TECHNISCHE UNIVERSITÄT WIEN

IFT

Master thesis

Multiphysical numerical simulation of laser based additive manufacturing processes: Direct Energy Deposition

executed to obtain the academic degree of

Master of Science

under supervision of

Univ. Prof. Dipl.-Phys. Dr.-Ing. Andreas Otto

(E311 Institute of Production Engineering and Photonic Technologies)

Proj. -Ass. MSc Rodrigo Gómez Vázquez

(E311 Institute of Production Engineering and Photonic Technologies)

submitted to the Vienna University of Technology

Faculty of Mechanical and Industrial Engineering

by

Michele Buttazzoni

01225412

Vienna, January 2020

TECHNISCHE UNIVERSITÄT WIEN
VIENNA UNIVERSITY OF TECHNOLOGY

Institut für Fertigungstechnik
und Photonische
Technologien

Declaration in Lieu of Oath

I hereby declare to be the sole author of this thesis and that no part of my work has been previously published or submitted for publication. I certify, to the best of my knowledge, that this thesis does not infringe upon anyone's copyright nor violate any proprietary rights concerning ideas, techniques, quotations, or any other material from the work of others. All knowledge arising from external sources has been fully acknowledged below in accordance with standard referencing practices.

Vienna, January 2020

.....
Michele Buttazzoni

TECHNISCHE UNIVERSITÄT WIEN

VIENNA UNIVERSITY OF TECHNOLOGY

ACKNOWLEDGEMENTS

This work would not have been possible without the constant support and encouraging words of many people, of whom I would like to especially thank my supervisor and dear colleague Rodrigo Gómez Vázquez. His teaching methods and brilliant “can-do”-attitude gave me the confidence to dive into a whole new subject matter and start learning new and exciting things from scratch. My sincerest gratitude also goes out to Andreas Otto, who did not hesitate to give me a chance and supported me with his wisdom and patience throughout this project. Furthermore I am thankful for all the great people at the Institute of Production Engineering and Photonic Technologies, who either helped out and supported me directly or made my working environment a really great one to be in, just by being kind and joyful people.

Most importantly I would like to express my deepest appreciation to my family, without whom I would (literally) not be here. Thank you to my parents, my brother and all the others who supported me throughout all the years in more ways I would have imagined and even know care to comprehend.

Last, and surely no less important, a big thanks to all of my friends over the years, who made my life so much better in every aspect, just by being awesome people.

ABSTRACT

Simulation of manufacturing processes can be used as an alternative to exclusively experimental studies in order to drastically reduce product development costs due to lacking the need for a physical prototype. The goal of such a technology is then of course to reproduce the physical world as accurately as possible.

In this thesis an existing multiphysical solver for laser aided manufacturing processes based on OpenFOAM[1] will be adjusted in order to produce realistic simulation results when simulating Laser Direct Energy Deposition (L-DED) additive manufacturing processes. [2]

The focus will mainly be on the code development side, which was necessary to achieve the goals of simulating standard L-DED as well as so-called Extreme High Speed L-DED (EHLA[3]). At the end of this work, a validation with experimental results for L-DED will be carried out and a proof of concept for the simulation of EHLA will be presented to show the capabilities of this solver to compute more sophisticated problems.

TABLE OF CONTENTS

1 Introduction

Computer aided engineering has become an important step in the design process of products as well as processes. In order to reduce development cost and time to market, many manufacturers choose to invest in simulation technologies as an alternative of going the traditional route of building and testing a physical prototype.

To achieve this goal, a robust framework of appropriate mathematical models and solvers is needed to reproduce real physical processes as accurately as necessary and as efficiently as possible.

The focus of this work will be on additive manufacturing processes, mainly on Laser Direct Energy Deposition (L-DED) by metal powder, which is generally used to repair damaged parts which are too expensive to replace, e.g. turbine blades.[4]

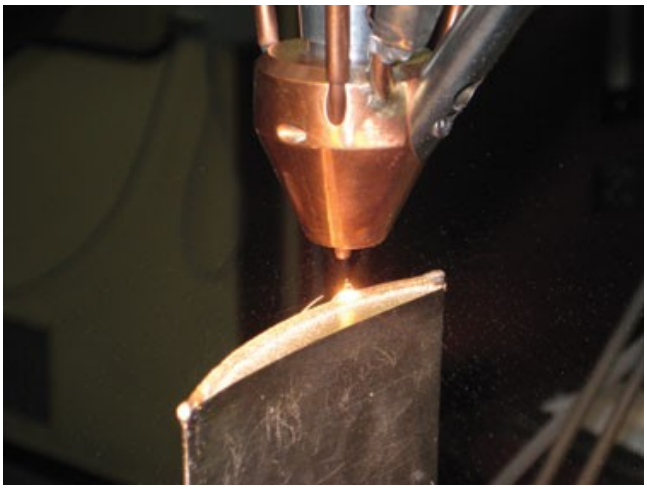


Figure 1: Turbine blade reconstruction using L-DED[21]

In L-DED, powder is usually injected through a conical nozzle, placed coaxially with a laser which acts as a heat source. As the laser head moves, the beam heats up and eventually melts the surface of the substrate, onto which the powder is directed. By properly adjusting the process parameters, the energy provided by the laser is enough to maintain a melt-pool where the entering powder particles get melted without reaching very elevated temperatures beforehand, which may induce the formation of a keyhole. As the bead cools down, a new structure is generated layer by layer by solidification of the melted materials. In this regard, the main difficulties nowadays lie in assuring the structural integrity of these final products during service conditions (i.e. extreme high temperatures, oxidizing atmospheres). Therefore knowledge of thermal gradients during solidification (i.e. as a way to evaluate the

feasibility of growing single crystal structures) and thermal cycles are of paramount importance. [5]

An additional process discussed in this work is Extreme-High-Speed L-DED, or EHLA ("Extremes Hochgeschwindigkeits-Laserauftragschweißen"), which is more of a coating process, since particles are already molten in the air by means of direct laser heating and the substrate is not heated to its melting point due to the high velocity of the laser. Nonetheless it is an important new development for the industry as such high velocities of 400 m/min have not been seen before in similar processes.[6]

The aim of this thesis is to improve upon an existing solver and extend models developed by the Institute for Production Engineering and Photonic Technologies of the Vienna University of Technology and based on OpenFOAM[1], in order to make it suitable for the simulation of L-DED, but still be versatile enough to simulate more sophisticated processes, such as EHLA.

The validation of the L-DED results was done using experimental results provided by Mitsubishi Heavy Industries, Ltd., while the simulation of EHLA should just serve as a proof of concept on which to tweak the algorithms in the future.

Following sections will be divided into development work, where problems, solutions and possible further improvements will be discussed, while the latter part of this thesis will present a verification of L-DED simulations and the proof of concept for EHLA.

2 Development Work

This section will focus on the improvements done to the source code of the solver in order to better the simulation capabilities.
Said changes are part of a series of improvements which can be broadly categorized into:

- particle injection (sections 2.1 and 2.2)
- particle physics (sections 2.3 through 2.5)
- additional physics (section 2.6)
- general performance improvements (section 2.7)

Each section contains a presentation of the problem that made the issue relevant enough to prompt the necessity to make changes to the source code. Furthermore the results of the improvements are presented, which are then critically assessed to gain confidence about potential benefits of implementing them. At the end of each section an overview of the current limitations of a particular solution are given and possible improvements outlined.

2.1 Injected mass rate: control algorithm

The main goal of this implementation was to achieve a relatively constant mass flow rate of so called Lagrangian particles (Inconel 718 powder) during the simulation. [7]

The previously existing approach let the user input a value defining how many particles to inject per second. This was a simple approach which worked for many use cases. However the experimental data available for L-DED usually includes information about particle injection density in units of a mass flow rate [$\frac{g}{s}$] instead of particles per unit time [$\frac{no}{s}$]. For this reason it was desirable to streamline the process and to let the user specify the value directly in mass flow rate units (i.e. in kg/s) without having to convert it.

The main issue however was that, when injecting a random assortment of particles with different diameters, it was not possible to achieve a constant mass flow rate while specifying only how many particles to inject per second.

Adding to that, the size distribution given in the experiments is in terms of volume fraction (instead of a more suitable form such as number of occurrences). It is also slightly skewed towards one side such as not to represent a normal distribution (see figure 40), but instead a Rosin Rammler distribution.[8]

For those reasons calculating a powder mass flow rate and subsequently converting it into particles per unit time was a relatively complex task.

Above mentioned uncertainties along with the pseudo-randomness of the injected particle size (within the bounds of such a distribution) caused the algorithm to make a significant error in the mass flow rate being injected even though the „correct“ amount of particles per second was used. Further complications were caused, when small amounts of powder had to be injected in each time step because of the use of following formula (for time steps that are reasonably small enough):

$$particlesToAdd = floor\left(\frac{currentTime - previousTime}{parcelsPerSecond}\right)$$

Equation 1: number of particles to be added using standard OpenFOAM injection approach

The floor-function is used to round down the calculated value to an integer. For time-steps where the value between current and previous time is less than the number of parcels per second, the floor function must return an integer and thus always results in no particles being injected.

A similar approach was tested, but using the ceil- (rounding up to the next integer) instead of the floor-function. This however resulted in injection at every time step (at least one particle), which resulted in an overestimation of the mass flow rate for very small time steps and/or very low particle densities.

$$particlesToAdd = ceil\left(\frac{currentTime - previousTime}{parcelsPerSecond}\right)$$

Equation 2: number of particles to be added using ceil function

Those difficulties lead to the need for another solution and thus an implementation of a control algorithm using mass flow rate per second as input parameter.

2.1.1 Implementation

After injection has started, a mass flow rate error is calculated via following equation where the powder mass feed rate is a user-given parameter:

$$massRateError = \frac{massInjected}{injectionTime} - powderMassFeedRate$$

Equation 3: mass rate error calculation

A positive mass rate error indicates a higher mass flow rate than required and thus no injection occurs. A negative value on the other hand causes particles to be injected in order to maintain the desired flow rate. Because of the random size of the injected particles within the given size distribution, an average number of particles,

which is required to reduce the mass rate error by one unit, is computed using following equation:

$$numberOfParticlesPerMRE = \frac{particlesAdded_{t-1}}{|(|MRE_t| - |MRE_{t-1}|)|}$$

Equation 4: marginal number of particles to inject

MRE in equation 4 and following text being the “mass rate error” which was previously calculated.

Once the value of the marginal number of particles to improve the injection rate by one unit MRE is known, one can easily calculate the number of particles to inject in order to stay within a certain tolerance of the target mass flow rate with equation 5.

$$particlesToAdd = ceil\left(2 \cdot |MRE| \cdot numberOfParticlesPerMRE\right)$$

Equation 5: number of particles to add in a given time-step

In order to obtain a faster convergence towards the target value, a convergence factor is given, which is used to increase the number of particles injected for each iteration in which the relative improvement in MRE is less than a given tolerance factor:

$$\frac{|(|MRE_t| - |MRE_{t-1}|)|}{|MRE_t|} \leq relativeToleranceFactor$$

Equation 6: condition for increasing the number of particles to be added per unit MRE

$$numberOfParticlesPerMRE_t = numberOfParticlesPerMRE_{t-1} \cdot convergenceFactor$$

Equation 7: adjustment of number of particles to be added per unit MRE

Similarly for a positive MRE and a slow rate of improvement, equation 9 is used to decrease the number of particles per MRE:

$$\frac{|(|MRE_t| - |MRE_{t-1}|)|}{|MRE_t|} \geq relativeToleranceFactor$$

Equation 8: condition for decreasing the number of particles to be added per unit MRE

$$numberOfParticlesPerMRE_t = \frac{numberOfParticlesPerMRE_{t-1}}{convergenceFactor}$$

Equation 9: adjustment of the number of particles to be added per unit MRE

A calibration of the value for the tolerance band and for the convergence factor was necessary. Setting a tolerance band which is too broad causes too many adjustments and thus results in loss of performance. The convergence factor, if set

too high, causes the algorithm to converge faster on the target value, but tends to overshoot it:

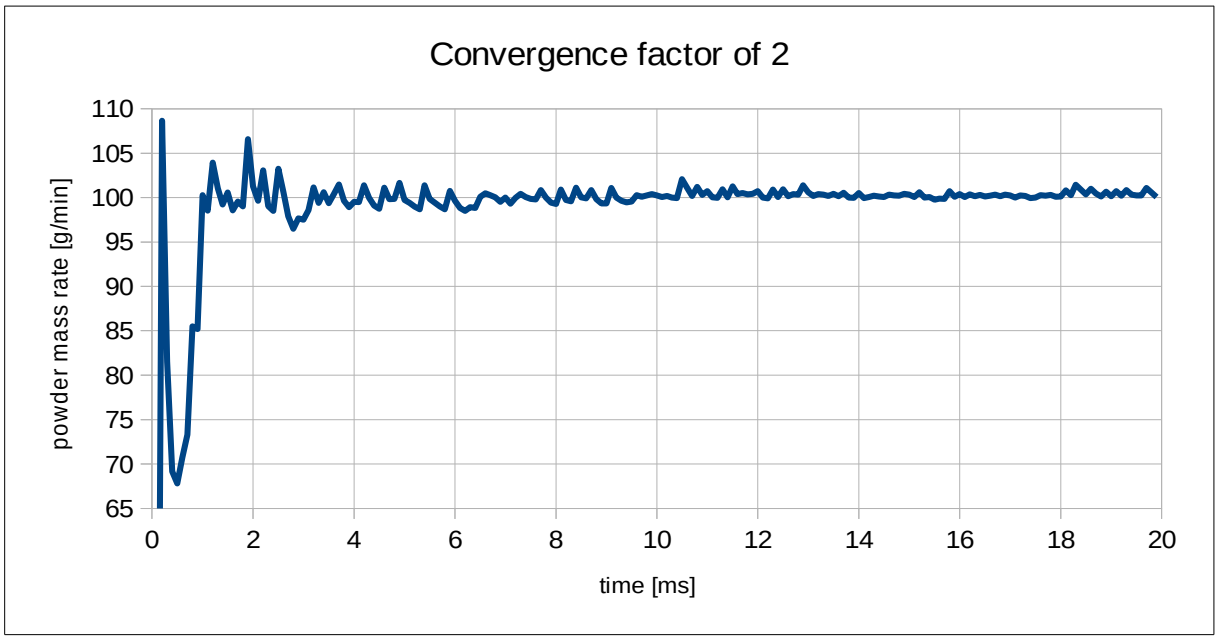


Figure 2: Injected mass flow rate using a high convergence factor; results in high overshoot of the target value

If set too low however it causes a slow convergence of the actual mass flow rate to the theoretical one, but less error once the target value is reached:

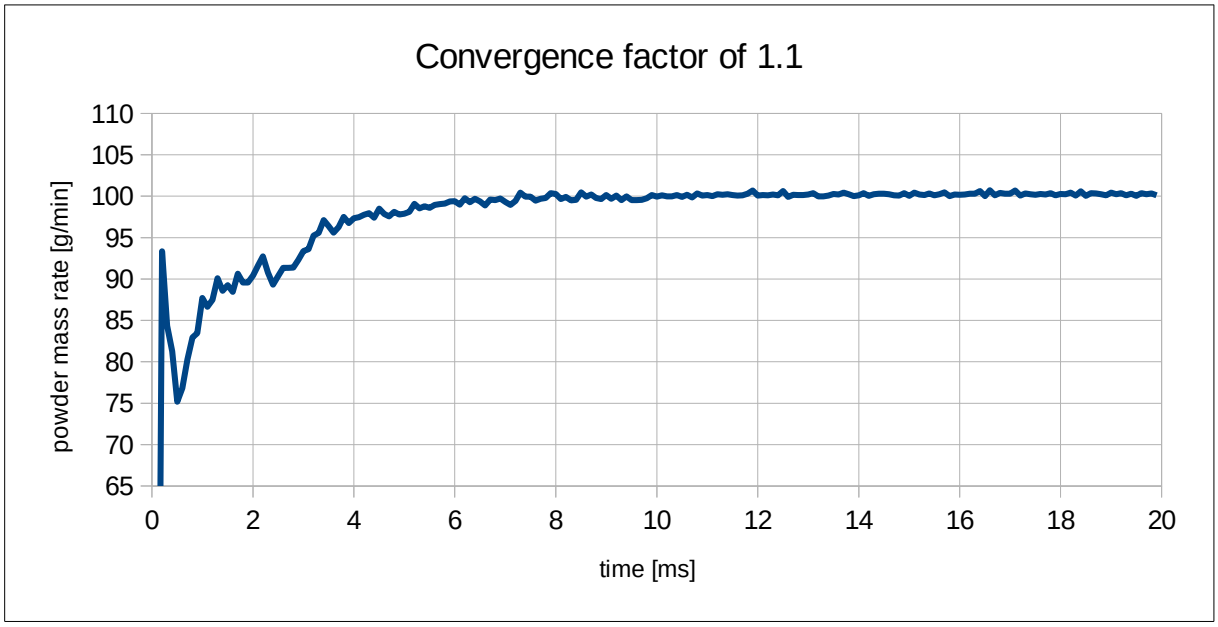


Figure 3: Injected mass flow rate using a low convergence factor; results in no initial overshoot above the target value

The higher the target mass feed rate is, the more this effect is noticeable, since the marginal number of particles per MRE is higher. For lower values such as in the case of typical L-DED conditions (e.g.: 1 to 2.2 g/min) this factor can be ignored.

The tolerance band should also be calibrated for the present study case, since it determines how big the improvement in MRE between time steps should be, in order to not change the number of particles per MRE. A higher value again causes too much of an adaptation and thus an overshoot over the target value, but can react much faster:

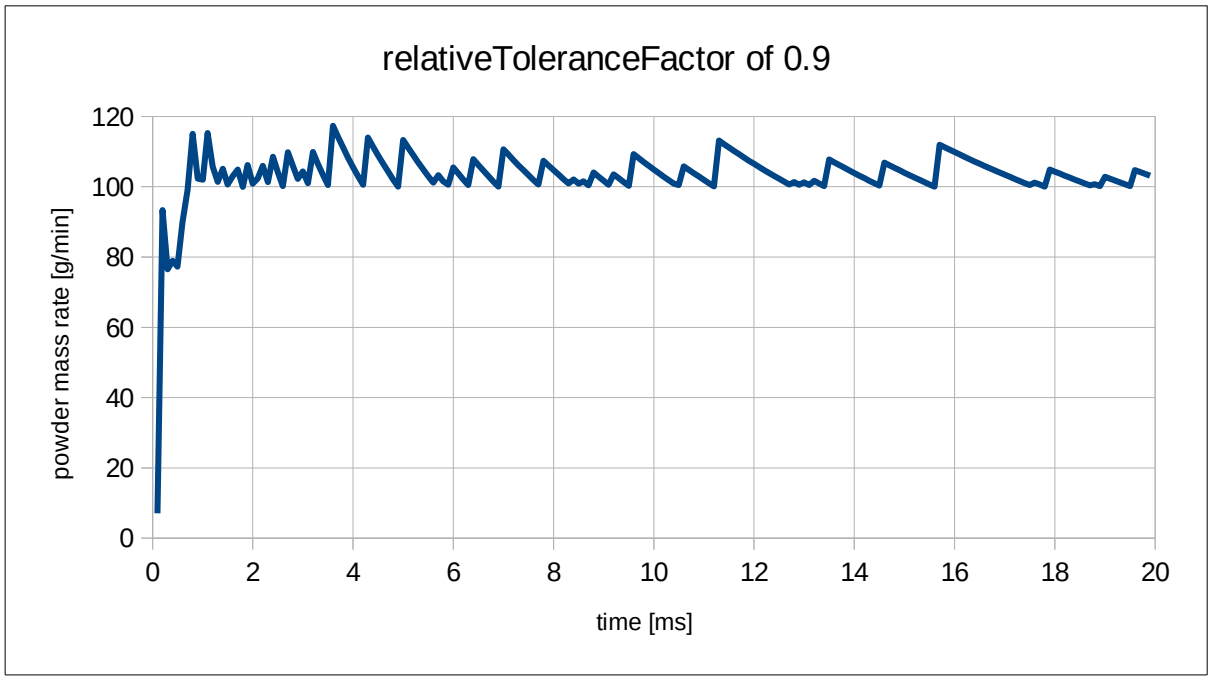


Figure 4: Injected mass flow rate using a high tolerance factor; results in low accuracy around the target value

For a lower value the method becomes slower but stays much more accurate:

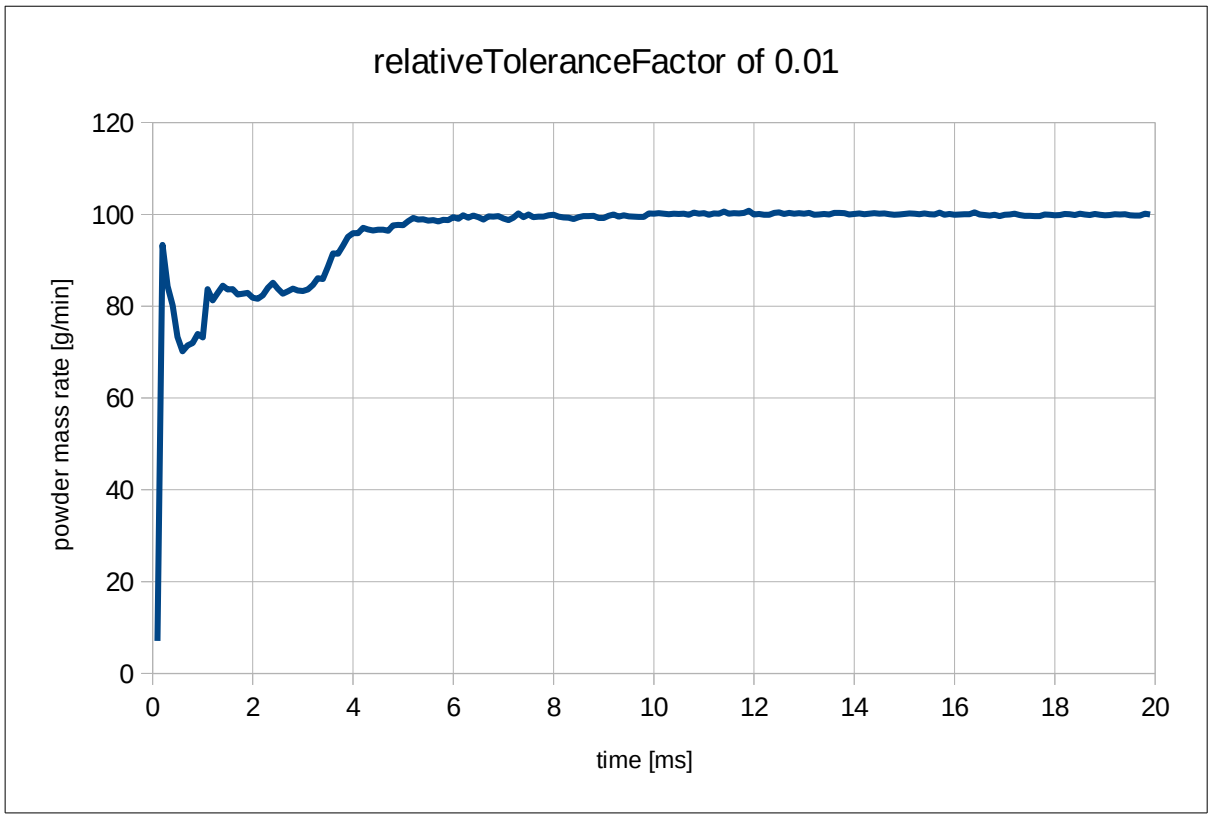


Figure 5: Injected mass flow rate using a low tolerance factor; results in high accuracy around the target value

2.1.2 Results assessment

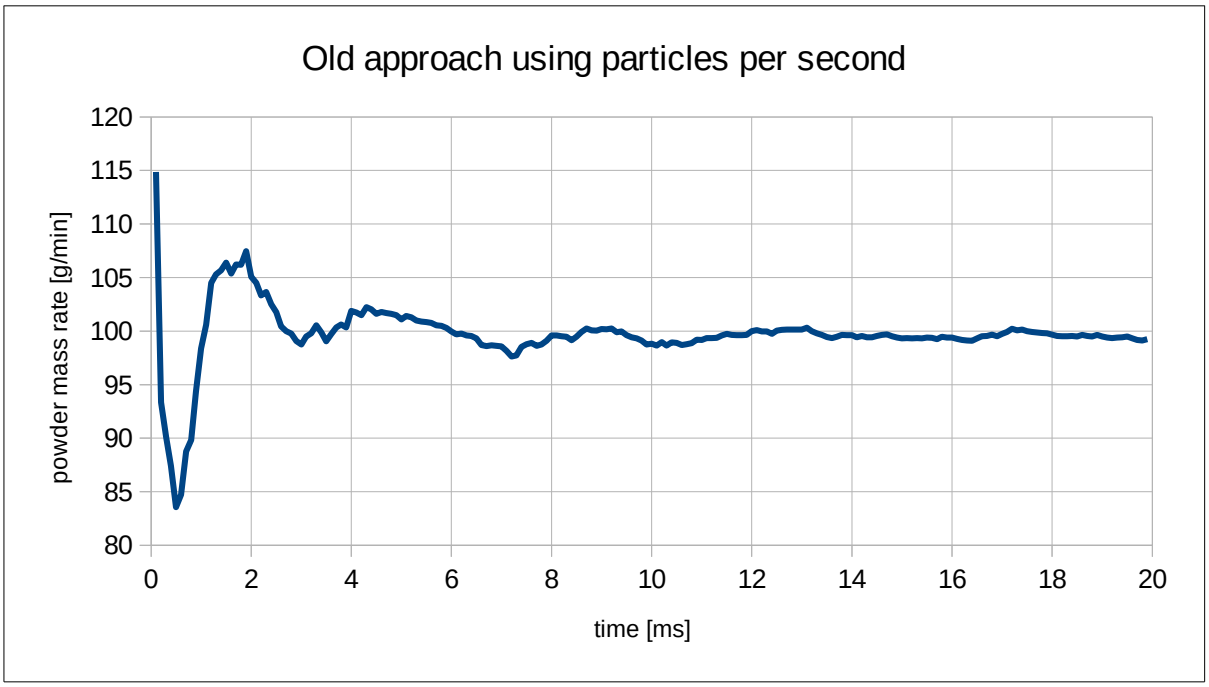


Figure 6: Injected mass flow rate using the old approach which used particles per second as an input

This new approach solves the problem of having to convert the given mass flow rate in g/min into particles per second for the model input. This caused unnecessary additional work when setting up a simulation model and was another possible source of errors. The bigger benefit however is the faster convergence rate toward the target value with the new approach.

Additional tuning of the parameters used by the control algorithm can lead to even higher accuracy at the target value, but at the cost of higher initial overshoot or slower convergence and vice-versa. The parameters set in figure 7 show a compromise between conflicting goals. These are the actual parameters used in the L-DED study:

- lower limit of tolerance band: 0.1
- upper limit of tolerance band: 1.9
- convergence factor: 1.5

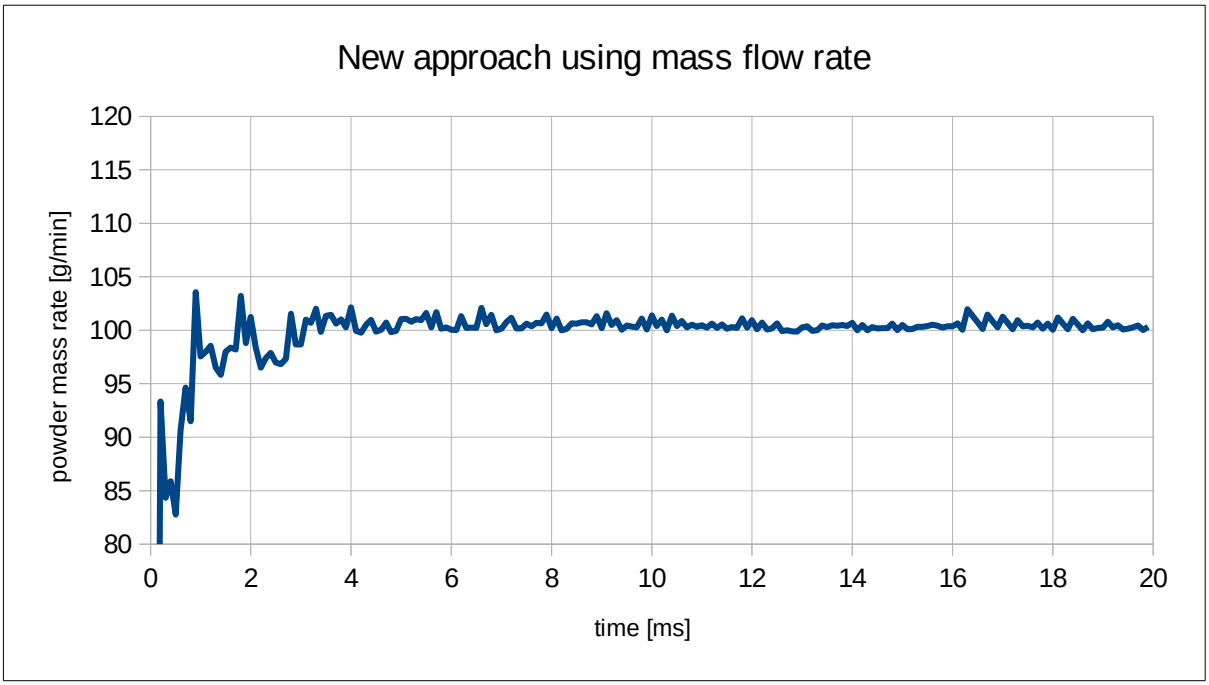


Figure 7: Injected powder mass feed rate using the new approach using mass feed rate as an input

2.1.3 Known limitations and possible improvements

The relative tolerance around the target mass flow rate value is hard coded and not thoroughly optimized. The same applies to the convergence speed factor. Both values have been calibrated empirically and thus could benefit of future analysis and optimization.

This is by no means a trivial task, because it should work for both extremely low and high target mass flow rates or be changed for individual applications and will probably yield just a slight improvement over the current implementation and thus, depending on the use case, may not be worth the time required to do so.

2.2 Injector speed correction

Due to the high velocity with which the laser and the injector move over the surface in EHLA, new “bugs” that previously went unnoticed were revealed. One of those was the particle at injection having the velocity of a particle being injected from an unmoving injector (meaning just having a relative velocity to the injector caused by the injection but not accounting for the movement of the nozzle itself). This of course lead to a huge error at injector speeds of 130m/min.[3]

2.2.1 Implementation

The injector speed was simply added to the calculation of the initial particle velocity.

$$\vec{U}_{injector} = \frac{\vec{s}_{endOfInjection} - \vec{s}_{startOfInjection}}{t_{endOfInjection} - t_{startOfInjection}}$$

Equation 10: Calculation of injector speed

$$\vec{U}_{particle} = U_{particle} \cdot \vec{d}_{injectionDirection}$$

Equation 11: Old implementation of particle initial velocity

$$\vec{U}_{particle} = U_{particle} \cdot \vec{d}_{injectionDirection} + \vec{U}_{injector}$$

Equation 12: New approach of particle initial velocity

2.2.2 Results assessment

In the following figures the rather important effects of the changes can be seen. In standard L-DED this error was not noticed because of the slow movement of the nozzle relative to the particle velocity.

In both pictures the movement of the nozzle is from right to left.

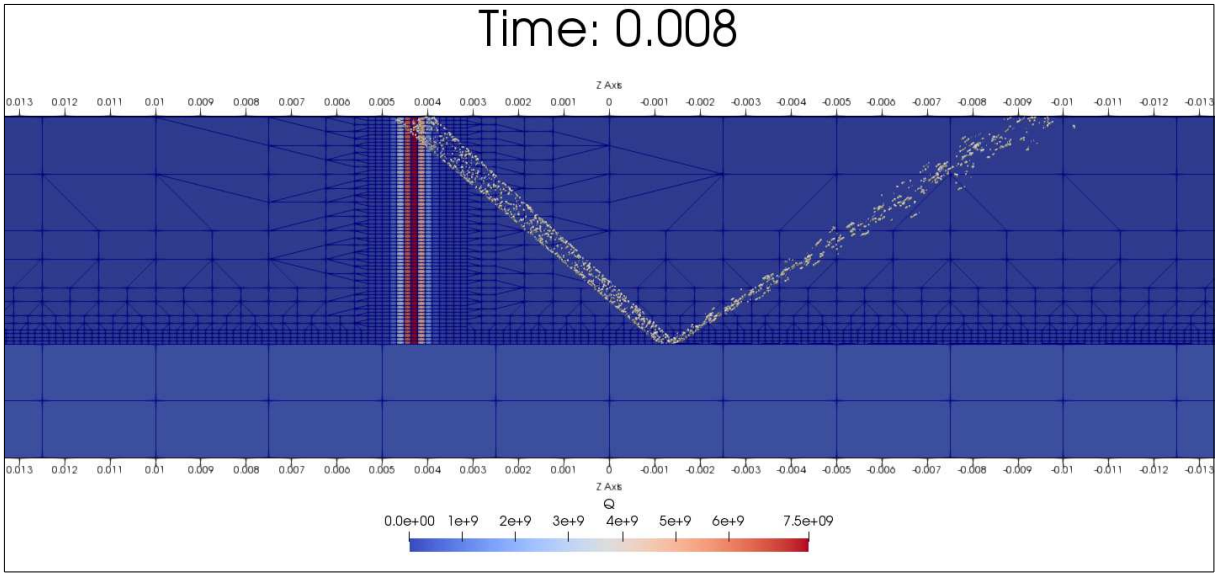


Figure 8: EHLA without the addition of the injector speed to the initial velocity of the particles (movement of the injector nozzle towards the left hand side)

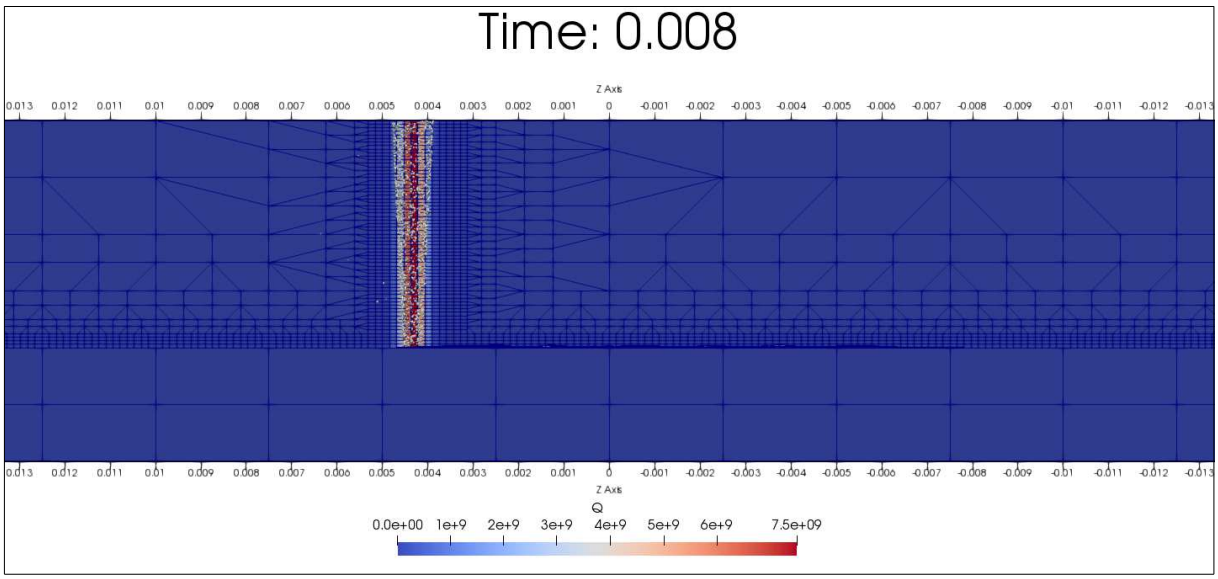


Figure 9: EHLA with the addition of the injector speed to the initial velocity of the particles (movement of the injector nozzle towards the left hand side)

2.3 Bouncing particles

The implemented approach for computing the collision and subsequent change in velocity of a particle hitting a solid surface consisted of checking if the inner surface normal vector, obtained from the solid gradient (direction in which the solid phase increases and thus vapour and liquid phases decrease within the simulation domain), and at least one component of the particle velocity vector point in the same direction and if the magnitude of said solid gradient surpasses a certain threshold.

This however has led to unphysical bouncing, due to the particle already being located inside the solid phase and changing direction from there and thus being reflected at the wrong position. In many occasions this also led to particles being trapped below the surface of the workpiece instead of being reflected at all, due to the high drag experienced once they entered the solid volume.

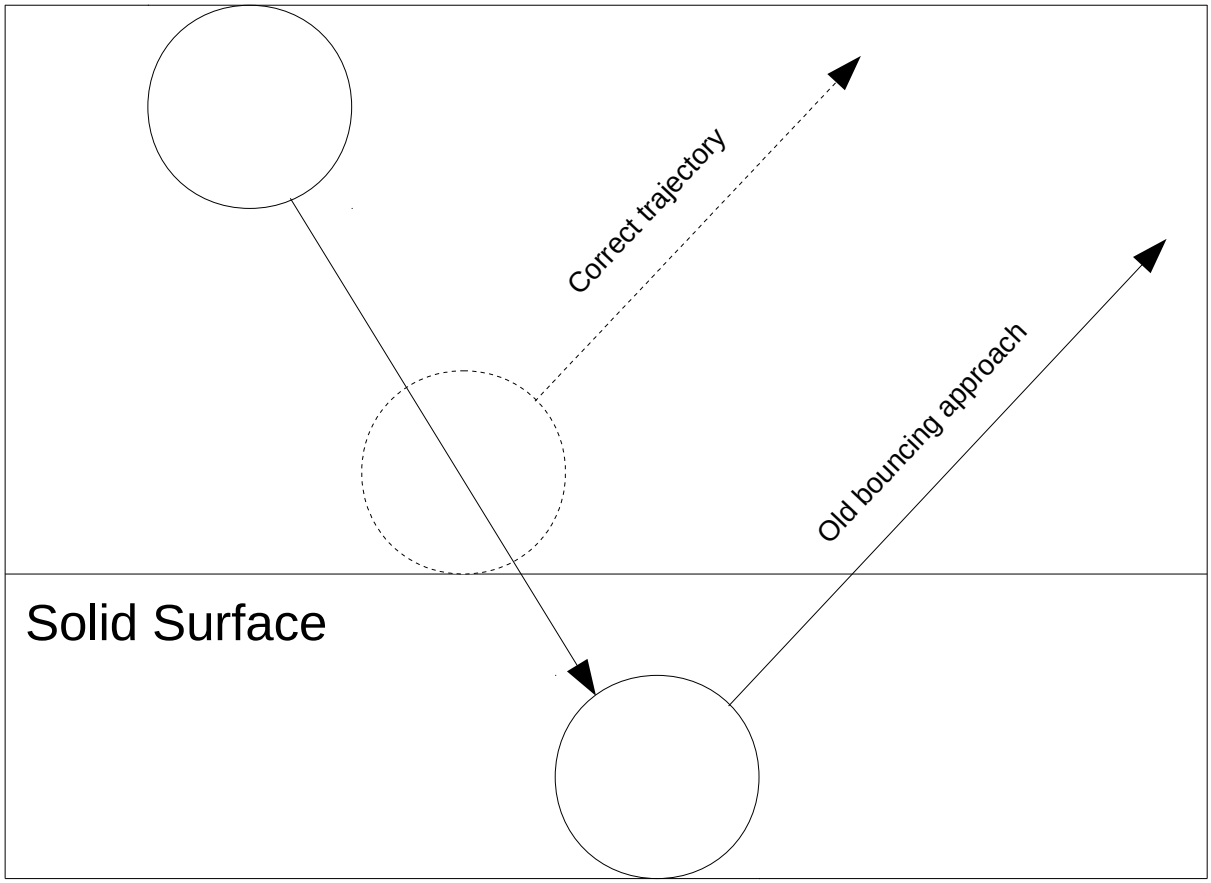


Figure 10: Unphysical vs. correct bouncing on solid surface

The goal of this new implementation is to ensure, that the particle bounces exactly on the solid surface (right where vapour phase changes to solid phase).

2.3.1 Implementation

The first condition for bouncing, namely that the particle velocity vector and the direction of the inner surface normal point in the same direction, remains the same, but the second one, where it is checked if the solid phase has surpassed a certain value is no longer needed. Instead a new theoretical position for the particle is calculated, while assuming that it continues with the current heading and speed for the whole time-step:

$$newPosition = oldPosition + \vec{u} \cdot dt$$

Equation 13: calculation of theoretical position assuming uninterrupted travel

where $\vec{u}$ is the particle velocity and dt the time-step length.

If at this new position the solid phase is larger than a certain user given threshold value, then at some time during the time-step, the particle is assumed to have bounced off a solid surface. The exact moment needs to be calculated as follows:

$$fractionUntilCollision = \frac{solidThresholdValue - solid_{t-1}}{solid_t - solid_{t-1}}$$

Equation 14: fraction of time step until bouncing will occur

where the calculated value is described as a fraction of the current time-step and the variable “solid” describes the amount of solid volume per cell volume found in the present cell.

Collision and thus bouncing happens at:

$$\vec{positionBounce} = (\vec{newPosition} - \vec{oldPosition}) \cdot fractionUntilCollision$$

Equation 15: position where bouncing will occur

The algorithm then splits the tracking of the particle into two distinct parts. The first part tracks the particle from the starting position to the solid surface and the second tracks it after the bounce with the new calculated velocity (see equation 16) until the end of the time step.

$$\vec{u_{new}} = \vec{u_t} \cdot (1 - frictionCoeff) - \vec{u_n} \cdot restitutionCoeff$$

Equation 16: particle velocity after bouncing

where $\vec{u_t}$ and $\vec{u_n}$ are the tangential and normal velocity components relative to the surface respectively.

2.3.2 Results assessment

A benchmark scenario with a pre-specified solid weld-bead (quickly set using OpenFOAM’s “setFields”-functionality) was simulated where a lot of particles were injected from the top left in the following figures. Blue particles have a downward component to their velocity, while red ones are travelling upwards. This was done to better illustrate which of them have bounced. Both simulations use the exact same setup, with the only difference being the bouncing method used.

As can be seen from figures 11 and 12, the marked particle on the right side should have bounced off the solid surface, but instead travelled through the surface when using the old approach.

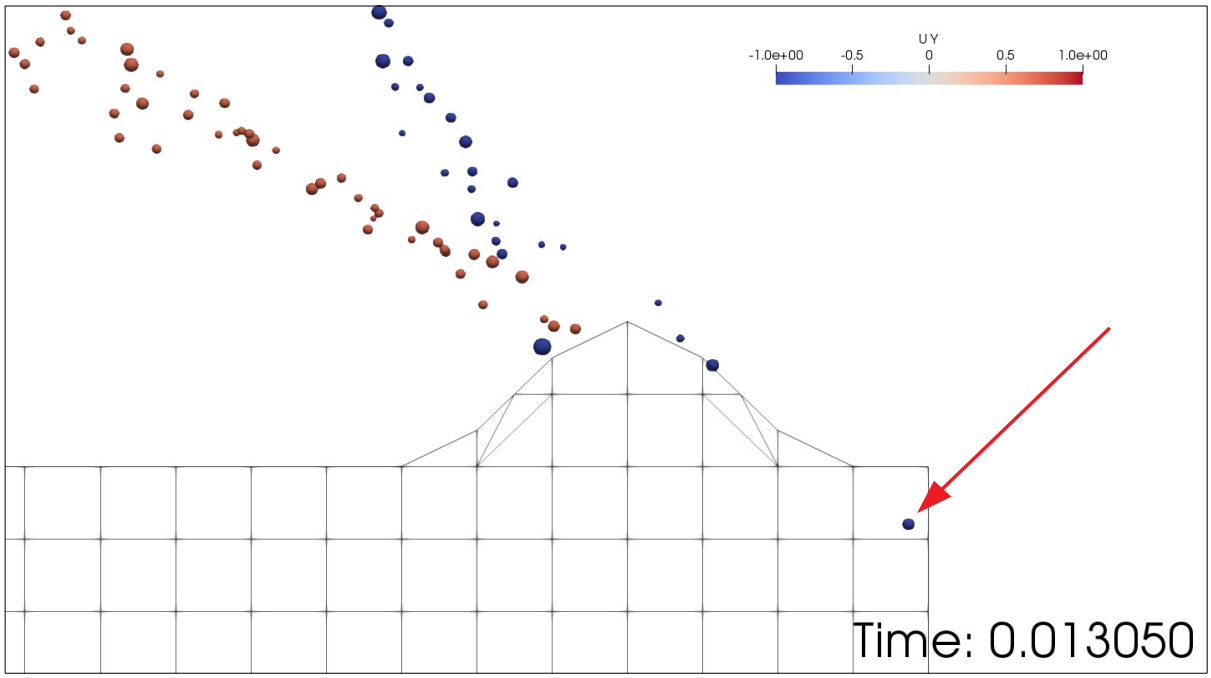


Figure 11: Old bouncing approach on weld bead

When using the new method, the same particle bounced off at the right location. Both figures show the same setup at the exact same moment in time as can be seen by the similarity of particles with downward (and most with upward) velocity.

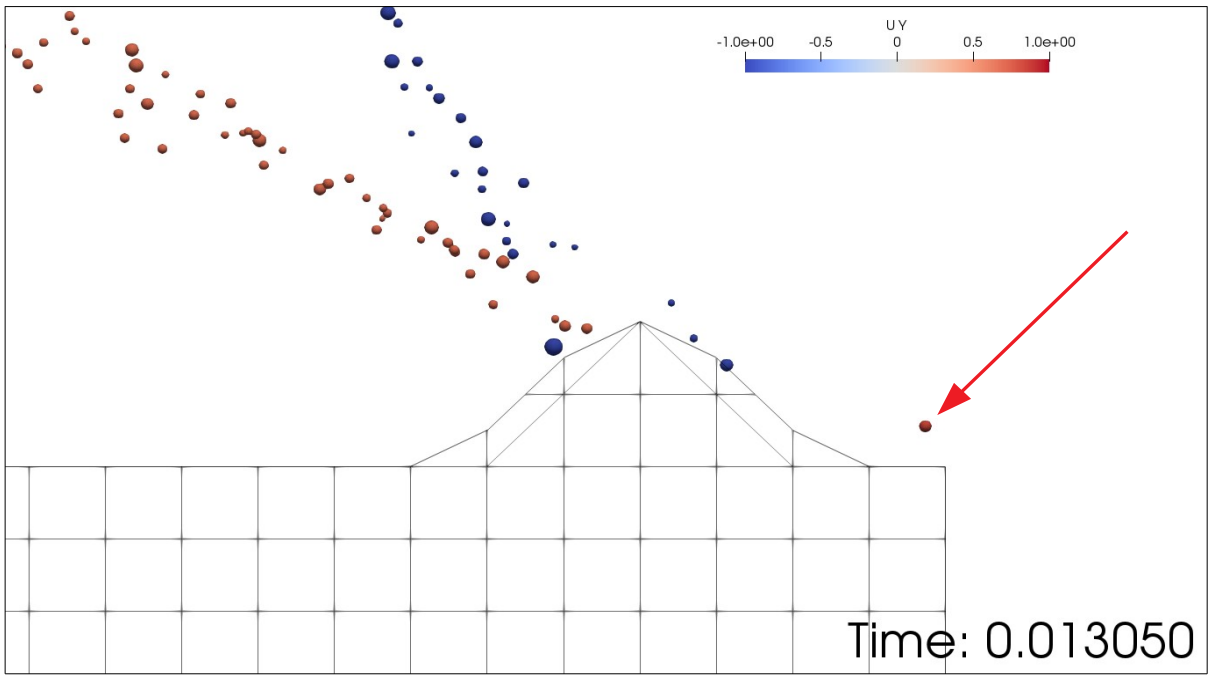


Figure 12: New bouncing approach on weld bead

2.3.3 Known limitations and possible improvements

At the moment a particle bounces whenever the cell centre reaches a solid surface, which is not entirely physically accurate as it should bounce exactly the distance calculated in equation 17 before.

$$\vec{x} = -\frac{r_{particle}}{\sin\left(atan\left(\frac{|\vec{u}_n|}{|\vec{u}_t|}\right)\right)} \cdot \frac{\vec{u}}{|\vec{u}|}$$

Equation 17: distance vector from current bouncing location to correct one

For particles of a small diameter, like in the L-DED simulations (< 200 µm), this effect is negligible and thus there is no need for a better model. However in the future the need to simulate larger particles may arise and thus the bouncing should happen at the edge of the particle and not at the centre.

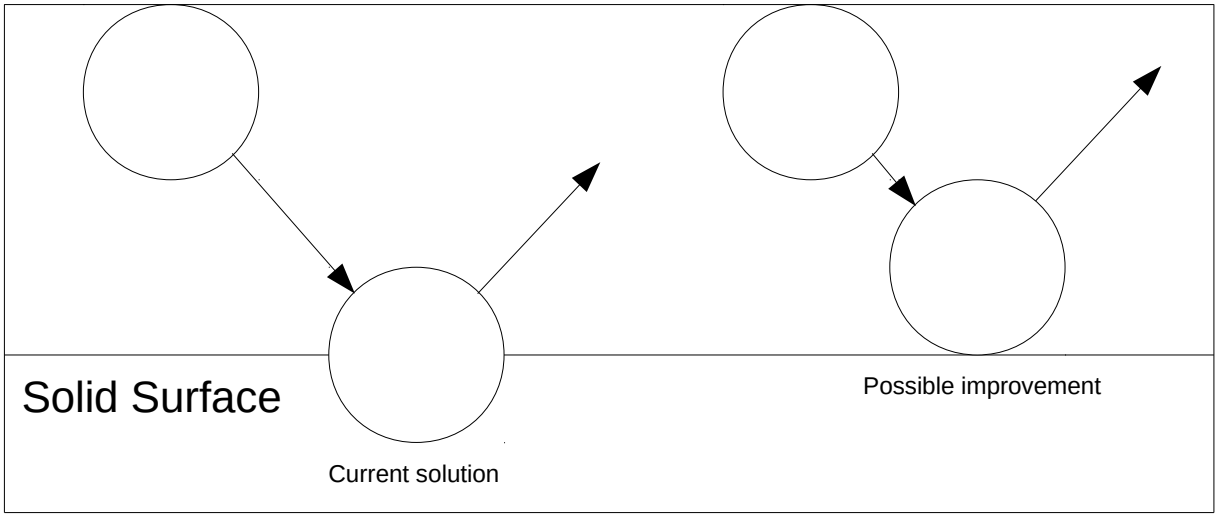


Figure 13: Current vs. possible implementation

2.4 Melting particles

Due to the two-way-coupling of Discrete Element Method (DEM) and Volume Of Fluid (VOF)[9], one has to be especially careful when designing interactions between particles and the surrounding fluid. During L-DED processes, when a particle melts, its mass, energy and momentum are transferred to the equivalent VOF-phase that is in most cases surrounding it (i.e. in the meltpool).[10] When a particle however melts before reaching the substrate during the L-DED process, the high density of this liquid VOF-phase causes other particles to experience a lot of extra drag forces and to slow down during the flow through the entire cell (since the melted particle is now

part of the fluid and thus it distributes evenly within one cell). This of course is unphysical especially for coarser meshes with larger cells.

One solution to this problem, which was implemented, is to force the particle to remain in a solid Lagrangian-DEM state while it still is surrounded by a gaseous phase, even though melting temperature has been surpassed. It would then only be transferred to the Eulerian VOF-formulation when entering into a solid or liquid phase.

In L-DED this approach worked well, because the laser usually had enough intensity and time to melt the substrate and thus the particle would fall into a liquid phase, melt, and transfer mass, momentum and energy to said phase, as mentioned before.

In EHLA however the laser moves so quickly relative to the substrate, that melting of the latter is not achievable with the used intensity values[3]. This causes the particle to bounce off the solid surface because it still retains its solid state, no matter how high the temperature. Thus a new approach is needed where the issues of high drag forces can be overcome while the particles are able to melt even on a solid surface, when their energy is high enough.

2.4.1 Implementation

Similar to the bouncing particle implementation, the algorithm checks whether the next cell, which is reached within one time step by a particle, contains a certain amount of solid (parameter given by the user). If the condition is true, the particle is allowed to “melt”[1] at the current position (assuming that the melting point has been reached). This leads to a liquid phase in the cells adjacent to the solid surface.

Of course a particle can still melt when being surrounded by a liquid phase.

2.4.2 Results assessment

Because of the high particle density the condition implemented beforehand (certain amount of solid in a cell) became true, while the particle was still in the air and thus melted there. This however causes a lot of drag on the surrounding particles and the simulation entered an infinite loop (finite, just so long to not be able to wait for the next time-step to be written out to disk) after a few seconds of run-time.

With the new condition, the particles melt in the cell before the surface and thus this problem does not exist any more.

A proof of the results can be seen in section 3.4, where the EHLA process is simulated and the particles do melt in order to form a bead on the surface.

[1]In this context melting means: to be transferred from Lagrangian-DEM to Eulerian-VOF formulation

2.4.3 Known limitations and possible improvements

In this approach all particles are treated as solid until they reach the substrate, thus all interactions between them are also treated as two solids colliding and then changing their respective velocities (if particle collision is desired and turned on as a setting in the simulation). This effect is negligible in L-DED and EHLA simulations because particle interactions are usually turned off to save on computational power and because they do not affect the outcome of the simulation in any significant way.

In the future the algorithm should be changed to include the possibility for a particle to melt in the air, but maybe retain its geometry, change to a liquid state, but still get tracked by the mesh-less DEM-formulation instead of becoming part of the VOF.

2.5 Laser energy absorption by the injected powder

The goal of this function is to calculate the fraction of a cell area which is getting blocked by the presence of particles, hereafter defined as eta and by extension the non-obscured area of each particle for the purposes of calculating the energy absorbed by each individual particle (area on which a particle can absorb laser energy, because it is exposed to the light).

The previous approach used the sum of the projected areas of particles within a cell for eta (see equation 18) and the projected area of a single particle for the absorbed energy.[11] This works fine for cases with either very small cells, where particles exist (eta is always computed per cell), or a low particle density.

$$eta_{old} = \frac{\sum_{particles} A_{particle}}{A_{cell}}$$

Equation 18: Old approach for eta calculation

$$A_{particle} = r_{particle}^2 \cdot \pi$$

Equation 19: Calculation of particle area

$$A_{cell} = V_{cell}^{2/3}$$

Equation 20: Calculation of cell area (no easy access to cell length inside the solver)

$$E_{\text{abs}} = Q_{laser} \cdot V_{cell} \cdot (1 - R) \cdot \frac{A_{particle}}{\sum_{particles} A_{particle}}$$

Equation 21: Absorbed energy by an individual particle

where:

$A_{particle}$: unobscured area of particle

$r_{particle}$: radius of particle

V_{cell}: volume of the cell

E_{abs}: absorbed energy per particle

R: reflectivity (material property)

Whenever a higher particle density is needed or the cell size is not small enough relative to the mean diameter of the particles, the error grows larger, because the probability of a particle being “obscured” (part of the particle’s area where the light is blocked by another particle and thus receives no energy from the laser) also grows larger. This would cause the previous algorithm to still take all of the particle’s areas into account when computing the fraction as shown in the following figures and this would mean that even a few particles stacked on top of each other would cause a large eta:

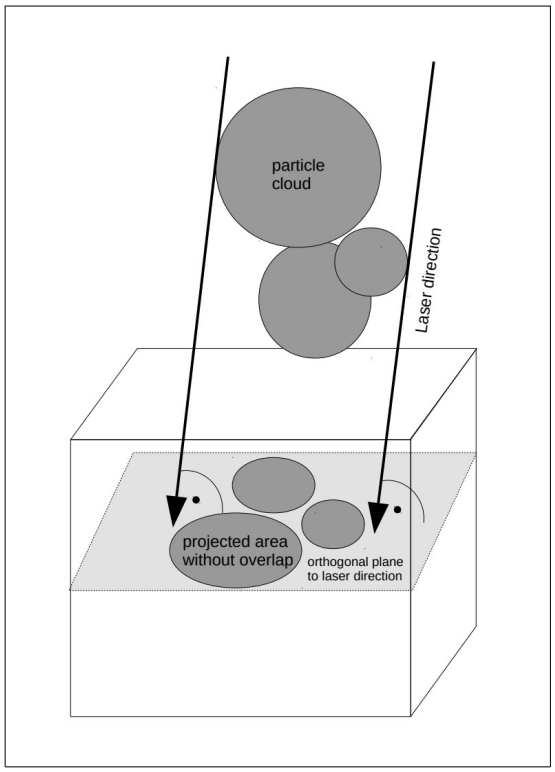


Figure 14: Projected area used by old approach

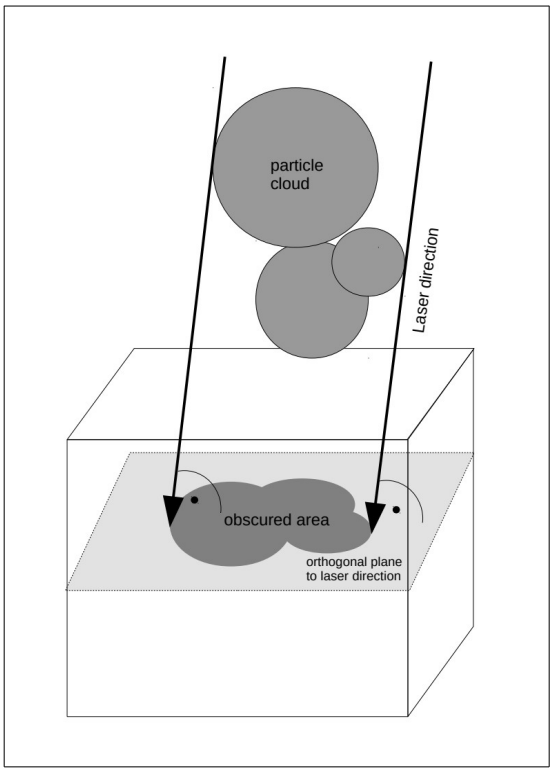


Figure 15: Projected area used by new approach

2.5.1 Implementation

The proposed method uses the number of particles per cell to determine the most efficient and least inaccurate way of calculating the projected area of the particles, taking into account laser direction and the order in which they lie within the cell relative to the laser source (uppermost particle cannot be blocked and subsequent particles get calculated afterwards) and thus eliminating previously described inaccuracies. It is divided into four different functionalities, for the presence of one, two, three and more than three particles per cell respectively.

In addition to those functionalities the algorithm needs to sort the particles by distance to the laser source in order to correctly calculate the absorbed energy per particle for each one, given its position and thus the amount of area that overlaps with others before it.

0 particles per cell:

If there are no particles present in a cell, no energy is absorbed by them and thus eta is zero.

1 particle per cell:

Since no overlap between more particles is possible within a cell, the calculation of eta is trivial:

$$eta_{1\,particle} = \frac{r_{particle}^2 \cdot \pi}{A_{cell}}$$

Equation 22: Calculation of eta for 1 particle per cell

Also the only particle present of course absorbs all of the laser energy relative to its size because it cannot be blocked by others.

2 particles per cell:

Again the sum of the projected areas is used for calculating eta, however the presence of a second particle has introduced the possibility of partial (calculation of overlap) or complete blockage (ignore the area of completely obscured particle) of energy. To accomplish this, the particles have to be sorted by distance to the laser source which is done by storing all the needed information in a two-dimensional array and sorting it by this value using the standard C++ - library's function sort().

For a more detailed calculation of the overlap please refer to the appendix (section 6.1).

3 particles per cell:

From here on to the end of the section, particles with the index A will denote the one nearest to the laser source, then in ascending order of distance B, C, etc. .

Again the area of common overlap of the particles is needed to correct for partially or completely obscured ones. However the presence of the third particle has rendered this task much more difficult, because while there were 3 different scenarios for 2 particles (no, partial or complete overlap), now there are 14 as shown in figure 16:

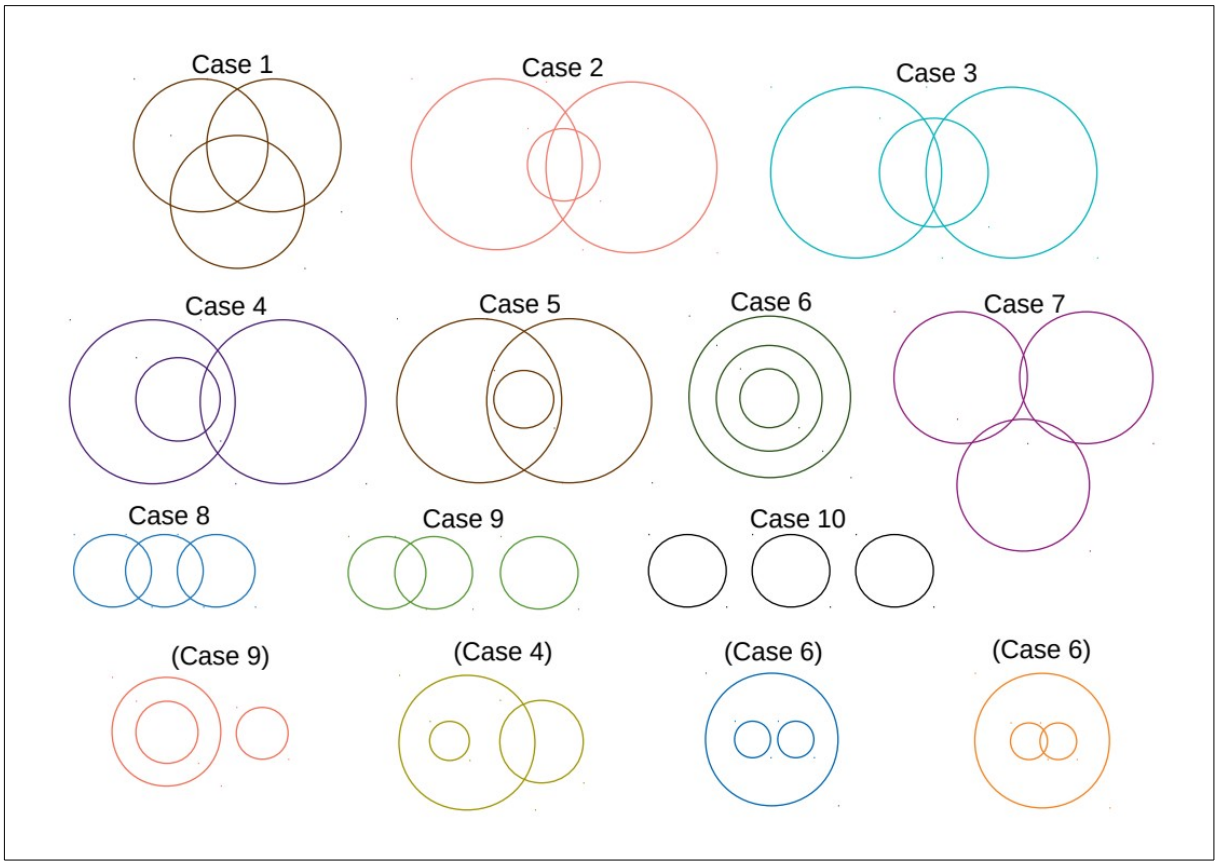


Figure 16: Three overlapping circles categorization

Even though 14 different cases can be differentiated topographically, for a complete computation of the area of common overlap only 9 are needed. Before the algorithm is able to differentiate between them, some further operations are needed. First, the particles need to be sorted by their size and their distance from the laser as has been described previously. Then their position on a plane orthogonal to the laser direction must be computed as has also been done for 2 particles. Finally the conditions for no, partial or complete overlap must be checked just as before, only repeating this for all the pairs of circles.

Using and expanding upon those conditions every possible scenario will be categorized using one of the 9 different scenarios in figure 16 and relying on the equations by Fewell, M.P.[12] for classification and calculation of the common overlap of case 1.

For a more detailed explanation of the case selection and the calculation of the unobscured particle area please refer to the appendix (section 6.2).

Sub-cell-discretization algorithm (more than 3 particles per cell):

For 4 or more circles the calculation of the projected area becomes an almost impossible task to accomplish analytically, because of the 173 different configurations in which 4 circles can overlap each other and the 16,951 ways 5 circles can and so on.[13] The task is thus solved, not by an exact analytical approach, but by means of a “brute-force” sub-cell-discretization algorithm:

First, the projected area is divided into sub-cells (the amount of which dictates the required computing power and the achieved accuracy; parameter is given by the user as a scalar value). Then the algorithm must project all particle positions onto an area orthogonal to the laser direction. This is done the same way as for a lesser number of circles as described before. Finally the distance of the farthest particle from the centre plus its radius times 2 dictates the maximum side length, so as to increase the accuracy for a given number of sub-cells (sub-cells are smaller in size):

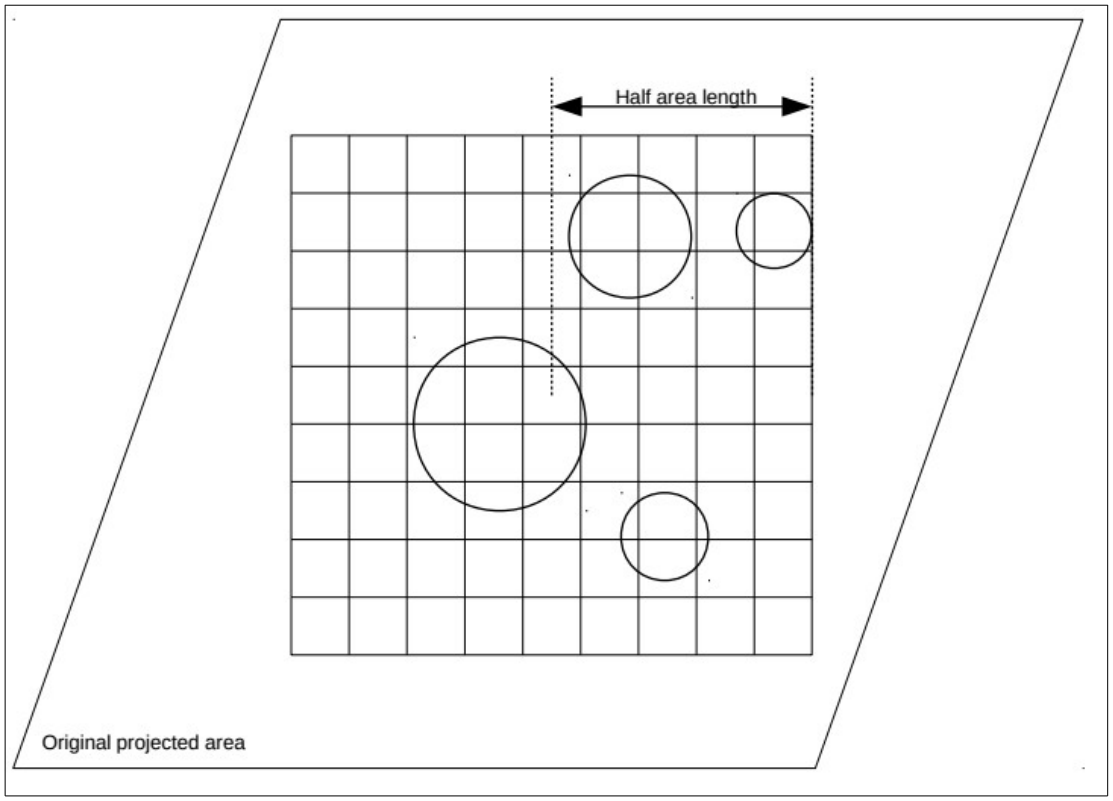


Figure 17: Original projected area and calculation area with given side length

The projected area, at the present moment, always spans a square and the length and width of one sub-cell is thus also the same and defined as follows:

$$l_{subCell} = \frac{l_{area}}{no_{subCells}}$$

Equation 23: Size of a sub-cell

where:

$l_{subCell}$: length of a single sub-cell

l_{area}: length of the area spanned by the sub-cell-grid

$no_{subCells}$: number of sub-cells

In each of those sub-cells the algorithm checks if it lies within one of the circles and changes its floating point value between 0 and 1, using the approach previously developed at the Institute for Production Engineering and Photonic Technologies of the Vienna University of Technology for calculating laser intensity inside a cell in the case of a top-hat energy distribution. Repeating this for all the sub-cells produces a grid as shown below (values between 0 and 1 are possible, though they were rounded for better visualization).

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0
0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0
0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0
0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0
0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0
0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0
0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0
0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0
0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0

Figure 18: Sub-cell grid visualization with a recognized circle

Repeating this method for all particles and taking the weighted sum as shown in equation 24 produces the value of the projected area.

$$\sum_{particles} A_{particle} = \left(\sum_{subCells} f_{subCell} \right) * l_{subCell}^2$$

Equation 24: calculation of the sum of all the areas

where:

$f_{subCell}$: function output between 0 and 1 in each sub-cell

This approach works reasonably well for calculating the sum of the projected areas. The assigning of the correct $A_{particle}$ to the correct particle is done by using the sorted list of particles as input for the brute force approach, which already loops over each particle separately. Then it is just a matter of saving the $A_{particle}$ of the previous particle and subtracting it from the current iteration:

$$A_{particleA} = \left(\sum_{subCells} f_{subCell} \right) * l_{subCell}^2$$

$$A_{particleB} = \left(\sum_{subCells} f_{subCell} \right) * l_{subCell}^2 - A_{particleA}$$

etc.

2.5.2 Results assessment

For one, two and three particles the algorithm is completely accurate because of its analytic nature. Validation was done by applying the brute force algorithm to a case where there are three particles per cell and comparing the results to each other.

Since 0 and 1 particles are trivial and 2 particles is just an easier version of the algorithm for 3 overlapping particles, only the latter will be checked formally.

The brute-force approach was difficult to validate quantitatively because of the larger number of particles. But since a comparison between the analytical approach and the brute-force one can be made for three particles, little deviation of those values from one another in the same simulation setting would indicate either good working of both approaches or the same errors in both. But since they work so differently one can easily see, that the latter option is virtually impossible and thus this kind of validation is sufficient.

First however it was checked that the brute-force approach produced the correct result for 4 non-overlapping particles, where the projected area can easily be calculated. Repeating this for different numbers of sub-cells yields an accuracy value for each case. Comparing this to the computing time per time-step as seen in the graph in figure 19 can support the user in the decision of how many sub-cells to use for a given scenario.

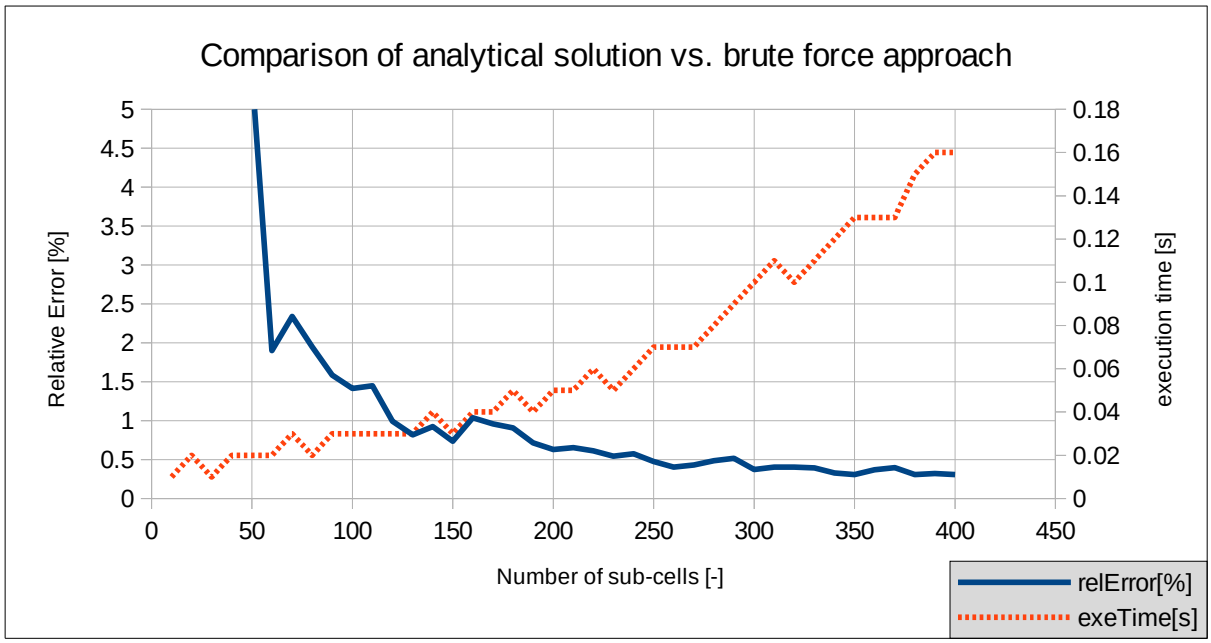


Figure 19: Accuracy and speed comparison for different numbers of sub-cells

For this application and others, where the size of the particles relative to the cells is similar, approximately 150 sub-cells would be sufficient to yield an error of <1% and still have a reasonably low computing time.

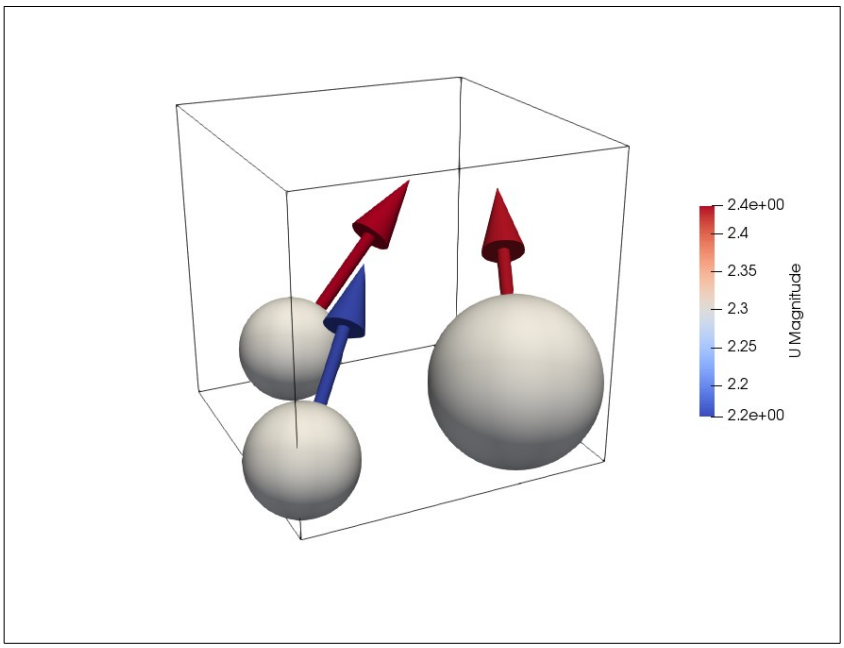


Figure 20: Testing setup for eta calculation

With this accuracy value in mind, a simulation, where 3 particles were injected into one single cell, was computed. Here the particles did not experience any forces acting on them and would bounce completely elastically at each interaction with the wall. Collisions between the particles were turned off so as to allow them to travel inside each other, which, for the purposes of this test, was entirely reasonable and would be the same as them passing “under” (in terms of relative distance to the laser source) each other. This was all done to increase the number of different scenarios, that the particles would experience, and to decrease computation time as debugging the analytical solution turned out to be the main problem area of this validation. In order to validate both approaches, the present model was simulated twice, once using the analytical solution and once using the brute-force approach and the results of the particle areas and the sum of all areas for eta calculation was compared:

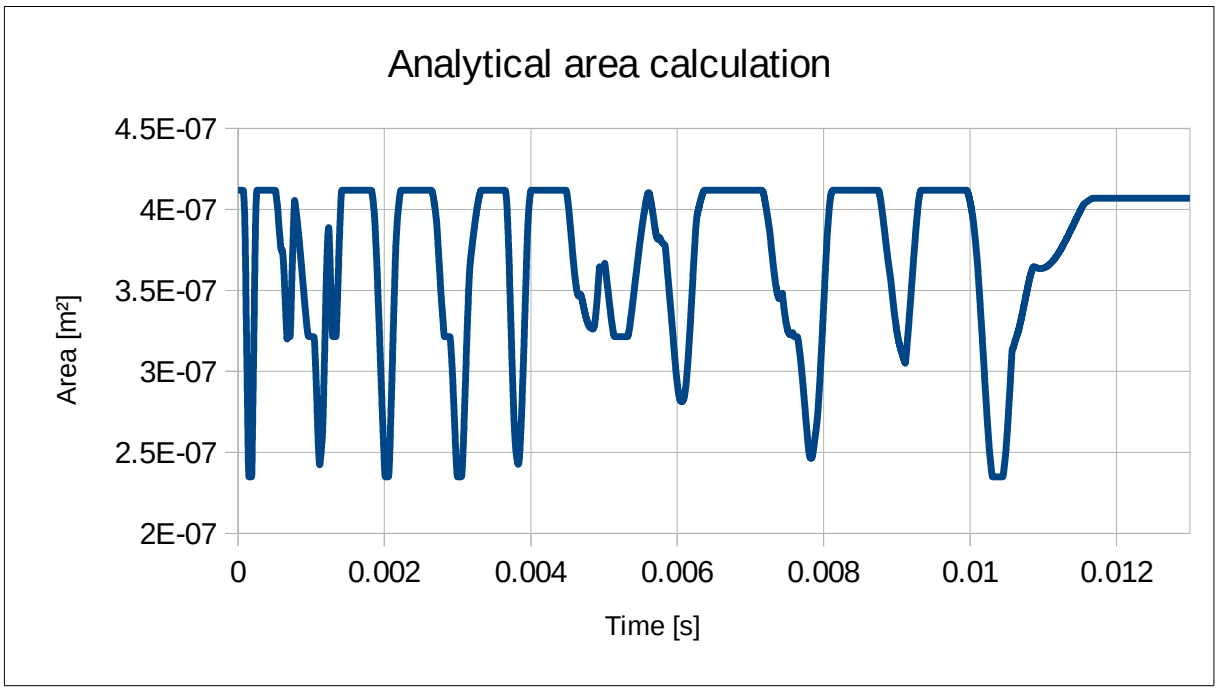


Figure 21: Sum of all particle areas using analytical approach

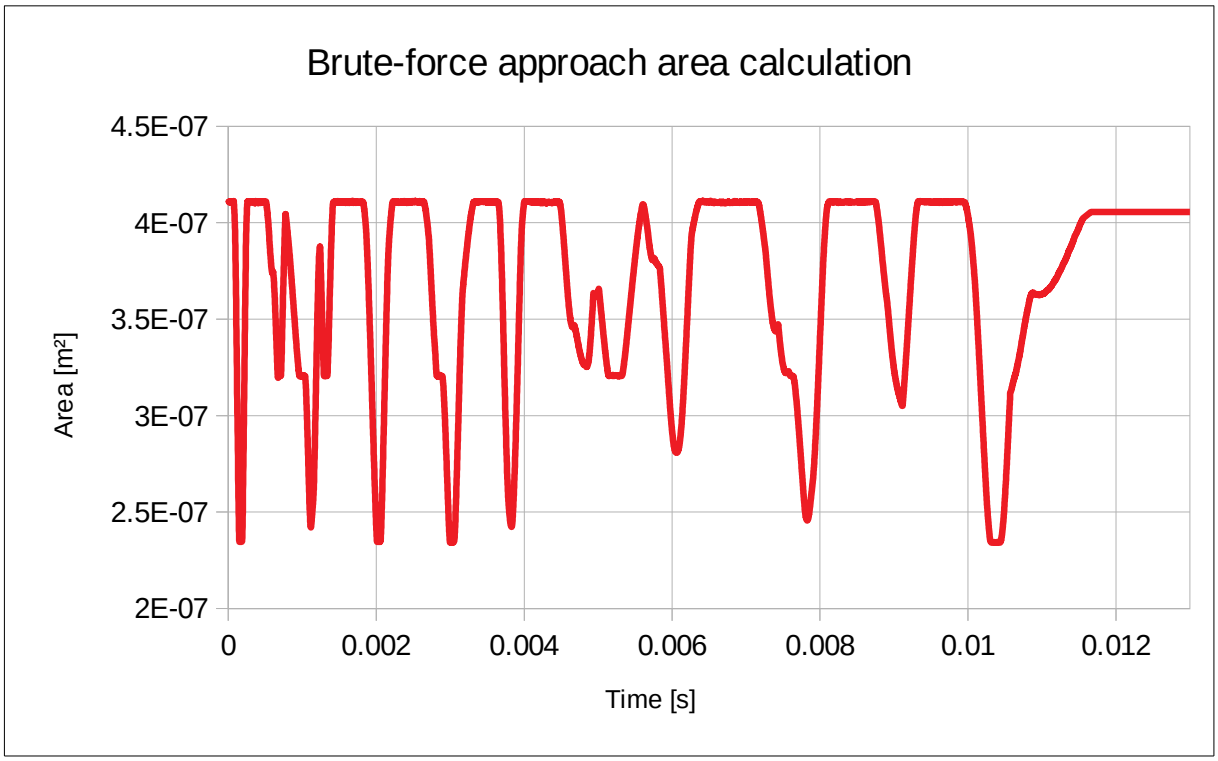


Figure 22: Sum of all particle areas using brute-force approach

Figure 23: Relative error between the two approaches

As can be seen from the graphs in figures 21, 22 and 23 the particles' areas overlapped with each other many times and the error between the two approaches remained well below 1%. This error is caused only by the finite resolution of the sub-cell grid in the brute-force approach.

In figure 24 the case number is plotted over time in order to show, that all cases have been entered (except for case 5, which would require one of the particles being much smaller than the other two and will be validated separately) and thus been validated by this method:

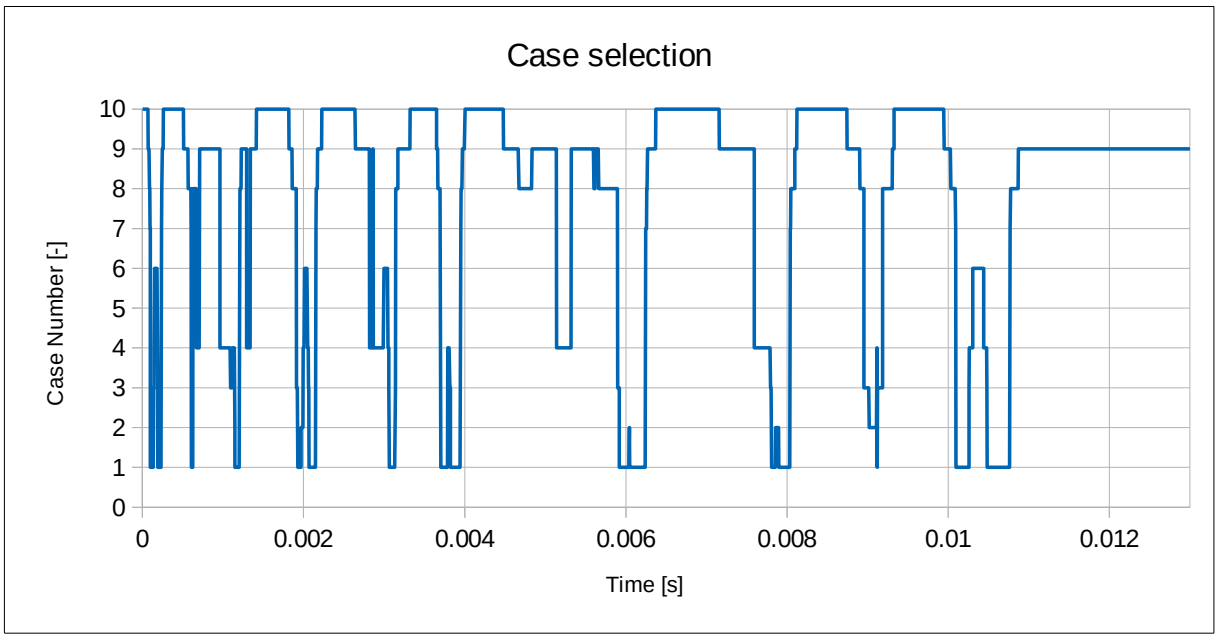


Figure 24: Case selection during the analytical approach

The following charts show a second run of validations where two particles have a diameter of 100 µm and the third one of 1 µm. This was done in order to validate case 5 as well.

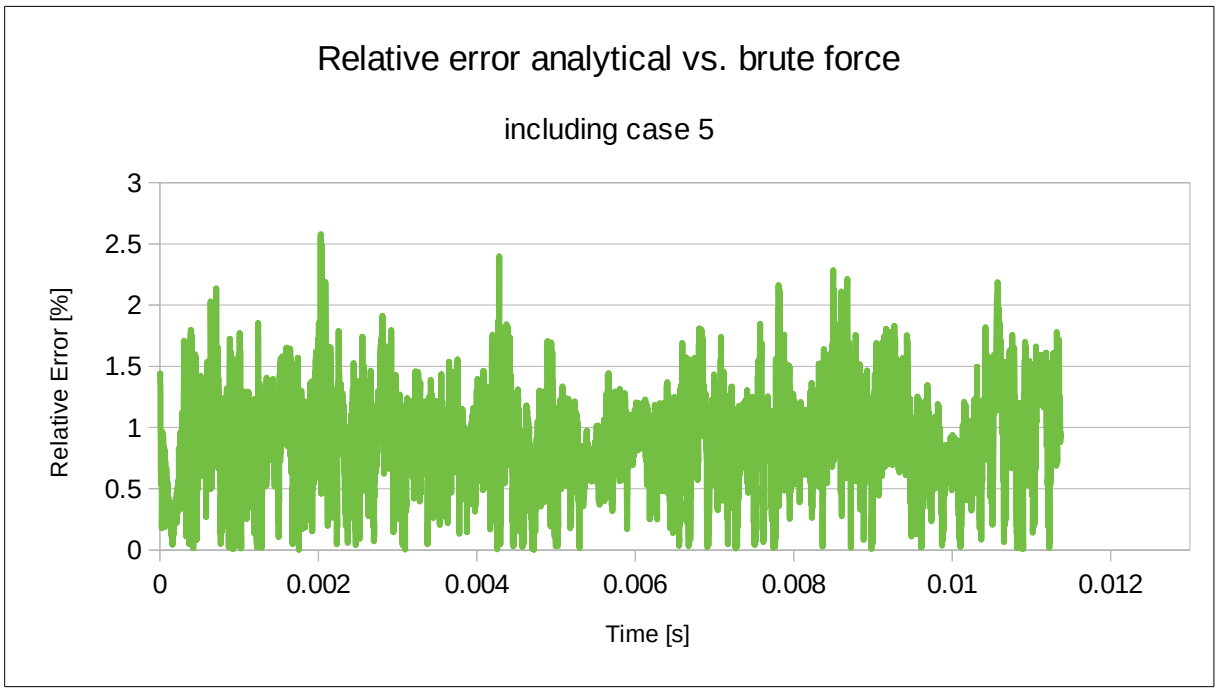


Figure 25: Relative error between the two approaches

The slightly higher relative error is caused by the very small size of the small particle compared to the cell size and thus the resolution of the sub-cell grid is not sufficient and could be increased to get more accurate results.

2.5.3 Known limitations and possible improvements

Further development of this algorithm should be considered if simulations with high powder mass rates and/or large cells need to be carried out. Specific areas, where to gain likely benefits for one's effort, include:

- optimize sequence of case selection and eliminate redundant calculation in order to increase the efficiency of the program
- the most accessible value for the cell area was obtained by calculating it from the volume of said cell by using following equation:

$$A_{cell} = \left(\sqrt[3]{V_{cell}}\right)^2$$

Equation 25: Calculation of cell area for eta calculation

This assumption however yields a significant error whenever the laser direction is not orthogonal to either of the side areas of cell. A possible, yet still not optimal approach would be to calculate the exact area normal to the photons' direction of travel which passes through the cell centre.

- The brute-force approach could benefit from a limitation on sub-cell grid side length in other directions so as to increase accuracy while using the same number of sub-cells.
- For a very large number of particles per cell a statistical approach to the area calculation could be considered to reduce computing time.
- The algorithm does not take into account the effect of particles in other cells (e.g. in cells between the current one and the laser).

2.6 Crystal growth orientation

Most metals in solid form are made up of microscopic crystalline structures. These structures define most of their physical, chemical and mechanical properties and thus are of paramount importance for nearly every application. In the case of high-end components being repaired by means of L-DED (i.e. turbine blades) the extreme operating conditions demand an extraordinary high strength and creep resistance at extreme temperatures. For this reason avoiding the presence of inter-granular boundaries, by ensuring that the microscopic structure is made of a single crystal, becomes an essential requirement for avoiding detrimental effects caused by the harsh environment experienced by the part during its life-cycle.[14] [15]

Unfortunately technological limitations on computational resources do not allow for a complete macroscopic simulation of those processes in an accurate and timely

manner because of the complexity and small scales involved. This leads to the need for simpler approaches.

One key aspect for determining the expected material properties after a L-DED process is the growth direction of the crystals, which is closely related to (and hence can be determined by using) the orientation of the temperature gradient at the solidification front.[16]

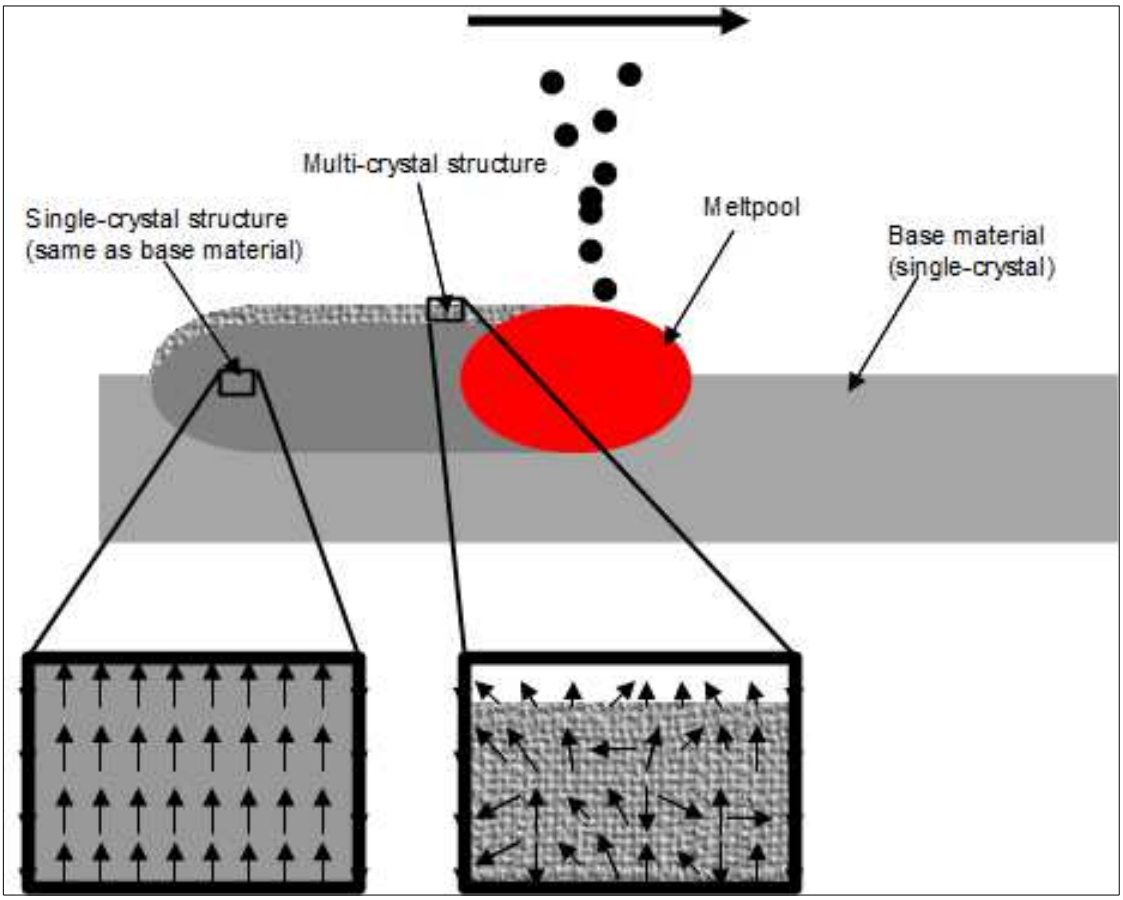


Figure 26: Crystalline structure of the weld bead during L-DED

2.6.1 Implementation

The normalized temperature gradient in each cell is multiplied by a floating point value between 0 and 1, representing the volume of re-solidified material per cell volume:

$$crystal\vec{O}rientation = \frac{\nabla(T)}{|\nabla(T)|} \cdot \frac{resolidifiedVolume}{cellVolume}$$

Equation 26: crystal growth orientation

This results in a vector field, which points along the direction of the temperature gradient right before the substrate/alloy solidifies out from the liquid phase.

2.6.2 Results assessment

The following picture illustrates the model setup for testing the crystal growth direction. At the beginning a spot in the centre of a solid steel plate gets heated to above the melting temperature by means of laser source.

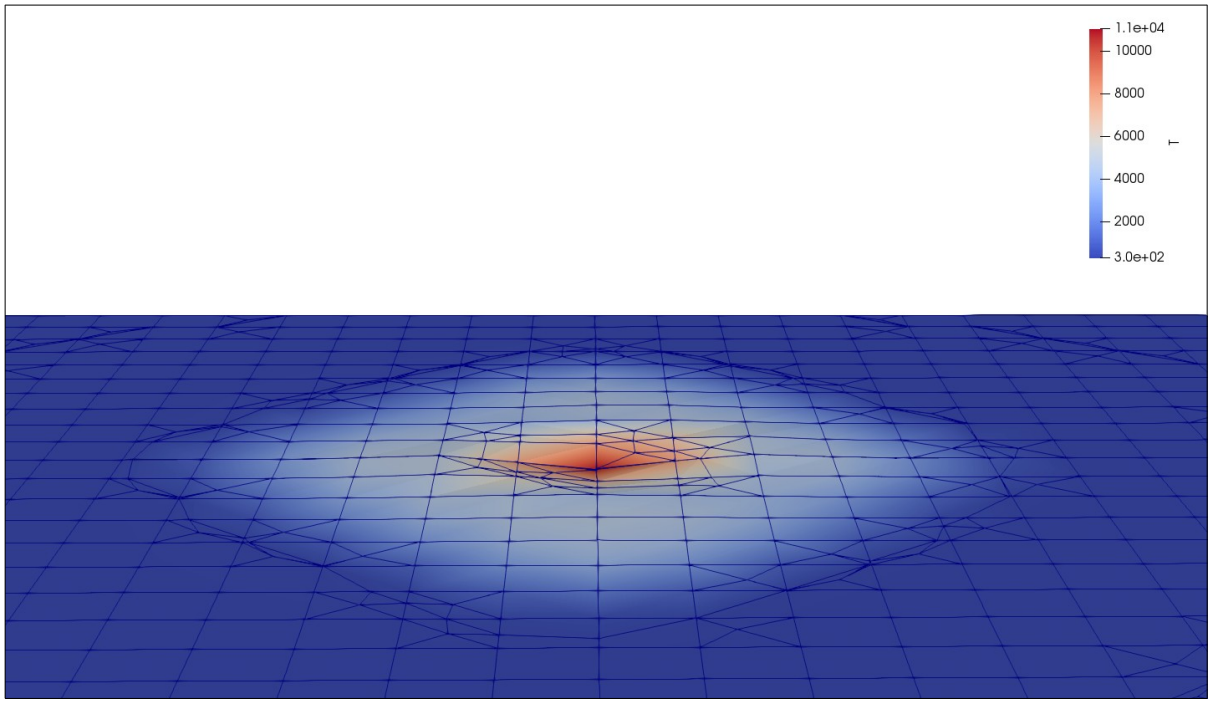


Figure 27: Simulation setup at the end of first part (when laser is turned off)

This results in a liquid phase where the temperature is high enough to allow melting of the substrate:

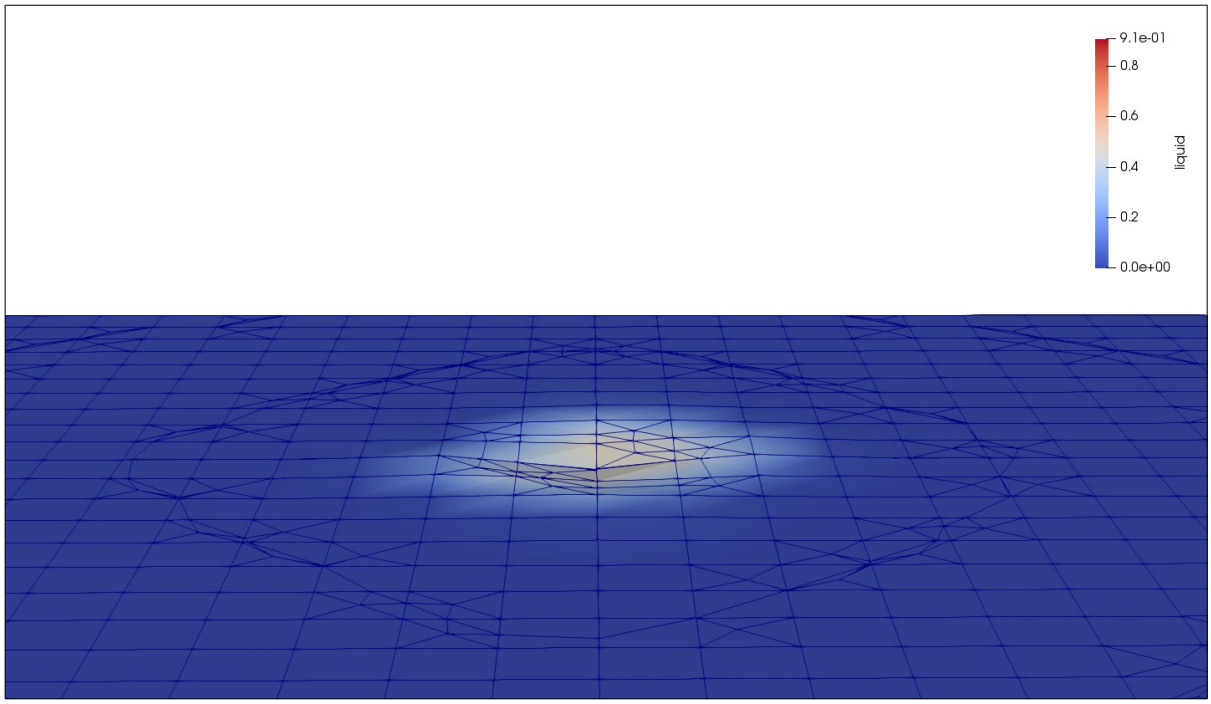


Figure 28: Liquid phase on steel plate

After letting the plate cool down until it reaches temperatures where the steel re-solidifies out of the melt pool. The orientation of the crystal growth can be visualized with vectors using the arrow glyph functionality built into the visualization tool ParaView [17]. In figure 29 the plate has been clipped on one side in order to better show the orientation vectors which are of course also present inside the plate.

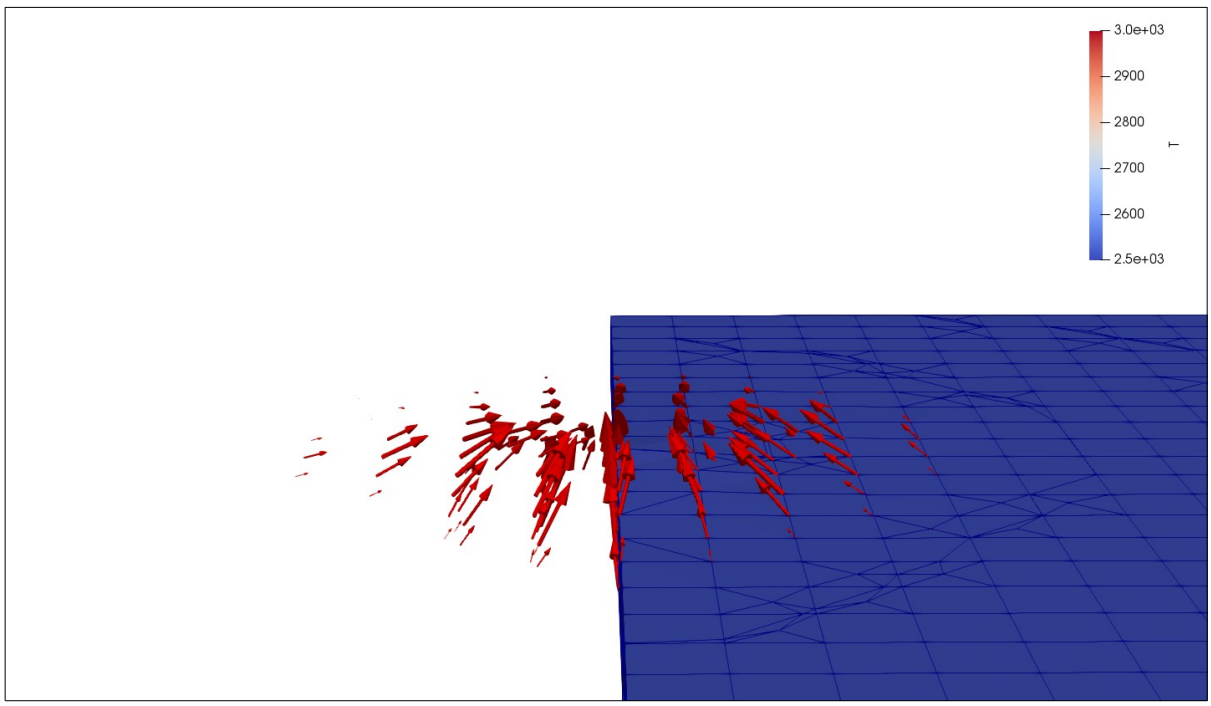


Figure 29: Orientation vectors of crystal growth direction

Figures 30 and 31 show the crystal growth direction inside the bead during L-DED simulations in a more advanced state.

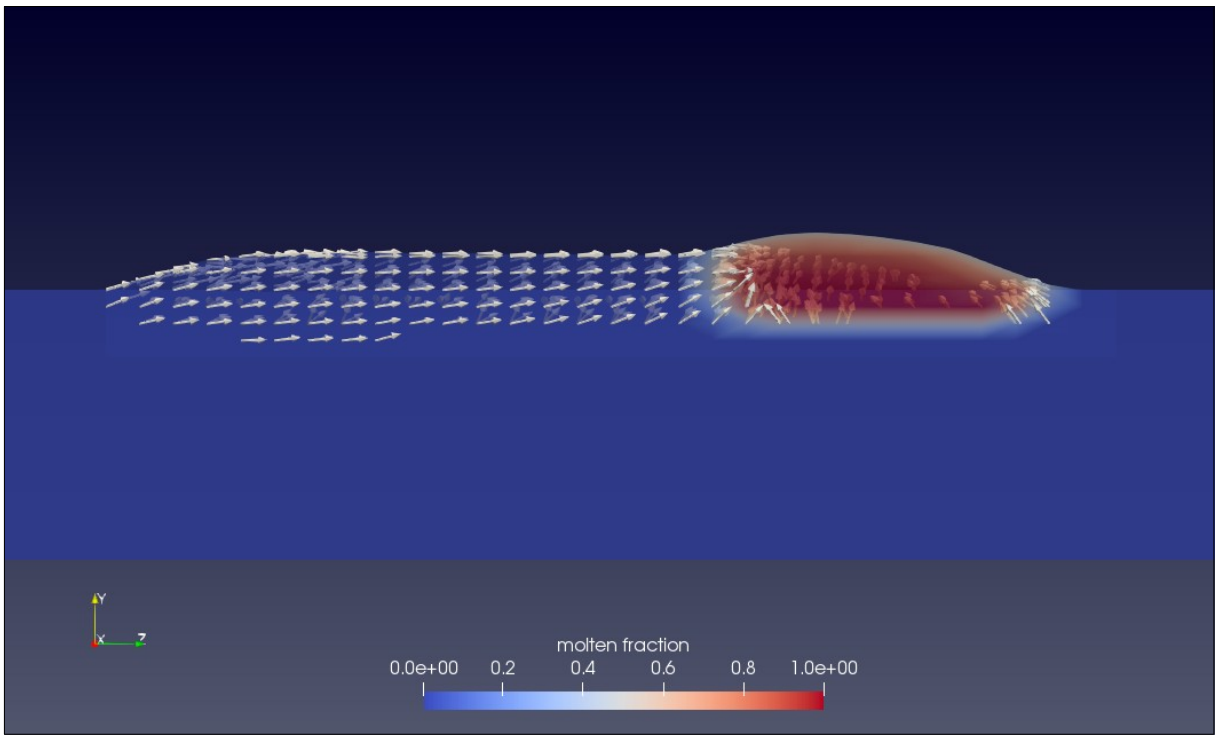


Figure 30: Crystal growth direction in bead during L-DED

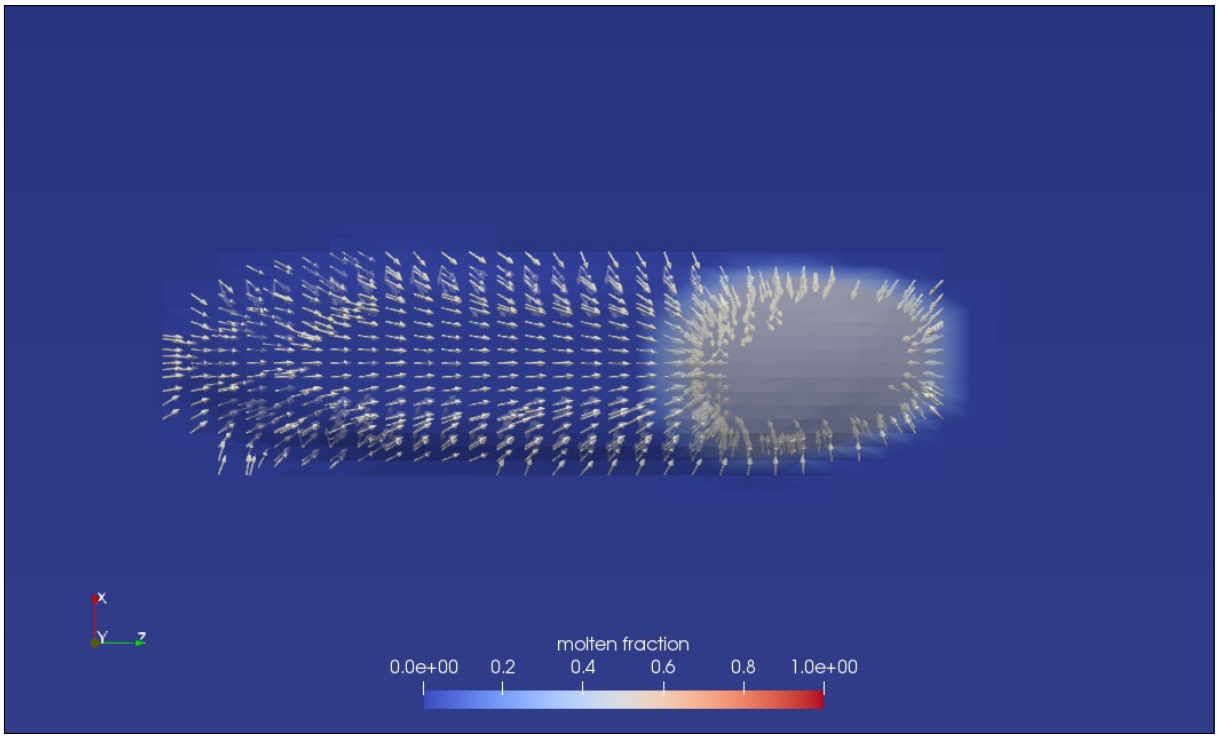


Figure 31: Top view of crystal growth direction in bead during L-DED

2.7 Solver performance improvement

The main bottleneck while simulating L-DED processes is often the updating of particle positions, especially when particle collisions (4-way coupling) are enabled. Even if said particles are no longer needed (negligible contribution to overall result) because they lie outside of the laser path and will never interact with it again. However if they still have a velocity magnitude larger than zero and have not left the simulation domain, they are updated by the program. This of course results in a lot of wasted computational resources.

So in the following section an approach will be presented to drastically speed up the simulations (depending on number of particles, geometry of the domain and general properties of the model) without compromising physical accuracy in the region of interest.

2.7.1 Implementation

The general idea behind this improvement is to predefine a geometrical region of interest inside the simulation domain (e.g. by using the “setFields”-utility of OpenFOAM) where all particles located inside of it are unconditionally updated by the algorithm at every time-step and those that happen to lie outside of it are to be removed from the simulation given a specific state (e.g. velocity under a certain threshold).

This is the most sensible approach for L-DED, because particles that come to lie outside of said field will not interact with the laser, not get molten and thus not be part of the final bead, but this field can be chosen to fit whatever the current modelling needs are.

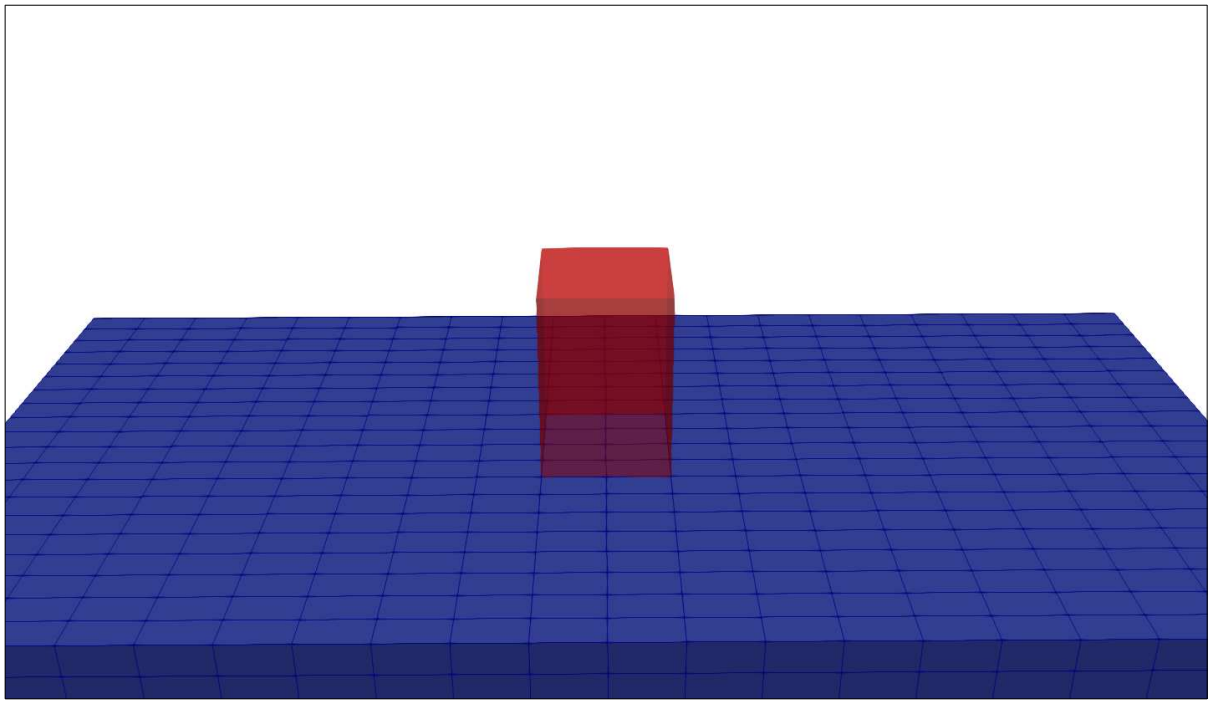


Figure 32: Region inside which particles are not deleted

As illustrated in figure 32, a region is placed in the centre of the steel plate and injection is started.

After it has been determined that the particle lies outside said region, the velocity is checked against a minimum velocity defined by the user. If the particle gets slower than that, it will get deleted. This part was added to keep particles which have the possibility of returning inside the laser's path due to their velocity. If said minimal velocity is set to a high value, a particle gets deleted instantly after leaving the region.

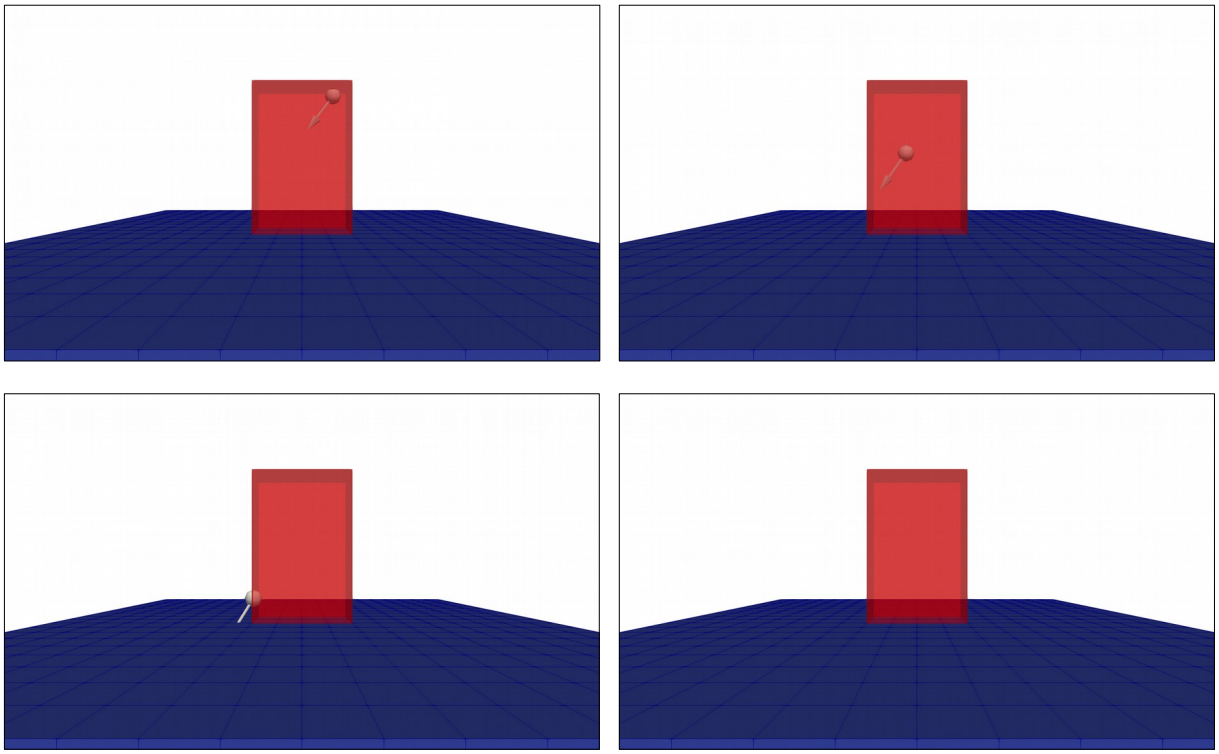


Figure 33: Particle deletion outside region of interest

2.7.2 Results assessment

Performance gains, while in many cases significant, are very difficult to quantify because of the wide variety of variables that play a role. A huge contribution is of course the sheer number of particles that have been injected. Other factors are the minimum velocity required (user given parameter), the size of the protected region and also the way particles are inserted into the simulation domain (injection in L-DED vs. powder bed generation as used for Laser Powder Bed Fusion (L-PBF)).

Figures 34 and 35 show a comparison of the same process, once without deleting the particles and once while deleting the particles after they leave the protected region and slow down to under 0.005 m/s.

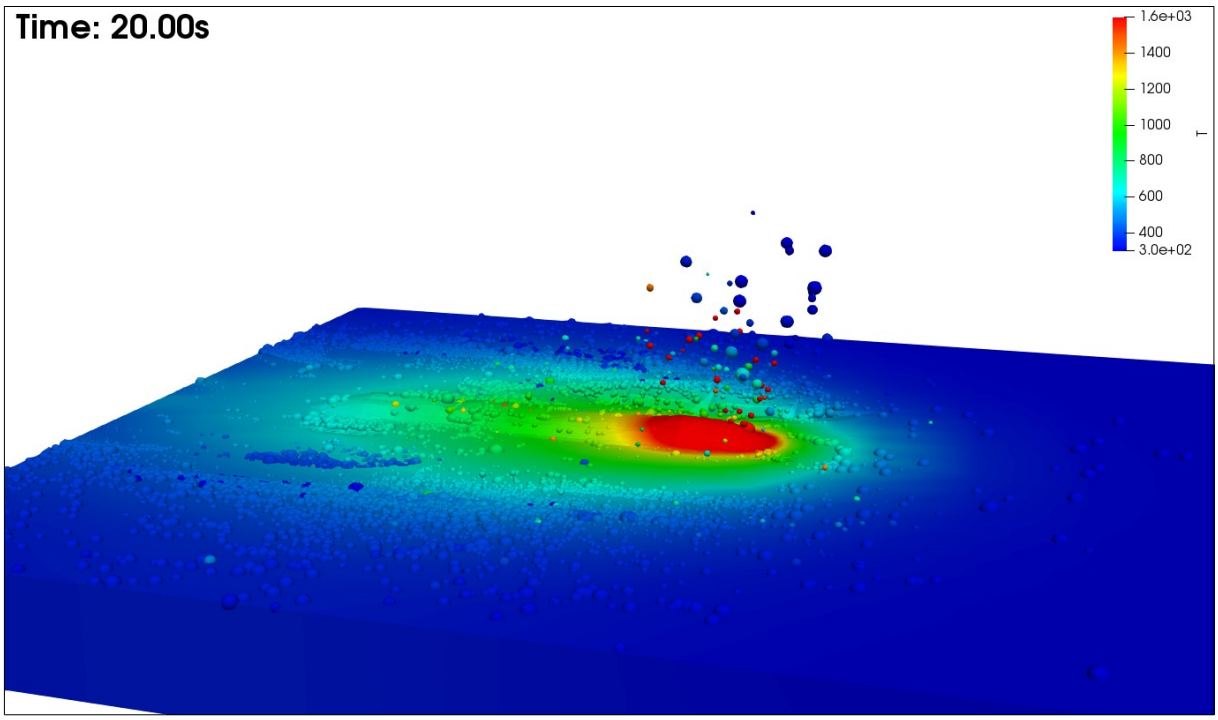


Figure 34: L-DED process without deleting particles

All of the particles lying on the surface of the plate still get updated. This uses a lot of computational power and thus requires more time to simulate.

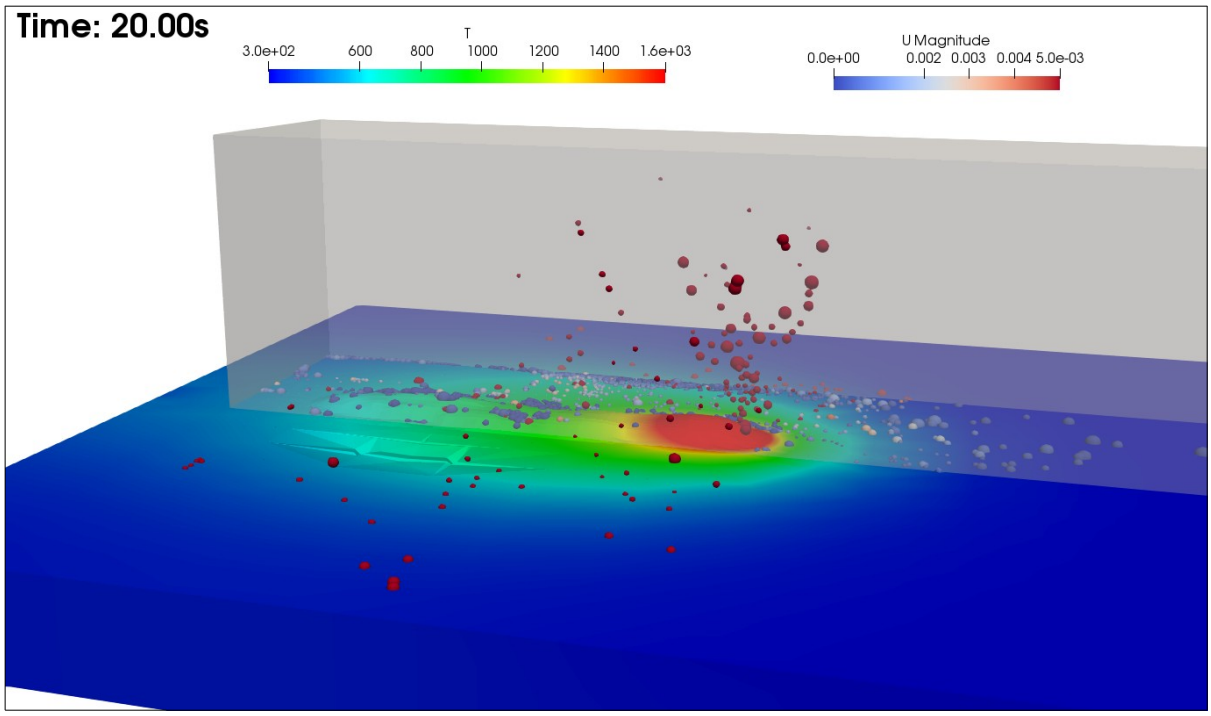


Figure 35: L-DED process with particle deletion

Here only particles that are still moving at a reasonable speed and/or lie inside the marked region of interest do still get updated at every time step. Both pictures were

taken at the same time in the process and thus represent the same state with the only variation being the deletion of particles in the second picture.

A possible argument for not deleting those particles is their heat capacity and thus a small increase in the temperature of the substrate could be expected. However due to the spherical nature of the powder particles, the contact area between them and the substrate is minimal, as is their heat exchange. Note that in the simulations no noticeable variation in weld bead geometry or thermal fields could be seen, which indicates that the effect of this tweak on the physical accuracy is minimal.

Figure 36 shows the simulation times for simulating those two cases and as can clearly be seen, the process with particle deletion finished much faster and also gained a bigger advantage the longer the simulation went on (the more particles were injected).

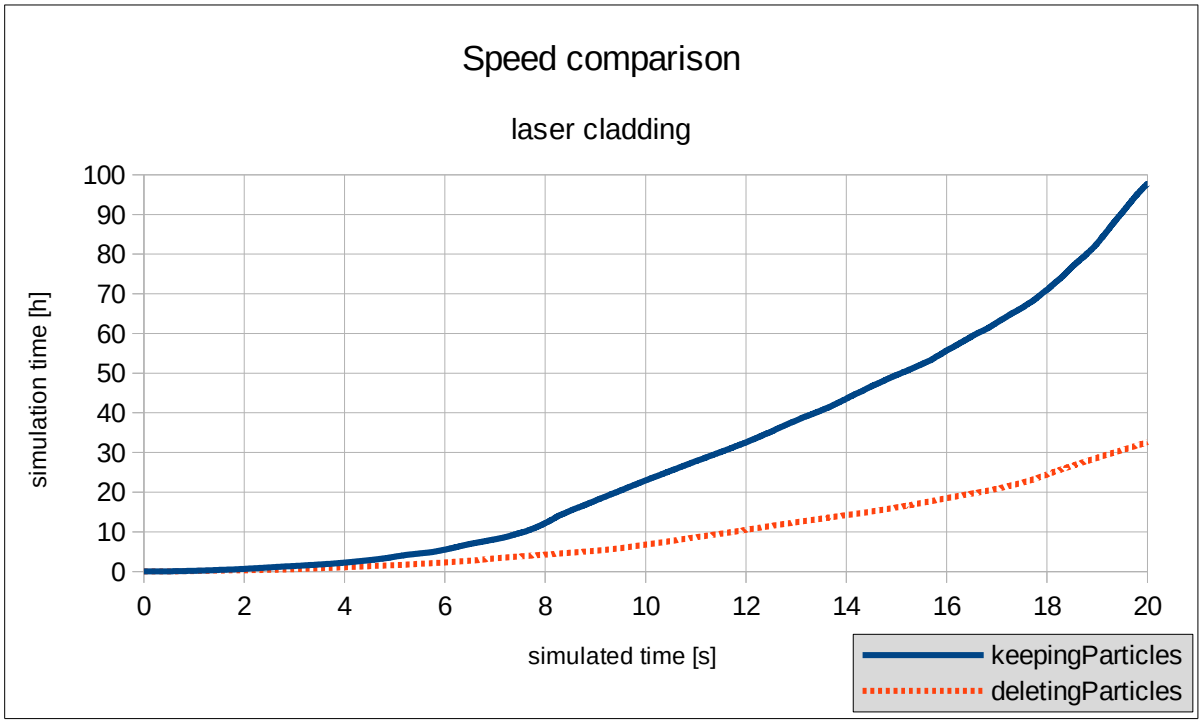


Figure 36: Simulation time comparison

2.7.3 Known limitations and possible improvements

With a proper selection of the region of interest the effect of this method on the physicality of the process can safely be considered to be negligible. Somewhat higher heat accumulation could be a negative consequence in L-DED, since the particles lying on the plate would absorb some heat (not much, due to small contact area of a sphere on a flat surface). It would be possible to completely eliminate this error, by either simply stopping the particle (so that it does not get updated) instead of deleting it or add a source term for the temperature equation at each of the

positions where the particles were deleted (again assuming a completely unmoving particle).

The effort and computational resources required to implement such an approach would probably not be worth the time required to do so and would further eliminate any gained performance benefits and possibly succeed previously needed processing power, especially on processes similar to DED, where the thermal effect of those particles is negligible.

3 Verification study

The main goal of all the improvements done to the solver code was to be able to get a better representation of the physical mechanisms during the L-DED process . To verify those improvements it was necessary to compare the simulation results to actual experimental data.

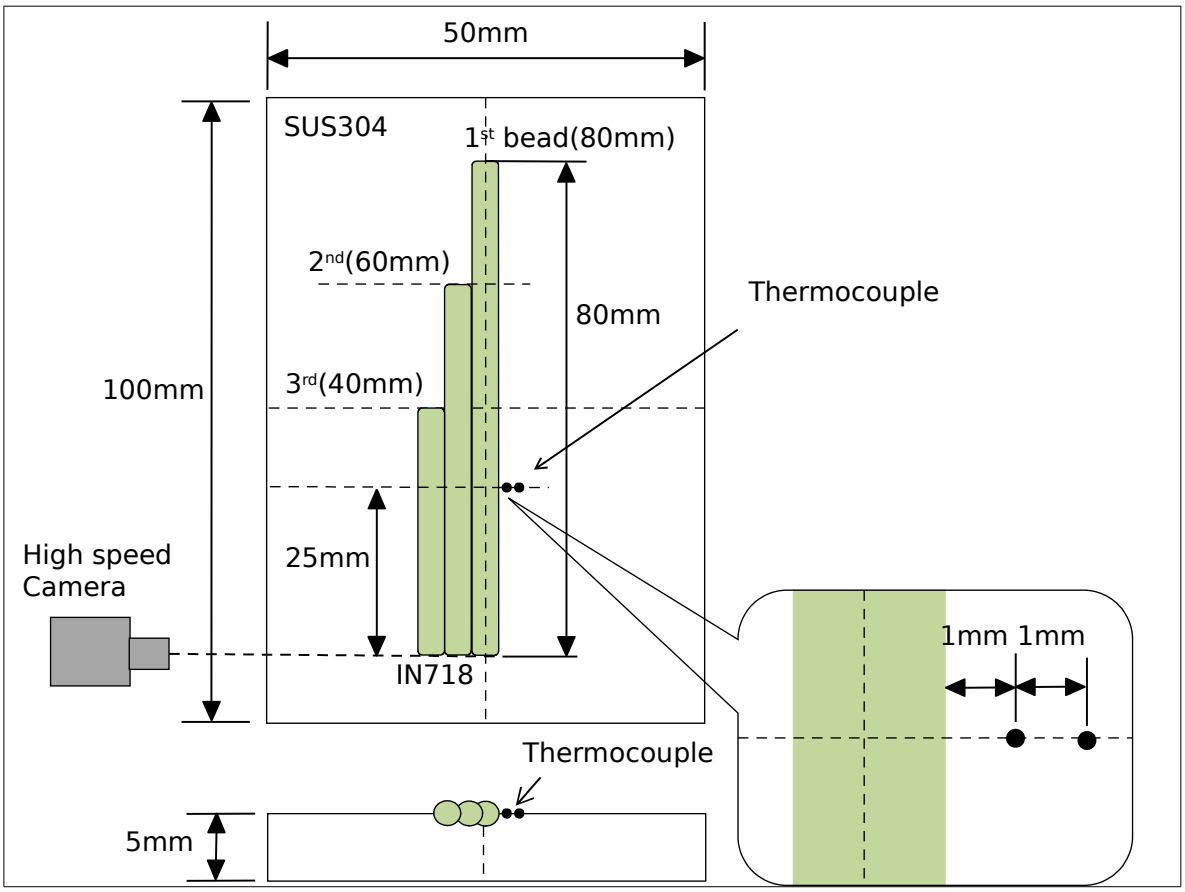


Figure 37: L-DED process setup schematic [22]

The experiments were carried out by Mitsubishi Heavy Industries, Ltd. and the resulting cross-sectional area measurements, the powder deposition efficiency and temperature plots were used as target values for checking the solver and calibrating the model. In this thesis, six different setups will be presented and compared to the simulation results. These represent multi-pass L-DED processes, where the laser and injector nozzle completed three passes side by side on a substrate with predefined length and offset from one pass to the next. Figure 37 shows the geometry of the experiments as well as measurement methods and locations which will be discussed further in the following section.

3.1 Experimental setup

The fundamental geometry is the same for all of the conditions. As can be seen from figure 37, the plate's dimensions are 100x50x5mm and is made of stainless steel AISI304 (X5CrNi18-10 by ISO-standard).

The laser is a 4kW fibre model manufactured by IPG Photonics, Ltd., while the injection of Inconel 718 powder is handled by a conical nozzle as seen in figure 38.

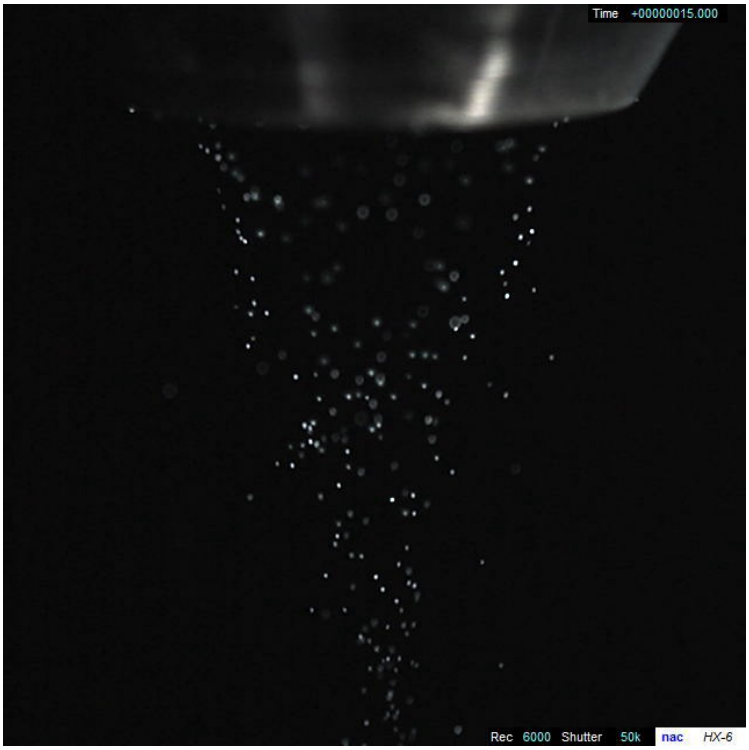


Figure 38: powder injection via conical nozzle[22]

The second common property of all conditions is the length of each individual pass, the first pass being the longest and the subsequent ones getting shorter.

The following table contains all of the relevant properties of the experimental setup, which are also used in the simulation:

Test designation	Laser power [W]	Nozzle & laser velocity [mm/min]	Focus position [mm]	Powder mass feed rate [g/min]	Shift amount after first pass [mm]	Shift amount after second pass [mm]
LA	1000	70	27	1.6	2	1.8
LB	750	70	27	1.6	1.7	1.8
LC	1250	70	27	1.6	2.5	2.3
LD	1000	100	27	1.6	2	1.8
LE	1000	70	27	1	2	1.8
LF	1000	70	27	2.2	2.3	1.8

Material (substrate)	Material (powder)	Gas type	Carrier gas (powder) [l/min]	Shielding gas [l/min]
SUS 304	Inconel 718	Argon	4	20

Injection nominal angle [deg]	Injection dispersion angle [deg]	Stand-off (nozzle) [mm]	Plate size (L x W x D) [mm]	Pass length [mm]
21 ± 4.4	4.4	12	100x50x5	See illustration

Figure 39: process parameters for the different conditions being studied

3.2 Simulation setup

The simulations were carried out using a smaller geometry, after it had been confirmed that heat accumulation would have a negligible effect when using the right boundary conditions. This was done in order to avoid the huge computational requirement, that would otherwise have been necessary for simulating six different cases multiple times with a large geometry and thus much greater amount of finite volume cells. The simulated substrate measured 40x20x5mm, while the length of each pass measured 20mm.

The other parameters of course were adopted without change to ensure physicality of the model.

To reproduce the experimental results as accurately as possible, the simulations must of course be set up with the right parameters. This attention to detail is of paramount importance for a good comparability and thus usefulness of the simulation model.

3.2.1 Powder size distribution

Information about the powder size distribution has been extracted from following graph, which was provided by Mitsubishi Heavy Industries, Ltd.

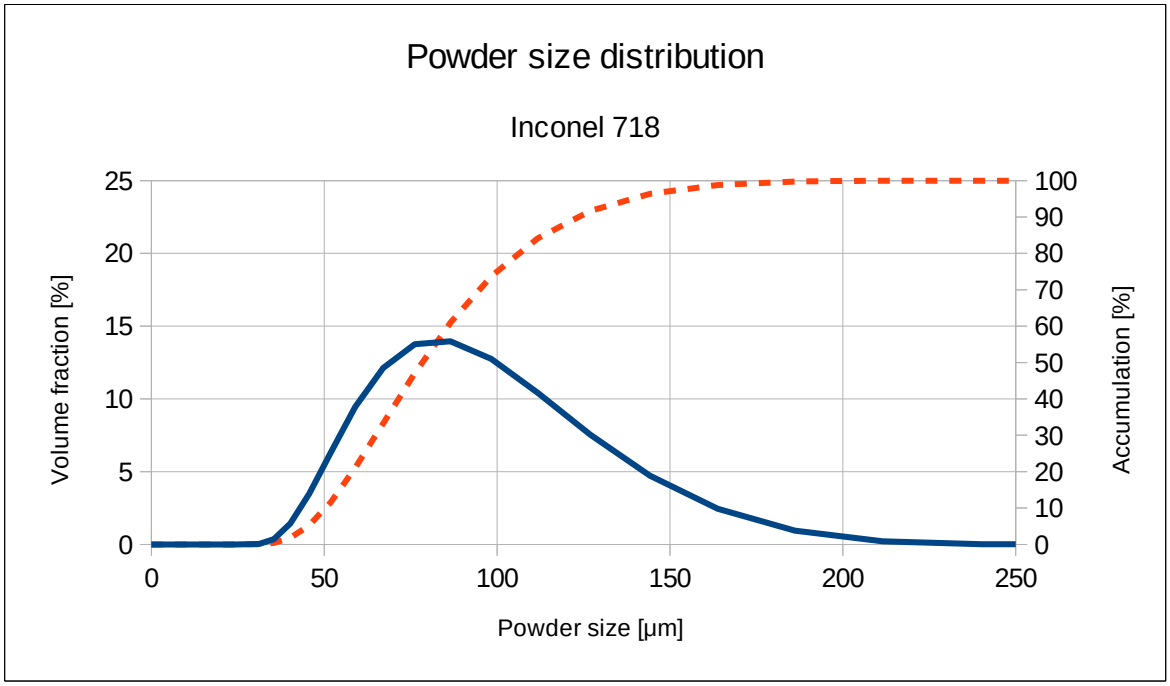


Figure 40: Powder size distribution used for the L-DED experiments. Courtesy of MHI, Ltd.

As can be seen from the graph in figure 40, it is a slightly skewed normal distribution. To simulate this in OpenFOAM, a model for general distribution was used, where the measurement points and respective volume fraction were input manually. Unfortunately the data behind the graph consists of only 19 valid discrete measurement points, which introduces a certain amount of uncertainty about the actual distribution.

3.2.2 Scripts for case creation

In order to simulate all the requested cases for the experimental verification quickly and autonomously, shell scripts in BASH language were prepared. By using them it was possible to automate the generation of different simulation scenarios for each of the 14 test cases (the verification part will only include 6 cases, where a lateral offset of the nozzle was tested). Using those scripts, it is possible to set the right geometry, material parameters, etc. once and then create new test cases based on a template model, while automatically altering values for e.g. laser power, injector and laser velocity, powder mass flow rate, etc. The corresponding code is provided in the appendix in separate sub-sections (namely LMD 1-project[2] (see section 6.3) and LMD 2-project[3] (see section 6.4)).

[2] internal name of laser metal deposition project in cooperation with MHI for single-pass-simulations
[3] internal name of laser metal deposition project in cooperation with MHI for multi-pass-simulations

3.2.3 Matlab file for internal probes

OpenFOAM offers two different utilities to gather information at specific locations in the simulation domain. Probes can be used if one is interested in the non-interpolated cell centre values, otherwise so called internal probes should be used.

For the L-DED simulations, there is a need to measure temperature at specific locations equivalent to the experiments. However those locations might lie exactly between two cells (maybe of different sizes) because of the dynamic refinement of the mesh at run-time. For this reason normal probes may not yield the right results, if they only extract cell values. This may cause a low temperature measurement because of a larger cell whose centre extends much further away from the laser than the actual desired measurement point.

Internal probes solve this problem by interpolating between neighbouring cell values. Unfortunately when using internal probes, the results are saved into one folder for each time step. For longer simulations plotting temperature over time becomes a considerable effort, so there was a need for automating this task. In order to achieve a .csv-file with all the necessary data in a timely and non-invasive manner, a Matlab script was created to merge all the output information into one file. This script can also be found in the appendix (see section 6.5).

The script should work without changes if copied into the working directory of an OpenFOAM case with internal probes which measure temperature and has to be changed slightly if trying to measure other parameters different from temperature “T”.

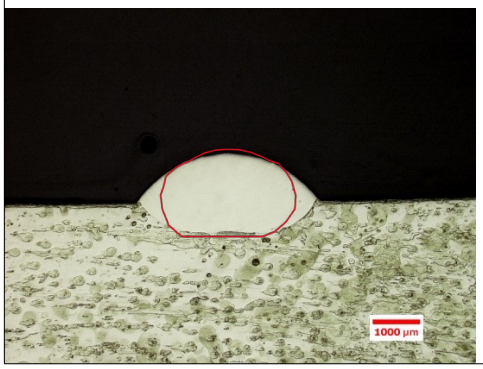
3.3 L-DED verification study

The following pages will contain all the studied comparisons of cross-sections and powder deposition efficiencies as well as an exemplary thermal cycle plot for the design condition LA.

LA:

The following figures are taken after each new pass and show the experimental cross-section on the left, the simulated one on the right and contain a table with a comparison of the measurements, as well as information about the powder deposition efficiency.

	Experimental	Simulation
Bead Height [mm]	1.05	1.0
Bead Width [mm]	3.8	3.0
Penetration depth [mm]	0.62	0.5
Powder deposition efficiency [%]	87.6	86.3



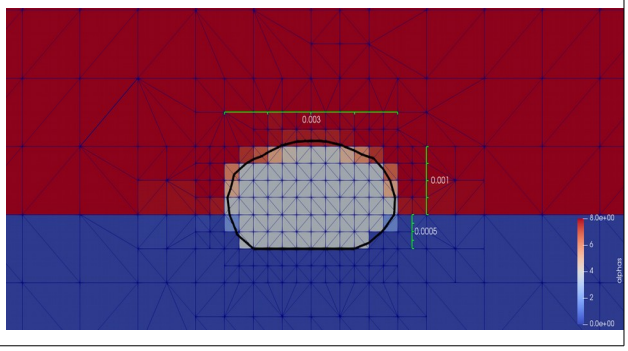
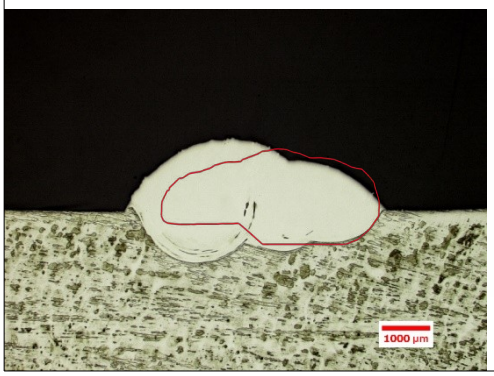


Figure 41: LA 1ST pass

	Experimental	Simulation
Bead Height [mm]	1.45	1.2
Bead Width [mm]	5.36	4.55
Penetration depth [mm]	1.1	0.5
Powder deposition efficiency [%]	87.6	76



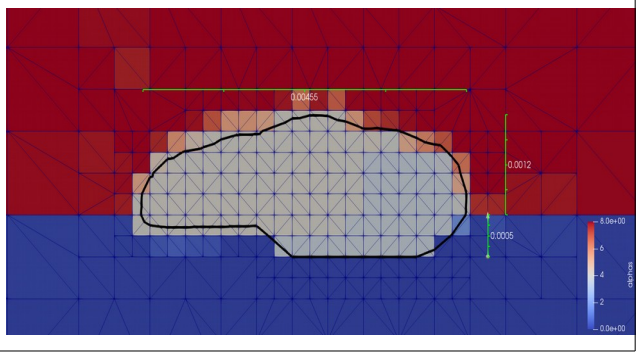
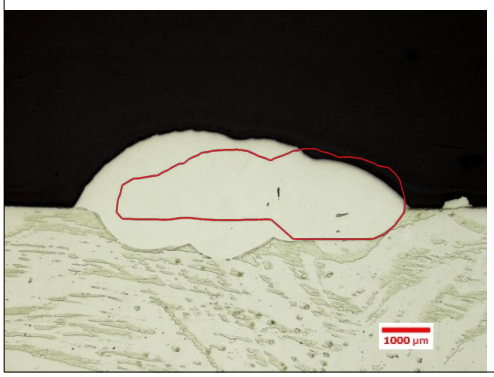


Figure 42: LA 2ND pass

	Experimental	Simulation
Bead Height [mm]	1.57	1.2
Bead Width [mm]	7.02	6.3
Penetration depth [mm]	0.86	0.5
Powder deposition efficiency [%]	87.6	73.2



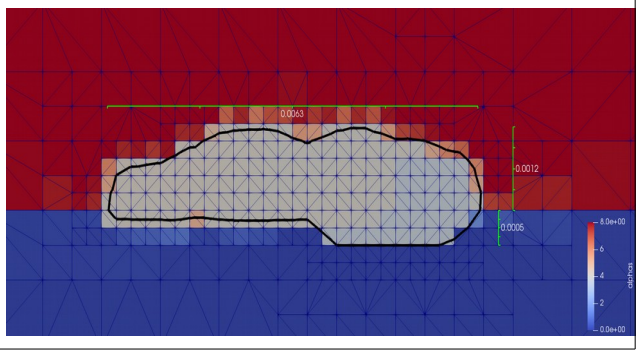
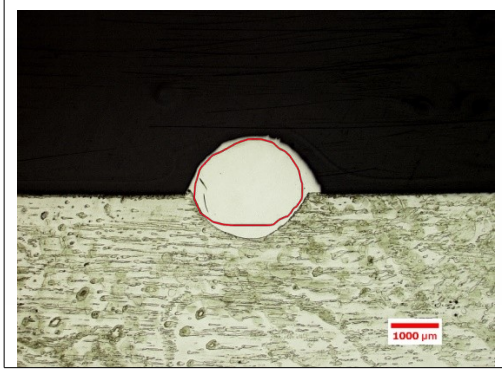


Figure 43: LA 3RD pass

LB:

Starting from the design condition LA, only the laser power was decreased to 750 W.

	Experimental	Simulation
Bead Height [mm]	1.24	1.0
Bead Width [mm]	2.76	2.2
Penetration depth [mm]	0.88	0.5
Powder deposition efficiency [%]	82.7	76.6



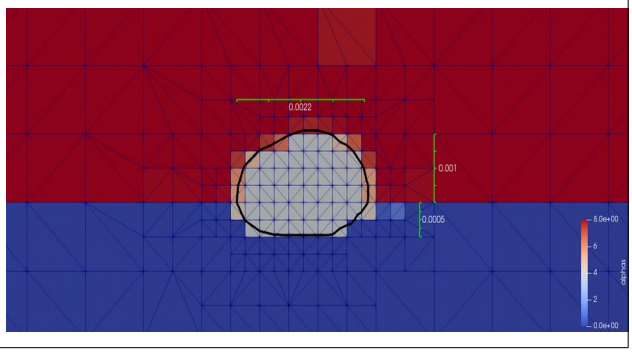


Figure 44: LB 1ST pass

	Experimental	Simulation
Bead Height [mm]	1.65	1.3
Bead Width [mm]	4.26	3.8
Penetration depth [mm]	0.64	0.5
Powder deposition efficiency [%]	82.7	71

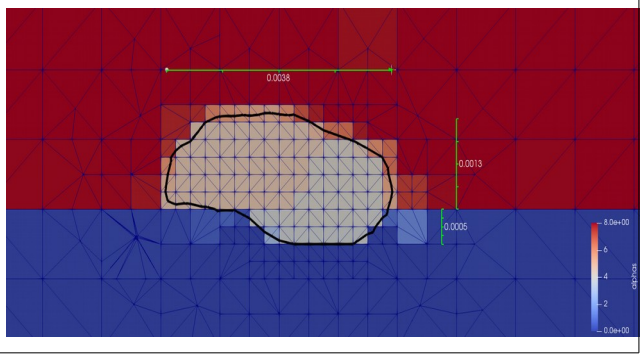
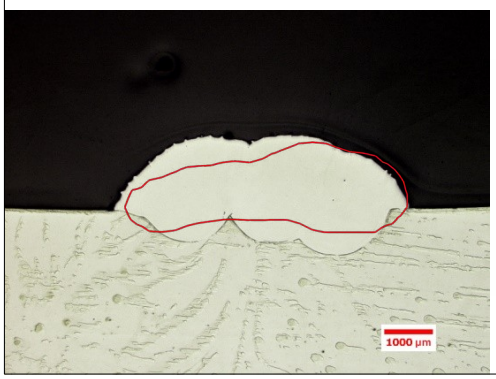


Figure 45: LB 2ND pass

	Experimental	Simulation
Bead Height [mm]	1.49	1.2
Bead Width [mm]	6.11	5.65
Penetration depth [mm]	0.70	0.5
Powder deposition efficiency [%]	82.7	67.8



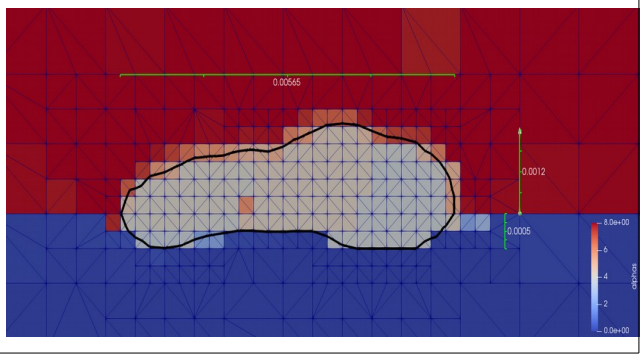


Figure 46: LB 3RD pass

LC:

Starting from the design condition LA, only the laser power was increased to 1250 W.

	Experimental	Simulation
Bead Height [mm]	0.92	0.75
Bead Width [mm]	4.46	3.5
Penetration depth [mm]	0.75	0.75
Powder deposition efficiency [%]	91.5	89.7

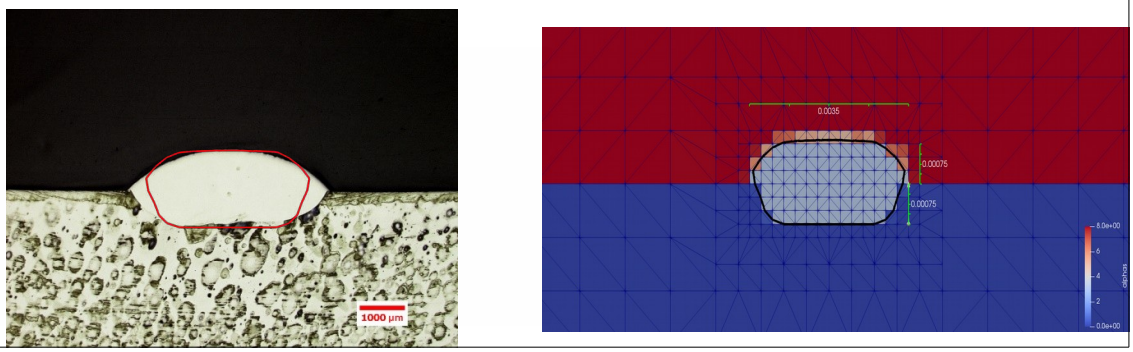


Figure 47: LC 1ST pass

	Experimental	Simulation
Bead Height [mm]	1.24	1.25
Bead Width [mm]	6.45	5.35
Penetration depth [mm]	0.73	0.75
Powder deposition efficiency [%]	91.5	83.7

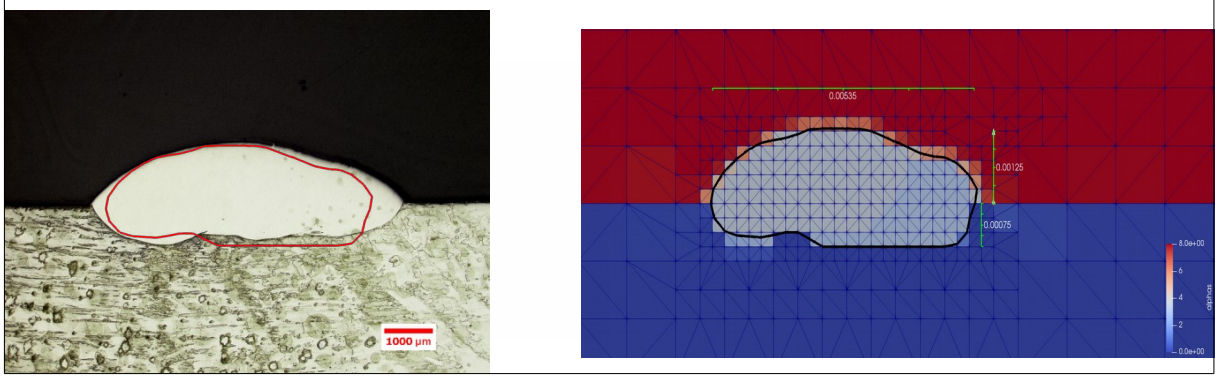
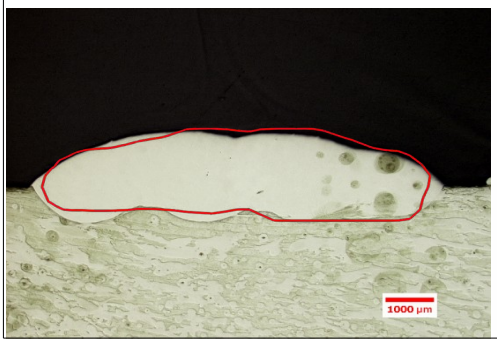


Figure 48: LC 2ND pass

	Experimental	Simulation
Bead Height [mm]	1.20	1.25
Bead Width [mm]	8.60	8.0
Penetration depth [mm]	0.81	0.75
Powder deposition efficiency [%]	91.5	82



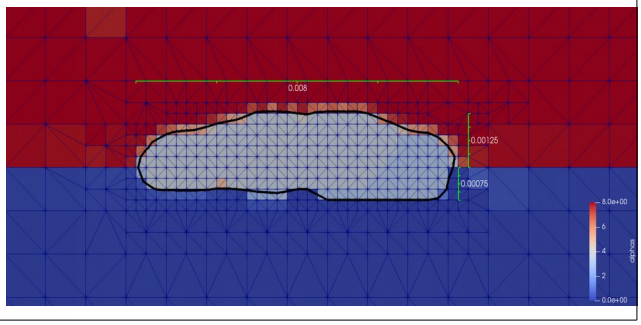
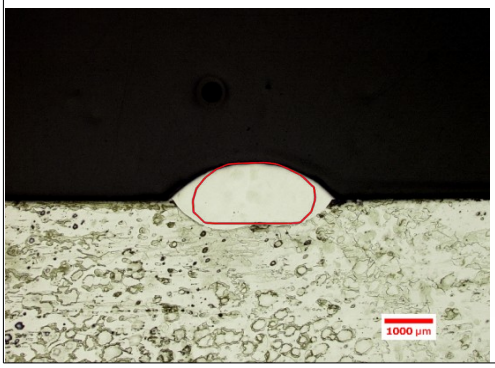


Figure 49: LC 3RD pass

LD:

Starting from the design condition LA, only the velocity was increased to 100 mm/min.

	Experimental	Simulation
Bead Height [mm]	0.82	0.75
Bead Width [mm]	3.46	2.5
Penetration depth [mm]	0.61	0.5
Powder deposition efficiency [%]	85.8	82.4



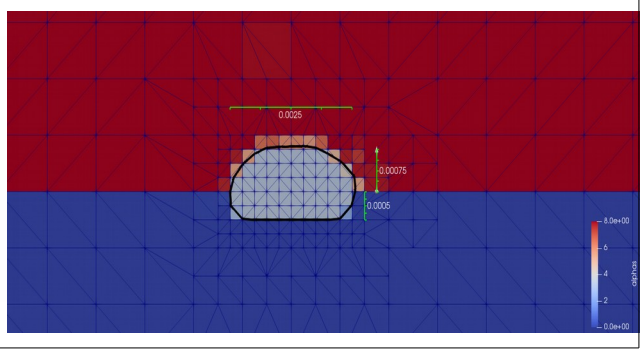
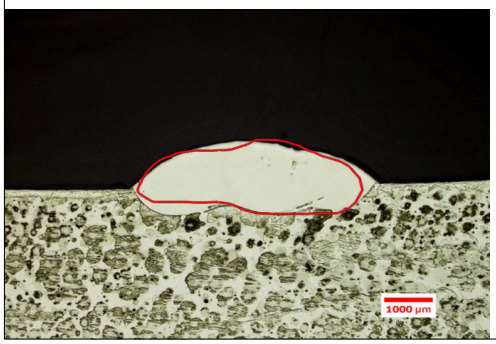


Figure 50: LD 1ST pass

	Experimental	Simulation
Bead Height [mm]	1.03	1.05
Bead Width [mm]	5.08	4.5
Penetration depth [mm]	0.63	0.5
Powder deposition efficiency [%]	85.8	74



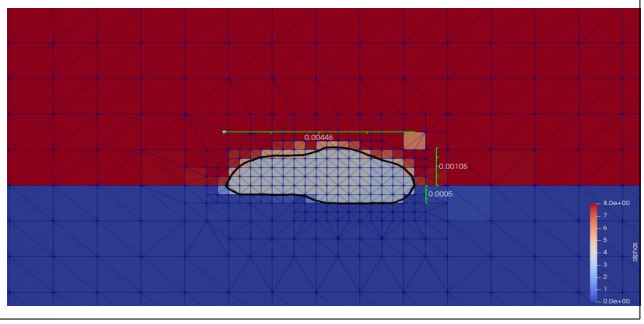
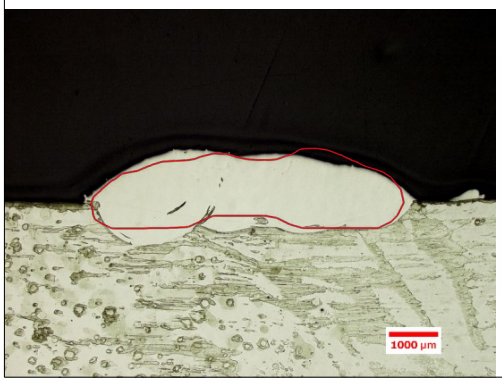


Figure 51: LD 2ND pass

	Experimental	Simulation
Bead Height [mm]	1.10	1.05
Bead Width [mm]	6.79	6.15
Penetration depth [mm]	0.84	0.5
Powder deposition efficiency [%]	85.8	73



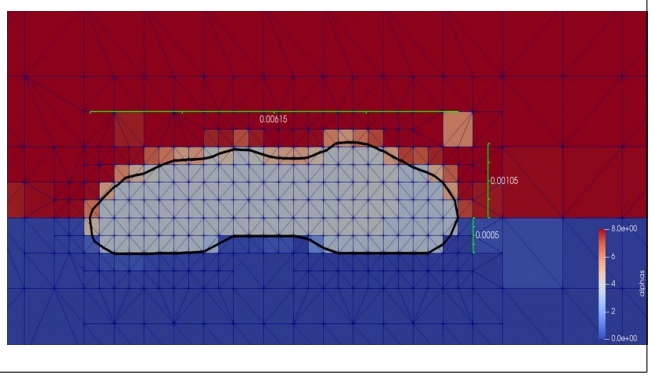
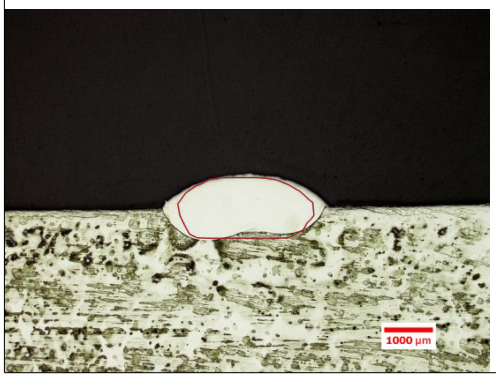


Figure 52: LD 3RD pass

LE:

Starting from the design condition LA, only the powder mass feed rate was decreased to 1 g/min.

	Experimental	Simulation
Bead Height [mm]	0.72	0.6
Bead Width [mm]	3.47	3.0
Penetration depth [mm]	0.64	0.75
Powder deposition efficiency [%]	86.2	78.1



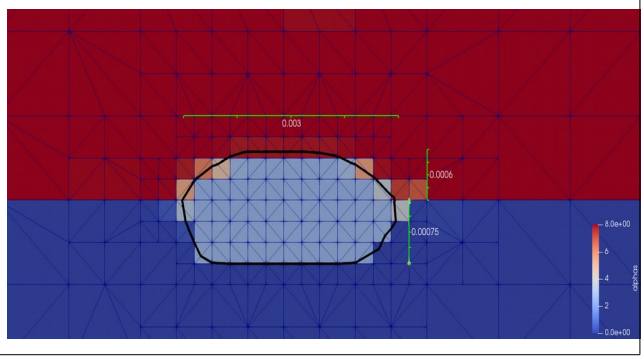
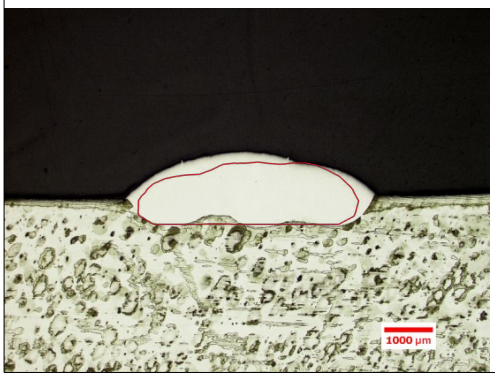


Figure 53: LE 1ST pass

	Experimental	Simulation
Bead Height [mm]	0.98	0.8
Bead Width [mm]	5.22	4.7
Penetration depth [mm]	0.54	0.5
Powder deposition efficiency [%]	86.2	72



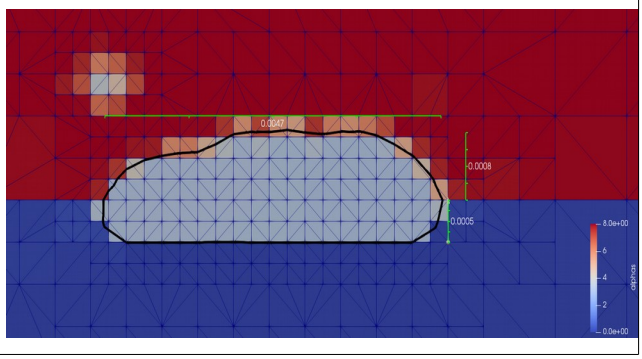


Figure 54: LE 2ND pass

	Experimental	Simulation
Bead Height [mm]	0.92	1.0
Bead Width [mm]	6.93	6.3
Penetration depth [mm]	0.51	0.5
Powder deposition efficiency [%]	86.2	71.6

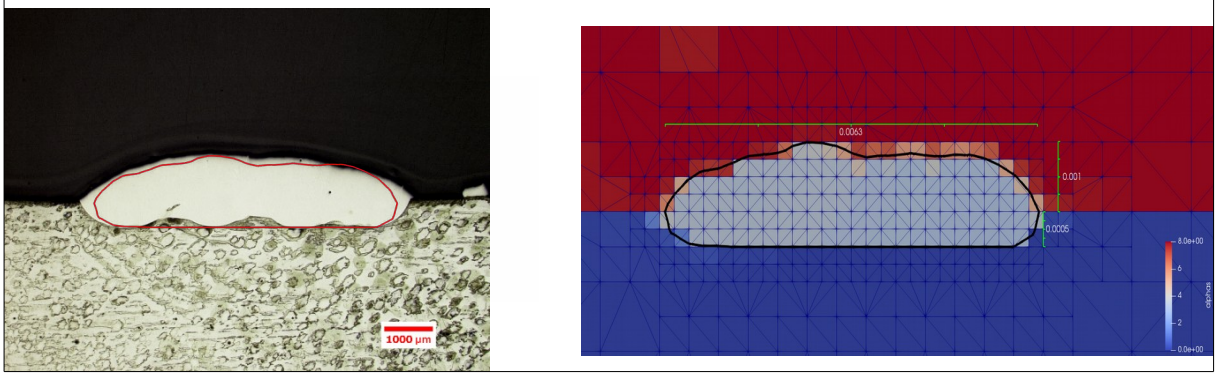


Figure 55: LE 3RD pass

LF:

Starting from the design condition LA, only the powder mass feed rate was increased to 2.2 g/min.

	Experimental	Simulation
Bead Height [mm]	1.37	1.25
Bead Width [mm]	3.40	3.5
Penetration depth [mm]	1.09	0.5
Powder deposition efficiency [%]	83	82

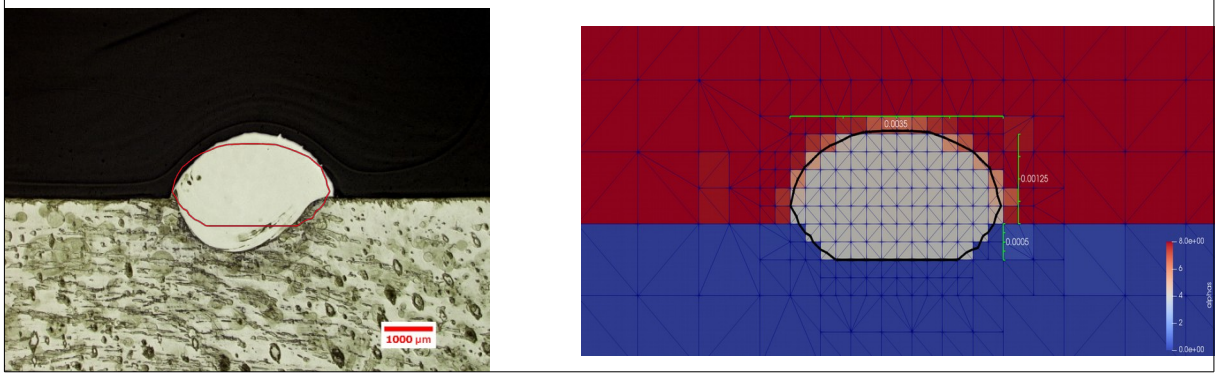
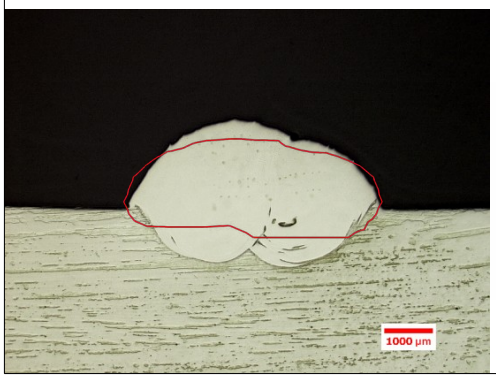


Figure 56: LF 1ST pass

	Experimental	Simulation
Bead Height [mm]	1.85	1.5
Bead Width [mm]	5.24	5.5
Penetration depth [mm]	1.09	0.5
Powder deposition efficiency [%]	83	78.8



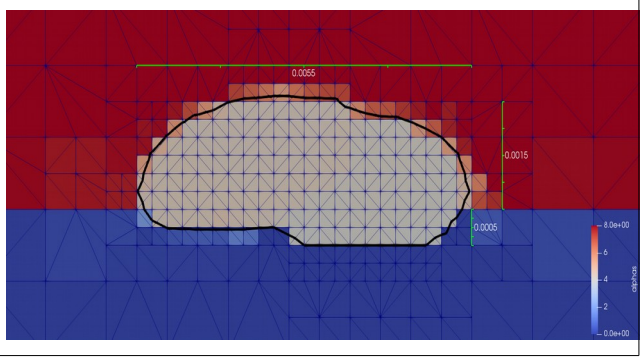
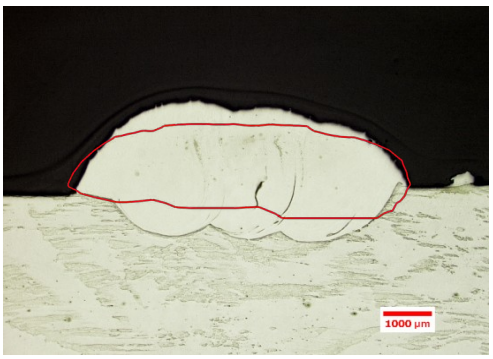


Figure 57: LF 2ND pass

	Experimental	Simulation
Bead Height [mm]	1.94	1.6
Bead Width [mm]	6.91	7.15
Penetration depth [mm]	0.91	0.5
Powder deposition efficiency [%]	83	75.5



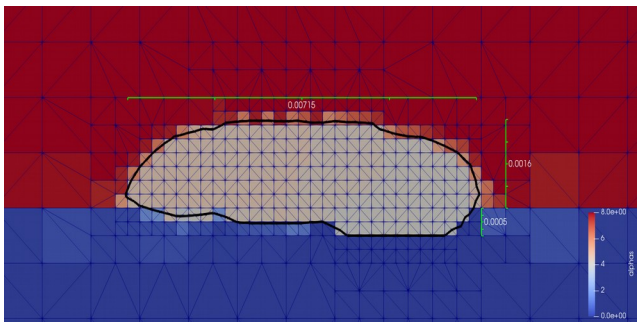


Figure 58: LF 3RD pass

Thermal cycle for condition LA:

This section will contain the temperature plots for each pass of LA. The experimental data will be split and for each pass there is a new plot, while the simulation data was combined into one figure.

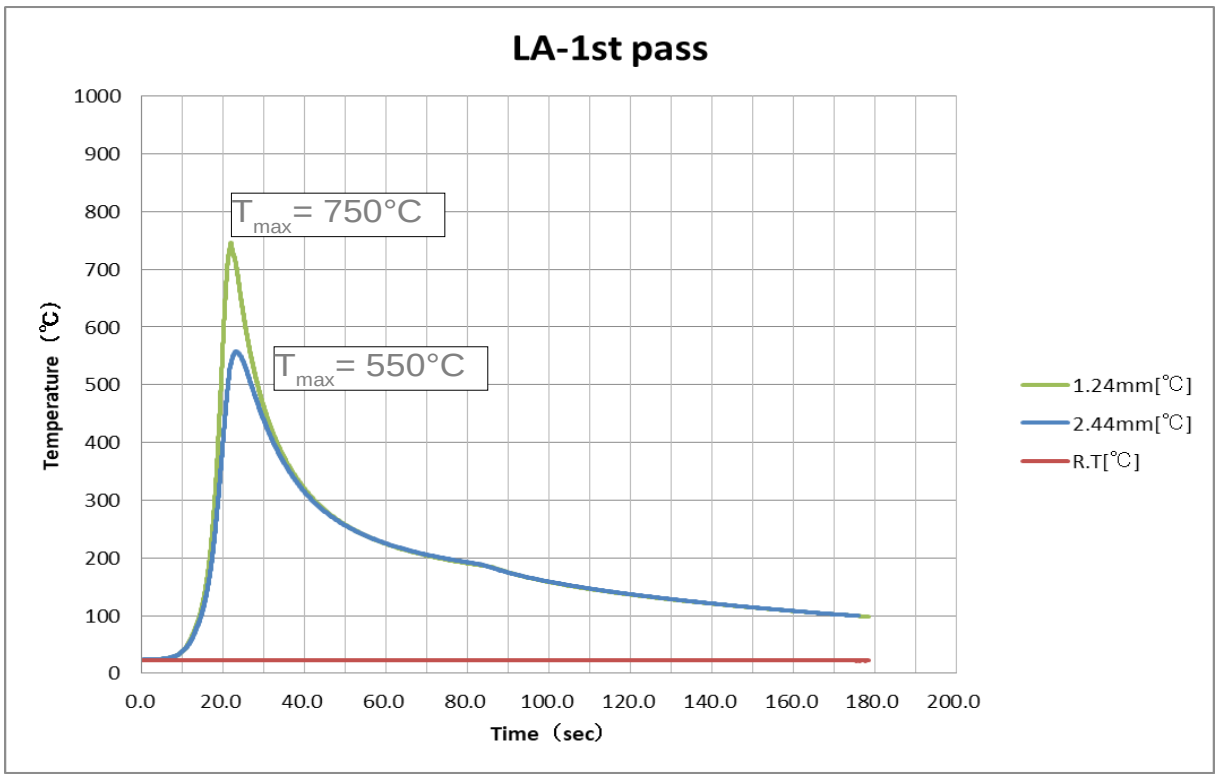


Figure 59: Thermal cycle for the 1ST pass of LA (experimental data)

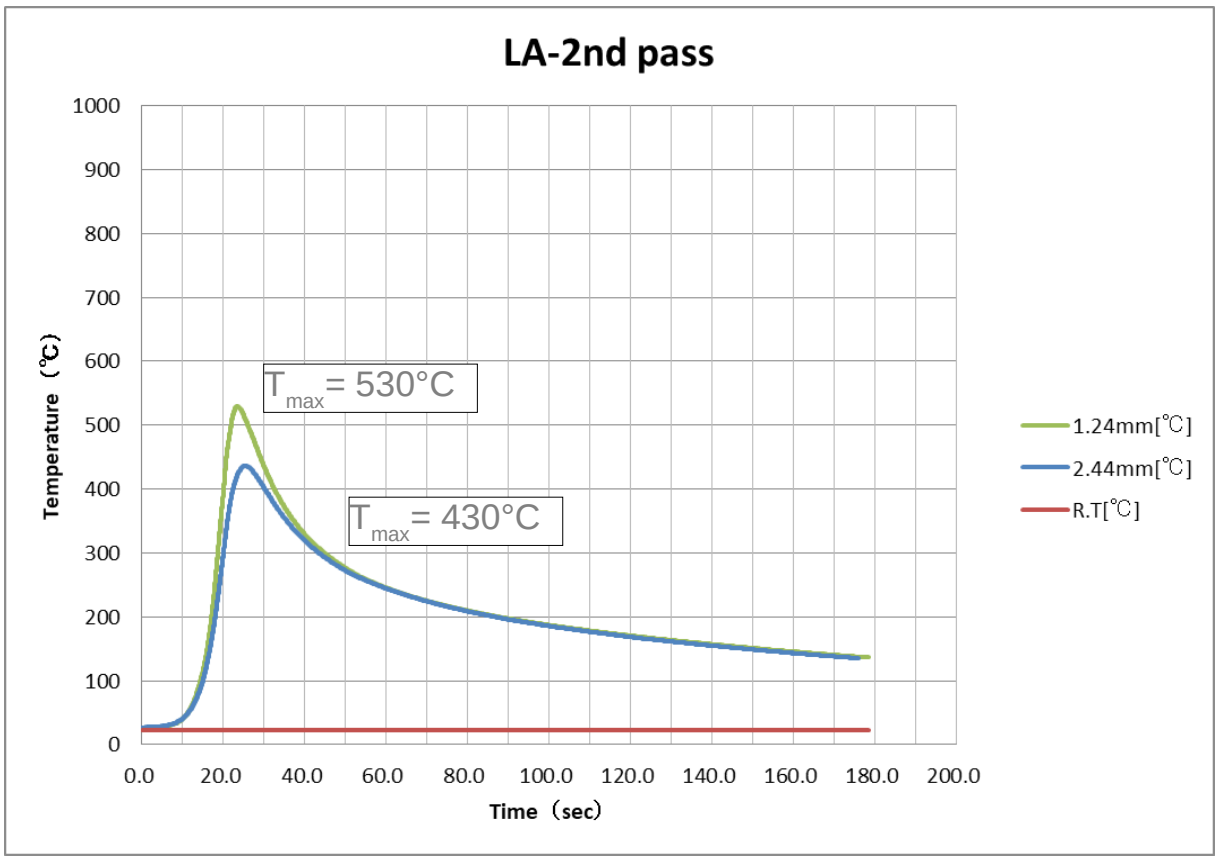


Figure 60: Thermal cycle for the 2ND pass of LA (experimental data)

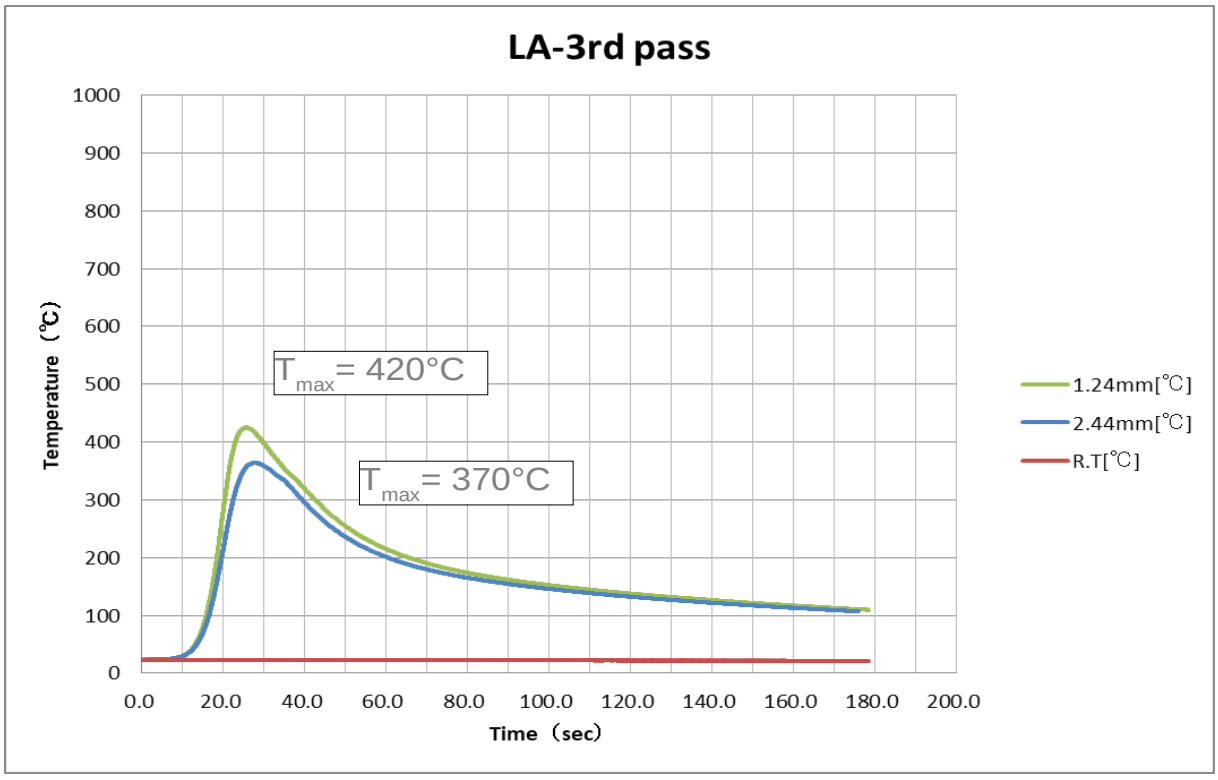


Figure 61: Thermal cycle for the 3RD pass of LA (experimental data)

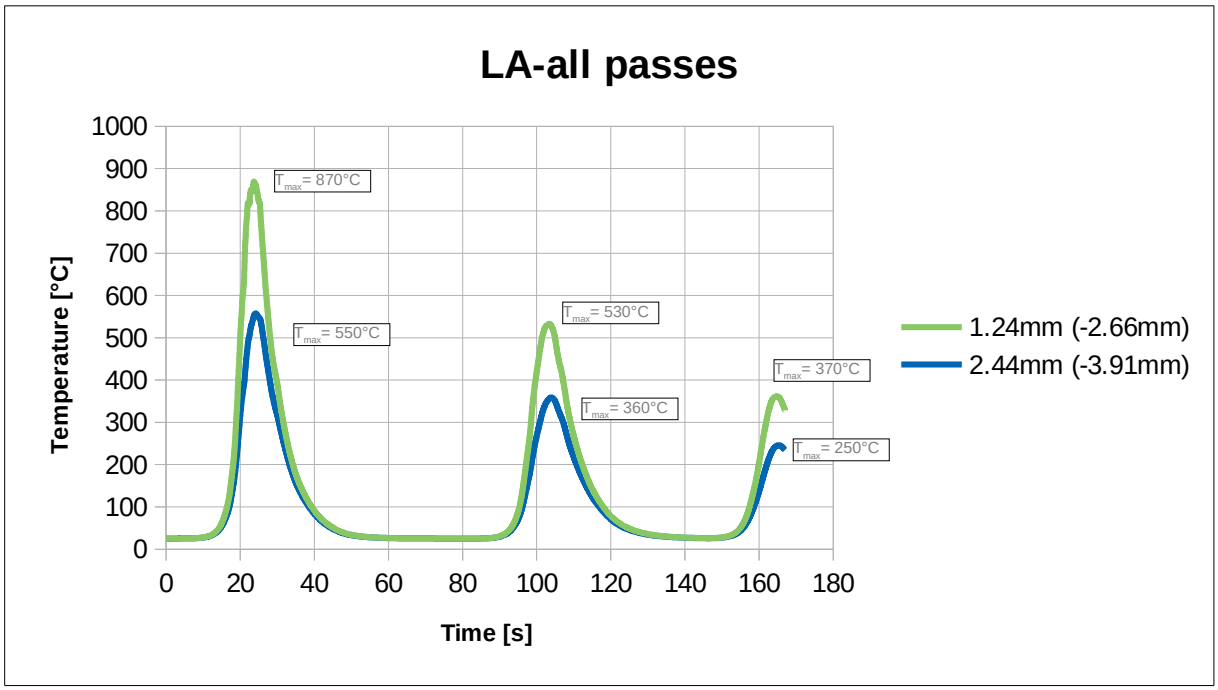


Figure 62: Thermal cycles for all passes of LA (simulated result)

3.3.1 Results assessment

In this section the simulation results will be presented and the cross-sections and powder deposition efficiencies (see equation 27) compared to the experimental results for each individual case.

$$powderDepositionEfficiency = \frac{weightPlate_{tEnd} - weightPlate_{tStart}}{injectedMass}$$

Equation 27: Calculation of powder deposition efficiency

As can be seen from the results, the cross-section of the first pass was simulated with reasonably high accuracy for all scenarios, while the subsequent ones show a significant discrepancy between experimental results (on the left hand side of the pictures) and the simulation (right hand side). This is mainly due to the difficulty of tuning parameters such as surface energy and absorptivity. These vary highly with temperature and mixture (steel and Inconel) and thus are very difficult to calibrate. Furthermore the values for surface energy are rarely experimentally measured for different materials, especially for high temperature applications.

The simulation uses an averaging approach between the phases to obtain a surface energy of the mixture, but this may not be the most accurate solution, mainly due to the fact that melt pools usually are coated by a thin oxide layer which of course radically changes the values of surface energy and thus can produce very different

shapes of cross-sections, mainly due to the so called Marangoni-effect. It states, that the fluid on the surface is transferred to areas of higher surface tension and thus can induce currents in the weld bead[18]. The direction of these currents probably caused the vastly different shapes between e.g. the experimental results of cases LA and LB after the first pass:

Figure 63: Flatter shape in case LA

Figure 64: Deeper weld bead penetration in case LB

In the longitudinal section the effect of the temperature and mixture gradient in the weld bead becomes apparent as at some points the liquid phase seems to be “pulled” backwards towards the already solidified part due to it (the simulation ran from right to left in figure 65):

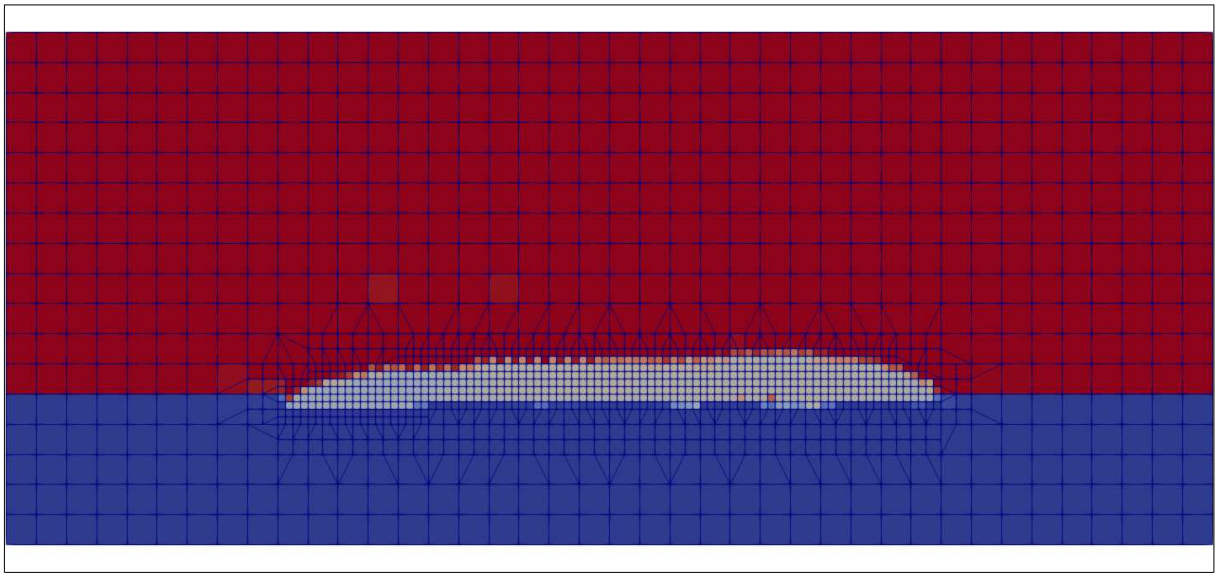


Figure 65: LA longitudinal section

Many of the cross-sections after the first pass are in good agreement with the experimental results, which in principle would indicate that the model is already well calibrated.

However, as mentioned before, the results seem to diverge for subsequent passes. Although finding the right diagnosis is not an easy task, a careful analysis of the deviation trends in the cross-sections suggests that probably this happens as a result of some suboptimal assumptions in the calculation of the surface energy of the molten mixture (note the different compositions of substrate and metal powder).

Another possible (and at least partial) explanation for the observed deviations could be existing fluctuations in the cross-section during the experiments. This was found by comparing the area of the section after each pass. As one can see from the table below in figure 66, the increase in area is not the same after each pass (contrary to the expected scenario in case of having an invariable power coupling efficiency). Of course one cannot avoid certain inaccuracies introduced by the visual software tool [19] used to measure the experimental cross-sections, but provided the acceptable image resolution of available cross-sections some fluctuation in the experiments seems to actually exist and thus should not be discarded.

Case	1ST pass	2ND pass	3RD pass	increase after 1ST pass	increase after 2ND pass
LA	4.72	9.23	12.44	4.51	3.21
LB	4.14	7.01	10.63	2.87	3.62
LC	5.59	9.62	13.28	4.03	3.66
LD	3.28	6.05	8.86	2.77	2.81
LE	3.33	5.78	7.76	2.45	1.98
LF	5.48	10.86	14.91	5.38	4.05

Figure 66: Increase in experimental cross-sectional area

Despite the high accuracy seen in most cases after the first pass, due to current limitations in experimental information, more data and especially more time would be required in order to improve the current calibration of the model. If enough experimental data was available, even machine learning algorithms could be implemented in order to optimize the simulation parameters.

Only one exemplary thermal cycle plot was shown in figure 62 and compared to the experimental data for condition LA (figures 59, 60 and 61). These show high accuracy between simulation and experiment.

An improvement of said cycle to better match the experimental data is unreasonable and unnecessary due to the inherent inaccuracies which were introduced by the measuring method during the experiments: Even though probe locations were given with an accuracy of 10µm, the thermocouples themselves measured 1mm across, making a comparison with infinitely small probes during the simulation obsolete.

3.4 Proof of concept for EHLA

Coating materials can be done approximately 10x faster at a speed of more than 500 m/min using the new EHLA - technology, which was developed and patented by a team at the Fraunhofer Institute for Laser Technology. This of course has many applications in the industry and thus the interest arises to simulate the process. [20]

Simulating this new technique however comes with its own challenges. Mainly the speed in which the laser and the injector move and the high particle density revealed some flaws in the current version of the solver, which previously went unnoticed due to the effect being negligible when studying conventional L-DED.

For these reasons the new calculation of particle initial velocity, the adaptation of the melting condition and the new approach for calculating the absorption(based on the exposed particle area “eta”-value) were added with the goal of making it possible to simulate EHLA-processes.

3.4.1 Results assessment

EHLA is not yet a state of the art technology and thus not many informations are freely available. For this reason, no direct comparison or verification has been carried out. The following simulation is just based on some parameters which were extracted from the patent claim[3]:

- laser power: 1 kW
- velocity: 130 m/min
- particle mass flow rate: 12 g/min

The following results show that the coating could be simulated at those high speeds. So these should just serve as a proof of concept, that the solver is capable to simulate such a complex scenario.

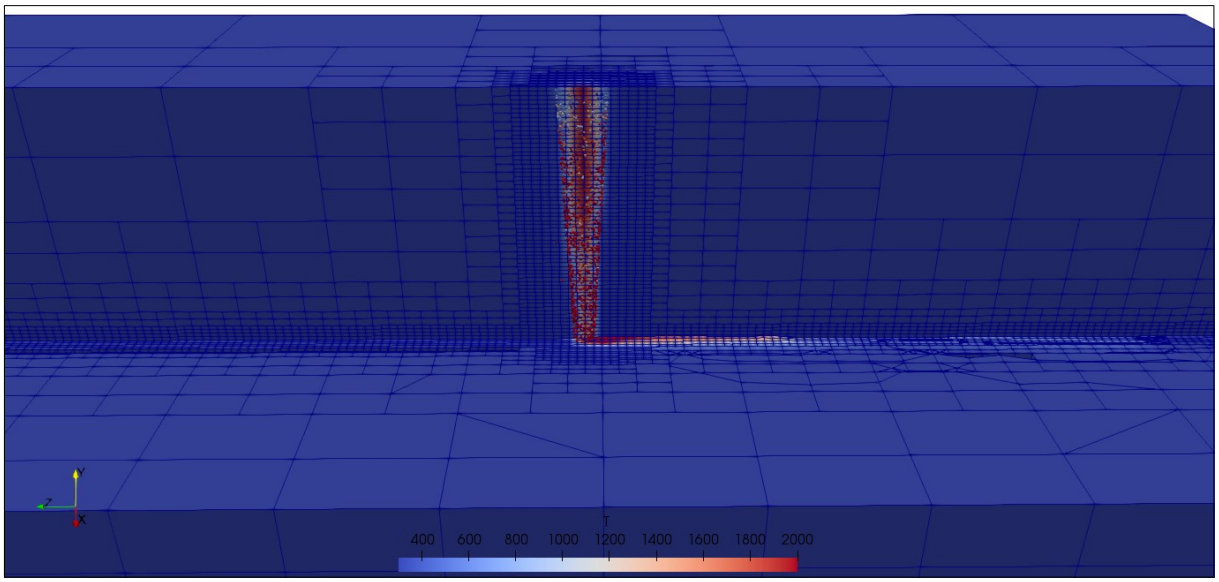


Figure 67: Overview of simulated EHLA

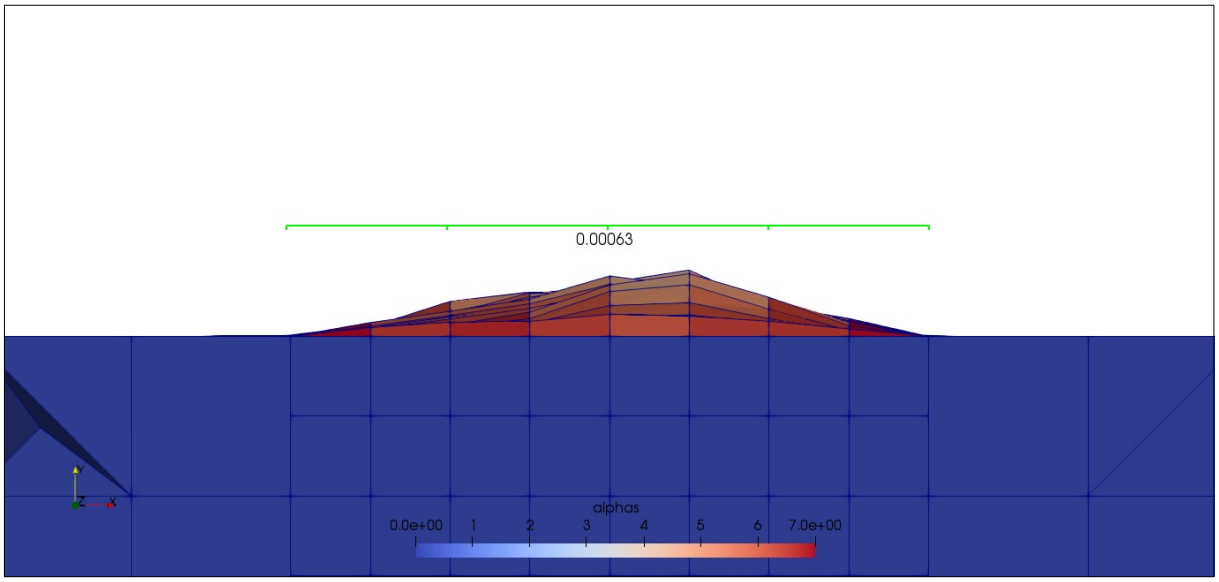


Figure 68: Cross-section of EHLA (distance value in m)

In the future it would surely be of interest to simulate the EHLA-process and compare the result to actual experimental data. A calibration will be needed to simulate it in a physically correct manner, more lessons will be learned and the solver's capabilities will be improved even further.

4 Conclusions and outlook

As widely described in quality management, improvement in small steps (KAIZEN[16]) is usually the best way to gain better results and thus this philosophy was used to improve upon already outstanding work done by my predecessors.

The main goal of this thesis was to extend the capabilities of a previously existing multiphysical model for the simulation of laser assisted manufacturing processes towards its application for L-DED AM processes can be considered to have been achieved.

Thanks to the dedicated modelling and setup work carried out hereby, the simulation results of L-DED processes seem quite promising, as they indicate that there is no fundamental flaw with the solver and that it is able to simulate those processes macroscopically and even to provide useful thermal information that is critical for assessing the final microstructural material characteristics. However a finer calibration of the model and its input will still be needed in the future. For this purpose extending the currently available experimental data would be especially useful.

In addition to the achievements in the simulation of L-DED, the basic principles of efficient and versatile programming were never compromised, which is proven by the fact that only small changes were necessary to the overall model in order to simulate EHLA.

While some changes improved the mentioned processes considerably, those tweaks also extend into current research topics, such as the simulation of L-PBF. The new laser energy absorption calculation being the most important one.

Also noteworthy is that, despite of having enabled a quick way to assess possible thermal cycle strategies based on the thermal gradient orientation, a more sophisticated model reproducing the actual growth of crystalline structures would offer a more robust and physically accurate alternative and should therefore be considered in the future.

In the long term more challenging tasks could be envisioned, such as the coupled implementation of thermal mechanics in order to predict the dimensional changes

during the process or even to achieve more efficient simulation methods that allow simulating the whole component repair process in a reasonable time frame.

5 List of literature

[1] OpenCFD, “OpenFOAM® - Official home of The Open Source Computational Fluid Dynamics (CFD) Toolbox.” [Online]. Available: http://www.openfoam.com. [Accessed: 11-Nov-2019].

[2] J. Ibarra-Medina, M. Vogel, and A. J. Pinkerton, “A CFD model of laser cladding: From deposition head to melt pool dynamics,” in *International Congress on Applications of Lasers & Electro-Optics*, Orlando, Florida, USA, 2011, pp. 378–386, doi: 10.2351/1.5062261.

[3] W. Küppers, G. M. Backes, and J. Kittel, “Extremes Hochgeschwindigkeitslaserauftragsschweißverfahren,” DE102011100456B4, 07-May-2015.

[4] M. Brandt, S. Sun, N. Alam, P. Bendeich, and A. Bishop, “Laser cladding repair of turbine blades in power plants: From research to commercialisation,” *Int. Heat Treat. Surf. Eng.*, vol. 3, pp. 105–114, Sep. 2009, doi: 10.1179/174951409X12542264513843.

[5] L. Xue and M. U. Islam, “Laser Consolidation - A Novel One-Step Manufacturing Process for Making Net-Shape Functional Components,” p. 49.

[6] C. Lampa and I. Smirnov, “High speed laser cladding of an iron based alloy developed for hard chrome replacement,” *J. Laser Appl.*, vol. 31, no. 2, Apr. 2019, doi: 10.2351/1.5096142.

[7] I. Tabernero, A. Lamikiz, E. Ukar, L. N. López de Lacalle, C. Angulo, and G. Urbikain, “Numerical simulation and experimental validation of powder flux distribution in coaxial laser cladding,” *J. Mater. Process. Technol.*, vol. 210, no. 15, pp. 2125–2134, Nov. 2010, doi: 10.1016/j.jmatprotec.2010.07.036.

[8] P. A. Vesilind, “The Rosin-Rammler particle size distribution,” *Resour. Recovery Conserv.*, vol. 5, no. 3, pp. 275–277, Sep. 1980, doi: 10.1016/0304-3967(80)90007-4.

[9] J. H. Ferziger and M. Peric, *Computational Methods for Fluid Dynamics*, 3rd ed. Berlin Heidelberg: Springer-Verlag, 2002.

[10] Codina Alvarez, Francisco Javier, “Improvement of Lagrangian approach for the multy-physical simulation of powder based additive manufacturing laser processes,” Master thesis, Vienna University of Technology, Vienna, 2019.

[11] I. Medina and J. Ramses, “Development and application of a CFD model of laser metal deposition,” 2013. [Online]. Available: /paper/Development-and-application-of-a-CFD-model-of-laser-Medina-Ramses/1adb3914a8a2c9e93400b328210fc7b36e9df2e7. [Accessed: 19-Dec-2019].

[12] M. P. Fewell, “Area of Common Overlap of Three Circles,” p. 30.

[13] Sloane, N.J.A., “The On-Line Encyclopedia of Integer Sequences.” .

[14] N. K. Arakere and G. Swanson, “Effect of Crystal Orientation on Fatigue Failure of Single Crystal Nickel Base Turbine Blade Superalloys,” *J. Eng. Gas Turbines Power*, vol. 124, no. 1, pp. 161–176, Jan. 2002, doi: 10.1115/1.1413767.

[15] L. Liu, T. W. Huang, J. Zhang, and H. Z. Fu, “Microstructure and stress rupture properties of single crystal superalloy CMSX-2 under high thermal gradient directional solidification,” *Mater. Lett.*, vol. 61, no. 1, pp. 227–230, Jan. 2007, doi: 10.1016/j.matlet.2006.04.037.

[16] D. Wang, C. Song, Y. Yang, and Y. Bai, “Investigation of crystal growth mechanism during selective laser melting and mechanical property characterization of 316L stainless steel parts,” *Mater. Des.*, vol. 100, pp. 291–299, Jun. 2016, doi: 10.1016/j.matdes.2016.03.111.
[17] “ParaView.” [Online]. Available: https://www.paraview.org/. [Accessed: 13-Nov-2019].
[18] K. C. Mills, B. J. Keene, R. F. Brooks, and A. Shirali, “Marangoni Effects in Welding,” *Philos. Trans. Math. Phys. Eng. Sci.*, vol. 356, no. 1739, pp. 911–925, 1998.
[19] “www.SketchAndCalc.com,” *SketchAndCalc*. [Online]. Available: https://www.sketchandcalc.com. [Accessed: 15-Nov-2019].
[20] “Extremes Hochgeschwindigkeits Laserauftragsschweissen - Hornet Laser Cladding.” [Online]. Available: https://www.hornetlasercladding.com/extremes-hochgeschwindigkeits-laserauftragsschweissen. [Accessed: 11-Nov-2019].
[21] “Laser cladding replaces TIG for industrial gas turbines,” *Industrial Laser Solutions*, 01-Nov-2012. [Online]. Available: https://www.industrial-lasers.com/cutting/article/16485311/laser-cladding-replaces-tig-for-industrial-gas-turbines. [Accessed: 11-Nov-2019].
[22] “Mitsubishi Heavy Industries, Ltd. Global Website.” [Online]. Available: https://www.mhi.com/. [Accessed: 11-Nov-2019].

6 Appendix

6.1 Position projection onto a plane and calculation of the overlap of two particles in a single cell

From here on to the end of the section, particles with the index 1 will denote the largest one, while the index 3 describes the smallest one and particles nearest to the laser source will have the index A, then in ascending order of distance B, C, etc.

The positions of the particles projected onto a plane orthogonal to the laser direction are also needed (see figure 21 and 22):

$$\vec{p_{iOrthogonal}} = \vec{p_i} - [(\vec{p_i} \cdot \vec{d_{laser}}) \cdot \vec{d_{laser}}]$$

Equation 28: Position on a 2D-plane orthogonal to laser direction

where:

$\vec{p_{iOrthogonal}}$: position of i projected onto plane orthogonal to laser

$\vec{p_i}$: distance between the centres of particle i and j

$\vec{d_{laser}}$: laser direction of unit length

Using this calculated positions the possibility of at least partial overlap of the projected areas is checked by following condition:

$$d_{ij} < \sum_{particles} r_i$$

Equation 29: Condition for at least partial overlap

where:

d_{ij}: distance between the centres of particle i and j

r_i: radius of particle i

Complete overlap (one particle is entirely blocked by another or simpler: two circles lie completely inside each other) is checked by following condition:

$$\vec{p_i} + r_i \cdot \frac{\vec{d_{ij}}}{|\vec{d_{ij}}|} \geqslant \vec{p_j} + r_j \cdot \frac{\vec{d_{ij}}}{|\vec{d_{ij}}|}$$

Equation 30: Condition for complete overlap

where:

$\vec{d}_{ij}$: vector between the centres of particle i and j

Then the unobscured area of the particle nearest the laser source is just:

$$A_{particleA} = r^2_{particleA} \cdot \pi$$

Equation 31: Projected unobscured area of particle nearest to the laser source

For calculation of the unobscured area of the second particle, the equation remains the same, minus the possible complete or partial overlap.

$$A_{particleB} = r^2_{particleB} \cdot \pi - A_{overlap}$$

Equation 32: Projected unobscured area of the second particle

where:

$$A_{overlap} = 0$$

Equation 33: Overlapping area in case of no overlap

$$A_{overlap} = r^2_{particle\,2} \cdot \pi$$

Equation 34: Overlapping area in case of complete overlap

When only partial overlap is detected, in order to calculate the overlapping area correctly, it is necessary to distinguish between the following two cases:

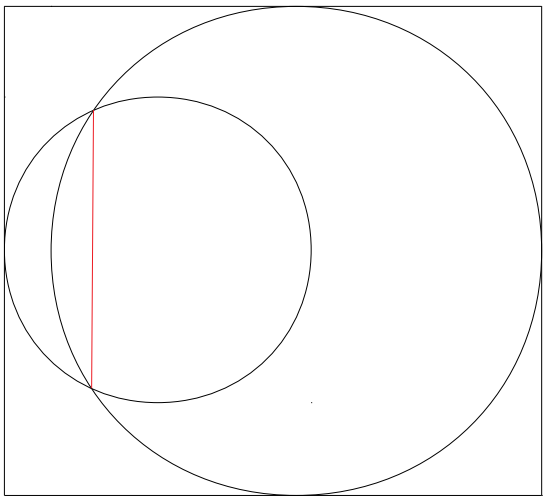


Figure 69: Case 1: The line connecting the two intersection points does not cross the line connecting the two centres

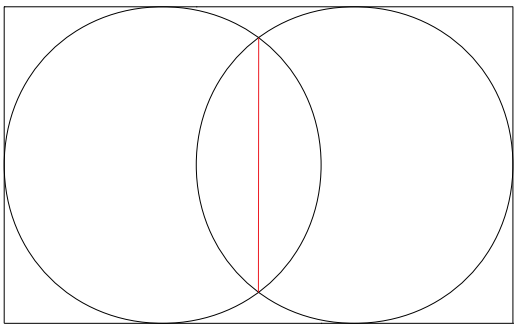


Figure 70: Case 2: The line connecting the two intersection points does cross the line connecting the two centres

$$A_{overlap} = r_i^2 \cdot \arcsin\left(\frac{y}{r_i}\right) - (d_{ij} + x) \cdot y + x \cdot y + r_j^2 \cdot \left(\pi - \arcsin\left(\frac{y}{r_j}\right)\right)$$

Equation 35: Overlapping area for case 1

$$A_{overlap} = r_i^2 \cdot \arcsin\left(\frac{y}{r_i}\right) + r_j^2 \cdot \arcsin\left(\frac{y}{r_j}\right) - y \cdot \left(x + \sqrt{r_j^2 - r_i^2 + x^2}\right)$$

Equation 36: Overlapping area for case 2

where:

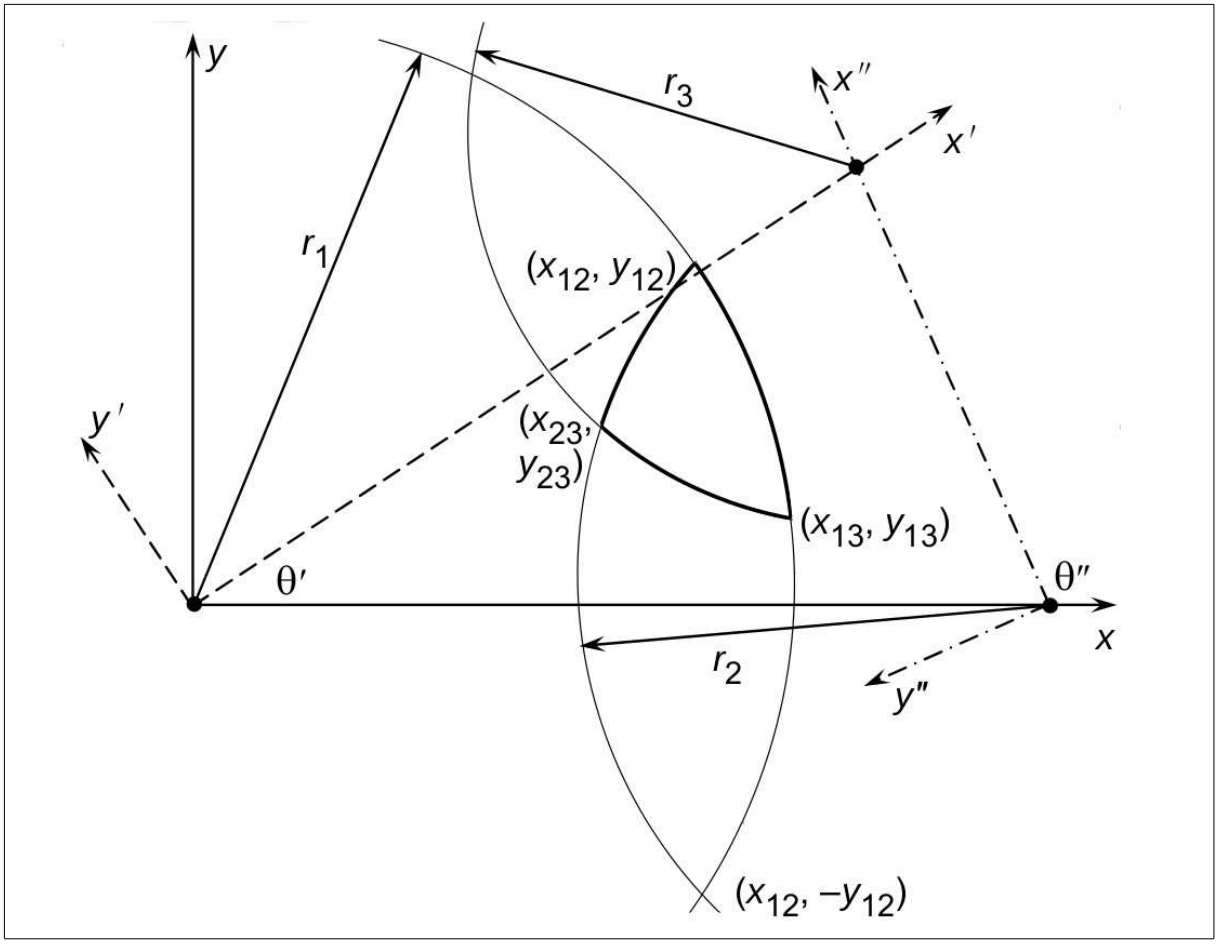


Figure 71: Topology of three intersecting circles (same notation used in the case of two intersecting circles)[12]

$$x_{ij} = \frac{r_i^2 - r_j^2 - d_{ij}^2}{2 * d_{ij}}$$

Equation 37: Calculation of x-component of intersection point for case 1 [12]

$$x_{ij} = \frac{r_i^2 - r_j^2 + d_{ij}^2}{2 * d_{ij}}$$

Equation 38: Calculation of x-component of intersection point for case 2 [12]

$$y_{ij} = \sqrt{r_2^2 - x^2}$$

Equation 39: Calculation of y-component of intersection point for case 1 [12]

$$y_{ij} = \sqrt{r_1^2 - x^2}$$

Equation 40: Calculation of y-component of intersection point for case 2 [12]

These values can now be used to calculate the absorbed energy per particle as well as the unobscured area of the second particle according to equations 18 and 24.

6.2 Case selection and calculation of unobscured particle area for three particles within a cell

From here on to the end of the section, particles with the index 1 will denote the largest one, while the index 3 describes the smallest one and particles nearest to the laser source will have the index A, then in ascending order of distance B, C, etc.

CASE 1:

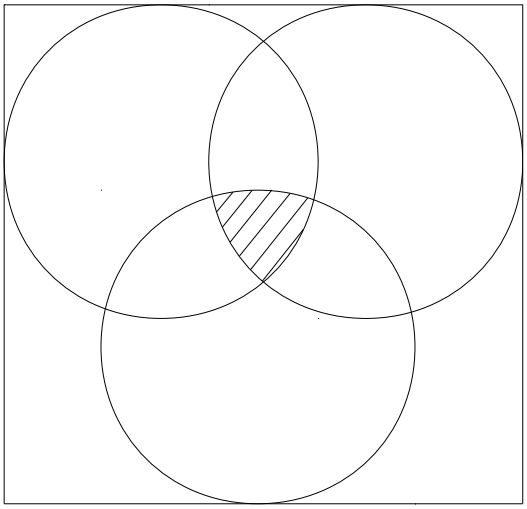


Figure 72: Circular triangle

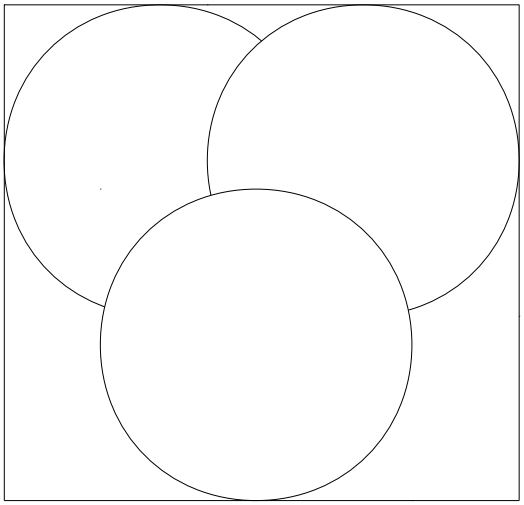


Figure 73: Particles, as seen from the laser source

Here the particles need to intersect at 6 distinct points to form an area of common overlap resembling a circular triangle. In other words every circle has to intersect every other circle. Further conditions besides the need for intersection and the procedure for calculating the area of overlap (circular triangle) are detailed in [12] and will not be discussed further in this thesis.

Using that information, the sorted list of particles, and the area of common overlap, one can assign the $A_{particle}$ to the proper particle and thus calculate the absorbed energy and, using the sum of all areas, eta. In this case, for programming purposes, it does not matter which particle is on top of which, as long as one has knowledge about the position, $A_{particle}$ can be calculated as follows:

$$A_{particleA} = r^2_{particleA} \cdot \pi$$

Equation 41: Area of particle nearest to laser source

$$A_{particleB} = r^2_{particleB} \cdot \pi - A_{overlapAB}$$

Equation 42: Area of particle second nearest to the laser source

$$A_{particleC} = r^2_{particleC} \cdot \pi - A_{overlapAC} - A_{overlapBC}$$

Equation 43: Area of particle farthest from the laser source

CASE 2:

This configuration is identified by the fact, that the farthest point of the circle, whose centre lies between the other two, in either direction, perpendicular to the vector connecting the two outer centres, lies between the two intersection points of those outer circles. Also the inner circle must only partially intersect the outer two:

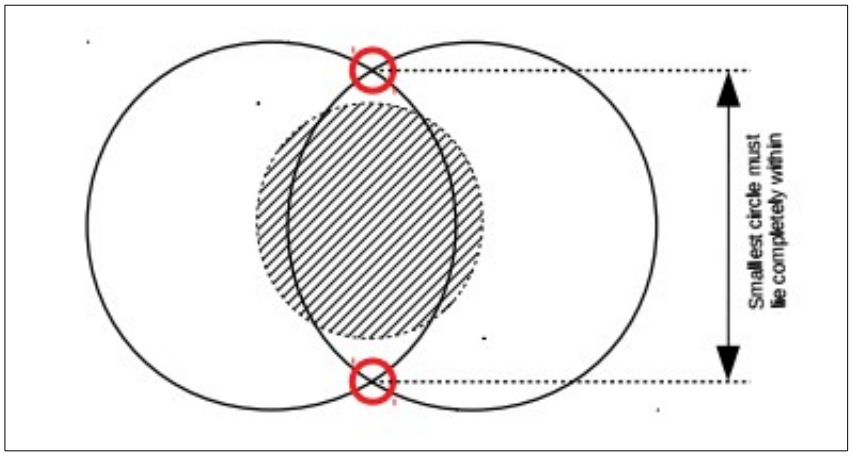


Figure 74: Case 2 schematic representation

In order to compute the necessary conditions for classification, the positions of the intersection points need to be known (those are also used for the computation of conditions for case 3 and 7):

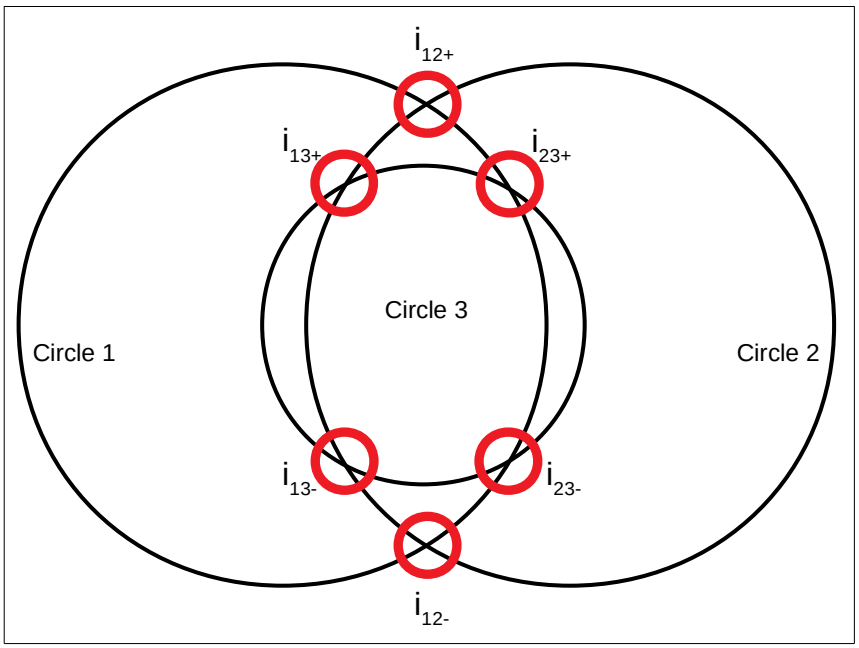


Figure 75: Intersection point denomination (shown assuming that circle 3 lies in-between the other two)

$$\vec{i_{12+}} = x_{12} \cdot \vec{e}_x + |y_{12}| \cdot \vec{e}_y$$

Equation 44: Intersection point between circles 1 and 2

$$\vec{i_{12-}} = x_{12} \cdot \vec{e}_x + (-1) \cdot |y_{12}| \cdot \vec{e}_y$$

Equation 45: Intersection point between circles 1 and 2 mirrored on line connecting the two circle centres

$$\vec{i_{23+}} = d_{12} \cdot \vec{e}_x + x_{23} \frac{\vec{d}_{23}}{|\vec{d}_{23}|} + |y_{23}| \cdot \left(\vec{e}_y \times \frac{\vec{d}_{23}}{|\vec{d}_{23}|}\right)$$

Equation 46: Intersection point between circles 2 and 3

$$\vec{i_{23-}} = d_{12} \cdot \vec{e}_x + x_{23} \frac{\vec{d}_{23}}{|\vec{d}_{23}|} + (-1) \cdot |y_{23}| \cdot \left(\vec{e}_y \times \frac{\vec{d}_{23}}{|\vec{d}_{23}|}\right)$$

Equation 47: Intersection point between circles 2 and 3 mirrored on line connecting the two circle centres

$$\vec{i_{13+}} = x_{13} \frac{\vec{d}_{13}}{|\vec{d}_{13}|} + |y_{23}| \cdot \left(\vec{e}_z \times \frac{\vec{d}_{13}}{|\vec{d}_{13}|}\right)$$

Equation 48: Intersection point between circles 1 and 3

$$\vec{i}_{13-} = x_{13}\frac{\vec{d}_{13}}{|\vec{d}_{13}|} + (-1)\cdot|y_{23}|\cdot(\vec{e}_z \times \frac{\vec{d}_{13}}{|\vec{d}_{13}|})$$

Equation 49: Intersection point between circles 1 and 3 mirrored on line connecting the two circle centres

where:

$\vec{e}_i$: is the unit vector in a coordinate described in figure 25.

In addition to the condition, that all circles overlap with one another, following conditions need to be evaluated as well:

$$c1 \wedge [(c2 \wedge c3 \wedge c4) \vee (c5 \wedge c6 \wedge c7) \vee (c8 \wedge c9 \wedge c10)]$$

Equation 50: Condition for categorization as case 2

where:

$$c1 = [[(\vec{d}_{13}\cdot\vec{d}_{12})\cdot\frac{\vec{d}_{12}}{|\vec{d}_{12}|}]\cdot[(\vec{d}_{23}\cdot\vec{d}_{12})\cdot\frac{\vec{d}_{12}}{|\vec{d}_{12}|}] < 0]$$

Equation 51: If true, the centre of circle 3 lies between the other two centres

$$c2 = [|\vec{i}_{12+} - \vec{p}_3| \geq r_3]$$

Equation 52: If true, one intersection point of circle 1 and 2 lies outside circle 3

$$c3 = [|\vec{i}_{12-} - \vec{p}_3| \geq r_3]$$

Equation 53: If true, one intersection point of circle 1 and 2 lies outside circle 3

$$c4 = [(\vec{i}_{12+} - \vec{p}_3)\cdot(\vec{i}_{12-} - \vec{p}_3) < 0]$$

Equation 54: If true, the two intersection points of circle 1 and 2 lie on either side of the centre of circle 3

$$c5 = [|\vec{i}_{23+} - \vec{p}_1| \geq r_1]$$

Equation 55: If true, one intersection point of circle 2 and 3 lies outside circle 1

$$c6 = [|\vec{i}_{23-} - \vec{p}_1| \geq r_1]$$

Equation 56: If true, one intersection point of circle 2 and 3 lies outside circle 1

$$c7 = [(\vec{i}_{23+} - \vec{p}_1)\cdot(\vec{i}_{23-} - \vec{p}_1) < 0]$$

Equation 57: If true, the two intersection points of circle 2 and 3 lie on either side of the centre of circle 1

$$c8=[|\vec{i}_{13+}-\vec{p}_2|\geq r_2]$$

Equation 58: If true, one intersection point of circle 1 and 3 lies outside circle 2

$$c9=[|\vec{i}_{13-}-\vec{p}_2|\geq r_2]$$

Equation 59: If true, one intersection point of circle 1 and 3 lies outside circle 2

$$c10=[(\vec{i}_{13+}-\vec{p}_2)\cdot(\vec{i}_{13-}-\vec{p}_2)<0]$$

Equation 60: If true, the two intersection points of circle 1 and 3 lie on either side of the centre of circle 2

where:

$\vec{p}_i$: centre of circle i

The calculation of the projected areas is done by differentiating between 3 cases, where the inner particle is either nearest to the laser source (a), second nearest (b) or the farthest away (c):

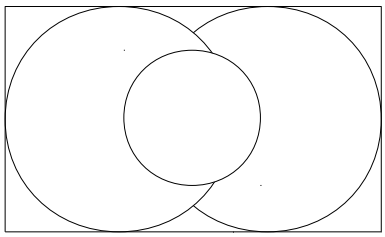


Figure 76: (a)

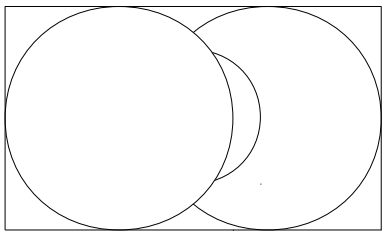


Figure 77: (b)

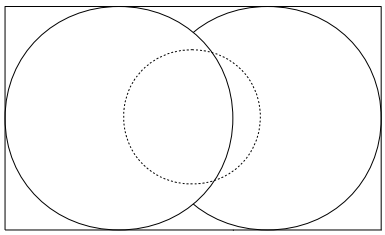


Figure 78: (c)

The equation, when tailored to each case, becomes:

for all cases:

$$A_{particleA}=r^2_{particleA}\cdot\pi$$

Equation 61: Area of particle nearest to the laser source

$$A_{particleB}=r^2_{particleB}\cdot\pi-A_{overlapAB}$$

Equation 62: Area of particle second nearest to the laser source

for (a) and (b):

$$A_{particleC}=(r^2_{particleB}+r^2_{particleC})\cdot\pi-A_{particleA}-A_{particleB}$$

Equation 63: Area of particle farthest from the laser source

for (c)

$$A_{particleC} = 0$$

Equation 64: Area of particle farthest the laser source

CASE 3:

This scenario is similar to the one previously described in case 2, with the exception that the farthest point of the innermost circle lies outside of the two intersection points as seen in figure 79:

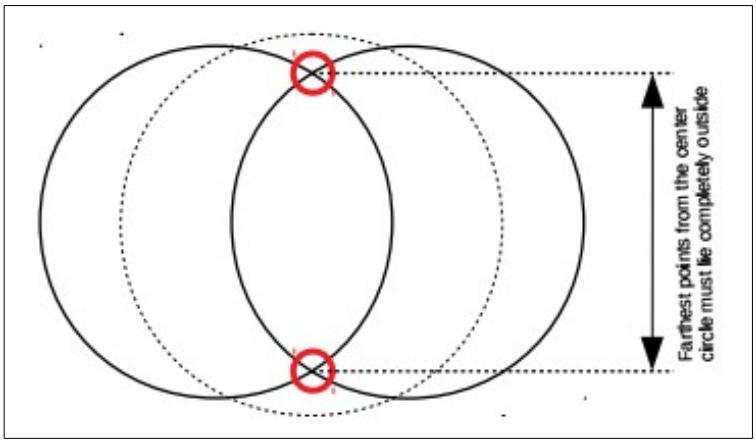


Figure 79: Case 3 schematic representation

The algorithm selects this case after checking following condition and using previously defined and calculated intersection points:

$$\left[max\left(\vec{i}_{12+} - \vec{p}_3, \vec{i}_{12-} - \vec{p}_3\right) \leq r_3\right] \vee \left[max\left(\vec{i}_{23+} - \vec{p}_1, \vec{i}_{23-} - \vec{p}_1\right) \leq r_1\right] \vee \left[max\left(\vec{i}_{12+} - \vec{p}_2, \vec{i}_{13-} - \vec{p}_2\right) \leq r_2\right]$$

Equation 65: Condition for categorization as case 3

For proper calculation of each projected area, a general differentiation between two sub-cases has to be made: (a) the third circle intersects the overlapping area of the other two and (b) the third circle encloses the overlapping area of the other two:

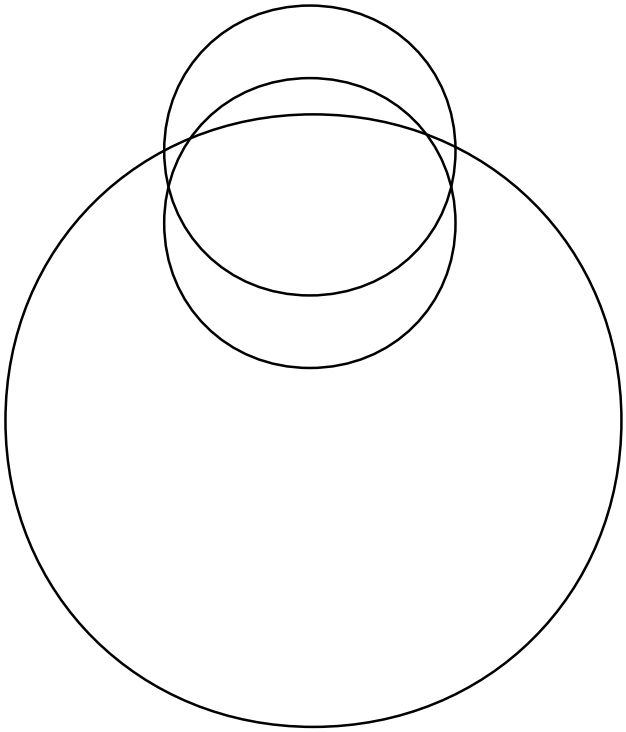


Figure 80: (a)

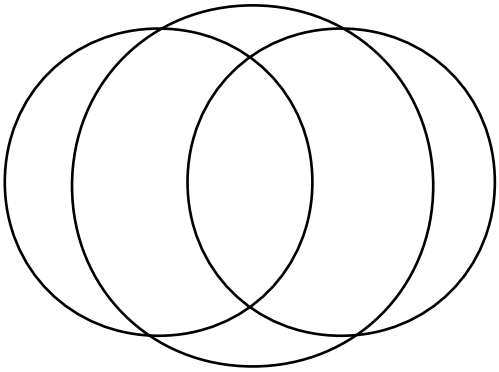


Figure 81: (b)

First, case a will be further divided into 6 different sub-scenarios according to the position of each circle relative to the others:

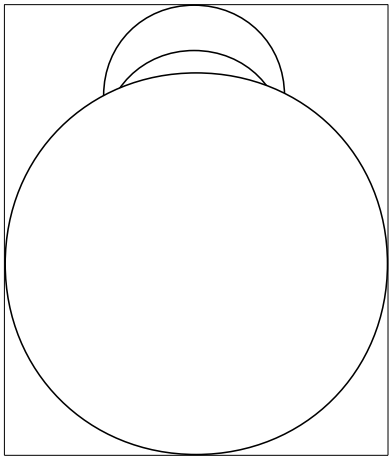


Figure 82: (a1)

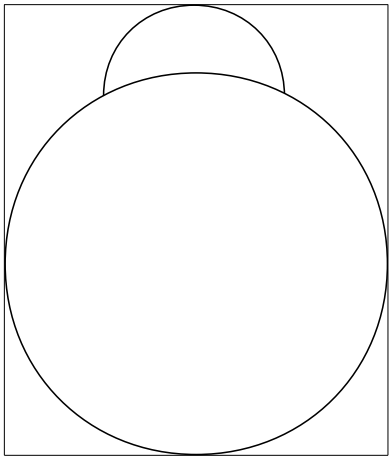


Figure 83: (a2)

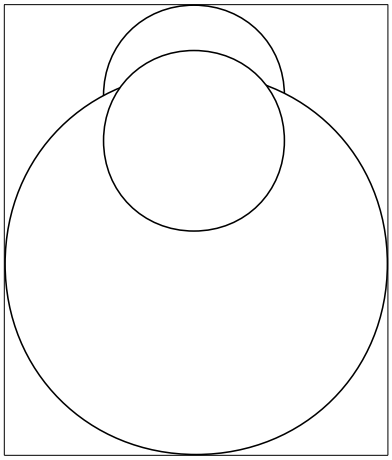


Figure 84: (a3)

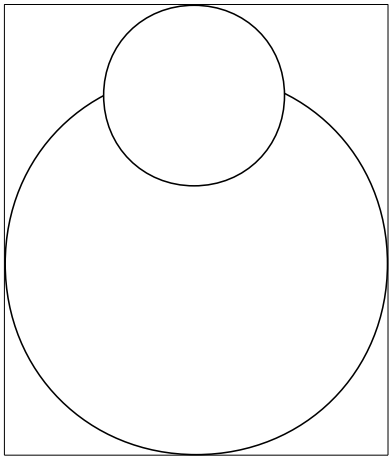


Figure 85: (a4)

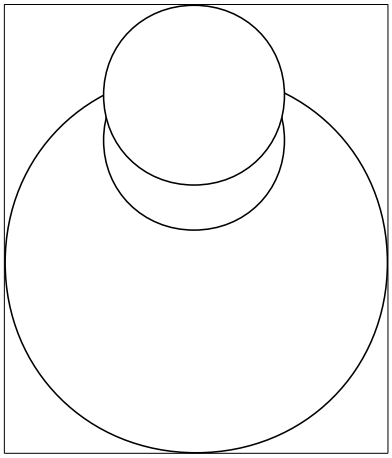


Figure 86: (a5)

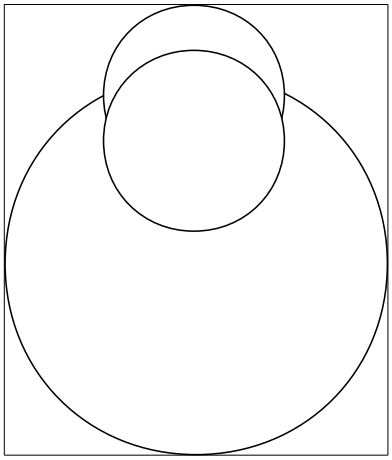


Figure 87: (a6)

Secondly we can do the same with case b, however here only those 3 cases are relevant for programming it:

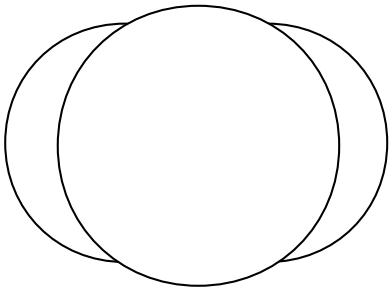


Figure 88: (b1)

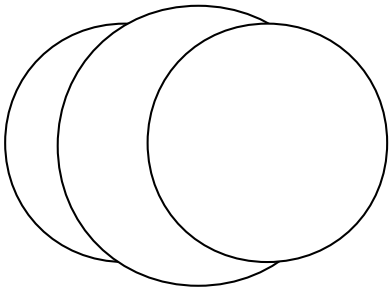


Figure 89: (b2)

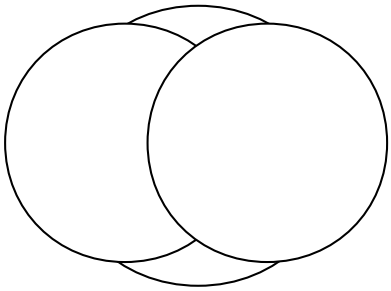


Figure 90: (b3)

The equation for the area of the two particles nearest to the laser can be programmed in the same way, independently from sub-scenario:

$$A_{particleA} = r^2_{particleA} \cdot \pi$$

Equation 66: Area of particle nearest to the laser source

$$A_{particleB} = r^2_{particleB} \cdot \pi - A_{overlapAB}$$

Equation 67: Area of particle second nearest to the laser source

To calculate the area of the particle farthest from the laser source, some more computations and conditions for the different sub-scenarios are necessary. Out of necessity and geometrical constraints, the intersection points need to be calculated differently, depending on the present case, therefore a condition is checked and then the right calculation for the intersection points is selected. The denomination of those points changes slightly from case 2, where instead of the index referring to the size of the circle, it refers to the position relative to the laser source:

$$if = ([\sqrt{((|\vec{p}_B - \vec{p}_A|)^2 + r_A^2)} < r_B] \vee [\sqrt{((|\vec{p}_B - \vec{p}_A|)^2 + r_B^2)} < r_A])$$

Equation 68: Condition when circles A and B intersect as described in case 1 (see figure 69)

$$\vec{x}_{AB} = \frac{r_B^2 - r_A^2 - (|\vec{p}_A - \vec{p}_B|)^2}{2 \cdot (|\vec{p}_A - \vec{p}_B|)} \cdot \frac{\vec{p}_A - \vec{p}_B}{|\vec{p}_A - \vec{p}_B|}$$

Equation 69: x-component of vector to intersection points AB

$$\vec{y}_{AB} = |(\sqrt{r_A^2 - |\vec{x}^2|})| \cdot [\frac{\vec{x}}{|\vec{x}|} \times \frac{\vec{d}_{laser}}{|\vec{d}_{laser}|}]$$

Equation 70: y-component of vector to intersection point AB

$$\vec{i}_{AB+} = \vec{p}_B + \vec{x} + \vec{y}$$

Equation 71: Vector to AB+ from B

$$\vec{i}_{AB-} = \vec{p}_B + \vec{x} - \vec{y}$$

Equation 72: Vector to AB- from B

else

Condition when circles A and B intersect as described in case 2 (see figure 70)

$$\vec{x}_{AB} = \frac{r_A^2 - r_B^2 + (|\vec{p}_B - \vec{p}_A|)^2}{2 \cdot (|\vec{p}_B - \vec{p}_A|)} \cdot \frac{\vec{p}_B - \vec{p}_A}{|\vec{p}_B - \vec{p}_A|}$$

Equation 73: x-component of vector to intersection points AB

$$\vec{y}_{AB} = |(\sqrt{r_A^2 - |\vec{x}^2|})| \cdot [\frac{\vec{x}}{|\vec{x}|} \times \frac{\vec{d}_{laser}}{|\vec{d}_{laser}|}]$$

Equation 74: y-component of vector to intersection points AB

$$\vec{i}_{AB+} = \vec{p}_A + \vec{x} + \vec{y}$$

Equation 75: Vector to AB+ from A

$$\vec{i}_{AB-} = \vec{p}_A + \vec{x} - \vec{y}$$

Equation 76: Vector to AB- from A

$$if = ([\sqrt{((|\vec{p}_C - \vec{p}_B|)^2 + r_C^2)} < r_B] \vee [\sqrt{((|\vec{p}_C - \vec{p}_B|)^2 + r_B^2)} < r_C])$$

Equation 77: Condition when circles B and C intersect as described in case 1 (see figure 69)

$$\vec{x_{BC}} = \frac{r_C^2 - r_B^2 - (|\vec{p}_B - \vec{p}_C|)^2}{2 \cdot (|\vec{p}_B - \vec{p}_C|)} \cdot \frac{\vec{p}_B - \vec{p}_C}{|\vec{p}_B - \vec{p}_C|}$$

Equation 78: x-component of vector to intersection points BC

$$\vec{y_{BC}} = |(\sqrt{r_B^2 - |\vec{x^2}|})| \cdot [\frac{\vec{x}}{|\vec{x}|} \times \frac{\vec{d_{laser}}}{|\vec{d_{laser}}|}]$$

Equation 79: y-component of vector to intersection points BC

$$\vec{i_{BC+}} = \vec{p}_C + \vec{x} + \vec{y}$$

Equation 80: Vector to BC+ from C

$$\vec{i_{BC-}} = \vec{p}_C + \vec{x} - \vec{y}$$

Equation 81: Vector to BC- from C

else

Condition when circles B and C intersect as described in case 2 (see figure 70)

$$\vec{x_{BC}} = \frac{r_B^2 - r_C^2 + (|\vec{p}_C - \vec{p}_B|)^2}{2 \cdot (|\vec{p}_C - \vec{p}_B|)} \cdot \frac{\vec{p}_C - \vec{p}_B}{|\vec{p}_C - \vec{p}_B|}$$

Equation 82: x-component of vector to intersection points BC

$$\vec{y_{BC}} = |(\sqrt{r_B^2 - |\vec{x^2}|})| \cdot [\frac{\vec{x}}{|\vec{x}|} \times \frac{\vec{d_{laser}}}{|\vec{d_{laser}}|}]$$

Equation 83: y-component of vector to intersection points BC

$$\vec{i_{BC+}} = \vec{p}_B + \vec{x} + \vec{y}$$

Equation 84: Vector to BC+ from B

$$\vec{i_{BC-}} = \vec{p}_B + \vec{x} - \vec{y}$$

Equation 85: Vector to BC- from C

$$if = ([\sqrt{((|\vec{p}_A - \vec{p}_C|)^2 + r_A^2)} < r_C] \vee [\sqrt{((|\vec{p}_A - \vec{p}_C|)^2 + r_C^2)} < r_A])$$

Equation 86: Condition when circles A and C intersect as described in case 1 (see figure 69)

$$\vec{x_{AC}} = \frac{r_A^2 - r_C^2 - (|\vec{p}_C - \vec{p}_A|)^2}{2 \cdot (|\vec{p}_C - \vec{p}_A|)} \cdot \frac{\vec{p}_C - \vec{p}_A}{|\vec{p}_C - \vec{p}_A|}$$

Equation 87: x-component of vector to intersection points AC

$$\vec{y_{AC}} = |(\sqrt{r_C^2 - |\vec{x^2}|})| \cdot [\frac{\vec{x}}{|\vec{x}|} \times \frac{\vec{d_{laser}}}{|\vec{d_{laser}}|}]$$

Equation 88: y-component of vector to intersection points AC

$$\vec{i_{AC+}} = \vec{p}_A + \vec{x} + \vec{y}$$

Equation 89: Vector to AC+ from A

$$\vec{i_{AC-}} = \vec{p}_A + \vec{x} - \vec{y}$$

Equation 90: Vector to AC- from A

else

Condition when circles A and C intersect as described in case 2 (see figure 70)

$$\vec{x_{AC}} = \frac{r_C^2 - r_A^2 + (|\vec{p}_A - \vec{p}_C|)^2}{2 \cdot (|\vec{p}_A - \vec{p}_C|)} \cdot \frac{\vec{p}_A - \vec{p}_C}{|\vec{p}_A - \vec{p}_C|}$$

Equation 91: x-component of vector to intersection points AC

$$\vec{y_{AC}} = |(\sqrt{r_C^2 - |\vec{x^2}|})| \cdot [\frac{\vec{x}}{|\vec{x}|} \times \frac{\vec{d_{laser}}}{|\vec{d_{laser}}|}]$$

Equation 92: y-component of vector to intersection points AC

$$\vec{i_{AC+}} = \vec{p}_C + \vec{x} + \vec{y}$$

Equation 93: Vector to AC+ from C

$$\vec{i_{AC-}} = \vec{p}_C + \vec{x} - \vec{y}$$

Equation 94: Vector to AC- from C

After having calculated the intersection points in the correct way and having them stored as a vector as opposed to a scalar in case 2, the algorithm now needs to differentiate between the cases (a1)-(a6) and (b1)-(b3):

$$if = \left(\left[\left| \vec{i}_{BC+} - \vec{p}_A \right| < r_A \right] \wedge \left[\left| \vec{i}_{BC-} - \vec{p}_A \right| < r_A \right] \right)$$

Equation 95: Condition for selecting either (a1), (a2) or (b1)

$$if = \left(\left[\left| \vec{i}_{AB+} - \vec{p}_C \right| < r_C \right] \wedge \left[\left| \vec{i}_{AB-} - \vec{p}_C \right| < r_C \right] \right)$$

Equation 96: Condition for selecting sub-scenario (a1)

$$A_{particleC} = \left(r^2_{particleC} - r^2_{particleB} \right) \cdot \pi - A_{overlapAC} + A_{overlapAB}$$

Equation 97: Area calculation for sub-scenario (a1)

$$else\ if = \left(\left[\left| \vec{i}_{AC+} - \vec{p}_B \right| < r_B \right] \wedge \left[\left| \vec{i}_{AC-} - \vec{p}_B \right| < r_B \right] \right)$$

Equation 98: Condition for selecting sub-scenario (a2)

$$A_{particleC} = 0$$

Equation 99: Area calculation for sub-scenario (a2)

$$else$$

Condition for selecting sub-scenario (b1)

$$A_{particleC} = r^2_{particleC} \cdot \pi - A_{overlapAC}$$

Equation 100: Area calculation for sub-scenario (b1)

$$else\ if = \left(\left[\left| \vec{i}_{AC+} - \vec{p}_B \right| < r_B \right] \wedge \left[\left| \vec{i}_{AC-} - \vec{p}_B \right| < r_B \right] \right)$$

Equation 101: Condition for selecting either (a3), (a4) or (b2)

$$if = \left(\left[\left| \vec{i}_{AB+} - \vec{p}_C \right| < r_C \right] \wedge \left[\left| \vec{i}_{AB-} - \vec{p}_C \right| < r_C \right] \right)$$

Equation 102: Condition for selecting sub-scenario (a3)

$$A_{particleC} = \left(r^2_{particleC} - r^2_{particleA} \right) \cdot \pi - A_{overlapBC} + A_{overlapAC}$$

Equation 103: Area calculation for sub-scenario (a3)

$$\boxed{else\ if = \left([|\vec{i}_{BC+} - \vec{p}_A| < r_A] \wedge [|\vec{i}_{BC-} - \vec{p}_A| < r_A]\right)}$$

Equation 104: Condition for selecting sub-scenario (a4)

$$A_{particleC} = 0$$

Equation 105: Area calculation for sub-scenario (a4)

$$\boxed{else}$$

Condition for selecting sub-scenario (b2)

$$A_{particleC} = r^2_{particleC} \cdot \pi - A_{overlapBC}$$

Equation 106: Area calculation for sub-scenario (b2)

$$\boxed{if = \left([|\vec{i}_{AB+} - \vec{p}_C| < r_C] \wedge [|\vec{i}_{AB-} - \vec{p}_C| < r_C]\right)}$$

Equation 107: Condition for selecting either (a5), (a6) or (b3)

$$\boxed{if = \left([|\vec{i}_{BC+} - \vec{p}_A| < r_A] \wedge [|\vec{i}_{BC-} - \vec{p}_A| < r_A]\right)}$$

Equation 108: Condition for selecting sub-scenario (a5)

$$A_{particleC} = \left(r^2_{particleC} - r^2_{particleA}\right) \cdot \pi - A_{overlapBC} + A_{overlapAB}$$

Equation 109: Area calculation for sub-scenario (a5)

$$\boxed{else\ if = \left([|\vec{i}_{AC+} - \vec{p}_B| < r_B] \wedge [|\vec{i}_{AC-} - \vec{p}_B| < r_B]\right)}$$

Equation 110: Condition for selecting sub-scenario (a6)

$$A_{particleC} = \left(r^2_{particleC} - r^2_{particleA}\right) \cdot \pi - A_{overlapBC} + A_{overlapAB}$$

Equation 111: Area calculation for sub-scenario (a6)

$$\boxed{else}$$

Condition for selecting sub-scenario (b3)

$$A_{particleC} = r^2_{particleC} \cdot \pi - A_{overlapAC} - A_{overlapBC} + A_{overlapAB}$$

Equation 112: Area calculation for sub-scenario (b3)

The areas can then be used to calculate the amount of absorbed area per particle and the sum of them can be used for computing eta.

CASE 7:

Following the flow of the algorithm for case differentiation, which was designed to render differentiation as easy and computationally efficient as possible, the next scenario is described by case 7. The algorithm selects this case via exclusion principle, as it is the last one left where all three circles intersect with one another. It is characterized by the fact that the third circle intersects the first two, but has no intersection point with the area of common overlap of those.

In contrast to the previous case, the calculation of the single areas is easy as there is only one scenario as all others are just rotations or mirrored images of this one:

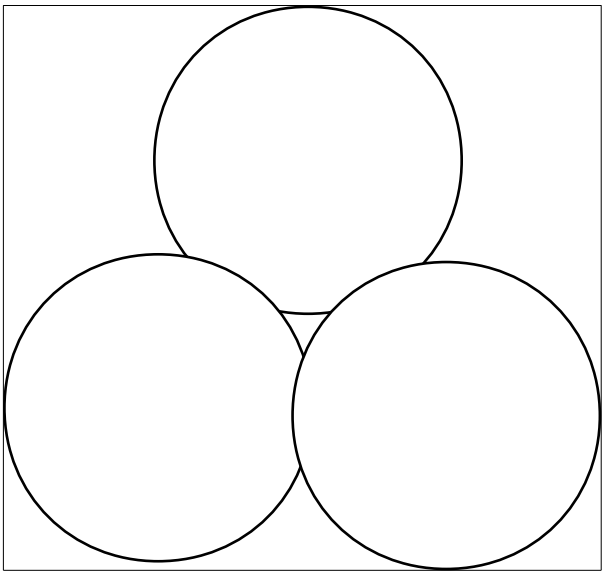


Figure 91: Case 7 schematic representation

$$A_{particleA} = r^2_{particleA} \cdot \pi$$

Equation 113: Area of particle nearest to the laser source

$$A_{particleB} = r^2_{particleB} \cdot \pi - A_{overlapAB}$$

Equation 114: Area of particle second nearest to the laser source

$$A_{particleC} = r^2_{particleC} \cdot \pi - A_{overlapAC} - A_{overlapBC}$$

Equation 115: Area of particle farthest from the laser source

CASE 6:

The next few cases all require as a first condition at least one complete overlap, which means that at least one circle is completely inside at least one other and is checked via the following condition:

$$|\vec{d}_{ij}|+r_j<r_i$$

Equation 116: Circle j lies completely inside circle i

The next scenario in the workflow of the program is case 6. The identifying condition here is that both smaller circles lie completely inside the largest one. For the purposes of calculating the sum of the projected area it is not necessary to differentiate further between this case and the ones where the inner circles do not intersect or intersect partially. This is also true for the case in which the largest particle is also the one nearest to the laser source and there is no need to differentiate between (a), (b) and (c):

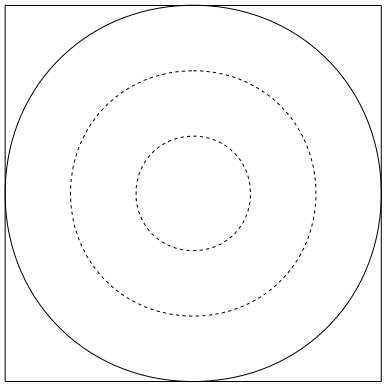


Figure 92: (a)

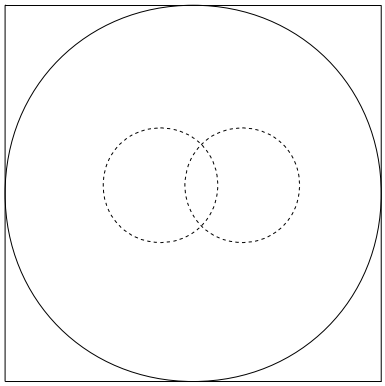


Figure 93: (b)

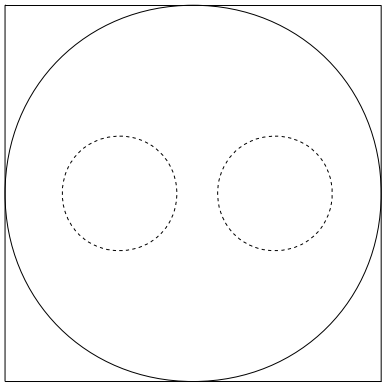


Figure 94: (c)

$$A_{particleA}=r^2_{particleA}\cdot\pi$$

Equation 117: Area of particle nearest to the laser source

$$A_{particleB}=A_{particleC}=0$$

Equation 118: Area of particle second nearest and farthest away from the laser source

When the largest particle lies in second nearest position to the laser source, the equation are also short and simple and the same for (d), (e) and (f):

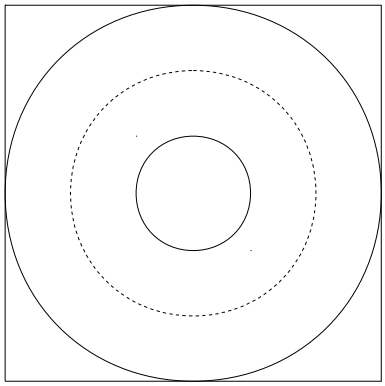


Figure 95: (d)

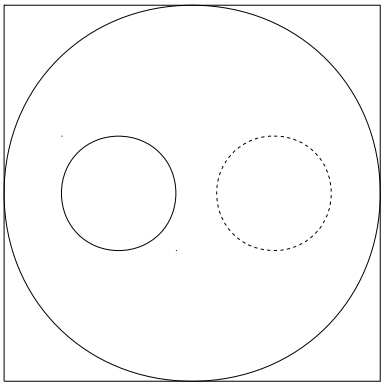


Figure 96: (e)

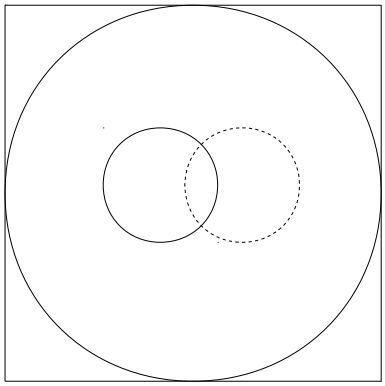


Figure 97: (f)

$$A_{particleA} = r^2_{particleA} \cdot \pi$$

Equation 119: Area of particle nearest to the laser source

$$A_{particleB} = r^2_{particleB} \cdot \pi - A_{particleA}$$

Equation 120: Area of particle second nearest to the laser source

$$A_{particleC} = 0$$

Equation 121: Area of particle farthest from the laser source

However, for assigning the correct partial area unobscured by another particle when the largest particle is at the bottom, one must take into account the order in which the particles lie relative to the laser direction and thus a differentiation between the following four scenarios is necessary:

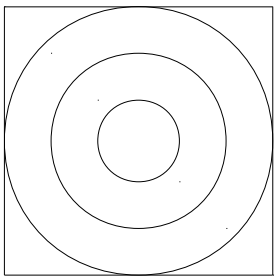


Figure 98: (g)

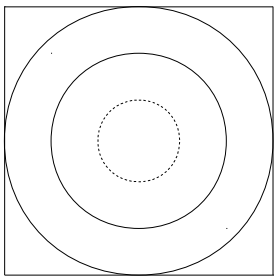


Figure 99: (h)

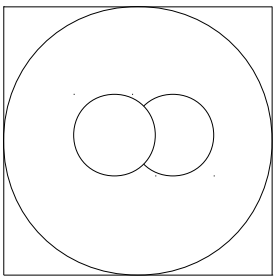


Figure 100: (i)

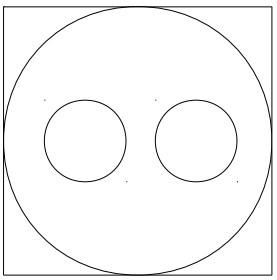


Figure 101: (j)

for all cases:

$$A_{particleA} = r^2_{particleA} \cdot \pi$$

Equation 122: Area of particle nearest to the laser source

for (g)

$$A_{particleB} = r^2_{particleB} \cdot \pi - A_{particleA}$$

Equation 123: Area of particle second nearest to the laser source

$$A_{particleC} = r^2_{particleC} \cdot \pi - A_{particleB}$$

Equation 124: Area of particle farthest from the laser source

for (h)

$$A_{particleB} = 0$$

Equation 125: Area of particle second nearest to the laser source

$$A_{particleC} = r^2_{particleC} \cdot \pi - A_{particleB}$$

Equation 126: Area of particle farthest from the laser source

for (i)

$$A_{particleB} = r^2_{particleB} \cdot \pi - A_{overlapAB}$$

Equation 127: Area of particle second nearest to the laser source

$$A_{particleC} = r^2_{particleC} \cdot \pi - A_{particleA} - A_{particleB} + A_{overlapAB}$$

Equation 128: Area of particle farthest from the laser source

for (j)

$$A_{particleB} = r^2_{particleB} \cdot \pi - A_{overlapAB}$$

Equation 129: Area of particle second nearest to the laser source

$$A_{particleC} = r^2_{particleC} \cdot \pi - A_{particleA} - A_{particleB}$$

Equation 130: Area of particle farthest from the laser source

CASE 4:

The necessary and sufficient condition for this scenario is that one, and only one, of the smaller circles lies completely inside either one of the larger two, not both.

Again, while calculating the sum is trivial, assigning the correct unobscured area to each individual particle requires some more thought and a differentiation between the following cases (a)-(f) must be carried out:

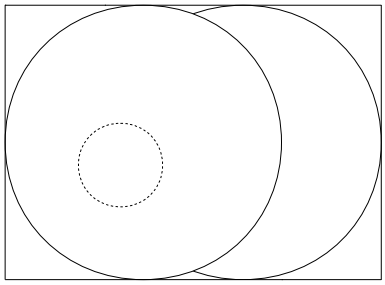


Figure 102: (a)

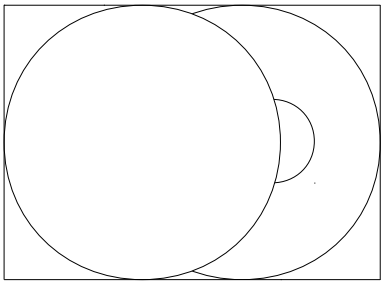


Figure 103: (b)

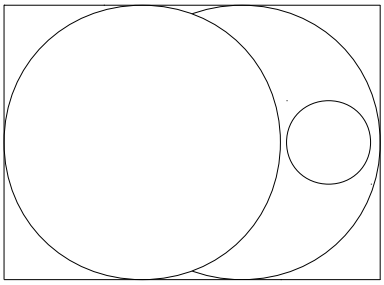


Figure 104: (c)

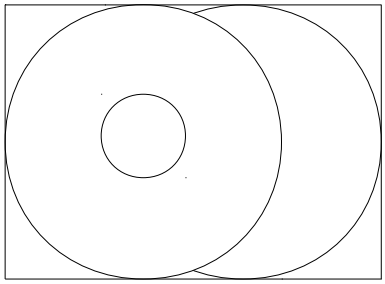


Figure 105: (d)

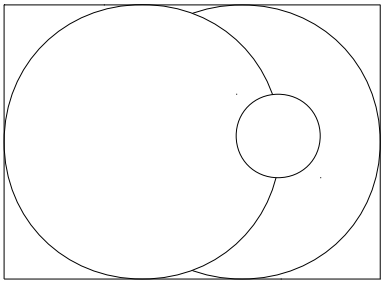


Figure 106: (e)

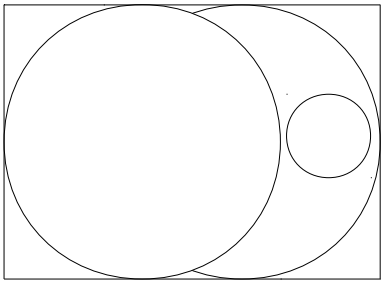


Figure 107: (f)

for all cases:

$$A_{particleA} = r^2_{particleA} \cdot \pi$$

Equation 131: Area of particle nearest to the laser source

for (a) (small particle in C position)

$$A_{particleB} = r^2_{particleB} \cdot \pi - A_{overlapAB}$$

Equation 132: Area of particle second nearest to the laser source

$$A_{particleC} = 0$$

Equation 133: Area of particle farthest from the laser source

for (a) (small particle in B position)

$$A_{particleB} = 0$$

Equation 134: Area of particle second nearest to the laser source

$$A_{particleC} = r^2_{particleC} \cdot \pi - A_{overlapAC}$$

Equation 135: Area of particle farthest from the laser source

for (b)

$$A_{particleB} = r^2_{particleB} \cdot \pi - A_{overlapAB}$$

Equation 136: Area of particle second nearest to the laser source

$$A_{particleC} = r^2_{particleC} \cdot \pi - A_{overlapAC} - A_{particleB}$$

Equation 137: Area of particle farthest from the laser source

for (c)

$$A_{particleB} = r^2_{particleB} \cdot \pi$$

Equation 138: Area of particle second nearest to the laser source

$$A_{particleC} = r^2_{particleC} \cdot \pi - A_{overlapAC} - A_{particleB}$$

Equation 139: Area of particle farthest from the laser source

for (d)

$$A_{particleB} = r^2_{particleB} \cdot \pi$$

Equation 140: Area of particle second nearest to the laser source

$$A_{particleC} = r^2_{particleC} \cdot \pi - A_{overlapBC}$$

Equation 141: Area of particle farthest from the laser source

for (e)

$$A_{particleB} = r^2_{particleB} \cdot \pi - A_{overlapAB}$$

Equation 142: Area of particle second nearest to the laser source

$$A_{particleC} = \left(r^2_{particleB} + r^2_{particleC}\right) \cdot \pi - A_{overlapBC} - A_{particleA} - A_{particleB}$$

Equation 143: Area of particle farthest from the laser source

for (f)

$$A_{particleB} = r^2_{particleB} \cdot \pi$$

Equation 144: Area of particle second nearest to the laser source

$$A_{particleC} = \left(r^2_{particleA} + r^2_{particleC}\right) \cdot \pi - A_{overlapAC} - A_{particleB}$$

Equation 145: Area of particle farthest from the laser source

CASE 5:

This case is very similar to case 4, with the additional condition, that the smallest circle must lie within the intersection area of the other two. This fact however simplifies the number of possible sub-scenarios immensely because the smallest particle is only then not covered, if it lies on top; thus only 3 scenarios need to be differentiated:

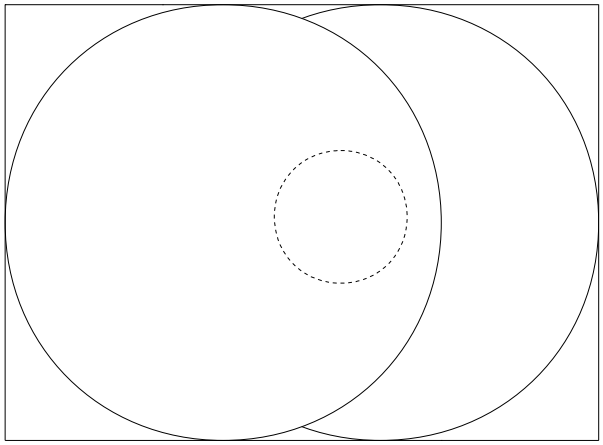


Figure 108: (a) small circle in position C and (b) small circle in position B

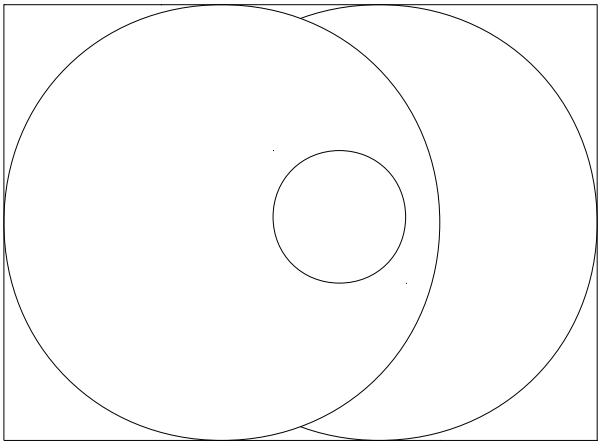


Figure 109: (c)

for all cases:

$$A_{particleA} = r^2_{particleA} \cdot \pi$$

Equation 146: Area of particle nearest to the laser source

for (a):

$$A_{particleB} = r^2_{particleB} \cdot \pi - A_{overlapAB}$$

Equation 147: Area of particle second nearest to the laser source

$$A_{particleC} = 0$$

Equation 148: Area of particle farthest from the laser source

for (b):

$$A_{particleB} = 0$$

Equation 149: Area of particle second nearest to the laser source

$$A_{particleC} = r^2_{particleC} \cdot \pi - A_{overlapBC}$$

Equation 150: Area of particle farthest from the laser source

for (c):

$$A_{particleB} = r^2_{particleB} \cdot \pi - A_{particleA}$$

Equation 151: Area of particle second nearest to the laser source

$$A_{particleC} = r^2_{particleC} \cdot \pi - A_{overlapBC}$$

Equation 152: Area of particle farthest from the laser source

CASE 8:

The algorithm now differentiates between the last three possible arrangements of circles by checking how many of the pairs of circles intersect with each other. Of course three intersections are not possible, otherwise it would already have been classified as case 1, 2, 3 or 7. Two separate intersections result in the configuration being classified as case 8 and having these possible sub-configurations:

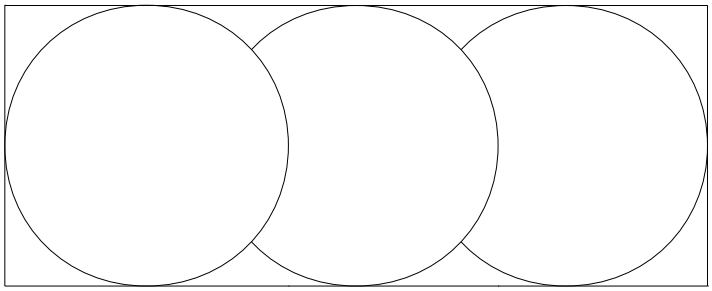


Figure 110: (a)

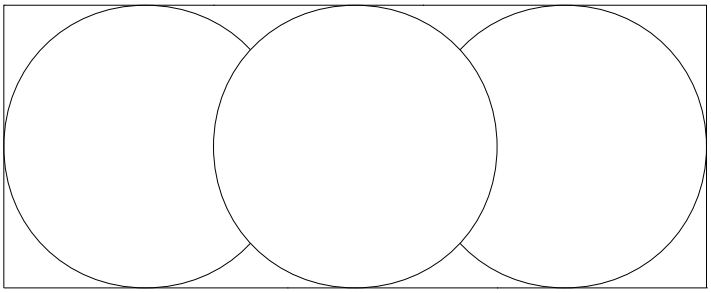


Figure 111: (b)

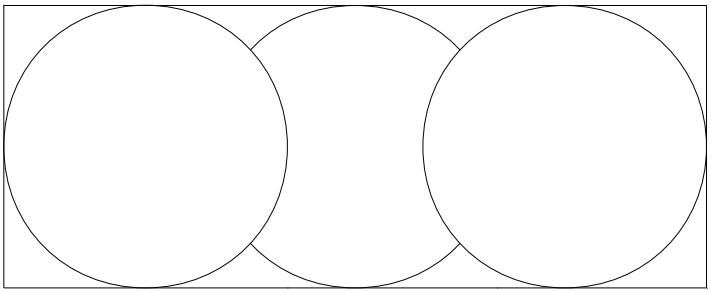


Figure 112: (c)

for all cases:

$$A_{particleA} = r^2_{particleA} \cdot \pi$$

Equation 153: Area of particle nearest to the laser source

for (a):

$$A_{particleB} = r^2_{particleB} \cdot \pi - A_{overlapAB}$$

Equation 154: Area of particle second nearest to the laser source

$$A_{particleC} = r^2_{particleC} \cdot \pi - A_{overlapBC}$$

Equation 155: Area of particle farthest from the laser source

for (b):

$$A_{particleB} = r^2_{particleB} \cdot \pi - A_{overlapAB}$$

Equation 156: Area of particle second nearest to the laser source

$$A_{particleC} = r^2_{particleC} \cdot \pi - A_{overlapAC}$$

Equation 157: Area of particle farthest from the laser source

for (c):

$$A_{particleB} = r^2_{particleB} \cdot \pi$$

Equation 158: Area of particle second nearest to the laser source

$$A_{particleC} = r^2_{particleC} \cdot \pi - A_{overlapAC} - A_{overlapBC}$$

Equation 159: Area of particle farthest from the laser source

CASE 9

This configuration of circles is similar to the previous one, with the exception, that now only two of the circles either partially or completely overlap each other. Thus 3 sub-scenarios can be differentiated for calculating the area:

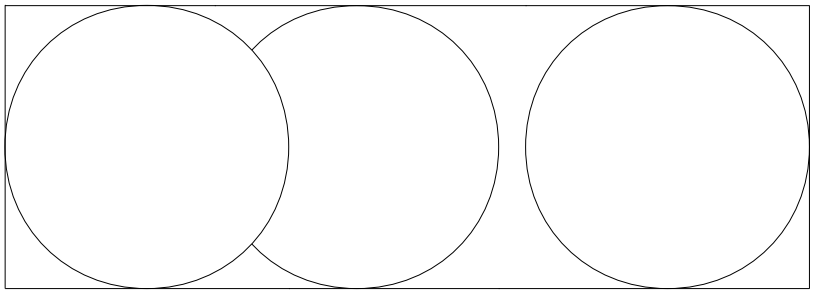


Figure 113: (a)

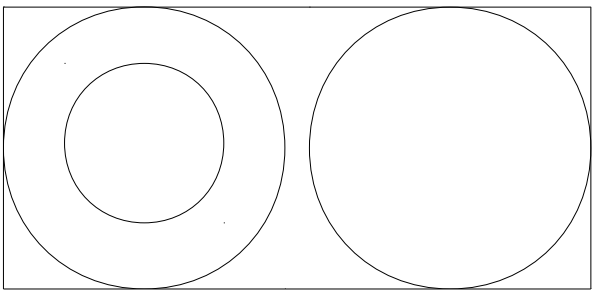


Figure 114: (b)

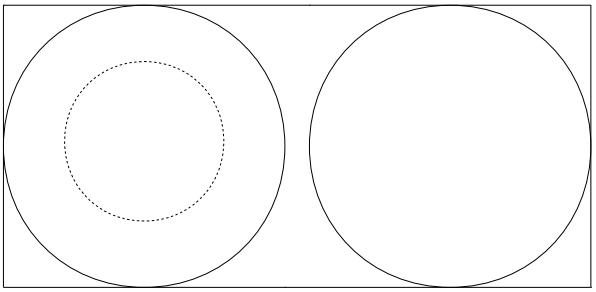


Figure 115: (c)

Assuming, for simpler presentation of the equations, that the non intersecting particle is in position C, the equations become for all cases:

$$A_{particleA} = r^2_{particleA} \cdot \pi$$

Equation 160: Area of the particle nearest to the laser source

$$A_{particleB} = r^2_{particleB} \cdot \pi - A_{overlapAB}$$

Equation 161: Area of particle second nearest to the laser source

$$A_{particleC} = r^2_{particleC} \cdot \pi$$

Equation 162: Area of particle farthest from the laser source

CASE 10:

The last scenario is made up of three particles not intersecting with each other in the projected plane. This is the simplest to calculate and the equations for the unobscured area:

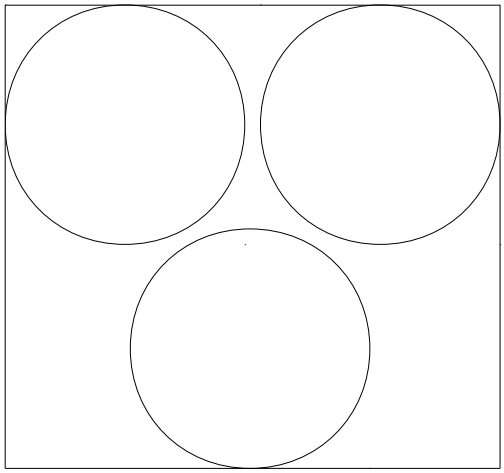


Figure 116: Case 10 schematic representation

$$A_{particleA} = r^2_{particleA} \cdot \pi$$

Equation 163: Area of particle nearest to the laser source

$$A_{particleB} = r^2_{particleB} \cdot \pi$$

Equation 164: Area of particle second nearest to the laser source

$$A_{particleC} = r^2_{particleC} \cdot \pi$$

Equation 165: Area of particle farthest from the laser source

6.3 Bash script for case generation LMD 1 - project

```
#!/bin/bash

WM_PROJECT_DIR=/opt/openfoam6

source $WM_PROJECT_DIR/etc/bashrc

echo "version: " && foamVersion


ORIG="cladding_template"

ORIG_STEP="step0"


CLOUD_PROP="constant/cloudProperties"

BEAM_PROP="constant/laser/step0/beam0/beamProperties"

MOVEMENT_DICT="constant/laser/step0/movementProperties"

LASER_FOLD="constant/laser"

GEOMETRY="system/geometryDefinitions"

CONTROL="system/controlDict.orig"


SPOT_SIZE="0.000948"

FOCUS_HEIGHT="0.0225"

M_SQUARE="61.1"


CONDITION=( 1 2 3 4 5 6 7 8 9 )

LASER_POWER=( 1000 750 1250 1000 1000 1000 1000 1000 1000 )

VELOCITY=( 0.00116666666666666667 0.00116666666666666667 0.00116666666666666667 0.00116666666666666667
0.00116666666666666667 0.00116666666666666667 0.00116666666666666667 0.00116666666666666667
0.00116666666666666667 )

POWDER_FEED_RATE=( 2.66666666666667E-05 2.66666666666667E-05 2.66666666666667E-05 2.66666666666667E-05
1.66666666666667E-05 3.66666666666667E-05 2.66666666666667E-05 2.66666666666667E-05 2.66666666666667E-05 )


for i in "${CONDITION[@]}"
```

```
do

    INDEX=$((($i - 1))

    DESTINATION="LMD1_$i"

    POWER_VALUE=${LASER_POWER[$INDEX]}
    VELOCITY_VALUE=${VELOCITY[$INDEX]}
    PMFR_VALUE=${POWDER_FEED_RATE[$INDEX]}

    echo -e "\n$DESTINATION:"

    cp -r $ORIG $DESTINATION

    foamDictionary -entry powderMassFeedRate -set $PMFR_VALUE -disableFunctionEntries $DESTINATION/$GEOMETRY
    foamDictionary -entry VELOCITY -set $VELOCITY_VALUE -disableFunctionEntries $DESTINATION/$GEOMETRY
    foamDictionary -entry POWER -set $POWER_VALUE -disableFunctionEntries $DESTINATION/$BEAM_PROP

    #change geometry def
    foamDictionary -entry POS0_X -set "0" -disableFunctionEntries $DESTINATION/$GEOMETRY

    foamDictionary -entry POS0_Y -set "-0.003" -disableFunctionEntries $DESTINATION/$GEOMETRY

    foamDictionary -entry POS0_Z -set "-0.035" -disableFunctionEntries $DESTINATION/$GEOMETRY
    foamDictionary -entry POS1_Z -set "0.035" -disableFunctionEntries $DESTINATION/$GEOMETRY

    #set simulation endTime, so no need to change deltaTime3 and up or other positions, since the sim will be aborted before.
    endTime=$(bc <<< "scale=10;0.07/$VELOCITY+1000")
    foamDictionary -entry endTime -set $endTime -disableFunctionEntries $DESTINATION/$CONTROL
```

```
foamDictionary -entry MAX_PARTICLE_SIZE -set "0.000174265755695" -disableFunctionEntries $DESTINATION/$CLOUD_PROP

foamDictionary -entry MIN_PARTICLE_SIZE -set "2.26027440321292E-05" -disableFunctionEntries $DESTINATION/$CLOUD_PROP

foamDictionary -entry INJECTOR.sizeDistribution -set "
        {
        type general;
        generalDistribution
        {
        distribution
        (
        ( 2.26027440321292E-05 9.20531732864767E-09 )
        ( 2.56804037382657E-05 5.45362263907675E-07 )
        ( 2.91771271321259E-05 0.006733274466046E-02 )
        ( 3.31499752247168E-05 0.139121669487172E-02 )
        ( 3.76637786312196E-05 1.1016614018375E-02 )
        ( 4.27921955043824E-05 4.239941383589E-02 )
        ( 4.86189135194051E-05 10.1285560781254E-02 )
        ( 5.52390155248126E-05 16.9884477204441E-02 )
        ( 6.27605311445845E-05 21.2207254309487E-02 )
        ( 7.13061996512424E-05 20.2005305727398E-02 )
        ( 8.10154728772013E-05 14.6075816533004E-02 )
        ( 9.2046790848742E-05 7.78439905710275E-02 )
        ( 0.000104580167277 2.85114754009532E-02 )
        ( 0.0001188201271 0.623055946460233E-02 )
        ( 0.000134999044002 0.062504912998508E-02 )
        ( 0.000153380932391 0.002223151590274E-02 )
        ( 0.000174265755695 3.88392903024245E-09 )
        );
        }
    }" -disableFunctionEntries $DESTINATION/$CLOUD_PROP
```

```
foamDictionary -entry subModels.injectionModels.modelA.sizeDistribution -set "
        {
        type general;
        generalDistribution
        {
        distribution
        (
        ( 2.26027440321292E-05 9.20531732864767E-09 )
        ( 2.56804037382657E-05 5.45362263907675E-07 )
        ( 2.91771271321259E-05 0.006733274466046E-02 )
        ( 3.31499752247168E-05 0.139121669487172E-02 )
        ( 3.76637786312196E-05 1.1016614018375E-02 )
        ( 4.27921955043824E-05 4.239941383589E-02 )
        ( 4.86189135194051E-05 10.1285560781254E-02 )
        ( 5.52390155248126E-05 16.9884477204441E-02 )
        ( 6.27605311445845E-05 21.2207254309487E-02 )
        ( 7.13061996512424E-05 20.2005305727398E-02 )
        ( 8.10154728772013E-05 14.6075816533004E-02 )
        ( 9.2046790848742E-05 7.78439905710275E-02 )
        ( 0.000104580167277 2.85114754009532E-02 )
        ( 0.0001188201271 0.623055946460233E-02 )
        ( 0.000134999044002 0.062504912998508E-02 )
        ( 0.000153380932391 0.002223151590274E-02 )
        ( 0.000174265755695 3.88392903024245E-09 )
        );
        }
    }" -disableFunctionEntries $DESTINATION/$CLOUD_PROP


#change laser position
```

```
        foamDictionary -entry straightCoeffs.lasPosStart -set "(\$POS0_X \$PLATE_OFFSET \$POS0_Z)" $DESTINATION/$LASER_FOLD/step0/movementProperties -disableFunctionEntries

        foamDictionary -entry straightCoeffs.lasPosEnd -set "(\$POS0_X \$PLATE_OFFSET \$POS1_Z)" $DESTINATION/$LASER_FOLD/step0/movementProperties -disableFunctionEntries


    #change laser beam properties

    foamDictionary -entry SPOT_SIZE -set $SPOT_SIZE -disableFunctionEntries $DESTINATION/$BEAM_PROP

    foamDictionary -entry FOCUS_HEIGHT -set $FOCUS_HEIGHT -disableFunctionEntries $DESTINATION/$BEAM_PROP

    foamDictionary -entry M_SQUARE -set $M_SQUARE -disableFunctionEntries $DESTINATION/$BEAM_PROP


done
```

6.4 Bash script for case generation LMD 2 – project

```
#!/bin/bash

WM_PROJECT_DIR=/opt/openfoam6

source $WM_PROJECT_DIR/etc/bashrc

echo "version: " && foamVersion


ORIG="cladding_template"

ORIG_STEP="step0"


CLOUD_PROP="constant/cloudProperties"

BEAM_PROP="constant/laser/step0/beam0/beamProperties"

MOVEMENT_DICT="constant/laser/step0/movementProperties"

LASER_FOLD="constant/laser"

GEOMETRY="system/geometryDefinitions"

CONTROL="system/controlDict.orig"


SPOT_SIZE="0.000916"

FOCUS_HEIGHT="0.027"

M_SQUARE="55.8"


CONDITION=( 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 )

LASER_POWER=( 1000 1000 750 750 1250 1250 1000 1000 1000 1000 1000 1000 500 1000 1000 )

VELOCITY=( 0.00116666666666666667 0.00116666666666666667 0.00116666666666666667 0.00116666666666666667
0.00116666666666666667 0.00116666666666666667 0.00116666666666666667 0.00116666666666666667
0.00116666666666666667 0.00116666666666666667 0.00116666666666666667 0.00116666666666666667
0.00116666666666666667 0.00116666666666666667 0.00116666666666666667 )

POWDER_FEED_RATE=( 2.66666666666667E-05 2.66666666666667E-05 2.66666666666667E-05 2.66666666666667E-05
2.66666666666667E-05 2.66666666666667E-05 2.66666666666667E-05 2.66666666666667E-05 1.66666666666667E-05
1.66666666666667E-05 3.66666666666667E-05 3.66666666666667E-05 2.66666666666667E-05 2.66666666666667E-05
2.66666666666667E-05 )
```

```
#0 = lateral 1 = vertical 2 = without powder

PASSES=( 0 1 0 1 0 1 0 1 0 1 0 1 1 0 2 )

SHIFT_AMOUNT_1=( 2 0.9 1.7 1.3 2.5 0.8 2 0.8 2 0.7 2.3 1.5 0.9 1.9 0 )

SHIFT_AMOUNT_2=( 1.8 1.1 1.8 0.9 2.3 0.9 1.8 0.7 1.8 0.8 1.8 1.1 0.6 1.7 0 )


for i in "${CONDITION[@]}"

do


   INDEX=$((${i} - 1))


   DESTINATION="LMD2_${i}"


   POWER_VALUE=${LASER_POWER[$INDEX]}

   VELOCITY_VALUE=${VELOCITY[$INDEX]}

   PMFR_VALUE=${POWDER_FEED_RATE[$INDEX]}

   PASSES_VALUE=${PASSES[$INDEX]}


   echo -e "\n$DESTINATION:"


   cp -r $ORIG $DESTINATION


   foamDictionary -entry powderMassFeedRate -set $PMFR_VALUE -disableFunctionEntries $DESTINATION/$GEOMETRY

   foamDictionary -entry VELOCITY -set $VELOCITY_VALUE -disableFunctionEntries $DESTINATION/$GEOMETRY

   foamDictionary -entry POWER -set $POWER_VALUE -disableFunctionEntries $DESTINATION/$BEAM_PROP


   #change laser beam properties

   foamDictionary -entry SPOT_SIZE -set $SPOT_SIZE -disableFunctionEntries $DESTINATION/$BEAM_PROP

   foamDictionary -entry FOCUS_HEIGHT -set $FOCUS_HEIGHT -disableFunctionEntries $DESTINATION/$BEAM_PROP

   foamDictionary -entry M_SQUARE -set $M_SQUARE -disableFunctionEntries $DESTINATION/$BEAM_PROP
```

```
#set simulation endTime

endTime=$(bc <<< "scale=10;((0.08+0.06+0.04)/$VELOCITY)+3*1000")

foamDictionary -entry endTime -set $endTime -disableFunctionEntries $DESTINATION/$CONTROL


foamDictionary -entry MAX_PARTICLE_SIZE -set "0.000272984826794" -disableFunctionEntries $DESTINATION/$CLOUD_PROP

foamDictionary -entry MIN_PARTICLE_SIZE -set "2.74287654351958E-05" -disableFunctionEntries $DESTINATION/$CLOUD_PROP

foamDictionary -entry INJECTOR_THICKNESS -set "0.000273"  $DESTINATION/$GEOMETRY -disableFunctionEntries

foamDictionary -entry PLATE_OFFSET_NEG -add "#calc \"-\$PLATE_OFFSET\";" -disableFunctionEntries $DESTINATION/$GEOMETRY


#lateral


if [ $PASSES_VALUE == 0 ];

then


#create additional passes

counter=1

while [ $counter -le 2 ]

  do

  STEP="step$counter"

  cp -r $DESTINATION/$LASER_FOLD/$ORIG_STEP $DESTINATION/$LASER_FOLD/$STEP

  ((counter++))

done


SHIFTX1=${SHIFT_AMOUNT_1[$INDEX]}

SHIFTX1M=$(bc <<< "scale=10;$SHIFTX1*0.001")

SHIFTX2=$(bc <<< "${SHIFT_AMOUNT_1[$INDEX]}+${SHIFT_AMOUNT_2[$INDEX]}")

SHIFTX2M=$(bc <<< "scale=10;$SHIFTX2*0.001")
```

```
#change geometry def

foamDictionary -entry POS0_X -set "0" -disableFunctionEntries $DESTINATION/$GEOMETRY

foamDictionary -entry POS1_X -set "$SHIFTX1M" -disableFunctionEntries $DESTINATION/$GEOMETRY

foamDictionary -entry POS2_X -set "$SHIFTX2M" -disableFunctionEntries $DESTINATION/$GEOMETRY


foamDictionary -entry POS0_Y -set "-0.003" -disableFunctionEntries $DESTINATION/$GEOMETRY

foamDictionary -entry POS1_Y -set "-0.003" -disableFunctionEntries $DESTINATION/$GEOMETRY

foamDictionary -entry POS2_Y -set "-0.003" -disableFunctionEntries $DESTINATION/$GEOMETRY


foamDictionary -entry POS0_Z -set "-0.04" -disableFunctionEntries $DESTINATION/$GEOMETRY

foamDictionary -entry POS1_Z -set "0.04" -disableFunctionEntries $DESTINATION/$GEOMETRY

foamDictionary -entry POS2_Z -set "0.02" -disableFunctionEntries $DESTINATION/$GEOMETRY

foamDictionary -entry POS3_Z -set "0" -disableFunctionEntries $DESTINATION/$GEOMETRY


#change laser position

foamDictionary -entry straightCoeffs.lasPosStart -set "(\$POS0_X \$PLATE_OFFSET_NEG \$POS0_Z)" $DESTINATION/$LASER_FOLD/step0/movementProperties -disableFunctionEntries

foamDictionary -entry straightCoeffs.lasPosEnd -set "(\$POS0_X \$PLATE_OFFSET_NEG \$POS1_Z)" $DESTINATION/$LASER_FOLD/step0/movementProperties -disableFunctionEntries

foamDictionary -entry straightCoeffs.lasPosStart -set "(\$POS1_X \$PLATE_OFFSET_NEG \$POS0_Z)" $DESTINATION/$LASER_FOLD/step1/movementProperties -disableFunctionEntries

foamDictionary -entry straightCoeffs.lasPosEnd -set "(\$POS1_X \$PLATE_OFFSET_NEG \$POS2_Z)" $DESTINATION/$LASER_FOLD/step1/movementProperties -disableFunctionEntries

foamDictionary -entry straightCoeffs.lasPosStart -set "(\$POS2_X \$PLATE_OFFSET_NEG \$POS0_Z)" $DESTINATION/$LASER_FOLD/step2/movementProperties -disableFunctionEntries

foamDictionary -entry straightCoeffs.lasPosEnd -set "(\$POS2_X \$PLATE_OFFSET_NEG \$POS3_Z)" $DESTINATION/$LASER_FOLD/step2/movementProperties -disableFunctionEntries


#change injector location

foamDictionary -entry INJECTOR.sizeDistribution -set "
        {
        type general;
        generalDistribution
```

```
        {
        distribution
        (
        ( 2.74287654351958E-05 0.00010223557394 )
        ( 3.11635511784214E-05 0.02486726451703 )
        ( 3.54068769279682E-05 0.35190268319800 )
        ( 4.0227987067801E-05 1.43929765724892 )
        ( 4.57055545118938E-05 3.49488098873001 )
        ( 5.19289645221089E-05 6.33925220768467 )
        ( 5.89997733346985E-05 9.44787848487467 )
        ( 6.70333653979134E-05 12.13011491988010 )
        ( 7.61608362642218E-05 13.75910582358510 )
        ( 8.65311318629717E-05 13.96697450606770 )
        ( 9.83134790630505E-05 12.74315901079730 )
        ( 0.000111700147188 10.41630317308820 )
        ( 0.000126909585551 7.53843853018428 )
        ( 0.000144189988196 4.71655082975088 )
        ( 0.000163823344043 2.43931608868185 )
        ( 0.000186130038495 0.95110974631168 )
        ( 0.000211474081623 0.22322304092568 )
        ( 0.000240269048242 0.01728391966330 )
        ( 0.000272984826794 0.00023888923669 )
        );
        }
    }" -disableFunctionEntries $DESTINATION/$CLOUD_PROP

foamDictionary -entry subModels.injectionModels.modelA.sizeDistribution -set "
        {
        type general;
        generalDistribution
```

```
        {
        distribution
        (
        ( 2.74287654351958E-05 0.00010223557394 )
        ( 3.11635511784214E-05 0.02486726451703 )
        ( 3.54068769279682E-05 0.35190268319800 )
        ( 4.0227987067801E-05 1.43929765724892 )
        ( 4.57055545118938E-05 3.49488098873001 )
        ( 5.19289645221089E-05 6.33925220768467 )
        ( 5.89997733346985E-05 9.44787848487467 )
        ( 6.70333653979134E-05 12.13011491988010 )
        ( 7.61608362642218E-05 13.75910582358510 )
        ( 8.65311318629717E-05 13.96697450606770 )
        ( 9.83134790630505E-05 12.74315901079730 )
        ( 0.000111700147188 10.41630317308820 )
        ( 0.000126909585551 7.53843853018428 )
        ( 0.000144189988196 4.71655082975088 )
        ( 0.000163823344043 2.43931608868185 )
        ( 0.000186130038495 0.95110974631168 )
        ( 0.000211474081623 0.22322304092568 )
        ( 0.000240269048242 0.01728391966330 )
        ( 0.000272984826794 0.00023888923669 )
        );
        }
    }" -disableFunctionEntries $DESTINATION/$CLOUD_PROP


foamDictionary -entry subModels.injectionModels.modelB -add "{
    \$INJECTOR;
    SOI             \$deltaTime2;
    duration        #calc \"((\$POS2_Z)-(\$POS0_Z))/\$VELOCITY\";
```

```
type            coneNozzleInjection;

injectionMethod     disc;


positions       table

(

    (   \$deltaTime2  (\$POS1_X    \$POS0_Y   \$POS0_Z))

    (   \$deltaTime3  (\$POS1_X    \$POS0_Y   \$POS2_Z))

);


direction       (0 -1 0);


innerDiameter       \$INNER_DIAM;

outerDiameter       \$OUTER_DIAM;


thetaInner      \$INNER_ANGLE;

thetaOuter      \$OUTER_ANGLE;


flowType        pressureDrivenVelocity;

Pinj            1.15e5;

flowRateProfile     one;


massFlowRate        0.0001092667213; // kg_argon/s  // WARNING: only if (coupled==false)


sizeDistribution

{

    type general;

    generalDistribution

    {

    distribution
```

```
            (
            ( 2.74287654351958E-05 0.00010223557394 )
            ( 3.11635511784214E-05 0.02486726451703 )
            ( 3.54068769279682E-05 0.35190268319800 )
            ( 4.0227987067801E-05 1.43929765724892 )
            ( 4.57055545118938E-05 3.49488098873001 )
            ( 5.19289645221089E-05 6.33925220768467 )
            ( 5.89997733346985E-05 9.44787848487467 )
            ( 6.70333653979134E-05 12.13011491988010 )
            ( 7.61608362642218E-05 13.75910582358510 )
            ( 8.65311318629717E-05 13.96697450606770 )
            ( 9.83134790630505E-05 12.74315901079730 )
            ( 0.000111700147188 10.41630317308820 )
            ( 0.000126909585551 7.53843853018428 )
            ( 0.000144189988196 4.71655082975088 )
            ( 0.000163823344043 2.43931608868185 )
            ( 0.000186130038495 0.95110974631168 )
            ( 0.000211474081623 0.22322304092568 )
            ( 0.000240269048242 0.01728391966330 )
            ( 0.000272984826794 0.00023888923669 )
            );
            }
        }
        parcelsPerSecond \$PPS;
        powderMassFeedRate \$powderMassFeedRate;

    }" -disableFunctionEntries $DESTINATION/$CLOUD_PROP

foamDictionary -entry subModels.injectionModels.modelC -add "{
        \$INJECTOR;
```

```
SOI             \$deltaTime4;

duration        #calc \"((\$POS3_Z)-(\$POS0_Z))/\$VELOCITY\";


type            coneNozzleInjection;

injectionMethod     disc;


positions       table

(

   (   \$deltaTime4  (\$POS2_X    \$POS0_Y   \$POS0_Z))

   (   \$deltaTime5  (\$POS2_X    \$POS0_Y   \$POS3_Z))

);


direction       (0 -1 0);


innerDiameter       \$INNER_DIAM;

outerDiameter       \$OUTER_DIAM;


thetaInner      \$INNER_ANGLE;

thetaOuter      \$OUTER_ANGLE;


flowType        pressureDrivenVelocity;

Pinj            1.15e5;

flowRateProfile     one;


massFlowRate        0.0001092667213; // kg_argon/s  // WARNING: only if (coupled==false)


sizeDistribution

{

   type general;

   generalDistribution
```

```
        {
        distribution
        (
        ( 2.74287654351958E-05 0.00010223557394 )
        ( 3.11635511784214E-05 0.02486726451703 )
        ( 3.54068769279682E-05 0.35190268319800 )
        ( 4.0227987067801E-05 1.43929765724892 )
        ( 4.57055545118938E-05 3.49488098873001 )
        ( 5.19289645221089E-05 6.33925220768467 )
        ( 5.89997733346985E-05 9.44787848487467 )
        ( 6.70333653979134E-05 12.13011491988010 )
        ( 7.61608362642218E-05 13.75910582358510 )
        ( 8.65311318629717E-05 13.96697450606770 )
        ( 9.83134790630505E-05 12.74315901079730 )
        ( 0.000111700147188 10.41630317308820 )
        ( 0.000126909585551 7.53843853018428 )
        ( 0.000144189988196 4.71655082975088 )
        ( 0.000163823344043 2.43931608868185 )
        ( 0.000186130038495 0.95110974631168 )
        ( 0.000211474081623 0.22322304092568 )
        ( 0.000240269048242 0.01728391966330 )
        ( 0.000272984826794 0.00023888923669 )
        );
        }
    }
    parcelsPerSecond \$PPS;
    powderMassFeedRate \$powderMassFeedRate;

}" -disableFunctionEntries $DESTINATION/$CLOUD_PROP
```

```
#vertical

elif [ $PASSES_VALUE == 1 ]

then


#increase vertical mesh heigth by 3mm

foamDictionary -entry PLATE_OFFSET -set "15e-3" -disableFunctionEntries $DESTINATION/$GEOMETRY #unnecessary if already defined as 15e-3 in orig


#create additional passes

counter=1

while [ $counter -le 2 ]

  do

  STEP="step$counter"

  cp -r $DESTINATION/$LASER_FOLD/$ORIG_STEP $DESTINATION/$LASER_FOLD/$STEP

  ((counter++))

done


#some calculations

SHIFTY0M=$(bc <<< "scale=10;-0.015+0.012") #12mm above plate

SHIFTY1=$(bc <<< "-15+${SHIFT_AMOUNT_1[$INDEX]}+12")

SHIFTY1M=$(bc <<< "scale=10;$SHIFTY1*0.001") #in m

SHIFTY2=$(bc <<< "$SHIFTY1+${SHIFT_AMOUNT_2[$INDEX]}")

SHIFTY2M=$(bc <<< "scale=10;$SHIFTY2*0.001") #in m


LASERSHIFTY1=$(bc <<< "scale=10;$SHIFTY1M-0.012") #in m

LASERSHIFTY2=$(bc <<< "scale=10;$SHIFTY2M-0.012") #in m


#change geometry def

foamDictionary -entry POS0_X -set "0" -disableFunctionEntries $DESTINATION/$GEOMETRY

foamDictionary -entry POS1_X -set "0" -disableFunctionEntries $DESTINATION/$GEOMETRY
```

```
foamDictionary -entry POS2_X -set "0" -disableFunctionEntries $DESTINATION/$GEOMETRY


foamDictionary -entry POS0_Y -set "$SHIFTY0M" -disableFunctionEntries $DESTINATION/$GEOMETRY

foamDictionary -entry POS1_Y -set "$SHIFTY1M" -disableFunctionEntries $DESTINATION/$GEOMETRY

foamDictionary -entry POS2_Y -set "$SHIFTY2M" -disableFunctionEntries $DESTINATION/$GEOMETRY


foamDictionary -entry POS0_Z -set "-0.04" -disableFunctionEntries $DESTINATION/$GEOMETRY

foamDictionary -entry POS1_Z -set "0.04" -disableFunctionEntries $DESTINATION/$GEOMETRY

foamDictionary -entry POS2_Z -set "0.02" -disableFunctionEntries $DESTINATION/$GEOMETRY

foamDictionary -entry POS3_Z -set "0" -disableFunctionEntries $DESTINATION/$GEOMETRY


#change laser position

        foamDictionary -entry straightCoeffs.lasPosStart -set "(\$POS0_X \$PLATE_OFFSET_NEG \$POS0_Z)"
$DESTINATION/$LASER_FOLD/step0/movementProperties -disableFunctionEntries

        foamDictionary -entry straightCoeffs.lasPosEnd -set "(\$POS0_X \$PLATE_OFFSET_NEG \$POS1_Z)"
$DESTINATION/$LASER_FOLD/step0/movementProperties -disableFunctionEntries

        foamDictionary -entry straightCoeffs.lasPosStart -set "(\$POS0_X $LASERSHIFTY1 \$POS0_Z)"
$DESTINATION/$LASER_FOLD/step1/movementProperties -disableFunctionEntries

        foamDictionary -entry straightCoeffs.lasPosEnd -set "(\$POS0_X $LASERSHIFTY1 \$POS2_Z)"
$DESTINATION/$LASER_FOLD/step1/movementProperties -disableFunctionEntries

        foamDictionary -entry straightCoeffs.lasPosStart -set "(\$POS0_X $LASERSHIFTY2 \$POS0_Z)"
$DESTINATION/$LASER_FOLD/step2/movementProperties -disableFunctionEntries

        foamDictionary -entry straightCoeffs.lasPosEnd -set "(\$POS0_X $LASERSHIFTY2 \$POS3_Z)"
$DESTINATION/$LASER_FOLD/step2/movementProperties -disableFunctionEntries


#change injector location

foamDictionary -entry INJECTOR.sizeDistribution -set "

        {

        type general;

        generalDistribution

        {

        distribution

        (
```

(2.74287654351958E-05 0.00010223557394)

(3.11635511784214E-05 0.02486726451703)

(3.54068769279682E-05 0.35190268319800)

(4.0227987067801E-05 1.43929765724892)

(4.57055545118938E-05 3.49488098873001)

(5.19289645221089E-05 6.33925220768467)

(5.89997733346985E-05 9.44787848487467)

(6.70333653979134E-05 12.13011491988010)

(7.61608362642218E-05 13.75910582358510)

(8.65311318629717E-05 13.96697450606770)

(9.83134790630505E-05 12.74315901079730)

(0.000111700147188 10.41630317308820)

(0.000126909585551 7.53843853018428)

(0.000144189988196 4.71655082975088)

(0.000163823344043 2.43931608868185)

(0.000186130038495 0.95110974631168)

(0.000211474081623 0.22322304092568)

(0.000240269048242 0.01728391966330)

(0.000272984826794 0.00023888923669)

);

}

}" -disableFunctionEntries $DESTINATION/$CLOUD_PROP

foamDictionary -entry *subModels.injectionModels.modelA.sizeDistribution* -**set** "

{

type general;

generalDistribution

{

distribution

(

```
( 2.74287654351958E-05 0.00010223557394 )
( 3.11635511784214E-05 0.02486726451703 )
( 3.54068769279682E-05 0.35190268319800 )
( 4.0227987067801E-05 1.43929765724892 )
( 4.57055545118938E-05 3.49488098873001 )
( 5.19289645221089E-05 6.33925220768467 )
( 5.89997733346985E-05 9.44787848487467 )
( 6.70333653979134E-05 12.13011491988010 )
( 7.61608362642218E-05 13.75910582358510 )
( 8.65311318629717E-05 13.96697450606770 )
( 9.83134790630505E-05 12.74315901079730 )
( 0.000111700147188 10.41630317308820 )
( 0.000126909585551 7.53843853018428 )
( 0.000144189988196 4.71655082975088 )
( 0.000163823344043 2.43931608868185 )
( 0.000186130038495 0.95110974631168 )
( 0.000211474081623 0.22322304092568 )
( 0.000240269048242 0.01728391966330 )
( 0.000272984826794 0.00023888923669 )
);
}
}" -disableFunctionEntries $DESTINATION/$CLOUD_PROP

foamDictionary -entry subModels.injectionModels.modelB -add "{
    \$INJECTOR;
    SOI             \$deltaTime2;
    duration        #calc \"((\$POS2_Z)-(\$POS0_Z))/\$VELOCITY\";

    type            coneNozzleInjection;
    injectionMethod     disc;
```

```
positions           table
(
    (   \$deltaTime2  (\$POS0_X    \$POS1_Y   \$POS0_Z))
    (   \$deltaTime3  (\$POS0_X    \$POS1_Y   \$POS2_Z))
);

direction           (0 -1 0);

innerDiameter       \$INNER_DIAM;
outerDiameter       \$OUTER_DIAM;

thetaInner          \$INNER_ANGLE;
thetaOuter          \$OUTER_ANGLE;

flowType            pressureDrivenVelocity;
Pinj                1.15e5;
flowRateProfile     one;

massFlowRate        0.0001092667213; // kg_argon/s  // WARNING: only if (coupled==false)

sizeDistribution
{
    type general;
    generalDistribution
    {
    distribution
    (
    ( 2.74287654351958E-05 0.00010223557394 )
    ( 3.11635511784214E-05 0.02486726451703 )
```

```
            ( 3.54068769279682E-05 0.35190268319800 )
            ( 4.0227987067801E-05 1.43929765724892 )
            ( 4.57055545118938E-05 3.49488098873001 )
            ( 5.19289645221089E-05 6.33925220768467 )
            ( 5.89997733346985E-05 9.44787848487467 )
            ( 6.70333653979134E-05 12.13011491988010 )
            ( 7.61608362642218E-05 13.75910582358510 )
            ( 8.65311318629717E-05 13.96697450606770 )
            ( 9.83134790630505E-05 12.74315901079730 )
            ( 0.000111700147188 10.41630317308820 )
            ( 0.000126909585551 7.53843853018428 )
            ( 0.000144189988196 4.71655082975088 )
            ( 0.000163823344043 2.43931608868185 )
            ( 0.000186130038495 0.95110974631168 )
            ( 0.000211474081623 0.22322304092568 )
            ( 0.000240269048242 0.01728391966330 )
            ( 0.000272984826794 0.00023888923669 )
            );
            }
        }
        parcelsPerSecond \$parcelsPerSecond;
    }" -disableFunctionEntries $DESTINATION/$CLOUD_PROP


foamDictionary -entry subModels.injectionModels.modelC -add "{
        \$INJECTOR;
        SOI             \$deltaTime4;
        duration        #calc \"((\$POS3_Z)-(\$POS0_Z))/\$VELOCITY\";

        type            coneNozzleInjection;
        injectionMethod     disc;
```

```
positions          table
(
   (   \$deltaTime4  (\$POS0_X    \$POS2_Y   \$POS0_Z))
   (   \$deltaTime5  (\$POS0_X    \$POS2_Y   \$POS3_Z))
);

direction          (0 -1 0);

innerDiameter      \$INNER_DIAM;
outerDiameter      \$OUTER_DIAM;

thetaInner         \$INNER_ANGLE;
thetaOuter         \$OUTER_ANGLE;

flowType           pressureDrivenVelocity;
Pinj               1.15e5;
flowRateProfile    one;

massFlowRate       0.0001092667213; // kg_argon/s  // WARNING: only if (coupled==false)

sizeDistribution
{
   type general;
   generalDistribution
   {
   distribution
   (
   ( 2.74287654351958E-05 0.00010223557394 )
   ( 3.11635511784214E-05 0.02486726451703 )
```

```
            ( 3.54068769279682E-05 0.35190268319800 )
            ( 4.0227987067801E-05 1.43929765724892 )
            ( 4.57055545118938E-05 3.49488098873001 )
            ( 5.19289645221089E-05 6.33925220768467 )
            ( 5.89997733346985E-05 9.44787848487467 )
            ( 6.70333653979134E-05 12.13011491988010 )
            ( 7.61608362642218E-05 13.75910582358510 )
            ( 8.65311318629717E-05 13.96697450606770 )
            ( 9.83134790630505E-05 12.74315901079730 )
            ( 0.000111700147188 10.41630317308820 )
            ( 0.000126909585551 7.53843853018428 )
            ( 0.000144189988196 4.71655082975088 )
            ( 0.000163823344043 2.43931608868185 )
            ( 0.000186130038495 0.95110974631168 )
            ( 0.000211474081623 0.22322304092568 )
            ( 0.000240269048242 0.01728391966330 )
            ( 0.000272984826794 0.00023888923669 )
            );
            }
        }
        parcelsPerSecond \$parcelsPerSecond;
    }" -disableFunctionEntries $DESTINATION/$CLOUD_PROP

#---------------------------------------------------------------------------------------------
#no powder single pass
#---------------------------------------------------------------------------------------------

elif [ $PASSES_VALUE == 2 ]
then
```

```
    #change geometry def

    foamDictionary -entry POS0_X -set "0" -disableFunctionEntries $DESTINATION/$GEOMETRY


    foamDictionary -entry POS0_Y -set "-0.003" -disableFunctionEntries $DESTINATION/$GEOMETRY


    foamDictionary -entry POS0_Z -set "-0.04" -disableFunctionEntries $DESTINATION/$GEOMETRY

    foamDictionary -entry POS1_Z -set "0.04" -disableFunctionEntries $DESTINATION/$GEOMETRY


    #change laser position

            foamDictionary -entry straightCoeffs.lasPosStart -set "(\$POS0_X \$PLATE_OFFSET_NEG \$POS0_Z)"
$DESTINATION/$LASER_FOLD/step0MovementProperties -disableFunctionEntries

            foamDictionary -entry straightCoeffs.lasPosEnd -set "(\$POS0_X \$PLATE_OFFSET_NEG \$POS1_Z)"
$DESTINATION/$LASER_FOLD/step0MovementProperties -disableFunctionEntries


    foamDictionary -entry INJECTOR.duration -set "0" -disableFunctionEntries $DESTINATION/$CLOUD_PROP

            foamDictionary -entry subModels.injectionModels.modelA.duration -set "0" -disableFunctionEntries
$DESTINATION/$CLOUD_PROP

    endTime=$(bc <<< "scale=10;0.08/$VELOCITY+1000")

    foamDictionary -entry endTime -set $endTime -disableFunctionEntries $DESTINATION/$CONTROL

    fi


done
```

6.5 Matlab script for merging of internal probes data

```
clear all

clc

pkg load io;

tempPWD = mfilename('fullpath');

PWD = fileparts(tempPWD);

probeWD = strcat(PWD,'/postProcessing/internalProbes/');

cd(probeWD)



if exist('finalResults.xlsx')

  disp('ERROR: finalResults already exists!')

else

  subFolders=dir;

  toBeRead = strcat(PWD,'/postProcessing/internalProbes/',subFolders(3).name, '/points_T.csv')

  temp = xlsread (toBeRead);

  finalValues = zeros(length(subFolders)-2, length(temp));



  for i = 1:length(subFolders)-2 %read, modify and save necessary data

    clear temp

    subFolder = subFolders(i+2).name;

    finalValues(i,1) = str2double(subFolder);

    toBeRead = strcat(PWD,'/postProcessing/internalProbes/',subFolder, '/points_T.csv');

    temp = xlsread (toBeRead);

    temp(:,3) = [];

    temp(:,2) = [];

    temp(:,1) = [];

    temp(1,:) = [];

    temp = transpose(temp);

    finalValues(i,2:end) = temp;
```

```
    end

    finalValues = sort(finalValues);

    xlswrite('finalResults', finalValues);

end
```