Reliable Haptic System for Collision-free 5-axis Tool Path Generation

Kun Chen and Kai Tang

The Hong Kong University of Science and Technology, mekchen@ust.hk, mektang@ust.hk

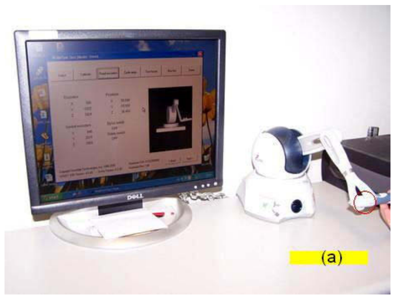
ABSTRACT

This paper presents a collision-free 5-axis tool path generation system that uses a haptic device to perform reliable man-machine interactions. The system is based on commercial haptic devices which can provide 3 degrees of freedom (DOF) force feedback and 6 DOF posture sensing. The system achieves three main functions: (1) a rendering conversion that uses 3 DOF force feedback haptic representation instead of the 5 DOF real world requirements; (2) an efficient force feedback design that helps get accurate results directly from the user's manipulation; and (3) a reduction of user's operation workload by providing heuristic leading forces. The system faithfully represents the real milling machining procedure both graphically and haptically. No matter an expert or not, the user is able to use the presented interactive system to conveniently plan and design collision-free 5-axis tool paths.

Keywords: 5-axis tool path generation; Collision detection; Haptic device; Haptic rendering design

1. INTRODUCTION

Nowadays 5-axis NC machining is widely used for machining sculptured surfaces such as aircraft parts, turbine blades, impellers, propellers, jewelries, molds and dies, etc. 5-axis machining enjoys tremendous advantages over the traditional 3-axis machining in terms of better tool accessibility, faster material removal rate, much improved surface finishing quality, and others. However in 5-axis machining, automatically generating collision-free tool paths still remains a major challenge. Traditional methods in automatic 5-axis tool path generation require huge computing time for calculating the tool orientation to avoid gouging and global interferences. Recently, haptic systems began to attract people's interests in varied aspects [11], [18-19], [22-23]. However, only very few of them (e.g., [11, 18]) apply the haptic system to 5-axis tool path generation. The MIT-Suzuki haptic system [11] brought us an intuitive man-machine interface for generating collision-free 5-axis tool path. However it is a specialized haptic system and the captured posture data is unreliable during the interacting procedure.



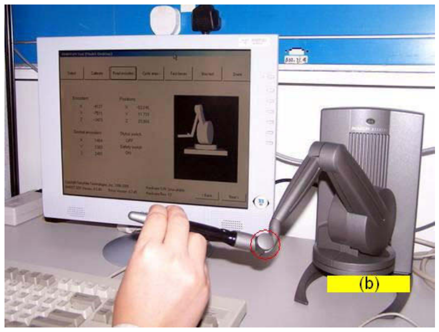


Fig. 1. SensAble PHANTOM® OMNI™ (a) haptic device and SensAble PHANTOM® DESKTOP™ (b) haptic device.

In this paper, we describe a collision-free 5-axis tool path generation system that uses commercial haptic devices to perform reliable man-machine interactions. The SensAble Technologies, Inc. [5] has delivered over 3500 haptic devices worldwide. Many systems [19], [22-23] are built on its open-end developing platform. In the proposed system, we support both SensAble PHANTOM® OMNI™ (Fig. 1a) and PHANTOM® DESKTOP™ (Fig. 1b) haptic devices.

They both can provide 3 DOF force feedback and 6 DOF posture sensing. The main differences between them are device port interface, precision and capacity. And both devices can provide fidelity enough force feedback and position sensing in the interactive manipulations. In the man-machine interactivities of tool path generation, the captured posture data always suffers from the accuracy problem. This is due to the inaccurate collision detection scheme, low density of detected data, and the human body physical limitation. This paper reports our experiments on how to settle these problems and make the recorded posture data reliable. A semi-automatic collision-free 5-axis tool path generation system is presented. It is user-friendly, reasonably accurate, fast, and easy to use. The user would benefit in two aspects: the designed leading force will reduce his workload of the manipulations; the reliable recorded data would reduce the post-processing time. The rest of the paper is organized as follows. We first review some related work in Section 2. Then the system architecture is presented in Section 3. Section 4 describes collision detection scheme of the system. And Section 5 describes the designing of haptic force rendering. The semi-automatic system comes out in this section. Finally we show some results and give the summary.

2. PREVIOUS WORK

Tool positioning problem has been studied with great interest in the last 15 years. It can be treated as a sweep procedure that covers the entire machined surface. The Isoparametric method, the cutter contact (CC) Cartesian method, the cutter location (CL) Cartesian method, and the automatically programmed tool (APT) type method are the four most common CC based tool path generation methods used in CAD/CAM [1]. Other methods such as the configuration space (C-space) approach, the “practical methods” for construction CL Z-maps [1], the non-isoparametric method [20], and the swept envelope approach [7] were also developed in recent years. Some researchers [4], [9], [14-17] also studied tool path generating problem from optimization perspective. Many commercial CAD/CAM software packages such as MasterCAM, AlphaCAM, CAMAX and SmartCAM can generate tool paths automatically and quickly. But all these only settle three DOF if in 5-axis machining. The other two DOF which define the tool orientation are always simply and conveniently assigned to be the surface normal. To prevent gouging and global interference, the user of these systems always has to modify the tool orientation based on simulation.

Though 5-axis milling machining offers much better machining efficiency and tool accessibility over 3-axis machining, it poses serious gouging and global interference problems. Many methods were proposed for the determination of collision-free tool orientations. The two-phase method [21] utilizes convex hulls to find quickly the feasible set of tool orientations and calculate the correction vector to avoid global interference. And in [2], collision detection is integrated into the tool path generation stage. Cylindrical approximation of the tool is assumed in those methods. A direct generation method of 5-axis roughing tool path was presented in [10] in which computer graphics techniques were applied. Recently, the floor-wall and ceiling (FWC) method [8] was introduced for determining the tool axes. An efficient tool path generation algorithm was proposed in [12] that has no restrictions on the shape of the tool. Some researchers also incorporated collision avoidance with machining optimization [3]. To generate collision-free tool path quickly and intuitively, haptic interfaces were also proposed in [11, 18]. They utilize 5-DOF joystick-like haptic devices which were proprietary and developed by the authors.

Today open haptic devices provided by SensAble are widely used in industry. Haptic devices can help user move and “feel” the touched objects in a virtual reality (VR) environment. Many researchers have worked on that. The Virtual DesignWorks system [19] uses PHANTOM® haptic device in design of 3D CAD models. The haptic virtual coordinate measuring machine (CMM) [22] greatly helps CMM measurement operations to be automatic and efficient. The virtual reality technology provided by haptic devices can also be expanded to reverse engineering, shape modeling, and mechanical analysis [23]. Haptic devices help build a user-friendly interface between man and machine which is intuitive and interactive in operation. Our work was inspired by the increasing popularity of open haptic devices in CAD, especially the commercial PHANTOM® haptic device, and aimed at designing a practical haptic computer system for 5-axis tool path generation.

3. HAPTIC CAM SYSTEM DESIGN

Existing research results provide many useful fundamental theories and algorithms in 5-axis tool path generation and haptic interface. Our 5-axis tool path generation system integrates many of these well-developed algorithms and techniques with the popular PHANTOM® haptic device. There are many choices in every sub scheme. Each choice is good and suitable only in its certain requirement. It’s very important to choose a proper one which fits our particular haptic VR environment for 5-axis tool path generation. First of all, we should describe how a PHANTOM® haptic device normally works.

As an intact system, a haptic device should faithfully send the position and orientation data for virtual display rendering. At the same time it should promptly receive the haptic events and perform real physical feeling. Including these two

critical requirements, it's very important that what the haptic device acts like should represent the real world operation correspondingly. As shown in Fig. 1, the SensAble PHANTOM® haptic device can provide fidelity position sensing in 6 DOF: x, y, z, pitch, roll and yaw. The pen grasped in the hand can be treated as the tool. The proxy point (marked in a red circle) can represent the cutter's tip and the pen's orientation can represent the cutter's orientation accordingly. When we move and/or rotate the pen, all these values update in real time. Since it is a 6 DOF sensing system and our application is 5-axis, we can ignore the yaw value here. By getting the 6 DOF posture values in high frequency, the tool can be rendered synchronously in the display screen. We use OpenGL to display the tool, the machined surface, and the working environment in 3D space. After settling the ocular problem in virtual reality, the real time force feedback must be performed to construct a comprehensive system. The SensAble PHANTOM® device used in our system can perform realistic 3D touch sensation by force feedback in only translational DOF: x, y and z. Here comes a problem: in the 5-axis machining application, how can the two rotational DOF be accounted for? Of course a straightforward resolution is using a much more expensive device with 6 DOF force feedback (both translational and rotational) which are also available from SensAble. But since the PHANTOM® DESKTOP™ and PHANTOM® OMNI™ systems are currently widely distributed and used in various daily life applications, a system based on them will be economical and hence desirable. We resolve the 3-DOF force feedback limitation by a conversion and we will show in the later sections why it is feasible and what we can benefit from it.
To describe the architecture of our system, we need to discuss the SDK package chosen by us. The GHOST® SDK has a higher level of abstraction for creating haptic environments than the OPENHAPTICS™ TOOLKIT. However, it does not support the PHANTOM® OMNI™ device. And its efficiency is not as good as OPENHAPTICS™ TOOLKIT. The GHOST® SDK is weak in thread-safe state synchronization between haptic thread and graphic thread. That would cause problem for multi-threaded programming. As an interactive system, real time is critical. Though GHOST® SDK is easy for implementation, we choose OPENHAPTICS™ TOOLKIT for its flexibility and compatibility. Under OPENHAPTICS™ TOOLKIT the system can execute with both devices without any modifications of the source codes.

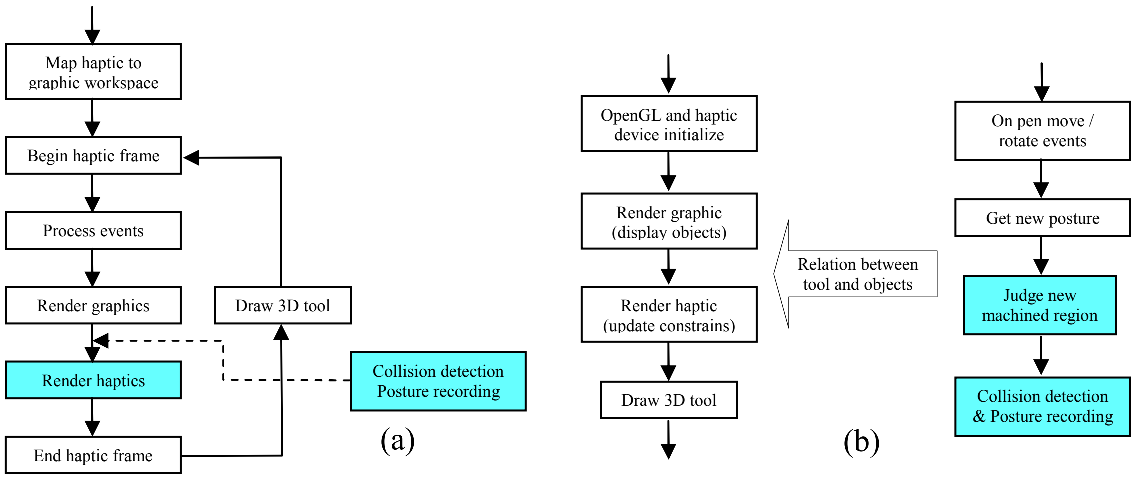

Map haptic to graphic workspace
Begin haptic frame
Process events
Render graphics
Draw 3D tool
Render haptics
Collision detection Posture recording
End haptic frame
(a)
OpenGL and haptic device initialize
Render graphic (display objects)
Relation between tool and objects
Render haptic (update constrains)
Draw 3D tool
On pen move / rotate events
Get new posture
Judge new machined region
Collision detection & Posture recording
(b)

Fig. 2. Flow chart: simple single thread system (a) and multi thread in our system (b).

A general system under OPENHAPTICS™ TOOLKIT would have the flow chart in Fig. 2(a). To deal with simple 3D objects, the scheme is very easy for implementation. In the simple scheme, refresh rate in graphic and haptic will both be reduced due to the cross-influence of themselves. We can not ignore this because in 5-axis machining the objects are usually complex and would normally have huge triangle meshes. A large rendering time step means that the interaction will not be real time. As a resolution, we instead use the multi thread scheme as shown in Fig. 2(b). In the figure, "Judge new machined region" is for marking purpose. This function has similar scheme with collision detection and the only difference between them is the judging strategy. So in the discussion of the scheme, we only mention the collision detection. Another important issue is the refresh rate of haptic rendering. Typically a graphics application needs to refresh the contents of the frame buffer more than 30 times a second in order to give the human eye the impression of continuous motion on the screen. Due to the psycho-physics of human perception, a haptic application needs to refresh the forces rendering more than 1000 times a second in order to give the kinesthetic sense of stiff contact. This again requires the system to use the scheme shown in Fig. 2(b). It also determines the collision detection scheme which we will discuss in the next section. Another benefit from the multi-thread programming in our system is the system extension. Since the haptic system prefers high refresh rate, parallel computing is a proper direction for reducing the computing time, and the multi-thread frame is preparative for this future extension.

4. COLLISION DETECTION SCHEME

As can be seen in Fig 2(b), the collision detection method in 5-axis tool path application heavily influences the refresh speed of haptic rendering. The collision detection algorithm is required to be as fast as possible. On the other hand, to prevent unreliable results, the collision detection accuracy should not be comprised. These two requests conflict each other and we need to balance them. Before that, we first execute an input data pre-processing module that will help save the computing time.

The part and other objects (such as fixtures) input to our system are given as tessellated surfaces in triangle mesh format. Most common 3D CAD data exchange files such as STL, VMRL, DXF and OBJ are supported by our system. Due to the file data structure, the representations of a triangle face are different. To those file formats in which the xyz coordinates of the three nodes of a triangle are explicitly output one by one, we will run a function called SamePointsRemoving. We achieve this by maintaining a sorted node list. When reading a new triangle face, we check whether the node has the same one in the list. We only insert the new node to the sorted list and only record three node indexes to the triangle. To greatly reduce the memory occupation and also the calculating time, this step is necessary. As an experimental example, a designed toy mesh has 25,469 triangle faces and only 13,430 unique nodes. A human body mesh data has 39,570 triangle faces and only 19,930 unique nodes. Without SamePointsRemoving, it is obvious that the nodes number will be three times the faces number. Now we get a clean node list and each of the triangle data only stores its vertex indexes. The next step then is mesh resampling.

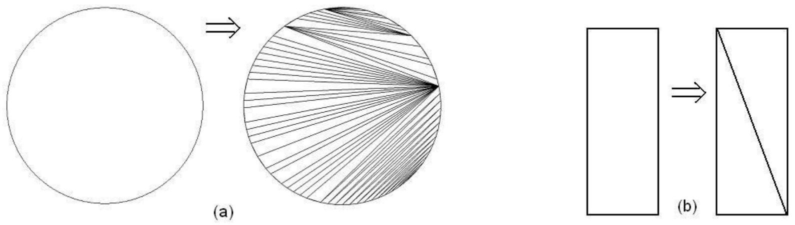


Fig. 3. Feature model to mesh model, represented by minimal triangle numbers.

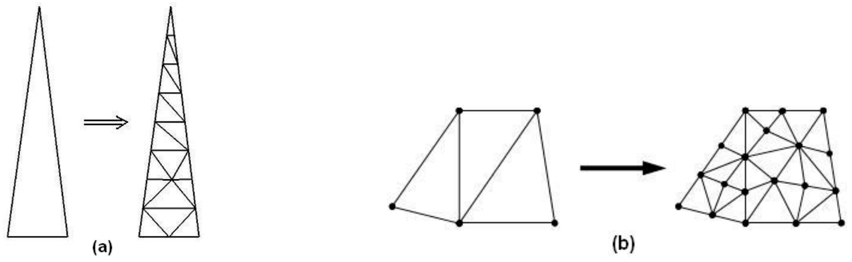


Fig. 4. Mesh resampling.

The input tessellated surfaces are generated from varied sources. Scanned point cloud usually makes a "good" triangle mesh that all triangles have similar size and shape. However, in any mesh obtained from a 3D CAD software, "bad" triangles must be expected. Shown in Fig. 3(a) is the mesh of a disk face output from a commercial CAD modeler, in which the disk is divided into a number of long and skinny triangles. A simple geometry such as a rectangle can be easily represented by two long and thin triangles without any deformation or approximation (Fig. 3(b)). These long and thin triangles are unsuitable in marking, collision detecting and orientation recording. We need to resample [13] those "bad" triangles to "good" ones, as illustrated in Fig. 4(a). After handling these long and thin triangles, we still need another kind of resampling [13] – increasing the density of the mesh. We calculate a threshold which is relative to the machine tool radius. We use this value to call the MakingDense function and the procedure can be illustrated in Fig.

4(b). After all the resampling processes, we call the SamePointsRemoving function again. Next we'll discuss the collision detection scheme and explain why it is needed to run the MakingDense function.

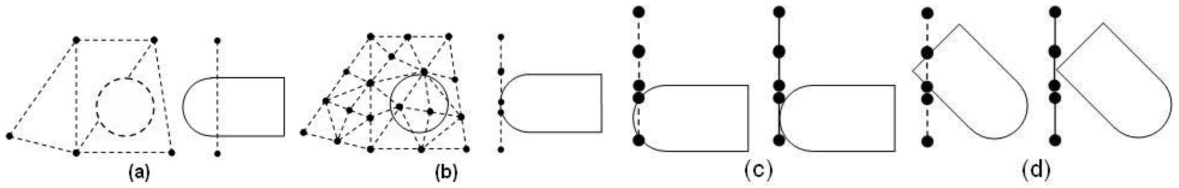


Fig. 5. Tool penetrates through the point cloud (a); denser points present tool's full penetration (b); errors due to partial penetration by the tool (c) and (d).

In collision detection, representation of the objects is important. It determines the algorithm, computing time and accuracy. We use triangle mesh in graphic rendering, so it is straightforward and realistic using triangles to detect collision. But calculating intersections with the triangles is much more complex than with a point cloud. As we have mentioned before that a haptic program needs high refresh frequency, collision detection must be completed in a tiny time step. Point cloud is suitable for this strict requirement. Since we have already done the redundant points removal, the number of points is only about half of that of the triangles. When using the point clouds scheme, some issues must be dealt with, as shown in Fig. 5. The density of the point cloud should be high enough so that the tool's penetration becomes impossible (Fig. 5(a) vs. Fig. 5(b)). And although dense enough, a point cloud might still allow partial penetration by the tool (Fig. 5(c)(d)). We can solve this problem by enlarging the tool body size a little bit to ensure that the tool body does not touch the part surface. Since we only enlarge the tool body, the accuracy of tool accessibility is not affected.

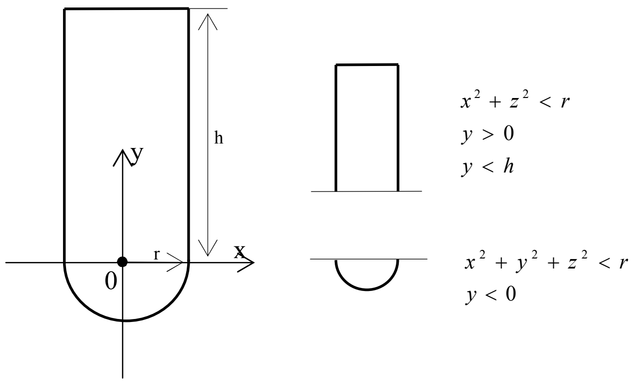


Fig. 6. Implicit equations representing the tool shape.

We also need to decide a good representation for the moving tool. It can be a triangle mesh, a point cloud, or an implicit representation [11]. Normally the machining tool is in regular shape which can be represented in the combination of algebraic equations. It is extremely fast to use these equations to detect collision. Fig. 6 illustrates how to implicitly represent a regular shape tool with multi constrains. Spheres, conics and cylinders or part of them can all be implicitly represented. After assembling them, we get the implicit representation of the whole tool shape. Though we have a fast intersection scheme, the huge point cloud size can easily swallow the computing time if we test all the points. A common 3D object may contain hundreds of thousands points in general, and test on all of them indiscriminately should be avoided. To accelerate the intersection test, space subdivision methods are usually adopted. We use a hierarchical bounding box representation – the *k*-Discrete Orientation Polytope (*k*-DOP) [6]. A simple 3-DOP is a Cartesian box with faces parallel to the *xy*, *yz* and *xz* planes. Increase *k* can enhance the performance, and at a certain value the trend reverses. Our system uses 27-DOPs to generate a binary tree. It is fast to traverse the *k*-DOP tree to mark the candidate set of the intersected points for collision detection. The *k*-DOP subdivision costs some extra searching time, but in return one gets a huge reduction in the number of candidate points for collision detection, which makes real time collision detection possible.

In 5-axsi tool path generation with haptic device, the collision detection is needed for two purposes: first, the tool's cutting surface (e.g., the hemisphere in case of spherical tool) must keep constant contact with the part surface and free of local gouging; second, the tool body must be clear of the part and the working environment. Previous works combine these two tasks into a single function to achieve and share the collision detection result for haptic rendering. However, in our system they are separated and the haptic constrains are calculated individually. Under the point cloud model, haptic rendering is very susceptible to the operator's shaking hand hence makes the data unreliable. The separation improves the reliability of the recorded data with no extra computing time requirement.

5. HAPTIC RENDERING DESIGN

Using collision detection result for haptic rendering is straightforward. But in the point cloud scheme, the errors cumulate and the accuracy of recording posture is lowered. The point cloud scheme leads penetration errors in both accessibility (Fig. 5(c), the hemispherical portion) and global interference (Fig. 5(d), the cylindrical portion). If the CL data is reliable, then we can eliminate the global interference error by enlarging the tool body portion as described before. This is reasonable because the tool's cutting surface is required to keep contact with the part surface while the tool body is desired to have a clearance gap with the part (and the working environment). But if we directly use the point cloud scheme in haptic rendering, CL errors occur in two aspects. The first one can be illustrated in Fig. 5(c); the other is force rendering. In the point cloud scheme, a force feedback should be calculated and reported when some points are inside the volume of the moving tool. Only one point can be processed in each time step due to the 1ms time limit. Actually there are usually more than one point inside the tool's volume and the scheme would always just deal with the first found one. This would cause the pen to shake and jump from the points due to the haptic rendering scheme but not the human's physical limitation. We propose a new method for dealing with this by utilizing surface constrains.

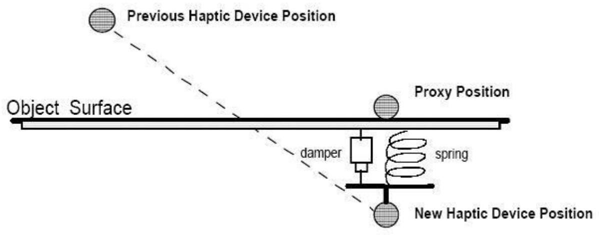


Fig. 7. Illustration of the haptic proxy.

The PHANTOM® haptic device provides abundant haptic effects for programming such as stiffness, damping, friction, spring, snap and etc. The OPENHAPTICS™ utilizes OpenGL depth buffer to generate touching feedback. This make it easy to render the rigid surface constrains. Refer to Fig. 7. When the pen moves to a new position that would penetrate the object surface, the haptic device will quickly calculate the proxy position. The forces can then be determined that will impede the motion of the haptic device's end-effector from further penetrating the contacted surface. Under the treatment like this, the pen (i.e., the tool) can be maintained to keep contact with the part surface. Thus, we settle the shaking and jumping problem in haptic rendering.

As we mentioned before, in a 5-axis haptic system the geometry of the tool should be represented faithfully. If we directly use the part surface as the haptic constrains, the device position should represent the CC point. This requires one to first calculate the contact point on the hemisphere surface (in the case of spherical tool) and then calculate the CL point for the tool posture in collision detection. Those computations adversely affect the refresh rate. We settle the problem by pre-processing the CL surface. Refer to the example shown in Fig. 8. The surface in gray color is the haptic constrain surface which will not be shown to the user. The device position represents the tool's reference point that will remain to lie on the CL surface (the haptic constrain surface). Borrowing the idea from collision detection, we only send the triangles near the guiding-plane to the haptic thread. We again use 27-DOPs to generate a binary tree of the CL surface. Therefore, only triangles in the vicinity of the end of the moving tool are used for haptic rendering and only points in the vicinity of the body of the moving tool are used for collision detecting. Reliability and real time refresh rate both are achieved.

Since only the 3-DOF force feedback device is used, we make a conversion to indicate to the user the collision between the tool body and the part surface or the working environment. If there are no collisions, user can move the pen (tool) freely along the part surface. If a collision occurs, the system will freeze the pen's translational movement and the user can only move it again after he has settled the collision problem by rotating the pen in a suitable orientation. Reducing the level of real world feeling will enhance the system's real time reaction. The computing time is reduced as well since we need not calculate the force vector when the tool body touches any object.

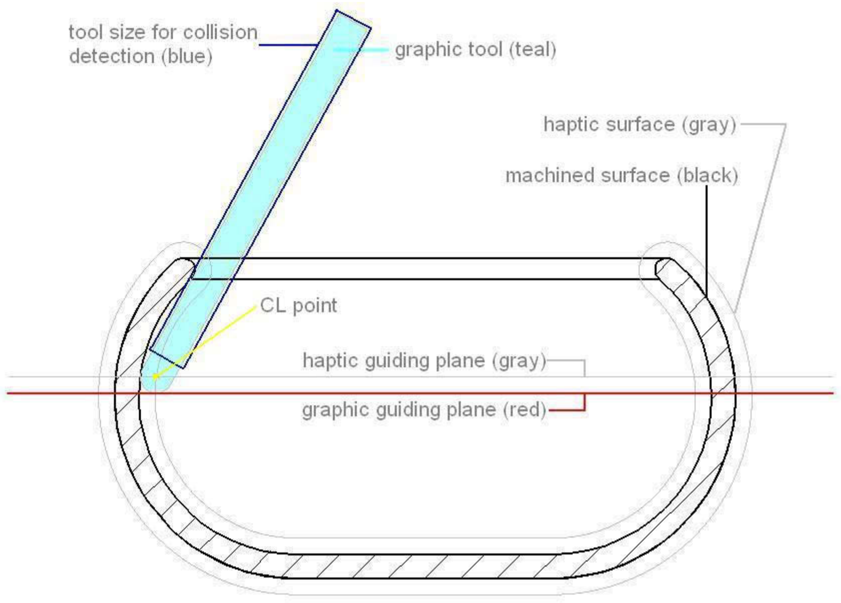

tool size for collision detection (blue)
graphic tool (teal)
haptic surface (gray)
machined surface (black)
CL point
haptic guiding plane (gray)
graphic guiding plane (red)

Fig. 8. Different surface/plane data in graphic and haptic rendering.

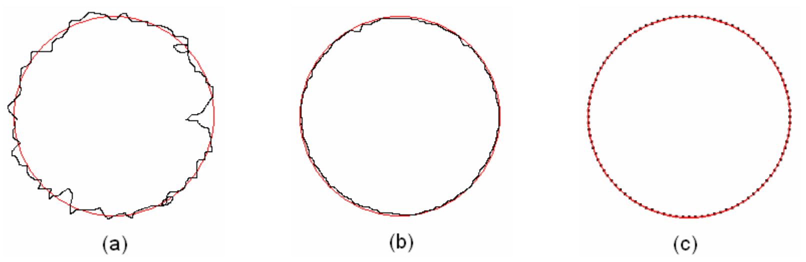


Fig.9. Captured position results.

5.1 Manual Manipulation under Basic Guiding

By now we have described a basic 5-axis collision-free tool path generation system. It can generate 5-axis collision-free tool path manually under some basic haptic guiding. In the vase sample example shown in Fig. 8, guide planes are provided to help user moving the pen steadily and smoothly. There are two Z-constant guiding planes in Fig. 8: the lower one is the OpenGL displayed plane and the upper one which has the offset r (tool radius) is the haptic constrain plane. Tool is colored in teal. Since we use point cloud in collision detection, we should consider the error as exemplified in Fig. 4(d). So the tool body is enlarged by a small size as shown in blue color. (Compare the different radii of the tool's shank and its bottom in Figure 8.) Only the tool body participates in the collision computing. Accessibility of the hemispherical portion is left to the hardware haptic rending. The CL point can always be kept on the haptic guiding plane. If tool contacts the part surface and no collision occurs, the current position and orientation of the pen (tool) will be recorded. To avoid unnecessary data, we compare new posture to the previous one. Only if the position changes up to a certain distance do we save the new posture and orientation. Fig 10 (left) shows the manual operation while the axis-constant guiding plane was turned on. The guiding plane is in yellow color and the object is in white color (un-machined part) and green color (machined part). The tool size (radius and length) can be specified by the user. We select some representative tool postures in the operation and display them at the same time. The tool in red color indicates collision detected and the tool can not move until user rotates the pen (tool) to avoid the collision. The user can also turn off the axis-constant guiding plane to have a full manual operation. Fig. 9 shows the comparison of the recorded results. The red circles indicate the theoretical (correct) CL data. When using the point cloud scheme in haptic rendering (Fig. 9(a)), it might incur large position errors, such as over penetration and weak contact. The pen in the operator's hand tends to shake under the point cloud scheme. In Fig. 9(b), only small errors occur under the proposed scheme of haptic constrains. The recorded posture data then become steady and reliable after a few post-processing operations (Fig. 9(c)).

5.2 Semi-Automatic Manipulations

We provide haptic guiding by axis-constant guiding plane and haptic CL surface. Due to the limit of the human's reaction speed, jumping errors, although small, still occur during the interaction. The pen is frozen when there is a collision, so the user changes the pen's orientation and tries to keep the pen moving on. When a detected collision is avoided (after changing the tool's rotation), a sudden release might cause the pen to be out of control. The operator may move the pen a long distance and make the recorded tool's orientation data very jaded. And he may also interrupt the interaction by moving the pen out of the guiding surface/plane and has to move the pen back to a previous position. All these demand that the user be familiar with the haptic system and be trained muscle-control expert. Despite these concerns, the manual mode is still necessary in repair and artistic applications.

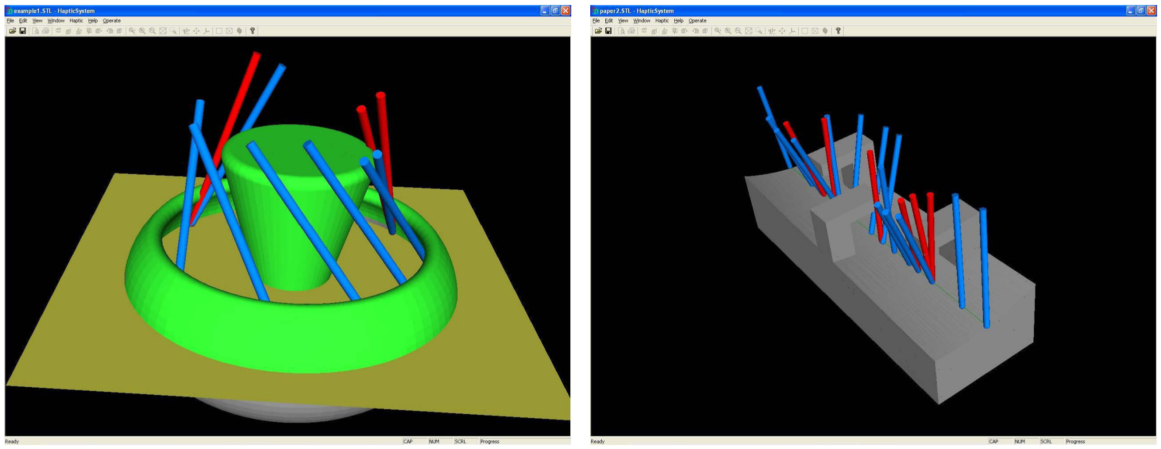

example1.STL - HapticSystem
File Edit View Window Haptic Help Operate
Ready
CAP NUM SCRL Progress
paper2.STL - HapticSystem
File Edit View Window Haptic Help Operate
Ready
CAP NUM SCRL Progress

Fig.10. Results of manual (left) and semi-automatic (right) manipulation.

As an improvement to the manual method, a semi-automatic collision-free 5-axis tool path generation method is designed in our system. The tool positioning problem has been extensively studied, and there exist many efficient positional tool path generation algorithms [1]. These algorithms can automatically compute roughing and finishing tool paths by defining strategies such as constant z-level, projected zig-zags, iso-scallop height, iso-parametric and etc. These tool paths are coined "positional" since they do not contain valid collision-free orientations. However, they can be utilized in haptic guiding and help the user find collision-free orientations eventually. The SensAble haptic device can perform snap effects of a plane, a line and a point. Here we use snap point effect as a leading force. In the example shown in Fig. 9(c), we calculate the CL points and display them as polygons on the screen. When the pen is snapped to a CL point, the position was lightly fixed and hence the user can rotate the pen at current position steadily. If interference happens between the tool and the part surface, we heavily fix the pen's position by means of the haptic rendering. If the interference disappears after the pen's rotation, the pen is released and can be snapped to the next CL point. This semi-automatic method greatly reduces unexpected movements of the pen as in the manual method. Only the tool axes at the CL points will be recorded. To save the computing time, we only keep two snap points, the current and next CL points, which shift in sequence. The user would feel like being drawn by a force leading him to accomplish the entire process. See Fig. 10 (right) for a test example. The green line is the given CL path along which the tool's reference point moves. When there is a global interference, the tool's position is frozen and it can only rotate. A red colored tool indicates the collision. We select some representative postures and show them in one picture.
Compared to the basic guiding, this semi-automatic guidance offers a more comfortable and efficient interaction environment for the user. Moreover, the data also become more reliable than the manual operation.

5.3 Roughing and Extension

We get the finishing tool path from manual or semi-automatic operation. For roughing, we can use the interior tool posture interpolation and tool posture correction to get a collision-free tool path. The algorithm and method can be found in [1] and [11]. We do not describe the detail here. An extension of the semi-automatic guidance system is in simulation and verification. The correctness of a tool path nowadays is always checked first through simulation on a computer. Like the snap point method, we now modify the pen position from the test tool path in sequence. When the current pen orientation aligns with the current test tool posture and no interference is detected, a haptic force will move the pen to the next test tool position. The user can correct the orientation when the tested posture causes a collision. We record the correct orientation and add this to the next test posture and make it to be current. The original orientation is the first choice if no collision occurs and the corrected orientation is a good reference for correction. During the process, we also provide a check on the rate of changing of the orientation.

6. SUMMARY

We have presented a collision-free 5-axis tool path generation system based on the commercial PHANTOM® haptic device. In a haptic application, the data reliability and the rendering refreshing speed always conflict each other. Our system balances and actually improves both. In our experiments, a part surface with 39,570 triangle takes a graphic rendering time of 41.276ms with GeForce6800 graphic card (256MB display memory) and of 95.419ms with Intel GMA900 (internal display card). Since the haptic rendering utilizes OpenGL depth buffer, a successful compression of the haptic rendering time is crucial for real time applications. By constructing sufficient haptic constraints and haptic effects, our system prevents hand jitter during the operation. This ensures the data accuracy and reliability. At the same time, by constructing a *k*-DOP binary tree to reduce the number of haptic constraint triangles, the haptic rendering time can be close to real time. We give the comparison results in Tab. 1 and Tab. 2. The maximum computing time is greatly reduced by the use of *k*-DOP tree while the minimum computing time only increases marginally. The *k*-DOP tree scheme is not sensitive to the mesh size either. Under our scheme, the haptic system works well even on objects of very larger mesh size. Most haptic systems consider the haptic rendering as a special case of the graphics rendering and use the same procedure for both. Our system separates the two and introduces an innovative scheme which can be expanded to other haptic applications.

	11722 Triangles	*22420 Triangles*	*139904 Triangles*
k-Dop Tree	0.034-0.369	0.036-0.365	0.037-0.562
Brute force	0.030-1.281	0.031-2.173	0.031-13.331

Tab. 1. Computing time (in ms) in triangle marking.

	6225 Points	*11552 Points*	*71275 Points*
k-Dop Tree	0.958-6.890	0.962-6.865	0.969-9.162
Brute force	0.953-26.345	0.952-47.742	0.966-327.894

Tab. 2. Computing time (in ms) in collision detection.

The proposed system focuses on collision-free 5-axis tool path generation with a ball-end tool. It can be extended in principle to the flat-end tool and other regular tool shapes. However, when dealing with a flat-end tool, the CL data should be dynamically changed when the user modifies the tool orientation. This is a challenging problem and will be one of our future research topics. Our system performs well in real time interaction for part surfaces with up to 30,000 triangles. However, triangle meshes of extremely large size (>100,000) are still a challenge in graphic rendering, haptic rendering, and collision detection. Full size rendering of such large size triangle meshes in both graphic and haptic swallows huge computing time and makes real time very difficult. The system performance can improve with the upgrading of the hardware such as RAM, CPU, and GPU. Using parallel computing in collision detection and pre- and post-processing is also an interesting research topic.

7. ACKNOWLEDGEMENT

This work is partially supported by the Hong Kong RGC Grant 05/06.EG620105.

8. REFERENCES

[1] Choi, B. K. and Jerard, R. B., *Sculptured surface machining – Theory and applications*, Kluwer Academic Publishers, 1998.

[2] Lauwers, B., Dejonghe, P. and Kruth, J. P., Optimal and collision free tool posture in five-axis machining through the tight integration of tool path generation and machine simulation, *Computer-Aided Design*, Vol. 35, No. 5, 2003, pp 421-432.

[3] Jun, C.-S., Cha, K. and Lee, Y.-S., Optimizing tool orientations for 5-axis machining by configuration-space search method, *Computer-Aided Design*, Vol. 35, No. 6, 2003, pp 549-566.

[4] Chiou, C.-J. and Lee, Y.-S., A machining potential field approach to tool path generation for multi-axis sculptured surface machining, *Computer-Aided Design*, Vol. 34, No. 5, 2002, pp 357-371.

[5] http://www.sensable.com

[6] Klosowski, J. T., Held, M., Mitchell, J., Sowizral, H. and Zikan, K., Efficient collision detection using bounding volume hierarchies of k-DOPs, *IEEE Transactions on Visualization and Computer Graphics*, Vol. 4, No. 1, 1998, pp 21-36.

[7] Chiou, J. C. J., Accurate tool position for five-axis ruled surface machining by swept envelope approach *Computer-Aided Design*, Vol. 36, No. 10, 2004, pp 967-974.

[8] Chiou, J. C. J., Floor, wall and ceiling approach for ball-end tool pocket machining, *Computer-Aided Design*, Vol. 37, No. 4, 2005, pp 373-385.

[9] Yoon, J.-H., Pottmann, H. and Lee, Y.-S., Locally optimal cutting positions for 5-axis sculptured surface machining, *Computer-Aided Design*, Vol. 35, No. 1, 2003, pp 69-81.

[10] Balasubramaniam, M., Laxmiprasad, P., Sarma, S. and Shaikh, Z., Generating 5-axis NC roughing paths directly from a tessellated representation, *Computer-Aided Design*, Vol. 32, No. 4, 2000, pp 261-277.

[11] Balasubramaniam, M., Ho, S., Sarma, S. and Adachi, Y., Generation of collision-free 5-axis tool paths using a haptic surface, *Computer-Aided Design*, Vol. 34, No. 4, 2002, pp 267-279.

[12] Ilushin, O., Elber, G., Halperin, D., Wein, R. and Kim, M.-S., Precise global collision detection in multi-axis NC-machining, *Computer-Aided Design*, Vol. 37, No. 9, 2005, pp 909-920.

[13] Cignoni, P., Montani, C., Perego, R. and Scopigno, R., Parallel 3D Delaunay Triangulation, *Computer Graphics Forum*, Vol. 12, No. 3, 1993, pp 129-142.

[14] Park, S. C. and Choi, B. K., Tool-path planning for direction-parallel area milling, *Computer-Aided Design*, Vol. 32, No. 1, 2000, pp 17-25.

[15] Sarma, S. E., The crossing function and its application to zig-zag tool paths, *Computer Aided Design*, Vol. 31, No. 14, 1999, pp 881-890.

[16] Park, S. C., Tool-path generation for Z-constant contour machining, *Computer-Aided Design*, Vol. 35, No. 1, 2003, pp 27-36.

[17] Kim, T. and Sarma, S. E., Optimal sweeping paths on a 2-manifold: a new class of optimization problems, *IEEE Trans. on Robotics and Automation*, Vol. 19, No. 4, 2003, pp 613-636.

[18] Zhu, W. and Lee, Y.-S., Five-axis pencil-cut planning and virtual prototyping with 5-DOF haptic interface, *Computer-Aided Design*, Vol. 36, No. 13, 2004, pp 1295-1307.

[19] Liu, X., Dodds, G., McCartney, J. and Hinds, B. K., Virtual DesignWorks – designing 3D CAD models via haptic interaction, *Computer-Aided Design*, Vol. 36, No. 12, 2004, pp 1129-1140.

[20] Lee, Y. S., Non-isoparametric toolpath planning by machining strip evaluation for 5-axis sculptured surface machining, *Computer-Aided Design*, Vol. 30, 1998, pp 559-570.

[21] Lee, Y. S. and Chang, T. C., Two-phase approach to global tool interference avoidance in 5-axis machining. *Computer Aided Design*, Vol. 27, No. 10, 1995, pp 715-729.

[22] Chen, Y, H., Wang, Y. Z. and Yang, Z. Y., Towards a haptic virtual coordinate measuring machine, *Int. J. Mach. Tools Manuf.*, Vol. 44, No. 10, 2004, pp 1009-1017.

[23] Chen, Y. H., Yang, Z. and Lian, L., On the development of a haptic system for rapid product development, *Computer-Aided Design*, Vol. 37, No. 5, 2005, pp 559-569.