Manual

Benjamin Schnabel

26. Oktober 2018

Inhaltsverzeichnis

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.

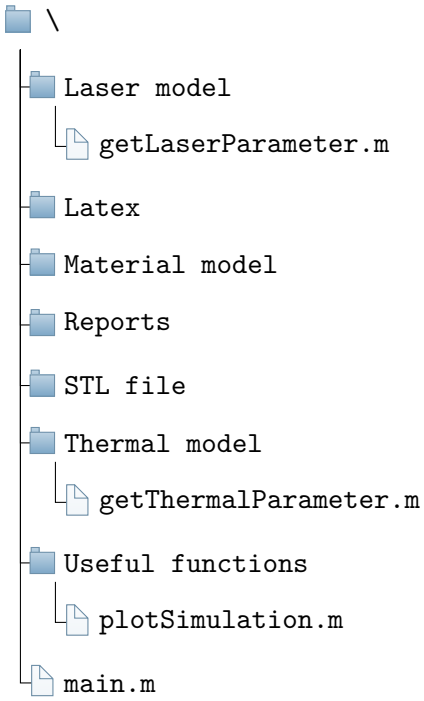


Abbildung 1: Verzeichnisstruktur

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.

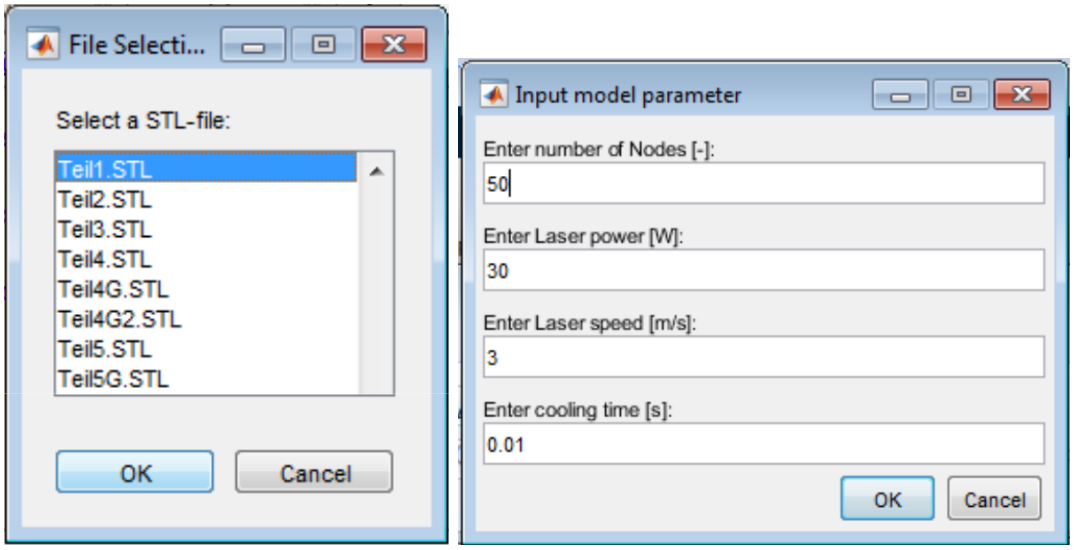
3 Laser model

In diesem Ordner sind alle Funktionen zur Beschreibung der Wärmestromdichte eines GauSS-Laserstrahls für eine TEM_{00}-Mode abgelegt.

3.1 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)]
```



(a) STL-Datei Auswahldialog (b) Parametereingabe

Abbildung 2: Blabka

```
    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
```

4 Material model

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

5 Reports

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

6 STL file

Die erstellen STL-Dateien werden in diesem Ordner abgelegt.

7 Thermal model

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

7.1 getThermalParameter

In dieser Funktionen können die Parameter zur Beschreibung des thermischen Eingenschaften 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 = 50;
    % 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
```

8 Useful functions

8.1 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,ySliced, ...)
    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, 'řC')
    alpha(h,0.2);
    stlName = stlName(1:end-4);
    fileName = [stlName, '-Layer', num2str(i)];
    orient(fig,'landscape')
    print(fig,'-bestfit',fileName,'-dpdf','-r0')
```

9 main.m

Main-Funktion zur Ausführung der Simulation. Im Folgenden wird nur die Drehung des Voxel-Modells entlang der z-Achse erklärt.

Über die Zuweisung eines Wertes zur Variablen *rotateZAxis* kann das Voxel-Modell entlang seiner z-Achse um 90, 180 und 270 Grad gedreht werden.

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

switch rotateZAxis
    case '0'
        voxelMatrix = rot90(voxelMatrix,0);
    case '90'
        voxelMatrix = rot90(voxelMatrix,1);
    case '180'
        voxelMatrix = rot90(voxelMatrix,3);
    case '270'
        voxelMatrix = rot90(voxelMatrix,2);
    otherwise
        disp('')
end

% Show Voxel plot (show or hide)
displaySTLPlot = 'hide';

switch displaySTLPlot
```

```
    case 'show'
        disp('')
        fig = figure();
        [vol_handle] = VoxelPlotter(voxelMatrix,1);
        alpha(0.8)
        az = -37;
        el = 30;
        view(az, el);
        set(gca,'xlim',[0 nx+1], 'ylim',[0 ny+1], 'zlim',[0 nz+1])
        xlabel('Nodes in x')
        ylabel('Nodes in y')
        zlabel('Nodes in z')

        exportSTLPlot = 'false';

        switch exportSTLPlot
            case 'true'
                orient(fig,'landscape')
                print(fig,'-bestfit','displaySTLPlot','-dpdf','-r0')
            case 'false'
                disp('')
        end
    case 'hide'
        disp('')
    otherwise
        disp('')
end
```

10 Ablauf Simulation

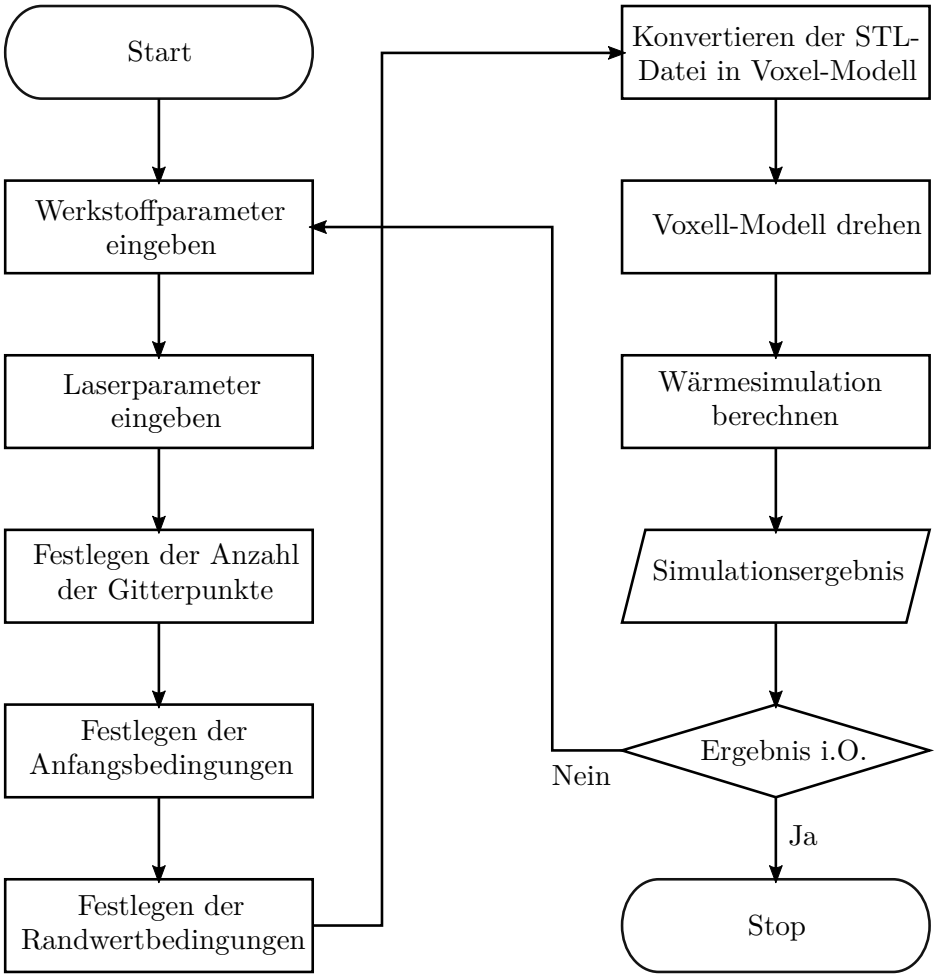


Abbildung 3: Programmablaufplan zur Durchführung der Simulation