Effect of infill parameters on spatiotemporal thermal distribution in fused deposition modeling

Pranav Addepalli

Academy of Science

31 May 2019

EFFECT OF INFILL PARAMETERS ON SPATIOTEMPORAL THERMAL DISTRIBUTION IN FUSED DEPOSITION MODELING

Abstract

Three-dimensional (3D) printing has grown into a widely used technology for consumer and industrial use. Most commercial 3D printers use fused deposition modeling (FDM), a printing technique where a solid thermoplastic filament is repeatedly melted and extruded onto a two-dimensional layer to produce a 3D object. In FDM printing, thermal stresses between layers due to variable thermal conduction during cycles of heating and cooling create distortions, known as warpage. Various parameters, especially infill percentage, cause thermal properties to become anisotropic because of thermal conduction through plastic, natural convection in air gaps, and the discontinuous nature of plastic. The effect of infill percentage on spatiotemporal temperature distribution was investigated, and a strong, positive association was hypothesized between infill percentage and thermal conductivity due to plastic's more effective means of heat transfer of plastic when compared to air. Polylactic Acid discs of 10%, 20%, and 30% infills were printed and negative temperature coefficient thermistors were embedded to collect spatiotemporal temperature distribution data. The center of temperature and mean temperature at the center was calculated for all times and the temperature gradient was calculated between an equilibrium steady-state point and the centers. The mean gradient for 30% was greater than the mean gradient for 20% ($p < 0.0001$) and the mean gradient for 20% was greater than the mean gradient for 10% ($p < 0.0001$), showing a positive relationship between infill percentage and net heat flow. Graphs of temperature centers showed that greater infill percentages led to increased uniformity in temperature distribution, with the center of temperature being closer to the equilibrium point.

EFFECT OF INFILL PARAMETERS ON SPATIOTEMPORAL THERMAL DISTRIBUTION IN FUSED DEPOSITION MODELING

Introduction

Three-dimensional printing, a form of additive manufacturing that uses a digital design to create structures and other materials, has grown into a widely used technology in many major industries. It has also expanded to the commercial side, with a variety of products for consumer use. It is becoming an attractive research project for chemical, material, and biomedical projects due to its advantages of easy operation, low cost, and high manufacturing speeds (Zhuang et al., 2017). Moreover, 3D printing is being used for rapid prototyping and manufacturing in situations where complex structures are required, including construction, dentistry, medicine, electronics, automotive, robots, military, oceanography, aerospace, and defense industries (Zhuang et al., 2017).

In 3D printing, parts are manufactured by layer. Computer software is used in 3D printing to slice a 3D digital model in one direction, usually the Z-axis, and create a set of commands for the printer to create a layer. Different techniques are used to convert a digital design into a solid part, most notably stereolithography (SLA), selective laser sintering (SLS), with the most common being fused deposition modeling (FDM) (Dizon, Espera, Chen, & Advincula, 2018). SLA printing implements photopolymerization, a process in which light links chains of molecules to create polymers, and uses the polymers formed to create the solid body of an object. SLS printing is a relatively new technology that uses a laser to sinter a powdered material and create a solid structure. SLS printing methods are used in direct metal laser sintering (DMSL) to melt and fuse powdered metal together with a laser. DMSL, SLS, and SLA, although having substantial capability, are expensive and are primarily only used in specific industrial applications. FDM printing, however, is employed in most commercial 3D printers. In FDM printing, a solid thermoplastic filament is melted and then extruded onto a bed to create a two-dimensional layer. The extruded semi-liquid polymer solidifies virtually immediately after leaving the extruder (less than one second after), and another layer is printed on top of it repeatedly to create a three-dimensional solid object from the digital design. The unique quality of FDM printing to melt and cool thermoplastics layer by layer, along with its high speeds and low costs, has led it to be the most commonly used technique for 3D printing (Dizon et al., 2018).

Polylactic acid (PLA) is the most widely used commercial thermoplastic filament for 3D printing with FDM because it is derived from renewable lactic acid as well as its biodegradable and bioactive nature (Trhlíková, Zmeskal, Psencik, & Florian, 2016). PLA's melting temperature, around 180° C to 220° C, is lower than those of other filament materials such as acrylonitrile butadiene styrene (ABS), which is around 105° C; therefore, high-temperature areas pose a problem for objects created with it (Zhuang et al., 2017). Compared to ABS, PLA has a better print quality and lower costs (Trhlíková et al., 2016).

However, FDM printing involves processing filament through thermal cycles which can cause distortions in the objects. During the printing process, after a layer of plastic is deposited, the cooling of the layer causes the plastic to contract and create stress along the object's lateral

surfaces, and an increased rate of cooling increases the stress (Ultimaker). This stress is greatest at corners of objects, causing the corners to be pulled both upwards and inwards. Any detachment of the object from the printer bed can cause issues with printing successive layers. The repeated heating and cooling cycles during the printing process repeats this issue for almost every layer, resulting in varying print qualities and levels of warpage. Armillotta, Bellotti, and Cavallaro in 2018 suggested a physical explanation of distortion after analyzing the warpage problem: the extension of thermal stresses to multiple layers due to heat conduction from the last layer.

The thermal conduction of a material is defined by Mathur as the flow of heat through an unequally heated body from places of higher to places of lower temperature (Mathur, 1970). Thus, conduction is the transfer of heat through many molecules until there exists a dynamic thermal equilibrium (or steady-state), where there is no net movement of heat through the object. The thermal conductivity of a material essentially gives a measure of the material's ability to transfer heat via conduction. It also expresses the anisotropy of an object and is evaluated using Fourier's Law for Heat Conduction, which states that heat flux density, or rate of heat transfer, is directly proportional to the negative temperature gradient and the thermal conductivity as shown in Equation 1,

$$\vec{q} = -k\nabla T \tag{1}$$

where $\vec{q}$ is the heat flux density or rate of heat transfer, k is the thermal conductivity, and ∇T is the temperature gradient.

Unlike conduction, convection is a major process in objects that are either hollow or porous, with air pockets scattered around the internal structure. Natural convection uses buoyancy forces and the different densities of warm and cold air to force the warmer air upwards, creating a cycle in which the warm air and cold air circulate through a space. Natural convection is evaluated most often with Newton's Law of Cooling, which, in convection, states the rate of heat transfer is directly proportional to the area of the object, the heat transfer coefficient of a material, and the difference between the object's surface temperature and the air temperature as shown in Equation 2,

$$\vec{q} = hA(T(t) - T_A) \tag{2}$$

where $\vec{q}$ is the heat flux density or rate of heat transfer, *h* is the heat transfer coefficient, *A* is the surface area of the object, *T(t)* is the temperature of the object at time *t*, and T_A is the ambient or air temperature.

When an object is heated or cooled using one of the methods of heat transfer, points of higher temperatures and lower temperatures are created, causing a temperature gradient where heat flows from the hotter areas to the cooler areas. This gradient is the spatial rate of change of heat flow, which is influenced by the thermal conductivity of the material, and the direction of heat flow, which is influenced by the position of the heat source with respect to the

rest of the object (Trhlíková et al., 2016). When there is a temperature gradient in a material, the temperature distribution of the object changes as some areas are heated more than others. Each type of heat transfer process, including convection, conduction, and radiation, influences the temperature distribution differently (Kim & Viskanta, 1984). Measuring the temperature of an object at different points and at various time intervals can give an accurate indication of the temperature distribution of an object.

Most modern 3D printers do not print a fully dense structure to reduce time and cost for each print; instead, objects are made with different internal structures called infills (Han, 2016). The infill percentage of a structure is the ratio of air, or empty space, to plastic within the structure, with 0% being completely hollow and 100% being fully dense. Varying infill percentage has multiple advantages. For example, prints with higher densities, or infill percentages, have higher tensile strength, are less easily compressed, and are more resistant to bending (Baich, Manogharan, & Marie, 2015). Lowering the infill percentage decreases the time taken for the object to be printed.

Various infill percentages in 3D printing affect the thermal properties of the object, causing them to become anisotropic, because of the use of a discontinuous medium, plastic. Zhuang et al. in 2017 supported this by using FDM printing of conductive PLA and ABS and adjusting the layer deposition to create materials with anisotropic heat distribution.

Within a printed object, conduction and convection occur and allow heat to be transferred throughout the material. Research shows that conduction takes priority over convection in plastics and enclosed spaces. The plastic within the structure has a higher thermal conductivity than air, allowing it to transfer more heat than air. Kim and Viskanta in 1984 show that increased wall heat conduction reduces the average temperature differences in a cavity, stabilizes the heat flow, and, most importantly, reduces the rate of heat transfer by natural convection. The most significant impact of their research is that they, along with Wang, Yang, Zhang, and Pan in 2015, who studied surface radiation on heat transfer on heat transfer in a horizontally porous layer, find that conduction is the superior heat transfer process in an enclosure such as the internal structure of a 3D printed object (2009). Generalizing their results supports the idea that plastic has a much greater impact on the thermal conductivity and overall temperature distribution of a 3D printed object than air. However, Han in 2016 simulated the thermal conductivity of PLA and found that increased densities led to a decrease in thermal conductivity. He noted that the only major discrepancy between the results, although being minimal, could be due to the natural convection caused by the air gaps.

Studies have been done on the heat transfer and thermal conductivity of porous structures for various materials, but none have studied extensively plastic, PLA, or the specific internal pattern on the heat transfer. Deng et al. in 2018 investigated the effect of 3D printed hollow structures in sand mold manufacturing and found that more hollow structures could be used as heat insulators due to the increased number of air cavities and less solid material. Their research shows how increased porosity in sand molds leads to a decrease in thermal

conductivity. Larkin and Churchill in 1959 studied the heat transfer through radiation in porous insulations theoretically and experimentally. They found that increasing bulk density of fiberglass and foam glass decreases the amount of radiant heat transfer. They also found that the bulk density increased the amount of heat transfer through conduction but were unable to produce explicit values for the trend.

Although there is much research on the thermal conductivity and heat transfer properties of porous media of different materials, there is limited research on the heat transfer of plastic itself. Zhuang et al. in 2017 were able to create objects with anisotropic heat distribution through 3D printing. Additionally, efforts have been made to 3D print heat exchangers; for example, Haertel and Nellis in 2017 used various designs through density-based topology optimization to create a fully developed 3D printed heat exchanger. They found that more thin walls and higher unit cell heights within the heat exchanger increased thermal conductivity.

Current research in heat transfer does not focus on the thermodynamics and heat transfer properties of the objects; rather, research focuses on application of 3D printing to various industries (Deng et. al., 2018). Additionally, research on the heat transfer of porous materials does not focus on the heat transfer properties of plastic. There is also limited research on the heat transfer effects in additive manufacturing, and research has only begun to start recently. For example, Zhang et. al in 2017 numerically analyzed the influence of conditions while 3D printing on heat transfer. Their research offers only an extremely specific mathematical model during and after printing; however, the conditions set by the researchers are impractical for use of the model in any other situation. For example, they assumed the objects printed were pore-free, essentially a 100% infill, which is impossible to recreate because of the slight errors in the extrusion of plastic from a 3D printer. They also neglected heat radiation within the object, polymer crystallization and energy balance, and thermal expansion (Zhang et. al, 2017).

Most studies support the idea that increased infill percentage and therefore density will cause an increase in thermal conductivity, but there is no current research on the actual spatiotemporal thermal distribution, and factors that affect it. This study aims to investigate the relationship between infill percentage and the spatiotemporal thermal distribution during the printing and cooling processes in fused deposition modeling.

It was hypothesized that greater infill percentages will have more uniform temperature distributions of temperature because conduction has been shown to be more effective than natural convection or radiation for heat transfer in enclosed structures. PLA discs of 10%, 20%, and 30% infill percentage and rectilinear infill pattern were printed, and temperature over time at eight positions was measured by embedding thermistors into each of the print. Temperature centers for every time and infill percentage were calculated, and gradients from the center to an equilibrium point were calculated and compared across infills. The 3D printer, the bed

temperature, slicer software, filament, and thermistor channel design were kept constant throughout the experimental trials.

Materials and Methods

A template disc was designed using CAD software, specifically, Fusion360, with a diameter of 8 inches and a height of 0.5 inches. Construction circles were created on the template disc with smaller diameters, incrementing by 1 inch, to create circles with diameters 7, 6, 5, 4, 3, 2, and 1 inches. Random positions on these construction circles were chosen to embed thermistors, and rectangular channels were designed so the thermistors and their wiring could be embedded. The channels connected each of the randomly chosen positions to a single channel along a horizontal diameter, which led to open ends on both sides of the print. These channels began at a height of 0.15 inches and ended at a height of 0.25 inches so that the entire model was composed of 0.15 inches of regular infill, 0.2 inches of channels and infill, and another 0.15 inches of regular infill from the bottom of the model to the top. Figures 1 and 2 show multiple views of the model.

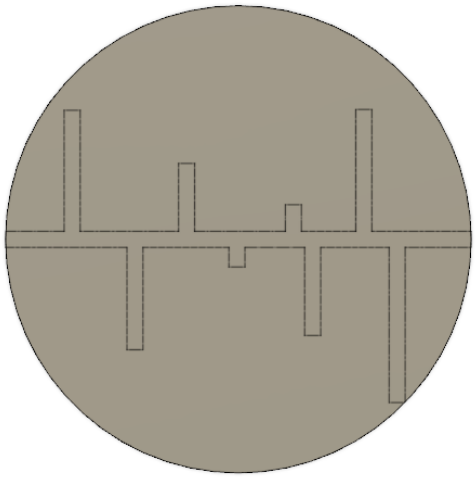

Figure 1.
Top view of disc template CAD model.

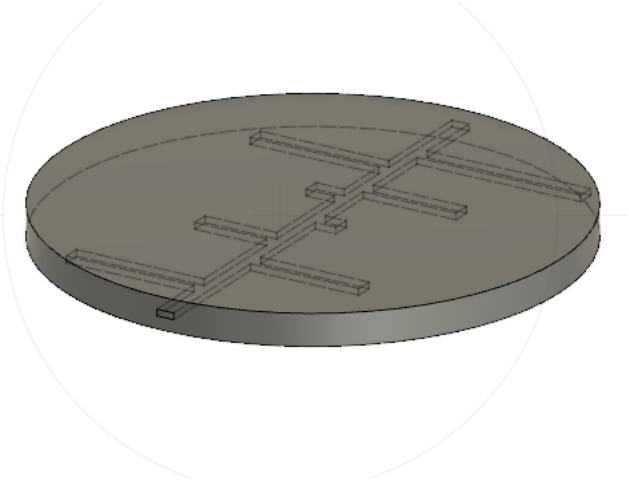

Figure 2.
Side view of disc template CAD model.

The model was opened and sliced three times using Ultimaker Cura to transform it to G-code for the printer. All print settings were kept as the default values as shown in Table 1, except for infill percentage.

Table 1: Important Printer and Slicer Settings

Setting	**Value**
Nozzle Temperature	200°C
Bed Temperature	80°C
Layer Height	0.2 mm
Infill Pattern	Rectilinear (Lines)
Print Speed	80 mm/sec

A post-processing script to pause after the channels had been printed, specifically, a height of 0.35 inches or 8.89 millimeters. G-code was produced for the 10% infill percentage, the 20% infill percentage, and the 30% infill percentage and saved to an SD card. A gCreate gMax 1.5 XT+ printer with heated bed addition and PLA filament from Hatchbox was used for all three prints.

8 voltage divider circuits were designed and built on a breadboard using an Arduino Mega, 10kOhm resistors, wiring, and 20kOhm NTC thermistors from Vishay/BC Components, model number NTCLG100E2203JB.

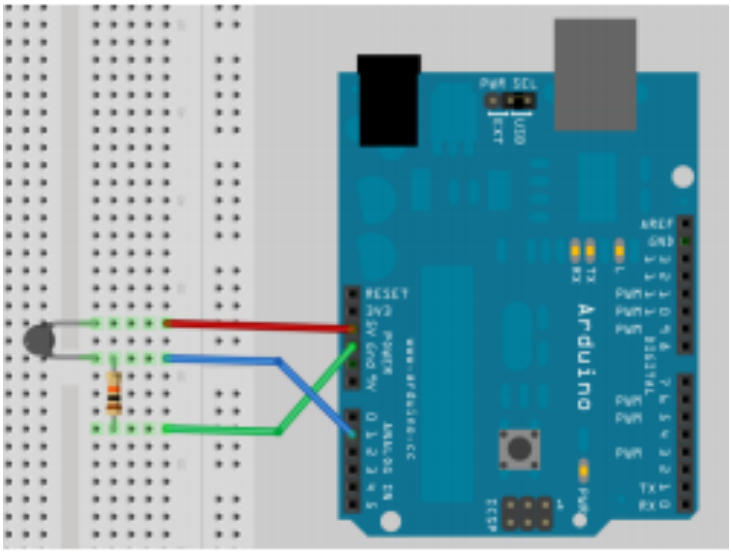

Figure 3.
A single Arduino voltage divider circuit for temperature measurement using thermistors. Voltage for the thermistor is measured using pin A0.

Voltage at each thermistor was measured from each Arduino pin, and the voltage was converted into temperature using the Steinhart-Hart Equation, T is the thermistor temperature in Celsius; T_A, T_B, and T_C represent the thermistor alpha, beta, and gamma coefficients

respectively; R0 represents the fixed resistance in the circuit, and V represents the analog thermistor reading.

$$T = \frac{1}{T_A + \left[T_B \ln\left(R_0\left(\frac{1024}{V} - 1\right)\right)\right] + \left[T_C \ln^3\left(R_0\left(\frac{1024}{V} - 1\right)\right)\right]} - 273.15 \quad (3)$$

Thermistor coefficients were determined by using a datasheet provided by the manufacturer.

Table 2: Thermistor Coefficients

Coefficient	**Value**
T_A	0.003354016
T_B	0.0002569850
T_C	0.000002620131

The Arduino was programmed to read the voltages, calculate temperature in Celsius, and output temperature values to a serial port for every second. puTTY was used to read the values in the serial port and log them to a text file.

After each of the prints paused, the thermistors were embedded into the channels. Gloves were worn to prevent burning, and safety glasses were worn to prevent injury. Electrical tape, glue, and forceps were used to place each of the thermistors in the channels. Four thermistors were placed in one side and wired to the main channel and to an opening on the right, and another four were placed on the other side and wired to the main channel and to an opening on the left. The wires from the openings were connected to the voltage divider breadboard circuit.

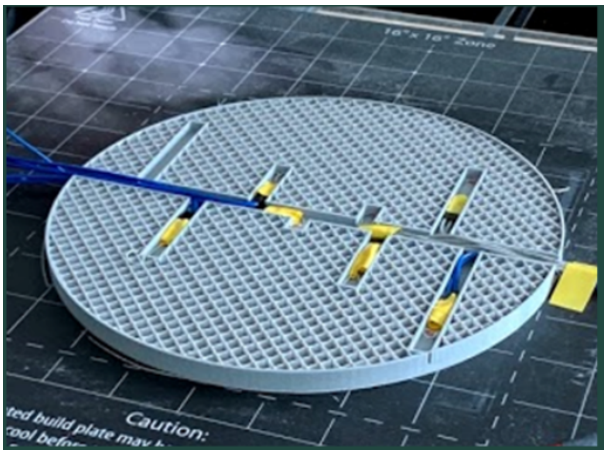

Figure 4.
Embedding thermistors into channels of a 10% infill print.

After embedding thermistors, puTTY was started and the Arduino's COM port, which was COM13, was selected for logging of all session output into a text file. The print was resumed such that PLA would cover the channels and print the rest of the disc. This process of printing and embedding thermistors was repeated for the 10%, 20%, and 30% infill percentage prints.

Data

Because the raw data from the thermistors is extremely large with each text file containing over 100,000 rows, only a sample is shown below in Tables 3, 4, and 5.

Table 3: Raw Data from 10% Infill

Time (sec)	Thermistor 1 (°C)	Thermistor 2 (°C)	Thermistor 3 (°C)	Thermistor 4 (°C)	Thermistor 5 (°C)	Thermistor 6 (°C)	Thermistor 7 (°C)	Thermistor 8 (°C)
0	26.76	25.07	24.36	26.76	25.27	23.86	24.46	32.51
1	26.66	25.07	24.26	26.86	25.27	23.75	24.36	32.51
2	26.76	25.07	24.26	26.86	25.27	23.86	24.46	32.61
3	26.76	25.07	24.26	26.76	25.27	23.75	24.46	32.51
4	26.76	25.07	24.36	26.86	25.27	23.75	24.36	32.51
5	26.76	25.07	24.36	26.86	25.27	23.75	24.36	32.61
6	26.76	25.07	24.26	26.86	25.27	23.86	24.46	32.61

Table 4: Raw Data from 20% Infill

Time (sec)	Thermistor 1 (°C)	Thermistor 2 (°C)	Thermistor 3 (°C)	Thermistor 4 (°C)	Thermistor 5 (°C)	Thermistor 6 (°C)	Thermistor 7 (°C)	Thermistor 8 (°C)
0	32.99	33.96	35.69	33.76	28.34	28.63	28.04	32.7
1	32.99	33.67	35.3	33.47	28.24	28.24	28.24	32.51
2	33.28	33.47	36.07	33.28	28.04	28.24	28.43	32.32
3	33.38	33.28	35.78	33.67	28.43	28.34	28.43	32.32

4	33.47	33.38	35.11	33.86	28.92	28.53	28.24	32.32
5	33.47	33.19	35.98	34.15	29.02	28.73	28.43	32.41
6	33.57	33.09	35.69	33.96	28.83	28.53	28.43	32.32

Table 5: Raw Data from 20% Infill

Time (sec)	**Thermistor 1 (°C)**	**Thermistor 2 (°C)**	**Thermistor 3 (°C)**	**Thermistor 4 (°C)**	**Thermistor 5 (°C)**	**Thermistor 6 (°C)**	**Thermistor 7 (°C)**	**Thermistor 8 (°C)**
0	35.30	25.97	26.36	32.9	31.26	27.75	27.25	33.28
1	35.30	25.97	26.46	32.61	33.19	28.14	27.16	33.38
2	35.21	26.07	26.46	32.12	35.11	28.73	26.86	33.47
3	35.01	26.07	26.36	31.84	36.75	29.31	26.96	33.47
4	34.82	26.07	26.17	31.64	37.81	29.99	26.96	33.38
5	34.73	26.07	26.17	31.35	37.23	30.09	27.06	33.38
6	34.53	26.07	26.17	30.97	36.55	30.29	27.06	33.47

Results

Due to the sheer size of the data, Python was used for data analysis with the following libraries: NumPy, SciPy, os, Matplotlib, and pandas. The data was loaded into multiple NumPy arrays, and data for the first 10,000 seconds was graphed as shown in Figures 5, 6, and 7.

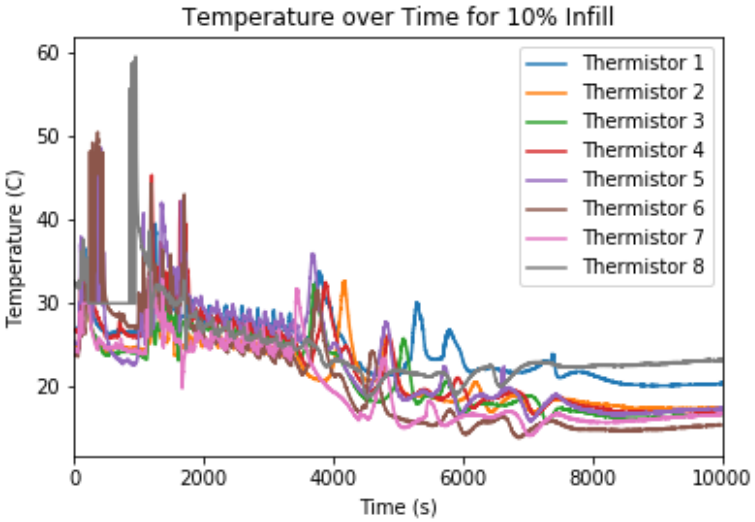

Figure 5. Raw Data Graph of Temperature over Time for 10% Infill.

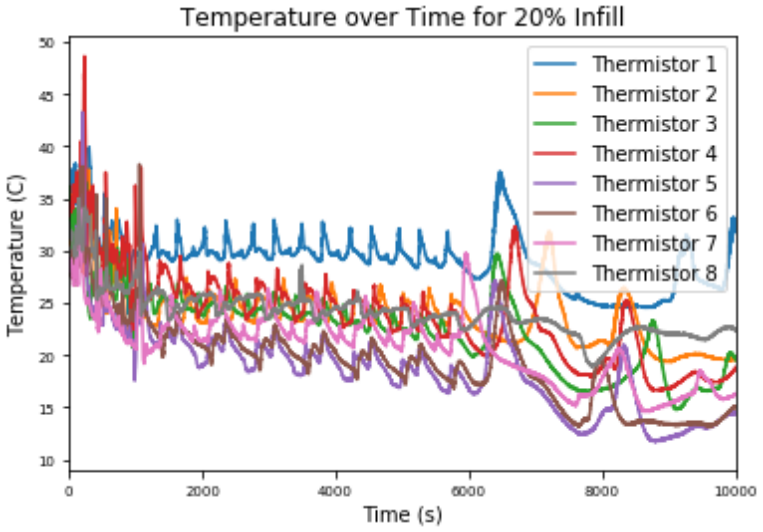

Figure 6. Raw Data Graph of Temperature over Time for 20% Infill.

,

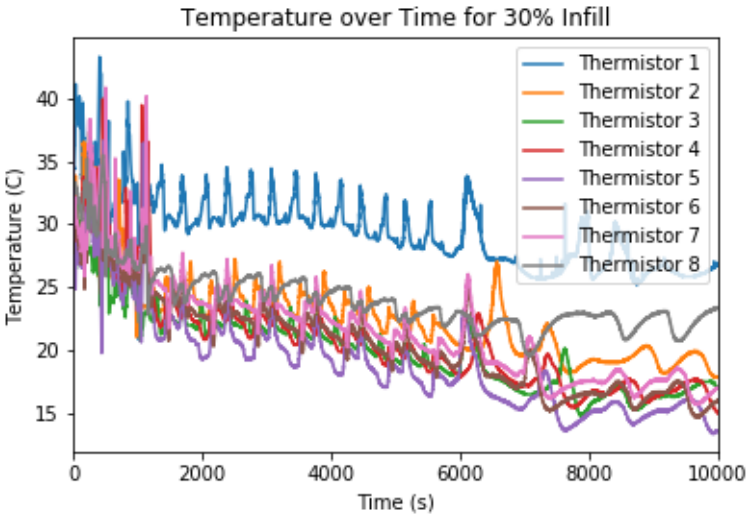

Figure 7. Raw Data Graph of Temperature over Time for 30% Infill.

The entire thermal system of the disc is constantly being influenced by the external factors, such as the room temperature and the printer extruder. Figures 5, 6, and 7 show multiple spikes in temperature for all the thermistors, which can be explained by the nozzle printing over the thermistor location. However, at any time *t* in seconds, the print is theoretically at a steady-state equilibrium, but as *t* increases, the equilibrium is moving.

Since temperature is a scalar function of the x and y position variables, the temperature at any point at a time *t* can be defined as *T(x, y)*. However, since *t* is changing and external factors change the entire system, the temperature function *T(x, y)* is re-defined for every *t*. This means that the function for temperature, based on 8 discrete points from the raw thermistor data, is changing throughout the entire print. However, the temperature function is difficult to model and derive using only 8 points, and it is equally difficult to compare functions between every time *t*.

Plotting all points on a Cartesian coordinate system using the x and y coordinates for each thermistor, determined using the CAD software, shows a map of where every thermistor is located. Table 6 shows the x and y coordinates for each thermistor, and Figure 8 shows the map of thermistor locations.

Table 6: Thermistor Locations in the Cartesian System

Thermistor	**X (mm)**	**Y (mm)**
1	70.0	-70.0
2	55.0	56.0
3	32.4	-41.8
4	24.0	15.0
5	0.0	-12.0
6	-22.5	33
7	-45	48
8	-72.5	56

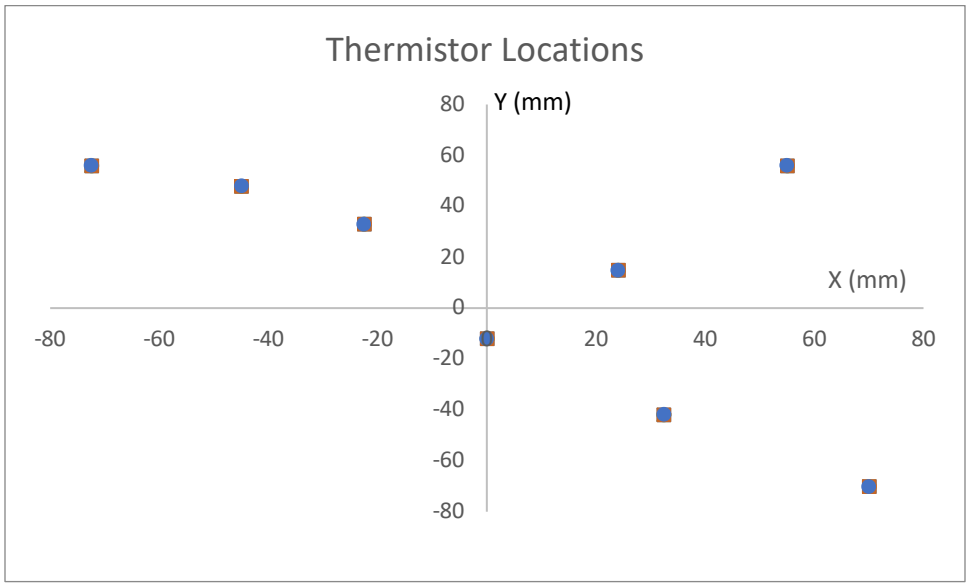

Figure 8. Locations of all thermistors in the Cartesian coordinate system.

As temperature is a scalar function, each of the discrete points has a specific real value for the temperature. Therefore, each of the points can be assigned a "weight" that represents the temperature at that point. As the system is at a steady state at time *t*, it can be modeled using Fourier's Law of Heat Conduction (Equation 1). The differential form of Fourier's Law (Equation 4) shows that the rate of temperature change is directly proportional to the rate of heat flow; therefore, the temperatures can be assumed to be linear and a center of temperature was calculated similar to the calculation of center of mass or gravity, except with the weight substituted by a temperature, for every time *t*.

$$Q = -kA\frac{dT}{dt} \tag{4}$$

Over time, the center of temperature will constantly be moving because of the primary external heat source, the nozzle, will also be moving. However, when all temperatures are 0 and, consequently, all points have a weight of 0, the center of temperature represents a perfect steady-state thermal equilibrium point when the temperature is perfectly distributed to all points. The movement of the center of temperature from the equilibrium point to a temperature center at *t* represents the net movement of heat in the object.

Each of the center points is defined with three values, each representing the x position in millimeters, y position in millimeters, and temperature in Celsius. The equilibrium point, C_e, for the 8 chosen thermistor locations, when all temperatures are 0, was found to be at the point represented by (5.175, 10.525, 0).

In this system, the center of temperature at a time *t*, C_t, was calculated by finding the weighted mean of positions in the horizontal (x) and vertical (y) directions as shown by Equation 5.

$$C_t = \left(\frac{\sum x_i T_i}{\sum T_i}, \frac{\sum y_i T_i}{\sum T_i}\right) \quad (5)$$

The temperature at C_t was found by taking the weighted mean of temperatures for each of the points as shown by Equation 6,

$$T = \frac{\sum r_i T_i}{\sum r_i} \quad (6)$$

where r_i refers to the distance from the equilibrium point C_e to the point *i* within the 8 thermistors.

The following python function (Figure 9), where the variable *points* refers to a list of the 8 points in the form of (X, Y, Temperature), implemented Equation 5 and 6 to process the raw data and find the center of temperatures for all times.

```
def center(points):
    global equilibrium
    global x_graph_list
    global y_graph_list
    weighted_mean_temp = sum((((((p[0])**2) +
                               ((p[1])**2))**0.5)
                               * p[2]) for p in points)
    tmp = (sum(((((point[0])**2) + ((point[1])**2))**0.5) for point in points))
    weighted_mean_temp = weighted_mean_temp / tmp

    x = sum([point[0] * point[2] for point in points]) / sum([point[2] for point in points])
    y = sum([point[1] * point[2] for point in points]) / sum([point[2] for point in points])
    return (x, y, weighted_mean_temp)
```

Figure 9. Python function to calculate center of temperature for any time t as well as the temperature at the center.

The calculated values for each infill percentage were recorded and graphed as shown in Figures 10, 11, and 12, where the red point is C_e and each blue point is a C_t.

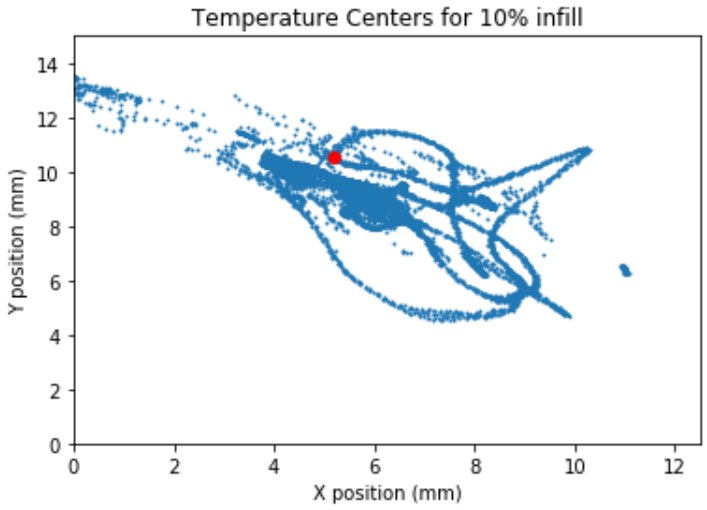

Figure 10.
10% infill temperature centers.
Standard Deviation in X: 0.87
Standard Deviation in Y: 0.59

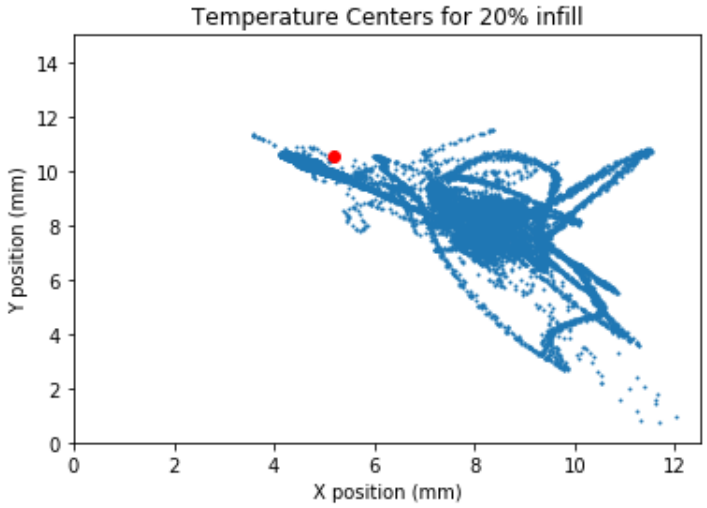

Figure 11.
20% infill temperature centers.
Standard Deviation in X: 1.40
Standard Deviation in Y: 0.94

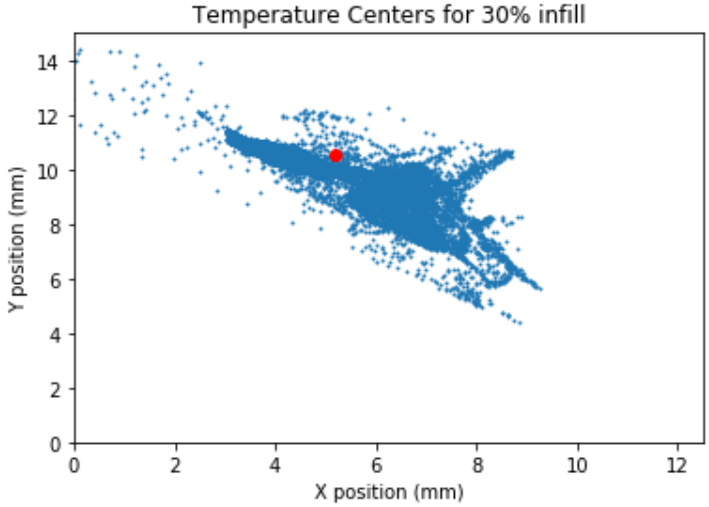

Figure 12.
30% infill temperature centers.
Standard Deviation in X: 0.87
Standard Deviation in Y: 0.64

Figures 10, 11, and 12 suggest that as the infill percentage is increased, the center of temperature locations tend to increasingly cluster around the equilibrium point. However, to ensure there was a statistically significant difference in the center of temperatures, a numerical value that corresponded to the difference between temperature points was calculated.

By taking the derivative of the scalar function for temperature, a gradient function can be found. However, since no actual function is defined for the temperature, the temperature gradient was calculated from the equilibrium point to each center of temperature. By using the temperature gradient as the numerical value for comparison, the direction and magnitude of the movement of temperature in terms of the location but also in terms of the actual change in temperature could be compared. The gradient vector was also used because it showed the direction and magnitude of temperature change in the direction of the fastest change in temperature, or the steepest ascent/descent. Horizontal and vertical components of the gradient vectors of the temperature function, ∇T, from C_e to each C_t was calculated using Equation 7, where *T* referred to the temperature function *T(x, y)*.

$$\nabla T = \left(\frac{\partial T}{\partial x}, \frac{\partial T}{\partial y}\right) \tag{7}$$

Figure 13 shows the python implementation of Equation 7 for this study, where *centers* represents a list of temperature centers calculated by the function in Figure 9, and the resultant gradient vector and direction was calculated and stored after finding the component vectors.

```
gradients = [[] for _ in range(3)]
for infill in range(0, 3):
    print("Calculating gradients for {}% infill...".format((infill + 1)*10), end="", flush=True)
    for c in centers[infill]:
        dx = c[0] - equilibrium[0]
        dy = c[1] - equilibrium[1]
        dt = c[2] - equilibrium[2]
        gradT = np.sqrt( ((dt / dx)**2) + ((dt / dy)**2) )
        direction = np.degrees(np.arctan(dy / dx))
        gradients[infill].append((dx, dy, dt, gradT, direction))
    print("Done!")
```

Figure 13. Python function to calculate gradient from C_e to all centers of temperature for each infill.

All compiled results of gradient vector magnitude, direction, X coordinate of center, Y coordinate of center, and temperature at center for every time *t* and each infill percentage was recorded. Due to the size of the data, only a sample of these compiled results is shown in Tables 7, 8, and 9.

Table 7: Compiled Results for 10% Infill

Time (s)	X (mm)	Y (mm)	Mean Temperature (C)	Gradient (C/mm)	Angle (deg)
0	-1.878999043	0.92035757	26.72183284	32.33001219	-26.09619902
1	-1.87824806	0.95771534	26.67198258	31.26104358	-27.01695348
2	-1.919255319	0.968817834	26.73615369	30.91334643	-26.7841149
3	-1.879347826	0.934500096	26.70114544	31.91009507	-26.43872471
4	-1.832394467	0.893215756	26.7044345	33.25981921	-25.98729966
5	-1.868675852	0.914542671	26.72400826	32.53300805	-26.0774403
6	-1.919255319	0.968817834	26.73615369	30.91334643	-26.7841149

Table 8: Compiled Results for 20% Infill

Time (s)	X (mm)	Y (mm)	Mean Temperature (C)	Gradient (C/mm)	Angle (deg)
0	2.172097714	-1.124157845	32.26039581	32.31296273	-27.36358876
1	2.127204544	-1.137483179	32.10523252	32.00669311	-28.13481732
2	2.251571327	-1.414290088	32.1929441	26.8806485	-32.13435565
3	2.214315144	-1.436382723	32.19157811	26.71384853	-32.97068234
4	2.20426565	-1.359133081	32.16523243	27.8031189	-31.65766126
5	2.179538335	-1.505202052	32.31317359	26.08949825	-34.62918762
6	2.20166457	-1.521038047	32.23373871	25.7574436	-34.63897422

Table 9: Compiled Results for 30% Infill

Time (s)	X (mm)	Y (mm)	Mean Temperature (C)	Gradient (C/mm)	Angle (deg)
0	-0.157452291	-1.586464586	30.51580321	194.7620156	84.33211084
1	-0.364016749	-1.659845817	30.66673453	86.24752852	77.63042885
2	-0.655421223	-1.618066019	30.83497631	50.75911377	67.94887412
3	-0.949028581	-1.53173571	30.93752419	38.34907083	58.21858722
4	-1.232149121	-1.372992984	31.0269595	33.83434878	48.09461443
5	-1.300172924	-1.277173044	30.99810705	34.02195972	44.48871394
6	-1.461243683	-1.097415176	30.97818964	35.30259309	36.90708008

The resultant gradient vectors for each infill percentage were then compared using two Welch's Tests, or two-sample T-tests for unequal variances, since the size of each dataset for each infill percentage was not equal. The null hypotheses for each of the tests was that the mean resultant gradient for the 10% infill percentage print was equal to that of the 20% infill percentage print, and likewise between the 20% infill percentage print and the 30% infill percentage print. The alternative hypothesis for each test was that the greater infill percentage had a greater mean resultant gradient. The results for the Welch's Tests are shown in Table 10.

Table 10: Welch's Tests p-values and Test Statistics

10% and 20% Infills	
Test Statistic	-34.9745
p-value (two-tailed)	<0.0001
p-value (one-tailed)	<0.0001
20% and 30% Infills	
Test Statistic	8.4609

p-value (two-tailed)	< 0.0001
p-value (one-tailed)	< 0.0001

Because the p-values for both tests were both less than the established significance level of α=0.05, the null hypothesis can be rejected. The Welch's Tests showed that the mean gradients for the 30% infill were significantly greater ($p<0.0001$) than the mean gradients for 20% infill. Additionally, the mean gradients for the 20% infill were also significantly greater ($p<0.0001$) than the mean gradients for the 10% infill.

Discussion

The initial hypothesis for this study was that greater infill percentages will have more uniform temperature distributions of temperature because conduction is more effective for heat transfer than natural convection or radiation.

The results from the Welch's Tests show that the mean gradient increases as infill percentage increases as well. This means that net movement of heat increases with infill percentage, and based on Fourier's Law of Heat Conduction (Equation 1), since the temperature gradient is directly proportional to the heat flux density, the heat flux density across the printed object as well as the net heat flow increase as infill percentage increases.

Figures 10, 11, and 12 support this as the temperature centers tend to cluster around the equilibrium point as the infill percentage is increased. This means that the distribution of temperature is becoming increasingly uniform. Since warpage occurs on the edges and corners of an object, the results of this study support the solution of increasing infill percentage to "move" the heat into the center of the object and evenly distribute temperature throughout the entire object, reducing the likelihood of warpage occurring.

Therefore, the data in this experiment supports the initial hypothesis, but the reasoning behind the hypothesis is flawed. Although it seems intuitive that the larger volume of plastic in an object from higher infill percentages would conduct more heat, the data shows the opposite. Thermal conductivity of an object is usually measured when the heat flow into the object is constant, but when this is true, an algebraic manipulation of Fourier's Law of Heat Conduction shows that there exists an inverse relationship between temperature gradient and thermal conductivity *k* when the rate of heat flow/heat flux density is kept constant.

$$\nabla T = \frac{-\vec{q}}{k}$$

Based on Fourier's Law of Heat Conduction, the increased temperature gradient caused by increasing infill percentage actually decreases the thermal conductivity of the object. This can be explained by the fact that within the actual print, lower infill percentages create larger air gaps for natural convection to occur. The air in these gaps is constantly heated by the rest of the object and only cools once the entire print has finished. However, increasing the infill percentage reduces the size of these air gaps, meaning that there is less hot air trapped in the print during the printing process. Thus, thermal conductivity decreases as the infill percentage is increased.

It can be concluded that increasing infill percentage is a viable solution to the warpage problem. However, systematic errors in the experimental design could limit the validity of this conclusion. Humidity plays a key role in the thermal properties of the print, and as it was not controlled for in this experiment, it may have been a confounding variable that influenced the results of the experiment. Additionally, in this study, PLA was used because it is most commonly used filament for 3D printing. However, the use of PLA, along with the use of a heated bed, causes the print to be less susceptible to warpage. However, temperature distribution and other thermal properties are likely similar when compared to printing with other filament or without the use of a heated bed. In the study, only three prints were made due to a time constraints and other obstacles in the design and use of the 3D printer. Therefore, repeating the experiment with multiple trials of each infill would reduce the variability of data for each infill percentage and allow for more accurate generalizations of the relationship, which may not be persistent for infill percentages past 30%, as higher infill percentages were not studied. Increasing infill percentage may not be a practical solution to the warpage problem. Large-scale prints would have to use substantially higher amounts of filament, leading to higher costs and longer print times, mitigating the benefit of 3D printing for rapid prototyping.

Future work should study other variables that influence warpage, such as infill type, filament material (such as ABS), bed temperature, or even filament color. More trials should be conducted with higher infill percentages to generalize the relationship to high infill percentages other than just 10%, 20%, and 20%. Finally, to accurately analyze the temperature gradients, the scalar temperature function must be known. In this study, it is difficult to generate a function from 8 discrete points to map temperature. Current research on methods to derive this function can be continued by using methods such as by using Deep Learning and Artificial Neural Networks for mapping function generation or by using analysis and simulation software like ANSYS to derive a function.

Ultimately, increasing infill percentage can be used as a solution to the warpage problem in 3D printing, but future research will have to be conducted to support the results of this study.

Appendix A: Full Python Script for Data Analysis

Written and compiled in Anaconda Distribution Python 3.7 with NumPy, os, Matplotlib, pandas, and SciPy packages installed.

Code begins on next page.

EFFECT OF INFILL PARAMETERS ON SPATIOTEMPORAL THERMAL DISTRIBUTION IN FUSED DEPOSITION MODELING

```
# -*-
coding:
utf-8 -
*-
            """
            @author: pranavaddepalli
            """
            #%% SETUP and LOAD RAW DATA

            import numpy as np
            import os
            import matplotlib.pyplot as plt
            import pandas as pd
            import scipy.stats as stats


            np.set_printoptions(precision=3, suppress=True)

            base_dir = os.getcwd()
            data_dir = base_dir + "/Data/"

            raw_10_percent = np.genfromtxt(data_dir + '10pLines', delimiter=',')
            print("10% infill data has {} columns and {}
            rows.".format(np.size(raw_10_percent, axis=1), np.size(raw_10_percent, axis=0)))

            raw_20_percent = np.genfromtxt(data_dir + '20pLines (1)', delimiter=',')[:, :8]
            print("20% infill data has {} columns and {}
            rows.".format(np.size(raw_20_percent, axis=1), np.size(raw_20_percent, axis=0)))

            raw_30_percent = np.genfromtxt(data_dir + '30pLines', delimiter=',')
            print("30% infill data has {} columns and {}
            rows.".format(np.size(raw_30_percent, axis=1), np.size(raw_30_percent, axis=0)))
            #%% PROCESS RAW DATA

            def point(raw, row, col):
                n = col
                temperature = raw[row, col]
                if col == 0:
                     x = 70
                     y = -70
                elif col == 1:
                    x = 55
                    y = 56
```

Appendix B: Full Arduino Code for Recording Raw Temperature Data