# 5-Axis 3D Printer Software

# Developer Manual

## 

Authors:

Connor Wyatt

CJ Young

Submitted in fulfillment of the requirements for COMP 4710 Senior Design to the

Department of Computer Science and Software Engineering, Samuel Ginn College of

Engineering, Auburn University

Auburn, Alabama

December 4, 2023

Comp 4710 Senior Design Project

Fall 2023

## Table of Contents

[**Table of Contents**](#_fuc69v3uilw8) **ii**

[1. System Metaphor](#_3ojh7mfepl97) 1

[2. System Design Overview](#_f2ndkml6461m) 2

[3. External Items](#_yvm8149nkj2t) 5

[4. G-Code Converter](#_x67i2etsshw6) 6

[5. Slicer](#_hjipfbgagjsv) 9

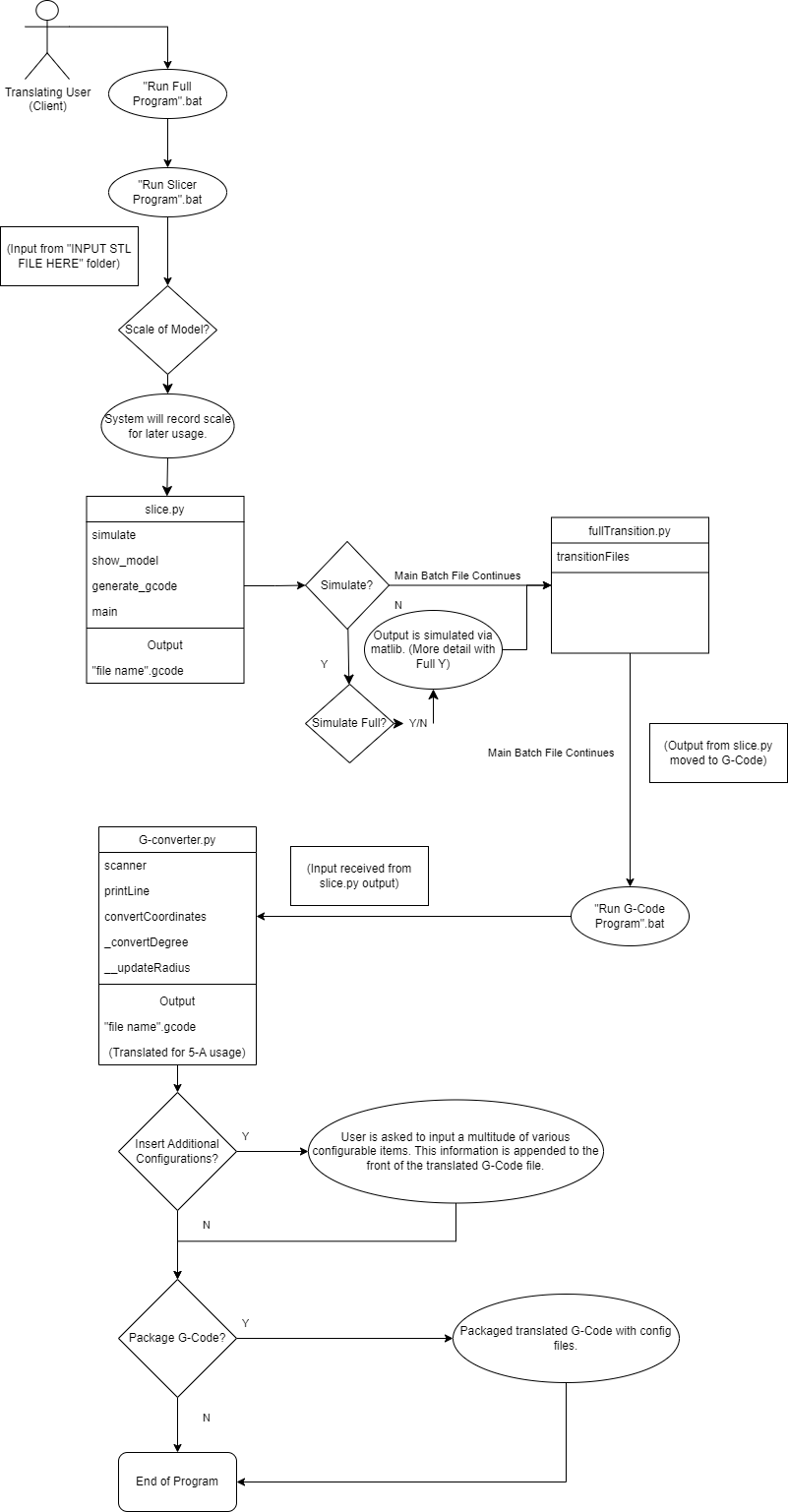
## 

### 1. System Metaphor

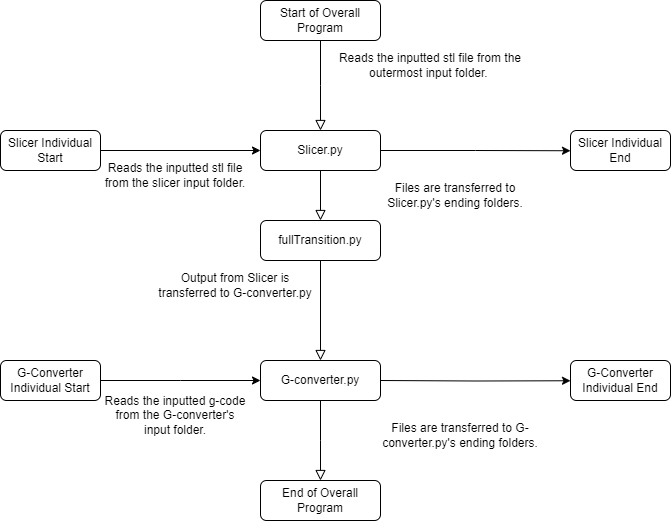
3D Printers are now widely known for their ability to replicate plastic objects by utilizing instructions known as G-Code. However, a standard 3D Printer suffers in a few fields: Larger models require numerous supports, general printers can have a difficult time producing specific shapes, and many shapes are simply infeasible due to the layout of the printer.

The 5-Axis 3D Printer provides a solution to these issues by utilizing a main rotatable support instead of a standard 3-Axis printer. However, this introduces an additional need: the G-Code utilized for standard printers does not correctly translate to the coordinates needed for the 5-Axis printer. This software provides that exact solution, and, by following this manual, you can continue to develop it as viable software. [taken from Cycle 3 Report, Fall 2023]

### 2. System Design Overview



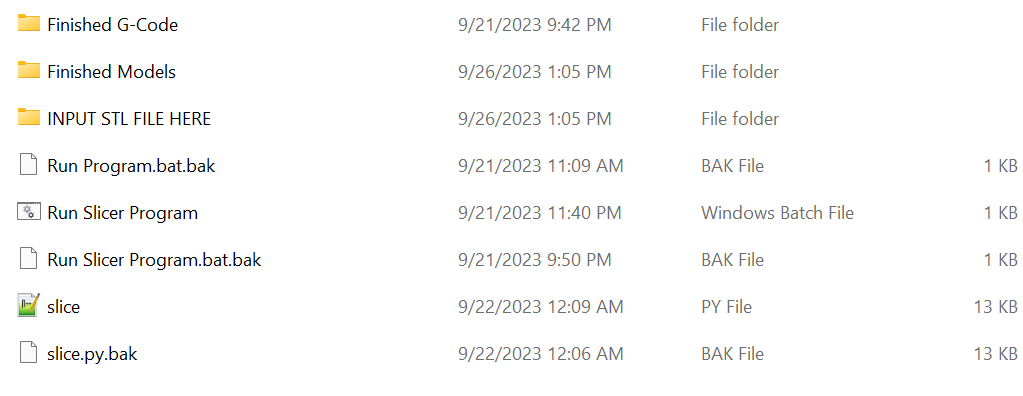
*Figure 1 UML Diagram [taken from Cycle 3 Report, Fall 2023]*



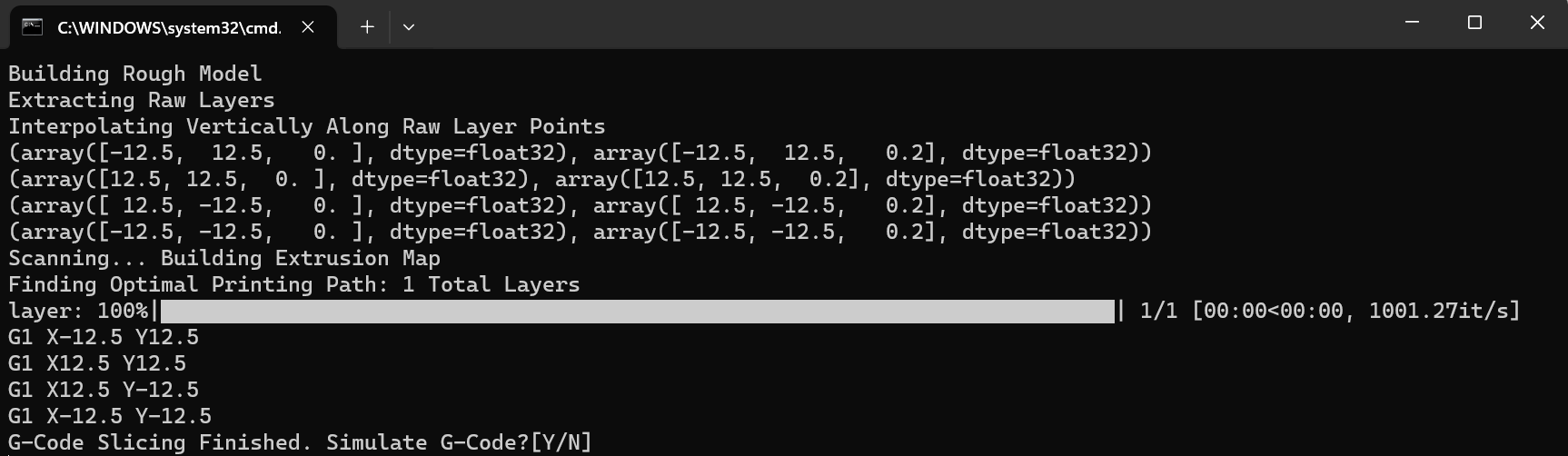
*Figure 2 Flow of the System [taken from Cycle 3 Report, Fall 2023]*

The program is based on a “module” design, allowing for different sections to be utilized separately. This has multiple advantages, a key one for development being that testing individual sections is much easier than testing the entire program.

The program utilizes folders, files, and terminals in order to run various components, and currently does not have any type of UI. There exists older builds that do contain UI, which may be a good avenue of development.

**

*Figure 3 Current External User Interface [taken from Cycle 3 Report, Fall 2023]*



*Figure 4 Current Internal User Interface [taken from Cycle 3 Report, Fall 2023]*

### 

### 3. External Items

The program was built with the intent of making each section usable individually or all together as a combined experience. As such, there are a few external items that stitch the entire program together:

**Run Overall Program (Batch File):** This file will start by swapping its overall input file with the Slicer’s input file, before running the slicer program. It will then swap the files back to their standard position. Afterwards, it will call fullTransition.py in order to transfer the finished standard G-Code to the G-Code Converter. Finally, it will run the G-Code Converter.

**fullTransition.py**: This program will navigate to the Finished G-Code folder in “SLICER” before selecting a given file and translating its file location to the G-Converter’s input folder. Note: It will only select the last file, as it currently iterates through every possible file, setting the name each time.

### 4. G-Code Converter

Currently, the program will navigate and get the current file in its input folder before reading it and modifying the present G-Code to something that works for the 5-Axis 3D Printer. There are a number of functions:

\_**convertDegrees:** Converts between units in G-Code and circular degrees.

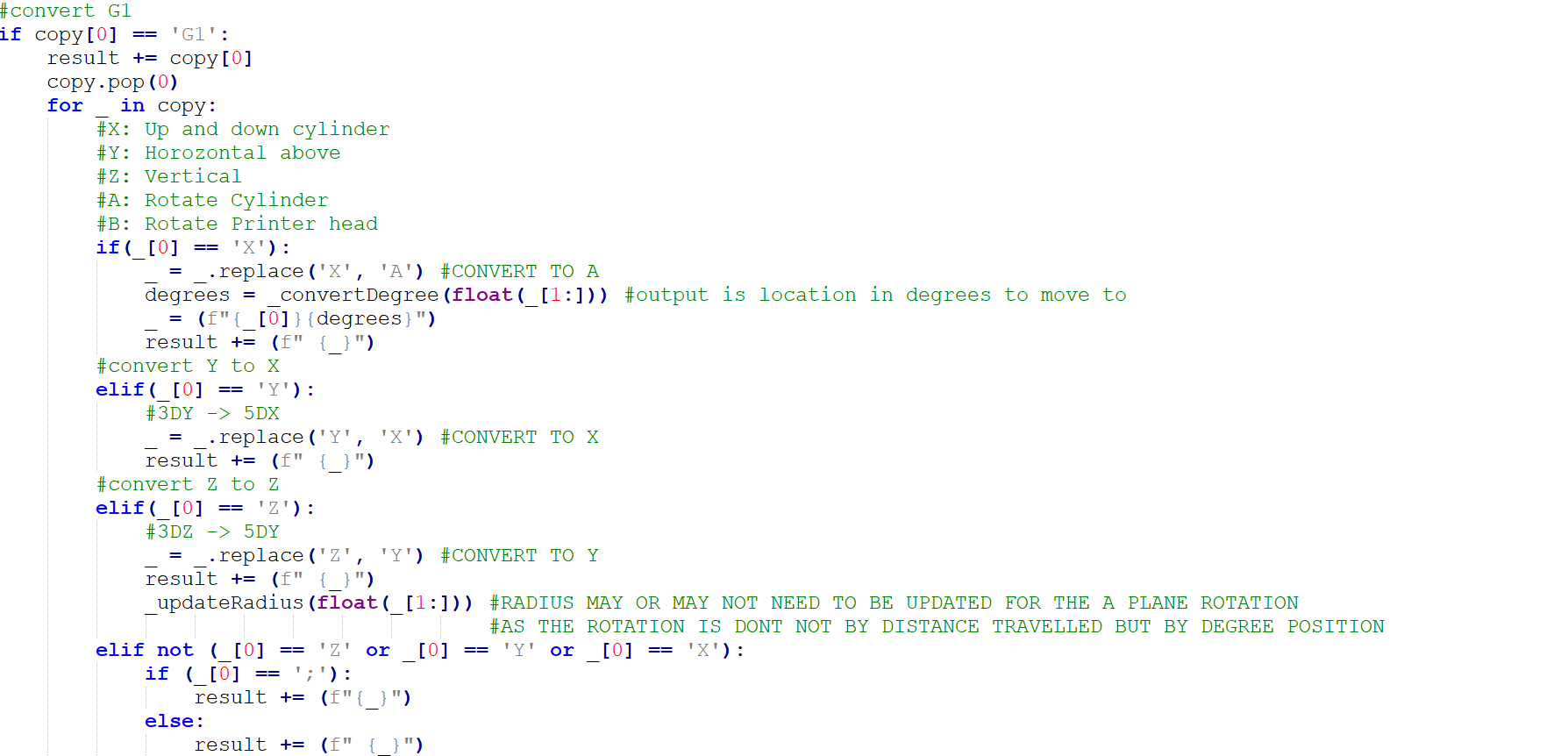
\_**updateRadius:** Keeps a specific “radius” variable updated based on how the Z-axis changes.

**get\_number\_from\_string:** As the name would imply, this function gets a specific number from a given string. Useful for pulling numbers out of G-Code strings.

**printConfig:** An additional feature that allows the user to set a number of important variables with ease. You can choose to fill the default value by simply pressing enter, or skip the input altogether by typing “SKIP”. For more information on the specific variables, please refer to the User Manual’s “G-Code Translation Process” section.

**getInput/getInputArray:** Both methods are designed to help get user input, and provide a number of generic checks to avoid error. getInputArray differs by allowing multiple answers from the user.

**convertCoordinates:** This method will take a single string (our line of G-Code) and run it through some use cases (some will delete the line, others will convert the coordinates, etc) before returning the translated line. There are a number of cases here, but the main ones to keep in mind are the Axis conversion lines, as they will provide the basis for the G-Code’s movement. For some context as to what’s being translated, “The most common command types are M or G commands that begin with the specified letter, G typically means movement of some kind and M is printer setting changes. This is accompanied by the coordinate conversion that I have thought up and done my best to explain in comments in the file. The math for the x -> A conversion is simply finding arc length.” [taken from Spring 2023 User Manual]



*Figure 5 convertCoordinates Axis Conversion Code*

**scanner:** The overall method for the system, which calls each individual line into printLine.

**printLine:** Given the lines from the scanner method, this method takes the line, runs convertCoordinates on it, and appends the returned line to our translated G-Code file. This will eventually be sent to its own config file or the “G-Code (Translated)” folder, depending on other methods.

**Future Development:** The next steps of development would involve performing testing on the existing code and ensuring that everything works well within the confines of the physical printer. On top of standard G-Code testing, you may also want to try utilizing a 3D slicer model as input for this system. Given that this is the vital section for the 3D Printer’s usage, it may be wise to utilize an out-of-house slicer for initial translation testing. To give an example, We utilized Ultimaker Cura for comparison testing.

### 5. Slicer

Slice.py’s purpose is to convert an STL file into a G-code file which can then be converted into 5-Axis G-Code. The slicer is constructed with multiple steps in mind. These steps are:

1. Read the STL file.
2. Extract the data points/vertices from the STL model.
3. Interpolate the extracted model or ‘raw’ model.
4. Scan the interpolated points from the printing surface to create printing layers based on physical parameters.
5. For each layer compute a path finding algorithm and return an extrusion path which will be a list of arrays of points.
6. Finally, convert the lists of points to G-code.

[Above Excerpt Constructed from Spring 2023 User Manual]

For more information on the operation of the program from the front-end, refer to “Slicer/Simulation Process” in the User Manual.

There are a large number of methods within this program, but we will touch upon the most important:

**get\_verticies:** pulls the vertice points directly from the model.

**get\_raw\_layers:** Collects points to be within each z-axis layer.

**get\_line\_points:** Collects the raw layers and condenses them into collective layers that split at 0.5. This helps with printer consistency.

**get\_intersections:** Gets the pathing for individual layers checks to ensure direct intersections do not occur.

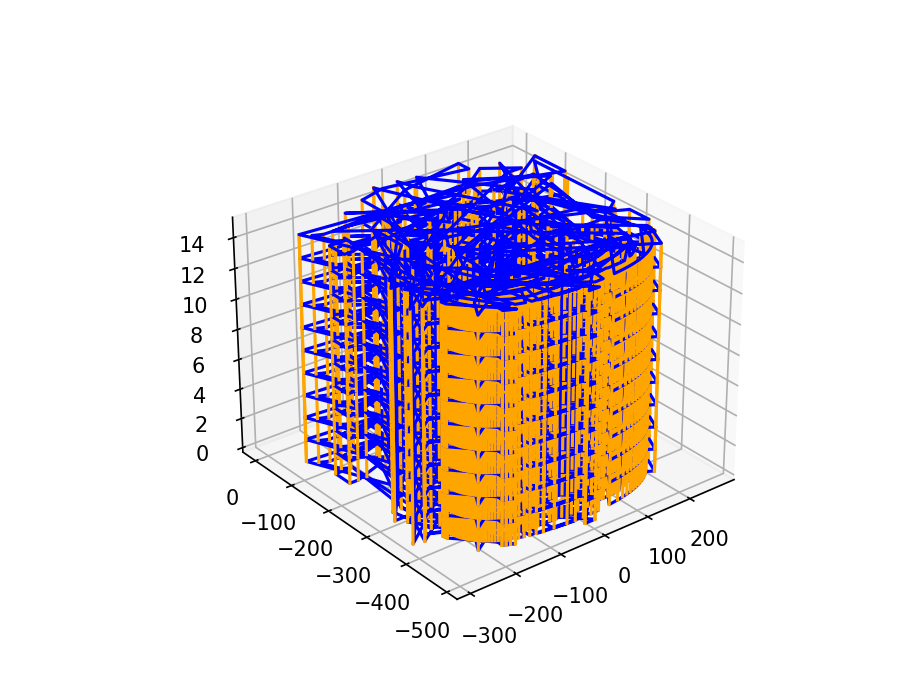
**two\_opt:** Calculates the pathing of each layer, utilizing “limited dijkstra”, which gives each node a path to the next 10 nodes.

**do\_dijkstras\_on\_layers:** Performs the two\_opt function on every layer, and provides a progress bar for timing the performance of the program.



*Figure 6 do\_dijkstras\_on\_layers Progress Bar*

**simulate**: Simulates the printing path in 3D space using the pre-created extrusion path [Spring 2023 User Manual]. An additional element of connections between the layers can be added.



*Figure 7 simulate() example [taken from Cycle 3 Slides, Fall 2023]*

**generate\_gcode**: Generates the G-code using the pre-created extrusion path. [Spring 2023 User Manual]

**Future Development:** The next way to take this program will be to test its slicing capabilities on the actual system. There are also a number of models that can test the slicer’s capabilities (if not provided, request them from sponsor). It may also be wise to take an in-depth look at the provided functions if testing proves difficult.