

Efficient milling part geometry computation via three-step update of frame-sliced voxel representation workpiece model

Jimin Joy¹ · Hsi-Yung Feng¹

Received: 2 September 2016 / Accepted: 15 February 2017 / Published online: 1 April 2017
© Springer-Verlag London 2017

Abstract This paper presents an efficient method of computing machined part geometry in general multi-axis milling. The method is based on the frame-sliced voxel representation of the workpiece geometry and involves a three-step update process. Virtual prototyping of the machined part geometry is useful for tool path verification and demands fast computations for quick feedback. The frame-sliced voxel representation (FSV-rep) model retains the advantages of efficiency and robustness of voxel modeling in model update along with the accuracy and boundary representation quality comparable to that of triangle meshing. The multi-level FSV-rep model allows a three-step update process of the model, which enables batch processing of the voxels and minimal intersection calculations to achieve fast and accurate modeling results. In a series of test cases, the FSV-rep-based computation has shown up to 3.3 times faster performance compared to that of the existing tri-dexel-based method with similar modeling accuracy. The better performance of the present method is attributed to the bulk workpiece volume to be removed and the complex multi-axis motions of the cutting tool with tilted orientations.

Keywords Part modeling · Milling · Geometry simulation · Efficiency · Accuracy

1 Introduction

Milling is a versatile machining process used for the production of many complex mechanical parts. Whether it is three-axis milling with linear tool motions and fixed tool orientations or five-axis milling with curved tool motions and varying tool orientations, modeling and simulation of the machined part geometry are often used as a preparatory step to verify and optimize the process prior to the actual machining operation [1]. Nowadays, elaborate simulation considering the cutting physics of the machining process has emerged and is termed as virtual machining. In milling simulation, whether cutting physics is considered or not, it is necessary to geometrically represent the workpiece blank and cutting tools and their paths. Such information is then used to compute the in-process workpiece geometry, instantaneous cutter-workpiece engagement geometry, and machined part geometry. The challenge is to perform these geometric computations for general milling operations with complex tool paths and changing tool orientations.

Computing the machined part geometry is one of the most important tasks in milling simulation. The machined part geometry is the shape of the resulting workpiece after the workpiece geometry is updated with all the tool paths. Modeling and prediction of the machined part geometry are important for two main purposes in general milling [2]. First, it can verify if the tool paths will reliably generate the desired part geometry within the specified tolerance [3]. Second, possible defects such as gouging and undercuts can be correctly detected. In particular, gouging is a serious issue for milling and requires advanced techniques for effective elimination [4, 5].

In order to efficiently model the machined part geometry, a method which is fast irrespective of the amount of material removed is needed. This is, however, an idealistic

✉ Hsi-Yung Feng
feng@mech.ubc.ca

¹ Department of Mechanical Engineering, The University of British Columbia, Vancouver, BC V6T 1Z4, Canada

target, and a method that is least affected by the volume of material removed shall be favored. Apart from efficiency, accuracy and robustness of the geometric modeling method are also important. As it is presently not possible to have a method that is accurate, robust, and efficient, a practical method should aim to achieve good performance in all the three aspects to the best extent possible. This is because improvements in accuracy and efficiency actually conflict with each other in the existing methods. Best levels of accuracy and robustness are indeed very desirable. However, the involved computational time and hence the efficiency are also critical.

Apart from computation of the machined part geometry, an efficient modeling method is also beneficial for other applications of machining simulation and verification such as machine/tool collision detection and avoidance, mechanistic simulation, and online process control. Collision detection and avoidance is a crucial part of machining simulation especially for multi-axis operations. The tool and tool holder motions need to be verified in order to avoid collisions. Interference checks using geometric models of the machine structure and simulation of the entire machine tool kinematic motion have been utilized for this purpose [6–8]. Mechanistic simulation of the machining process to predict cutting forces and the onset of chatter have been used to verify and optimize the process plan [9–11]. Geometric modeling is used to obtain the essential inputs of in-process workpiece and cutter-workpiece engagement required for the mechanistic simulation. The online process monitoring and control is another machining technology that calls for effective geometric modeling. Research studies towards assisting the augmented reality to enhance observation of an ongoing machining process have been reported [12, 13]. A CAD/CAM system fully integrated into a machining center is also an emerging reality. All of the machining technologies stated above will benefit from an efficient method to simulate the machining operation, apart from process verification which is the primary focus of this work.

Geometric modeling methods currently used for milling simulation can be broadly classified into solid modeling, vector modeling, and space partitioning methods. Solid modeling methods are best in accuracy, whereas vector modeling and space partitioning methods are good in efficiency. In this work, a recently developed modeling method referred to as the frame-sliced voxel representation (FSV-rep) method [14] is employed to achieve the best attainable levels of efficiency and robustness while providing good accuracy for the machined part geometry in multi-axis milling simulation.

The rest of the paper is structured as follows: Sect. 2 will review the existing geometric modeling methods used for milling simulation and discuss their suitability for machined

part geometry computation. Section 3 will describe the main features of the FSV-rep method employed in this work. Section 4 will give the efficient three-step update process of an FSV-rep model for fast computation of the machined part geometry. Section 5 will provide specific implementation details. Case studies and implementation results are to be discussed in Sect. 6, followed by conclusions in the last section.

2 Existing methods

Solid modeling methods aim to give an exact model of the workpiece and are widely used since initially for milling geometry simulation [15, 16]. The workpiece solid model has to be updated with the Boolean subtraction operation from the swept volume of the cutting tool along a tool path. As a result, as the number of tool paths increases, the computational load to maintain and process the tree of numerous tool swept volumes in order to generate the final part geometry grows significantly. This makes the exact solid modeling methods less viable even though they are the most accurate. Triangle mesh, a linearized boundary representation scheme of solid modeling, has also been applied to milling simulation [17, 18]. The involved workpiece update process is much simpler than exact solid modeling since only triangle-triangle intersections are to be carried out. Triangle mesh models can provide acceptable modeling accuracy by adjusting the triangle size according to the local surface curvature. However, the need to constantly update the adjacency and connectivity data in order to maintain a manifold triangle mesh model affects the performance of simulating a repeatedly modified workpiece model. To facilitate the process and also to identify the intersecting triangles faster, a localization technique has been reported [19]. In practice, still a large number of triangles are often needed for acceptable representations of freeform surfaces present in machining simulation. This increases the number of triangle-triangle intersections, and it is still a concern.

Vector modeling methods using one or more sets of parallel line segments called vectors are a class of discrete modeling techniques used for machining simulation. The Z-map model which is a 2D array of vectors with one end on a fixed plane and the other end on the top surface of the modeled workpiece is a typical example [20, 21]. Vector models are updated by a simple clipping operation which cuts away portions of the vectors corresponding to the removed workpiece material. Since the vectors can be directly accessed from an indexed array, the update computations can be localized much easier than those for the solid modeling methods. Thus, the Z-map method is quite useful for simulating the three-axis milling operations for parts which do not have overhangs. One issue with the Z-

map method is its lack of sensitivity for tool motion directions almost parallel to the Z vector direction. More importantly, the Z -map method is not applicable to multi-axis milling due to the likely presence of overhangs on the machined part. The dexel representation which is an extension of the Z -map method and uses more than one line segment at each Z vector location has been introduced to handle this situation [22]. The tri-dexel method (or the triple-ray representation), which is essentially three orthogonal dexel models combined together, is the most advanced vector modeling method that resolves the issue of poor sensitivity along the vector direction of any single set of parallel vectors [23, 24].

Vector modeling is known to be able to perform efficient update of workpiece models with simple localization for the removed workpiece volume. However, for direct computations of the machined part geometry, it still lacks the ability to minimize calculations as the workpiece model update process always happens at the finest grid resolution. This is quite an overhead especially for the tri-dexel model as the dexels from all the three orthogonal sets will have to be incrementally trimmed for every tool path segment. Further, intersection calculations are needed for all the dexels crossing the bounding box of a tool instance for identifying possible intersections.

Space partitioning is the third class of geometric modeling methods employed in milling simulation. The basic operation involved in the workpiece model update is the direct deactivation of volume elements that are in the material removal path of the cutting tool. This operation is of very low computational cost compared against the operations involved in the update of other class of models. A basic uniform grid voxel model is the easiest to implement [25, 26] but requires a high grid resolution to achieve modeling accuracy comparable to that of methods with acceptable accuracy such as triangle meshing. With the use of multi-level octree-based voxel grids [27], the number of voxels to be updated becomes much lower. This means that implementations with practically affordable memory consumptions become possible with the multi-level voxel grids even though memory usage is often deemed the highest for the space partitioning methods in general. Recently, an octree-based space partitioning method has been augmented with composite adaptively sampled distance fields [28] in order to improve the model accuracy without further increasing the grid resolution. However, the method is adversely affected by the computationally intensive steps to obtain the part surface from the distance fields, thereby reducing the overall computational efficiency.

In this work, the volumetric space partitioning approach is deemed the most promising approach that can provide the best combination of accuracy, efficiency, and

robustness. In particular, a new machined part modeling format referred to as the FSV-rep model has been introduced by the authors to keep the model size down and achieve the subvoxel resolution and accuracy [14]. To ensure the computational efficiency, a three-step process to update the FSV-rep model of the machined part geometry has been developed in this work. It will be shown that the method is much faster than the state-of-the-art tri-dexel method for the same level of accuracy.

3 FSV-rep model

FSV-rep is developed with a focus on achieving the best level of efficiency and accuracy for machining simulation [14]. FSV-rep is fundamentally based on voxels. The entire modeling space is first represented by a coarse voxel grid that is affordable memory-wise. The resolution and spacing of the coarse grid correspond to a voxel size which can be efficiently updated by the cutting tools employed in the machining operation. FSV-rep uses surface voxels, which are the voxels through which the surface of an object is passing, to represent the shape of the object. The use of surface voxels alone provides a sparse representation, thereby reducing the voxel model size in representing the object. Further, once the coarse-level surface voxels are identified from the coarse voxel grid, subgrids of smaller spacing are defined within the coarse-level surface voxels in order to get finer voxels along the boundary surface of the workpiece volume without the need to increase the resolution of the initial coarse grid. Smaller fine-level surface voxels are identified from the finer subgrids, making FSV-rep a sparse multi-level voxel representation.

This sparse multi-level FSV-rep model is memory-efficient with the model size roughly in the order of $\mathcal{O}(N_F^2)$ where N_F is the effective fine-level grid resolution for the modeling space [14]. Sparse representation was previously achieved for machining simulation using space partitioning-based geometric modeling. It was mostly done with the octree space subdivision approach [6, 19, 28]. Space partitioning with just three or four subdivision levels and with varying subdivision factor between the levels using dynamic B-trees have also been reported for applications in the animation industry [29]. However, the size of the leaf node octants still has to be extremely low for acceptable accuracy in machining simulation. In other areas of application such as computer graphics, octree models augmented with additional information to enhance accuracy have been attempted. The use of contour planes within the surface octants is an example of such an improvement [30]. Even though this gives appreciable visual

display improvements, the actual continuity of the surface approximation is not available. Further, such a representation could be applied for a static model needed for display. It would be computationally expensive to update such a model in machining simulation.

As stated earlier, the use of voxel models with a high resolution is undesirable due to the resulting large model size and computational load. Hence, the finest grid resolution of an FSV-rep model is set according to a practical limit considering the memory usage and computational load. In order to further improve the accuracy of the model, the FSV-rep model uses frame-sliced (FS) voxels. FS voxels are defined by the frame-crossing (FC) points, which are the intersection points of the workpiece surface with the edge frame of the fine-level surface voxels. Because the FC points can be anywhere on the voxel edge frame, they are able to approximate the workpiece surface geometry at a subvoxel level. This is achieved by generating a triangle mesh-based boundary representation for the machined part geometry from the FS voxels. Subvoxel geometric details can thus be represented through the use of memory-efficient FC points [14].

It should be noted that the FC points are solely associated with the surface voxels of the finest resolution in an FSV-rep model. As a result, the involved workpiece surface and voxel edge frame intersection calculations are only needed in the last step of generating the final machined part surface. For the intermediate bulk material removal from the workpiece throughout every tool path, unlike other methods which continuously carry out intersection calculations, the FSV-rep model update shall only involve the simple binary operation of marking/unmarking for the affected coarse voxels.

4 Three-step FSV-rep model update process

Geometry of the final machined part can be accurately and efficiently computed via a three-step update process for the involved FSV-rep model (Fig. 1). The coarse-level voxels are to be updated first to quickly remove the bulk of unwanted material from the workpiece model for a given set of tool paths. The fine-level surface voxels and their corresponding FC points are then updated in sequence in order to attain the desired model accuracy without much added computational load.

4.1 Coarse update

The coarse voxels in an FSV-rep model are used for quickly removing the bulk volume from the workpiece. Such an update only needs to be approximate, and the objective is to assure that all the coarse voxels are

classified (or marked) properly. Using sampled tool instances from all the tool paths, the coarse-level voxels can be classified as (I) definitely inside a tool instance, (II) definitely outside all the tool instances, or (III) possibly intersecting the envelope surface of a tool instance. Tool path sampling which takes tool instances along a tool path at a regular interval is one option to update the in-process workpiece in milling simulation and has been previously used to generate approximate cutter swept volumes [18, 31]. Other approaches such as solid modeling [32, 33], two-parameter family of spheres [21], and analytical definitions [28] have been used to obtain the cutter swept volume and update the in-process workpiece. Nonetheless, this work employs the method of sampled tool instances as it is a generic method that is applicable to all types of tools and tool paths and comparatively simple to implement for the voxel model update.

The coarse update step will delete the category I voxels from the FSV-rep model and mark the category III voxels as the near-field (NF) voxels for each corresponding tool instance. The category II voxels shall be untouched and remain in their current state. In particular, the coarse update will classify voxels inside the workpiece as category I, II, or III and then perform the associated marking/unmarking operations in order to reflect the removed volume from the workpiece. A relaxed proximity check of the voxel center against the tool envelope surface is used to identify the NF voxels. If the voxel side length is L_{vc} and the distance of the voxel center from the tool envelope surface is d_{te} , a voxel is deemed to be an NF voxel for the tool instance if

$$d_{te} < \frac{\sqrt{3}L_{vc}}{2} \quad (1)$$

where $\sqrt{3}L_{vc}$ is the “thickness” of the near-field region for the tool envelope which sits in the middle of the near-field region, as shown in dashed boundaries in Fig. 2.

The overall coarse update step is outlined in the algorithm 1 below. The coarse update step is done in two parts. In part 1, all the interior voxels of all the tool instances along all the tool paths are deactivated as not part of the workpiece volume. Then, in part 2, a list of near-field voxels $\{\mathbf{NFV}\}$ is created for each tool instance TI according to Eq. (1). After the two parts are done for all the tool paths, the coarse voxels along the final machined part surface are present in the resulting $\{\mathbf{NFV}\}$ sets. It should be noted that part 2 of the algorithm should be done after part 1 for all the tool paths in order to avoid a coarse voxel from entering into the $\{\mathbf{NFV}\}$ of a tool instance if some subsequent tool instance will deactivate it. Such voxels do not need to be considered as they will not contribute to the final machined part surface geometry.

Algorithm 1. Coarse Update

```

{TP} ← list of the milling tool paths
for each TP in {TP} do //Part 1
  {TI} ← set of sampled tool instances along the TP
  for each TI in {TI} do
    {BV} ← coarse voxels inside/intersecting TI's bounding box
    for each V in {BV} do
      if V active and V completely inside TI then
        Set V inactive
      end if
    end for
  end for
end for
for each TP in {TP} do //Part 2
  {TI} ← set of sampled tool instances along the TP
  for each TI in {TI} do
    {BV} ← coarse voxels inside/intersecting TI's bounding box
    for each V in {BV} do
      p ← center point of V
      if V active and p inside near-field of TI's envelope then
        {NFV} ← {NFV} + V
      end if
    end for
  end for
end for

```

An important output of the coarse update step is the mapping of a specific TI to its corresponding {NFV}, and it can now be used for the fine update step. Recall that the coarse update is a collective voxel removal at a lower resolution. Since the model update is done at the lower resolution, the involved computational time is not linked to the accuracy-dependent finest voxel resolution. The coarse update time is only dependent on the resolution for the coarse voxel level and the tool path sampling. In this work, the maximum voxel size for the coarse update is set via the cutting tool size as

$$L_{vc} < \frac{D}{\sqrt{3}} \quad (2)$$

where D is the diameter of selected cutting tool. More specifically, if the ratio of the fine to coarse voxel size is 1:4, each coarse voxel removal from the model corresponds to 64 fine voxels. Thus, voxel removal at the coarse level is done almost 64 times faster than at the fine level. This will be the case for all the voxels falling in category I of the coarse voxel update.

Thus, the majority of the coarse update is performed at an execution time faster by a cubic power of the subdivision factor from the coarse to fine levels when compared to the update of a voxel model with a single fine-level voxel grid. It is also important to have the multi-level voxel model starting at a coarse resolution governed by Eq. (2) rather than a complete octree subdivision of the entire modeling space. This is because only those levels with voxel sizes smaller than the cutting tool size will contribute to fast removal of the model volume. In particular, only voxels completely inside the cutting tool will be marked as removed, and this will not happen for any higher levels with the voxel size larger than the tool size.

4.2 Fine update

From the $TI \rightarrow \{NFV\}$ mapping obtained after the coarse update, an inverse mapping of $NFV \rightarrow \{TI\}$ can be obtained between each coarse NF voxel and the set of tool instances

possibly crossing it. Then, in the fine update step for each coarse NF voxel, the finer voxels within each coarse NF voxel have to be classified with respect to each tool instance in $\{\mathbf{TI}\}$ as (I) definitely inside, (II) definitely outside, or (III) intersecting the tool instance envelope surface. Algorithm 2 outlines the overall fine update step.

voxel model obtained after the fine update step is 26-separating. A 26-separating voxel model is one that ensures that all the voxels which an object surface passes through by crossing the edge frames of the voxels are part of the model. This facilitates the generation of a closed two-manifold triangle mesh for the desired model accuracy [14].

Algorithm 2. Fine Update
NVR \leftarrow number of coarse voxels to refine
for i from 1 to NVR do
CV \leftarrow i th coarse voxel to refine
$\{\mathbf{TI}\} \leftarrow$ tool instances crossing CV
$\{\mathbf{FV}\} \leftarrow$ fine-level voxels within CV
for each FV in $\{\mathbf{FV}\}$ do
$p \leftarrow$ center point of FV
for each TI in $\{\mathbf{TI}\}$ do //Part 1
if FV active and FV an interior voxel for TI then
Set FV inactive
end if
end for
for each TI in $\{\mathbf{TI}\}$ do //Part 2
if FV active and p inside near-field of TI's envelope then
if FV a surface voxel for TI then
Add FV to surface voxel model
end if
end if
end for
end for
end for

Like in the coarse update step, category I of the fine voxels is removed from the model (part 1 of algorithm 2 above) and category II voxels are untouched. Category III voxels are the new surface voxels for the updated model which are now identified as those voxels with confirmed intersection with the final machined part surface. These voxels are identified by exactly classifying every corner point with respect to the tool instance as inside or outside. For this purpose, a signed distance value [34], d_i , for each corner point has been used. A voxel is deemed as category III only if the condition of -8

$< \sum_{i=0}^7 \frac{d_i}{|d_i|} < 8$ is satisfied. The FSV-rep model is updated (part 2 of algorithm 2 above) with the new surface voxels to get the machined part geometry represented as a surface voxel model at the finest resolution. It should be noted that the surface

4.3 Frame update

Once all of the surface voxels at the finest resolution are obtained, frame update has to be done to these fine surface voxels in order to create the FS voxels. The FS voxels are created by computing the FC points for all the fine surface voxels. For a fine surface voxel edge, all the tool instances intersecting the edge can be retrieved from the NFV $\rightarrow \{\mathbf{TI}\}$ mapping. Line-surface intersection points are then calculated between each voxel edge and the tool instances in $\{\mathbf{TI}\}$ as a set of potential FC points. From the potential FC point set, those intersection points not inside of any tool instance are the actual FC points on the surface of the final machined part (Fig. 3). For each FC point, its location on the voxel edge has to be stored in a way to facilitate the subsequent reconstruction of the closed two-manifold triangle mesh surface from the FS

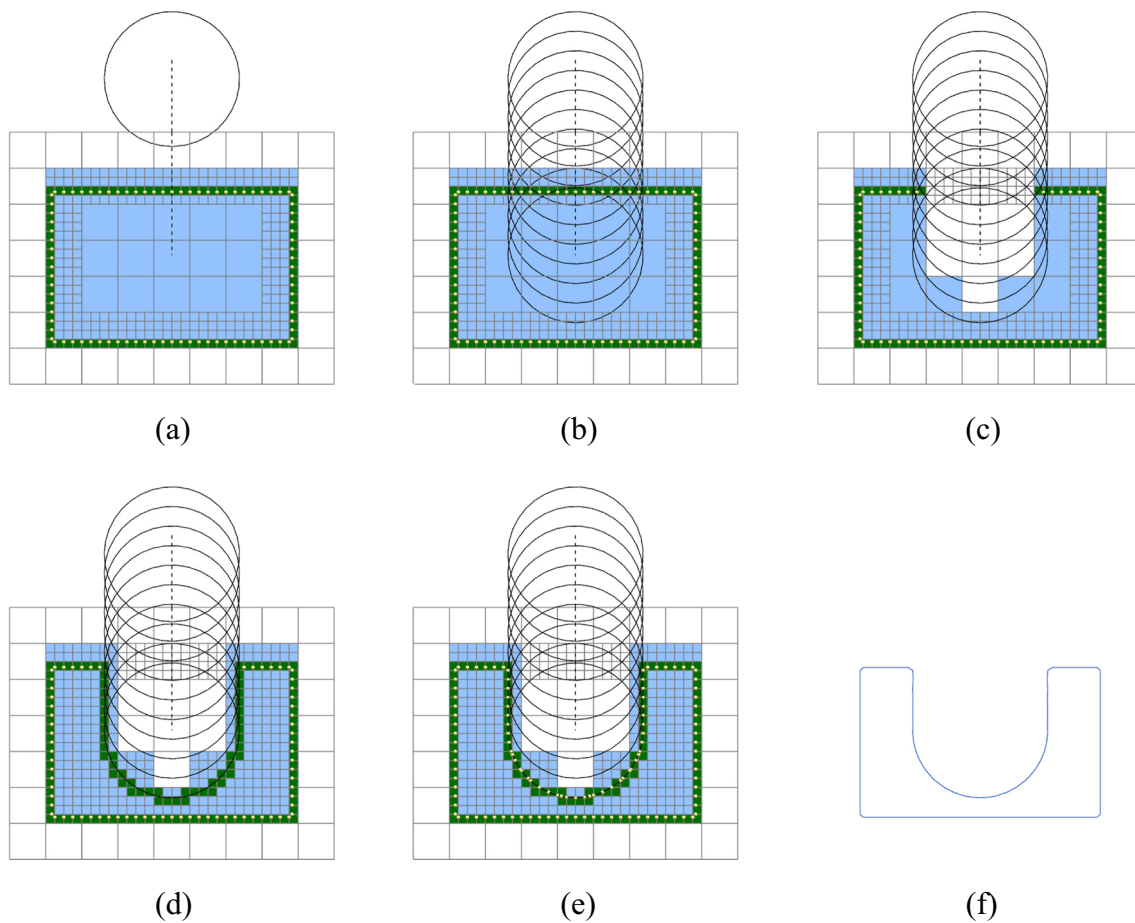


Fig. 1 Three-step update process for the machined part geometry. **a** Initial FSV-rep model with tool and tool path. **b** Created tool instances. **c** Coarse update. **d** Fine update. **e** Frame update. **f** Final machined part geometry

voxels. The FSV-rep model uses a pair of point locations on each voxel edge to define the FS voxels [14]. An FC point is to be stored as the first or second point in the pair according to the surface normal of the generating tool instance. If the surface normal component along the voxel edge is positive, the FC point is stored at the second location in the pair and at the first location if the surface normal component is negative. This ensures proper orientation of the triangles in the reconstructed triangle mesh surface.

5 Implementation details

The FSV-rep model with two levels of voxel grids (a coarse grid and a fine grid) was employed in the implementation of this work. The ratio of the fine grid voxel size to the coarse grid voxel size was set as 1:4. A one-dimensional array of binary variables (bit array) was used to represent the three-dimensional grid of voxels making the voxel space. As stated previously, the coarse grid is to span the entire modeling space, and the fine-level voxel grids are needed only within the coarse surface voxels that need refinement. In the FSV-rep

model, an integer ID is used to identify the bit in the bit array corresponding to a specific voxel, thereby achieving access to any voxel for activation or deactivation with a constant computing time.

For the surface voxelization of the original input workpiece shape, the bits corresponding to the surface voxels are set to 1 after setting all the bits to 0 initially. To facilitate the model update process, a volume voxel model for the coarse voxels is also needed which is generated by setting the bits corresponding to the voxels inside the model volume to 1 as well. All the tool paths specified in the milling operation are sampled individually with the sampling distance sufficiently small to make sure that all of the affected voxels are included in the model update process. A tool path is to be defined by the trajectories of two points on the tool axis with one point being the tool tip and the other being the point along the tool axis at a particular height from the tool tip [35]. The tool path is then sampled according to the sampling interval length on the two trajectories between two sampled tool instances. The sampling interval length on the tool path trajectories has to be set equal to or less than the voxel edge length in order to capture all of the affected voxels. The list of tool instances from each tool path

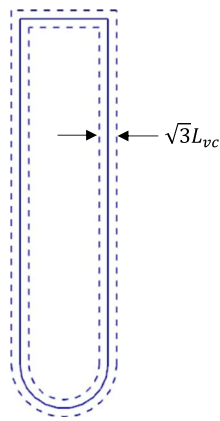


Fig. 2 Near-field region for the tool envelope (within the *dashed boundaries*)

for the entire milling operation is thus generated and used to update the FSV-rep model created for the original input workpiece.

During the update process of an FSV-rep workpiece model at the coarse and fine levels, the bits corresponding to the coarse voxels inside the tool instances are set to 0 in the bit array and the NF voxel lists are also created according to algorithm 1. The bit array representation of the fine-level voxel grid within each coarse NF voxel is also updated similarly according to algorithm 2: An active bit in the bit array for the fine-level voxel grid is set to 0 if the fine voxel is inside the tool instance, and if a fine-level voxel is found to be a surface voxel, an FS voxel is then computed as stated in Sect. 4.3.

Computing the FC points for all the edges of a fine surface voxel is not an efficient task as the computation will be repeatedly done for the same edge from all the four incident voxels. To avoid the redundant computations, the FS voxel holding the FC points only on the primary edges (as defined by Joy and Feng [14]) is to be used. In essence, an edge is to be deemed as the primary edge with respect to only one voxel.

As a result, every intersecting voxel edge will be the primary edge for just one of the fine surface voxels. In order to calculate the FC points for a primary edge, a wire body corresponding to the portion of the primary edge inside the original workpiece volume is defined first. Then, a Boolean subtraction operation is performed on that wire body using the solid bodies of the tool instances crossing the primary edge as the Boolean tools. The end points of the resultant wire bodies (excluding those coinciding with the voxel corners) are the FC points for the primary edge.

6 Case studies

A series of case studies have been carried out to demonstrate the improvement in the computational time of the present method based on the FSV-rep modeling to compute the milling part geometry as compared to that of the existing method based on the tri-dexel modeling. The tri-dexel method is employed as a comparison benchmark as it has been recognized as providing the best combination of modeling accuracy, robustness, and computational speed among the reported methods in the literature. Similar to the way the FSV-rep workpiece model was updated, the tri-dexel workpiece model was updated using sampled tool instances along the milling tool paths. However, as there had been no development of multi-level representations of tri-dexels, the model update had to be done in the finest resolution with incremental updates of the affected dexels with each sampled tool instance.

Three basic case studies were devised to illustrate the increasing complexity of the milling tool paths (Fig. 4). In all these cases, the machining was done by flank milling with a flat end mill. Only one flat end mill was used to make sure that the computing results were not dependent on the tool type but on the tool paths and tool orientations. In case I, the tool axis was always vertical along the tool paths in the machining of

Fig. 3 Intersection points (*blue* or *orange*) and actual FC points (*orange*) for an FS voxel

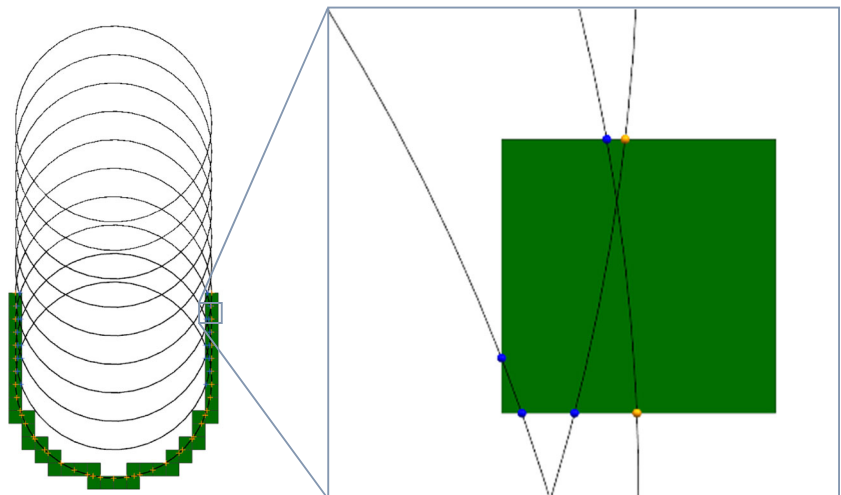
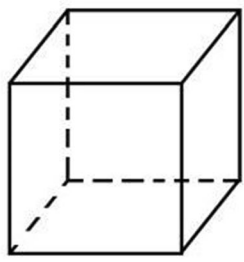
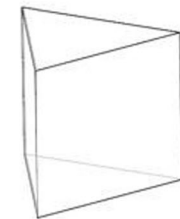
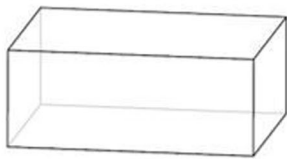
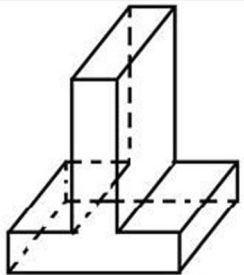
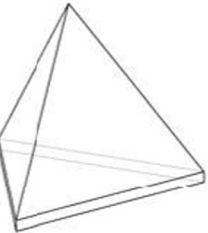
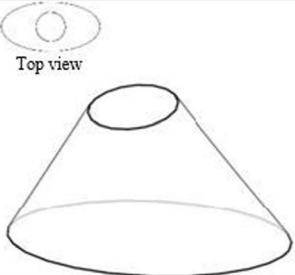


Fig. 4 Basic case studies: fixed vertical tool orientation (I), fixed tool orientation but tilted in one axial plane (II), and arbitrary and varying tool orientation (III)

Tool axis	No-tilt	1-tilt	2-tilt
Blank workpiece			
Target shape			
	Case I	Case II	Case III

the reverse-T part. The bounding box for each tool instance was the minimal in this case. Case II used a tilted tool with a constant orientation along each tool path and the tool axis being parallel to one of the axial planes of the workpiece coordinate system. The tool bounding box became larger in this case. Case III had the tool axis changing along each tool path and thus attaining an arbitrary 3D orientation. Compared with cases I and II, case III had the largest bounding boxes for the involved tool instances.

Figure 5 shows the computational time of the FSV-rep and tri-dexel methods for the three basic milling cases. It can be seen that the FSV-rep method gives faster performance in all the three cases, and the faster performance is more pronounced from case I to case III. The improvements are primarily from two factors as shown in Figs. 6 and 7. Figure 6 shows the execution time for simulating the machining of the

reverse-T part with the increasing value of the total axial depth of cut h_c . For very small values of h_c , the tri-dexel method is faster as the multi-level coarse and fine update of the FSV-rep method does not have much advantage. However, after h_c is larger than the coarse voxel grid spacing, the FSV-rep method becomes faster, and as h_c further increases, the advantage of the FSV-rep method becomes evident. The FSV-rep method achieves this via the collective volume removal by batch processing at the coarse voxel level first before moving to the fine voxel update and FC point computation. This facilitates the bulk material removal simulation at a much faster rate as compared to the sole procedure of intensive intersection calculations of the tri-dexel method to reach the final machined surface geometry. With the coarse update and identification of the NF voxels of the FSV-rep method, only those coarse voxels in the vicinity of the final machined surface are considered for

Fig. 5 Execution time comparison for computing the machined part geometry

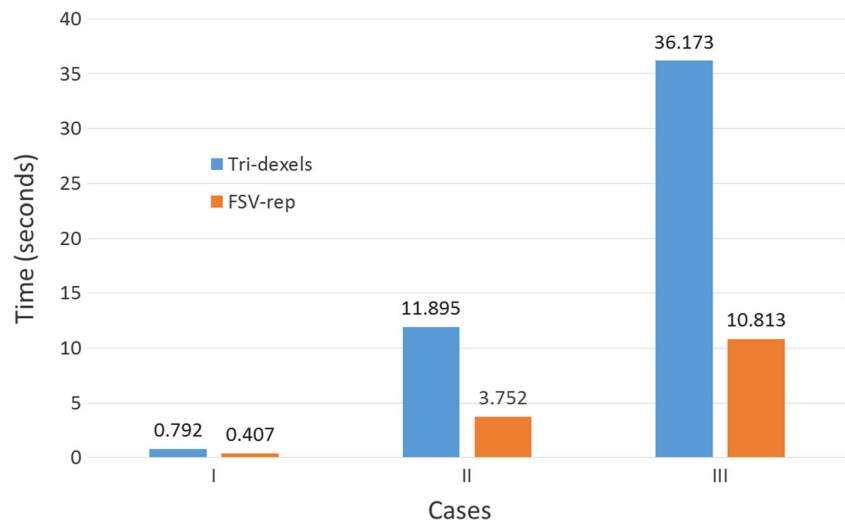
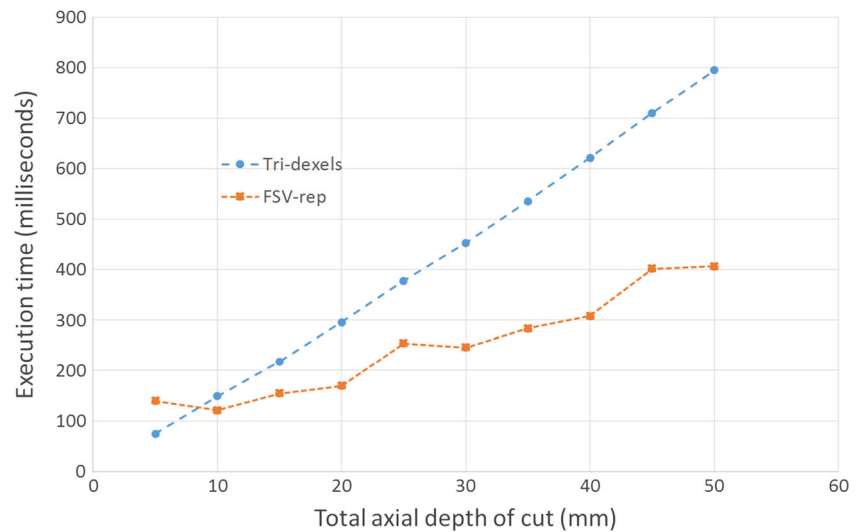


Fig. 6 Execution time with the increasing total axial depth of cut for the reverse-T part



the subsequent fine and frame update steps. As a result, fine surface voxels and FC points are computed only within the coarse voxels relevant to the final machined part surface.

A comparison of the respective execution time has been made among the coarse, fine, and frame update steps in order to have a better understanding on the proportional workload of the three different steps. As seen in Fig. 8, most of the execution time is spent on the coarse update. Since the coarse update step mostly involves a simple binary marking/unmarking operation, the large proportional workload gives the reason to the much faster performance of the FSV-rep method. The results also confirm that to obtain the FS voxels via the frame update to yield the higher model accuracy, the computational time needed is relatively insignificant after the coarse and fine updates.

Figure 7 illustrates the second factor contributing to the observed performance improvement of the FSV-rep method. It shows the execution time for simulating the machining of the reverse-T part with the width of the side cuts being only

half of the tool diameter. The machining simulation was done for different values of the forward tilt angle of the flat end mill along the tool path. It should be noted that the side cuts were completed using only one half-immersion tool pass with no tool path overlap. Hence, there is no advantage present for the FSV-rep method from the aspect of bulk volume removal. As can be seen from Fig. 7, after a particular forward tilt angle of the tool axis, the FSV-rep method becomes faster than the tri-dexel method and the time difference gets bigger with further increase in the tool tilt. This is due to the volume increase of the tool bounding box as the end mill becomes more and more tilted. With a larger bounding box, more elements (dexels or voxels) need to be considered and processed. Nonetheless, in the case of the FSV-rep method, rapid check at the coarse voxel level is attainable, and hence, the effect of the increased bounding box volume is much less. Furthermore, only point-to-tool distance classifications are involved in the identification of near-field and surface voxels, whereas for the tri-dexels, actual intersection calculations on those dexels

Fig. 7 Execution time with the increasing forward tilt of the flat end mill in the half-immersion side cuts for the reverse-T part

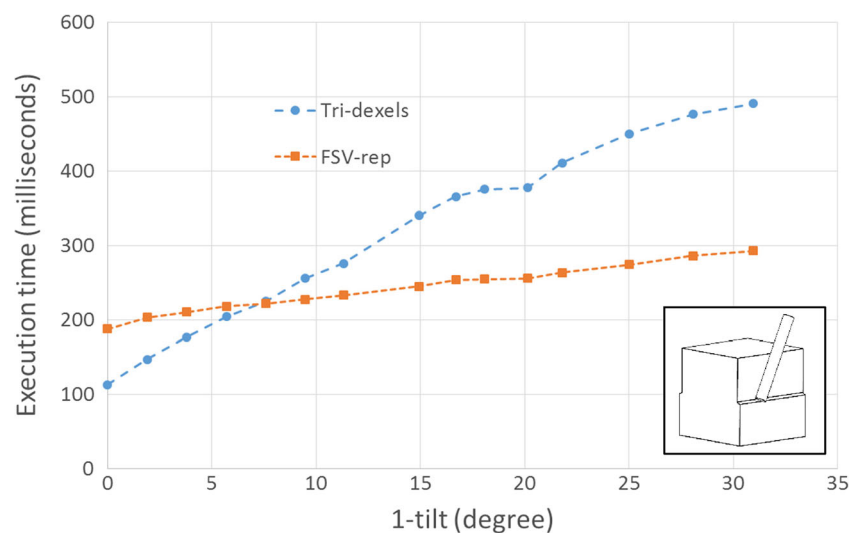
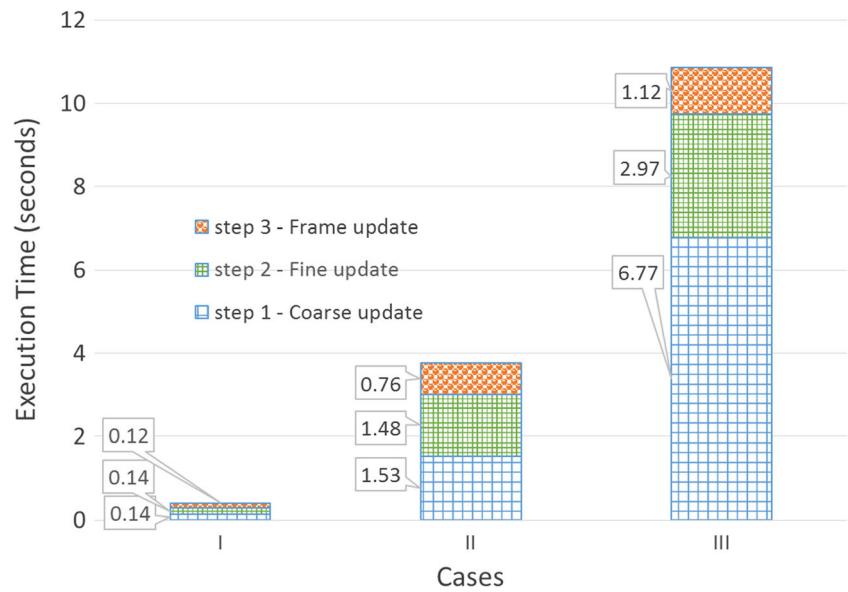


Fig. 8 Time splits among the coarse, fine, and frame update steps in the FSV-rep method



covered by the bounding box are needed to even confirm the intersections.

Figure 9 depicts the matching of the FC points determined in the FSV-rep method with the end points of the dixel line segments from the tri-dixel method for cases I and III with two representative zoom-in views. Close to perfect matching was obtained with virtually all of the FC points coincident with all of the tri-dixel end points except for some rare cases. Specifically, all of the tri-dixel end points were attained by the FSV-rep method in case I, and only 6 out of 103,270 tri-dixel end points were not attainable via the FSV-rep update process in case II and 4 out of 60,448 unattainable in case III. The minute difference is caused by an implementation restriction in the FSV-rep method which requires a maximum of two FC points to be stored on a voxel edge. The restriction is put in place for easier data management and subsequent identification of the surface orientation in reconstructing the triangle mesh from the FS voxels. The number of mismatch is seen to be fairly insignificant in general as noted from the extensive computational tests.

Generation of a triangle mesh surface for the machined part geometry is quite straightforward from an FSV-rep

model [14]. The triangle mesh surfaces obtained for the machined part geometry in the three test cases are shown in Fig. 10. The meshes are all of good quality and thus useful for the visual verification of the associated machining operations. More importantly, the meshes will be useful when performing a quantitative comparison against their reference design models for identifying potential machining errors such as gouging and undercuts. It should be pointed out, however, that the triangle mesh models obtained do not have sharp machined edges between faces. The improved accuracy of the FSV-rep model over the basic voxel model is due to the triangle mesh surface generated from the FS voxels and the associated FC points. The FS voxels are still not sensitive enough to capture the sharp machined edges and corners that are not coincident with the voxel edge frame. This is in fact a well-known issue for the discrete dixel or voxel representations. Since the deviation is only along the sharp edges of the machined part, this is a localized issue and only affects a relatively small area of the model. It can thus be easily resolved by a variety of triangle mesh-processing methods, for example, the method developed and demonstrated by Ren et al. [36].

Fig. 9 Matching of the FC points (green) from the FSV-rep method with the end points of dexels (blue lines) from the tri-dixel method for case I (left) and case III (right)

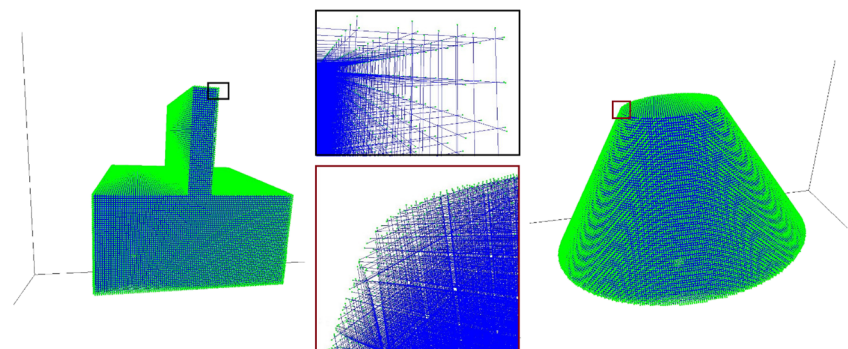
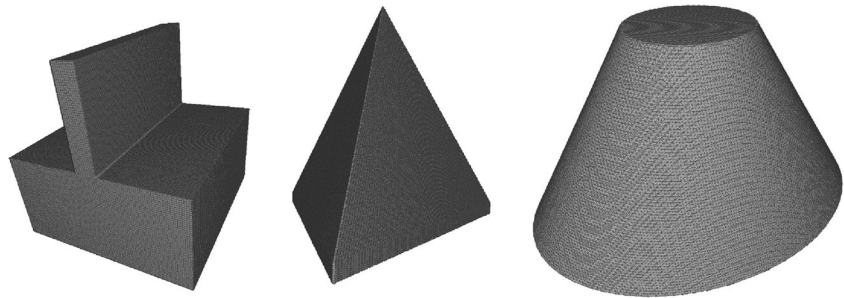


Fig. 10 Triangle mesh surfaces generated from the FSV-rep models



The case studies presented above only involve workpiece model updates with a flat end mill. Nonetheless, the overall model update process is general, and all types of milling cutters can be used. The use of sampled tool instances along a tool path for the workpiece model update represents an approximation to the exact tool swept volume. It will result in a series of “sampling scallops” left between sampled tool instances. It is evident from Fig. 10 that with a conservative value for the sampling interval length, the resulting sampling scallop size will be relatively small and not visible.

In order to demonstrate the applicability of FSV-rep-based simulation in a real industrial scenario, another case study was done to compute the in-process workpiece (IPW) geometry of an integrally bladed rotor (IBR). The case study was to obtain the IPW geometry after the machining operations to create one blade. Figure 11 shows the blank workpiece as the initial input and the IPW geometry as a triangle mesh generated from the updated FSV-rep model. The case study involved three milling operations using three ball end mills and 41,616 tool motion commands. The tool motions were mostly multi-axis. The

simulation execution time for both FSV-rep and tri-dexel-based IPW generation is listed in Table 1. It can be seen that the execution speed of FSV-rep is about 2.3 times faster than that of tri-dexels for this case. The portions of execution time spent for the three FSV-rep model update steps are also given in Table 1. The time split across the three update steps is consistent with the general trend observed for the basic case studies in Fig. 8. Also, it is worth noting that the improvement in execution speed for FSV-rep is mainly from the tilted tool orientation as depicted in Fig. 7 for the basic case I. The other factor due to bulk volume removal as depicted in Fig. 6 for the basic case I has less effect here. This is because the machining operations created more surface area per unit volume removed, thereby effectively having less bulk volume removed. Nevertheless, contributions from both factors give a combined faster performance.

As for model accuracy in terms of the sample points on the machined part surface, the FSV-rep-based IPW in the above case is very much comparable to tri-dexels with only 830 out of 196,794 dexel end points not matched with the FC points in the FSV-rep model. The discrepancy is higher than that observed in the basic cases. The reason is mainly due to the relatively large scallop areas produced by the ball end mill. The tip of the scallop may create a small hanging voxel frame segment (shorter than the edge length of the fine-level voxel in FSV-rep and not attached to any voxel corner point). These hanging frame segments are ignored in FSV-rep if no other portion of the edge frame of that particular FS voxel is active. This is treated as a computational compromise in the implementation of the FSV-rep model update. It leads to the small difference of only 0.42% in the complex industrial case. Ignoring such small hanging segments does not create much

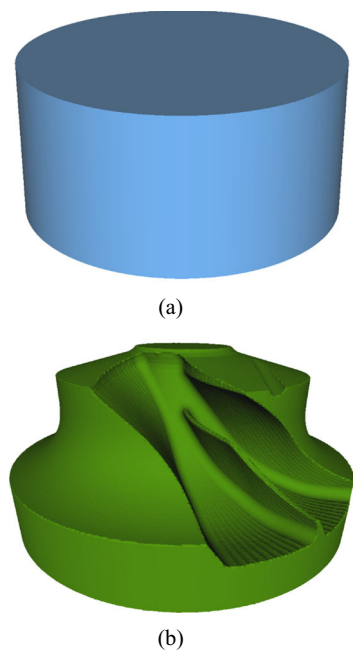


Fig. 11 Industrial case study. **a** Blank workpiece. **b** In-process workpiece of an IBR with one blade machined

Table 1 Execution time comparison for the industrial case study

Modeling method	Execution time (s)	
Tri-dexels	Total	28.198
FSV-rep	Coarse update	4.731
	Fine update	4.866
	Frame update	2.516
	Total	12.113

impact on the model geometry. Any sharp features lost can be easily restored via post-processing the generated triangle mesh.

7 Conclusions

The three-step FSV-rep model update process presented in this paper gives an efficient way to compute the machined part geometry with acceptable accuracy and memory usage. The developed FSV-rep method has demonstrated to be faster than the tri-dexel method due to two primary factors: (1) The bulk volume removal can be made faster with the coarse update of the FSV-rep model and (2) the tilted tool orientation has less effect on updating the FSV-rep model again thanks to the coarse update. Further, the FSV-rep method is able to carry out the majority of the voxel model update steps involving only simple point classifications and binary marking/unmarking operations. The computationally demanding intersection calculations are just used for the frame update step, thereby limiting the calculations to the final machined surface. The triangle mesh model for the machined part surface is a straightforward output from the FSV-rep model using the FS voxels.

As discussed in Sect. 6, two issues exist in the simulated machined part geometry based on the FSV-rep model. First, the triangle mesh obtained from the FSV-rep model has missing sharp machined features. Some existing triangle mesh post-processing algorithm will have to be implemented and performed to restore the sharp features. Second, sampling scallops are present on the triangle mesh surface due to the use of sampled tool instances to approximate the exact tool swept volume. To attain visually smooth surfaces, the tool path sampling interval has to remain small, but this hurts the overall computational efficiency. Further study is needed to optimize the tool path sampling interval according to the fine voxel grid spacing or even eliminate the need to sample the linear tool paths in three-axis milling. Such a study will directly contribute to even shorter computational time to generate the machined part geometry.

Acknowledgements This research has been funded by the Natural Sciences and Engineering Research Council of Canada (NSERC) under the CANRIMT Strategic Network Grant as well as the Discovery Grant. The authors wish to thank Dr. Jack Szu-Shen Chen in our research group for providing the initial source code for the ball end mill implementation as well as for the tool paths for the industrial case study.

Compliance with ethical standards The research does not involve human participants and/or animals. Consent to submit the paper for publication has been received explicitly from all co-authors.

Conflict of interest The authors declare that they have no conflict of interest.

References

- Altintas Y (2012) Manufacturing automation: metal cutting mechanics, machine tool vibrations, and CNC design, 2nd edn. Cambridge University Press, Cambridge
- Zhang Y, Xu X, Liu Y (2011) Numerical control machining simulation: a comprehensive survey. *Int J Comput Integr Manuf* 24:593–609
- Oliver JH, Goodman ED (1990) Direct dimensional NC verification. *Comput Aided Des* 22:3–9
- OuYang D, Feng HY, Van Nest BA, Buchal RO (2009) Effective gouge-free tool selection for free-form surface machining. *Comput-Aided Des Appl* 6:839–849
- Du J, Yan X, Tian X (2012) The avoidance of cutter gouging in five-axis machining with a fillet-end milling cutter. *Int J Adv Manuf Technol* 62:89–97
- Ding S, Mannan MA, Poo AN (2004) Oriented bounding box and octree based global interference detection in 5-axis machining of free-form surfaces. *Comput Aided Des* 36:1281–1294
- Ilushin O, Elber G, Halperin D, Wein R, Kim M (2005) Precise global collision detection in multi-axis NC-machining. *Comput Aided Des* 37:909–920
- Lee YS, Chang TC (1995) 2-phase approach to global tool interference avoidance in 5-axis machining. *Comput Aided Des* 27:715–729
- Erkorkmaz K, Altintas Y, Yeung CH (2006) Virtual computer numerical control system. *CIRP Ann Manuf Technol* 55:399–402
- Merdol SD, Altintas Y (2008) Virtual cutting and optimization of three-axis milling processes. *Int J Mach Tools Manuf* 48:1063–1071
- Yousefian O, Tarbuton JA (2015) Prediction of cutting force in 3-axis CNC milling machines based on voxelization framework for digital manufacturing. *Procedia Manuf* 1:512–521
- Ong SK, Yuan ML, Nee AYC (2008) Augmented reality applications in manufacturing: a survey. *Int J Prod Res* 46:2707–2742
- Mujber TS, Szecsi T, Hashmi MSJ (2004) Virtual reality applications in manufacturing process simulation. *J Mater Process Technol* 155–156:1834–1838
- Joy J, Feng HY (2016) Frame-sliced voxel representation: an accurate and memory-efficient modeling method for workpiece geometry in machining simulation. *Comput Aided Des* (accepted)
- Spence AD, Abrari F, Elbestawi MA (2000) Integrated solid modeller based solutions for machining. *Comput Aided Des* 32:553–568
- El-Mounayri H, Elbestawi MA, Spence AD, Bedi S (1997) General geometric modelling approach for machining process simulation. *Int J Adv Manuf Technol* 13:237–247
- Aras E, Yip-Hoi D (2008) Geometric modeling of cutter/workpiece engagements in three-axis milling using polyhedral representations. *ASME J Comput Inf Sci Eng* 8:031007
- Gong X, Feng HY (2016) Cutter-workpiece engagement determination for general milling using triangle mesh modeling. *J Comput Des Eng* 3:151–160
- Roy U, Xu Y (1999) Computation of a geometric model of a machined part from its NC machining programs. *Comput Aided Des* 31:401–411
- Fussell BK, Jerard RB, Hemmett JG (2003) Modeling of cutting geometry and forces for 5-axis sculptured surface machining. *Comput Aided Des* 35:333–346
- Aras E, Feng HY (2011) Vector model-based workpiece update in multi-axis milling by moving surface of revolution. *Int J Adv Manuf Technol* 52:913–927
- Stifter S (1995) Simulation of NC machining based on the dexel model: a critical analysis. *Int J Adv Manuf Technol* 10:149–157

23. Benouamer MO, Michelucci D (1997) Bridging the gap between CSG and Brep via a triple ray representation. In: Proceedings of the fourth ACM Symposium on Solid Modeling and Applications, pp 68–79
24. Lee SW, Nestler A (2012) Virtual workpiece: workpiece representation for material removal process. *Int J Adv Manuf Technol* 58: 443–463
25. Jang D, Kim K, Jung J (2000) Voxel-based virtual multi-axis machining. *Int J Adv Manuf Technol* 16:709–713
26. Wou SJ, Shin YC, El-Mounayri H (2013) Ball end milling mechanistic model based on a voxel-based geometric representation and a ray casting technique. *J Manuf Process* 15:338–347
27. Karunakaran KP, Shringi R, Ramamurthi D, Hariharan C (2010) Octree-based NC simulation system for optimization of feed rate in milling using instantaneous force model. *Int J Adv Manuf Technol* 46:465–490
28. Sullivan A, Erdim H, Perry RN, Frisken SF (2012) High accuracy NC milling simulation using composite adaptively sampled distance fields. *Comput Aided Des* 44:522–536
29. Museth K (2013) VDB: high-resolution sparse volumes with dynamic topology. *ACM Trans Graph* 32(3):27
30. Laine S, Karras T (2011) Efficient sparse voxel octrees. *IEEE Trans Vis Comput Graph* 17:1048–1059
31. Ferry W, Yip-Hoi D (2008) Cutter-workpiece engagement calculations by parallel slicing for five-axis flank milling of jet engine impellers. *ASME J Manuf Sci Eng* 130:051011
32. Weinert K, Du S, Damm P, Stautner M (2004) Swept volume generation for the simulation of machining processes. *Int J Mach Tools Manuf* 44:617–628
33. Yang Y, Zhang WH, Wan M, Ma YC (2013) A solid trimming method to extract cutter-workpiece engagement maps for multi-axis milling. *Int J Adv Manuf Technol* 68:2801–2813
34. Osher S, Fedkiw R (2003) Signed distance functions. In: *Level Set Methods and Dynamic Implicit Surfaces*, pp 17–22
35. Langeron JM, Duc E, Lartigue C, Bourdet P (2004) A new format for 5-axis tool path computation, using Bspline curves. *Comput Aided Des* 36:1219–1229
36. Ren Y, Zhu W, Lee YS (2008) Feature conservation and conversion of tri-dexel volumetric models to polyhedral surface models for product prototyping. *Comput-Aided Des Appl* 5:932–941