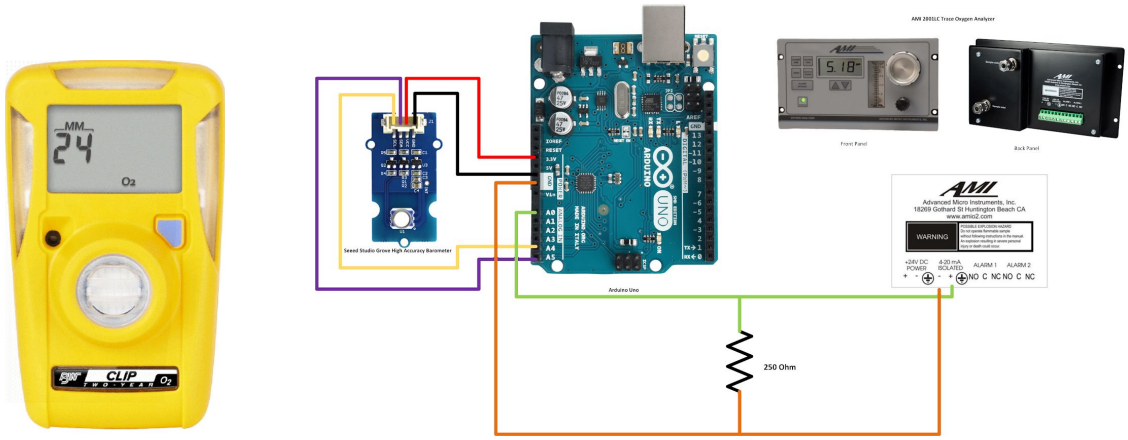
The oxygen sensor is an AMI 2001LC Trace Oxygen Analyzer. We needed to have a sensitivity around 1000 ppm level. We searched for a number of different systems and settled on on the 20001LC due to a combination of cost and functionality. The system outputs a 4-20mA signal that we convert to a 1 to 5 volt signal using a 250 Ohm resistor. This voltage is used by the arduino's adc to calculate the ppm. This result is then sent to the UI on the computer to be used by the human operator. The setup and connection between the oxygen sensor and the arduino can be seen below as well as an image of the sensor itself in figure Figure 6.

Pressure Sensor

The pressure sensor is a Seed Studio Grove high-accuracy barometer sensor that has uses a HP206C to detect temperature and pressure. It communicates via an I2C interface connected to an arduino. The arduino then outputs the readings in hPa and Celsius via serial to the UI on the computer. There are no safety actions triggered by the temperature and pressure readings. They are instead used by the human operator to take action if deemed necessary. The connection between the I2C ports on the sensor and the arduino can be seen below in Figure 6.

External Oxygen Alarm

In order to meet necessary safety requirements, we use a standalone external oxygen alarm while our system is operating. This BW Honeywell Clip 2.0 monitors for an excessive drop in oxygen level and will sound an alarm if necessary. This alarm signals to the operator that they need to evacuate the room immediately. The main instance where this would be necessary is in the case of a leak in the nitrogen gas tank. The device can be seen in figure Figure 6 below.



(a) (b)

Figure 6. (a) BW Honeywell Clip 2.0 (b) Seed Studio Grove High Accuracy Barometer, Arduino Uno, AMI 2001LC Oxygen Analyzer.

Sensor Serial Communication

The Arduino which interfaces with the internal oxygen, pressure, and temperature sensors communicates with the main desktop PC via a serial connection. We devised a very simple serial protocol to communicate this information to the GUI software.

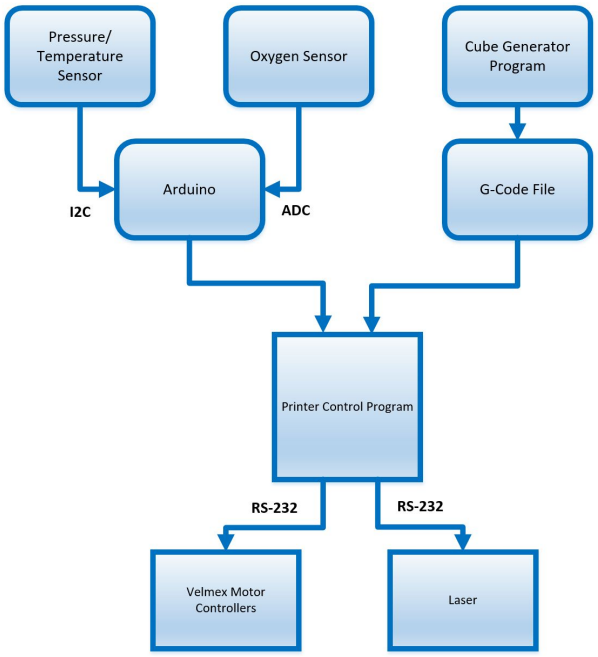


Figure 7: Sensor system integrated to communicate with GUI software.

Software Overview

The control software for the printer was all written in C# using the WPF Framework. This software runs on a dedicated PC we acquired for the project. There are two major sections of the software: G-code cube generation, and the G-code interpreter. The G-code interpreter software takes any compliant G-code file as input, transposes it to Velmex motor controller commands, and prints in on the 3D Metal Printer. While printing, this software shows progress with command selection as well as sensor readings.

The G-code generator software generates .gcode files which will be compliant and can be used with the G-code interpreter software.

G-Code Generator

After refocusing our efforts and narrowing scope at the beginning of the semester, the primary software goal shifted from slicing and processing existing CAD files to generating G-code for rectangular prisms to be printed. This satisfies the needs of the client, and allowed us to focus more time on developing a variety of user configurable infill generation algorithms.

The G-code generator algorithms take the following inputs and output G-code for drawing the resultant shape:

- Laser spot size
- Z layer thickness

- Height of rectangular prism
- Number of perimeter lines per Z layer
- How to alternate infill hatch direction (checkerboard, random, same direction)
- Number of perimeter lines per infill square
- Infill square size
- Number of infill squares (x/width)
- Number of infill squares (y/height)
- Order of printing infill squares (sequential, random, every other)
- Defect area box location and dimensions

This gives researchers a good number of parameters to play with in order to print shapes which meet their needs. The infill patterns implemented match those specified by our client verbally in team meetings. Furthermore, a feature has been added which allows the creation of a 'defect' in the part. In order to create a defect, the laser is turned off while passing through this region of the part. This will create an area of unsintered metal, which would be a mechanical defect to the part's structure.

All algorithms to generate the G-code are contained in the CubeGenerator.cs file, and the code is commented such that a future student or researcher should be able to understand and modify it.

In order to simplify the G-code generation, our software outputs all x-y movements of the laser head as standard G1 commands, but abstracts Z movement and layer changes into custom defined 'M' commands. Thus, if someone wants to input a G-code program from a standard slicer into our G-code parser, they must modify the parser to implement our custom 'M' commands. This is very similar to how one would specify they brand of plastic 3D printer using a standard slicer, custom handlers for layer changes and other parameters would have to be set up for the slicer. Table 1 summarizes the commands created by the team to operate the printer.

Table 1: Custom 'M' Commands

Command	**Description**	**Example**
M200 X	Execute layer change of height X	M200 0.25
M201	Turn the melt laser on	M201
M202	Turn the melt laser off	M202

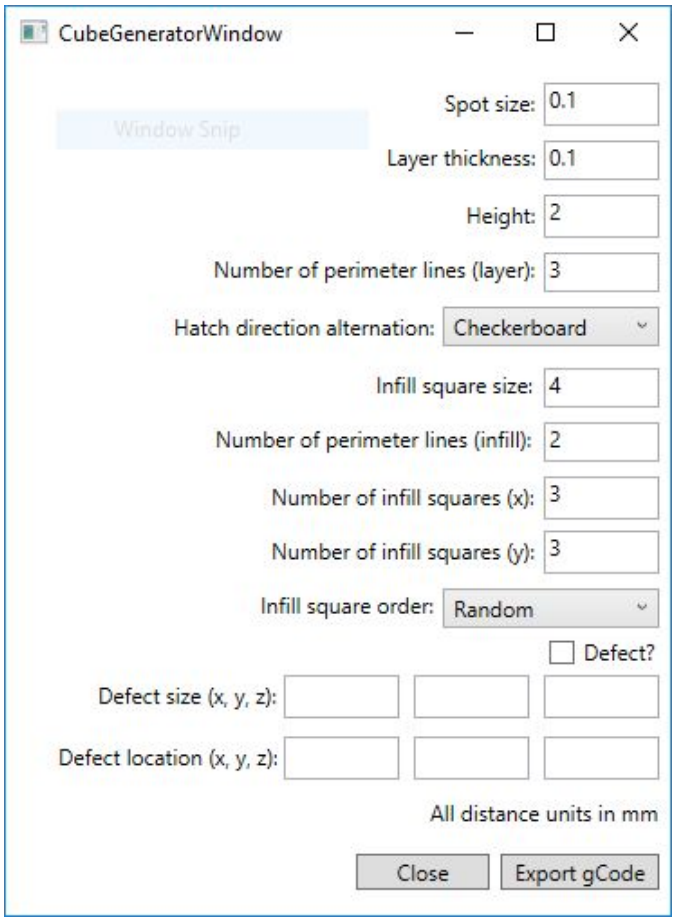


Figure 8: The front end interface for the cube generation algorithms.

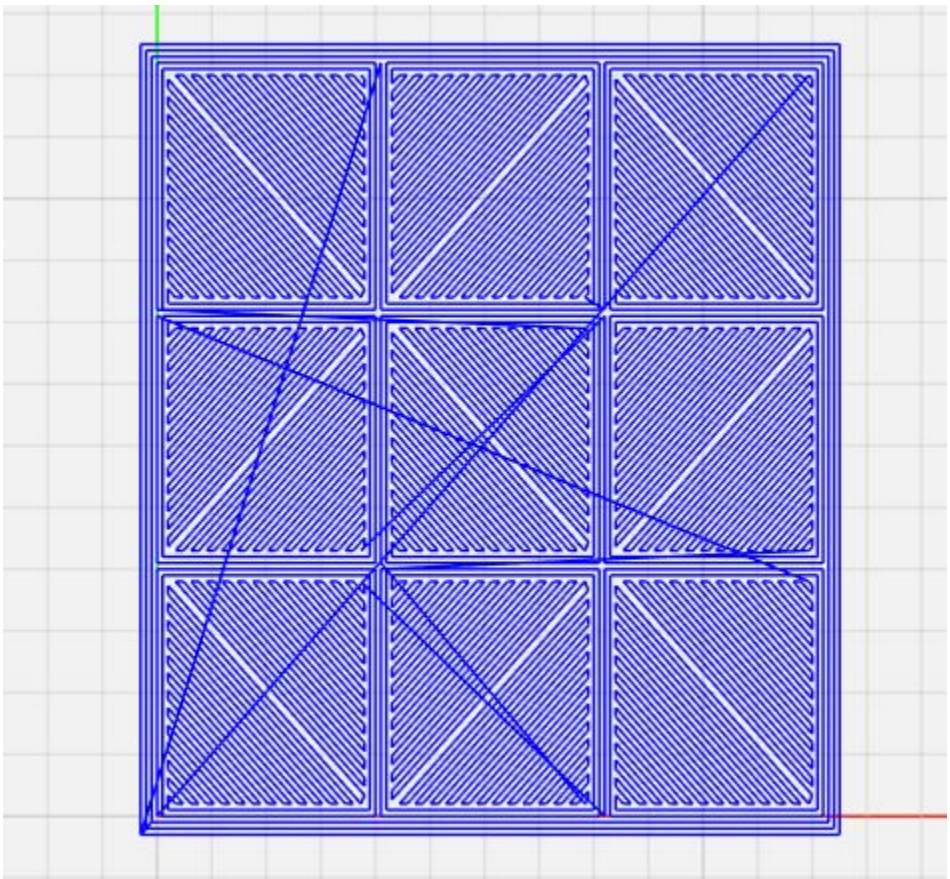


Figure 9: A visualization of the output of the G-code generated by the above screenshot. This shows a top down view of one of the layers. The long lines crossing the entire square show movement of the laser head, but the laser will not be on for these movements.

G-Code Interpreter

The main function of the G-Code Interpreter is to parse inputted G-Code and generate Velmex commands. After generating, the Velmex commands can then be outputted serially which will control the stepper motors given the outputted data. The commands can be outputted one at a time for incremental testing or executed all with another button. Individual commands can be selected to run if required. Also, the parser has a speed parameter which is necessary to set the base speed of the stepper motors.

The generated Velmex commands are separated as four different actions, velmex_move, laser_on, laser_off, and layer_change. These are defined in PrinterAction.cs. Velmex_move is a standard movement of the stepper motors in the x and y direction. Laser_on is an action that will later be used to enable the laser and laser_off will disable the print laser. The layer_change action is multiple velmex commands that are always the same that move the

powder bed and supply and the roller in the correct order and timing to setup the next layer for printing.

All algorithms to parse the G-code and generate the Velmex commands are contained in the GCodeParser.cs file, and the code is commented such that a future student or researcher should be able to understand and modify it.

The sensor data is from the Arduino which interfaces with the internal oxygen, pressure, and temperature sensors, and the Arduino sends its values to the computer through the serial protocol.

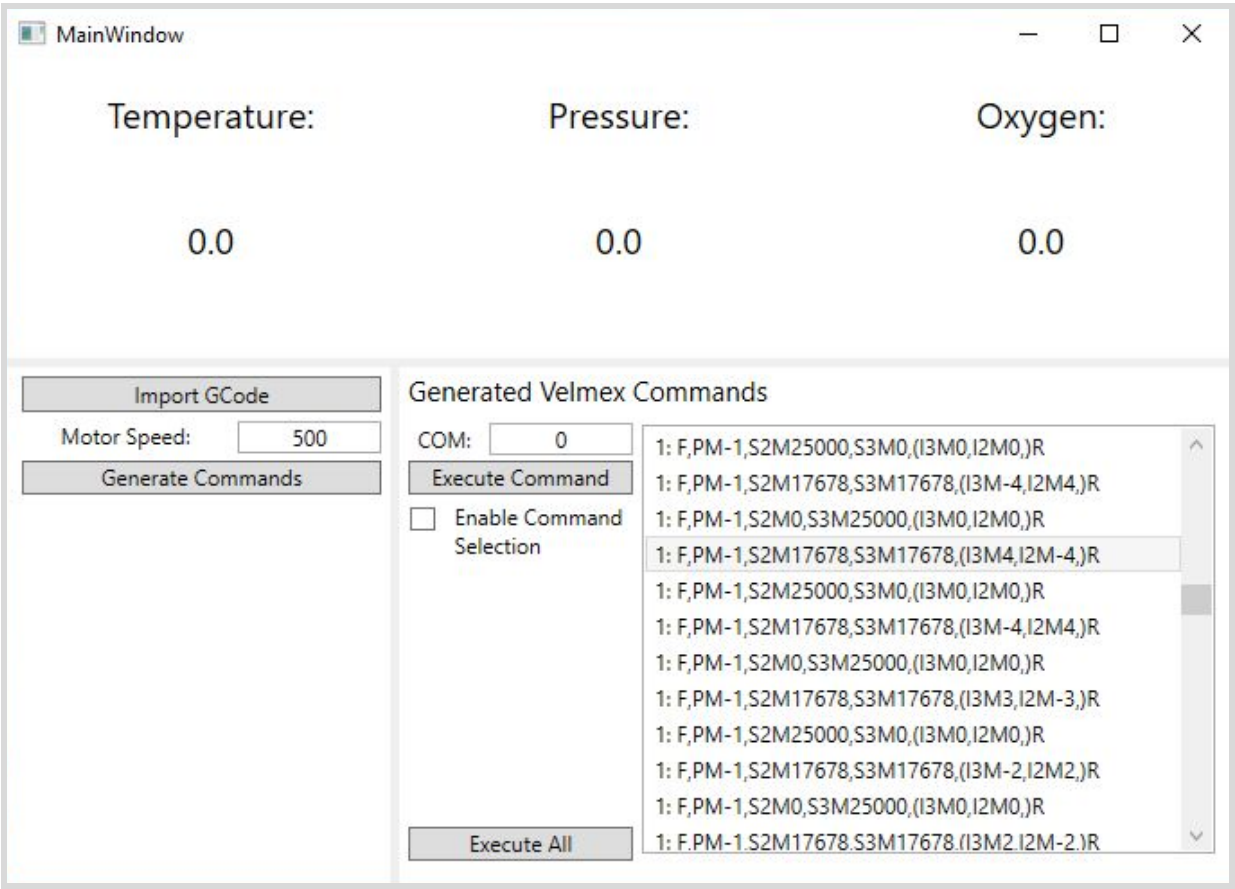


Figure 10. The front end interface for temperature, pressure, and oxygen sensor reading and velmex command generation and execution from G-Code input.

Testing Procedure and Results

Sensor Testing

We tested the sensors through a number of different methods in conditions that weren't always ideal, but provided us enough information to ensure functionality. The easiest sensor to test was the temperature sensor on the Grove barometer. We tested it by taking it to a number of different environments with a known temperature and comparing to the output of the sensor as communicated by the Arduino serial output. We first tested it outside and compared it to the temperature on several weather apps. The day we tested the sensor outside was very cold, but the sensor was still able to accurately display the temperature. We then took it to two different rooms inside of Applied Sciences II and compared the output to the in-room thermostat. The sensor reported accurate results in both instances.

The next sensor we tested was the pressure sensor on the Grove barometer. This sensor was more challenging to test as we can't easily change the pressure in the way we can change the temperature. Our first and most basic test was to simply ensure the regular output of the senor matched the expected value for Ames. The sensor consistently outputted near 1 atm as we would expect. The next test was to ensure it could detect a change in pressure. Even though we had access to a vacuum chamber, there was no way to get readings from the sensor while it was inside. Instead, we put the sensor in one of the release valves as we brought the chamber down to vacuum. This allowed us to see a drop in the pressure readings from the sensor. This result is as we expect since it was experiencing some degree of vacuum.

The final major sensor component is the oxygen sensor. Unfortunately, the oxygen sensor can only be tested in a nitrogen environment. A long list of very specific steps has to be followed during setup to avoid the sensor becoming saturated with oxygen. Then, once the sensor is set up it has to be in a non-oxygen environment or it will become saturated. Due to a delay in getting environmental health and safety approval, we do not have a nitrogen container available in the room. This means that we are unable to test the oxygen sensor without saturating it and rendering it unusable for months. Everything is ready to be tested and used by a future group working on this project.

Software Testing

The software for our project was divided into a few sections, as discussed above in the software implementation details section. The two pieces of software which the most extensive testing was done on are the G-code cube generator, and the code within the G-code interpreter which generates Velmex commands.

The most effective way to test the G-code generator was to first verify the code which generates one infill square is correct, and then verify the code to generate a full layer is correct,

and then verify that the entire .gcode file is correct for the input parameters. We found that using tools to visualize the program outputs was a very effective way of testing the G-code generation without having to involve the Velmex portion of the code at all. Since G-code is a standard for defining machine motion, we were able to use 3rd party software to visualize the generated .gcode files. The first revisions of the generation algorithms had many geometry bugs in their output, which were quickly fixed with the use of these visualization tools.

Testing the Velmex command generation boiled down to ensuring the syntax and timing of our serial commands met the expectations of the motor controllers. Much of this was done during the stage of the project where we were still determining if the Velmex controllers would be a viable solution. The most difficult bug to overcome with the Velmex controllers was achieving coordinated motion of two motor axes. After some trial and error, we were able to find the command structure which worked best for moving both motor axes at the same time in a coordinate manner. There were some small intricacies, such as the order of the motors to move in the command, which ended up being crucial to the command parsing correctly.

Mechanical Results

The final test plan for the slide movement consists of attaching a writing utensil, such as a fine tip Sharpie, to the laser head mounting points and observing the pattern drawn on a piece of paper. This is in lieu of actually turning on the laser, and is done for two reasons. First, it will be important to verify the path without the slides enclosed in the box, as we can not easily just look in the box while it is filled with nitrogen. Second, the laser head (a part outside the scope of our project) is still being manufactured and will not be ready before the end of the semester.

The vacuum chamber will be tested by pressurizing the sealed chamber and applying soapy water to identify leaks.

Related Products and Literature

As additive manufacturing is becoming more and more widely used in industry, there are many examples of commercially available 3D metal printers. In fact, in the early stages of the product we toured a lab at Iowa State which has a printed owned by CIRAS. The printer we saw was a 3D Systems ProX DMP 300 (https://www.3dsystems.com/3d-printers/prox-dmp-300). This printer has a build volume of around 10x10x12 inches, which is much larger than our printer's 4x4x8 centimeter build volume. This printer also prints much faster, as it uses a moving mirror to point the laser rather than a 2 axis motor system. A small mirror can be angled much faster than an entire laser head. However, the end application of our printer is to have several other NDE lasers following the melt laser, which we determined would be much more complex with a rotating mirror based system.

As far as related works which involve the evaluation of a 3D metal printed part during manufacture, we found a NIST journal which describes a project focused on this idea. However, the project we found does not use lasers for the NDE, rather it uses an ultrasonic transducer

embedded in the build plate of the part being printed. The scope of our project did not include design of the NDE system, so it is hard to do a just comparision between this project and ours. However, it is worth noting that this idea is being researched in other ways elsewhere. The following is a link to the journal: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4487290/.

Appendix

Appendix 1- Operation Manual (2-3 pages, step-by-step on how to setup/demo/use system)

G-Code Generator Instructions

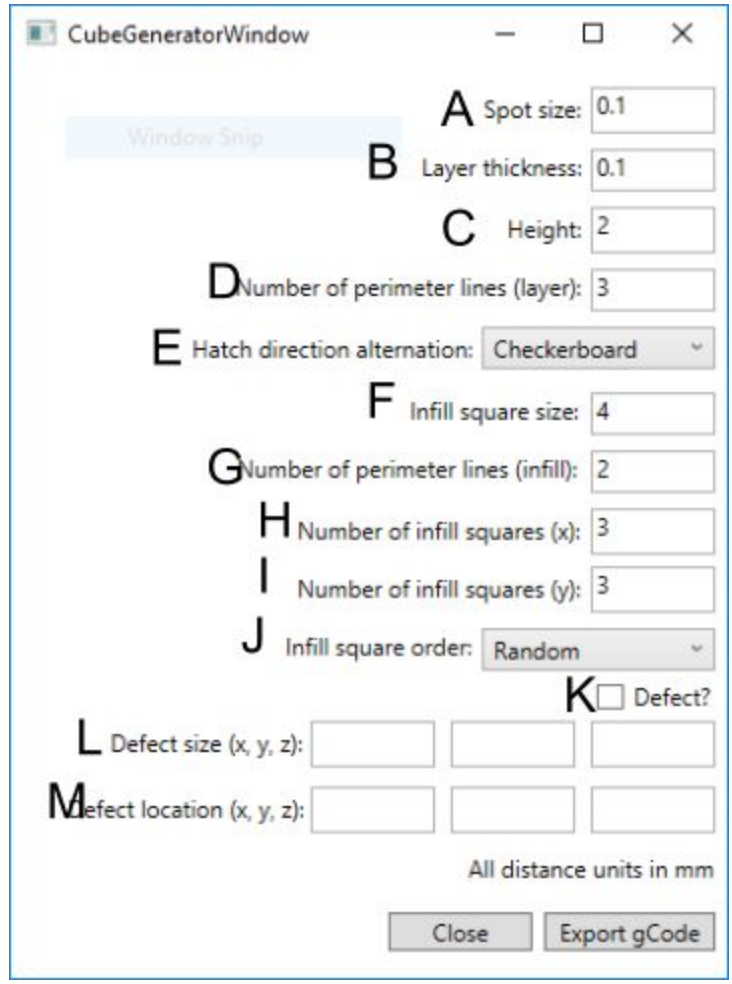
The G-code generator window can be opened by clicking the ‘Cube Generator’ button on the main GUI window. A new window will open, where parameters for the shape to be printed can be specified. After all parameters are filled, clicking the ‘Export gCode’ button will prompt the user for a location to save the .gcode file, and then the file will be generated.

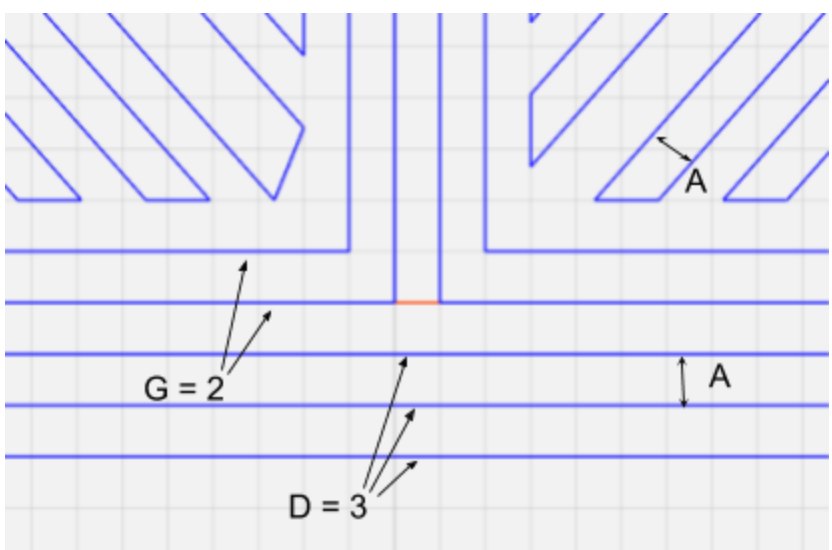
After the .gcode file is generated, it can be previewed by any software package which can interpret G-code. When testing the algorithms, we used https://ncviewer.com/ to visualized the output. It is recommended that you only generate one or two layers (make the part height equal to the layer thickness) when visualizing the output, as the viewer will not interpret the custom layer change command properly.

The following is a description of what each parameter in the generation window means. All distance dimensions in this window are in millimeters.

Feature List

A. Laser spot size
B. Z layer thickness
C. Height of rectangular prism
D. Number of perimeter lines per Z layer
E. How to alternate infill hatch direction (checkerboard, random, same direction)
F. Infill square size
G. Number of perimeter lines per infill square
H. Number of infill squares (x/width)
I. Number of infill squares (y/height)
J. Order of printing infill squares (sequential, random, every other)
K. Defect enabled (if unchecked, following text boxes ignored)
L. Size of defect bounding box (x, y, z)
M. Location of origin of defect bounding box (x, y, z)

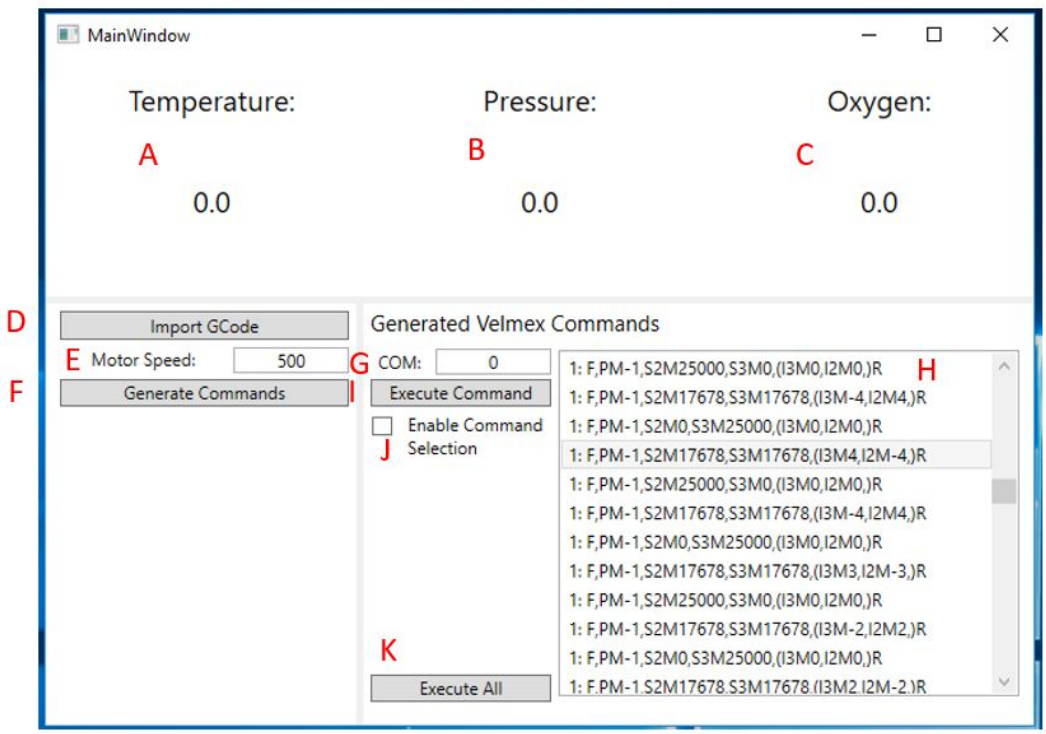




G-Code Interpreter Instructions

The G-Code Interpreter window can be opened by clicking the ‘G-Code Interpreter’ button on the main GUI window. A new window will open, with sensor data if connected. To generate Velmex commands, click ‘Import GCode’ and select your G-Code file then change the speed textbox with a integer value or leave as default. Next if you click ‘Generate Commands’, the scrollable list viewer will be filled with your commands and the first one will highlighted.

To run the commands, you can select ‘Execute Command’ and the stepper motors will receive the highlighted command. You may continue clicking that same button to continue to execute the commands in order. If you need to execute a command that is not highlighted yet, check the box labeled ‘Enable Command Selection’. This allows you select the command in the list viewer and then click ‘Execute Command’ to send your selected command. To send all of the commands from the highlighted command to the end, click ‘Execute All’ and each command will be sent back to back as fast as possible.



A. Temperature Value (℃)
B. Pressure Value (hPa)
C. Oxygen Value (ADC)
D. Opens File Selector for a .gcode file
E. Motor speed value
F. Generates Velmex commands in H
G. COM port number for communication with stepper motors

H. List of generated commands
I. Send highlighted command to velmex stepper motors
J. Enables the ability to select commands in H
K. Sends all commands in H from highlighted command to the end to the stepper motors

Appendix 2- Alternative/ Unused Versions of Design

STL Slicing Software

During the first semester of this project, our team had planned on having the ability for the printer to print any STL format CAD file. This would require us to develop software which could slice any STL file into 2D layers, and then generate infill patterns for those layers.

First semester, we had completed software which as able to slice an STL file into layers. However, we found that generating the infill patterns for these layers would be the most complicated part of the process. Furthermore, we were hitting several roadblocks and complications with the vacuum chamber and mechanical design. Thus, as recommended by the faculty panel at the end of the semester review, we narrowed our scope and removed STL slicing and processing from the final project goals.

The software we developed for the slicing still exists and has been given to our client should future groups wish to continue the development.

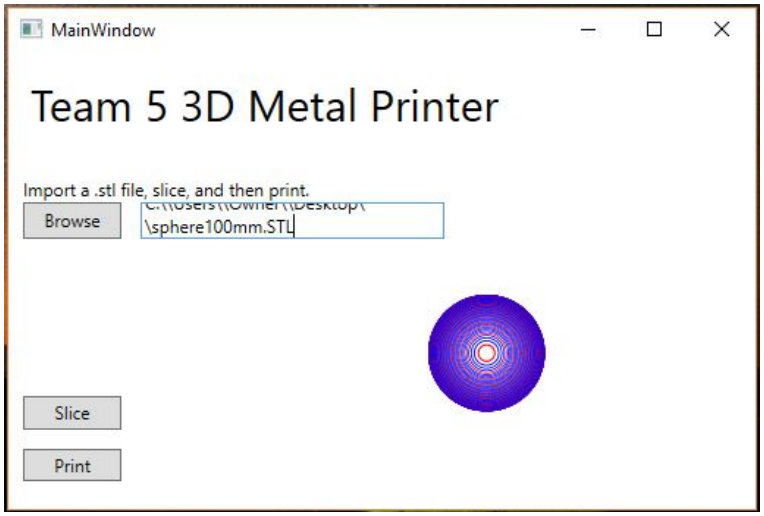


Figure 11. A screenshot of the in progress STL slicing and processing software.

Mechanical Design Changes

Our initial design had a completely different plan for moving the laser. Instead of slides, the team had motors to rotate a mirror to redirect the laser into the correct print location. This

provided a whole set of challenges that we weren't sure were solvable. One of the biggest problems with this design was the changes in power as the laser moved across the print surface. Each time the mirror was adjusted, the laser power would have to be adjusted to account for changes in angles and beam lengths. This would have been a major challenge. We also had concerns about residue buildup on the mirror due to smoke rising off the print surface. This would likely have led to the mirror getting to hot and eventually shattering. We were unable to find a mirror that suitably avoided these concerns. The main advantage of using a mirror over other options would have been an increase in print speed.

Another design idea that eventually got scrapped was a viewing window in the side of the printer. There are numerous advantages to having a window, but the primary one is an ability for a human operator to monitor the print status and take action if need arises. This was primarily scrapped due to laser safety concerns. There are coatings and products out there that block for certain laser wavelengths and even ranges of wavelengths. The problem with this is that the final product after future groups work on it will include three separate lasers. It will include the melt laser that we currently incorporate as well as two lasers used for non destructive evaluation. Each of the three lasers have different wavelengths and thus need to blocked in order to have a window. We were unable to find any kind of a solution that allowed for blocking each of the three wavelengths which resulted in the removal of the window. We then moved to trying to incorporate both thermal and traditional cameras into the design that could send a live feed to the UI. This idea was primarily scrapped due to cost constraints. Thermal cameras that were fast enough to be effective were very cost prohibitive. Traditional cameras that are both vacuum and high temperature approved are also prohibitively expensive.

Initially our project was going to include incorporation of the NDE lasers and software. Upon beginning work with the project it became clear to both our team and Dr. Bigelow that incorporating non destructive evaluation into our scope would be too much for a single year of senior design. This idea was pushed back and is now going to be completed by a future senior design group.

Appendix 3- Project Costs

Table 2: Spending Summary

Item	Cost
Trace Oxygen Sensor (internal sensor)	$1,840.00
Mechanical hardware order (vacuum chamber, frame)	$698.38
Waterjet stock order	$396.03
Sensor order (external oxygen, internal pressure, arduino)	$156.85
Connectors	$41.96
Power strip, wire, wiring accessories	$144.25
Kapton wire	$118.62
Vacuum feedthroughs	$1,455.00
Desktop PC	$400.00
Boyd Lab Labor for Frame Machining	$230.00
Velmex Slides and Motor Controllers	$11,880.00
Vacuum Chamber Labor	$3,925.00
Total	**$21,286.09**

Acknowledgement

We would like to thank Dr. Timothy Bigelow, Associate Professor of Electrical and Computer Engineering at Iowa State. Dr. Bigelow serves as the faculty advisor for this project. He provides guidance, technical advice, and design constraints in each of our weekly meetings. Additionally, the majority of the funding for the project comes from a research grant obtained by Dr. Bigelow.

What we've learned

A project like this consists of many small tasks and challenges that one wouldn't necessarily think of at first. This could include things like purchasing a computer, finding a suitable table, finding ways to not blow fuses with all of our electronics, and many other small issues.