Construction of Control system for syringe dispenser based on Printrbot 3D printer

Qi An

Master's Thesis
Main field of study: Electronics
Credits: 30
Semester/Year: 2/2019
Supervisor: Henrik Andersson
Examiner: Bengt Oelmann
Course code/registration number: EL038A
Abstract

3D printers require a reliable and robust control system to provide the proper quality for printed parts. Dispenser 3D printers are widely used in various fields of scientific research. The goal of this project is to build a dispenser 3D printer based on Printrbot 3D printer, design and implement the control system and software. This system was able to control the dispenser, performed the correct operation according to the instructions. The operating system was built by LabVIEW for file reading and printer control.

Keywords: 3D printer, Dispenser, Control System, Operating System.
Acknowledgements

I am very grateful to my supervisor, Henrik Andersson. Without his help, I could not finish this project. It is very useful in the initial explanation of the basics of 3D printers. He always helps me when I am in trouble. When designing the operating software, he suggested me to use LabVIEW, which is very effective and easy to use. He gives me a view on how to finish the whole part of the operating system. Thank you very much for your selfless help.
Table of Contents

Abstract ............................................................................................................... iii
Acknowledgements .......................................................................................... iv
Table of Contents ............................................................................................ v
Terminology ......................................................................................................... viii
Acronyms .............................................................................................................. viii
Mathematical notation ......................................................................................... viii
List of Figure ........................................................................................................ x
List of Table ......................................................................................................... xii

1 Introduction ...................................................................................................... 1
  1.1 Background and problem motivation ......................................................... 1
  1.2 Problem formulation .................................................................................. 2
  1.3 Scope ........................................................................................................... 2
  1.4 Concrete and verifiable goals ..................................................................... 2
  1.5 Outline ......................................................................................................... 2

2 Theory ............................................................................................................... 3
  2.1 Construction of Printrbot Simple Metal .................................................... 3
  2.2 Operating system ....................................................................................... 4
  2.3 Basic formula ............................................................................................. 4
    2.3.1 Stepper Motor Speed .......................................................................... 4
    2.3.2 Current Regulation .............................................................................. 6
    2.3.3 Running distance of control system .................................................... 6

3 Model ............................................................................................................... 8
  3.1 Toolchain of 3D printing ............................................................................. 8
  3.2 Overall composition ................................................................................... 8
  3.3 Structure of control system ....................................................................... 9
  3.4 Main program control system ................................................................... 9
    3.4.1 The format of commands ................................................................... 10
    3.4.2 G-code classification ......................................................................... 11
    3.4.3 Get commands .................................................................................. 12
    3.4.4 G-code analysis ................................................................................ 13
### 3.4.5 Process commands

3.4.6 Activity management

#### 3.5 Planner

3.5.1 Planning a straight path

3.5.2 Trapezoidal curve speed recalculation

#### 3.6 Motion control system

3.6.1 AVR Timer/Counter

3.6.2 Interrupts

3.6.3 Application in motion control systems

3.6.4 Three-axis linear motion

### 4 Implementation

4.1 Control board

4.2 Dispenser

4.3 Stepper motor

4.4 Microstepping driver

4.5 Endstop

4.6 Power supply

4.7 Final assembly

### 5 Software interface

5.1 Interactive function

5.2 Layout of the operation panel

5.3 Implementation of programming

5.3.1 Block diagram

5.3.2 Block diagram design

5.3.3 Program debugging

### 6 Calibration and analysis

6.1 Feed rate adjustment

6.1.1 Printing Material

6.1.2 A straight line testing

6.1.3 A cube testing

6.1.4 A five-pointed star testing

6.2 Communication rate

6.3 Application on printed electronics technology

6.3.1 Limit parameter

6.3.2 Printing on circuit track

6.4 Result and analysis

### 7 Conclusions and future work
7.1 Control system and interface .............................................55
7.2 Social and environmental aspects ........................................55
7.3 Future work .....................................................................55

References .............................................................................57

Appendix: Block Diagram of LabVIEW ....................................62
  Operation Panel ...................................................................62
  File Reader ..........................................................................63
  Instruction Input and Feedback Review .................................63
Terminology

Acronyms

LabVIEW    Laboratory Virtual Instrumentation Engineering Workbench.
RAMPS     RepRap Arduino Mega Pololu Shield.
SPM       Steps Per Millimeter.
SPR       Steps Per Revolution
BP        Belt Pitch
PT        Pulley Tooth Count
STP       Screw Thread Pitch
ERatio    Extruder gear Ratio
BD        Extrusion wheel screw Diameter
WGM       Waveform Generation Mode bits
CS        Clock Select bits
COM1A     Compare Match Output A Mode bits
COM1B     Compare Match Output B Mode bits
TIMSK1    Timer Interrupt Mask Register
OCF1      Output Compare Flag
TCN1T     Timer/Counter

Mathematical notation

Symbol     Description
$f_{step}$      Frequency of a square wave
$v$        Speed of motor
<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n_m$</td>
<td>Microstepping level of motor driver</td>
</tr>
<tr>
<td>$\theta_{step}$</td>
<td>Motor full step angle</td>
</tr>
<tr>
<td>$xV_{REF}$</td>
<td>Reference potential voltage of motor driver</td>
</tr>
</tbody>
</table>
List of Figure

Figure 1 The structure of Printrbot......................................................... 3
Figure 2 The toolchain of 3D printing...................................................... 8
Figure 3 The Overall Structure and Schematic of The Dispenser 3D Printer ................................................................................................. 9
Figure 4 The Flow Chart of Main Program ......................................... 10
Figure 5 The G-code File of a Star shaped 3D object. ..................... 11
Figure 6 The Beginning of a G-code file .............................................. 11
Figure 7 The Switch Structure in Main Program ............................. 14
Figure 8 The Code of Activity Management................................. 15
Figure 9 The Flow Chart of Planner ..................................................... 16
Figure 10 Flow Chart of Motion Control ............................................. 19
Figure 11 The Bits of TCCR1A[29]...................................................... 20
Figure 12 The Bits of TCCR1B[29] ...................................................... 20
Figure 13 The Bits of TIMSK1[29] ...................................................... 21
Figure 14 The Principle of Bresenham’s Line Algorithm[30]............. 23
Figure 15 The Flow Chart of Motion Control Method ..................... 24
Figure 16 The Control Board with Arduino and RAMPS ............... 25
Figure 17 The Extruder of Printrbot[32] ............................................. 26
Figure 18 The Dispensing System of Paste Dispenser 3D Printer .... 26
Figure 19 Two Phase Hybrid Stepping Motor[33] ............................. 27
Figure 20 The Schematic Diagram of The Stepper Motor[34] .......... 28
Figure 21 The Schematic of The DRV8825 Driver Connection [26] . . 28
Figure 22 The Basic Pulse Diagram [26] ............................................... 29
Figure 23 The Setting of Mircostep Resolution ............................... 29
Figure 24 The Schematic Diagram of Endstop Connection [35] ....... 30
Figure 25 The Proximity Sensor ............................................................ 30
Figure 26 The Connection of Proximity Sensor[36] ........................... 31
Figure 27 The Connection of the Power Supply ................................. 32
Figure 28 The Entire Electrical Assembly ............................................ 32
Figure 29 The Front Interface of Control System .............................. 34
Figure 30 The Final Block Diagram of File Reading .......................... 36
Figure 31 The Final Block Diagram of Button Control .................... 37
Figure 32 The Setting of VISA Configure Serial Port ......................... 38
Figure 33 The Setting of VISA Write and VISA Read ....................... 38
Figure 34 The Main Loop of the Block Diagram ............................... 39
Figure 35 The Edit Panel of the Event Structure ................................. 40
Figure 36 The I/O File Diagram ............................................................ 41
Figure 37 The Highlight Execution Button .......................................... 42
Figure 38 A Straight-line Test in Different Feed Rate ....................... 45
Figure 39 A Square Test in Different Feed Rate ....................46
Figure 40 Part of the G-code file of a Square .....................47
Figure 41 A Five-pointed Star Test in Different Feed Rate ..........48
Figure 42 The Side View of Two Layers Star .......................49
Figure 43 The Side View of Three Layers Star ......................49
Figure 44 The Side View of Four Layers Star .......................49
Figure 45 The Side View of Five Layers Star .......................50
Figure 46 The Circuit Trace with Silver Ink .......................51
Figure 47 The result of Parameter Test ...............................52
Figure 48 The Printing Result on Tracks ............................53
List of Table

Table 1 Stepping Format of DRV8825 .......................................................... 5
Table 2 Main Function of Control Program .................................................. 9
Table 3 Essential G-code and Corresponding Features .............................. 12
Table 4 Important Parameters in the Structure .......................................... 16
Table 5 The Setting of Timer ..................................................................... 21
Table 6 Common Substance Visosity[42] .................................................... 45
Table 7 The Effect of Different Feed Rate ................................................... 48
Table 8 The effects of silver ink printing ..................................................... 53
1 Introduction

1.1 Background and problem motivation

Additive Manufacturing sometimes called 3D printing is an emerging technology [1]. It is about building 3D objects using plastic and other materials from a digital design. As a technology, it has been around and in use by engineers and designers for more than 30 years. 3D printers have evolved, from giant printers to small desktop printers that can be used from the comfort of one’s home and office [2]. Nowadays, due to the lower cost and technological advances [3], the usage of 3D printers is increasing year by year, more and more individuals own and use 3D printers to create their items.

The types of printers usually categorized as filament printers [4], powder bed printers [5][6] or resin printers [7], these printers are based on the different 3D printing methods which were developed to build 3D structures and objects. In filament printers, a plastic thread is melted and extruded from a nozzle. In powder bed printers a μm sized powder, usually plastic but also metal, is melted using a laser, and in resin printers, a liquid resin is cured by a laser or an array of LEDs.

One of the 3D printing methods is using the viscous pastes of materials to build the objects which are not a common way in 3D printer area. However, it has several advantages, one of which is that as an paste-extruded 3D printer [8], the materials can be used in a paste form, these paste-like materials can minimize the limitation of printed materials and greatly increase the application range and practicality of 3D printing. Paste-like materials, such as graphene [9] and silver inks [10], have very high research value in scientific research. This viscous material can be easily changed to various shapes or applied to various scenes using a dispenser 3D printer. Different materials are widely used in various fields relying on paste-dispensed 3D printers, such as design of 3D battery [12], tissues engineering [11], cell culture [13][17], living tissue [14], food [15], medical implants [16][18-20], ceramics, cement [22] and composite materials [17]. It is an important research tool that provides new possibilities for research methods.
1.2 Problem formulation
The overall aim of this project is to reconstruct an already available filament 3D printer to a paste-dispensed 3D printer which with the syringe dispenser, build a system to control the printer and test print a simple object.

1.3 Scope
The study has its focus on building the control system of an dispenser 3D printer, in the design of the printer’s motion trajectory, the curve motion is not considered. The printer moves along a straight line and travels the route.

1.4 Concrete and verifiable goals
There are several goals should be achieved: T1: Determine if the existing control card in the printer can be used, reprogrammed or be replaced, or an additional card is added. T2: Replace or add cards as needed, make any necessary electrical connections and modifications of the printer. T3: Make a control program that is programmed into the controller card that accepts G-code commands which are standard command set for 3D printers. T4: Make a program that can control the x and y-axis stepper motors to set the position of the printer table and the y-axis and syringe dispenser motors to move the syringe and dispense paste. T5: Make a program that is able to read a G-code file prepared from a standard 3D cad STL file and print the content of this file using the syringe dispenser. T6: Verify the system and make a simple printing test.

1.5 Outline
The rest of the report is organized in different chapters as follows: Chapter 2 shows the theory used in this project. Chapter 3 describes the methodology of how to build the control system. Chapter 4 presents the implementation and design of the 3D printer. Chapter 5 illustrates the solution of software interaction interface for manipulating printers. Chapter 6 includes the calibration of the parameters and the result of the final printing test. The last chapter covers the conclusion of the project and the future work part that shows the approach for improving the algorithm and firmware code.
2 Theory

2.1 Construction of Printrbot Simple Metal

The entire project used Printrbot Simple Metal as a prototype for the 3D printer which was rebuilt to remove the previous filament printhead and replaced with a paste dispener system. The overall structure is shown in Figure 1.

![Figure 1 The structure of Printrbot](image)

The paste dispenser 3D printers consist of mechanical and electronic components.

The mechanical part mainly forms the frame of the entire printer, including the dispenser [2], print bed [2], stepper motor and endstop [23]. The dispenser uses a syringe which is filled with the paste-like material and the dispensing of the material is controlled by squeezing the plunger. Endstop is used to specify the print area and as the origin of each coordinate axis. The stepper motor is an important part of the 3D printer. By controlling the rotation of the stepper motor, the printer can move along each axis. The 3D printer uses timing belts and pulleys as its transmission along its X and Y axes to provide fast and accurate positioning and to position the Z axis with higher precision using a threaded rod or lead screw.

The electronics are used to control these mechanical parts. The RAMPS [28] consists of a separate Arduino Mega [24] microcontroller board and a specially designed shield that plugs on top of the Arduino. The Arduino
Mega provides the brain of the platform, while the shield provides switching hardware for the endstops, along with the option to interface with up to five stepper driver boards. The stepper motor driver is responsible for the control of the stepper motor.

The function of a paste-dispensed 3D printer is that the printer can move linearly in three directions by controlling the stepper motor of the corresponding coordinate axis. The printer can accept, interpret and execute its standard command G-code. And finally print out a complete graphic.

2.2 Operating system

The operation interface is a very important part to control the printer. In this project, LabVIEW was used to build a software interface. The LabVIEW program is a virtual instrument (VI) that looks and operates like a real physical instrument [25]. VI uses functions to get information input from channels such as the user interface, and then display or transfer the information to a computer. By constructing the operator interface, the printer can be intuitively manipulated to achieve simple functions. By tapping the buttons on the operator interface, the printer can move linearly along each coordinate axis, and the distance of each movement can also be changed. Most importantly, the operating system can read the G-code file, a project file that stores the print instructions, for complete printing.

2.3 Basic formula

Here are some mathematical formulas described that must be understood to calculate and adjust related parameters.

2.3.1 Stepper Motor Speed

The first step in configuring the DRV8825 [26] requires the desired motor speed and microstepping level. If the target application requires a constant speed, then a square wave with frequency $f_{\text{step}}$ must be applied to the STEP pin.

If the target motor startup speed is too high, the motor will not spin. Make sure that the motor can support the target speed or implement an acceleration profile to bring the motor up to speed.

For a desired motor speed ($v$), microstepping level ($n_m$), and motor full step angle ($\theta_{\text{step}}$), the frequency can be calculated by Equation 1,
θ step can be found in the stepper motor data sheet or written on the motor itself.

For the DRV8825, the microstepping level is set by the MODE pins and can be any of the settings in Table 1.

Table 1 Stepping Format of DRV8825

<table>
<thead>
<tr>
<th>MODE2</th>
<th>MODE1</th>
<th>MODE0</th>
<th>STEP MODE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Full step</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1/2 step</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1/4 step</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>8 microsteps/step</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>16 microsteps/step</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>32 microsteps/step</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>32 microsteps/step</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>32 microsteps/step</td>
</tr>
</tbody>
</table>

Higher microstepping will mean a smoother motor motion and less audible noise, but will increase switching losses and require a higher $f_{step}$ to achieve the same motor speed.

In this thesis, the highest microstepping was selected which is 32 microsteps/step, and the $\theta_{step}$ is 1.8 degree/step.

Therefore, the formula between desired speed ($v$) and frequency of a square wave $f_{step}$ will be further modified to Equation 2,
2.3.2 Current Regulation

In stepping motors, current regulation is used to vary the current in the two windings in a semi-sinusoidal fashion to provide smooth motion.

The reference voltage was calculated to set the current regulation. This step is not very complicated but absolutely necessary to protect the stepper motor and the driver. The motor can draw more current than it or the driver can handle without an appropriate current limit, this is likely to damage one or both of them.

To set the current limit I measured a reference voltage and adjusted the on-board potentiometer accordingly.

This quantity depends on the \( xV_{\text{REF}} \) analog voltage and the sense resistor value (RSENSE). During stepping, \( I_{\text{limit}} \) defines the current chopping threshold (ITRIP) for the maximum current step. The gain of DRV8825 is set for 5 V/V.

The current limit is calculated in Equation 3,

\[
I_{\text{limit}} = \frac{xV_{\text{REF}}(V)}{A_v \times R_{\text{SENSE}}(\Omega)} = \frac{xV_{\text{REF}}(V)}{5 \times R_{\text{SENSE}}(\Omega)} \tag{3}
\]

2.3.3 Running distance of control system

In the 3D printer control system, the printing total distance will be controlled to adapt to different position of the printing scenarios. Since the 3D printer has many mechanical transmission structures like timing belts, pulleys, threaded rod, or lead screws. In order to be able to move the stepper motor and print the item at the specified position, the conversion factor between the motor step size and the final moving distance will be calculated.

The mechanical structures of X-axis and Y-axis are the same, but different from Z-axis, so they have different calculation methods.
The Equation 4 used to calculate this key relationship about timing belt system like X and Y-axis,

\[
SPM_{X\text{ and } Y} = \frac{SPR}{n_m} \cdot \frac{BP}{PT}
\]  

(4)

The Equation 5 present the formula in leadscrew driven systems Z-axis,

\[
SPM_Z = \frac{SPR}{n_m} \cdot \frac{BP}{PT}
\]

(5)

Where SPM is the steps per millimeter of each axis motor; SPR is the steps per revolution; \( n_m \) is the microstepping level; BP is the value of Belt Pitch; PT is the number of Pulley Tooth; STP is the value of Screw Thread Pitch.

The dispenser system has a different system with relative complicated. The relationship of extruder system will be expressed in Equation 6,

\[
SPM_E = \frac{SPR}{n_m} \cdot \frac{ERatio}{\pi \cdot BD}
\]

(6)

Where ERatio is the ratio of dispenser gear; BD is the extrusion wheel screw diameter.
3 Model

This chapter will focus on the entire printing system and the method of building control system, some key parts of main and motion program, and the basic function of the system.

3.1 Toolchain of 3D printing

The general workflow wouldn’t be possible without a series of applications and hardware that will be used at each stage of the process.

The general workflow of a standard 3D printing process is shown in the Figure 2.

![Figure 2 The toolchain of 3D printing](image)

Referring to this simplified illustration, the printer control application will bring in the 3D model and send it to a slicer application, if needed. The printer control then communicates with firmware, which is a specialized set of code, which runs on the electronics platform. The firmware controls the electronics hardware to build 3D objects according to the instructions received from the printer control. Building firmware is critical to the overall project.

3.2 Overall composition

The printer system is mainly composed of a printer, a control board and a power supply. The power supply supplies power to the control board and various components in the printer, the control board controls the printer to complete the printing task by properly connecting the control board to the various components of the printer. The overall structure of
paste-dispenser 3D printer and the schematic of connection are shown in Figure 3.

![Figure 3 The Overall Structure and Schematic of The Dispenser 3D Printer](image)

### 3.3 Structure of control system

The whole system uses the Arduino IDE writing environment and is written with the support of Arduino open source hardware. The advantage of this open source hardware is to provide the driver of the peripheral part of the hardware, the call is simple, and supports the C++ writing environment.

The entire system consists of two parts, the main program part and the motion control part. These two parts constitute the entire control system.

### 3.4 Main program control system

The main control program is mainly responsible for the following functions that are shown in the Table 2.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Communicate with the host computer to obtain the G command</td>
<td></td>
</tr>
<tr>
<td>G command analysis, distinguish between instruction content and command parameters, and convert the parameters into integers</td>
<td></td>
</tr>
<tr>
<td>Classification execution of G command</td>
<td></td>
</tr>
<tr>
<td>Limit switch control and activity management</td>
<td></td>
</tr>
</tbody>
</table>
The flow chart of the main control program is as following Figure 4 shown,

![Flow Chart of Main Program](image)

**Figure 4 The Flow Chart of Main Program**

### 3.4.1 The format of commands

The 3D printing control system use a standard machine language G-code [27], to let computerized machine tools know how to do something. G-code consists of G-commands that have an assigned movement or action.

The basic G-code string contains several parts, the usage looks like this,

G0[F<rate>] [X<pos>] [Y<pos>] [Z<pos>] [E<pos>]

The instruction is preceded by a capital letter G followed by a number indicating which G-code command is this. The parameters of the command can be modified, [F<rate>] represent the maximum movement rate of the dispenser move, [X<pos>] means the coordinate on the X axis, the Y and Z axis parameters can also be changed by the same method. [E<pos>] indicates the dispensed material capacity. The real G-code file is shown in Figure 5.
Figure 5 The G-code File of a Star shaped 3D object.

The G-code command in the figure describes a five-pointed star. The red square in the figure is a standard G-code, which means that the printer moves 62.871mm along the X axis and 57.317mm along the Y axis. The plunger pushes 1.06588mm to dispense the corresponding volume of paste material. However, since these values are too small, it is impossible to print such precision and accuracy. So, the control system first converts these decimals into integers.

3.4.2 G-code classification

G-code files usually have a fixed header that is used to describe initialization settings, such as zeroing each axis, setting the dispenser position to the starting position, and some related parameters for printing.

The Figure 6 is the beginning of a five-pointed star G-code file.

Figure 6 The Beginning of a G-code file

The G28 usually appears at the very beginning of the file, used to place the printer in the start position, and G92 appears later, usually used on dispenser to zero the dispenser position.

A simple 3D printing can be done with three types of G-code, the control system was designed to be able to interpret and execute these types of G-code that are described in the Table 3 below.
Table 3 Essential G-code and Corresponding Features.

<table>
<thead>
<tr>
<th>The type of G-code</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>G0 and G1</td>
<td>Linear Move</td>
</tr>
<tr>
<td>G28</td>
<td>Auto Home</td>
</tr>
<tr>
<td>G92</td>
<td>Set Position</td>
</tr>
</tbody>
</table>

G0 and G1 are the linear move commands, after receiving one of these commands the printer will do a linear movement. A linear move traces a straight line from one point to another, ensuring that the specified axes will arrive simultaneously at the given coordinates. This is the most important G-code because the G-code file which is executed eventually is composed of many linear motions.

G28 is used to clear the position of each axis, it will auto-home one or more axes, moving them towards their endstops until triggered. After executing this G-code the coordinate information of each position will be set to zero. This is critical for the entire system, otherwise the entire coordinate position will not be set because there is no reference origin.

The role of G92 is to set the current position. Normally the entire printed coordinate system should be built before the print job begins. With the G28 command, the origin position was established, the position of the dispenser motor is normally initialized by the G92 command. Set the current dispenser position to the initialization position with G92 E0.

3.4.3 Get commands

Getting the instruction is the beginning of the whole system. 3D printing is an instruction-driven system. The control system accepts the instructions and executes them one by one to form the entire printing process.

The Serial function is used for communication between the Arduino board and a computer or other devices.

Serial.begin() is the function to initialize the serial port, the data rate should be set in bits per second (baud) for serial data transmission. The
default baud was set to 115200 in the communication to ensure that the data is transferred fast enough and that no data is lost.

`Serial.available()` is used to get the number of bytes (characters) available for reading from the serial port. This is data that’s already arrived and stored in the serial receive buffer. It will return the number of bytes available to read. Therefore, it is usually used as a condition for judging, when it is true, it means that data is passed in.

`Serial.read()` refers to fetching and reading a byte data from the buffer of the serial port.

Both `Serial.print()` and `Serial.println()` are described as the function of printing data to the serial port as human-readable ASCII text. The difference is that the latter will follow a carriage return character and a newline character. These commands can take many forms. Numbers are printed using an ASCII character for each digit. Floats are similarly printed as ASCII digits, defaulting to two decimal places. Bytes are sent as a single character. In this project, all data is transmitted as a string.

### 3.4.4 G-code analysis

The next step is a relatively critical step for the entire system, which is the analysis of the G commands. Since the instructions are mainly composed of letters and numbers, the control system need to distinguish in one G command string which are letters, and which are numbers.

Not only that, there may be parameters in a single instruction that are not required by the 3D printer, such as comments. The control system will separate the comments from the G-code instruction, and find the key content.

In the process of interpreting the entire instruction, the letter G is first found. Based on the structure of the instruction, the number after the letter G represents the sequence number of the execution action. After the function is recognized, the system will look for the letters X, Y, Z or E to find the corresponding motor that controls the axis. The numbers following it indicate the distance moved by the motor rotation.

In order to find this important letter, a pointer was used, the function `strchr()` will help the pointer to find the desired letter, and the pointer will point to the position of the letter. Because of the construct of the command,
the numbers that determine the function and distance of movement are also found, because numbers are always followed by letters. Then the function `strtod()` converts the string to a floating point number.

### 3.4.5 Process commands

The processing and execution of instructions is the core of the entire main program. In this part, the system should respond to different actions according to different command.

The Switch…Case… structure is very suitable for this requirement which is shown in Figure 7. The system finds the letter G from an instruction and performs actions corresponding to different numbers.

![Figure 7 The Switch Structure in Main Program](image)

When the G0 or G1 command is executed, because it is an action command, the system determines the action direction and distance of the action through a series of calculations, and finally prepares for execution. The positional accuracy of printing is often accompanied by decimals in a printing process, however, the system cannot control the stepper motor to achieve such a precise position, so the system converts these precise parameters into integers before printing, which is convenient to control the pulse generation.

The G28 command plays a different role and it is the beginning of all print jobs. To execute this function, each coordinate axis moves quickly to the origin. When it touches the limit switch of each coordinate axis, it stops moving and moves in the opposite direction for a short distance, and then approaches the origin again at a relatively slow speed to ensure accuracy.

To execute G92 instruction the system will read the value behind the axis in each command. Meanwhile, set the value to the current position. Normally this code is used on zeroing dispenser position.
### 3.4.6 Activity management

The printer is not always in the printing state during the entire printing process. In the non-printing state, the motors of the respective coordinate axes are in operation, which is very wasteful. In other words, if the execution interval between the two commands is greater than the maximum inactivity time of the stepper motor, this time can be changed. The system will terminate all stepper motors, the codes are shown in Figure 8. Based on this consideration, activity management is very important throughout the system design process.

![Figure 8 The Code of Activity Management](image)

The endstop is a limit switch for specifying the minimum position, when the printing is executing, the system will detect the boundary of the printing area, the one of the axes touch the minimum position, the endstop will be triggered. At this moment the stepper motor will be disabled to protect the components of printer.

### 3.5 Planner

In 3D printer systems, the motion of the x and y axes tends to change very frequently: not only is the speed at each update position different, but the speed of each segment also needs to undergo acceleration, constant speed and deceleration. This is determined by the inertial characteristics of the mechanical system: if the speed between the different actions is not well connected, it will cause a strong current impact on the circuit system. Especially in the 3D printing process, this speed change has tens of thousands of print jobs per job, which means that the circuit life will be greatly reduced.

Based on the consideration of this factor, the stepping motor needs to have three processes of acceleration, uniform speed and deceleration during the printing process. The planner will calculate all the parameters and put them into a structure `block_t` before printing starts.

A pointer of type `block_t` can conveniently position any element in the structure. The `plan_buffer_line()` function is called whenever the 3D printer
parses the displacement instruction. The new `block_t` is first created and queued at the end of the queue; then `calculate_trapezoid_for_block()` is executed to calculate the critical speed node of the new `block_t` and its corresponding number of steps; then update the connection speed of all `block_t` in the queue. The ending speed and related speed node of the `block_t` at the end of the team will be updated. Finally, `st_wake_up()` is called to ensure that the interrupt executed by stepper is turned on.

The flow chart of the straight path planning is shown in the Figure 9 below.

![Figure 9 The Flow Chart of Planner](image)

There are a number of parameters in the `block_t` structure that need to be calculated and eventually called in the motion control subsystem to control the acceleration and deceleration of the stepper motor.

The main members in the structure are shown in below Table 4.

<table>
<thead>
<tr>
<th><strong>Table 4 Important Parameters in the Structure</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>The number of step events required to complete the block</td>
</tr>
<tr>
<td>The index of the step event on which to stop acceleration</td>
</tr>
<tr>
<td>The index of the step event on which to start deceleration</td>
</tr>
</tbody>
</table>
The acceleration rate used for acceleration calculation

The direction bit set for this block which is used to set the direction of step motor

The nominal speed for this block which unit is in mm/sec

Entry speed, the speed from the previous block to the block which unit is in mm/sec

Maximum entry speed, entry speed cannot exceed this value which unit is in mm/sec

The total travel of this block which unit is in mm

The acceleration which unit is in mm/sec^2

3.5.1 Planning a straight path

The `plan_buffer_line()` function mainly calculates the value of the relevant variable in the motion control block `block_t`. Since the motion of the stepper motor is actually measured in steps, and the unit of the input command is millimeter, the conversion from millimeter to step is necessary. This uses the amount of Step Per Millimeter (SPM) calculated before. The result of the calculation is the target position of each stepper motor, then the number of step events required to complete the block is the target position minus the current position.

Next, the direction of each stepper motor rotation should be judged by comparing the target position and the current position. If the target position is greater than the current position, the stepper motor will rotate in the forward direction. The direction bit will be assigned to “1”. If the target position is smaller than the current position, it means that the current direction needs to be moved in the opposite direction, and the stepper motor will move in the reverse direction. At this time, the direction bit will be assigned to "0". In the motion control section, the motor will receive the direction bit information to perform the corresponding motion.

At the end of the `plan_buffer_line()` function, `calculate_trapezoid_for_block()` is called. The function is to calculate the entry speed and exit speed of the block into steps per second. This is just the initial calculation.
3.5.2 Trapezoidal curve speed recalculation

In `plan_buffer_line()`, after calling `calculate_trapezoid_for_block()`, the function `planner_recalculate()` is called. This function contains three components. `Planner_reverse_pass()` checks and calculates the entry speed of the block from the back to the front. `Planner_forward_pass()` checks and calculates the entry speed of the block from the beginning to the end. The essence is that these two functions are responsible for the processing of different blocks, all the blocks can be connected by `planner_recalculate_trapezoids()`.

`Planner_recalculate_trapezoids()` is the final calculation of the entry and exit speeds, which connect the trapezoidal curves of all blocks. The most important part of this function is to calculate the displacement of the acceleration phase, the constant velocity phase and the deceleration phase. When the rated speed is not reached, the displacement in the uniform velocity phase is zero.

The reason for recalculating the speed of the trapezoidal curve is mainly because the curve velocities of each block are calculated separately in the initial calculation, which means that there is no connection between each block. This may lead to a lot of situations.

The `planner_reverse_pass()` function is used to solve the entry speed is not at the maximum entry speed, that means the current block is in state of deceleration condition. In this case, the entry speed of the current block is already greater than the entry speed of the next block. The entry speed of the next block was entered as the final of the current block, reset the entry speed of the current block. If this speed is greater than the maximum speed of the current block, the entry speed of this block is set to the maximum entry speed. As the name of this function, this function reads each block from the back to the front and performs the calculation.

Another function is `planner_forward_pass()`, this function is used to deal another problem. If the previous block is an acceleration block, but it is not long enough to complete the full speed change within the block, the entry speed need to be adjusted accordingly. In this situation the entry speed of current block would be calculated by the entry speed of previous block. This function is followed behind the `planner_reverse_pass()` function, in that function the entry speeds have already been reset, maximized, and reverse planned by the reverse planner. This function will read through the current plan from the front to the back.
3.6 **Motion control system**

As the name suggests, the main content of this part is the control of the movement. The control of the stepper motor is the core of the entire motion control.

In the main program, the initial timing of the Timer1 is set first, and after waiting for the interrupt, the motion in the block is interrupted. First, the system will take a block from the block buffer pool, analyze the motion parameters in the block, set the direction of motion, and then adjust the motion of the stepper motor. The time of the Timer1 is the moving speed of the stepping motor, so in terms of controlling the speed of the stepping motor, it can be realized by time calculation. Each interrupt executes one or several steps in a block. This parameter is set when the system is initialized.

The flow chart of the motion control program is as following Figure 10 shown,

![Flow Chart of Motion Control](image)

*Figure 10 Flow Chart of Motion Control*
3.6.1 **AVR Timer/Counter**

As mentioned before, for the motion control of stepper motors, the Timer1 which is one of the timers on ATmega2560 microchip was used, it is like a clock and can be used to measure time events. There are six timers in this microchip, first three timers called Timer0, Timer1 and Timer2. Timer0 and Timer2 are 8bit timer, where Timer1 is a 16bit timer [29]. The most important difference between 8bit and 16bit timer is the timer resolution. 8bits means 256 values, where 16bit means 65536 values which is much higher resolution. Timer3, Timer4 and Timer5 are all 16bit timers, similar to Timer1.

Each of the timers has a prescaler that generates the timer clock by dividing the system clock by a prescale factor such as 1, 8, 64, 256, or 1024. The Arduino has a system clock of 16MHz and the timer clock frequency will be the system clock frequency divided by the prescale factor.

The Timer/Counter Control Registers (TCCRnA and TCCRnB) hold the main control bits for the timer. Timer/Counter (TCNT) is the main working part, because Timer1 was used, the working mode of TCNT1 is determined by Timer/Counter Control Registers (TCCR1A and TCCR1B). These registers hold several groups of bits are shown in Figure 11 and Figure 12.

![Figure 11 The Bits of TCCR1A][29]

![Figure 12 The Bits of TCCR1B][29]

In this project, Clear Timer on Compare Match (CTC) Mode [40] by Timer1 was selected, Output Compare (OCR1A) Register are used to manipulate the counter resolution. In CTC mode the counter is cleared to zero when the counter value in TCNT1 matches the OCR1A. The OCR1A define the top value for the counter, hence also its resolution.
3.6.2 Interrupts

A timer can generate different types of interrupts. Interrupts are a way of interacting with the CPU and other CPUs that are asynchronous to the CPU. In this way, it is unnecessary to wait for the completion and trigger status of some other tasks in the CPU, and other hardware to trigger the timing of entering the main CPU process.

When using interrupts, it is important to enable the corresponding interrupt bit in the Timer Interrupt Mask Register (TIMSK1) which looks like Figure 13.

![Image of TIMSK1](Image)

An interrupt can be generated at each time the counter value reaches the TOP value by using the Compare Match Flag (OCF1A) according to the register used to define the TOP value. If the interrupt is enabled, the interrupt handler routine can be used for updating the TOP value.

3.6.3 Application in motion control systems

To start using timer, the most important settings are the last three bits in TCCR1B, which are CS12, CS11, and CS10. These dictate the timer clock setting. By setting these bits in various combinations, the timer will run at different speeds like Table 5 shows.

<table>
<thead>
<tr>
<th>CS12</th>
<th>CS11</th>
<th>CS10</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>No clock source (Timer/Counter stopped).</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>clkI/O/1 (No prescaling)</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>clkI/O/8 (From prescaler)</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>clkI/O/64 (From prescaler)</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>clkI/O/256 (From prescaler)</td>
</tr>
</tbody>
</table>
The divider was used as 8, resulting in a 2MHz timer frequency on a 16MHz MCU, thus the CS setting was set as 010.

The most important thing in the motion control program is to update the interrupt time, which generates a pulse to drive the motor motion as each interrupt is executed. In the CTC mode, an interrupt generation pulse occurs when the value of Timer1 reaches the value of OCR1A. As default, the value of OCR1A is set to 2000 which means the frequency of the interruption is 1MHz.

As mentioned earlier, the Planner will calculate all the required parameters, and the system will load these parameters and execute them each time the interrupt (ISR) is executed. In each interrupt, the control system set the direction of each stepper motor and generate a pulse. At the same time, the corresponding coordinate axis will reduce the number of steps by one, thereby realizing the control of the moving distance. When the endstops are touched, the system must be able to record the current position and stop the motor. Since the stepper motor controls the motor speed by changing the number of pulses, the more the number of pulses, the faster the speed and vice versa. Therefore, at the same time as each interrupt is executed, the system updates the new value in OCR1A, thereby changing the interval of interrupt operation, which is the basis of the overall system control.

3.6.4 Three-axis linear motion

When the printer is performing a print job, it is very basic and very important to move the extruder to the specified position by turning each axis motor. The motion of the stepper motor is actually planned by a command statement composed of G1. The system accepts the command to move the stepper motor to the specified coordinate point, and the series of points are joined together to form a linear motion of the entire print.
Construction of Control system for syringe
dispenser based on Printrbot 3D printer
Qi An

2019-10-03

Since there are two stepping motors on the axes to control the movement of the dispenser in one plane, how to ensure that the movement between the two points is a straight line is critical. Here, the Bresenham algorithm was used to ensure the successful implementation of straight movement.

Bresenham’s line algorithm is a line drawing algorithm that determines the points of an n-dimensional raster that should be selected in order to form a close approximation to a straight line between two points. Figure 14 shows the basic principles of the Bresenham’s line algorithm.

![Figure 14 The Principle of Bresenham’s Line Algorithm][30]

As this figure shown, consider drawing a line on a raster grid where the allowable slopes of the line to the range $0 \leq m \leq 1$. There is a line that passes through the two points $(x - 1, y)$ and $(x + 2, y + 1)$, but there are two points in the process that cannot determine the coordinates. Assuming that the error corresponds to the ordinate $y$ of each actual plotted point is $\epsilon$, then the true value of this point is $(x, y + \epsilon)$. This error is between -0.5 and 0.5. Moving from $x$ to $(x + 1)$, the ordinate is also increased by $m$, if $y + \epsilon + m < y + 0.5$, then choose to draw point $(x + 1, y)$. The new error will be $(y + \epsilon + m) - y$. Otherwise, the point will be $(x + 1, y + 1)$, which the error is $(y + \epsilon + m) - (y + 1)$.

Because of $m = \frac{\Delta y}{\Delta x}$, $\epsilon + m \rightarrow \epsilon + \frac{\Delta y}{\Delta x}$, if we assume $\mathcal{E} = \epsilon \cdot \Delta x$, then the formula will be $\mathcal{E} + \Delta y$. Thus when $\mathcal{E} + \Delta y < 0.5\Delta x$ is established, the point $(x + 1, y)$ will be drawn, which $\mathcal{E} \leftarrow \mathcal{E} + \Delta y$ because the ordinate has no added value. When $\mathcal{E} + \Delta y > 0.5\Delta x$ is established, the point $(x + 1, y + 1)$ will be drawn, which $\mathcal{E} \leftarrow \mathcal{E} + \Delta y - \Delta x$.

Based on this algorithm, in the printing control system, the system’s decision condition is changed and the flow chart is shown in Figure 15.
In motion control, the system first determines which is the long axis and uses the long axis as a reference for the movement and change the decision condition of the previous algorithm to,

$$\epsilon + \Delta y - 0.5 \text{step\_event\_count} < 0,$$

\text{step\_event\_count} is the number of step events required to complete one block that means the number of steps on the long axis.

Using pseudocode to represent the application of the algorithm in motion systems,

```
counter_x = -0.5 \text{step\_event\_count};
counter_x = counter_x + \text{step\_x};
if (counter_x) > 0
{
    move one step:
    counter_x = counter_x - \text{step\_event\_count};
}
```

Using this method, the system determines which axis motor is running during the printing process, ensuring that the extruder moves linearly between points.
4 Implementation

After describing the basic structure of the complete control system and the main methods used, this chapter will focus on how to assemble the hardware system, including electrical connections and some circuit modifications.

4.1 Control board

The original control board of Printrbot 3D printer is Printrboard Rev.F6, it is a very powerful control board. It has built-in stepper motor drivers, and endstop interface [31]. However, it was replaced by a combination of RAMPS and Arduino Mega 2560 like Figure 16 shows, mainly because the former is very difficult to upload code to. Since the aim of the project is to construct a control system, it is very important to be able to upload code conveniently. The Arduino features serial communications interfaces, it is very convenient to communicate via ports. The micro controllers are typically programmed using the programming languages C and C++. The Arduino Mega microcontroller makes uploading firmware much easier. RAMPS has many features as well that make it ideal as a control board for 3D printers. It can connect up to 5 stepper motors and has 6 connections for mechanical endstops or optical endstops.

![Figure 16 The Control Board with Arduino and RAMPS](image)

4.2 Dispenser

The dispenser system is the part that differs most of the extruded 3D printer compared to traditional 3D printers.

The original extruder of Printrbot was a conventional filament extruder, it typically uses a Polylactic Acid (PLA) filament which is a biodegradable
thermoplastic as a printing material, conveyed using a stepper motor, melted at the hot end and extruded through a nozzle.

Figure 17 The Extruder of Printrbot[32]

Paste-dispenser 3D printers use an alternative dispensing system instead of a conventional extruder. It consists of two components, one is the syringe, the other is the leadscrew driver system. The structure is shown in Figure 18. By contrast, the paste dispenser replaces the hot end of a conventional extruder with a syringe structure, mainly due to the filamentary 3D printer mainly prints filamentous plastic, while the paste 3D printer mainly prints paste-like materials.

Figure 18 The Dispensing System of Paste Dispenser 3D Printer

The syringe is a simple reciprocating pump consisting of a plunger that fits tightly within a cylindrical tube called a barrel. It is very commonly used in medical or laboratory applications. In an extruded 3D printer, the syringe is used as the carrier for paste materials.

The leadscrew driven system contains two gears, a leadscrew and a stepper motor. The basic working principle is that the stepper motor drives the leadscrew up and down through the gear. The end of the leadscrew
has a plane, it will contact the plunger in the syringe when the leadscrew moves down. The plunger will be linearly pushed along the inside of the tube, allowing the syringe to expel the material of printing through a nozzle which is installed at the front end of the tube. The nozzle I used in this project is metal and has a diameter of 1mm, the diameter of printing can be changed by using different sized nozzles.

4.3 Stepper motor

A stepper motor is one kind of electric motor used in the robotics industry, there are a variety of stepper type which are permanent magnet (PM), variable reluctance (VR) and a combination of PM and VR: hybrid (HY) stepper [33].

There are four stepper motors in my project, one for each axis, mainly for controlling the movement of each axis, and one for controlling the dispensing of the dispenser.

PM stepper motor is one of the common stepper motors. The rotor of the permanent magnet stepping motor is made of a permanent magnet material, and the number of poles of the rotor is the same as the number of poles of the stator. It is characterized by good dynamic performance and large output torque, but this motor has poor precision and large step angle.

![Figure 19 Two Phase Hybrid Stepping Motor][33]

The most popular stepper motor is hybrid stepping motor, it combines the advantages of reactive and permanent magnets. It has multi-phase windings on the stator, permanent magnet material on the rotor, and multiple small teeth on the rotor and stator to improve step accuracy. The stepper in Printrbot is a two-phase hybrid stepping motor which is shown in Figure 19, its basic step angle is 1.8° per step, and its step angle can be subdivided by 32 times with the subdivision motor driver.

The schematic diagram of the stepper motor usually looks like Figure 20,
Figure 20 The Schematic Diagram of The Stepper Motor[34]

The stepper motors are connected to the stepper motor driver, based on the structure of the stepper motor, there are four pins that corresponding to two sets of windings. The order of the wire is ABCD. In this way the stepper motors can rotate smoothly.

4.4 Microstepping driver

The stepping motor needs to be driven by a dedicated stepping motor driver, and the driver is composed of a pulse generation control unit, a power driving unit, and a protection unit. The power driving unit amplifies the pulse generated by the pulse generation control unit and directly couples with the stepping motor and belongs to the power interface of the stepping motor and the microcontroller. The driver in this case is DRV8825 which is shown in Figure 21 with the schematic of connection.

Figure 21 The Schematic of The DRV8825 Driver Connection [26]

Stepper motors move a known interval for each pulse of power. These pulses of power are provided by a stepper motor driver and are referred to as a step.

Each time an electrical pulse is input, the motor rotates an angle further. The angular displacement of the output is proportional to the number of pulses input, and the rotational speed is proportional to the pulse frequency. Changing the order in which the windings are energized, the motor will reverse. Therefore, the number of control pulses, the frequency,
and the energization sequence of the windings of each phase of the motor can be used to control the rotation of the stepping motor.

The pulse diagram is shown below Figure 22,

![Figure 22 The Basic Pulse Diagram](image22)

Although the stepper motor of Printrbot itself has a very small step size which is 1.8°, higher resolution is required to achieve precise control of the print position. Microstepping motor driver is used to improve resolution by allowing intermediate step locations, which are achieved by energizing the coils with intermediate current levels. There are different modes corresponding to different stepper motor subdivisions. In this project, the microstep resolution was set as 1/32 step, the resolution selector inputs will be connected with jumpers to place each pin to the high level, as shown in Figure 23.

![Figure 23 The Setting of Mircostep Resolution](image23)

### 4.5 Endstop

The length of travel for each linear axis is limited by endstops, in Printrbot 3D printer there are two different types of endstops, one is mechanical endstop, the other is proximity sensor, for different functions.

The mechanical endstop is the most basic type of endstop, it is a switch that positioned to trigger when the axis of the printing system reaches the
end of its motion. It can be used on X and Y-axis. It is very easy to implement because there are only two wires that should be connected which are Digital In and Ground. The schematic is shown in Figure 24.

![Schematic Diagram of Endstop Connection](image)

**Figure 24 The Schematic Diagram of Endstop Connection [35]**

When the switch is off like in the schematic above, it connects signal to ground. When the switch is triggered, the ground connection is cut and the signal is connected to 5v through the pull-up resistor.

Usually there is another wire that should be connected to power with a resistor, however the Arduino ATmega has internal pullup resistors that means the external resistor is not needed. The switch can simply be connected to the signal and ground pins.

The limit switch of Z-axis uses different solution. Unlike the mechanical limit switches of the X and Y axes, the Z axis uses an Inductive Proximity Sensor Detection Switch, the type is LJ12A3-4-Z/BY shows in Figure 25.

![Proximity Sensor](image)

**Figure 25 The Proximity Sensor**

This is mainly due to the special mechanical structure of the Z-axis. Since the syringe containing the printing material is mounted on the Z-axis, the
syringe has a certain length. The general mechanical switch is small in size and cannot be mounted on the front end of the syringe to protect the syringe. The proximity sensor itself is cylindrical and can be mounted on the Z-axis in parallel with the syringe for good protection.

![Figure 26 The Connection of Proximity Sensor][36]

The sensor is widely used for detecting, controlling and noncontact switching. When proximity switch is close to some target object, it will send out control signal. Three wires should be connected, as the Figure 26 shows, the brown and blue cables connect to the power supply. The black wire should be connected to the RAMPS board, the signal generated by the sensor is almost 12V however the RAMPS can only receive up to 5V. Therefore, a voltage dividing circuit is indispensable, two resistors are used with values of 10k and 15k as shown in the above schematic, to reduce the voltage of the signal.

### 4.6 Power supply

I selected a fixed power supply [39] for powering the entire system, the specifications of the power supply are 13.8VDC-6A. Although the recommended voltage for both Arduino Mega 2560 and RAMPS is 7 to 12 volts, if using more than 12V, the voltage regulator may overheat and damage the board. The power supply voltage is slightly larger than the recommended voltage due to the connection of other expansion boards and electrical equipment. The power supply is connected to the power pin portion of the RAMPS. The RAMPS has two power connections. One part is used to power devices that require high power, such as heating heads and hot beds. However, these parts are not available in my system. The system only need to power two boards and four stepper motors. The connection is shown in Figure 27.
4.7 Final assembly

Although the entire project is based on the transformation of the existing Printrbot 3D printer, the framework does not need to be changed, but it is necessary to make some electronic components connection.

The different electronic components on the 3D printer require many lengths of wire in order to supply both power and communication between components. If not properly managed, the amount of wire required could quickly become a mess and make it difficult to track which wire runs to which component.

The final assembly of the entire unit is shown below Figure 28,
5 Software interface

This chapter mainly describes how to construct an operation interface, including the selection of functions, the layout of the interface, and the implementation of the software.

5.1 Interactive function

As a hub for connecting printed parts, the control software should have many functions. The following functions are necessary. Based on these functions, the printer can be operated manually.

As an operation system, the most important function is to be able to control the movement of stepper motors in three axes. The stepper motors of each axis can be operated separately to control the individual movement of each axis. The operator interface should be able to control the forward and reverse rotation of the motor so that the printer can move on all six axes. It will be allowed to test each function of the printer before printing begins, ensuring that each axis is functioning properly. The stepper motor driver allows the stepper motor to have a small step size, for the X and Y axes, each step of 1mm needs to advance 160 steps, so the accuracy can reach 0.00625mm/step. The accuracy of the z-axis is higher, and its accuracy can reach 0.00025mm/step due to the screw structure.

The control of the print head motor is another key part. Since the syringe is used as the print head, the print material should be filled into the syringe before each print, then it is necessary to use the print head motor to adjust the dispenser to the discharge state. The printhead control motor also has high precision which up to 0.00125mm/step to ensure that the material is not squeezed out before printing begins.

When printing a model, the data of the model is converted into a single command statement by the slicing software and saved in the .gcode file. Therefore, loading 3D printed project files, reading and executing each G-command statement one by one is a function that the operating system must have. Not only that, the operating software should be able to receive the status returned by the printer and display it to eliminate possible faults, ensure that each component works properly, and each instruction executes smoothly.

Through the control software, commands can be sent to the firmware to control the operation of each motor. The command is sent in the form of
G-code. Each G-code instruction can be up to 96 bytes long. When the input is completed, press the Enter key and the printer will execute the corresponding command to the desired position.

5.2 Layout of the operation panel

The user interface is designed based on the needs of the function. In LabVIEW, the user interface is mainly arranged by placing the front panel controls.

The front panel is a graphical user interface. It is also the virtual instrument panel of VI. There are two types of objects: user input and display output. The switches, knobs, graphics and other controls and display objects were used to design the front panel.

The final design of the front panel is shown in the Figure 29.

![Figure 29 The Front Interface of Control System](image)

The front panel mainly consists of three areas, the first part is the button control area, the second part is the command input area, and finally the file import area.

The button control area mainly controls the rotation of the corresponding stepping motor through buttons, including three coordinate axes and an dispenser control motor. With the Knob control the operation system can
adjust the resolution to control the distance each axis moves. Accuracy from 0.1 to 100 mm allows the dispenser to be moved to various positions. There is also a button that is homing. Since auto-home is a very important function, a button is set up separately to facilitate the homing of each axis. The instruction input area is the window used to input the instruction. The instruction is sent in the form of G-code. The maximum length is 96 bytes, usually including the target position of each axis, or different function instructions. When the command input is completed, the printer will execute immediately. The last part is the file loading section. This part is used to select files and read them into the system one by one. Since the system will continuously feedback the running status, the content of the fault, the front panel should be able to receive and display the current information. A string indicator was put on the front panel to receive the information in real time.

5.3 Implementation of programming

5.3.1 Block diagram
The LabVIEW block diagram is a collection of graphical source code, also known as block diagram code. Once the front panel is created, graphical functions are used to add source code to control the objects on the front panel. Normally, VI contains block diagram objects such as terminals, functions, and wires.

The terminal is used to indicate the data type of the input control or display control. On the block diagram, the input control or display control of the front panel can be displayed as an icon or data type terminal. A terminal is an input/output port that exchanges information between a front panel and a block diagram. The data entered in the front panel input control enters the block diagram through the input control terminal, then the data enters the data budget, and finally outputs the new data value, and transmits the data to the display control terminal to update the data in the front panel display control.

The data transfer of the objects on the block diagram is done by wire. Each connection has only one data source, but it can be connected to multiple VIs and functions that read the data. Connections of different data types have different colors, thicknesses, and styles. The broken line is shown as a black dotted line with a red cross in the middle.
In addition, the block diagram must have a structure to repeat the code block, conditionally execute or execute the code in a specific order. The structure is a graphical representation of the loop and conditional statements in the text programming language.

5.3.2 Block diagram design
The final design of the entire block diagram is shown below Figures, Figure 30 depicts the block diagram of the file reading section, Figure 31 shows the block diagram of the button control part. The specific role of each module will be introduced below.

![Block Diagram]

Figure 30 The Final Block Diagram of File Reading
Figure 31 The Final Block Diagram of Button Control

(1) Use of the VISA function library

LabVIEW’s main method of interacting with the printer as an interface is to communicate through the serial port. Each operation on the front panel is converted into a corresponding command, which is sent to the printer system through the serial port. The system accepts the instructions, analyzes and executes them.

The serial communication between Arduino and LabVIEW is the basic and beginning of the entire process block diagram which is established by VISA plugin. VISA is essentially a general term for an I/O port software library and its specifications. Use VISA nodes for serial communication programming in LabVIEW. LabVIEW separates these VISA nodes into a single sub-module, which consists of eight nodes, which implement initialization of serial port, serial port write, serial port read, interrupt and serial port shutdown. In communication, the serial port is initialized first, and the port number, baud rate, stop bit, parity bit, and data bit of the serial port are set by using the VISA Configure Serial Port node. Then, the VISA Read node and the VISA Write node are used to read and write the serial port. After completing the communication, close the serial port, stop all read and write operations, and release the serial port resources.

The Figure 32 shows the setting of VISA Configure Serial Port node, the VISA resource name is the port name in the host computer which should
be selected from the front panel, the baud rate is set to 115200 which is the suitable rate of 3D printing.

![Figure 32 The Setting of VISA Configure Serial Port](image)

The VISA Write used to write the G-command to the serial port, the data was sent to the write buffer as shown in Figure 33 left. All instructions are sent to the processor in this way. On the right side of the Figure 33 is VISA Read, it can read specified number of bytes from the 3D printer and returns the data in buffer read. In the control system it will receive the statues of the printer.

![Figure 33 The Setting of VISA Write and VISA Read](image)

(2)While loop structure

Since the entire program needs to run all the time, this requires a loop structure to implement this function. Here, the While loop was used to repeatedly execute the program inside the loop. The block diagram of the While loop is a variable-size block, and the program written to the loop will be executed repeatedly until the Boolean value received by the condition terminal is FALSE.
Figure 34 The Main Loop of the Block Diagram

The Figure 34 shows the main loop body of the operation interface. When the program starts executing, the program continuously detects if a button is triggered.

(3) Event structure

The event structure is similar to the traditionally programmed interrupt structure, and it runs the relevant part of the code when it is triggered by a condition.

Using user interface events in LabVIEW keeps front panel user operations synchronized with the block diagram model, and events allow to execute a specific event processing branch each time a user performs a specific operation. If there are no events, the block diagram must poll the state of the front panel object in a loop to check for any changes. Polling the front panel object requires a considerable amount of CPU time, and if the change occurs too quickly, it is impossible to detect the change. By responding to specific user actions through events, it is not necessary to poll the front panel to determine what action the user performed, and LabVIEW proactively notifies the block diagram each time a specified interaction occurs.

As the Figure 34 shows, the event structure was used to respond to changes in the state of the button, sending the corresponding different instructions to the processor. The specific implementation method is, in
the edit of events, the button is set as the event source. When the mouse
down that means the button is clicked, the event happened, the edit panel
is shown in Figure 35. This is an example of x button which controls the
movement of X-axis.

![Image](image.png)

**Figure 35** The Edit Panel of the Event Structure

(4) Case structure

The case structure is used to perform different block diagram functions
under different conditions. Case structure contains two or more subrou-
tines, which one depends on the value of an integer, a non-unary string,
or an identifier that is connected to the external interface of the selected
segment or selection object. It must be set to a default case to handle out-
of-range values. Each subroutine in the case structure occupies its own
process block. The case structure is mainly used to change the distance
moved by each axis of the printer. The Knob has eight different positions
representing 0.1mm, 1mm, 10mm and 100mm forward movements, and
the other four represent moving in the opposite direction with same dis-
tance range. The Knob plays a conditional role to decide how far to move.
The structure is shown in Figure 34. When a data connection is used to
connect nodes inside and outside the case structure, the system automat-
ically generates a data tunnel on the case structure framework, and uses
this data tunnel to implement data interaction between the internal and
external nodes of the case structure. The data tunnel is divided into an
input data tunnel and an output tunnel data. The input data tunnel is a
solid rectangular box in any case, and each branch of the case structure
can obtain externally input data through the tunnel. In the output data
tunnel, the same type of data is connected to the output tunnel in each
branch. If there is a branch that does not assign a value to the output data
tunnel, then this output tunnel is a hollow rectangle.
(5) File I/O

When actually performing a 3D print job, the printed G-code file should be read and loaded into the printer system, which requires the File I/O function. When reading a file, it is generally necessary to follow the steps of opening the file, reading the file, and closing the file. The Open File function used to open the target file, in this function, the address of the file being loaded need to be set, since the address can be set manually, which means files from any location can be opened. When the file is loaded, the Read File function will read the contents of the file. Since the G-code file is also a text file, each instruction exists as a string in the file, so the function will be set to read the content one by one. And use the previously mentioned VISA Write function to communicate through the serial port, each instruction is sent to the printing system, waiting for execution. When the entire file is loaded, the Close File function closes the file and waits for the next action to occur. The entire block diagram is shown below Figure 36.

![Figure 36 The I/O File Diagram](image)

(6) Shift Register

Since the front panel button controls the rotation of the motor, LabVIEW will send G-code to the printer each time the button is pressed. The printer uses an absolute coordinate system, which means that if you want the X axis to move 10mm forward, you need to send G1X10 to the system. If you want it to move 10mm forward one more, send G1X20 to the system and these can only be done with one button. This requires us to write a program similar to the accumulator, adding or subtracting the corresponding coordinates each time it moves. The resolution is using a Shift Register function which is shown in above Figure 34. The shift register can implement this function by taking the result of the i-th cycle execution as the input of the i+1th cycle.

The shift register is executed as follows: At the end of each loop, the right terminal of the shift register saves the data passed to it and passes the data to the left terminal before the next loop starts, so that the left terminal gets the last loop. The ending output value, which can be used for the
next iteration. The shift register will be initialized before the start of the loop. The initialized value is read once before the loop starts and is not read after the loop starts.

5.3.3 Program debugging

After the block diagram is built, debugging is needed. The LabVIEW basic debugging method includes highlighted execution code, single step, breakpoint and probe to track the data flow through the VI.

If a VI program has a syntax error, the Run button on the panel toolbar will change to a broken arrow indicating that the program cannot be executed. This is the button called the error list. Clicking on it will bring up the error list window.

On the toolbar of the LabVIEW block diagram panel, there is a button that draws a light bulb. This button is called Highlight Execution which is shown in Figure 37. Click this button to make the block diagram execution process highlight. Then click the Run button, the VI program will run at a slower speed. The code that is not executed is displayed in gray. After execution, the code is highlighted and the data stream is displayed. The data values on the line so that the execution of the program can be tracked based on the flow state of the data.

Figure 37 The Highlight Execution Button

In order to find logic errors in the program, sometimes the flowchart program is expected to execute on one node and one node. Use the breakpoint tool to abort program execution at a point in the program, and view the data in a probe or single step. When using the breakpoint tool, click to set or clear the breakpoint. The power-off display is indicated by a red box for the node or frame and a red dot for the connection. When the VI program runs to the set breakpoint, the program is paused at the node to be executed and flashes. Pressing the single step button, the flashing node enters execution and the next node to be executed becomes flashing indicating that it will be executed.

When the data stream on the block diagram flows through a certain connection line, the probe tool will be used to view the data values at this time. Select the probe tool from the tool plate and click on the link where
the probe needs to be placed. A probe display window will appear, which is always displayed on the front panel window or the flowchart window.
6 Calibration and analysis

Calibration is an important and fundamental step before a complete 3D printing to ensure that the printer works accurately. It is necessary to check the movement of each axis, test whether the endstops are working properly, and whether the dispenser discharges smoothly. Not only that, since this is a paste-dispenser 3D printer, a series of parameters from the G-code file need to be adjusted that is generated by the slicing software.

6.1 Feed rate adjustment

As mentioned before, due to the physical characteristics of the plastic syringe and the materials used for printing in an extruded 3D printer, the extrusion speed of the 3D printer cannot be as fast as the traditional 3D printer, so adjusting the print speed is the key to determining whether the printer can print the ideal graphics.

There are many different speeds in a print project file, such as the idle movement speed of each axis, the homing speed, etc. The most important one is the speed associated with the dispenser, called the feed rate. A suitable feed rate means the integrity of the print. Too fast a feed rate can cause the print to be intermittent and unable to be lined up. On the contrary, if the feed rate is too slow, the printed material will repeatedly pile up in the same position, so that it cannot be printed accurately.

An experiment was designed to find the most appropriate feed rate. The experiment was divided into three parts, printing straight line, square and five-pointed star. The goal is to test the effect of the same shape on different feed rate. The complexity of the graphics increases in order to find the feed rate that is appropriate for all printed graphics.

6.1.1 Printing Material

Throughout the experiment, the wood filler, a white paste-like material, was chosen to print on different shapes. For paste printing materials, one characteristic is very critical, that is, viscosity [42]. Viscosity is a measure of a fluid's resistance to flow [43]. It describes the internal friction of a moving fluid. A fluid with large viscosity resists motion, a fluid with low viscosity flows easily. The table shows the viscosity of several common substances at room temperature [42].
Table 6 Common Substance Viscosity[42]

<table>
<thead>
<tr>
<th>Substance</th>
<th>Viscosity (mPa·s)</th>
<th>Temperature (°C)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Water</td>
<td>1.0016</td>
<td>20</td>
</tr>
<tr>
<td>Mercury</td>
<td>1.526</td>
<td>25</td>
</tr>
<tr>
<td>Olive oil</td>
<td>56.2</td>
<td>26</td>
</tr>
<tr>
<td>Honey</td>
<td>2000-10000</td>
<td>20</td>
</tr>
<tr>
<td>Ketchup</td>
<td>5000-20000</td>
<td>25</td>
</tr>
</tbody>
</table>

The viscosity of wood filler is similar to that of honey. The speed calibration of the entire experiment is based on this viscous paste material. Therefore, for paste materials of other viscosities, the feed rate setting should be adjusted as appropriate.

6.1.2 A straight line testing

The effect at different feed rate in simplest straight-line printing is shown below,

![Figure 38 A Straight-line Test in Different Feed Rate](image)

As shown in Figure 38, in the straight-line test, the printer prints a straight line of the same length from the right to the left of the picture. At different feed rates, the effect of printing is also different. For a clearer view, the
printed line is indicated by a solid green line, and the unprinted part is indicated by a green dotted line. The unit of feed rate is mm/min.

When the feed rate is 3000, only a small part is printed, and many materials are stacked at the end. A similar effect is also shown when the feed rate is 2000. Continue to reduce the feed rate. When the feed rate is 1000, the printed solid line portion increases significantly, but the solid line portion breaks. This is because the speed at that time the print head moves is still greater than the extrusion speed of the material. At a feed rate of 500, the printed solid line length continues to increase, but still cannot print a complete continuous solid line. When the feed rate is equal to 300, things changed, and the printer can print a continuous solid line. When the feed rate is 100, successive solid lines are printed, and the length is closest to the preset length.

Through experiments, it is obviously to find that in the extrusion printing process, the feed rate should be less than 1000 in order to print a continuous straight line.

6.1.3 A cube testing

The effect at different feed rate in cube testing is shown below,

![Figure 39 A Square Test in Different Feed Rate](image)

In a square print test, the feed rate was selected below 1000 to print relatively complex and multi-line graphics.
The Figure 40 shows parts of G-code file which indicate the profile of the square.

![G-code file](image)

**Figure 40 Part of the G-code file of a Square**

With these G-code commands, the printer will move 18mm along X-axis and 8mm along Y-axis, the diameter of nozzle to print wood filler is 1mm, it can be calculated that this is a 20*10mm square. This provides a standard for the final print results.

As shown in Figure 39, the feed rate ranges from 100 to 400, the graphics also show different effects. In the previous straight-line printing test, the print rate was the best when the feed rate was equal to 100, but in the square test, due to the many crossovers of the graph, when the feed rate was 100, a large amount of material was accumulated in the print result. Many lines are blended together, and it is impossible to distinguish the boundaries. The size of the square has a large error. After printing, the length of the square becomes 22mm and the width is 12mm. The error rate is 10% and 20% respectively.

Increasing the feed rate, at 300mm/min, the square length and width are 21.5mm and 11mm respectively, the error is reduced to 7.5% and 10%, the error is significantly reduced, and the size approaches the standard size. However, the square frame appears slightly deformed and somewhat curved. Further increase the feed rate. When it reaches 400mm/min, the square length and width are 20.5mm and 10.2mm respectively, the error is reduced to 2.5% and 2%, and the frame is straight and has no deformation.

The changes in the different feed rates can be clearly seen in the table below.
Table 7 The Effect of Different Feed Rate

<table>
<thead>
<tr>
<th>Feed rate (mm/min)</th>
<th>Size (mm*mm)</th>
<th>Error (% length:width)</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>20*10</td>
<td>0:0</td>
</tr>
<tr>
<td>100</td>
<td>22*12</td>
<td>10:20</td>
</tr>
<tr>
<td>300</td>
<td>21.5*11</td>
<td>7.5:10</td>
</tr>
<tr>
<td>400</td>
<td>20.5*10.2</td>
<td>2.5:2</td>
</tr>
</tbody>
</table>

The experimental results clearly show that when the feed rate is at 400, the printed graphics lines are the clearest and closest to the target size.

6.1.4 A five-pointed star testing

The effect at different feed rate in five-pointed star is shown below,

![Figure 41 A Five-pointed Star Test in Different Feed Rate](image)

The five-pointed star print test is designed to understand the effect of different print layers on extruded print items. Figure 41 shows that as the number of layers printed increases, the lines of the graphic become blurred.
Figure 42 The Side View of Two Layers Star

Figure 42 shows a side view of stars with 2 layers, each line of the five-pointed star is very clear, the corner of the five-pointed star is very sharp, the second layer falls well above the first layer.

Figure 43 The Side View of Three Layers Star

Increasing the number of printed layers to 3 layers, the outline between the layers becomes unclear, and the wood filler is piled up at the sharp corners of the stars. But the overall shape of the five-pointed star is still very obvious.

The view of 4 layers start is shown in Figure 44.

Figure 44 The Side View of Four Layers Star

The lines of many layers are merged together, and the boundaries become unclear. The material of top layer slide down to the bottom, moreover, since the thickness of each position is different, even the paste material of the same layer cannot be in the same plane.

The five-pointed limit printed with wood filler as the printed material is five layers. When the layers increase to 5, Figure 45 is shown below, the size of the five-pointed star is significantly reduced because many layers
melt together and squeeze the original hollow area inside the star. The shape of the object will become increasingly blurred if the number of layers was increased continually.

![Image](image.png)

**Figure 45 The Side View of Five Layers Star**

Different feed rates have an effect on multi-layer printing. The five-pointed star with a feed rate of 400 mm/min is significantly clearer at the corner than the five-pointed star with a feed rate of 300 mm/min.

### 6.2 Communication rate

The G-code file transfers each instruction to the system through serial communication. The serial communication is realized by the VISA control of LabVIEW. Then, if the speed of reading the instruction is too fast, when the printer does not complete the current instruction, the new instruction has been read in, which will cause communication jam. To solve this problem, the speeds of reading each instruction were slowed down by adding a delay to the block diagram of LabVIEW.

The test was carried out on the premise that the feed rate was 400 mm/min. When the delay is 50ms, the communication is blocked, the printer does not run, and printing cannot be completed. The control panel is shown below.

When the delay is increased to 500ms, printing can be performed, but the instructions are not read one by one. Since the previous instructions are not completed, the new instructions cannot be stored in the cache, causing the printing process to skip.

<table>
<thead>
<tr>
<th>Delay time (ms)</th>
<th>Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>The communication is blocked</td>
</tr>
<tr>
<td>500</td>
<td>Skip some G-code instruction</td>
</tr>
</tbody>
</table>
Based on the above experiment, the delay is finally set at 800ms. Under the current delay, each instruction can be executed smoothly without missed reading or long pause.

### 6.3 Application on printed electronics technology

Printed electronics [44] is one of the fastest growing technologies in the world, with a wide range of applications including photovoltaics [45], displays [46], sensors [47] and biomedical devices [48]. Silver conductive ink [10] is an important material in printed electronics technology and has been widely studied due to its relatively low cost and simple instrument [10].

However, for most board work, silver inks have not been widely used for conduction due to their low electrical conductivity and relatively high electrical resistance. When silver ink is used to print circuit tracks, the idea is to increase the track cross-sectional area to reduce the resistance of the track, thereby increasing the conductivity.

Silver ink is a paste material, which means it can be used in a paste dispenser 3D printer to print on the track to increase its conductivity. Based on this assumption, a line of silver ink was printed on a printed silver ink circuit trace to decrease the resistance. The original circuit trace as shown in Figure 46.

**Figure 46 The Circuit Trace with Silver Ink**

### 6.3.1 Limit parameter

Before the formal test, the parameters must be determined first, because the rail resistance should be minimized, the straight line should be printed as thick as possible. However, the paste-like silver ink needs to be dried in the oven, which means that if it is too thick, it will not be
evenly dried or even broken, resulting in a short circuit. Since the viscosity of silver ink is similar to that of wood filler, the initial feed rate was selected as 400mm/min.

The result of test is shown in Figure 47.

![Figure 47 The result of Parameter Test](image)

As shown in Figure 47 the performance of straight line with 400mm/min is very well. In order to make the line thicker, the feed rate was reduced to 200mm/min. The line can still be electrically conductive, but as can be seen from the color, the line appears to be significantly uneven. It is inability to quantify the results. When the feed rate was slowed down to 100mm/min, the lines are clearly bent and cannot conduct electricity. Therefore, setting the feed rate to 400mm/min, the nozzle diameter to 2mm, is the best choice.

### 6.3.2 Printing on circuit track

The printed result on the circuit track is as shown in the Figure 48.
Through real printing, it is obvious to see that the resistance of the three sets of circuit tracks are reduced to varying degrees. After the first set, the resistance decreased from the initial $3.67\,\Omega$ to $3.05\,\Omega$, a decrease of $16.7\%$. In the second group, the resistance dropped from $1.19\,\Omega$ to $0.99\,\Omega$, a decrease of $16.7\%$. In the third group, the resistance was reduced from $2.21\,\Omega$ to $1.72\,\Omega$, a decrease of $22\%$. The effects are shown in Table.

<table>
<thead>
<tr>
<th>Width of track (mm)</th>
<th>Resistance before printing (Ω)</th>
<th>Resistance after printing (Ω)</th>
<th>Decrease percentage (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.25</td>
<td>3.67</td>
<td>3.05</td>
<td>16.7</td>
</tr>
<tr>
<td>2.5</td>
<td>2.21</td>
<td>1.72</td>
<td>22</td>
</tr>
<tr>
<td>5</td>
<td>1.19</td>
<td>0.99</td>
<td>16.7</td>
</tr>
</tbody>
</table>

### 6.4 Result and analysis

Through multiple sets of experiments, the feed rate can usually not be higher than $1000\,\text{mm/min}$ during the dispenser 3D printing.

When printing some simple single-layer graphics, the slower the feed rate, the better the integrity of the graphics, which can be reached to $100\,\text{mm/min}$. $400\,\text{mm/min}$ is a good choice when printing relatively complex multi-layer graphics. According to the complexity of the printed...
items and different paste-like materials, the feed rate has also changed accordingly.

The communication rate needs to match the feed rate. The communication rate should also be increased under high-speed motion. Conversely, if the feed rate is slow, the communication rate should also be slowed down. When the feed rate is 400, the usual communication rate is 800ms.

In the experiment of printing silver ink on circuit tracks, the dispenser 3D printer plays an important role, the electrical performance of the circuit track is improved by printing. With this 3D printing technology, the printing area and the amount of printed materials can be effectively controlled for better research.
7 Conclusions and future work

7.1 Control system and interface
The thesis focused on building the control system of a dispenser 3D printer, assemble the electronic and mechanical components, and designed the operator interface. Through the final print test, the original idea was realized, the control system can analyze, process the accepted G-code instructions, and finally execute.

The system is mainly divided into two main parts, the main program part and the motion control part. The main program relies on a large Switch structure. The instructions are transmitted through the serial communication and stored in the cache. The control system analyzes the corresponding parameters by identifying the key letters and numbers in the command, calculates the parameters through the route planning part. The motion control section performs the corresponding action. The motion control part is realized by a timer interrupt. After each interruption occurs, each motor moves one step according to the algorithm, the running speed of the motor is realized by changing the interrupt trigger interval time.

The operation interface realizes the manipulation of the printer through serial communication, it can load G-code files and read the instructions one by one to communicate to the printer system and can conveniently control the movement of each axis of the printer and the extrusion of materials. By adjusting the parameters, the appropriate feed rate and communication rate were found to print a relatively complete and consistent pattern.

7.2 Social and environmental aspects
Paste-dispenser 3D printing technology is a very socially friendly technology that allows us to break through the limitations of materials. This technology is suitable for everyone’s use because of its ease of operation, and there is no environmental pollution. This technology will provide new possibilities for research in various fields in the future.

7.3 Future work
Although the original idea was basically completed, there are still many places where need to be improved to achieve better results. In terms of system design, the system can only execute a small number of necessary
G-code commands, cannot perform curve motion. Only the absolute coordinate position can be used for each axis movement. Only simple printing is possible. The design of the operation interface is relatively simple, the file reading part and the button operation part cannot be placed in the same LabVIEW file. Continuously improving the design of the entire system, the paste dispenser 3D printer will have better performance.
References

https://en.wikipedia.org/wiki/3D_printing
Retrieved 9 September 2019


[3] 3DINSIDER,” How Much Does a 3D Printer Cost?”
https://3dinsider.com/cost-of-3d-printer/

https://en.wikipedia.org/wiki/3D_printing_filament
Retrieved 12 September 2019

https://www.3diligent.com/3d-printing-service/powder-bed-fusion/

https://en.wikipedia.org/wiki/Powder_bed_and_inkjet_head_3D_printing

https://i.materialise.com/blog/en/resin-3d-printing-stereolithography-explained/
Retrieved 2 June 2017


https://en.wikipedia.org/wiki/Graphene
Retrieved 13 September 2019

Construction of Control system for syringe dispenser based on Printrbot 3D printer

Qi An

2019-10-03


[17] Cellink,”Printing Life”
https://cellink.com/global/

[18] Cellink,” Spheroid formation of lung cancer cells”
https://cellink.com/global/technology/cell-tissue-models/spheroid-formation/

[19] Cellink,” Modeling the nervous system”

[20] Cellink,” HEP X-series for liver tissue models”
https://cellink.com/global/technology/cell-tissue-models/liver/

[21] Cellink,” PAN X for pancreatic β cells”
https://cellink.com/global/technology/cell-tissue-models/pancreas/


Rev. 8025I–AVR–02/09


[34] CSDN,”Linear speed control of stepper motor”, https://blog.csdn.net/u010650845/article/details/76855476
[35] Arduino,”Proximity sensor detection”,
https://forum.arduino.cc/index.php?topic=497937.0
Retrieved 01 September 2017.

[36] Core 3D,”Wiring a Mechanical End Stop”,
https://core3d.tech/2017/09/13/proximity-sensor-detection-lj12a3-4-z-bx-vs-lj12a3-4-zby-wiring/

http://www.3dprinterprices.net/how-to-build-a-3d-printer-getting-started-guide-tips/

[38] Ke Sun, Teng-Sing Wei, Bok Yeop Ahn, Jung Yoon Seo, Shen J. Dillon,* and Jennifer A. Lewis*, 3D Printing of Interdigitated Li-Ion Microbattery Architectures,2013.

https://reprap.org/wiki/Power_Supply

[40] Projects from Tech,” Arduino Interrupt Stepper Driver - CTC Mode”

[41] CSDN,” LabVIEW study notes (11) - Serial communication using VISA”

https://en.wikipedia.org/wiki/Viscosity
Retrieved 19 September 2019

[43] Physics.info,” Viscosity”
https://physics.info/viscosity/

https://en.wikipedia.org/wiki/Printed_electronics


Appendix: Block Diagram of LabVIEW

Operation Panel
Construction of Control system for syringe dispenser based on Printrbot 3D printer
Qi An 2019-10-03

File Reader

Instruction Input and Feedback Review