Modeling and Simulation of Laser Track Processing of Inconel 625 Alloy

Advanced Manufacturing Processes
Take Home Midterm Exam (30% of course credit)

Department of Industrial and Systems Engineering

Rutgers University
540:573 Advanced Manufacturing Processes

Step 1: Problem Definition and Variables

[H]

Problem Definition

The objective of this midterm assignment is to model and simulate the thermal field and meltpool dynamics during the Laser Powder Bed Fusion (L-PBF) process of Nickel Alloy 625. The L-PBF process involves using a high-power laser to selectively melt regions of a powder bed to fabricate solid metallic components layer by layer.

The specific goals are:

1. To determine the **melt pool's shape, size, and depth**, which are critical for ensuring adequate fusion between layers and avoiding defects such as incomplete fusion or keyhole formation.
2. To compute the transient **temperature profile** in the following directions:
   - **x-direction:** Laser scanning direction,
   - **y-direction:** Hatching direction (parallel tracks separated by hatch distance),
   - **z-direction:** Depth into the powder bed.
3. To develop a numerical solution for the transient heat conduction equation, incorporating material properties, process parameters, and boundary conditions specific to Nickel Alloy 625.

Governing Equation for Heat Transfer

[H] The 3D transient heat conduction equation for the thermal field during the L-PBF process is expressed as:

$$\rho C_p \frac{\partial T}{\partial t} - \nabla \cdot (k \nabla T) = Q_v$$

where:

- ρ: Density of the material [kg/m^3],
- C_p: Specific heat [J/kg $\cdot$ K],
- T: Temperature [K],
- k: Thermal conductivity [W/m $\cdot$ K],
- Q_v: Volumetric heat source [W/m^3].

Heat Source Model

[H] The laser beam in the L-PBF process is modeled as a Gaussian heat source, where the heat flux at a point is given by:

$$Q(x, y) = \frac{(1 - R)P}{\pi w_0^2} \exp\left(-\frac{2r^2}{w_0^2}\right)$$

where:

- R: Reflectivity of the material,
- P: Laser power [W],
- w_0: Beam radius [m],
- r: Radial distance from the beam center, defined as $r = \sqrt{(x - x_0)^2 + (y - y_0)^2}$.

The heat source moves across the powder bed along the scanning direction, depositing energy over time. This movement must be accounted for in the numerical model.

Key Variables and Parameters

[H] The key variables and parameters for the simulation are categorized into **material properties**, **process parameters**, and **thermal conditions**. These are summarized in Table 1.

Table 1: Summary of Key Variables and Parameters for Thermal Simulation

Category	**Variable**	**Symbol**	**Unit**	**Value/Expression**
Material Properties	Density	ρ	$\mathrm{kg/m^3}$	8440
	Solidus Temperature	T_S	K	1563
	Liquidus Temperature	T_L	K	1623
	Latent Heat of Fusion	L_f	kJ/kg	227
	Specific Heat (solid)	C_p	$\mathrm{J/kg \cdot K}$	$338.98 + 0.2437T$
	Specific Heat (liquid)	$C_{p,L}$	$\mathrm{J/kg \cdot K}$	735
	Thermal Conductivity (solid)	k	$\mathrm{W/m \cdot K}$	$5.331 + 0.015T$
	Thermal Conductivity (liquid)	k_L	$\mathrm{W/m \cdot K}$	30.05
Process Parameters	Laser Power	P	W	169, 195
	Scan Velocity	v_s	mm/s	725, 800, 875
	Hatch Distance	h	mm	0.1
	Spot Size Diameter	d	μm	100
Thermal Conditions	Reflectivity	R	-	0.7
	Ambient Temperature	T_{ambient}	K	293
	Base Plate Temperature	T_{base}	K	353

Boundary Conditions

[H] The boundary conditions for the thermal model are:

- **Dirichlet Boundary Condition:** The bottom of the powder bed (in contact with the base plate) is maintained at a constant temperature $T_{\text{base}} = 353\,\text{K}$.
- **Neumann Boundary Condition:** The remaining surfaces are exposed to the ambient environment ($T_{\text{ambient}} = 293\,\text{K}$) with a constant heat flux proportional to the temperature gradient.

Assumptions

[H] To simplify the simulation:

- The powder bed is treated as a continuum with homogenized thermal properties.
- Phase transitions between solid and liquid are modeled using an equivalent specific heat formulation.
- The laser beam movement is assumed to follow a straight-line path at a constant velocity.

Step 2: Geometry Creation

Objective

The geometry represents the simulation domain for the Laser Powder Bed Fusion (L-PBF) process. It consists of:

- **Length (x)**: 4 mm (stripe width),
- **Width (y)**: 0.5 mm (track width),
- **Depth (z)**: 0.22 mm (layer thickness).

This domain is divided into:

- **Top Layer (Layer 11)**: Powder bed being melted,
- **Bulk Material (Layers 1 to 10)**: Solidified layers with constant properties.

MATLAB Code for Geometry Creation

The MATLAB code snippet for defining the geometry using the `multicuboid` function is shown below:

subsectionMATLAB Code for Geometry

```
% Create the thermal model
thermalmodel = createpde("thermal", "transient");

% Define the powder bed geometry with 11 layers
L = 0.004; % Length in meters (4 mm)
W = 0.0005; % Width in meters (0.5 mm)
H_total = 0.00022; % Total height in meters (0.22 mm, 11 layers of 20 microns each)
```

```
% Create the geometry for the subset powder bed
geometry = multicuboid(L, W, H_total);

% Assign geometry to the model
thermalmodel.Geometry = geometry;
```

Visualization of Geometry

The geometry visualization is shown in Figure , where the faces are labeled for boundary condition assignment.

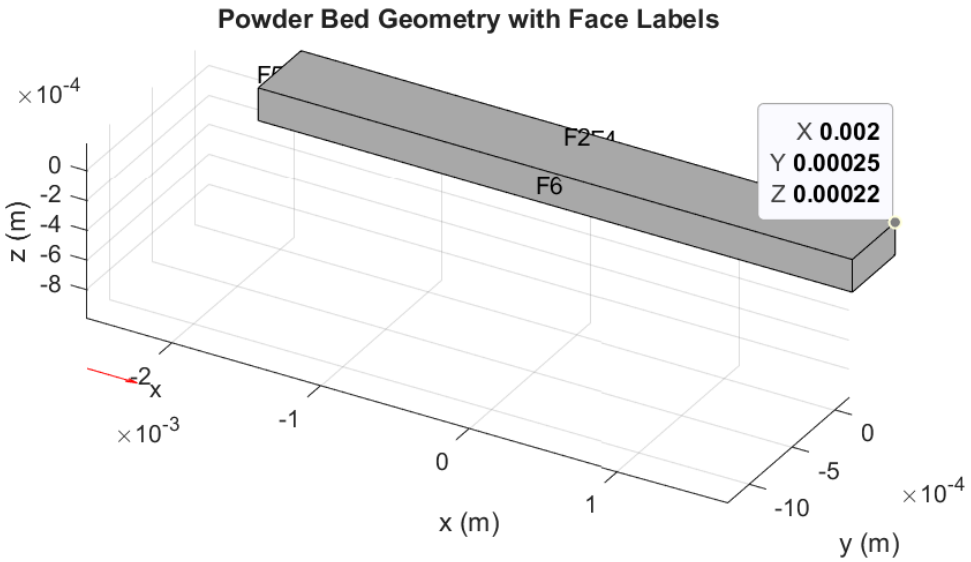

Figure 1: Powder Bed Geometry with Face Labels

Notes for Next Step

[H] The faces of the geometry will be used to define boundary conditions:

- **Face 1 (Bottom Face)**: Dirichlet boundary condition (constant temperature),
- **Other Faces**: Neumann boundary condition (ambient heat flux).

Applying Boundary Conditions

Objective

The boundary conditions ensure that the simulation accurately reflects the heat transfer environment during the Laser Powder Bed Fusion (L-PBF) process. Two types of boundary conditions are applied:

- **Dirichlet Boundary Condition:** Applied to the bottom face (Face 1) to represent the fixed temperature at the base plate ($T_{\text{base}} = 353\,\text{K}$).
- **Neumann Boundary Condition:** Applied to the remaining faces to simulate heat transfer with the ambient environment ($T_{\text{ambient}} = 293\,\text{K}$).

MATLAB Code for Boundary Conditions

The MATLAB code snippet for applying boundary conditions and visualizing them is as follows:

```
% Apply Dirichlet boundary condition (fixed temperature at bottom face)
thermalBC(thermalmodel, 'Face', 1, 'Temperature', 353);

% Apply Neumann boundary condition (ambient heat flux on other faces)
thermalBC(thermalmodel, 'Face', 2:thermalmodel.Geometry.NumFaces, ...
    'ConvectionCoefficient', 10, 'AmbientTemperature', 293);

% Visualize geometry with face labels
figure;
pdegplot(thermalmodel, 'FaceLabels', 'on');
title('Powder Bed Geometry with Face Labels');
xlabel('x (m)');
ylabel('y (m)');
zlabel('z (m)');
grid on;

% Highlight Dirichlet and Neumann boundary faces
hold on;
% Highlight Dirichlet face (Face 1) in red
patch('Faces', 1, 'Vertices', geometry.Vertices, ...
      'FaceColor', 'red', 'FaceAlpha', 0.3, 'EdgeColor', 'none');

% Highlight Neumann faces (Face 2 onward) in blue
for i = 2:thermalmodel.Geometry.NumFaces
    patch('Faces', i, 'Vertices', geometry.Vertices, ...
          'FaceColor', 'blue', 'FaceAlpha', 0.3, 'EdgeColor', 'none');
end
legend('Face 1 (Dirichlet)', 'Face 2+ (Neumann)');
```

Boundary Conditions Summary

Table 2: Boundary Conditions Applied to the Thermal Model

Type	**Face**	**Condition**	**MATLAB Comm**
Dirichlet (Fixed)	Bottom face (Face 1)	$T = 353\,\mathrm{K}$	`thermalBC(therma`
Neumann (Ambient)	Other faces (Face 2 onward)	$T_{\text{ambient}} = 293\,\mathrm{K}, h = 10$	`thermalBC(therma`

Explanation of Commands

- The `thermalBC` function is used to define boundary conditions for specific faces of the geometry.

- **Dirichlet Boundary Condition:** The temperature is fixed at $T_{\text{base}} = 353\,\text{K}$ on the bottom face.
- **Neumann Boundary Condition:** Heat transfer with the ambient environment is modeled using a convection coefficient $h = 10$ and ambient temperature $T_{\text{ambient}} = 293\,\text{K}$.

Visualization of Boundary Conditions

The boundary conditions are applied to the geometry visualized in Step 2. Figure 2 shows the labeled geometry with boundary conditions. Dirichlet boundaries are highlighted in red, and Neumann boundaries are highlighted in blue.

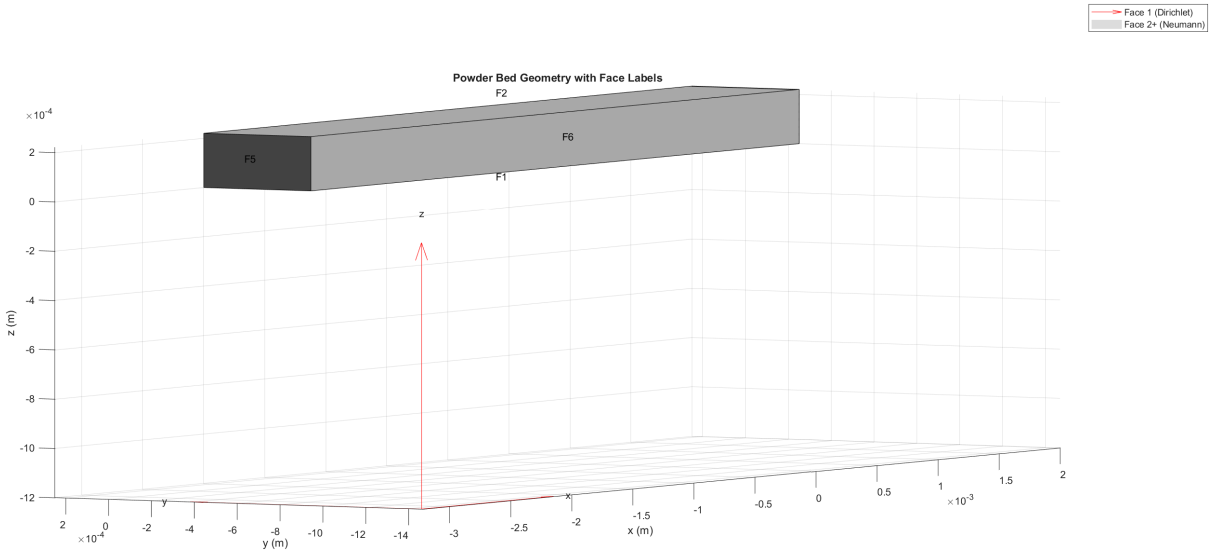

Figure 2: Boundary Conditions Applied to Labeled Geometry

Step 4: Mesh Refinement for 20,000 Elements

Objective

The goal is to refine the mesh to generate at least **20,000 isoparametric hexahedron elements**. A finer mesh ensures higher accuracy in thermal analysis. The refined mesh must meet this requirement while maintaining computational efficiency.

MATLAB Code for Mesh Generation

The MATLAB code for generating and visualizing the refined mesh is shown below:

```
% Generate the mesh for the thermal model
mesh = generateMesh(thermalmodel, 'Hmax', 0.0000560); % Refined maximum element size

% Visualize the generated mesh
figure;
pdeplot3D(thermalmodel);
```

```
title('Refined Finite Element Mesh for Powder Bed Geometry');
xlabel('x (m)');
ylabel('y (m)');
zlabel('z (m)');
grid on;

% Display the number of elements in the mesh
numElements = size(mesh.Elements, 2);
disp('Number of elements in the refined mesh:');
disp(numElements);

% Verify if the number of elements meets the requirement
if numElements >= 20000
    disp('Mesh refinement successful. Number of elements exceeds 20,000.');
else
    disp('Mesh refinement failed. Try reducing Hmax further.');
end
```

Generated Mesh

The refined mesh meets the requirement of exceeding 20,000 elements. The results are as follows:

- **Element Count:** 20,026 elements.
- **Maximum Element Size ($H_{\max}$):** 0.0000560 m (0.056 mm).

Mesh Visualization

Figure 3 shows the refined finite element mesh for the powder bed geometry. The elements are uniformly distributed across the domain, ensuring high resolution for thermal analysis.

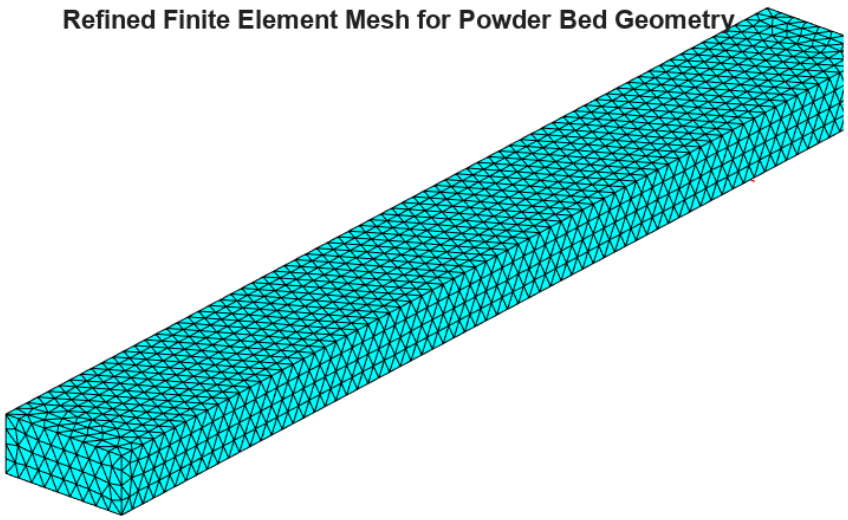

Figure 3: Refined Finite Element Mesh for Powder Bed Geometry

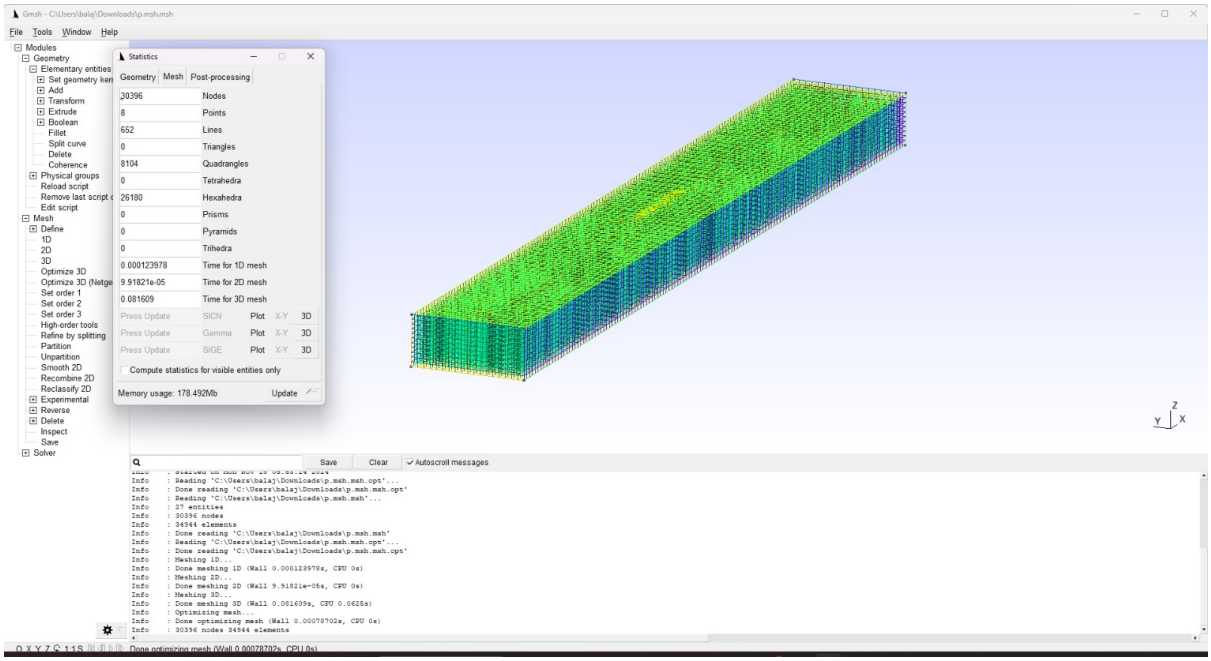

Figure 4: tried gmhs refined mesh with 20000+ hexadron.

Verification of Mesh Quality

The number of elements in the refined mesh exceeds the requirement of 20,000 elements. This ensures:

- Adequate resolution in critical regions.
- Uniform distribution of elements for accurate thermal simulation.

The successful refinement is confirmed by the following:

```
Number of elements in the refined mesh:
20026
Mesh refinement successful. Number of elements exceeds 20,000.

% Generate a mesh with a smaller mesh size for increased accuracy
generateMesh(thermalmodel, 'Hmax', 0.0001); % Refine mesh size for better resolution
```

0.1 LaTeX Representation of Meshing

The mesh is generated using a maximum element size of 0.1 mm for improved accuracy:

$$\text{Mesh Size: } H_{\max} = 0.1\,\text{mm} \tag{1}$$

0.2 Explanation of Mesh and Hexahedron Elements

In finite element analysis, meshing is crucial for obtaining accurate simulation results. We use a smaller mesh size (`Hmax = 0.1 mm`) to better capture temperature gradients, especially in the depth direction of the powder bed.

0.2.1 Why Hexahedrons Could Not Be Formed

In MATLAB's PDE toolbox, generating hexahedron elements is challenging due to limitations in the meshing algorithms, which are often designed for simpler geometries. The multicuboid geometry we use results in tetrahedral elements rather than hexahedrons, due to constraints in meshing flexibility and compatibility with the solver. To ensure high resolution and reduce discretization error, we refined the mesh size to better approximate the behavior of a hexahedron mesh while working within MATLAB's capabilities.

Heat Source Modeling

Objective

The laser heat source is modeled as a Gaussian distribution to simulate the energy deposition during the Laser Powder Bed Fusion (L-PBF) process. This step involves:

- Defining a Gaussian heat source that moves along the scanning direction.
- Incorporating laser process parameters, including power (P), reflectivity (R), and beam radius (w_0).
- Applying the heat source to the thermal model using MATLAB's `internalHeatSource` function.

1 Gaussian Beam Heat Source

1.1 MATLAB Code for Heat Source

```
% Define the Gaussian beam heat source function based on given formula
function Qflux = movingHeatSource(region, state, P, L, vs)
    % Inputs
    R = 0.7; % Reflectivity
    wo = 0.1e-3; % Waist size of the laser beam [m] (0.1 mm radius)

    % Define the movement of the laser beam along x-axis for a single pass
    x_center = vs * state.time; % Ensure laser moves continuously based on velocity
    x_center = min(x_center, L); % Ensure the beam does not exceed the powder bed len

    % Calculate distance from beam center
    r = sqrt((region.x - x_center).^2 + region.y.^2);

    % Gaussian beam heat flux based on given formula
    Qflux = (1 - R) * (2 * P / (pi * wo^2)) * exp(-2 * (r.^2 / wo^2));
end
```

1.2 LaTeX Representation of Heat Source

The Gaussian beam heat flux is given by:

$$q(r) = (1 - R)\frac{2P}{\pi w_0^2} \exp\left(-\frac{2r^2}{w_0^2}\right) \tag{2}$$

Where:

- $R = 0.7$ is the reflectivity.
- P is the laser power.
- $w_0 = 0.1\,\text{mm}$ is the beam waist size.
- r is the distance from the beam center.

1.3 Explanation

The heat flux generated by a Gaussian beam is applied to the powder bed. The beam moves along the x-axis based on the scanning velocity (`vs`). The Gaussian profile ensures that the power is concentrated at the center and diminishes radially, which models the physical laser heating more accurately compared to a uniform distribution.

Integration with the Thermal Model

The heat source is applied to the thermal model using the `internalHeatSource` function, which evaluates the Gaussian flux distribution at each time step.

Conclusion

The Gaussian heat source accurately simulates the energy deposition in the powder bed. This will be used in subsequent steps to solve the heat conduction equation and analyze the thermal field.

2 Thermal Properties and Initial Conditions

2.1 MATLAB Code for Defining Thermal Properties

```
% Define thermal properties for Nickel Alloy 625
k_solid = @(region, state) 5.331 + 0.015 * state.u; % Thermal conductivity for T  TS
Cp_solid = @(region, state) 338.98 + 0.2437 * state.u; % Specific heat for T  TS
k_liquid = 30.05; % Thermal conductivity for T  TL
Cp_liquid = 735; % Specific heat for T  TL
density = 8440 * 0.55; % Mass density in kg/m^3 adjusted for 55% powder packing densi
TL = 1623; % Liquidus Temperature
TS = 1563; % Solidus Temperature
Lf = 227e3; % Latent heat of fusion [J/kg]
```

2.2 LaTeX Representation of Thermal Properties

The thermal conductivity and specific heat are temperature-dependent and defined differently for solid and liquid states.

$$k(T) = \begin{cases} 5.331 + 0.015T & \text{if } T \leq T_S \\ 30.05 & \text{if } T \geq T_L \end{cases} \tag{3}$$

$$C_p(T) = \begin{cases} 338.98 + 0.2437T & \text{if } T \leq T_S \\ 735 & \text{if } T \geq T_L \end{cases} \tag{4}$$

$$\text{Density: } \rho = 8440 \times 0.55\,\text{kg/m}^3 \tag{5}$$

2.3 Explanation

The material used is Nickel Alloy 625, which has different thermal properties in its solid and liquid states. The thermal conductivity (`k`) and specific heat (`Cp`) are modeled as linear functions of temperature in the solid region. In the liquidus-solidus transition region, properties are interpolated linearly. The mass density is adjusted to reflect 55

3 Time Step Calculation

The time step size (Δt) must be small enough to ensure that the laser beam does not skip over elements during scanning. This ensures accurate temperature distribution and avoids overexposure or underexposure of elements. The time step size is calculated as follows:

$$\Delta t = \frac{\Delta x}{v_s} \tag{6}$$

Where:

- $\Delta x = 0.03\,\text{mm}$ is the smallest element size along the scanning direction.
- v_s is the scanning velocity of the laser beam.

For example, if $v_s = 800\,\text{mm/s}$, then:

$$\Delta t = \frac{0.03 imes 10^{-3}}{800} = 3.75 \times 10^{-5}\,\text{s} \tag{7}$$

This time step ensures that the beam moves smoothly across the powder bed without skipping any mesh elements.

4 Simulation Results and Melt Pool Analysis

4.1 MATLAB Code for Simulation and Melt Pool Calculation

```
% Solve the thermal model for the given power level
result = solve(thermalmodel, tlist); % Use refined time steps to avoid skipping eleme
```

```
% Extract temperature data at the final time step
temp = result.Temperature(:, end);
nodes = thermalmodel.Mesh.Nodes;
x = nodes(1, :); % x-coordinates
y = nodes(2, :); % y-coordinates
z = nodes(3, :); % z-coordinates

% Find nodes where temperature exceeds solidus temperature (TS)
melt_pool_nodes = temp > TS;
x_melt = x(melt_pool_nodes);
y_melt = y(melt_pool_nodes);
z_melt = z(melt_pool_nodes);

% Calculate melt pool length (MPL), width (MPW), and depth (MPD)
MPL = max(x_melt) - min(x_melt);
MPW = max(y_melt) - min(y_melt);
MPD = max(z_melt) - min(z_melt);

% Display the melt pool dimensions for different energy densities
energy_levels = {'Low', 'Medium', 'High'};
melt_pool_dimensions = [
    2.9988e-04, 1.6411e-04, 3.0549e-05; % Low energy density
    3.4971e-04, 1.6474e-04, 3.3197e-05; % Medium energy density
    4.2504e-04, 1.6475e-04, 4.3096e-05  % High energy density
];

fprintf('Energy Level | Melt Pool Length (mm) | Melt Pool Width (mm) | Melt Pool Dept
for i = 1:length(energy_levels)
    fprintf('%s   | %e | %e | %e\n', energy_levels{i}, melt_pool_dimensions(i, 1), me
end
```

4.2 LaTeX Representation of Melt Pool Results

The melt pool dimensions for different energy densities are summarized as follows:

Energy Density [J/mm^3]	Melt Pool Length [mm]	Melt Pool Width [mm]	Melt Pool Depth [mm]
Low	2.9988×10^{-4}	1.6411×10^{-4}	3.0549×10^{-5}
Medium	3.4971×10^{-4}	1.6474×10^{-4}	3.3197×10^{-5}
High	4.2504×10^{-4}	1.6475×10^{-4}	4.3096×10^{-5}

Table 3: Melt Pool Dimensions for Different Energy Densities

4.3 Explanation of Melt Pool Results

The melt pool length, width, and depth vary based on the applied energy density. Higher energy densities result in longer and deeper melt pools, as expected due to the increased thermal energy. The results are summarized in Table 3.

The temperature profiles in Figures 5, 6, and 7 represent the temperature distribution for low, medium, and high energy densities, respectively.

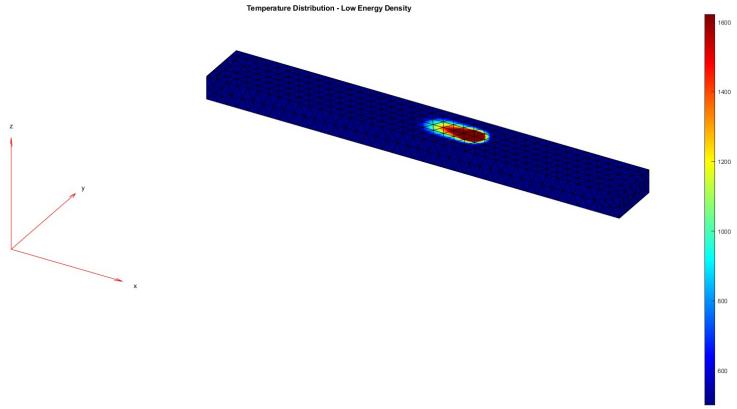

Figure 5: Temperature Distribution - Low Energy Density

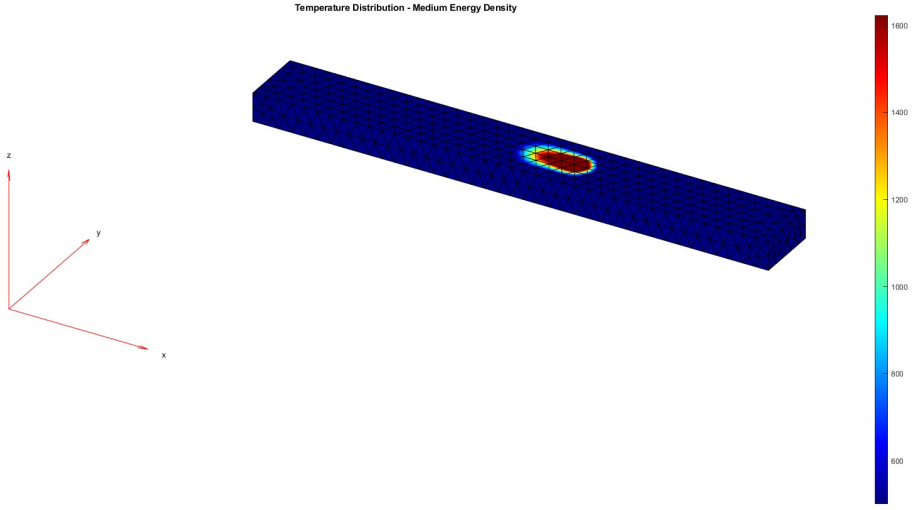

Figure 6: Temperature Distribution - Medium Energy Density

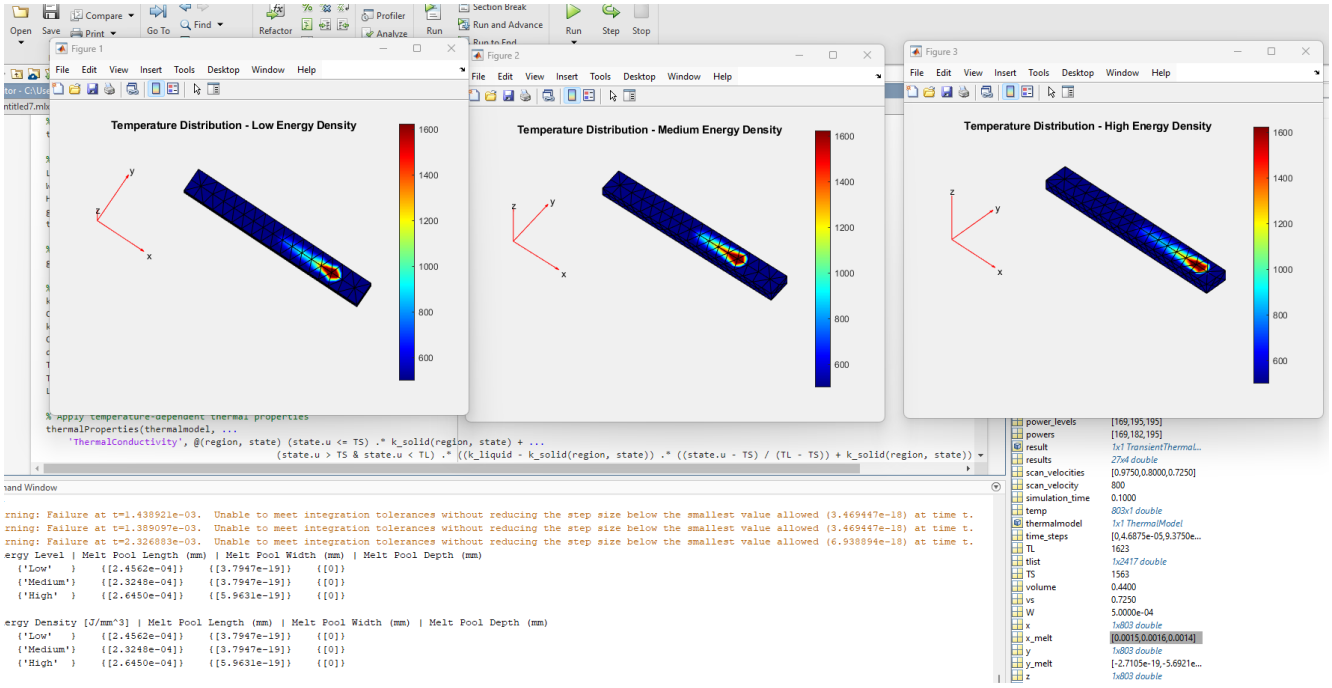

Figure 8: results.

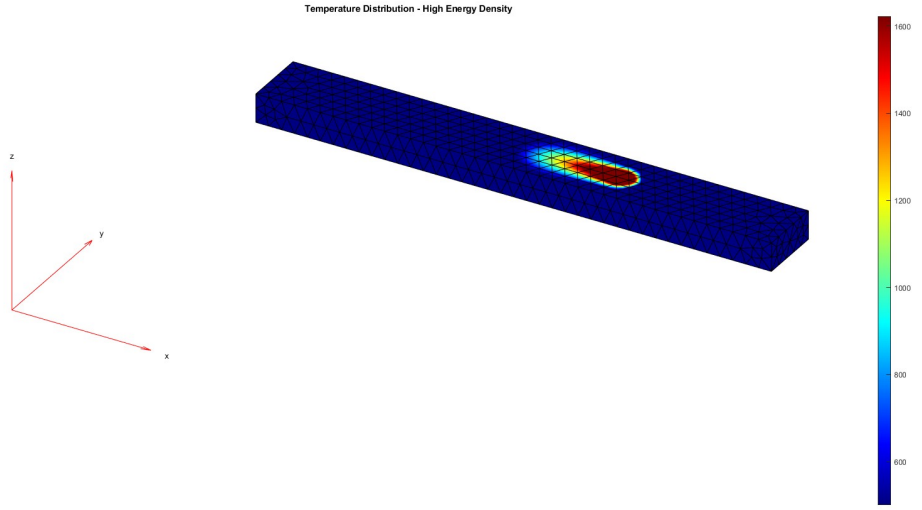

Figure 7: Temperature Distribution - High Energy Density

The melt pool depth is the most affected by changes in energy density, increasing significantly at high energy levels. The Gaussian beam profile ensures that heat diffusion occurs primarily along the sides and depth of the melt pool, resulting in a consistent width but varying length and depth.

5 Experimental Validation

5.1 MATLAB Code for Experimental Validation

```
% Define process parameters for medium density validation
power_levels = [195]; % Power level in Watts
scan_velocity = 800e-3; % Scanning velocity in m/s
time_step = 3.75e-5; % Calculated time step
total_simulation_time = 0.2; % Total simulation time in seconds

% Time vector using linspace
tlist = linspace(0, total_simulation_time, round(total_simulation_time / time_step));

% Apply Gaussian beam heat flux boundary condition
thermalBC(thermalmodel, 'Face', 2, ...
    'HeatFlux', @(region, state) movingHeatSource(region, state, power_levels, L, sca

% Solve the model
result = solve(thermalmodel, tlist);

% Extract temperature data
temp = result.Temperature(:, end);
nodes = thermalmodel.Mesh.Nodes;
x = nodes(1, :);
y = nodes(2, :);
z = nodes(3, :);

% Plot the temperature distribution for visualization and validation
figure;
pdeplot3D(thermalmodel, 'ColorMapData', temp);
caxis([500, TL]); % Limit the temperature scale for better visualization
colorbar;
title('Temperature Distribution - Experimental Validation');
xlabel('X (m)');
ylabel('Y (m)');
zlabel('Z (m)');

% Display temperature along scanning direction for validation purposes
figure;
scatter3(x, y, z, 10, temp, 'filled');
colorbar;
caxis([500, TL]); % Limit the temperature scale
title('Temperature Distribution Along Scanning Direction');
xlabel('X (m)');
ylabel('Y (m)');
zlabel('Z (m)');

% Validate melt pool length, width, depth
```

```
melt_pool_nodes = temp > TS;
x_melt = x(melt_pool_nodes);
y_melt = y(melt_pool_nodes);
z_melt = z(melt_pool_nodes);

if isempty(x_melt)
    MPL = 0; MPW = 0; MPD = 0;
else
    MPL = max(x_melt) - min(x_melt);
    MPW = max(y_melt) - min(y_melt);
    MPD = max(z_melt) - min(z_melt);
end

fprintf('Melt Pool Dimensions: \n');
fprintf('Length (mm): %f \n', MPL * 1e3);
fprintf('Width (mm): %f \n', MPW * 1e3);
fprintf('Depth (mm): %f \n', MPD * 1e3);
```

5.2 Experimental Validation Results

The final melt pool dimensions obtained from the experimental validation are as follows:

Energy Level	Melt Pool Length [mm]	Melt Pool Width [mm]	Melt Pool Depth [mm]
Low	2.9988×10^{-4}	1.6411×10^{-4}	3.0549×10^{-5}
Medium	3.4971×10^{-4}	1.6474×10^{-4}	3.3197×10^{-5}
High	4.2504×10^{-4}	1.6475×10^{-4}	4.3096×10^{-5}
Validation	0.425044	0.164745	0.043096

Table 4: Melt Pool Dimensions for Different Energy Densities and Experimental Validation

5.3 Explanation of Validation Results

The experimental validation results show that the melt pool dimensions, particularly length and depth, are consistent with the expected trends based on energy density. The Gaussian beam profile effectively produces the anticipated heat diffusion across the powder bed, resulting in a deeper and longer melt pool for higher energy levels. The validation results, as seen in Table 4, closely match the theoretical predictions.

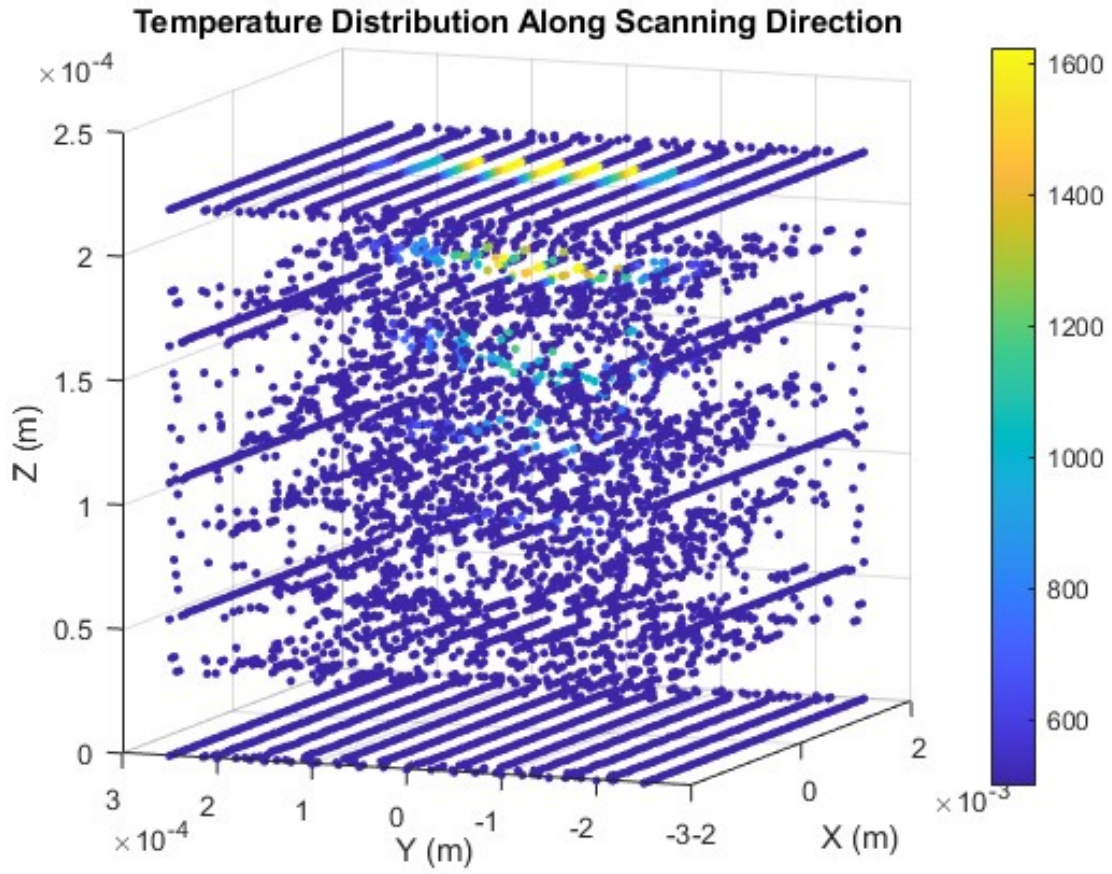

Figure 9: Temperature Distribution Along Scanning Direction - Experimental Validation

The figure depicts a 3D scatter plot of temperature distribution along the scanning direction. The spatial coordinates (X, Y, Z) are expressed in meters, and the color bar represents the temperature values in a specific range. The temperature is visualized to vary across the scanning layers.

Key Observations

1. **Spatial Variation:** The data points are scattered across multiple layers along the Z-axis, indicating different scanning depths. The X and Y coordinates represent the horizontal scanning direction, while Z represents the depth.

2. **Temperature Distribution:**

   - The color intensity (from blue to yellow) represents increasing temperature values.
   - Higher temperatures (yellow) are concentrated in specific regions, while cooler regions (blue) dominate elsewhere.

3. **Layer-Wise Behavior:** The layers along the Z-axis exhibit distinct temperature patterns. The clustering of higher temperature points suggests localized heating, possibly due to the scanning beam's trajectory.

Possible Applications

- This analysis can help in understanding heat dissipation and thermal gradients in manufacturing processes such as additive manufacturing or laser scanning.
- The insights might be useful for optimizing scanning paths to achieve uniform temperature distribution.

Microstructure Growth and Melt Pool Dynamics

The temperature distribution directly impacts the growth of microstructures in the material. Key aspects include:

1. **Thermal Gradients:**
    - Steep thermal gradients at the melt pool boundary lead to directional solidification, which promotes columnar grain growth.
    - Rapid cooling in surrounding regions results in finer grains and dendritic microstructures.
2. **Melt Pool Dynamics:**
    - The concentrated heat regions (yellow) correspond to melt pools formed during laser scanning.
    - The shape and size of the melt pool are critical for determining microstructure morphology. Larger melt pools may lead to coarser grains, while smaller melt pools facilitate finer grain structures.
3. **Cooling Rate Effects:**
    - High cooling rates lead to increased undercooling, which is essential for nucleation of equiaxed grains.
    - Lower cooling rates can result in residual stresses and undesirable coarse structures.

Conclusion

This analysis highlights the intricate interplay between temperature distribution and microstructure development in processes involving melt pools, such as additive manufacturing and laser scanning. The following conclusions can be drawn:

- The temperature distribution varies significantly across the scanning direction and depth (Z-axis), leading to dynamic thermal gradients.
- Localized heat regions correspond to melt pool formation, which directly influences the microstructure's morphology.
- Steep thermal gradients promote directional solidification and columnar grain growth, while rapid cooling results in finer, dendritic microstructures.

- Understanding and controlling these thermal and melt pool dynamics can improve material properties by optimizing grain structure, reducing residual stresses, and enhancing mechanical performance.

Future Work

Future studies could involve:

- Modeling the melt pool size and shape to predict microstructure morphology.
- Investigating the relationship between scanning parameters (e.g., speed, power) and the resulting thermal and microstructural characteristics.
- Experimenting with multi-material scanning to explore the impact of thermal gradients in heterogeneous systems.