



Efficient part orientation algorithm for additive manufacturing in industrial applications

Antonio Bacciaglia¹ · Alfredo Liverani¹ · Alessandro Ceruti¹

Received: 5 December 2023 / Accepted: 18 June 2024
© The Author(s) 2024

Abstract

Over the past few decades, the scientific community's and industry's interest in additive manufacturing technologies has surged. This technology is distinguished by the layer-by-layer deposition of the raw materials and the piece's growth in a predetermined build orientation. This factor impacts the process' overall cost, surface quality, and other crucial parameters. Numerous methods to solve competing aspects have been proposed in the literature, with the more promising that iteratively uses ray-tracing techniques. Existing algorithms iterate for each discrete element of the model's bounding box projection onto the building platform. However, when optimisation algorithms are used on real-life industrial parts, computational time problems arise due to the high number of faces in the models. A new computational technique to determine the appropriate part orientation to reduce the support volume is proposed to address the problem. The method reduces the computational time, cycling the ray-tracing only on the triangles where the model surface is discretised. This approach has been integrated into an enhanced particle swarm optimisation algorithm to prove its efficiency. The approach is intended for industrial applications where it is necessary to handle complicated geometries quickly and efficiently to find the best orientation. Based on the computer's resources and the complexity of the faceted model, a set of case studies with an industrial engineering significance is used to demonstrate the approach's effectiveness.

Keywords Additive manufacturing · Particle swarm optimisation · Build orientation · Part orientation · Support material

1 Introduction

Unlike the conventional manufacturing process based on chip removal, additive manufacturing (AM) is characterised by the layer-by-layer deposition of raw material [1]. Its employment has surged in all fields of industrial engineering, including automotive, aerospace, and biomedicine [2, 3]. AM is suitable for all applications requiring limited, customised production quantities and quick prototyping of shrunk-down models. AM technology offers some advantages, such as a short design-to-manufacturing cycle, design flexibility, the ability to build complex structures in one

piece, reduced raw material waste, and the potential to replicate light and bioinspired geometries.

One of the key technical characteristics of AM technologies is how the component is produced by depositing material layer by layer and growing on its own along the building orientation. This manufacturing process variable impacts the created object's cost, surface quality, and other crucial attributes. Even though AM is a promising technology that is changing the economics and production workflow, there are still many unresolved issues, including high surface roughness [4], small AM machine's building volume [5], high mechanical anisotropy [6], and non-trivial numerical assessment of mechanical performances [7], on which the scientific community is concentrating its efforts. High raw materials and equipment costs and a slow certification process also impact AM products. This is due to the significant structural performance variability driven by variations in raw material quality, adjustments (often minor) made to machine settings or environmental conditions, and how AM structures react to fatigue stresses.

✉ Alessandro Ceruti
alessandro.ceruti@unibo.it

Antonio Bacciaglia
antonio.bacciaglia2@unibo.it

Alfredo Liverani
alfredo.liverani@unibo.it

¹ Department of Industrial Engineering (DIN), University of Bologna, Bologna, Italy

The model layering is a stage in all AM techniques required to plan the operations for material deposition. This procedure is carried out by giving the object a build orientation that is orthogonal to the machine platform. The raw material is deposited during manufacturing, and the previous layer often supports the newly constructed layer against gravity. The actual layer may be overexposed in comparison to the preceding one for some complex geometries. The overhang area, defined as the portions of surfaces whose normal vector is perpendicular to the construction direction, could be candidates for support infrastructure for some AM techniques [8]. The primary results of support generation are an increase in the material amount, build time, and subsequent cost. Last but not least, supports reduce the quality of the sections of surfaces they come into touch with. As a result, choosing the appropriate build orientation in an additive manufacturing scenario is a complex challenge that has been extensively discussed in the literature, as shown by several review papers on the subject [8–10].

Specifically, it should be essential to replicate the complete manufacturing process for the given component orientation to estimate the support volume accurately. However, this method is associated with high and unacceptable computing demands in an optimisation environment. So, simplification techniques are required to estimate the amount of support material with high precision. The literature suggests many quick and easy methods to assess 1D [11] and 2D [12–14] characteristics connected to support material, such as the area of supports [15]; however, their accuracy is relatively poor, and the actual volume of support material is not available.

Several contributors ([16–18]) established fairly precise approaches to estimate the actual quantity of support material; however, they only considered the support linked directly to the building plate and ignored the internal overhangs. A voxel-based technique to estimate the total support material, considering both plate and internal supports, is suggested by [19] and [20] to resolve these challenges. However, voxelisation is a labour-intensive graphical representation method, making it unsuitable for use in an optimisation framework where it is necessary to analyse several orientation configurations to select the best one quickly. Das et al. [21] suggest using NURBS-based modelling rather than the tessellation technique to determine the appropriate quantity of support material. However, a professional operator is required to create a cloud of points that is coherent with the component's surface using a NURBS-based technique, as this is a very challenging procedure. Several methods in the literature address the support material prediction problem by using machine learning (ML) techniques embedded in an optimisation framework [22–24]. However, all ML techniques must be trained on massive datasets to obtain accurate results that may not be accessible in industrial applications, especially for start-up businesses.

To address the abovementioned problems, [25] and [26] proposed two precise approaches to calculate the support material required to manufacture a product given a specific component orientation based on existing techniques, i.e. ray-tracing [15]. The methodology, which is based on a 2D tessellation technique of the component's footprint that is quicker than voxelisation (a 3D tessellation), can capture both internal and plate supports. Thus, this method may be used to implement optimisation methods to determine the ideal component orientation in a typical scenario. However, there could be some problems with the computing needs, particularly in industrial applications where the STL file might include hundreds of thousands of facets. This approach has a computational cost proportional to the product of the number of facets of the STL file and the number of tessellation elements of the printing bed. Indeed, the traditional algorithm is constituted by an iterative cycle that analyses each grid element of the 3D model footprint: for each grid element, the algorithm checks for triangle facets in an overhang status. In this scenario, the computational power is proportional to the number of grid elements times the number of triangular facets of the model. On the one hand, the model complexity in the case of real-life components is large. On the other, the number of grid elements should be large enough to achieve accurate results. The consequent high computational time can be acceptable in the case of a single run. However, when optimisation algorithms are applied to find optimal solutions, thousands of processes should be run, with unacceptable computational times. Thus, a more effective and consistent strategy is therefore required.

This work proposes an original approach for support volume estimation with a computational demand proportional to the number of triangle facets. Moreover, the algorithm can recognise the facets that are in overhang with respect to the building platform so that only the grid elements that contain the footprint of triangular facets in overhangs are investigated. The innovative support volume estimation routine is included in an optimisation framework to test it in a scenario where the algorithm is called iteratively. The optimisation can determine the ideal component orientation by exploring every conceivable configuration with a modified version of a particle swarm optimisation (PSO) method [27]. Additionally, the framework can provide the expected time required for the complete simulation, knowing the computer performances and the number of facets of the component, thanks to a proven dataset. The contribution of this paper to the current literature is to develop a fast algorithm for supporting volume estimation with accuracy similar to existing approaches. Developing an efficient algorithm with a reduced computational cost can be helpful in all the applications where support volume routines are called repetitively or with complex parts. Indeed, the approaches in the literature suit the typical benchmark geometries used

in papers but show their limits when dealing with complex industrial components. The computational demand is crucial in all the cases where the support volume estimation function is called iteratively, such as in optimisation frameworks based on evolutionary algorithms.

The manuscript is organised as follows. The novel approach to estimating the support volume is described in Sect. 2, along with how it fits into the overall optimisation strategy. The implementation of the optimisation framework in FreeCAD is covered in depth in Sect. 3. Section 4 presents three case studies for evaluating the strategy’s effectiveness and discusses the findings. Finally, Sect. 5 summarises the study and offers recommendations and potential directions for the established research area.

2 Methodology

In the following subsections, the mathematical method behind the functions implemented in the proposed framework will be described in detail. Particular focus will be given to the support volume estimation (described in Sect. 2.4), but other functions, such as the area and volume estimation of the STL file (described in Sect. 2.1), the rotation operation (described in Sect. 2.2), and the analysis of

$$\text{Volume}_i = \frac{1}{6}(V_{x1}(V_{y2} \cdot V_{z3} - V_{y3} \cdot V_{z2}) + V_{y1}(V_{z2} \cdot V_{x3} - V_{z3} \cdot V_{x2}) + V_{z1}(V_{x2} \cdot V_{y3} - V_{x3} \cdot V_{y2}))$$

The whole volume of the body is contained by the surfaces described in the STL file and can be found by summing all the contributions of the individual tetrahedrons.

2.2 Body rotation

For clarity, a brief paragraph about the 3D rotation of a surface mesh file is included in the manuscript: rotation angles will be selected as input variables in the algorithm to search for optimal orientation.

The Euler angles — defined by three angles, namely pitch (around the Y-axis), roll (around the X-axis), and yaw (around

$$\text{rotMATRIX} = \begin{bmatrix} c(\text{yaw}) \cdot c(\text{pitch}) & c(\text{yaw}) \cdot s(\text{pitch}) \cdot s(\text{roll}) - s(\text{yaw}) \cdot c(\text{roll}) & c(\text{yaw}) \cdot s(\text{pitch}) \cdot c(\text{roll}) + s(\text{yaw}) \cdot s(\text{roll}) \\ s(\text{yaw}) \cdot c(\text{pitch}) & s(\text{yaw}) \cdot s(\text{pitch}) \cdot s(\text{roll}) + c(\text{yaw}) \cdot c(\text{roll}) & s(\text{yaw}) \cdot s(\text{pitch}) \cdot c(\text{roll}) - c(\text{yaw}) \cdot s(\text{roll}) \\ -s(\text{pitch}) & c(\text{pitch}) \cdot s(\text{roll}) & c(\text{pitch}) \cdot c(\text{roll}) \end{bmatrix} \quad (5)$$

2.3 Growth ratio computation

Computing the growth ratio means dividing the body into a set of slicing planes (N_{slices}) and checking the area obtained intersecting the body with the single slicing plane. The slicing

the growth ratio (described in Sect. 2.3), are included for a complete overview of the proposed STL analysis and optimisation framework.

2.1 Body area and volume estimation

The volume of the body to print is computed according to the methodology used in [28]. The STL format is based upon discretising a single or a set of surfaces using triangles. For each i th triangle, the STL format lists the position of the vertices $[P_{1X}, P_{1Y}, P_{1Z}]_i, [P_{2X}, P_{2Y}, P_{2Z}]_i, [P_{3X}, P_{3Y}, P_{3Z}]_i$, and the normals $[N_X, N_Y, N_Z]_i$. The external area (Surf_T) can be evaluated as half of the cross product in \mathbb{R}^3 of the two vectors connecting vertices 2 and 1 ($V_2 - V_1$) _{i} , as well as 3 and 1 ($V_3 - V_1$) _{i} :

$$\text{Surf}_T = \sum_{i=1}^{N_B} \frac{1}{2} \sqrt{(T_i)^2 + (M_i)^2 + (R_i)^2} \quad (1)$$

where:

$$\begin{aligned} T_i &= (P_{y2} - P_{y1})_i \cdot (P_{z3} - P_{z1})_i - (P_{z2} - P_{z1})_i \cdot (P_{y3} - P_{y1})_i \\ M_i &= (P_{z2} - P_{z1})_i \cdot (P_{x3} - P_{x1})_i - (P_{x2} - P_{x1})_i \cdot (P_{z3} - P_{z1})_i \\ R_i &= (P_{x2} - P_{x1})_i \cdot (P_{y3} - P_{y1})_i - (P_{y2} - P_{y1})_i \cdot (P_{x3} - P_{x1})_i \end{aligned} \quad (2)$$

The volume of a single tetrahedron is as follows:

$$(3)$$

the Z-axis) — can be used to compute a rotation of an STL file [29]. The rotated coordinates of the vertices $[P_{jx}, P_{jy}, P_{jz}]_i$, indicated with the R superscript ($[P_{jx}, P_{jy}, P_{jz}]_i^R$) can be found using Eq. 4 ($j=1:3$, i th triangle):

$$\begin{bmatrix} P_{jx} \\ P_{jy} \\ P_{jz} \end{bmatrix}_i^R = \text{rotMATRIX} \times \begin{bmatrix} P_{jx} \\ P_{jy} \\ P_{jz} \end{bmatrix}_i \quad (4)$$

Using the abbreviated form $s(x) = \sin x$ and $c(x) = \cos x$, the rotation matrix rotMATRIX is defined in Eq. 5:

is carried out along the directions where the layers are deposited once a slicing step Δz is set by the user.

Every single triangle is checked, and if an intersection is found with the current plane, the two intersection points $[BX_1, BY_1, BZ_1]_i$ and $[BX_2, BY_2, BZ_2]_i$ of i th triangle are computed (see Fig. 1

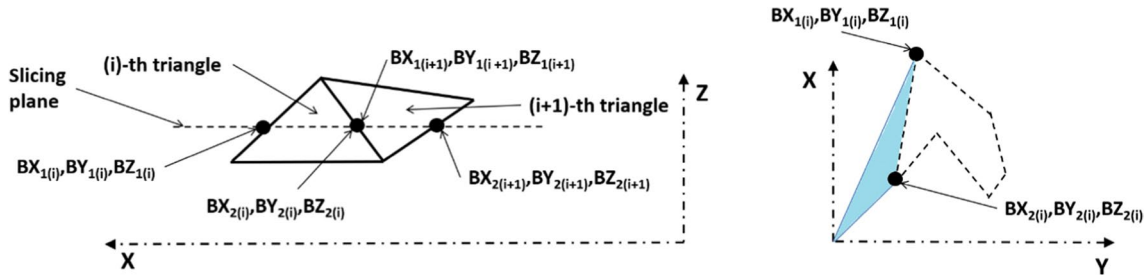


Fig. 1 Computation of the cross area along slicing planes

left). All those points are ordered up to obtaining a closed loop, as defined in [30], or more than a closed loop if two or more are detected. Summing up the area of the triangles connecting a common fixed origin and two consecutive points of the loop allows the computation of the layers' area (see Fig. 1 right).

2.4 Overhang area and support volume

The method to evaluate the overhang area is described below. Each triangle in the STL is cycled, and for each one, the angle (θ) between the i th triangle normal vector $[N_x, N_y, N_z]_i$ and the floor's normal vector $N_{\text{floor}} = \sqrt{(N_x)^2 + (N_y)^2}$ is computed:

$$\theta = \arctan\left(\frac{N_z}{N_{\text{floor}}}\right) \tag{6}$$

Given a threshold angle (θ_{tr}), usually set equal to 45–55°, a triangle is considered to belong within the threshold area if: $\theta < (\theta_{tr} - 90^\circ)$. In this latter case, the area of the triangle is summed to the overhang area of the body. A check is carried out to verify if the triangles are lying on the building plate, meaning that support material is not required: This condition is assumed if N_z is close to 1 and the distance between the building plane and the triangle is close to 0.

About the support volume, the traditional methodology, employed mainly in literature and described in papers such as [25] or [26], is based upon the ray-intersecting Möller-Trumbone mathematical approach [31]. However, as Fig. 2 depicts, this methodology requires cycling for each grid element in which the body's footprint bounding box is divided by the number of all the mesh triangles to find the intersection points (black circles in Fig. 2). However, much computational power (referring to red grid elements) is inefficiently spent because no triangular facet's footprint is contained on that specific element of the grid.

In this way, the total number of computations to be carried out is proportional to N_{sq} times N_{tr} , as seen from the methodology flowchart proposed in Fig. 3. In the case of a real-life body, this number could be enormous because N_{tr} can be up to thousands of triangles. The traditional methodology requires significant computational time, which is inadequate for iterative optimisations, where several evaluations must be done.

A new original methodology has been developed to cope with this problem. The methodology flowchart is changed, as shown in Fig. 4.

As can be seen, this algorithm is proportional only to N_{tr} number of triangles. For each triangle, only the grid elements belonging to the facet footprint will be investigated through the ray-intersecting method. This way, the computational demand can be drastically decreased. Using the same example seen in Fig. 2, the new methodology will investigate each triangle (in cyan) and find only the grid's element on which an overhang region footprint overlaps on that particular grid element (in dark green), as seen in Fig. 5.

The proposed methodology uses the gift-wrapping algorithm to find which grid elements belong to the facet's footprint under investigation [32]: once the triangle of vertices $(V_x, V_y, V_z)_A$, $(V_x, V_y, V_z)_B$, and $(V_x, V_y, V_z)_C$ is selected, only the possible

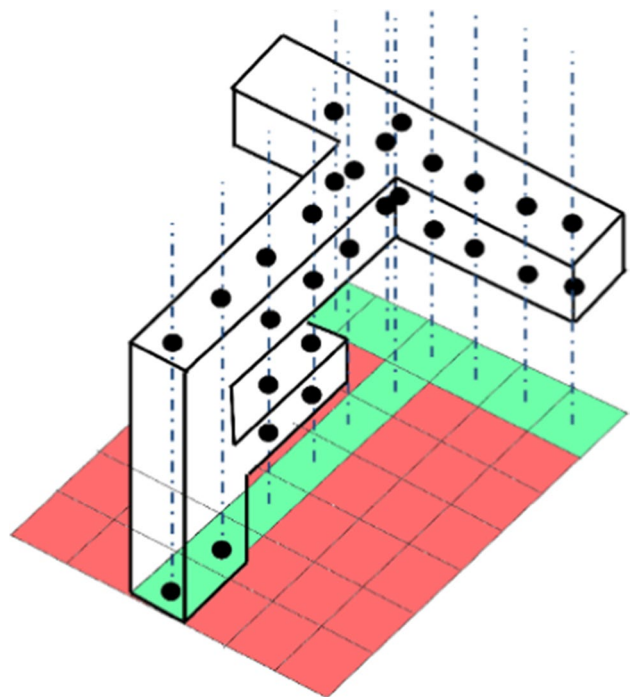


Fig. 2 Support volume estimation in literature is based on calculating perpendicular rays from each coloured square of the grid (red and green); however, just the green elements will support the component

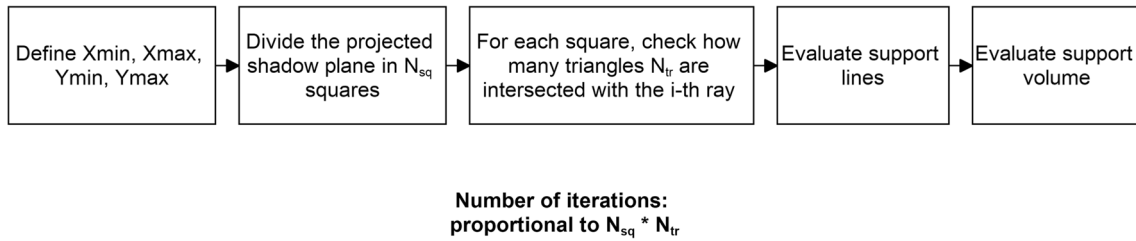


Fig. 3 Traditional computation methodology for support computation

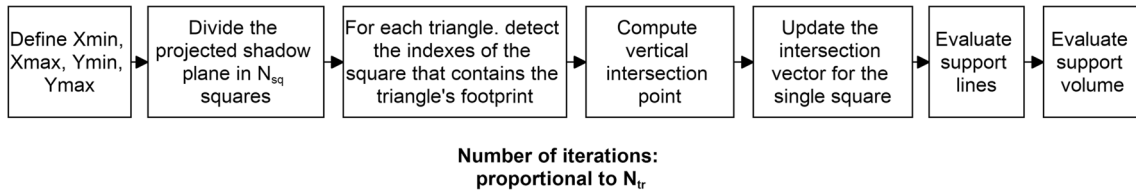


Fig. 4 New computation methodology for identifying triangles lying over checkerboard (proportional to N_{tr}) divisions and further support volume evaluation

grid elements under it are considered. For each grid element, the midpoint coordinates $(V_X, V_Y, V_Z)_n$ are used to examine the number of points present in the convex hull formed by connecting the triangle's vertices with the point under consideration (Eq. 7).

The grid midpoint lies in the interior of the triangle if $a, b > 0$ and $a + b < 1$, where a and b are defined according to Eq. 8.

$$\begin{cases} D_1 = (V_{X_n} - V_{X_A})(V_{Y_C} - V_{Y_A}) - (V_{Y_n} - V_{Y_A})(V_{X_C} - V_{X_A}) \\ D_2 = (V_{X_A} - V_{X_n})(V_{Y_B} - V_{Y_A}) - (V_{Y_A} - V_{Y_n})(V_{X_B} - V_{X_A}) \\ D_D = (V_{X_B} - V_{X_A})(V_{Y_C} - V_{Y_A}) - (V_{Y_B} - V_{Y_A})(V_{X_C} - V_{X_A}) \end{cases} \quad (7)$$

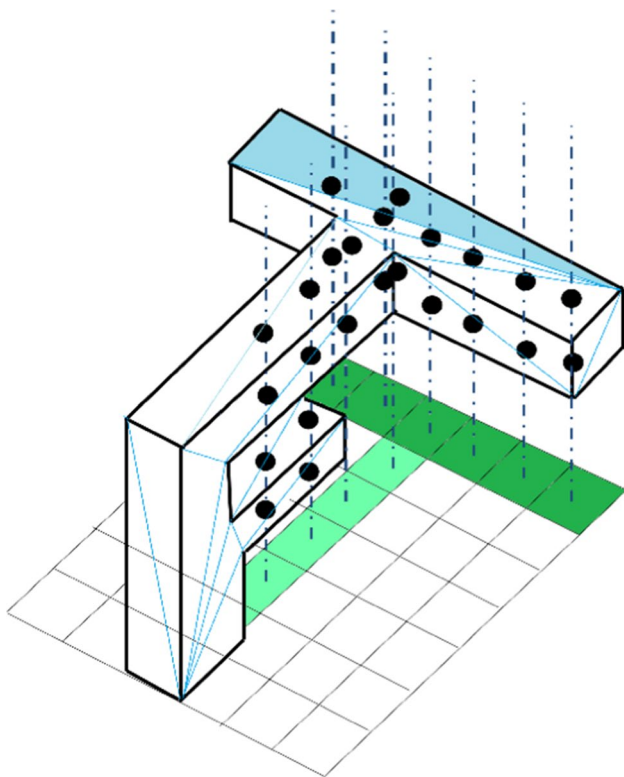


Fig. 5 Innovative support volume estimation is based on calculating perpendicular rays just from grid elements that belong to the component's footprint of a specific triangle (in cyan); this way, only the green elements that will support the part are investigated

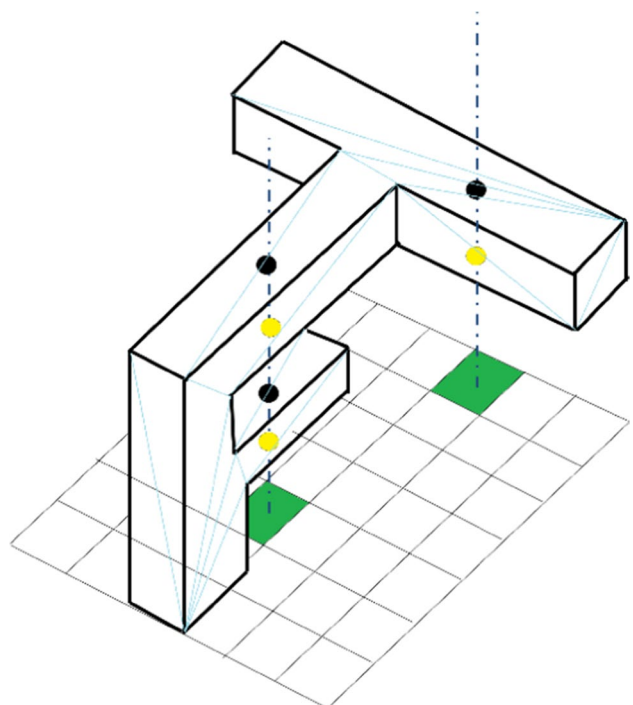
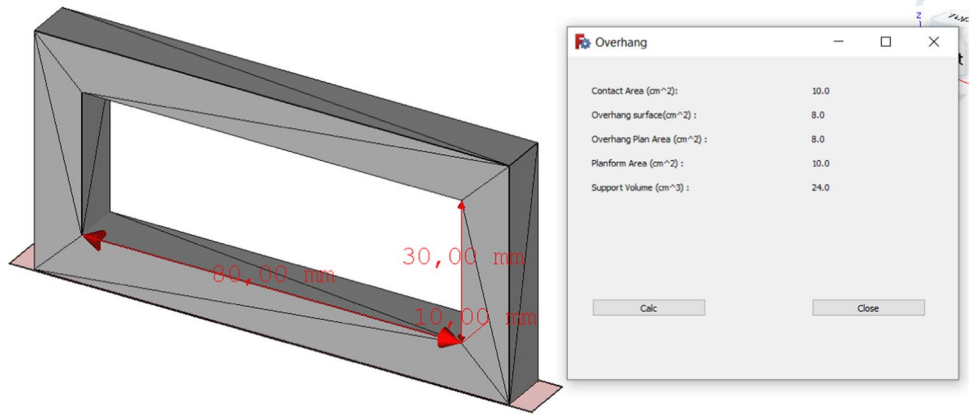


Fig. 6 For each ray originating from the printing bed, only the relative heights of the odd intersections (in yellow) contribute to the support volume estimation

Fig. 7 Example of support volume estimation: The amount is precisely the volume gap of the square hole



$$\begin{cases} a = \frac{D_1}{D_D} \\ b = \frac{D_2}{D_D} \end{cases} \quad (8)$$

If the mathematical conditions are satisfied, the element grid under the i th triangle is activated (a true value is saved in the grid matrix footprint) because it belongs to the component's footprint. Moreover, if the triangle over the activated element is in overhang, an additional flag is initiated to save this property.

The vertical intersection point for the specific grid element is evaluated by a weighted average of the z -coordinates of the three vertices of the triangle; the weights are computed as the reciprocal of the square of the distance between the node and the triangle's vertex, as seen in Eq. 9:

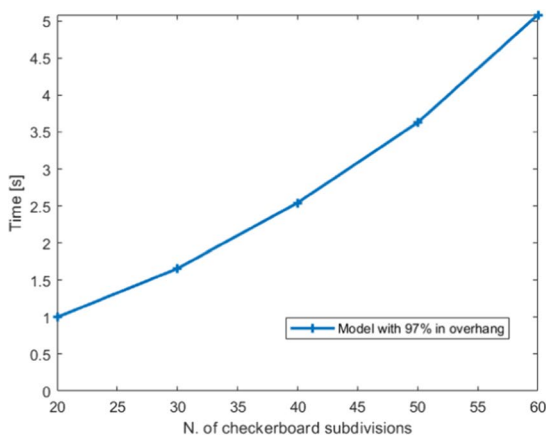
$$\begin{cases} K_A = \frac{1}{(V_{x_n} - V_{x_A})^2 + (V_{y_n} - V_{y_A})^2} \\ K_B = \frac{1}{(V_{x_n} - V_{x_B})^2 + (V_{y_n} - V_{y_B})^2} \\ K_C = \frac{1}{(V_{x_n} - V_{x_C})^2 + (V_{y_n} - V_{y_C})^2} \end{cases} \quad (9)$$

Then, the average z height of the intersection point for the specific grid element can be approximated as follows:

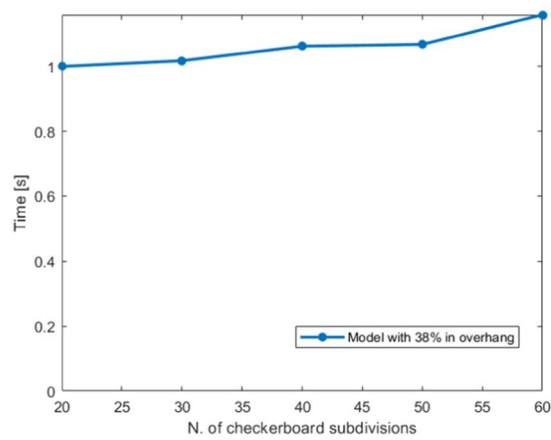
$$z = \frac{K_A \cdot V_{z_A} + K_B \cdot V_{z_B} + K_C \cdot V_{z_C}}{K_A + K_B + K_C} \quad (10)$$

One or more intersection points may be found for each active (green referring to Fig. 5) square element of the grid belonging to the component's footprint. These heights are sorted in ascending order and categorised if belonging to an overhang (yellow dots in Fig. 6). Starting from $z = 0$ (printing bed level) and rising to the highest intersection, and for each odd intersecting point, the support volume contribution can be approximated as the relative height of the intersecting point with respect the previous one, times the area of the element grid.

An incremental sum of all contributions can evaluate the overall support volume. The obtained value is considered valid only if a 100% infill density is used when setting the manufacturing parameters. To support the description of the innovative methodology to estimate the support volume, the pseudocode follows:



a)



b)

Fig. 8 The computational time versus the number of linear checkerboard subdivisions for different % of overhang triangles: **a)** a model with overhang triangles occupying 97% and **b)** a model with 38% in overhang

<i>Support volume function</i>	
Make a $n \times n$ grid with the dimensions of $X \times Y$ bounding box of the STL file	1
$Area_grid$ = area of the single grid element	2
For each i triangle of the STL	3
Evaluate $X \times Y$ bounding area of the i -th triangle	4
Verify how many grid elements are under the i -th triangle	5
If node (j,k) of grid is inside the i -th triangle	6
If i -th triangle is in overhang	7
$Flag_overhang_node(j,k)=1$	8
End	9
$Num_tria_inist(j,k)$ =Incremental sum of number of triangles whose footprint	10
is contained within the (j,k) elem. of the grid	
Save index i of i -th triangle in $index_tria_inis(j,k,l)$ where l is the number of	11
triangles whose footprint is contained within on (j,k) element found on the	
previous line	
Evaluate the height of the i -th triangle from the printing bed and save it in	12
$height_z_tria(j,k,l)$	
If i -th triangle is in overhang	13
$Height_z_tria_overh(j,k,l)=1$	14
Else	15
$Height_z_tria_overh(j,k,l)=0$	16
End	17
End	18
End	19
For each elem. of the grid (j,k) that contains at least an overhang triangle footprint	20
Save the height of intersection of a perpendicular ray starting from (j,k) elem.	21
contained in $height_z_tria$ and sort them in increasing order	
Among the intersections, isolate those referring to overhang regions	22
For $m=1$:half of (j,k) intersections	23
If $m=1$ % support directly attached to printing bed	24
save the height of the intersection in $height(m)$	25
Else % support also inside the component	26
Evaluate the relative height between the odd and even intersection	27
and save the result in $height(m)$	
End	28
SupportVolume= incremental sum of $height(m)*area_grid$	29
End	30
End	31

Fig. 9 **a** Simple geometry is used as a benchmark to compare the proposed algorithm for support volume estimation with the literature; **b** the support volume visible in blue can be analytically evaluated and compared with numerical results

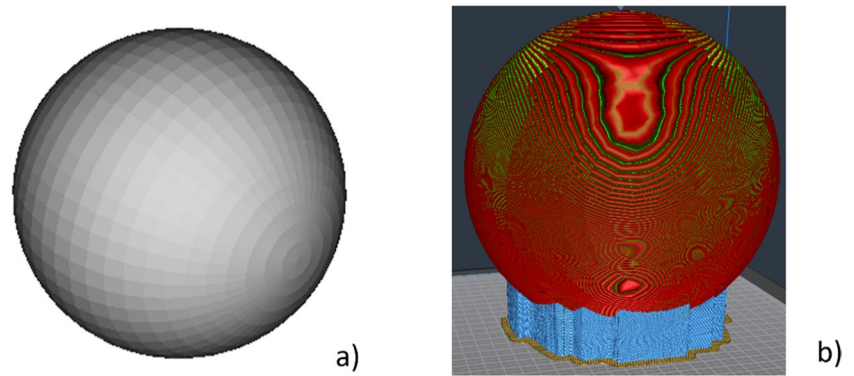


Figure 7 shows an example of the support volume estimation required to build the 3D model. This approach can detect both internal and on-plate supports differently from several methodologies available in the literature.

The support volume estimation described in this section depends on the orientation of the 3D model, where the orientation can be described thanks to the pitch and roll angles. The approach we propose should be included in an optimisation framework described in the following paragraphs to minimise the support volume and find the best orientation.

In this research, the component's footprint is divided into 30×30 grid elements; this value will be used for all the examples and case studies in this manuscript. This parameter affects the accuracy of the support volume estimation. On the one hand, the finer the footprint's checkerboard, the more precise the estimation of the support volume; on the other hand, a coarser grid will reflect a worse estimation. The computation of the support volume requires knowledge of the triangles lying on a grid element of the checkerboard and understanding which triangles are in overhang and require supports. This additional workload, typical of the traditional methodologies and the new one herein proposed, affects the whole computational time of support volume. However, for parts where overhang triangles occupy a high portion of the checkerboard division of the footprint, the more demanding computational load is required by the identification of

the triangles lying on the single grid element. The image in Fig. 8 shows the impact of the checkerboard division size on the computational time when overhang triangles occupy 97% (Fig. 8a) and 38% (Fig. 8b) of the footprint.

In the case of 38% of triangles in overhang, finding the triangles lying on the single element of the checkerboard requires more time than the other operations. In this latter case, it is evident that our approach to compute the triangles lying on a single element is proportional to N_{tr} without being influenced by the size of the grid: The computational time for a 20×20 grid has been taken as a reference in Fig. 8, and it is evident that the increase of computational time for a 60×60 grid is low.

Figure 9 shows a simple spherical geometry with a 100-mm radius, with 2352 facets (whose STL file has a dimension of 115 KB) used as a benchmark to compare the proposed algorithm for support volume estimation with the approach proposed by [25], used as a reference. The support volume can be analytically estimated due to the simple topology; with an overhang angle threshold of 45° , the support material is visible in Fig. 9b. The approach described in [25] has been implemented and tested on the same geometry, varying the underlying grid resolution. A comparison of the computational time and the error in the support volume estimation is collected in Table 1.

Table 1 The computational time comparison between the proposed approach and the method presented in [25]

N. of checkerboard subdivisions	Proposed algorithm running time [s]	Support volume estimation error with analytical value [%]	Running time for the algorithm presented in [25] [s]	Support volume estimation error with analytical value [%]	Speedup time ratio ([25]/proposed)
20	1.29	25.94	1.91	24.17	1.48
30	2.93	12.88	3.83	12.99	1.31
40	6.33	14.57	7.35	14.32	1.16
50	8.63	11.05	11.03	10.91	1.28
60	14.63	10.84	20.66	11.08	1.41

2.5 Particle swarm optimisation

A variety of strategies, including gradient-based [33], Monte Carlo [34], meta-heuristic approaches [35], or even machine learning [36], can be used to set up an optimisation process. A meta-heuristic algorithm was selected as the foundation for the optimisation method in this study from the broad spectrum of optimisation philosophies available in the literature because it can be seen as a trade-off between a purely random approach and a more mathematically sophisticated methodology. Generally, the process used to evaluate the fitness function is usually called many times in an optimisation framework. The authors' main contribution is the support volume estimation algorithm. The optimisation framework described in this manuscript is the enclosure used to test the support volume estimation algorithm. A robust and well-consolidated optimisation algorithm has been implemented to cooperate with the support volume algorithm to find the best STL orientation. However, other novel optimisation approaches can be used to implement the proposed approach.

Among the meta-heuristic strategies, which mimic how the physical world or nature functions, the particle swarm optimisation (PSO) method ensures good results using plain and straightforward code [37]. It has been used in various industrial investigations [38]. It operates by considering the location of a potential solution in the design space, where some parameters define a single solution. The location that a single particle $X_i(t)$ has during the design space exploration will depend on both the best position of each individual member of the population (personal best B_i) and the best position that the entire set will find (global best G).

Equation 11 is used to determine the locations $X_i(t)$ and velocities $V_i(t)$ of each particle at the following iteration using the iteratively updated values of B_i and G , where i is the single particle's index and t denotes the algorithmic step.

$$\begin{cases} V_i(t+1) = w \cdot V_i(t) + c_1 \cdot \text{rand}(0,1) \cdot (B_i - X_i(t)) + c_2 \cdot \text{rand}(0,1) \cdot (G - X_i(t)) \\ X_i(t+1) = X_i(t) + V_i(t+1) \end{cases} \quad (11)$$

The inertia weight w determines how the actual velocity will affect the following generation. The impact of the prior velocity on the current velocity increases with increasing inertia weight value and vice versa. The result of the random function $\text{rand}(0,1)$ is a number between 0 and 1. It is necessary to predetermine the learning parameters c_1 and c_2 , because the PSO algorithm performs significantly differently depending on their values, according to convergence criteria available in the literature [39].

All particles' velocities are initially zero, and their positions are picked randomly from acceptable values when the algorithm is initialised.

In the classical optimisation algorithm, there are two approaches to accomplish the stopping condition for the simulation. In the first, the procedure terminates after a preset maximum number of repetitions n_{max} is reached. A second exit strategy can be triggered when no advancement in the solution is made for a predetermined number of consecutive iterations.

However, the traditional PSO can reduce its efficacy once the global best particle is trapped in a local minimum: all particles will eventually be attracted there. Thus, it would be hard for the other particles to escape the local

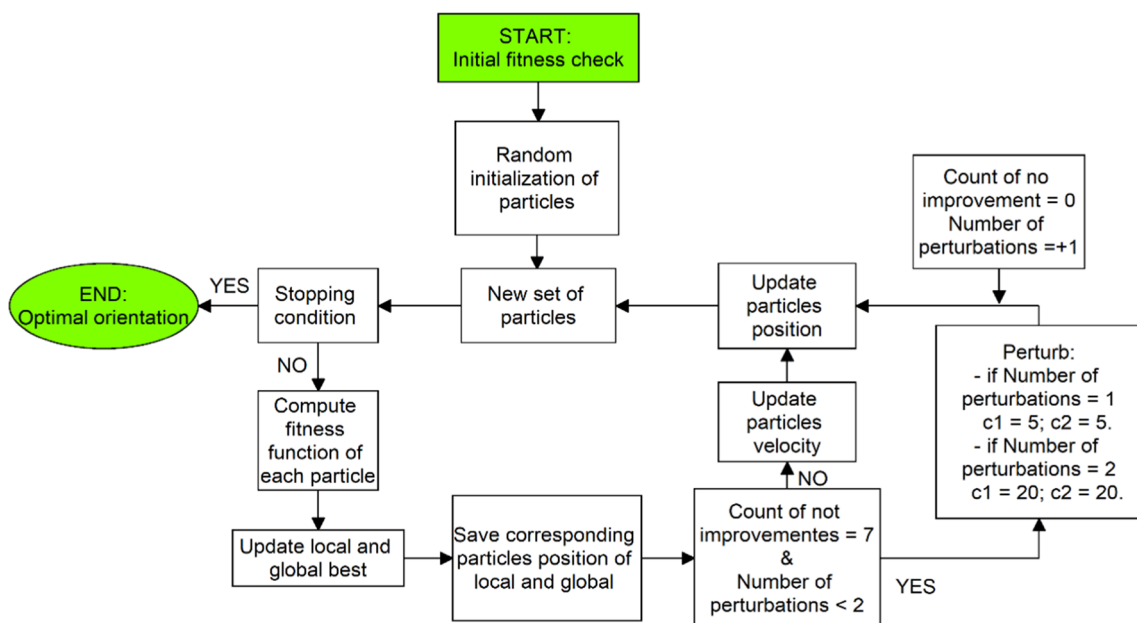


Fig. 10 Optimisation strategy based on the improved PSO of [27] to find the best part orientation

minimum since they follow the global best particle. The improved particle swarm optimisation algorithm, described in [27], has been applied in this research to avoid this undesired condition. The following paragraph includes a detailed description of the optimisation methodology followed in this study.

2.6 Improved PSO for build orientation in AM

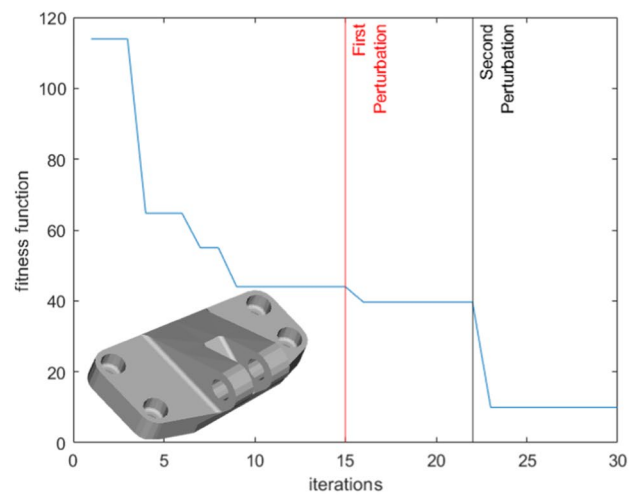
The overall support volume, calculated using the method given in Sect. 2.4 as a function of the rotations around the two principal axes (X and Y) of the printing bed, roll and pitch, is the fitness function $F(X)$ that the PSO should minimise in this specific application (Eq. 12). Figure 10 displays a thorough flowchart to help discuss the overall optimisation technique.

$$\begin{cases} \text{find } X | \min F(\{\text{roll}; \text{pitch}\}) \\ -180 < \text{roll} < 180 \\ -180 < \text{pitch} < 180 \\ F(X) \geq 0 \end{cases} \quad (12)$$

First, a check is made in the fitness value when the digital model is imported with the default orientation to see if the support volume quantity has already reached its absolute minimum, which is zero. After the check, a new set of particles is created by initialising the particles with zero velocity and random positions in the design space defined in Eq. 12. At this point, the iterative optimisation process begins, and the steps listed below are repeated until the stopping criteria are met. The stopping criteria are verified either when the iterations have reached the maximum number permitted by the user, chosen a priori, or when the optimisation process has been stuck in a minimum for longer than seven cycles despite the application of two numerical perturbations on the particle’s positions (the particle’s perturbation is defined in detail in [27]). If Eq. 13 is verified, meaning that the global fitness value has not significantly improved after two successive iterations, the optimisation process is said to be stuck in a local minimum at the $t + 1$ iteration and a counter value is updated until reaching a value equal to 7 for the simulations carried out. A threshold can be set to implement the termination criterion:

$$|G(t + 1) - G(t)| < \text{threshold} \quad (13)$$

A threshold value equal to 0.01 has been set in the tests carried out. If the terminating criteria are not met, the algorithm assesses each particle’s fitness function, updating B_i and G to check if any improvements are noted and save the relevant rotations in the memory. Then, the algorithm verifies any improvement in the fitness function thanks to the new iteration by Eq. 12. If the support volume decreases by a relevant amount comparing two consecutive iterations, the algorithm updates the velocities and positions of all the



a)

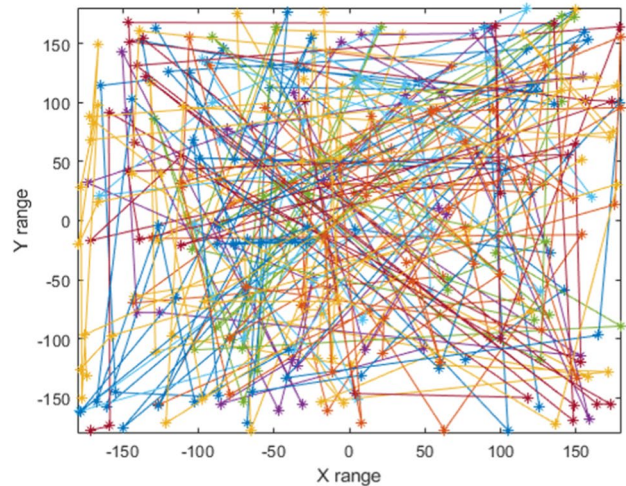
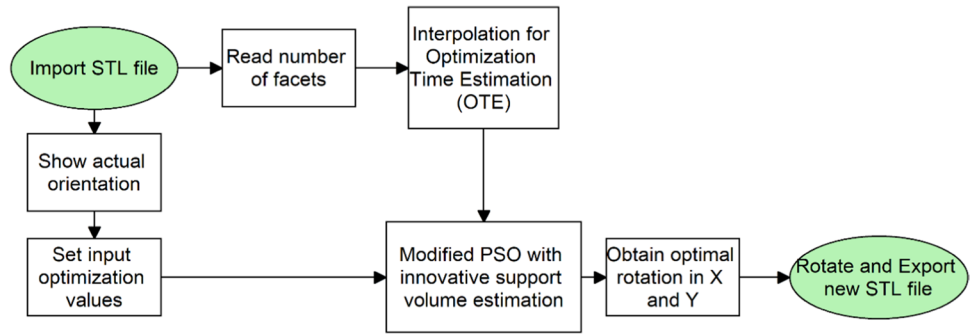


Fig. 11 Improved PSO to escape from local minima applied to the research of optimal part orientation: **a** fitness function behaviour during the iterations with two perturbations highlighted by vertical lines; **b** 10 particles moving inside the design space and converging in $X = 140$ and $Y = 129$

particles, knowing the renewed values of the local and global minima. Otherwise, if the counter of not improving iterations reaches a value of 7, a perturbation on the particle’s position is applied. According to [27], a perturbation applied to the particles consists of altering learning factors c_1 and c_2 if the fitness value does not improve after 7 consecutive training cycles. In the occasion that particles have already reached the global solution, the algorithm, however, also saved the current best personal and global solutions. This step is added twice to the conventional PSO strategy with the assumption that it will be sufficient to reach the global minimum and escape from local minima. From the experimental results available in [27], the values of the applied learning parameters directly affect how the perturbations behave. In

Fig. 12 AMVER flowchart explaining the optimisation framework implemented in FreeCAD



this specific research, for the first perturbation $c_1 = c_2 = 5$ is adopted, while for the second one, $c_1 = c_2 = 20$ is assigned to increase the effect of the perturbations, following the suggestions contained in [27].

Figure 11a perfectly shows how the modified PSO behaves on the fitness function value during the optimisation process, while particles move inside the design space (Fig. 11b). Thanks to the perturbations highlighted with vertical lines in Fig. 11a, the fitness function of the improved

PSO reaches an optimised value which is 76% lower than a traditional PSO that stops after the first plateau at around $F(X) = 40$.

The previous case study demonstrates the benefits of the improved version of the PSO, which is implemented in the AMVER environment where authors have implemented a GUI to optimise the part orientation to minimise the support volume required to manufacture a specific 3D model with AM technologies.

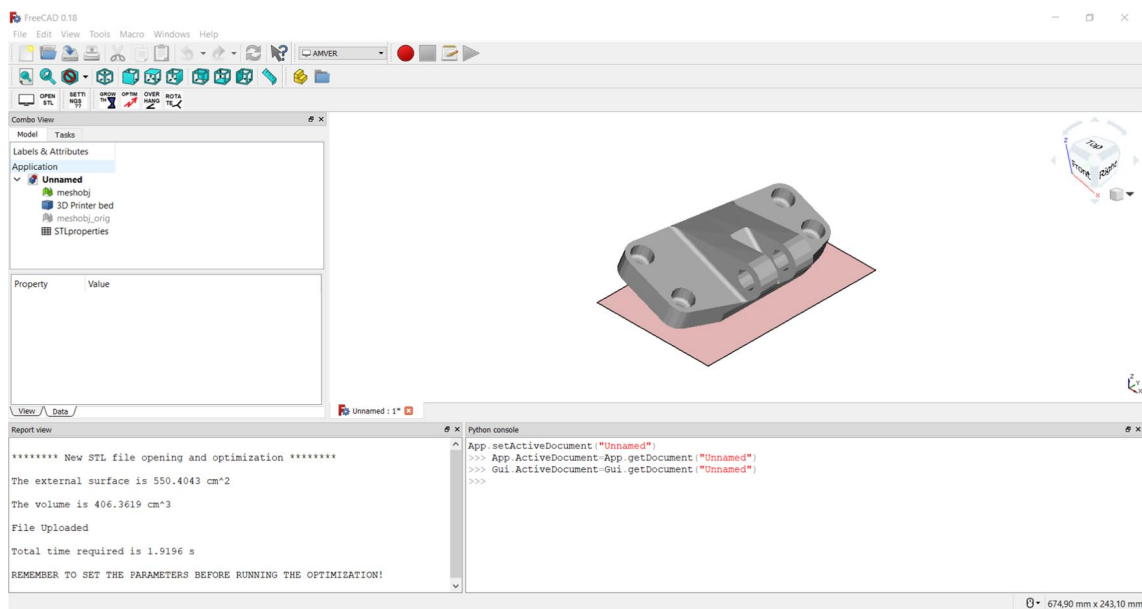
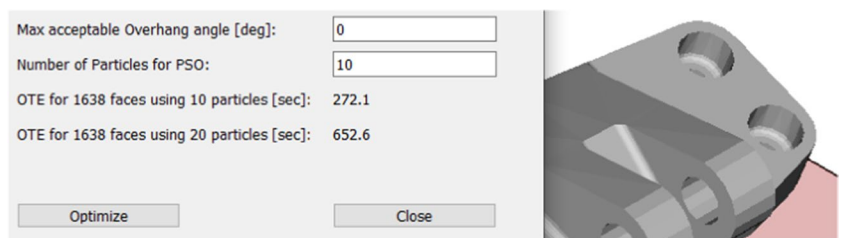


Fig. 13 AMVER graphic interface implemented in FreeCAD

Fig. 14 Input parameters required for the optimisation through a graphic window in AMVER



3 FreeCAD implementation

The developed framework, AMVER, to improve the build orientation of 3D STL models is described in this section. It has been created to be user-friendly for both seasoned industrial operators and newcomers to the field of AM. Figure 12 depicts the flowchart of the philosophy behind AMVER.

FreeCAD [40], an open-source computer-aided design (CAD) package, has been used to structure the Python-coded AMVER framework to optimise the building direction of 3D models.

Figure 13 illustrates the AMVER Interface. The 3D digital model is imported inside the AMVER framework with the original and non-optimal orientation in the 3D space after acquiring an STL file.

Before starting the optimisation process, the framework can automatically estimate the time needed to optimise the model's orientation using the modified PSO approach. This is done by saving important information about the STL file, such as the number of vertices and facets. In Sect. 3.1, the evaluation of the optimisation time estimation (OTE) will be covered in depth.

As shown in Fig. 14, the user is prompted to enter the maximum permitted overhang angle in degrees and the number of particles he wishes to use in the modified PSO before the optimisation process can begin. Moreover, two additional lines inform the designer of the OTE when using 10 and 20 particles (good preset values) for the optimisation process.

The improved PSO approach mentioned in Sect. 2 is executed once the user clicks the "Optimise" button. The framework then returns the minimal fitness value and the ideal orientation to apply to reduce the support volume after the optimisation process is complete. Downstream the optimisation, the user can rotate the object with the resulting angles, which reorients the digital model with respect to the building platform. The user can export the modified 3D model with the ideal orientation by choosing the STL Mesh from the feature design tree as the final step.

Furthermore, the AMVER framework can quickly assess several significant manufacturing and geometric aspects of the actual 3D model. For instance, the software can calculate the following characteristics while knowing the actual orientation:

- Contact area: the amount of surface in contact between the 3D model and the printing bed
- Overhang surface: total external surface in overhang with respect to the building platform
- Overhang plan area: projected area of the overhang regions on the building platform
- Platform area: total projected area of the object on the building platform

- Support volume: the total support volume required to sustain the overhang regions with the actual orientation
- Mean layer surface: the average cross-sectional area of the component along the growth direction during the production process
- Layer standard deviation: the standard deviation of the cross-sectional areas of the component along the growth direction during the production process

The computation of the growth ratio has been included in Sect. 2.3, and other factors, such as surface finish, residual stress, build time, and cost, could be integrated based on the several approaches proposed in the literature [25]. The PSO fitness function could consider simultaneously all these factors, and weights could be added to differentiate the importance of the single requirements. However, in this study, we want to show our approach's efficiency in computing the support volume, which is a computationally intensive task for large parts. Therefore, in the case study included in this paper, the fitness function will only consider reducing the support volume.

3.1 Optimisation time estimation

This section provides a thorough explanation of how the OTE was appropriately evaluated. Before starting the optimisation, an OTE is correctly displayed on the screen in an appropriate window to properly advise the designer on how long it will take to finish the optimisation.

The complexity of the 3D model, which can be correlated with the number of facets in the STL file, and the processing capacity available on the computer where the AMVER framework is installed are directly related to the OTE. A numerical metric, called CPU performance index, based on Eq. 14, is established to consider CPU performance.

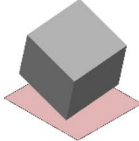
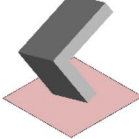
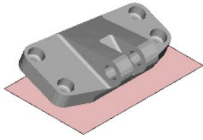
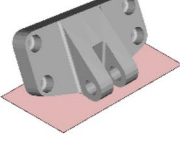
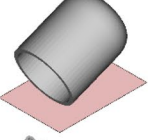
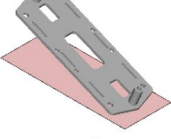
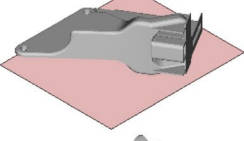
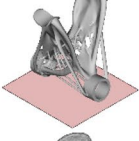
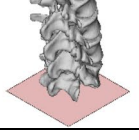
$$CPU_{\text{index}} = \frac{1}{t_{\text{test}}} \quad (14)$$

The evaluation of the timing value t_{test} , which the computer needs to iteratively calculate the factorial number of the first 5000 integer numbers, is the basis for this index. The more powerful of two machines will have a higher CPU_{index} value. The mathematical assessment takes 3.26 s on a PC with 16 GB of RAM and an i7-6700HQ CPU running at 2.6 GHz; hence, the CPU performance index is $CPU_{\text{index}} = 0.3027$.

By pressing the relevant button in AMVER, the previously mentioned mathematical test is automatically done once at the initial run of the programmed environment for each computer on which the AMVER framework is installed.

Once the CPU_{index} has been established, the goal is to construct a fitting surface function that can forecast the OTE, given the CPU_{index} and the number of facets in the 3D digital

Table 2 The computational time required to complete the optimisation process for a dataset of 9 models for $CPU_{index} = 0.9987$ using 10 and 20 particles

	STL dimension [KB]	N. of facets	Computational time with 10 particles [s]	Computational time with 20 particles [s]
	4	12	15.72	36.32
	6	20	24.67	50.41
	434	1638	79.28	157.71
	434	1638	72.47	219.53
	1039	3756	164.12	234.64
	1056	3918	85.92	250.57
	3654	74816	834.87	2979.86
	19627	401952	6893.07	11259.41
	43930	899682	12980.2	28963.8

model. The authors chose a small dataset with nine digital models to establish a relationship between the OTE and the facet number. Ten and twenty particles have been used as an excellent preset value for the modified PSO algorithm to find optimal orientation. Some models have been pre-rotated

from the best orientation to access the capacity of the optimiser to determine and return to the initial position. The timings for a single computer with a CPU_{index} of 0.9987 are collected in Table 2, and the same dataset is used on two more PCs with CPU_{index} of 0.3027 and 0.5980.

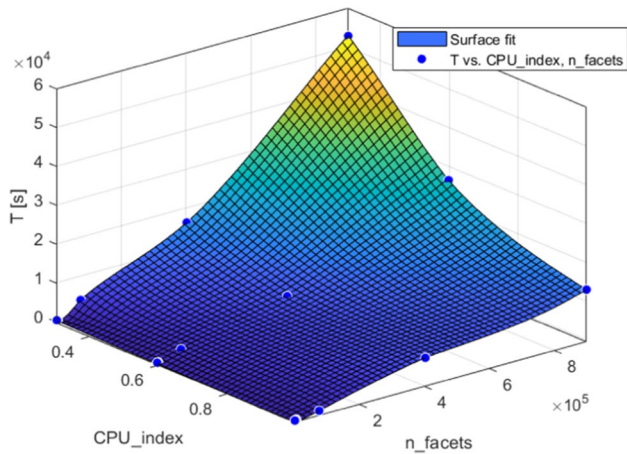


Fig. 15 Fitting surface of the computational time required to optimise the orientation built using a small dataset of 3D models as a function of the CPU_{index} and the number of facets (n_{facets}) of the STL files using ten exploring particles; T represents the time required for the optimal orientation algorithm to find the best solution (in seconds)

Once all of the simulations have been run on the three different computers, a 3D surface may be constructed to accommodate the data for a fixed number of particles (Fig. 15), and the OTE can then be assessed using the nearest-neighbour interpolation available in the Python *scipy* module.

In the following section, different case studies involving industrial 3D models will be described in depth. The results demonstrate encouraging results and good accuracy for the OTE prediction.

4 Case studies

Three industrial case studies are described in this section to assess the effectiveness of the proposed methodology and comprehend how optimisation influences the production process. The digital models of significant parts are depicted in Fig. 16 in their original orientation. These models, representative of parts used in industrial applications,

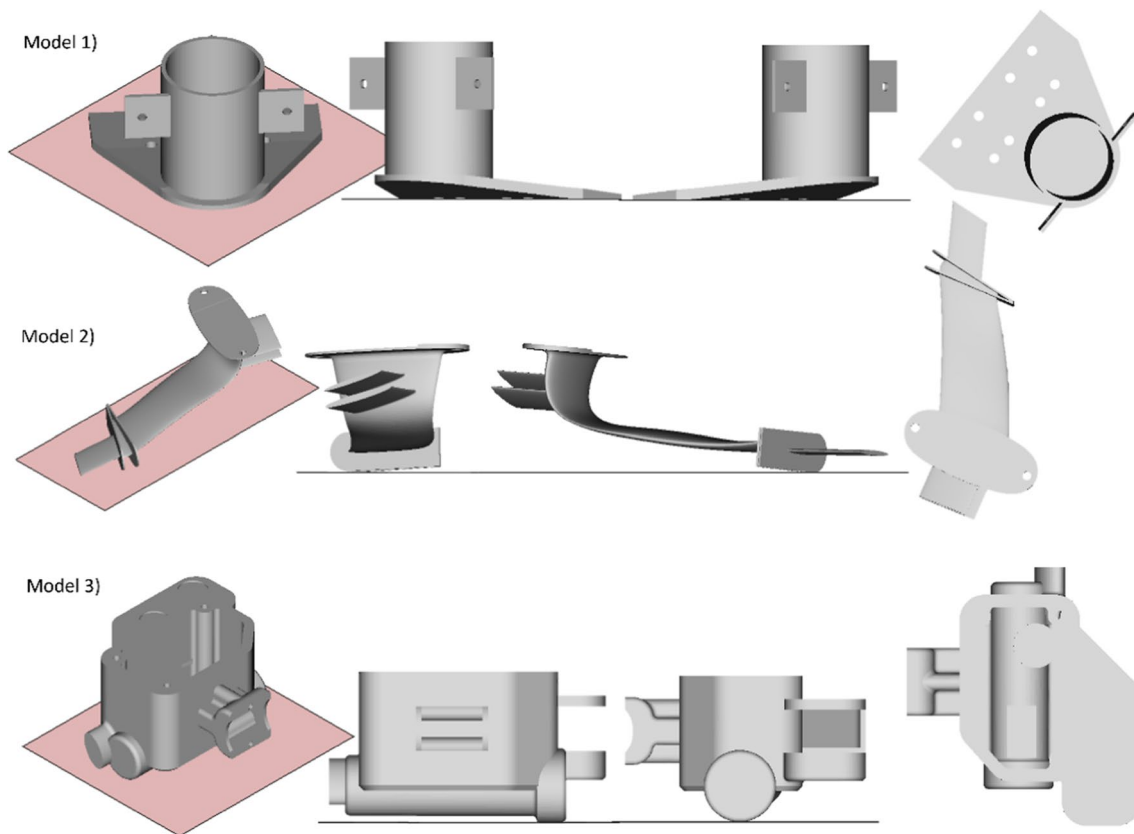


Fig. 16 Three digital models were used to test the proposed approach: from left to right, the original orientation is shown using isometric, frontal, lateral and upper views

Table 3 The computational time comparison between the proposed approach and the method presented in [25] for all the three digital models used as the case study of this research

N. of checkerboard subdivisions	Proposed algorithm running time [s]	Support volume estimation error [%]	Running time for the algorithm presented in [25] [s]	Support volume estimation error [%]	Speedup time ratio ([25]/Proposed)
Model 1 — 2510 kb; 51,392 facets; 75×100×61 bounding box $L \times W \times H$ [mm]					
20	1.22	2.10	45.53	1.21	37.44
30	2.50	7.52	112.27	2.09	44.89
40	4.33	0.97	154.48	0.78	35.69
50	6.35	6.25	370.90	1.79	58.43
60	9.45	2.94	549.50	1.01	58.15
Model 2 — 3838 kb; 78,594 facets; 159×59×60 bounding box $L \times W \times H$ [mm]					
20	0.89	5.93	194.93	0.34	219.77
30	1.87	4.89	315.25	0.92	168.49
40	3.40	3.79	637.41	0.14	187.53
50	5.22	3.03	894.75	1.89	171.44
60	7.45	2.82	1143.80	1.48	153.49
Model 3 — 17, 287 kb; 354,034 facets; 105×97×59 bounding box $L \times W \times H$ [mm]					
20	1.30	0.27	324.34	1.72	249.68
30	2.66	0.25	739.99	2.33	277.88
40	4.45	0.13	969.49	1.74	218.01
50	6.96	0.85	1487.40	1.58	213.77
60	10.17	1.56	2649.70	1.42	260.59

Table 4 Some data about the three models concerning the original orientation

	Support material [cm ³]	Manufacturing time [min]	Raw material mass [g]	% of overhang triangles in the footprint
Model 1	9.04	207	27	28
Model 2	102.28	840	145	47
Model 3	101.53	1438	274	58

are characterised by a complicated shape that precludes a clear preferred orientation that minimises the use of support material. Therefore, they can be regarded as good candidates to test the AMVER framework.

The topological characteristics of these components are compared in terms of the required support volume amount to manufacture the objects using the AM process. A second comparison uses the hypothetical time needed to manufacture the digital models. Fused deposition modelling (FDM) has been chosen for easy comparison, although the same results may be applied to any AM method. Moreover, additional data demonstrate how the improved PSO behaves during optimisation.

A comparison of the computational time and estimation accuracy between the proposed support volume algorithm and the method described in [25] is collected in Table 3 when applied to the geometries shown in Fig. 16. These data come from a single support volume estimation algorithm run

not included in an optimisation framework. An overhang angle threshold of 55° has been set. A typical evolutionary/meta-heuristic algorithm requires hundreds of runs of the volume evaluation routine; thus, the time spared is magnified in those cases.

The AMVER framework is used to analyse the volume of support material needed, and Cura Ultimaker [41], a slicing software explicitly designed for the FDM technique, is used to evaluate the amount of raw material and manufacturing time required with the original orientation. The amount of support material is assessed with a 100% support infill density, and a maximum overhang angle of 55° is considered. The resulting information is gathered in Table 4, and the amount of support material is evaluated with AMVER using a 30×30 checkerboard grid. In the last column of Table 4, the percentage of footprint covered by triangles in overhang aligns with the assumptions considered in Sect. 2.4.

The digital models have been imported one at a time into the AMVER framework to optimise the part orientation and reduce the raw material required to produce the component through AM techniques, focusing on support material minimisation. The simulations were run on a workstation with 128 GB of RAM and an Intel i9-11900 at 2.50 GHz CPU, which is associated with a $CPU_{index} = 0.9852$. Additionally, the authors selected $c_1 = 3$, $c_2 = 1.1$ as learning parameters and $w = 0.5$ as the velocity weight for the PSO. These values follow the convergence criteria for the PSO and can be assumed to be good candidates for preset values.

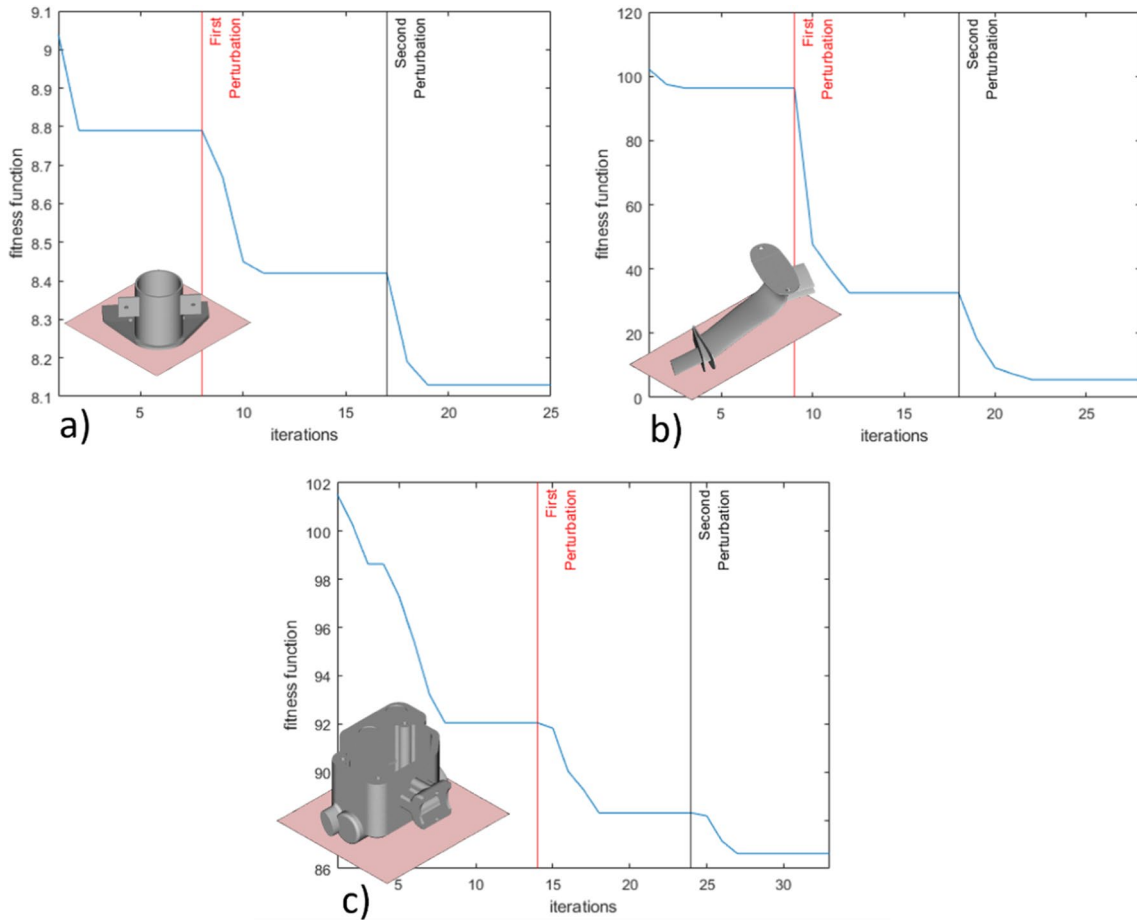


Fig. 17 Fitness function behaviour (support material volume in cm³) of the three case studies during the iterations of the optimisation algorithm: **a)** Model 1, **b)** Model 2 and **c)** Model 3

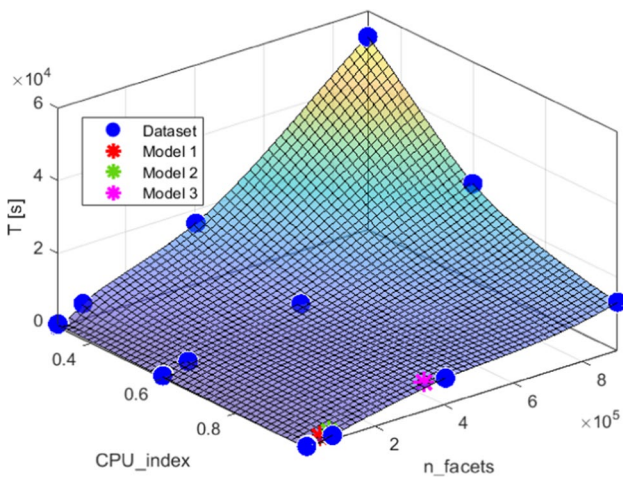


Fig. 18 OTE of the three case studies shows good agreement with the built surface fit for ten particles; T represents the time required for the optimal orientation algorithm to find the best solution (in seconds) as a function of the CPU_{index} and the number of facets (n_{facets}) of the STL files

The maximum number of iterations the optimiser can use has been adjusted at $n_{\text{max}} = 40$ after some testing to avoid making the simulations run too long. It is rare to reach a high number of iterations because the two sets of perturbations of the improved PSO, as stated in Sect. 2.6, always function appropriately before the 40th iteration. Indeed, Fig. 17 collects the fitness function behaviour during the iterative process that involves the PSO methodology for the three case studies proposed in this section.

Table 5 Main results of the optimisation of part orientation using the improved PSO

	N. of iterations	Improved PSO running time [s]	Optimised support volume [cm ³]	Roll[°]	Pitch[°]
Model 1	25	509	8.13	-2	88
Model 2	28	1124	5.57	152	85
Model 3	33	6435	86.62	-12	-4

Table 6 Comparison between original and optimised orientation according to some manufacturing parameters

	Support volume [%]	Manufacturing time [%]	Raw material mass [%]	Manufacturing time of supports [%]
Model 1	-10.10	+1.97	-11.11	-12.33
Model 2	-94.55	-52.98	-74.48	-84.62
Model 3	-14.69	+0.42	-5.47	-6.12

The AMVER framework provides an OTE for each of the three digital models before beginning the optimisation. The OTEs coming from AMVER, in terms of a surface fitting function, and the actual simulation times needed to find the global optimum are represented visually in Fig. 18. From a quantitative point of view, the case studies' errors on the OTE are 10%, 8%, and 2%, respectively. This indicates a generally positive agreement and suggests that the models used to construct the dataset presented in the previous section were adequately selected. In an industrial context where the design-to-manufacturing cycle has stringent deadlines, the user must know in advance how long the optimisation process will take, and the computational times should be reduced as much as possible.

The design space was explored using 10 particles to determine the ideal orientation. Table 5 aggregates the key outcomes from the improved PSO technique, including the total number of iterations, the running time, and the optimal amount of support material.

Table 6 summarises the percentage variations of a few metrics comparing the original and optimum orientations, considering all the information presented in this section to provide a clear picture of how the suggested framework influences manufacturing time and costs.

4.1 Discussion of the results

The results gathered in the previous section are thoroughly examined to demonstrate the effectiveness of the suggested methodology, which aims to optimise the part orientation of STL models to reduce the volume of support material needed for proper manufacturing and, as a result, comply with DfAM rules. Especially with manufacturing techniques based on metallic powders, removing support is a manual operation that is time-consuming and painstaking.

As the primary outcome, Table 1 and Table 3 compare the performances of the proposed algorithm with the approaches available in the literature; among them, the authors have implemented the algorithm based on the description included in the paper ([25]). The results in Table 1 show a significant decrease in computational time, with 1.33 times fast computation without compromising the accuracy of the support volume estimation (13% mean error compared to 12% of the [25] method). The advantages of the proposed

approach are magnified when dealing with the complex geometries typical of industrial applications. Indeed, the speedup efficiency increases by 47, 180, and 244 times for models 1, 2, and 3, reducing to a few seconds instead of several minutes of the approach presented in [25]. This result demonstrates that the computational gain increases with the increase in geometry complexity. The estimation error is limited to a maximum of 7% to the actual amount of support compared to 2% of the approach described in [25].

Thus, implementing the proposed approach in an iterative optimisation framework amplifies the computational gain by far. The algorithm herein presented can be an alternative to approaches such as the one presented in [25] when complex shapes must be analyzed or iterative procedures are implemented: The decrease in support volume precision is compensated by far with a dramatic decrease in computational time. Therefore, when a high precision in the computation of the support volume is required, the approach in [25] is preferable, but when a trade-off between precision and computational time is necessary, our algorithm shows advantages.

Moving to the optimisation framework, the OTE function deserves special attention since it provides a reliable estimate of the time needed to conclude the optimisation process with just about a 10% error compared to the actual running time of the AMVER framework for all three models. This error can be considered acceptable because of the philosophy of this evaluation. The OTE computation is based on a match with a surface fitting function developed using a dataset that aims to gather various potential scenarios that the AMVER framework could encounter.

The AMVER orientation optimisation framework, which incorporates a tailored version of the PSO, achieves encouraging results in the reduction of support material needed to manufacture the items by comparing the data collected in Table 4 (original orientation), Table 5 (optimised orientation), and Table 6 (variation analysis). The support volume decreases by 10, 94, and 14%, respectively, for models 1, 2, and 3. A straightforward consequence of support volume reduction is the lower time spent to deposit material to build the supports during the AM process. Just to spotlight a case study where the algorithm can be applied, by reducing the support volume, the total raw material consumption consistently reduces by 11, 74, and 5%, respectively, for models 1, 2, and 3. The improved PSO algorithm took less than 40

iterations (threshold value) to achieve this result (respectively 25, 28, and 33 iterations) thanks to two perturbations of the particles. As expected, the overall optimisation framework running time is proportional to the number of facets of the STL file that describe the complexity of the geometry. The improved PSO running time for the specific cases was 509 up to 6435 s. The algorithm herein presented could also be implemented in all the optimisation procedures requiring iterative evaluations.

From the outcomes collected in Table 6, it is straightforward that the digital model's topology significantly impacts the approach's effectiveness. Model 2, which has a considerable elongation along one dimension, significantly reduces support material (94%) because the original orientation was unfavourable. Models 1 and 3, which have comparable lengths in all three dimensions, have a lower support material reduction (10–15%) since the initial orientation was near the ideal one. However, this encouraging result shows that the AMVER framework can determine a component orientation that minimises the amount of support material, fully accomplishing the initial aim of the suggested technique.

Moreover, Fig. 17 demonstrates the effectiveness of perturbations in the PSO algorithm. In all three case studies, the general behaviour of the fitness function slightly changes: a sudden drop after a few iterations (model 3) or a more stable behaviour (model 1). However, what characterises all the case studies is that after the PSO is trapped in a local minima, the first perturbation can move the particles properly to escape from the plateau region. Without the implementation of the improved PSO, thus with the traditional PSO, the fitness function drop would be reduced to -2.88% , -5.81% , and -9.34% , respectively, for models 1, 2, and 3; on the other hand, to avoid particles trapped for the remaining iterations, the improved PSO reaches -10.10% , -94.55% , and -14.69% support volume reduction. Thus, it is immediately apparent that the improved PSO demonstrates encouraging results.

Lastly, Cura Ultimaker's estimates of the raw material consumption in each of the proposed case studies consistently show some percentage points drop (from 5 to 74% depending on the case studies). This finding is of the utmost significance when seen from the perspective of a product's whole development cycle.

5 Conclusions

The amount and position of support material are the most significant aspects that may impact additive manufacturing production. The build orientation affects the quantity of support material in most additive manufacturing methods. Therefore, accurately predicting the ideal build orientation is essential in today's competitive global market to minimise costs and increase the quality of the surfaces.

An accurate assessment of support volume factors that can be evaluated quickly and effectively is necessary to determine the appropriate build direction reliably: This is particularly true for enterprises because of the tight timelines for the product development cycle. As the manuscript shows, the more relevant works in the literature discuss many support volume estimation approaches that are precise and reliable, allowing its integration into iterative optimisation frameworks due to low computational times (less than a second). However, this manuscript applies those methodologies to three complex geometries typical of the industrial environment: A high computational time (several minutes) is noticed in this case, thus limiting the applicability in optimisation algorithms. The high computational time can be blamed for the ray-tracing methodology applied to the entire ground footprint of the model to analyse and determine possible overhangs. However, only certain regions of the model may require support material, thus inefficiently using the computational resources.

The strength of the method proposed in this paper is the numerically efficient way to estimate the support material accurately. Thanks to the three case studies in this paper, the comparison with the literature approach reveals a computational time gain of 47, 180, and 244 times compared to [25], with a 6% precision in the support volume estimation compared to the actual amount of support. These results are achieved by the automatic detection of the checkboard elements leaning under the triangles of the mesh and the application of the ray-tracing technique that is applied only to the underlying grid elements that contain at least one overhang triangle facet footprint of the model: This means that in that particular region, some overhangs may be present. The suggested method is particularly well suited for optimising situations where the solution requires numerous iterations and repeated evaluations. To prove its reliability, the support material estimation is integrated into an improved particle swarm optimisation method to prevent local minima and offer the best solution. An add-on capable of estimating the overall optimisation running time as a function of the computational power of the PC and the STL file's complexity has been developed and widely tested. The results demonstrate good accuracy (with less than a 10% error if comparing the OTEs and the actual optimisation running timings), indicating satisfactory attention for creating the dataset to build the fitting function for estimating optimisation times. Indeed, a proper time estimation could be fundamental in industrial applications to forecast and plan the different design and production cycle stages.

Three case studies with different complexities and random orientations have been used to validate the optimisation approach for the support material. The proposed algorithm can find the optimal orientations efficiently (support volume reduction from 10 up to 90% compared to original orientations) with running times compatible with the company's needs (an average of 0.7 h, which is negligible when

compared to the time required for creating the support material). The decrease in the amount of support material directly affects the manufacturing time and the amount of raw material used to manufacture the components, with reductions of up to 52% and 74%, respectively. This result is significant when considering a product's entire development cycle because companies could economise on the production process thanks to AM and an intelligent strategy of part orientation.

The tests included in this paper consider only the computation of the volume of the supports. Other simulations could be carried out considering other requirements linked to the growth ratio, the overhang area and surface, and the footprint on the building plate. In the future, the capabilities of the developed framework will be enlarged by including these critical factors in the AM process, which should be considered in the search for the optimal part orientation. In future work, implementing functions to consider such aspects could be integrated to carry out multidisciplinary optimisations.

In conclusion, the provided findings show that the proposed function used to estimate the support volume shows encouraging results with a considerable decrease in computational resources, making it appropriate for inclusion in commercial software for iterative optimisation: The reduction of computational speed is obtained paying a slight reduction in support volume precision respect to methods such as what presented in [25]. The tool used to test the algorithm developed herein can return the ideal component orientation for additive manufacturing procedures to minimise the support volume quickly. By doing so, the tedious process of trial and error to identify an adequate orientation to reduce the supports is eliminated, resulting in a significant improvement in time to market, a decrease in operator effort, and an increase in precision and final quality of components.

Acknowledgements The authors wish to thank Andrea Pasquali, CEO of Proxera Company, Italy, and Federico Ferrari, R&D engineer from ZARE Company, Italy, for their fruitful discussions and suggestions.

Author contribution All authors contributed to the study's conception and design. All authors read and approved the final manuscript.

Funding Open access funding provided by Alma Mater Studiorum - Università di Bologna within the CRUI-CARE Agreement.

Declarations

Competing interests The authors declare no competing interests.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not

permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

- Zhai Y, Lados DA, LaGoy JL (2014) Additive manufacturing: making imagination the major limitation. *JOM* 66(5):808–816. <https://doi.org/10.1007/s11837-014-0886-2>
- Murr LE (2016) Frontiers of 3D printing/additive manufacturing: from human organs to aircraft fabrications†. *J Mater Sci Technol* 32(10):987–995. <https://doi.org/10.1016/j.jmst.2016.08.011>
- Frizziero L et al (2021) An innovative and cost-advantage CAD solution for cubitus varus surgical planning in children. *Appl Sci* 11(9):4057. <https://doi.org/10.3390/app11094057>
- Donnici G, Ferretti P, Montalti A, Francia D, Liverani A (2024) FDM technology: overhangs versus layer height printability performance correlation. In: Carfagni M, Furferi R, Di Stefano P, Governi L, Gherardini F (eds) *Design tools and methods in industrial engineering III*. Cham, Springer Nature Switzerland, pp 58–65
- Leite M et al (2018) 3D printing of large parts using multiple collaborative deposition heads – a case study with FDM, presented at the 3rd International Conference on Progress in Additive Manufacturing (Pro-AM 2018). <https://doi.org/10.25341/D4WS3X>
- Lee CS, Kim SG, Kim HJ, Ahn SH (2007) Measurement of anisotropic compressive strength of rapid prototyping parts. *J Mater Process Technol* 187–188:627–630. <https://doi.org/10.1016/j.jmatprotec.2006.11.095>
- Ferretti P, Fusari E, Alessandri G, Freddi M, Francia D (2023) Stress-based lattice structure design for a motorbike application. *F1000Research* 11:1162. <https://doi.org/10.12688/f1000research.125184.2>
- Di Angelo L, Di Stefano P, Guardiani E (2020) Search for the optimal build direction in additive manufacturing technologies: a review. *J Manuf Mater Process* 4(3):71. <https://doi.org/10.3390/jmmp4030071>
- Bushra J, Budinoff HD (2021) Orientation optimization in additive manufacturing: evaluation of recent trends. In: Volume 5: 26th Design for Manufacturing and the Life Cycle Conference (DFMLC), Virtual, Online: American Society of Mechanical Engineers. V005T05A003. <https://doi.org/10.1115/DETC2021-71958>
- Qin Y, Qi Q, Shi P, Scott PJ, Jiang X (2021) Status, issues, and future of computer-aided part orientation for additive manufacturing. *Int J Adv Manuf Technol* 115(5–6):1295–1328. <https://doi.org/10.1007/s00170-021-06996-6>
- Zhang Y, Bernard A, Harik R, Karunakaran KP (2017) Build orientation optimization for multi-part production in additive manufacturing. *J Intell Manuf* 28(6):1393–1407. <https://doi.org/10.1007/s10845-015-1057-1>
- Singhal SK, Jain PK, Pandey PM, Nagpal AK (2009) Optimum part deposition orientation for multiple objectives in SL and SLS prototyping. *Int J Prod Res* 47(22):6375–6396. <https://doi.org/10.1080/00207540802183661>
- Ga B, Gardan N, Wahu G (2018) Methodology for part building orientation in additive manufacturing. *Comput-Aided Des Appl* 16:113–128. <https://doi.org/10.14733/cadaps.2019.113-128>
- Zwier MP, Wits WW (2016) Design for additive manufacturing: automated build orientation selection and optimization. *Procedia CIRP* 55:128–133. <https://doi.org/10.1016/j.procir.2016.08.040>

15. Zhang X, Le X, Panotopoulou A, Whiting E, Wang CCL (2015) Perceptual models of preference in 3D printing direction. *ACM Trans Graph* 34(6):1–12. <https://doi.org/10.1145/2816795.2818121>
16. Pham DT, Dimov SS, Gault RS (1999) Part orientation in stereolithography. *Int J Adv Manuf Technol* 15(9):674–682. <https://doi.org/10.1007/s001700050118>
17. Khodaygan S, Golmohammadi AH (2018) Multi-criteria optimization of the part build orientation (PBO) through a combined meta-modeling/NSGAII/TOPSIS method for additive manufacturing processes. *Int J Interact Des Manuf IJIDeM* 12(3):1071–1085. <https://doi.org/10.1007/s12008-017-0443-7>
18. Qie L, Jing S, Lian R, Chen Y, Liu J (2018) Quantitative suggestions for build orientation selection. *Int J Adv Manuf Technol* 98(5–8):1831–1845. <https://doi.org/10.1007/s00170-018-2295-0>
19. Paul R, Anand S (2015) Optimization of layered manufacturing process for reducing form errors with minimal support structures. *J Manuf Syst* 36:231–243. <https://doi.org/10.1016/j.jmsy.2014.06.014>
20. Chowdhury S, Mhapsekar K, Anand S (2018) Part build orientation optimization and neural network-based geometry compensation for additive manufacturing process. *J Manuf Sci Eng* 140(3):031009. <https://doi.org/10.1115/1.4038293>
21. Das P, Chandran R, Samant R, Anand S (2015) Optimum part build orientation in additive manufacturing for minimizing part errors and support structures. *Procedia Manuf* 1:343–354. <https://doi.org/10.1016/j.promfg.2015.09.041>
22. Mele M, Campana G, Bergmann A (2022) Optimisation of part orientation and design of support structures in laser powder bed fusion. *Int J Interact Des Manuf IJIDeM* 16(2):597–611. <https://doi.org/10.1007/s12008-022-00856-7>
23. Qin Y, Qi Q, Shi P, Scott PJ, Jiang X (2021) Automatic determination of part build orientation for laser powder bed fusion. *Virtual Phys Prototyp* 16(1):29–49. <https://doi.org/10.1080/17452759.2020.1832793>
24. Karim KF et al (2015) Feature extraction and optimum part deposition orientation for FDM. *Appl Mech Mater* 793:642–646. <https://doi.org/10.4028/www.scientific.net/AMM.793.642>
25. Di Angelo L, Di Stefano P, Dolatnezhadzomarin A, Guardiani E, Khorram E (2020) A reliable build orientation optimization method in additive manufacturing: the application to FDM technology. *Int J Adv Manuf Technol* 108(1–2):263–276. <https://doi.org/10.1007/s00170-020-05359-x>
26. Sheng H, Xu J, Zhang S, Tan J, Wang K (2023) Build orientation determination of multi-feature mechanical parts in selective laser melting via multi-objective decision making. *Front Mech Eng* 18(2):21. <https://doi.org/10.1007/s11465-022-0737-8>
27. Kundu R, Das S, Mukherjee R, Debchoudhury S (2014) An improved particle swarm optimizer with difference mean based perturbation. *Neurocomputing* 129:315–333. <https://doi.org/10.1016/j.neucom.2013.09.026>
28. Ceruti A, Bombardi T, Marzocca P (2017) A CAD environment for the fast computation of added masses. *Ocean Eng* 142:329–337. <https://doi.org/10.1016/j.oceaneng.2017.07.026>
29. Hashim HA (2019) Special orthogonal group SO(3), Euler angles, angle-axis, Rodriguez vector and unit-quaternion: overview, mapping and challenges. <https://doi.org/10.48550/ARXIV.1909.06669>
30. Qu X, Stucker B (2005) Circular hole recognition for STL-based toolpath generation. *Rapid Prototyp J* 11(3):132–139. <https://doi.org/10.1108/13552540510601255>
31. Möller T, Trumbore B (1997) Fast, minimum storage ray-triangle intersection. *J Graph Tools* 2(1):21–28. <https://doi.org/10.1080/10867651.1997.10487468>
32. Jarvis RA (1973) On the identification of the convex hull of a finite set of points in the plane. *Inf Process Lett* 2(1):18–21. [https://doi.org/10.1016/0020-0190\(73\)90020-3](https://doi.org/10.1016/0020-0190(73)90020-3)
33. Keshavarzadeh V, Meidani H, Tortorelli DA (2016) Gradient based design optimization under uncertainty via stochastic expansion methods. *Comput Methods Appl Mech Eng* 306:47–76
34. Fishman G (2013) Monte Carlo: concepts, algorithms, and applications. Springer Science & Business Media
35. Goldberg DE (2006) Genetic algorithms. Pearson Education India
36. Shi P et al (2023) Learn to rotate: part orientation for reducing support volume via generalizable reinforcement learning. *IEEE Trans Ind Inform* 19(12):11687–11700. <https://doi.org/10.1109/TII.2023.3249751>
37. Kennedy J, Eberhart R (1995) Particle swarm optimization. In: Proceedings of ICNN95 - International Conference on Neural Networks, Perth, WA, Australia: IEEE 1942–1948. <https://doi.org/10.1109/ICNN.1995.488968>
38. Alicandro M, Di Angelo L, Di Stefano P, Dominici D, Guardiani E, Zollini S (2022) Fast and accurate registration of terrestrial point clouds using a planar approximation of roof features. *Remote Sens* 14(13):2986. <https://doi.org/10.3390/rs14132986>
39. Trelea IC (2003) The particle swarm optimization algorithm: convergence analysis and parameter selection. *Inf Process Lett* 85(6):317–325. [https://doi.org/10.1016/S0020-0190\(02\)00447-7](https://doi.org/10.1016/S0020-0190(02)00447-7)
40. FreeCAD (2001) FreeCAD project association (FPA). Accessed: Feb. 29, 2024. [Multi-platform]. Available: <https://www.freecad.org/>
41. Ultimaker Cura. Accessed: May 30, 2022. [Online]. Available: <https://ultimaker.com/it/software>

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.