

Numerical Modeling of Thermal Predictions in Wire-Arc Additive Manufacturing for Online Implementations

Kathryn Kelly^{*†}, Andrew Hill^{*}, Dr. Christopher Saldaña^{*}, Dr. Kyle Saleeby^{*}

**George W. Woodruff School of Mechanical Engineering
Georgia Institute of Technology
801 Ferst Drive, Atlanta, GA 30332, USA
†kkelly97@gatech.edu*

Abstract

Robotic wire-arc additive manufacturing (R-WAAM) has experienced greater industry adoption due to its balance of cost and flexibility in additive manufacturing. While existing numerical thermal models have been established for traditional gas-metal arc welding (GMAW) methods, no such model has been created utilizing aluminum and the lower heat-input cold metal transfer (CMT) process. These models use external software, making them less suited for implementation in online control methods. In this work, a Python-based thermal model is presented that uses the thermal constants of aluminum and experimentally determined inputs to create a simulation that is accurate for individualized systems with the increasing part scale that R-WAAM is capable of. Simulation predictions had a mean error of 10% with an improvement of 6-15% over deposition time. This thermal model can be implemented in future work as part of an online control schema supporting temperature sensors where readings may be inaccessible.

Introduction

Wire-arc additive manufacturing (WAAM) is one of the most widely studied additive manufacturing techniques in the deposition of metals. In this direct energy deposition (DED) process, a wire material is fed through the torch and melted by an electric arc between the wire and substrate, leading to a layer-by-layer deposition of material [1]. WAAM has been widely praised for its reduction in material cost of 7-69% and its higher than average deposition rates of 15-130 grams per minute when compared to other AM methods [2]. There are various welding techniques for WAAM that are applied: tungsten inert gas (TIG) welding, metal inert gas welding (MIG) welding, and cold metal transfer (CMT) welding. The last welding technique, CMT, focuses on reducing heat input and spatter much more than the other two methods while still maintaining significant arc stability and control. This is accomplished through the oscillatory movement of the wire feeder allowing for the arc melting process to be a series of short circuits instead of a continuous arc like in other methods [3].

One of the major benefactors of the CMT process is aluminum welding, which is traditionally difficult to deposit due to its high sensitivity to heat. However, by combining CMT welding with wire-arc additive manufacturing, aluminum parts can be produced with greater control and reduced thermal distortion. Aluminum is often the focus of industry study as it has a smaller weight-to-strength ratio and is cost-effective when compared to other materials [2]. Because the heat input to aluminum welding is so important, researchers have often sought to observe and predict the behavior of the temperature distribution in aluminum WAAM. Many

methods exist in literature, ranging from use of machine learning (ML) models to help the predictions to pure numerical methods.

In numerical methods, equations are derived by researchers and solved programmatically in their simulation. Typical methods explore implementations of a Gaussian distribution of the heat source. One work in powder bed focuses on deriving these equations from Rosenthal’s moving heat source equation [4], while another in wire-arc derives the equations using the velocity and pressure distribution of the molten metal [5]. However, a more complex answer involves adjusting the gaussian distribution in an elliptical shape. Gupta et al. [6] is only one example of many that implements the Goldak double ellipsoid heat-source model [7] in WAAM. In combining these numerical methods with machine learning, results can be obtained much faster. A study by Le et al. focuses on utilizing five experimentally-obtained datasets to train a feed-forward neural network (FFNN) [8]. In this work, the time to create the training dataset and training time of the model took 26 hours total resulting in a 38-second computation time for a single temperature evaluation.

Existing methods use external FEA solver architectures like ANSYS [6, 8], Abaqus [9], LS-DYNA [10], or Cast3M [11]. To the authors’ knowledge, there is no known fully Pythonic implementation of numerical heat simulation in the WAAM literature. This work explores and proposes a purely Pythonic model with a runtime that is fast enough to be incorporated into future online control methods. The key component here is the “Pythonic” aspect, allowing data to be transferred seamlessly throughout any additive manufacturing system with Python communication. Utilizing experimentally determined variables such as voltage and current of the welder, as well as known inputs like wire feed speed (WFS) and traverse speed (TS), the model can accurately predict the temperatures of aluminum parts.

Main Method

A. Simulation Set-Up

To explore how heat behaves during aluminum WAAM under different deposition strategies, a high-fidelity simulation framework was built using Python and the NumPy library [12]. The model was designed to be modular, efficient, and scalable—with an eye toward future integration into real-time robotic WAAM (R-WAAM) control systems. The simulation runs on a 3D Cartesian grid that represents both the aluminum substrate, and the material deposited during the welding process. The domain is represented as voxels, where each voxel stands for a fixed physical volume of 1 mm^3 and is initialized at an ambient temperature of 300 K. All numerical operations and grid setup are handled with NumPy, taking full advantage of its vectorization and broadcasting for speed and simplicity. Temperature data, thermal properties, and material states are stored in NumPy arrays. Two masks are used to track what each voxel represents: the substrate mask, initialized at the beginning to define the base plate and assign it aluminum thermal properties; and the bead mask, updated on the fly during deposition to add new material and update local properties. This dual-mask setup allows for dynamically capturing thermal changes as they happen in real time, making the simulation responsive to ongoing material changes.

Two different deposition strategies are evaluated, each with a specific substrate configuration designed to minimize boundary effects. Substrate sizes were chosen to give a 25.4 mm clearance on all sides for heat to naturally diffuse. The first deposition strategy was a straight 127 mm single-bead wall laid along the x-axis on a 152.4×50.8 mm substrate. The path plan reversed direction every other layer to reduce directional heat and material buildup. The second deposition strategy was a 127×127 mm closed single-bead square deposited on a larger 203.2×203.2 mm substrate. The trajectory remained in the same direction but alternated which corner the deposition started in a counterclockwise direction. A visualization of both building strategies can be seen in Figure 1.

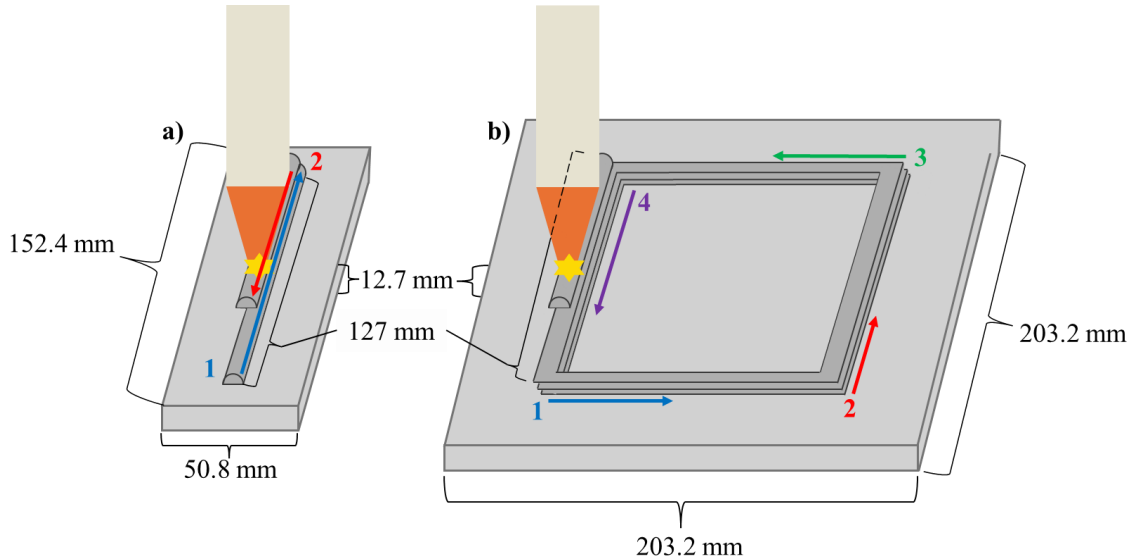


Figure 1: a) Straight wall path plan with back-and-forth motion; b) wall path plan with alternating start points.

Each deposition point adds a direct heat input, calculated using the average current (\bar{I}) and average voltage (\bar{U}) during the CMT process. These values were experimentally found through preliminary depositions of single-bead walls using the wire-arc system at the Georgia Institute of Technology [13] and are described in the next section. A fixed thermal efficiency ($\eta=0.8$, experimentally determined in [14]) was used. Therefore, to find the net power input Q in Watts added at each step, (1), [15] combined the power and thermal efficiency of our CMT process for aluminum welding (originally derived in [15]).

$$Q = \eta(\bar{U}\bar{I}) \quad (1), [15]$$

The increase in temperature is calculated using our derived heat input and Rosenthal's equation for a moving point source, found in (2). We define our variables as: $T(x, y, z)$, the temperature at point (x, y, z) (Kelvin); T_i , the temperature grid at the previous timestep (K); k , the material-specific thermal conductivity (W/mm·K); v , our travel speed (mm/s); and γ , the material-specific thermal diffusivity (mm²/s). The thermal diffusivity is calculated from the equation $\gamma = k/(\rho \cdot c)$, where ρ is the density (kg/mm³), and c is the specific heat capacity (J/kg·K).

The values of r and $(x - x_0)$ require further explanation. First, r is defined as the distance between the heat source and each point in the temperature field in mm, calculated by the Euclidean distance $r = \sqrt{(x - x_T)^2 + (y - y_T)^2 + (z - z_T)^2}$ where (x_T, y_T, z_T) is the 3D location of the welding torch, the heat input to the simulation. $s = \sqrt{(x_T - x_c)^2 + (y_T - y_c)^2}$ where (x_c, y_c) is the 2D distance to the point where the torch changes direction (for the case of the wall, the end of the bead; for the case of the square, the next corner where the torch will then traverse in a new direction in x and y). We do not calculate $z_T - z_c$ because it will always be equal to zero given that our torch is on the same layer height as our present path plan.

$$T(x, y, z) = T_i + \frac{Q}{2\pi kr} \cdot e^{-\frac{vs}{2\gamma}} \quad (2)$$

After each deposition step, conductive and convective cooling is calculated. First, the temperature field is updated using the 3D transient heat conduction equation (3), solved using a finite differences approach. NumPy's *np.roll* function makes this process fast by efficiently computing second-order central differences in all directions, shifting the temperature grid to access neighbors without loops. The bead and substrate material properties are accounted for separately to allow for more accurate temperature distribution. In (3), $(\partial^2 T / \partial x^2 + \partial^2 T / \partial y^2 + \partial^2 T / \partial z^2) = \nabla^2 T$ represents the Laplacian of the temperature field and $\partial T / \partial t$ is the change in temperature over time. It is noted that α is dependent on the latest material layout and therefore is the thermal diffusivity of the *temperature field* rather than of the *material* (γ). The field thermal diffusivity is therefore updated voxel-by-voxel using the thermal conductivity of the *temperature field* (λ), which is derived from the thermal conductivity of the *material* (k). Here, our thermal conductivity is calculated by the equation $\lambda = -k \nabla T$, where ∇T is the 3-D gradient of the temperature field. To recalculate α we use the equation $\alpha = \lambda / (\rho \cdot c)$.

$$\alpha \left(\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} + \frac{\partial^2 T}{\partial z^2} \right) = \frac{\partial T}{\partial t} \quad (3)$$

Between layers, we simulate a 10 second dwell time to serve as the simulations cooling period using Newton's Law of Cooling (4). This enables measurement of heat build-up over multiple layers. Note that the welding system used for validation of the simulation does not have any active cooling measures, and so similarly the simulation only uses convective cooling. In the equation, T_{env} is the ambient temperature (K) set to 300K and $\Delta t = \psi / v$ is the time between each movement step in the simulation (seconds), where $\psi = 1$ mm (the voxel size selected for each timestep). The parameter characterizing the response to the step-input is $\tau = \rho V c / h A$, where h is the heat transfer coefficient (W/mm²K), V is the volume of the bead (mm³), and A is the bead's surface area (mm²), of which the latter two were increased by a constant value for each time step.

$$T(t) = T_{env} + (T_i - T_{env}) \cdot e^{-\Delta t / \tau} \quad (4)$$

In these calculations, some assumptions were made to increase the efficiency of the algorithm. These assumptions were:

- Temperature distribution is a first-order linear time invariant system.
- Bead geometry is a rectangular prism.
- A constant mass of wire is deposited at each time step.
- The ambient temperature is always at 300 K.
- Each material has set values for conductivity and diffusivity, regardless of temperature.
- Phase changes are not modeled (melting, solidifying, or latent heat from the weld).
- Heat loss is modeled through Newtonian convection, while radiation and fluid flow are left out.
- Each bead has fixed dimensions that were experimentally determined from the preliminary experiments performed to find current & voltage.

In the real world, the values of conductivity and diffusivity for aluminum vary with temperature. However, for this work, constant values are used just before the melting point of aluminum (in other words, just after solidification of the material). As previously stated, the values of material diffusivity (γ) and conductivity (k) remain constant but are used in calculations for the overall *temperature field* diffusivity and conductivity values, which are greatly affected by temperature of the part. Taking this macroscopic view of the thermal distribution similarly allows phase changes not to be modeled within the simulation while still maintaining enough accuracy for the goals of this research.

The full algorithm implemented in Python can be seen below. First, the path plan and all material constants are initialized to minimize calculations throughout the rest of the simulation. Then, a 3D mask is initiated as a 3D matrix of zeroes covering the full build volume. The substrate volume is then changed to values of “1”; as each voxel of the bead is deposited, similarly, the mask value changes from “0” to “1”. In doing this, the heat equations only happen in voxels where a “1” is found in the mask, ensuring conduction and convection happens only in existing material instead of to-be-deposited material.

There are two phases for the simulation: the “deposition” phase, where heat is applied, and the “cooling” phase, where a dwell period is simulated. In the “deposition” phase, the temperature increase is calculated from the direct moving point heat source and the conduction throughout the part is also calculated; in the “cooling” phase we calculate both the conductive spread of heat and the convective cooling of the part. Maximum, average, and minimum temperatures are logged for every timestep regardless of whether we’re in the “deposition” phase or “cooling” phase. All data is saved in layer-specific CSV files for later analysis and comparison with real-world measurements. A live Matplotlib heatmap shows the temperature evolution during the build, allowing visual of the process in real time.

ALGORITHM: HEAT MODEL SIMULATION

1	Initialize path plan, temperature grid, and heat & material constants
2	Initialize bead and substrate masks
3	While <i>running simulation</i>
4	Initialize visualization for simulation and temperature scale

5	For <i>layer in numlayers</i>
6	Update <i>dz</i> variable
7	Update heat input equation based on layer selection
8	Generate list of bead positions from path planner
9	For each position pair (previous and current) in path
10	Convert <i>torch position (x, y)</i> to grid indices
11	Clamp <i>grid indices</i> to grid boundaries
12	Mark <i>bead mask (torch position)</i> as active
13	Calculate <i>step length</i> between positions
14	Call <i>apply heat source</i> to temperature grid at current position
15	Call <i>update temperature xyz</i> to temperature grid at current position
16	Update total simulation time
17	Write time, average, min, and max temperatures for current layer to <i>csvwriter</i>
18	Update plot and display with new temperature data
19	Adjust color scale to reflect new max temperature
20	Apply <i>log_dwell_cooling</i> function
21	Log time, average, min, and max temperatures for current layer during dwell time to <i>global CSV</i>
22	Stop updating display
23	Show final display
24	Stop logging to <i>csvwriter</i>
25	End <i>running simulation</i>

B. WAAM CMT Cell

All experiments will take place utilizing the WAAM cell as pictured in Figure 2. A Fronius TPS/I 400 power supply is connected to a Robacta CMT drive on the welding torch which has been integrated into a Fanuc LR Mate 200iD/7L robot. The build plate substrate is rigidly held by a vise, which in turn is bolted to a heavy steel table. The welding plate of the robot is significantly below the base of the robot to allow for a greater vertical workspace in other research work.

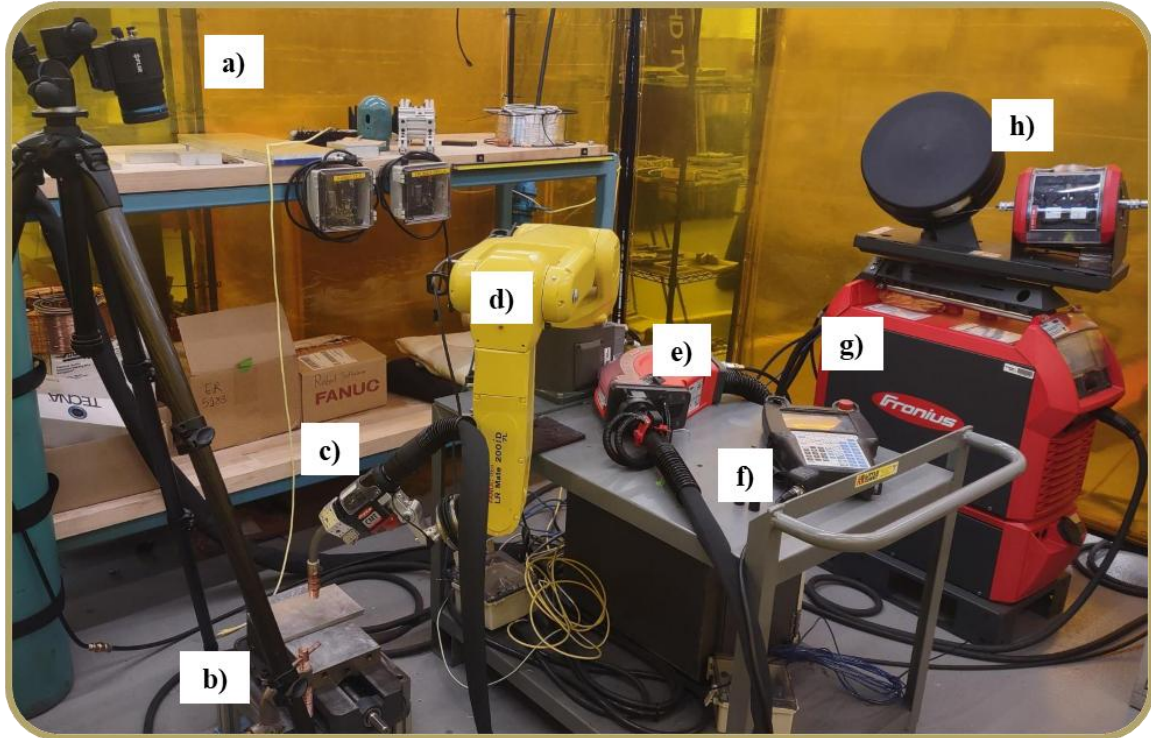


Figure 2: The WAAM-CMT cell used to carry out all experiments: a) FLIR A700; b) build plate, vice, & substrate; c) CMT drive & torch; d) FANUC robot; e) welding split box; f) robot controller; g) Fronius welding system; h) wire spool & feeder

The material evaluated in this work is Al 6061 for the substrate and Al 5183 for the depositing wire, with a diameter of 1.2 mm. The compositions of these materials can be seen in Table 1. Process parameters were selected based on previous work from the primary author of this report [13]. In the first layer, the power input needs to be increased to ensure proper adhesion to the plate, whereas subsequent layers can have a lower heat input to allow for less porosity and smoother bead geometry. Therefore, the average current and voltage found in the first layer from preliminary work was 141 amps & 16.4 volts, whereas subsequent layers have decreased values of 83 amps & 13 volts. Using this information the simulation can accurately change the amount of heat input in the first layer.

Table 1: Chemical Composition of Wire and Substrate [13]

Component	Composition %											
	<i>Al</i>	<i>Cr</i>	<i>Cu</i>	<i>Fe</i>	<i>Mg</i>	<i>Mn</i>	<i>Ni</i>	<i>Si</i>	<i>Ti</i>	<i>Zn</i>	<i>Zr</i>	<i>Other</i>
6061 Substrate	95.1-	0.4-	0.05-	0-	0.8-	0-	0-0.05	0.4-	0-	0-	0-	0.15
	98.2	0.8	0.4	0.7	1.2	0.15		0.8	0.15	0.25	0.25	
5183 Wire	<i>Al</i>	<i>Cr</i>	<i>Cu</i>	<i>Fe</i>	<i>Mg</i>	<i>Mn</i>	<i>Be</i>	<i>Si</i>	<i>Ti</i>	<i>Zn</i>	<i>Other</i>	
	Bal.	0.05-0.25	0.1	0.4	4.3-5.2	0.5-1.0	0.0003	0.4	0.15	0.25	0.15	

A FLIR A700 camera is used to record the temperature distribution throughout the part. The camera records a top-down view, allowing the researchers to obtain the maximum and average temperatures. Aluminum CMT welding has two complications when trying to use an IR camera: 1) the surface of a deposited aluminum bead is very reflective, and 2) there is a lot of aluminum oxide soot released as a byproduct, particularly in the first layer deposition where a higher heat

input is used. As a result, the emissivity (ϵ) value used for the reflective areas of aluminum was experimentally found through comparing the temperature reading of the IR camera to a thermocouple at the same location. A Python script was used to run through each image, where the oxide took on a more yellow color (absorbing more light) while the reflective surfaces remained orange and red. The script was separated into sections based on these colors, and the oxide and aluminum temperatures were calculated separately where an emissive value of 0.95 was used for the oxide. The process can be seen in Figure 3.

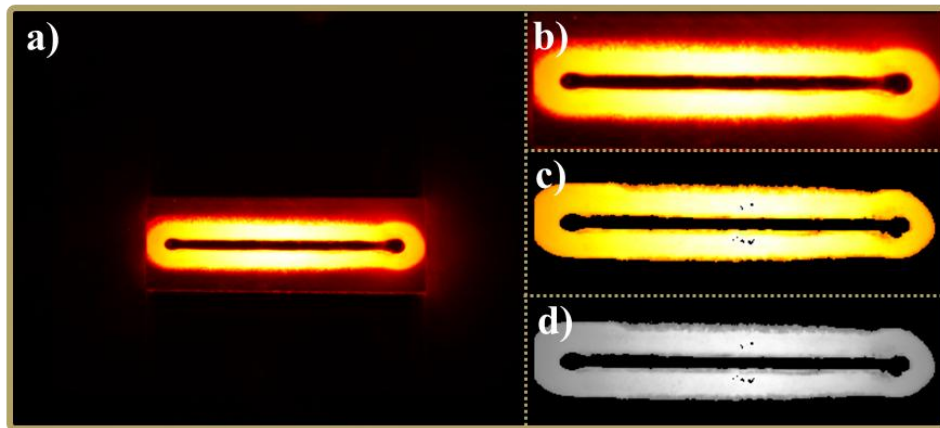


Figure 3: a) raw IR image; b) cropped image; c) yellow filter applied; d) yellow filter averaged and gray-scaled.

Results and Discussion

A. Simulation Visualization

The simulation was carried out entirely in the *matplotlib.pyplot* package using Python's native CSV logger to collect data. Visualization of the plots can be seen in Figure 4 for both the wall and square depositions. Notably, both live temperature recordings and the heat distribution can be seen in both simulations. The color bar was synced such that the maximum temperature visible was the brightest, with room temp (300K) always being the darkest. These images can be compared to the real-world IR images (corrected for the varying emissivity between the aluminum material and oxide soot) seen in Figure 5a & d, where the most recently welded section of the part is the brightest color and the areas of the part which are farther from the heat source are darker. Additionally, with the parameters provided, we're able to obtain smooth surfaces in the part geometry for both the wall and square depositions.

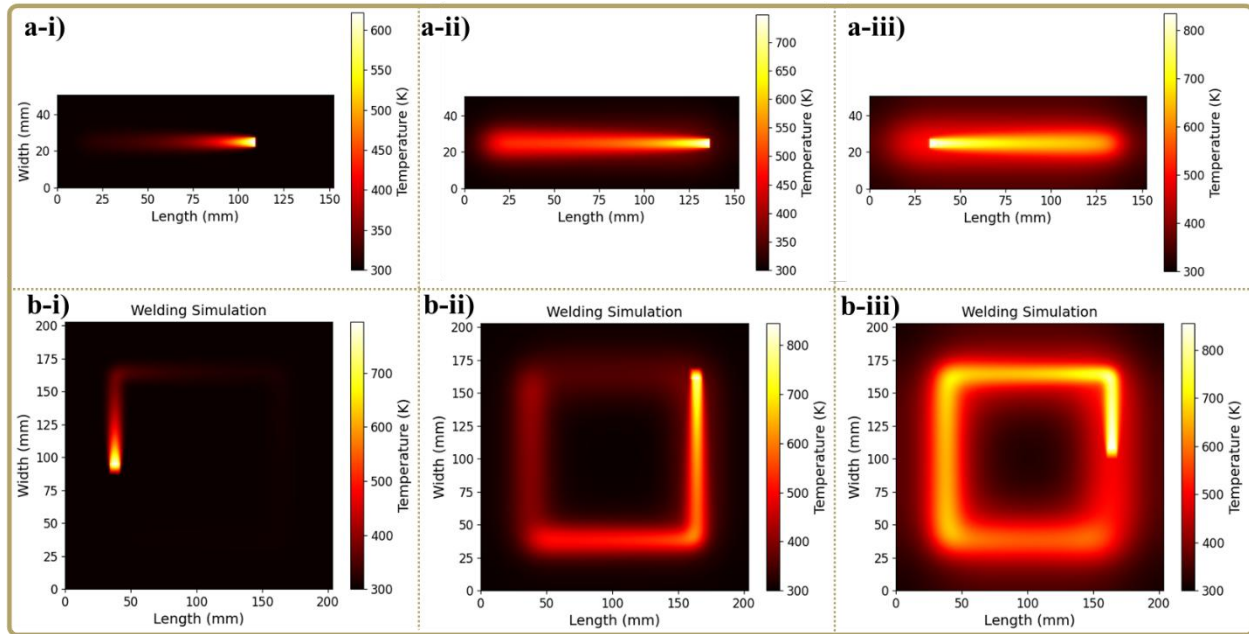


Figure 4: Screenshots of the simulation's plotting software depositing the a) single-bead wall and b) square designs after i) one layer, ii) five layers, and iii) ten layers.

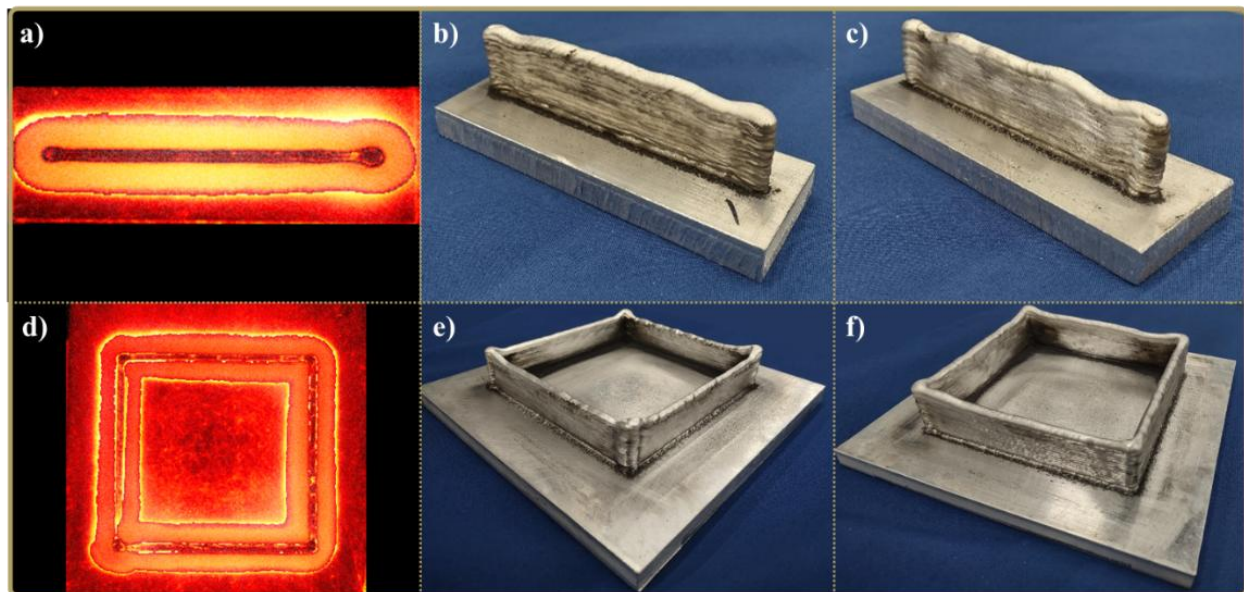


Figure 5: Images of the real-world depositions of the a-c) wall and d-f) square, with a&d) IR images during the dwell.

B. Temperature

The most important part of the present work is its accuracy to real-world experimentation. Temperature comparisons and percentage error results can be seen in Figure 6 for both the wall and the square build. For the first 3-4 layers, while the temperature during welding is accurate, the simulation predicts the cooling curves to go much lower than the IR camera. This can be due to several factors including the limited range of the IR camera and the residual heat from the aluminum oxide reflecting off the beads. However, we can see that the simulation becomes more

accurate as our layer height increases with the percent error between the simulation and experimental data decreasing by 6-15% from layer 1 to layer 10.

This is most noticeable with the exponential fit curve for the wall error, where the relative percent error decreased from 25.96% to 10.45% on average from the first layer to the tenth layer. The mean error was 11.27% overall for the entire build. For the square, the error per layer decreased from 16.18% in the first layer to 10.50% in the tenth layer, with a mean error of 9.73% for both builds. To further quantify these percentages, the mean absolute error (MAE) for each trial of the wall build was 79.91 K & 76.83 K, while the square had 76.31 K and 71.29 K. It is likely that the increase in accuracy over time is due to the material constants that were selected for the simulation being for a hotter material temperature; as we reached the ideal temperature for those constants, the simulation was able to keep up with more accurate values.

Additionally, added stress can be seen on one of the single-bead walls (Figure 5c), which corresponds to experiment “Wall 2” in Figure 6a. It’s possible that the spikes in error are due to this unforeseen deformation: while the experimental Wall 2 was able to maintain a lower temperature compared to Wall 1, it also showed significantly more difference when compared with the simulation. This is likely a result of the simulation being based on an “ideal scenario”. Future work could focus on applying a filter which changes the “reliability” of the model versus sensors in larger control schema or adjusting the accuracy of the model based on in-situ current and voltage data.

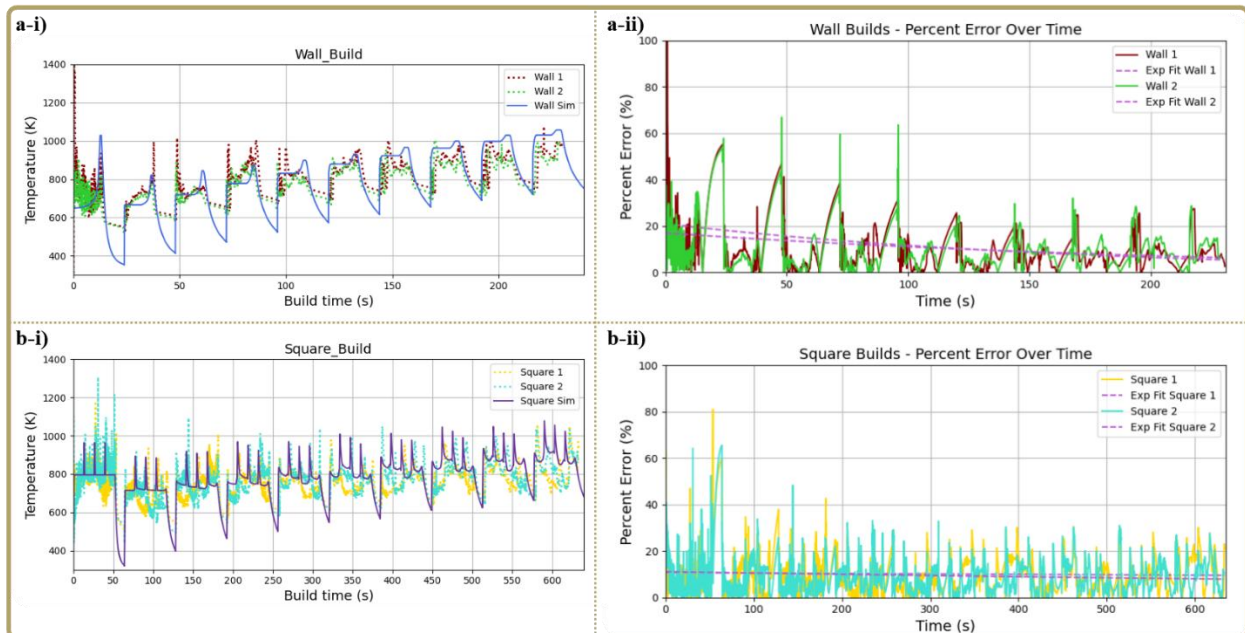


Figure 6: Simulation data (solid line) and experimental data (dotted line) for both the a) wall and b) square designs comparing the i) temperature data and showing the ii) percent error over time.

C. Time Performance

Table 2: Average time performance of the algorithm at different benchmarks, in seconds, at the end of each designated layer.

Design	With plotting & data recording			Only recording data			Without saving or plotting data		
	<i>1</i>	<i>5</i>	<i>10</i>	<i>1</i>	<i>5</i>	<i>10</i>	<i>1</i>	<i>5</i>	<i>10</i>
Wall	10.3958	49.6326	98.3958	2.3303	11.3790	22.7766	2.2687	11.1462	22.2578
Square	100.0949	497.9816	994.3512	62.7888	314.9365	628.8821	59.2862	298.1311	597.8356

The results of the simulation’s time performance can be seen in Table 2. With approximately 1 mm^3 per voxel, without saving the information to a CSV writer or plotting the live results, the simulation was able to create a single wall layer in ~ 2.23 seconds and a single square layer in ~ 57.47 seconds. Notably, the wall had a 3D grid of $152 \times 50 \times 37$ voxels and the square had a grid of $203 \times 203 \times 37$ voxels. This drastic change would indicate an obvious delay in time between the two shapes. However, what keeps the times low compared to other methods is the “active” and “inactive” mask applied in the simulation, making sure that temperatures are only calculated for “active” (deposited) voxels.

When comparing with modern methods, a finite element (FE) deposition technique in Cast3M 2018 (with C++ post-processing to adjust nodes deemed “too hot”) took 3 hours for 8 layers of a 80 mm long bead having 10% error [11], which is the same error from this simulation where the present work is over a larger part. In other external software, it is mentioned that the time take to process a $250 \times 90 \times 6$ mm wall is a challenge that still needs to be overcome [9]. Trying to improve these speeds with a feed-forward neural network (FFNN) model in ANSYS took 20h to create a dataset, 6h to train, and 38s to evaluate a 60mm bead on a $200 \times 80 \times 10$ mm substrate compared to 5h for a single evaluation without training from an FE model [8]. The present work shows how with preliminary depositions (taking less than an hour thanks to WAAM’s efficiency) to collect current and voltage, the simulation not only is able to calculate the resulting temperature on the scale of seconds to minutes but also stays within a 10% error of the experimental data and is able to improve 6-15% over time as part temperatures increase and become constant. As the error is within 10% towards the end of the tenth layer, it can be concluded that accuracy would remain within that percentage or decrease for higher layers.

Conclusion

In this work, a fully Pythonic numerical simulation of thermal distribution in wire-arc additive manufacturing using the cold-metal transfer welding technique with aluminum material is developed and tested. The simulation was run up to ten layers with experimental validation of the simulated data. The following conclusions can be drawn from this work:

- As the layer height increases and the part increases in temperature, the simulation heat predictions increase in accuracy, with an improvement of 6-15%.

- The mean percent error of a ten-layer build, regardless of the part design, is ~10%, which is equivalent to ~70 Kelvin. This is within the range of the present literature on numerical thermal prediction in WAAM.
- The current proposed method is extremely fast, although computation time scales with part size. The algorithm was run on a consumer-available computer, with a run time of ~2.23 seconds per layer for a single-bead wall ($152 \times 50 \times 37$ voxels) and ~57.47 seconds per layer ($25\times$ increase) for a square with a voxel grid $5\times$ larger.
- For these results, experimentally determined variables can be utilized from a prior deposition: the voltage and current of the welder (as recorded) and the wire feed speed (WFS) and traverse speed (TS) of the system (as manually input). From just these variables, as well as the material constants of aluminum, the model can accurately predict the temperatures of aluminum parts.

This algorithm would be suitable for future work investigating an online control scheme that requires thermal prediction of a build during the WAAM process. For work focusing on more detailed models, such as phase changes in the melt pool, further investigation is required into varying material constants, and the effect voxel resolution would have on performance. Another application for future work could be an analysis of the thermal profiles of the part after deposition and how deposition properties (such as tensile strength) are affected. From a less detailed view, in the ideal scenario where any visual of the IR camera is blocked, the simulation can fill in what is supposed to be happening in the lost visual. In an online control schema, the simulated temperature can also be used to inform other pieces of a full system, helping predict more accurate temperatures where the IR camera may be inaccurate.

Acknowledgements

This material is based on research sponsored by Air Force Research Laboratory under agreement number FA8650-20-2-5700. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of Air Force Research Laboratory or the U.S. Government.

References

- [1] Li, Y., Su, C., & Zhu, J. (2022). Comprehensive review of wire arc additive manufacturing: Hardware system, physical process, monitoring, property characterization, application and future prospects. *Results in Engineering*, 13, 100330. <https://doi.org/10.1016/j.rineng.2021.100330>
- [2] Vimal, K. E. K., Naveen Srinivas, M., & Rajak, S. (2021). Wire arc additive manufacturing of aluminium alloys: A review. *Materials Today: Proceedings*, 41, 1139–1145. <https://doi.org/10.1016/j.matpr.2020.09.153>
- [3] Bellamkonda, P. N., Dwivedy, M., & Addanki, R. (2024). Cold metal transfer technology - A review of recent research developments. *Results in Engineering*, 23, 102423. <https://doi.org/10.1016/j.rineng.2024.102423>
- [4] Promoppatum, P., Yao, S.-C., Pistorius, P. C., & Rollett, A. D. (2017). A Comprehensive Comparison of the Analytical and Numerical Prediction of the Thermal History and Solidification Microstructure of Inconel 718 Products Made by Laser Powder-Bed Fusion. *Engineering*, 3(5), 685–694. <https://doi.org/10.1016/J.ENG.2017.05.023>

- [5] Ogino, Y., Asai, S., & Hirata, Y. (2018). Numerical simulation of WAAM process by a GMAW weld pool model. *Welding in the World*, 62(2), 393–401. <https://doi.org/10.1007/s40194-018-0556-z>
- [6] Gupta, D. K., & Mulik, R. S. (2025). Numerical simulation and experimental investigation of temperature distribution during the wire arc additive manufacturing (WAAM) process. *Progress in Additive Manufacturing*, 10(1), 631–645. <https://doi.org/10.1007/s40964-024-00647-4>
- [7] Goldak, J., Chakravarti, A., & Bibby, M. (1984). A new finite element model for welding heat sources. *Metallurgical Transactions B*, 15(2), 299–305. <https://doi.org/10.1007/BF02667333>
- [8] Le, V. T., Bui, M. C., Pham, T. Q. D., Tran, H. S., & Van Tran, X. (2023). Efficient prediction of thermal history in wire and arc additive manufacturing combining machine learning and numerical simulation. *The International Journal of Advanced Manufacturing Technology*, 126(9), 4651–4663. <https://doi.org/10.1007/s00170-023-11473-3>
- [9] Ling, Y., Ni, J., Abdel Wahab, M., Antonissen, J., & Vande Voorde, J. (2021). A Heat Transfer Finite Element Model for Wire-Arc-Additive-Manufacturing Process. In M. Abdel Wahab (Ed.), *Proceedings of the 8th International Conference on Fracture, Fatigue and Wear* (pp. 201–215). Singapore: Springer. https://doi.org/10.1007/978-981-15-9893-7_14
- [10] Hackenhaar, W., Mazzaferro, J. A. E., Montevecchi, F., & Campatelli, G. (2020). An experimental-numerical study of active cooling in wire arc additive manufacturing. *Journal of Manufacturing Processes*, 52, 58–65. <https://doi.org/10.1016/j.jmapro.2020.01.051>
- [11] Chergui, A., Villeneuve, F., Béraud, N., & Vignat, F. (2022). Thermal simulation of wire arc additive manufacturing: a new material deposition and heat input modelling. *International Journal on Interactive Design and Manufacturing (IJIDeM)*, 16(1), 227–237. <https://doi.org/10.1007/s12008-021-00824-7>
- [12] Harris, C. R., Millman, K. J., Walt, S. J. van der, Gommers, R., Virtanen, P., Cournapeau, D., ... Oliphant, T. E. (2020). Array programming with NumPy. *Nature*, 585(7825), 357–362. <https://doi.org/10.1038/s41586-020-2649-2>
- [13] Thien, A., Kelly, K. M., Massey, C. E., & Saldana, C. J. (2023). ON PROCESS STABILITY IN WAAM-CMT OF ALUMINUM ALLOYS. Retrieved from <https://hdl.handle.net/2152/124446>
- [14] DuPONT, J. N., & Marder, A. R. (n.d.). Thermal Efficiency of Arc Welding Processes.
- [15] Li, C., Gu, H., Wang, W., Wang, S., Ren, L., Wang, Z., ... Zhai, Y. (2020). Effect of Heat Input on Formability, Microstructure, and Properties of Al–7Si–0.6Mg Alloys Deposited by CMT-WAAM Process. *Applied Sciences*, 10(1), 70. <https://doi.org/10.3390/app10010070>