

# Handbuch

Benjamin Schnabel

5. November 2018

## Inhaltsverzeichnis

<b>1</b>	<b>Verzeichnisstruktur</b>	<b>2</b>
<b>2</b>	<b>Schnellstart</b>	<b>2</b>
<b>3</b>	<b>main.m</b>	<b>3</b>
3.1	Wärmestromdichte grafisch darstellen . . . . .	3
3.2	Drehung des Voxel-Modells . . . . .	3
3.3	Voxel-Modell grafisch darstellen . . . . .	4
3.4	Video der Simulation erstellen . . . . .	4
3.5	Eigenwerte der Temperaturmatrix bestimmen . . . . .	4
3.6	Excel-Datei als Latex Tabelle exportieren . . . . .	5
<b>4</b>	<b>Laser model</b>	<b>5</b>
4.1	getLaserParameter . . . . .	5
<b>5</b>	<b>Latex</b>	<b>6</b>
<b>6</b>	<b>Material model</b>	<b>6</b>
<b>7</b>	<b>Plots</b>	<b>6</b>
<b>8</b>	<b>Reports</b>	<b>6</b>
<b>9</b>	<b>STL file</b>	<b>6</b>
<b>10</b>	<b>Thermal model</b>	<b>6</b>
10.1	getThermalParameter . . . . .	6
<b>11</b>	<b>Useful functions</b>	<b>7</b>
11.1	plotSimulation . . . . .	7
<b>12</b>	<b>Ablauf Simulation</b>	<b>8</b>

# 1 Verzeichnisstruktur

In Abbildung 1 ist eine Übersicht über die Verzeichnisstruktur des Modells zur Simulation der thermischen Eigenschaften beim Selektiven Lasersintern dargestellt. Im Folgenden werden nur die für die Durchführung der Simulation wichtigsten Funktionen erklärt.

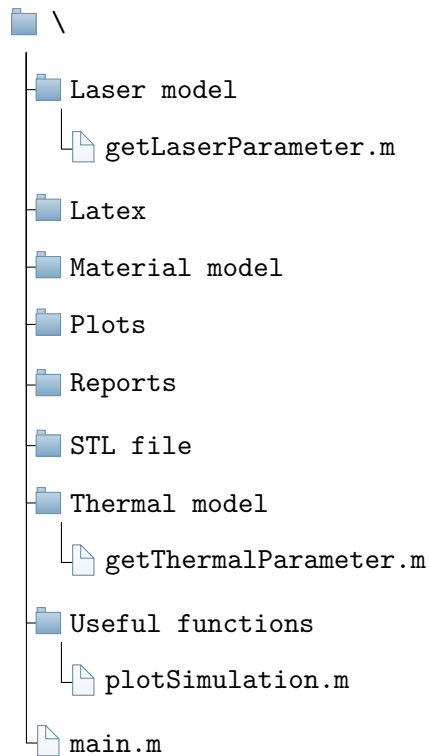
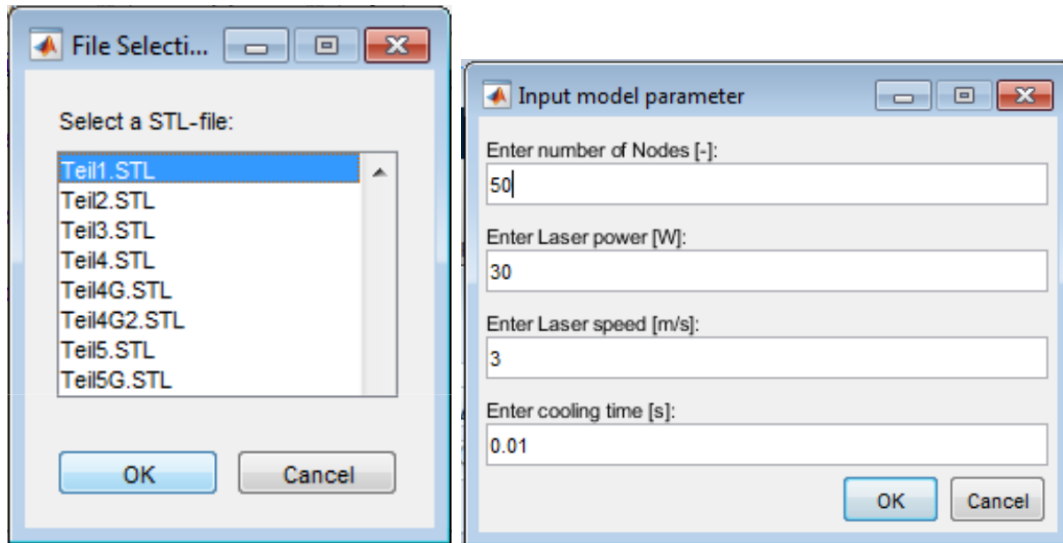


Abbildung 1: Verzeichnisstruktur

## 2 Schnellstart

Für den Schnellstart der Simulation wird die STL-Datei im Ordner *STL file* abgelegt. Im Anschluss wird die Matlab-Datei *main.m* geöffnet und über den Befehl *Run main* bzw. über die Taste *F5* gestartet. In Abbildung 2a ist der Auswahldialog der STL-Datei dargestellt. In diesem wird einfach die zuvor abgelegte Datei ausgewählt. Im nächsten Schritt werden noch die wichtigsten Simulationsparameter abgefragt. Dies ist in Abbildung 2b dargestellt.



(a) STL-Datei Auswahldialog

(b) Parametereingabe

Abbildung 2: Eingabefenster

### 3 main.m

Main-Funktion zur Ausführung der Simulation. Im Folgenden werden nur die Module (Funktionen) beschrieben, die eine Auswahlmöglichkeit bieten.

#### 3.1 Wärmestromdichte grafisch darstellen

Erstellte Datei wird im Ordner Plots abgelegt.

```
% Display the heat flux intensity in a figure (true or false)
displayHeatFluxIntensityFigure = 'false';
switch displayHeatFluxIntensityFigure
    case 'true'
        % exportAsPDF (true or false)
        plotHeatFluxIntensity(0.02, 'false', P, r_0, r_w);
    case 'false'
    otherwise
        disp('Error: displayHeatFluxIntensityFigure')
end
```

#### 3.2 Drehung des Voxel-Modells

```
% Rotate Voxel- Matrix along z axis
rotateZAxis = '0';

switch rotateZAxis
```

```

case '0'
    voxelModel = rot90(voxelModel,0);
case '90'
    voxelModel = rot90(voxelModel,1);
case '180'
    voxelModel = rot90(voxelModel,3);
case '270'
    voxelModel = rot90(voxelModel,2);
otherwise
    disp('Error: rotateZAxis')
end

```

### 3.3 Voxel-Modell grafisch darstellen

Erstellte Datei wird im Ordner Plots abgelegt.

```

% Display the voxel model in a figure (true or false)
displayVoxelModelFigure = 'false';
switch displayVoxelModelFigure
case 'true'
    % exportAsPDF (true or false)
    plotVoxelModel(voxelModel, n, 'false');
case 'false'
otherwise
    disp('Error: displayVoxelModelFigure')
end

```

### 3.4 Video der Simulation erstellen

```

% Create a movie plot (true or false)
createVideoPlot = 'false';
switch createVideoPlot
case 'true'
    plotSimulationForVideoExport(TemperatureArray, ...
        n, n_new, x, y, z, xSliced, ySliced, region, ...
        stlFileNameShortened, fileDate);
case 'false'
otherwise
    disp('Error: createVideoPlot')
end

```

### 3.5 Eigenwerte der Temperaturmatrix bestimmen

```

% Create a plot of the eigenvalues (true or false)
createEigenvaluesPlot = 'true';
switch createEigenvaluesPlot

```

```

case 'true'
    % exportAsPDF (true or false)
    fig = plotEigenvalues(TemperatureArray, 'true', ...
        stlFileNameShortened, fileDate);
case 'false'
otherwise
    disp('Error: createEigenvaluesPlot')
end

```

### 3.6 Excel-Datei als Latex Tabelle exportieren

```

% Export excel-file as Latex table
createLatexTable = 'false';
switch createLatexTable
case 'true'
    exportAsLatexTable(fileName)
case 'false'
otherwise
    disp('Error: createLatexTable')
end

```

## 4 Laser model

In diesem Ordner sind alle Funktionen zur Beschreibung der Wärmestromdichte eines GauSS-Laserstrahls für eine TEM<sub>00</sub>-Mode abgelegt.

### 4.1 getLaserParameter

In dieser Funktionen können die Parameter zur Beschreibung des Lasers vordefiniert werden. Alle in Abbildung 2b definierten Parameter werden aus dieser Funktion ausgelesen.

```

function parameter = getLaserParameter()
    % Stefan-Boltzmann constant [W/(m^2 K^4)]
    parameter.stefanBoltzmanConstant = 5.670367 * 10^-8;
    % Focal length [mm]
    parameter.focalLength = 120.0;
    % Wave length [ym]
    parameter.waveLength = 10.63;
    % Laser power [W]
    parameter.laserPower = 30.0;
    % Raw beam radius at focusing lens [mm]
    parameter.rawBeamRadius = 16.0;
    % Distance to focal point [mm]
    parameter.distanceFocalPoint = 10.0;
    % Laser Speed [m/s]

```

```
parameter.laserSpeed = 3.0;  
end
```

## 5 Latex

## 6 Material model

In diesem Ordner sind alle Funktionen zur Beschreibung der Eigenschaften des Werkstoffes abgelegt. Dies beinhaltet die spezifische Wärmekapazität  $c_p$ , Wärmeleitfähigkeit  $\kappa$ , Dichte  $\rho$  und Temperaturleitfähigkeit  $a$ .

## 7 Plots

## 8 Reports

Nach jeder Simulation werden die erstellten Berichte in diesem Ordner automatisch abgelegt.

## 9 STL file

Die erstellen STL-Dateien werden in diesem Ordner abgelegt.

## 10 Thermal model

In diesem Ordner ist die Funktion zur Berechnung der Wärmeübertragung, die Implementierung des Explizites Euler-Verfahrens und die Funktion zum Vordefinieren der thermischen Parameter abgelegt.

### 10.1 getThermalParameter

In dieser Funktionen können die Parameter zur Beschreibung des thermischen Eigenschaften vordefiniert werden. Die Anzahl der Gitterpunkte und die Abkühlungszeit pro Schicht (siehe Abbildung 2b) werden aus dieser Funktion ausgelesen.

```
function parameter = getThermalParameter()  
    % Number of nodes  
    parameter.numberOfNodes = 25;  
    % Chamber temperature [K]  
    parameter.chamberTemperature = 273.15 + 175.0;  
    % Powderbed temperature [K]  
    parameter.powderbedTemperature = 273.15 + 163.0;  
  
    % Boltzmann constant [J/K]
```

```

parameter.boltzmannConstant = 1.3806504 * 10^-23;
% Speed of light [m/s]
parameter.speedOfLight = 299792458;
% Planck constant [Js]
parameter.planckConstant = 6.62606896 * 10^-34;

% Cooling time [s]
parameter.coolingtime = 0.01;
end

```

## 11 Useful functions

### 11.1 plotSimulation

Die Transparenz kann über den Befehl  $\alpha(h, 0.2)$  angepasst werden, wobei der Wert nach dem  $h$  die Transparenz bestimmt.

```

function [fig] = plotSimulation(fin, x, y, z, xSliced, ...)

    fig = figure(1);
    fin(fin == 0) = NaN;
    h = slice(x,y,z,fin-273.15,xSliced,ySliced,Inf);
    axis(region);
    set(h,'EdgeColor','none')
    set(gca,'xtick',[])
    set(gca,'ytick',[])
    set(gca,'ztick',[])
    xlabel('x')
    ylabel('y')
    zlabel('z')
    cb = colorbar;
    ylabel(cb, '\{\}^{\circ}C', 'Interpreter','tex')
    alpha(h,0.2);
    set(gca,'fontsize',12)

    stlName = stlName(1:end-4);
    fileName = [stlName, '-Layer', num2str(i)];
    orient(fig,'landscape')
    print(fig,'-bestfit',fileName,'-dpdf','-r0')
end

```

## 12 Ablauf Simulation

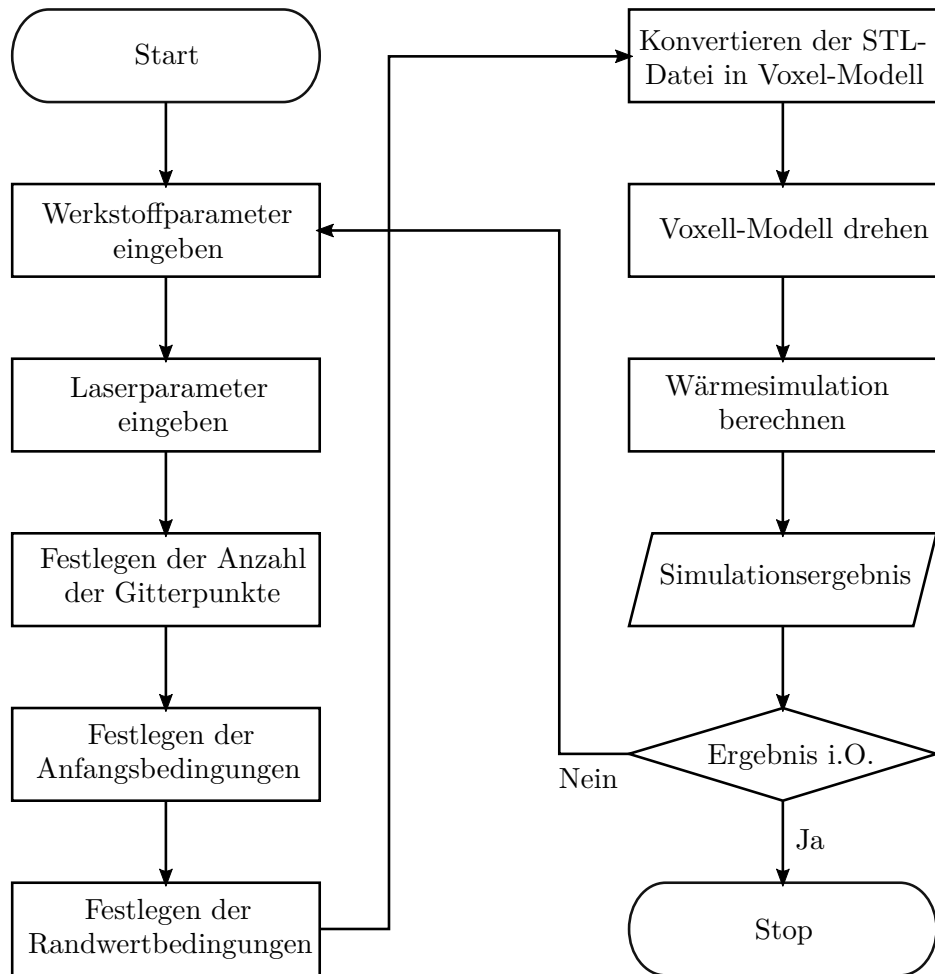


Abbildung 3: Programmablaufplan zur Durchführung der Simulation