Safe and Short Tool Length Generation for 3+2 Axis NC Machining of a Ball-End Cutter Using Graphics Hardware

Qingzhen Bi, Han Ding, and Yuhan Wang

State Key Lab. of Mechanical System and Vibration, School of Mechanical Engineering,
Shanghai Jiaotong University,
1954 Hua Shan Road, Shanghai 200030, P.R. China
{biqz,hding,yhwang}@sjtu.edu.cn

Abstract. In 3+2 axis NC machining, only three linear axes are executed while 5-axis machine's two rotational axes are locked. Compared with conventional 3-axis machining, the main advantage of 3+2 machining is that it allows use of a shorter cutter. The magnitude of tool deflection and the stability of cutting process strongly depend on the slenderness of the tool. However, the tool overhang length can't be easily shortened because of collision avoidance. In this paper the accessibility cone of a milling cutter is newly defined to generate collision-free cutting orientations. The accessibility cone and safe tool length can be efficiently computed by a novel method using graphics hardware. The proper cutting orientation of 3+2 machining is then obtained by optimizing the safe and shortest tool length. The computational example and cutting experiment of sculptured surface machining confirm the validity of the approach.

Keywords: 3+2 machining; pTool length; Cutting orientation; Access-based machining.

1 Introduction

3+2 axis NC machining is a technique whereby only three linear axes are executed with the cutting tool locked in a titled position using the 5-axis machine's two rotational axes, hence the name, 3+2 axis machining. The fixed cutting tool orientation distinguishes 3+2 axis machining from continuous or simultaneous 5-axis machining. Generally locking the rotary axes generates fewer fluctuations in the feed rate than simultaneous 5-axis machining and results in a more consistent surface finish with lower variations in the cutting force and torque. The other main advantage of 3+2 axis machining is that it allows use of a shorter, more rigid cutter that would not be permissible with conventional 3-axis machining. Since the magnitude of tool deflection and the stability of cutting process strongly depend on the slenderness of the tool, short cutters can improve the rigid of the whole cutting process and significantly reducing deflection and vibration [1, 2]. As a result, higher cutting speed can be achieved without putting excessive load on the cutter and, so increasing tool life and reducing breakages.

C. Xiong et al. (Eds.): ICIRA 2008, Part II, LNAI 5315, pp. 348–355, 2008.

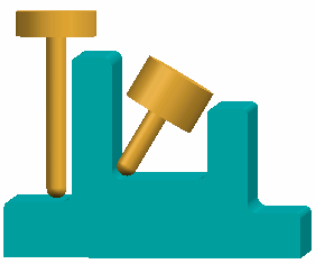


Fig. 1. The different safe tool lengths with different tool orientations

However the tool overhang length can't be easily shortened because the tool holder will collide with the workpiece. Collision avoidance becomes the most important consideration to use a short cutting tool. Though some methods [3-5] are helpful to get collision-free tool path, the shorter tool length has not been considered. The collision-free short cutters of multi-axis NC machining are usually considered in machining simulation process. The minimal cutter extension can be calculated by the simulation software such as Vericut [6]. Kim [7] proposed a method to calculate short and safe cutter length by safe space in NC machining. The problem of these methods is that the safe tool length can only be determined based on a predefined tool path. However the safe and shortest tool length (SSTL) of a tool path is usually determined by the tool orientations as shown in Fig. 1. Therefore the tool orientation is the most important to use a short cutter for 3+2 axis machining.

Recently commodity graphics hard has been enormous improvements in terms of programmability and computational speed [8]. Morimoto [9] proposed a GPU based algorithm to optimize cutting direction and the algorithm conservatively considers collision avoidance. Expired by the existing works, a new GPU-based approach is proposed to efficiently generate safe and short cutter length when tool paths are planned. The SSTL for each accessible orientation is first computed at each CC points. The collision-free tool orientations are then optimized by considering cutter length.

The remainder of the paper is organized as follows. In Section 2, a GPU-based algorithm is proposed to compute accessible cones for CL points of a cutting tool. Section 3 describes a method to compute the SSTL at each CL point for each accessible orientation. Section 4 presents the whole process to obtain optimal tool orientation and SSTL. A computational example is given in Section 5 and Section 6 verifies the example by a cutting experiment. Section 7 draws some conclusions.

2 GPU-Based Accessibility Cone Computation

Since collision avoidance can be viewed approximately as access assurance of the milling cutter at each CL point, collision avoidance at discrete CL points is first considered.

2.1 Basic Concept of Accessibility Cone

Usually a milling tool and tool holder can be regarded as a cylinder with fixed radii respectively to detect global collision as shown in Fig. 2. The CL point of a ball-end cutter is the center of the end half sphere. The tool orientation is represented as $\mathbf{v}_c$ and the cutter length is expressed as l_c. A disk in spatial coordinates can be represented as $D(\mathbf{p},\mathbf{v},r)$, where the three parameters are the center, the direction perpendicular to the disk plane and the radius respectively.

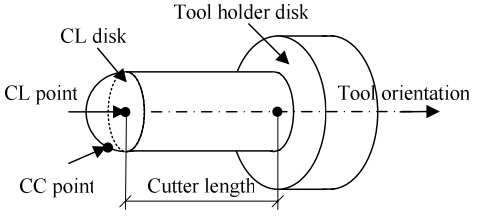


Fig. 2. Model of a ball-end cutter

Definition 1. (The CL Disk of the Milling Cutter Along an Orientation). As shown in Fig. 2, let $\mathbf{p}_c \in \mathbf{R}^3$ be the CL point of a milling tool. The CL disk $CLD(\mathbf{p}_c,\mathbf{v}_c)$ of the milling tool is defined as

$$CLD(\mathbf{p}_c, \mathbf{v}_c) = D(\mathbf{p}_c, \mathbf{v}_c, r_c) \tag{1}$$

where r_c is the radius of the milling tool.

Definition 2. (The Tool Holder Disk of the Milling Cutter Along an Orientation). As shown in Fig. 2. The tool holder disk $THD(\mathbf{p}_c,\mathbf{v}_c)$ is defined as

$$THD(\mathbf{p}_c,\mathbf{v}_c) = D(\mathbf{p}_t, \mathbf{v}_c, r_t) \tag{2}$$

where $\mathbf{p}_t = \mathbf{p}_c + \mathbf{v}_c \bullet l_c$ and r_t is the radius of the tool holder.

Definition 3. (The Complete Visibility Cone of a CL Disk). The view direction $\mathbf{v}_v$ is set as the opposite direction of the tool orientation. A CL disk $CLD(\mathbf{p}_c, \mathbf{v}_c)$ in a set of obstacles S is completely visible in a viewing orientation $\mathbf{v}_v$ if all points of the CL disk are visible when the observer locates outside the bounding sphere of S and $CLD(\mathbf{p}_c, \mathbf{v}_c)$. Complete visibility cone of $CLD(\mathbf{p}_c, \mathbf{v}_c)$, $CVC(CLD(\mathbf{p}_c, \mathbf{v}_c), S)$ is defined as all tool orientations from which $CLD(\mathbf{p}_c, \mathbf{v}_c)$ is completely visible.

The complete visibility cone of a tool holder disk is analogously defined as $CVC(THD(\mathbf{p}_c, \mathbf{v}_c), S)$. Based on the concepts of the CL disk and tool holder disk, the accessibility detection of the milling cutter is conducted to justify the complete visibility of $CLD(\mathbf{p}_c, \mathbf{v}_c)$ and $THD(\mathbf{p}_c, \mathbf{v}_c)$.

Definition 4. (Accessibility Cone at a CL Point). At a CL point, a milling tool is accessible along an orientation $\mathbf{v}_c$ if both the CL disk and the tool holder disk are completely visible. The accessibility cone of the CL point for the milling tool is defined as

$$AC(\mathbf{p}_c, S) = \{\mathbf{d} | \mathbf{d} \in CVC(CLD(\mathbf{p}_c, \mathbf{v}_c), S) \cap CVC(THD(\mathbf{p}_c, \mathbf{v}_c), S)\} \tag{3}$$

2.2 Accessibility Detection

Definition 4 reduces the accessibility detection to problems of check the complete visibility of disks. Taking advantage of robust and efficient performance of a graphics card to implement occlusion query, the complete visibility of the CLD and the THD in each candidate orientation is detected based on the depth representation of obstacles.

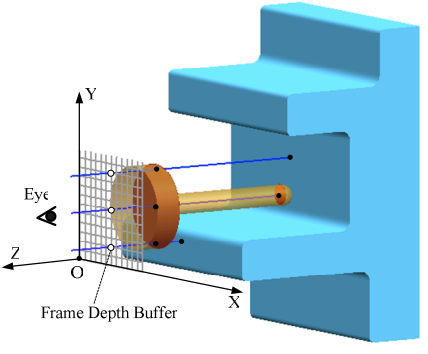


Fig. 3. Complete visibility detection of a CL disk

2.2.1 GPU-Based Visibility Detection

GPU-based algorithm means that the visibility detection of the disks is implemented by the graphics card. The detailed explain about the graphics card can be referred to Ref. [10]. Fig. 3 shows an eye position and screen coordinate system to detect the complete visibility of the CL disk. The orthographic projection mode of OpenGL is used to create the rectangle viewing volume which encloses the disks. Graphical calculations first convert obstacle objects to a set of pixels into the frame buffer where the surface appears. The depth values associated with pixels describe the minimal relative distances between the screen and the objects. Therefore the complete visibility of the CL disk can be detected by depth representation of the CL disk.

2.2.2 Accessibility Detection

Definition 4 shows the accessibility of a cutter is determined by the complete visibility of both CLD and THD of the cutter. With the help of collision query function of the GPU, the accessibility of the cutter in an orientation can be detected by follows.

Algorithm 1. GPU-based accessibility detection for a cutting tool

Step 1. Enable back face culling and standard depth test

Step 2. Set view matrix parameters as described in Section 2.2.1

Step 3. Render obstacle geometry

Step 4. Set the depth function to *GL_GEQUAL* and disable depth writes.

Step 5. Render the CLD. Call an occlusion query to test the number n_{CLD} of rendered pixels. The CL point is not accessible if n_{CLD} is not equal to zero.

Step 6. Render the THD. Call an occlusion query to test the number n_{THD} of rendered pixels. The CL point is not accessible if n_{THD} is not equal to zero. Otherwise, the cutter is accessible to the CL point in the orientation.

2.2.3 Accessibility Cone Computation

The accessibility cone at a CC point represents the set of directions along which the cutting tool is collision free. Theoretically there are infinite directions in the accessibility cone. In practice only some sampled directions in the Gauss Sphere are used to decrease the compute cost. These sampled discrete orientations are regarded as candidate orientations of the accessibility cone.

The algorithm like Ref. [5] is employed to tessellate the Gauss Sphere. The vertexes of the resulted triangle mesh represent a fairly homogenous sampling of the Gauss Sphere and the orientations represented by the vertexes can be regarded as the candidate orientations. The accessibility of a CC point along each candidate orientation is determined by Algorithm 1. The accessibility cone of a CC point will then be obtained by collecting all accessible orientations of the CC point.

3 GPU-Based SSTL Computation

The idea is to find the minimum tool length without collision between tool holder and the part surface in each accessible direction. The tool length is initially set as larger value l_{int} by experience to compute accessibility cones for all CC points. The SSTL of each CC point is computed only for each accessible orientation.

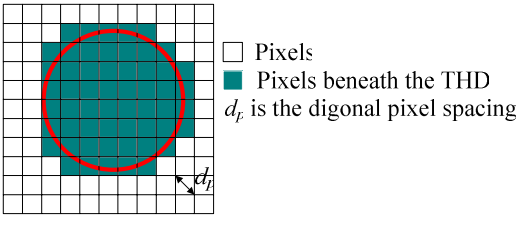


Fig. 4. Pixels beneath the THD

When the obstacles are rendered into a frame buffer, each pixel is associated with a depth value. A depth test guarantees that pixels with larger depth-buffer values are overwritten by pixels with smaller values. Therefore the SSTL can be determined by the pixels of obstacle object beneath the THD. For machining security of tool path, the THD are redefined as

$$THD(\mathbf{p}_c, \mathbf{v}_c) = D(\mathbf{p}_t, \mathbf{v}_c, r_t+d_p/2) \tag{4}$$

where d_p is the diagonal pixel spacing for a resolution to render the scene as shown in Fig. 4 and r_t is the tool holder radius. The depth values of pixels representing the THD are all equal because the THD is perpendicular to the view direction as shown in Fig. 3. The SSTL in the accessible direction is calculated by

$$l_{shortest} = l_{init}-(d_t-min(d_i, i=1\ldots n_{pixel})) \tag{5}$$

where $l_{shortest}$ is the safe and shortest tool length,

d_t is the depth value of the THD in screen coordinate system,

d_i are the depth values of obstacle pixels beneath the THD,

n_{pixel} is the number of pixels beneath the THD.

4 Tool Orientation and SSTL Optimization

The whole process to optimize the tool orientation and SSTL is described based on Section 2 and Section 3. A series of CL points calculated based on cutter center point representing the cutting trajectory. The computation of SSTL and tool orientation must be applied to each CC point. The GPU-based algorithm to generate SSTL and tool orientation for all CL points is described as follows.

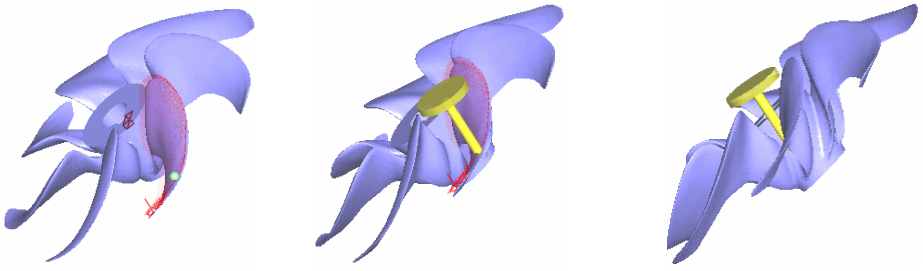
Algorithm 2. GPU-based SSTL and tool orientation generation

```
Input: CO_i: Candidate orientations and i=1,…,n_co,
       CL_j: Successive CL points and j=1,…,n_cl,
       S: Obstacles.
Output: l_sstl: SSTL•TO_sstl: Optimized tool orientation.
l_sstl=l_init.  /*Initial tool length*/
FOR (Each CO_i with i=1,…,n_co)
{ Render S
  FOR (Each CL_j with j=1,…,n_cl)
  { Detect accessibility of CL_j for CO_i by Algorithm 1
    IF (CL_j is not accessibility)  CO_i is invalid.
  }/*End of FOR_CL_j */
  IF (CO_i is valid)
  { FOR (Each CL_j with j=1,…,n_cl)
     { Calculate SSTL l_sstl_ij. /* the algorithm in Section
3*/
     }/*End of FOR_CL_j */
     l_sstl_i = max(l_sstl_ij,j=1…n_cl);
  }/*End IF*/
}/*End of FOR_CO_i */
Find the index k with minimal value in l_sstl_i (i=1,…,n_co);
l_sstl= l_sstl_k; TO_sstl = CO_k;
END.
```

5 Computational Examples

The proposed algorithms have been implemented in a prototype system using Visual C++ programming language and OpenGL library. The input to the system is polygonal mesh obstacle model, a group of CL points of a ball end cutter and the cutter model. The system output the tool orientation and SSTL for the CL points. As shown in Fig. 5. The part model consists of 291,506 triangles and represents the shape of a propeller. The maximum radius of the propeller is about 100mm. The example is performed using the graphics card, NVIDIA Quadro NVS 140M. The pixel resolution of the frame buffer is set 1000×1000.

A R3 ball-end cutter is employed for finishing milling. The tool length is initialized to 55mm and the diameter of tool holder is 36mm. The approximation tolerance of the driven surface is set 0.01 mm and the maximal step over distance is set 0.15 mm. 22,122 CL points are obtained as shown in Fig. 5. After 1,026 candidate orientations are achieved, it takes 337.687 seconds to compute the accessibility cone and optimize tool orientation. Six accessible orientations are generated and the accessibility cone is



(a) (b) (c)

Fig. 5. (a) Accessibility cone, (b) Optimized tool orientation, (c) Optimized SSTL

shown in Fig. 5 (a). The SSTL of the tool path is 37.27 mm and the optimized cutting tool is shown in Fig. 5 (b) and Fig 5 (c).

6 Cutting Experiment

With the collision-free tool orientation and the SSTL generated by the proposed algorithm, the cutting experiment is performed in a five-axis machine tool with a dual-rotary table structure. The tool length in the actual machining process is 40mm because some remained materials still exist in the neighboring leafs. Fig.6. shows the cutting process and the shape after being cut. No collision takes place in the machining process.

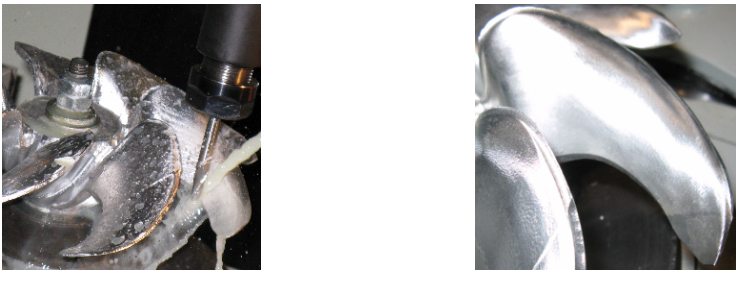


Fig. 6. Cutting process and the shape after being cut

7 Conclusions and Future Work

The SSTL of tool path for 3+2 axis machining of sculpture parts is considered in tool path generation. Based on the newly defined accessibility of a ball-end cutter, a fast and robust algorithm is generated to compute accessible orientations and to optimize SSTL using graphics hardware. The computational example and cutting experiment of sculptured surface machining confirm the validity of the proposed algorithm.The SSTL generation algorithm for a simultaneous 5-axis machining of sculpture parts will be further studied. The tool orientation of the 5-axis machining can be changed more flexibly, which allows use of a shorter cutter compared with 3+2 axis machining.

Acknowledgements

This work is supported by the National Basic Research Program of China under Grant 2005CB724103.

References

1. Salgado, M.A., De Lacalle, L.N.L., Lamikiz, A., Munoa, J., Sanchez, J.A.: Evaluation of the stiffness chain on the deflection of end-mills under cutting forces. International Journal of Machine Tools and Manufacturing 45, 727–739 (2005)
2. De Lacalle, L.N.L., Lamikiz, A., Sanchez, J.A., Salgado, M.A.: Effects of tool deflection in the high-speed milling of inclined surfaces. International Journal of Advanced Manufacturing Technology 24, 621–631 (2004)
3. Ilushin, O., Elber, G., Halperin, D., Wein, R., Kim, M.S.: Precise global collision detection in multi-axis NC-machining. Computer-Aided Design 37, 909–920 (2005)
4. Yin, Z.P., Ding, H., Xiong, Y.L.: Visibility theory and algorithms with application to manufacturing processes. International Journal of Production Research 38, 2891–2909 (2000)
5. Balasubramaniam, M., Laxmiprasad, P., Sarma, S., Shaikh, Z.: Generating 5-axis NC roughing paths directly from a tessellated representation. Computer-Aided Design 32, 261–277 (2000)
6. CGTech Company, `http://www.cgtech.com`
7. Kim, S.J.: Short and safe tool setting by safe space in NC machining. International Journal of Advanced Manufacturing Technology 33, 1017–1023 (2007)
8. Owens, J.D., Luebke, D., Govindaraju, N., Harris, M., Krüger, J., Lefohn, A.E., Purcell, T.J.: A Survey of General-Purpose Computation on Graphics Hardware. Computer Graphics Forum 26, 80–113 (2007)
9. Morimoto, K., Inui, M.: A GPU based algorithm for determining the optimal cutting direction in deep mold machining. In: Proceeding of the 2007 IEEE international symposium on assembly and manufacturing, pp. 203–208. IEEE press, Michigan (2007)
10. Shreiner, D., Woo, M., Neider, J., Davis, T.: OpenGL Programming Guide: The Official Guide to Learning OpenGL. Addison-Wesley Professional, Reading (2005)