Journal of Zhejiang University SCIENCE A
ISSN 1009-3095 (Print); ISSN 1862-1775 (Online)
www.zju.edu.cn/jzus; www.springerlink.com
E-mail: jzus@zju.edu.cn

Surfel-based surface modeling for robotic belt grinding simulation*

REN Xiang-yang†[1], MUELLER Heinrich[2], KUHLENKOETTER Bernd[1]
([1]*Robotics Research Institute, University of Dortmund, Dortmund 44227, Germany*)
([2]*Informatik VII, University of Dortmund, Dortmund 44227, Germany*)
†E-mail: xiangyang.ren@uni-dortmund.de

Received Apr. 10, 2006; revision accepted Apr. 19, 2006

Abstract: The new free-form surface modelling technology for robotic belt grinding simulation presented in this paper is based on discrete surfel elements generated from the surface approximation point set and can facilitate the simulation implementation. A local process model exploits the advantage of surfel representation to compute the material removal rate and the final surface grinding error can be easily carried out. With the help of this system, robot programmers can improve the path planning and predict potential problems by visualizing the manufacturing process.

Key words: Surface modelling, Surfel, Belt-grinding simulation
doi:10.1631/jzus.2006.A1215 **Document code:** A **CLC number:** TP39

INTRODUCTION

The simulation and verification technologies of numerical controlled (NC) manufacturing processes have been developed since the end of the 1970s, and a lot of significant accomplishments have been achieved. This kind of technology has important impact on product development and quality control. The designed CAD model can be produced to virtually reduce or even eliminate expensive experiments on testing material. If any potential problem, such as collision, improper parameters or gouge is found during the simulation, the manufacturing process can be adjusted to ensure the quality. Due to these features, simulation and verification have great promise for cost reduction, quality improvement and time-to-market shortening.

Normally, NC cutting process simulation technologies can be divided into two groups: analytical methods and approximation methods. The former can accurately describe the manufacturing process and calculate the material removal rate. But the complexity of swept volume formulation and time consuming boolean set operations prevent it from wide application. Instead, the second method approximates the workpiece by a set of discrete basic elements and the process simulation is represented by modifying these elements. It achieves better performance without seriously harming the accuracy.

Analytical approaches

These approaches are mostly direct solid modelling approaches, such as constructive solid geometry (CSG) and boundary representation (B_Rep) modeling systems capable of simulating the material removal process through a series of regularized boolean difference operations to subtract successive tool swept volumes from the workpiece. The result is an explicit solid model of the workpiece. Kawashima *et al*.(1991) used a special geometric modeling method called Graftree to speed up this solid modeling approach, which allowed accurate and precise representation of the workpiece and the tool. An octtree helped to decrease the number of ray-intersection calculations in rendering. Another approach to find the exact representation of the envelope of a swept

* Project supported by the Deutsche Forschungsgemeinschaft (DFG) as a part of the research group 366 (Simulation-Aided Offline Process Design and Optimization in Manufacturing Sculptured Surfaces)

volume was chosen by Sourin and Pasko (1996). Although these approaches can theoretically simulate and verify the process accurately, their application remains limited by the complexity of swept volume formulation and the time-consuming rendering process. So a number of approximation approaches have been devised.

Approximate approaches

van Hook (1986) developed a real-time shaded display of a model suitable for manufacturing simulation. The workpiece and tool geometries are represented by dexels and the geometry update is achieved by boolean set operations on the one-dimensional dexels. Huang and Oliver (1994) extended this method by overcoming the view point dependency problem and introduced the possibility of error assessment. König and Gröller (1998) developed a new method to simulate inhomogeneous materials based on the dexel model. Furthermore, their approach has been optimized for low-end graphic hardware. Saito and Takahashi (1991) also used an extension of the z-buffer method (called G-buffer) to simulate NC machining. Jerard *et al*.(1989) proposed a completely different approach. The designed surface is approximated by a set of points. The workpiece is represented by these points and the normal vectors associated with them. The vectors are shortened to the amount of over or undercutting error when a tool moves over them. This method is very efficient for error evaluation but inconvenient for calculating the material removal rate. Ayasse (2003) used a very coarse mesh as the support mesh, on which a continuous vector field is standing, with each vector length being decided by a discrete height field. The simulation procedure is implemented using V-projection and V-shooting. Glaeser and Gröller (1998) applied differential geometric techniques to efficiently generate the swept volume of a moving cutter. Intersection calculations are carried out with a data structure called Γ-buffer. Yang and Lee (1996) also developed a method suitable for wire-EDM, where cutting is only done at the sides of the workpiece. A comparison of those approaches can be found in (Glaeser and Gröller, 1998).

Motivation

Most of the approaches mentioned above were originally developed for NC milling processes. But robot controlled belt-grinding has its own characteristics. Its removal volume is a function of many variables including cutting parameters, environmental parameters and material features, not just the envelope of cutter profiles. It is used as a finishing process with the stock roughly machined, which means that the difference between the workpiece stock and the designed object is quite small. Compared with the simulation of milling or turning processes, the simulation of belt grinding is much more difficult. So it is important to find a suitable way to represent the workpiece, which can facilitate the simulation of material removal. This kind of method is designed to take into account the characteristics of belt grinding and maintain the advantage of previous approximation technologies. We exploit the recent achievement in the area of point-based rendering and combine it with the discrete vector method to develop a new workpiece representation for free-form belt grinding simulation. The dimensional grinding error assessment can be easily carried out with this representation.

SURFEL-BASED MODELLING OF FREE-FORM SURFACES

Fundamentally, all the approximation simulation technologies discretize the workpiece to take advantage of image space rendering. They include a selected base (support) and a set of certain kind of basic elemental component. van Hook (1986) and Saito and Takahashi (1991) used a chosen x-y plane as the base and approximate the solid body with one-dimensional elements called dexel or G-buffer. Jerard *et al*.(1989) based their discretization process on continuously smooth surface and applied a collection of normal vectors associated with points on that surface to represent the workpiece while Ayasse (2003) just took the surface triangle mesh as the support base. With the development of point-based rendering technology (Levoy and Whitted, 1985; Pfister *et al*, 2000; Rusinkiewicz and Levoy, 2000; Zwicker *et al*., 2001), it does make some sense to represent the workpiece by uniformly and densely sampled points and some kind of basic elements associated with them.

Surface discretization

The term "surfel" is an abbreviation for surface element or surface voxel in the volume rendering and discrete topology literature. Herman (1992) defined a surfel as oriented (d−1)-dimensional object in $\mathbb{R}^d$. For (d=3), this corresponds to an oriented unit square (voxel face) and is consistent with thinking of voxels as little cubes. Pfister *et al*.(2000) modified the definition as a zero-dimensional n-tuple with shape and shade attributes that locally approximate an object surface. In keeping with the convention of dexel or G-buffer, we adapt it to our application and redefine it as:

A surfel is an oriented round disc, which locally approximates an object surface, extends in the normal direction and includes information on shape, shade, displacement and neighborhood attributes.

Accordingly, the design of the surfel data structure should include the position and orientation, texture information, local differential geometry, neighborhood and some modification flags. Actually, it looks more like a dexel representation method in parameterized space as shown in Fig.1 (see page 1222).

The first step to generate the surfel-based representation of smooth surfaces is to approximate the surfaces by a set of sampled points. Two aspects should be considered when deciding the sampling density. One is the simulation error of the grinding process. From another point of view, it is also the surface approximation error. Since a point is a piecewise constant surface approximant, the resulting approximation power is linear, which means given an average spacing h between the samples P_i, the approximation error with respect to each coordinate function is of the order $\Theta(h)$. The resulting point number is square proportional to the required precision. Another aspect is the visual artifact. Normally the surfaces should be sampled uniformly, but occasionally we have to change it to improve the rendering effect as shown in later sections.

Usually, sampling methods perform object discretization as a function of geometric parameters of the surface, such as curvature or silhouettes. This object space discretization typically leads to too many or too few primitives for rendering. Pfister *et al*.(2000) discretized the object aligning to image space to match the expected output resolution. They sampled geometric models from three sides of a cube into three orthogonal LDIs, called a layered depth cube (LDC). Ray casting records all intersections and generates surfels at each intersection point. The surface normals are perturbed by bump and displacement mapping. After the LDC sampling, the resulting point set has to be reduced to eliminate the redundancy. Mueller *et al*.(2003) developed a volume representation of objects called multi-dexel using similar technologies. It consists of three orthogonally conventional dexel volumes and is quite suitable for milling simulation. Surface elliptical splats were first proposed for rendering purposes by Zwicker *et al*.(2001). According the differential geometry, such an ellipse is the best linear approximation to a smooth surface. So it is superior to triangle meshes but still maintains the C^{-1} continuity and provides better topological flexibility.

Although the elliptical splat seems to be the best approximation of an object for rendering purpose, it is not suitable for material removal process simulation, which requires very dense sampling of workpiece surfaces to ensure the required precision. If we base the surfel data structure on ellipses, then in certain areas we may get some ellipses with a long u_i or v_i axis. During the machining process, the cutter may contact a small part of them, and then it becomes very hard to handle such a situation because the large ellipse should be slipped and re-sampled. So we adopt densely sampled points as the basis of the surfel data structure. LDC sampling is easy to implement and can generate well-distributed points after reduction. But it has also some drawbacks when representing surfaces with high curvature variation and sharp features. In the current stage most free-form surfaces to be processed are represented as spline surfaces, which means that they can be regarded as two-dimensional domains and that the vertices of regular grids in these domains can be taken as the point set. The more grids are generated by u and v isolines, the higher the density of the surface sampling we get. Since the surface is described analytically and smoothly, more differential geometry attributes can be obtained.

In surfel-based model representation, the support basis is the approximate point set of the designed surface and the workpiece stock is represented by the displacement of each point along the normal direction. So after the discretization of the designed surfaces, the next step is to compute the displacement. Since we have already got the position and orientation of each

point, a corresponding ray is formed and used to calculate the intersection with stock surfaces. The procedure is quite similar to ray tracing. As a finishing process, the belt grinding removes a small amount of material from the stock. So the resulting displacement is small and more intersection points between one ray and workpiece stock is unlikely to happen. Otherwise, it needs to be specially handled to avoid artifact effect. Although it is a one-time process prior to simulation, the generation speed is still an important problem, especially for large and complex workpieces. So to accelerate the computation, we arrange the stock surface in a spatial data structure.

Neighborhood and normals

There are two kinds of normal vectors associated with the surfel data structure. One is the normal direction of the designed surface approximate point. It is also the extension direction of the base point. Another is the normal of the displaced point after the displacement calculation. The former can be directly obtained when sampling the designed surface while the latter has great impact on the process rendering and has to be generated using other technologies. One way is to estimate it by analyzing the local neighborhood of each point. These local neighborhoods are usually constructed using either Euclidean neighborhoods or *k*-nearest neighbors without considering the connectivity information.

By considering the surface sampling method previously adopted, we can find that the point set consists of the vertices of regular grid in a 2D parametric domain and that the connectivity information is explicit. So normal perturbation technology, such as bump mapping or displacement mapping, seems to be an ideal choice to estimate the normal vector of the displacement point. Blinn (1978) introduced bump mapping as a technique that makes a surface appear rough or wrinkled. This effect is achieved by perturbing surface normals on a surface as if a height field displaced the surface in the direction of the original surface normal. Since bump mapping only changes the appearance of an object, it makes certain approximations. Standard techniques for bump mapping assume that the bumpiness is only a bulk of micro-displacements and hence the magnitude of the height field is negligible. Since the geometry remains unchanged, some drawbacks on the silhouette and shadow are introduced. Displacement mapping is a method to present surface details by defining an offset (displacement) from a base surface. It differs from bump mapping in that the surface geometry is also modified instead of only perturbing the normals. This results in a more realistic rendering where the displaced geometry can be seen in the silhouettes too. The basic idea of displacement mapping is very simple. The base surface is perturbed along its normals using displacement values specified in the displacement map. As a result the new displaced surface is created. Doggett and Hirche (2000) defined this in mathematical framework as shown in Fig.2.

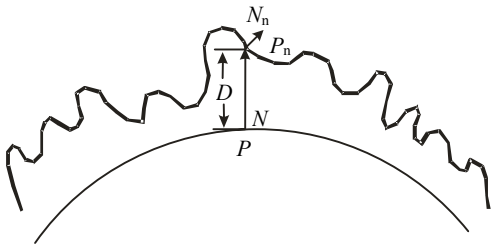

Fig.2 Displacement mapping algorithm

The base surface is defined by a vector function $P(u,v)$ that defines 3D points (x,y,z) on the surface. Normals for the base surface are defined by $\hat{N}(u,v)$ and the displacement scale fields by $D(u,v)$. Both of these are defined over the same domain as the base surface $P(u,v)$. Following these notations the points on the new displaced surface $P_n(u,v)$ are defined by

$$P_n(u,v) = P(u,v) + D(u,v) \cdot \hat{N}(u,v), \qquad (1)$$

where $\hat{N}(u,v) = \dfrac{N(u,v)}{|N(u,v)|}$.

However, things are a little different in our application. The base surface is the splatted designed surface while the displacement map is not a given variable. Instead, it is the outcome derived from the ray tracing procedure. This means that the $N(u,v)$ in Eq.(1) is known beforehand. Our task is only to determine the normal vectors of points in the new offset surface. The procedure is the same as described by Blinn (1978). The normal vector to this new surface is derived by taking the cross product of its partial derivatives.

$$N_{\mathrm{n}} = P'_{\mathrm{n}u} \times P'_{\mathrm{n}v}, \tag{2}$$

while

$$\begin{aligned} P'_{\mathrm{n}u} &= \frac{\mathrm{d}}{\mathrm{d}u} P_{\mathrm{n}} = \frac{\mathrm{d}}{\mathrm{d}u}(P(u,v) + D(u,v)\cdot \hat{N}(u,v)) \\ &= P'_u + D'_u \cdot \hat{N} + D \cdot \hat{N}_u, \\ P'_{\mathrm{n}v} &= \frac{\mathrm{d}}{\mathrm{d}v} P_{\mathrm{n}} = \frac{\mathrm{d}}{\mathrm{d}v}(P(u,v) + D(u,v)\cdot \hat{N}(u,v)) \\ &= P'_v + D'_v \cdot \hat{N} + D \cdot \hat{N}_v. \end{aligned} \tag{3}$$

As mentioned above, the range of the displacement map $D(u,v)$ is negligibly small compared with the overall dimension. So $D \cdot \hat{N}'_u$ and $D \cdot \hat{N}'_v$ can be discarded and the new normal vector can be simplified to the following equation.

$$\begin{aligned} N_{\mathrm{n}} &= (P'_u + D'_u \cdot \hat{N}) \times (P'_v + D'_v \cdot \hat{N}) \\ &= P'_u \times P'_v + (D'_u \cdot \hat{N}) \times P'_v + \hat{P}_u \times (D'_v \cdot \hat{N}) \\ &\quad + (D'_u \cdot \hat{N}) \times (D'_v \cdot \hat{N}) \\ &= N + D'_u \cdot (\hat{N} \times P'_v) + D'_v \cdot (\hat{N} \times P'_u). \end{aligned} \tag{4}$$

Since the normal vectors of affected areas keep changing during the machining process, it is important to record the neighbor points in order to compute the perturbation dynamically. In case of large scale deformations, not only is the displacement mapping not valid any more, also serious visual artifacts will take place. In such a situation, the technology described here is no longer suitable and boolean operations on surfel-bounded solids may be considered as an alternative (Adams and Dutre, 2003).

Sharp feature representation

When using surfels to represent the workpiece with sharp features, some rendering problems will emerge. One is aliasing artifacts which occur mostly across the boundary of patches that do not have any neighboring patches. It turns out to be more serious when the surfel radius becomes larger. Zwicker *et al*.(2001) used a partial coverage of surfaces in fragments to implement edge anti-aliasing. Chhugani and Kumar (2003) suggested a solution which reduces the point size on the boundary or the silhouettes. It may result in an uncovered area or holes after the surface sampling.

Sharp features are usually composed by some patches connected in a non-smooth manner, which means that the normal values of the boundary have appreciable discontinuity in neighboring patches. These features cause serious visual problems by surfel representation as shown in the left picture of Fig.3. Chhugani *et al*. alleviated this issue by smoothing normals near the boundary. These normals are averaged with the nearby boundary points on the adjacent patch. But it has a fatal drawback when adopted in surfel representation because it changes the extension direction of the base point. Pauly *et al*.(2003) used an extension of the surface splatting technique presented in (Zwicker *et al*., 2001). This clipping technology can generate perfect visual effects. But some irregular shapes may be derived from clipping, which are difficult to describe by our new surfel elements. So we developed another kind of splatting technology, called "boundary re-sampling" to eliminate both edge aliasing and sharp feature representation as shown in Fig.3.

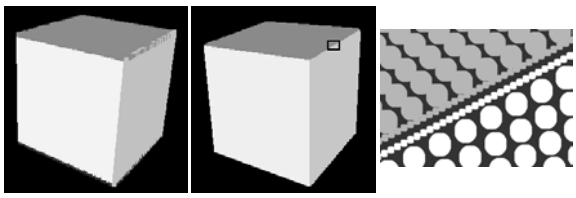

Fig.3 Sharp features representation example

The idea is inspired by the work of Adams and Dutre (2003), who used clipping and splitting to approximate the intersection line between two solids. In our method, we scan every surfel after the designed surface discretization. If it is located on the patch boundary, then the distance to the next surfel on the same boundary curve L is calculated. According to a user defined splitting interval e, the number of inserted new surfels can be calculated as $n=L/e$. The center of these surfels is $C_i=C_0+i\cdot e$ and the radius is assigned as e. Those surfels, whose radii are larger than the distance to boundary curves are also found and the radii are adjusted to improve rendering.

ROBOTIC BELT GRINDING SIMULATION USING SURFELS

Based on the above mentioned technologies, we

can now implement the simulation of robotic belt grinding processes. The workpiece is represented by surfels and the elastic contact wheel, with grinding belt and fixtures in a robot manufacturing cell consisting of polygons. The pre-defined grinding paths are composed of contact points. In each contact point, the contact situation is determined by a localization procedure in order to calculate the acting force. Then the real removal distribution is computed by the local process model. The material removal can be simulated by continuously modifying the surfels in affected areas. In this way, the generation of the removal volume is eliminated without sacrificing accuracy if the distance between two contact points is small enough.

The designed workpiece to be ground is usually represented by B_Rep and consists of some spline patches. In order to generate the surfel representation, it must be splatted. The density of sampling is related to the resolution of the FEM meshes, which is decided by user specified precision. The finer the meshes are, the denser the surface sampling rate is. For the boundaries of each patch, which is a component of a sharp feature, a re-sampling procedure is applied to meet the displaying quality requirements. Then for a given stock, the displaced point of each surfel element is calculated with ray casting.

Removal calculation

Within the grinding process of free-form surfaces the material removal computation is one of the most important, maybe also the most difficult aspect. The whole simulation system is driven by incrementally subtracting material from the workpiece stock. As mentioned above, it cannot use boolean set operations between the tool envelop and workpiece like turning or milling processes. Instead, the calculation should be based on an empirical model taking into account many influencing parameters. In such a complicated situation, the linear global grinding model given by Hammann (1998) is not suitable anymore. Particularly, the local non-uniform force distribution in the contact area must be considered and the influence of other manufacturing parameters also needs to be studied. The procedure can be divided into three steps: contact situation determination, force distribution calculation and removal computation. The first one describes the geometric intersection between the grinding belt and the workpiece, which will be used to obtain the pressure in the contact area in the second phase. Then other parameters are included to get the final removal. The whole process can be illustrated as shown in Fig.4 (see page 1222).

In the application, the workpiece is represented by surfels and the stock is composed of those displaced points of the surfels. On each contact point of the pre-planned path, those surfels whose displaced points are located above the contact tangent plane are to be found. We select the tangent plane as the X-Y plane and project each affected surfel displaced point to this plane to get its Z value. According to the requirement of FEM, these randomly distributed points are used to generate a regular grid mesh by extrapolating or interpolating Z values in those grid points where no data exists. Kriging (Oliver and Webster, 1990) is one of the more flexible methods and is useful for gridding almost any type of dataset. With most datasets, kriging method with a linear variogram is quite effective and generates the best overall interpretation. For larger datasets, however, kriging can be rather slow. So the faster but less precise Shepard's method (Shepard, 1968) is adopted to improve the simulation efficiency.

The acting force is not easy to measure because the sensors are difficult to install in order to obtain the fast-changing local force distribution during high speed grinding. So, approximate solution is very popular in this area if the deformation is small enough. Then it can be regarded as a wholly elastic deformation and the classic theoretical model of strain-stress can be applied. Once the deformation information on the elastic contact wheel is known, the acting force in the contact area can be also computed according to the classic strain-stress relation. The calculation turns out to be a Signorini problem and the FEM is a self-evident technique in the first place to solve it (Blum and Suttmeier, 2000; Blum *et al*., 2003).

Finally, the influence of other process parameters should be combined together to produce the results, which is denoted by the term process model in grinding. The local model does not simply take a single quantity as the force or the removal but present the local situation of the entire contact area using a pattern of values. The local model generalized by Hammann's model (Hammann, 1998), can be ex-

pressed as

$$\boldsymbol{r} = C_{\mathrm{A}} \cdot K_{\mathrm{A}} \cdot k_{\mathrm{t}} \cdot \frac{V_{\mathrm{b}}}{V_{\mathrm{w}} \cdot L_{\mathrm{w}}} \cdot f(\boldsymbol{F}_{\mathrm{A}}), \tag{5}$$

where $\boldsymbol{r}$ is the material removal rate, C_{A} is a constant decided by experiments, K_{A} is the combination constant of resistance factor, k_{t} is the grinding belt wear factor, V_{b} is the grinding velocity, V_{w} is the workpiece in-feed speed, L_{w} is the width of the grinding area and $\boldsymbol{F}_{\mathrm{A}}$ is the acting force between contact wheel and part. Particularly,

$$\boldsymbol{F}_{\mathrm{A}} = \begin{pmatrix} F_{11} & \cdots & F_{1n} \\ F_{21} & \cdots & F_{2n} \\ \vdots & \ddots & \vdots \\ F_{m1} & \cdots & F_{mn} \end{pmatrix}, \ \boldsymbol{r} = \begin{pmatrix} r_{11} & \cdots & r_{1n} \\ r_{21} & \cdots & r_{2n} \\ \vdots & \ddots & \vdots \\ r_{m1} & \cdots & r_{mn} \end{pmatrix},$$

in which m and n are the mesh sizes of the discretized contact area in two directions. In this model, the force $\boldsymbol{F}_{\mathrm{A}}$ or the removal rate $\boldsymbol{r}$ is no longer a one-valued parameter, but a matrix that represents the entire discrete information in the contact area. It contains more information than the global grinding model and is applicable to the grinding of free-form surfaces. Illustrations of these steps on one grinding point are shown in Fig.5 (see page 1222).

Belt grinding simulation

Each contact point in the prescribed paths has two attributes: position and orientation. The position is the Cartesian coordinate $P\{x,y,z\}$ in the robot base coordinate system and the orientation is the posture of the local framework on that point. Although different types of robots have different ways to represent the orientation, we choose quaternion $Q\{w,x,y,z\}$ as the standard method. Conversions may be required when applied to various robots. To ensure the approximation accuracy, the distance of neighboring contact points should be small enough. So for pre-planned paths, some extra points may need to be added. The position of these points can be linearly interpolated as shown in Fig.6a.

Concerning the rotation, it is more complicated. As we know, all unit quaternions are mapped onto a hypersphere in 4D space. The problem with the linear interpolation (lerp) of quaternions is that it interpolates

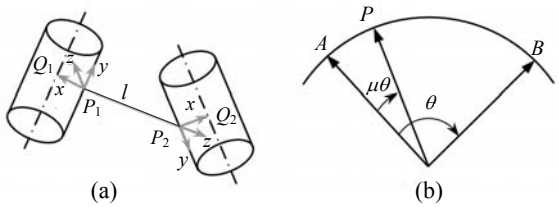

Fig.6 Interpolation of contact points. (a) Linear interpolation of position; (b) Spherical linear interpolation of orientation

along the secant line between the two quaternions but not their spherical distance. As a result, the interpolated motion does not have smooth velocity: it may speed up too much in some sections. Spherical linear interpolation (slerp), as illustrated in Fig.6b, removes this problem by interpolating along the arc lines instead of the secant lines. Although higher order interpolation may achieve C^2 continuity, slerp can obtain satisfying result with relatively low computation costs. For the ith point between Q_1 and Q_2, the interpolation result is:

$$\begin{aligned} &\mu = i/n, \\ &P_i = (1-\mu)\cdot P_1 + \mu \cdot P_2, \\ &Q_i = \frac{\sin(1-\mu)\theta}{\sin\theta}\cdot Q_1 + \frac{\sin\mu\theta}{\sin\theta}\cdot Q_2, \end{aligned} \tag{6}$$

where $\theta=\arccos(Q_1\cdot Q_2)$.

Fig.7 (see page 1222) shows an example of water tap grinding simulation along one of the paths.

Grinding error assessment

One of the simulation goals is to help the robot programmer estimate the final part surface quality. So it is of great importance to assess the grinding error to see whether any problem such as gouge or undercut happens. The error is usually described as the discrepancy between the surfel-based grinding surface and the designed smooth surface. For a surfel element, it means the distance between the surfel displaced point and the designed surface. So this verification procedure is usually converted to the calculation of surface near points. Most of the near point computing algorithms for sculptured surfaces are based on such a fact that the minimum distance between a space point P and a surface S occurs at a surface point Q at which the vector $\boldsymbol{PQ}$ is perpendicular to the surface tangent plane. This property forms a system of non-linear

equations and is solved by using a Newton/Raphson iterative search procedure. In a dexel model, all the elements are perpendicular to the chosen basis plane and the problem of error estimation is converted into a problem of calculating the surface near point between the dexel point and the designed surfaces. It can be solved by the method mentioned above if well-chosen initial points have been selected. Huang and Oliver (1994) suggested a pre-processed voxel data structure constructed in a bounding box of all the design surfaces to provide close-enough starting points.

Compared with dexel data structure, the surfel-based representation has a great advantage in analyzing the grinding error. The base points of surfels are the output of the design surface splatting and are exactly located on the surface, while the displaced points are merely moved a certain distance along the directions of the surface normal vectors. All what we need to do is to compute the distance between the base point and displaced point for each surfel element and compare it with the error tolerance limit. If the final length is longer than the upper limit (assuming that the tolerance range is $[t_l,t_h]$), then undercut happens. Otherwise if it is beneath the lower limit, then it leads to gouge. The iteration procedure to find the surface near point is omitted and the time of computation is only proportional to the surfel element number ($\Theta(n)$). Since only those affected surfel on one contact point need to be updated, the localization procedure has already limited the search area. So the

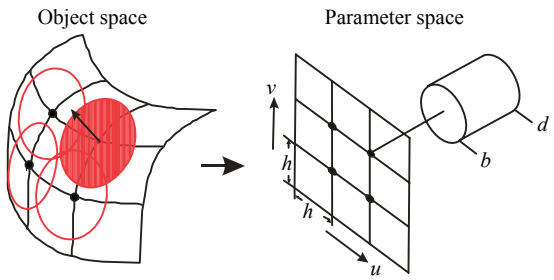

Fig.1 Parameterized surfel coordinate system

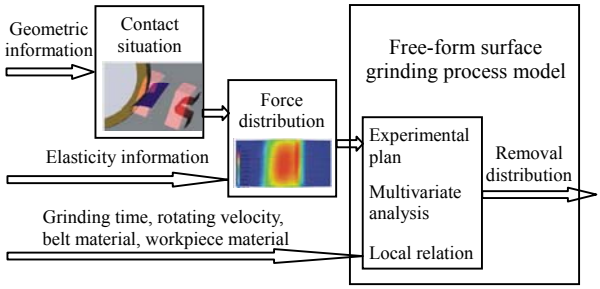

Fig.4 Removal calculation procedure

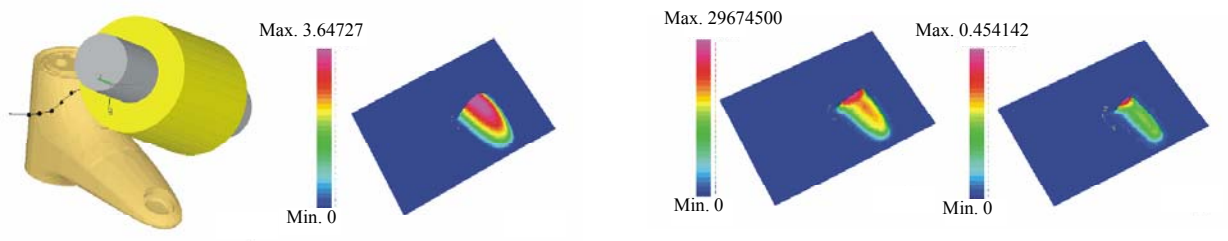

Fig.5 Removal calculation of one contact point. (a) The contact point in a grinding path; (b) Contact situation of that point; (c) Force distribution computed from the contact situation; (d) The final removal distribution in that contact area

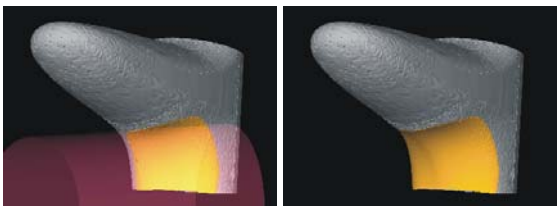

Fig.7 Faucet grinding simulation. (a) Workpiece stock ground after 3 grinding points; (b) Workpiece stock ground after whole grinding path

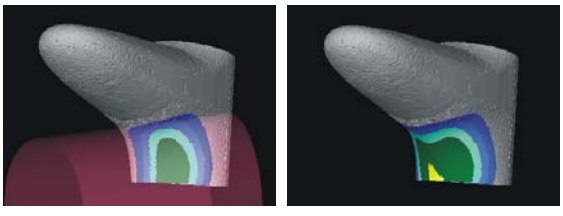

Fig.8 Grinding error assessment. (a) Analysis after 3 points grinding; (b) Analysis after the whole path grinding

efficiency can be further improved. The results of grinding error assessment are visualized by several hues depicting the grinding depth. The hue index is built based on the error arrange and for a given grinding error of one surfel, a corresponding color is picked to represent it. For example, in Fig.8b (see page 1222), the error of the dark green area is within the tolerance range relative to the nominal designed surface while the outside brighter area represents undercut and the inside yellow represents the amount of gouge.

There are two ways to implement the assessment: online assessment and post analysis. Online grinding error assessment calculates the difference between the designed surface and the stock and displays it dynamically. This method is more vivid and instant for observers. In contrast, post analysis computes the error after all the tool paths have been processed. It is obviously more efficient.

CONCLUSION

We have presented a surfel-based surface modelling technology for robot controlled belt grinding simulation. It combines the recent development of point-based modelling and rendering technology with the traditional discrete vector simulation approach and therefore is suitable for representation of free-form workpieces. With this modelling technology, the material removal process can be visualized in an interactive way and the grinding error is easy to be evaluated. By incorporating a local grinding model, the simulation system can efficiently simulate and precisely verify the robot programs. Based on the outcome, the robot programmer can optimize and correct the planned grinding paths in case of any unexpected situation such as undercut, gouge or singularity points.

ACKNOWLEDGEMENT

Special thanks are due to Xiang Zhang and Malik Cabaravdic for their constructive comments and advice.

References

Adams, B., Dutre, P., 2003. Interactive boolean operations on surfel-bounded solids. *ACM Trans. Graph.*, **22**(3): 651-656. [doi:10.1145/882262.882320]

Ayasse, J., 2003. Discrete Displacement Fields: A Versatile Representation of Geometry for Simulation in Computer-Aided Manufacturing. Ph.D Thesis, University Dortmund, Dortmund.

Blinn, F.J., 1978. Simulation of Wrinkled Surfaces. Proceedings of the 5th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH'78. New York, USA, p.286-292. [doi:10.1145/800248.507101]

Blum, H., Suttmeier, F.T., 2000. An adaptive finite element discretisation for a simplified Signorini problem. *Calcolo.*, **37**(2):65-77. [doi:10.1007/s100920070008]

Blum, H., Schroeder, A., Suttmeier, F.T., 2003. A Posteriori Error Bounds for Finite Element Schemes for a Model Friction Problem. Witten-Bommerholz.

Chhugani, J., Kumar, S., 2003. Budget Sampling of Parametric Surface Patches. Proceedings of the 2003 Symposium on Interactive 3D Graphics. New York, USA, p.131-138.

Doggett, M., Hirche, J., 2000. Adaptive View Dependent Tessellation of Displacement Maps. Proceedings from the ACM SIGGRAPH/EUROGRAPHICS Workshop on Graphics Hardware. New York, USA, p.59-66.

Glaeser, G., Gröller, E., 1998. Efficient Volume-generation During the Simulation of NC-milling. *In*: Hege, H.C., Polthier, K. (Eds.), Mathematical Visualization. Springer, Heidelburg, p.89-106.

Hammann, G., 1998. Modellierung des Abtragsverhaltens Elastischer Robotergefuehrter Schleifwerkzeuge. Ph.D Thesis, University Stuttgart, Stuttgart, Germany.

Herman, T.G., 1992. Discrete multidimensional jordan surfaces. *CVGIP: Graph. Models Image Process*, **54**(6):507-515. [doi:10.1016/1049-9652(92)90070-E]

Huang, Y., Oliver, H.J., 1994. NC Milling Error Assessment and Tool Path Correction. Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH'94, p.287-294. [doi:10.1145/192161.192231]

Jerard, R.B., Hussaini, S.Z., Drysdale, R.L., Schaudt, B., 1989. Approximate methods for simulation and verification of numerically controlled machining programs. *The Visual Computer*, **5**(6):329-348. [doi:10.1007/BF01999101]

Kawashima, Y., Itoh, K., Ishida, T., Nonaka, S., Ejiri, K., 1991. A flexible quantitative method for NC machining verification using a space-division based solid model. *The Visual Computer*, **7**(2-3):149-157. [doi:10.1007/BF01901185]

König, A.H., Gröller, E., 1998. Real Time Simulation and Visualization of NC Milling Processes for Inhomogeneous Materials on Low-end Graphics Hardware. Proceedings of the Computer Graphics International 1998, p.338-349.

Levoy, M., Whitted, T., 1985. The Use of Points as a Display Primitive. Technical Report, Computer Science Department, University of North Carolina at Chapel Hill.

Mueller, H., Surmann, T., Stautner, M., Albersmann, F., Weinert, K., 2003. Online Sculpting and Visualization of

Multi-dexel Volumes. Proceedings of the Eighth ACM Symposium on Solid Modelling and Applications, p.258-261.

Oliver, M.A., Webster, R., 1990. Kriging: a method of interpolation for geographical information system. *Int. J. Geographical Information Systems*, **4**(3):313-332.

Pauly, M., Keiser, R., Kobbelt, P.L., Gross, M., 2003. Shape modeling with point-sampled geometry. *ACM Trans. Graph.*, **22**(3):641-650. [doi:10.1145/882262.882319]

Pfister, H., Zwicker, M., van Baar, J., Gross, M., 2000. Surfels: Surface Elements as Rendering Primitives. Procedings of SIGGRAPH'00, p.335-342.

Rusinkiewicz, S., Levoy, M., 2000. QSplat: A Multiresolution Point Rendering System for Large Meshes. Proceedings of SIGGRAPH'00, p.343-352.

Saito, T., Takahashi, T., 1991. NC Machining with G-buffer Method. Proceedings of the 18th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH'91, ACM Press, p.207-216. [doi:10.1145/122718.122741]

Shepard, D., 1968. A Two-dimensional Interpolation Function for Irregularly-spaced Data. Proceedings of the 1968 23rd ACM National Conference. New York, USA, p.517-524. [doi:10.1145/800186.810616]

Sourin, I.A., Pasko, A.A., 1996. Function representation for sweeping by a moving solid. *IEEE Transactions on Visualization and Computer Graphics*, **2**(1):11-18. [doi:10.1109/2945.489382]

van Hook, T., 1986. Real-time Shaded NC Milling Display. Proceedings from the 13th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH'86. ACM Press, p.15-20. [doi:10.1145/15922.15887]

Yang, M., Lee, E., 1996. NC verification for wire-EDM using an r-map. *Computer Aided Design*, **28**(9):733-740. [doi:10.1016/0010-4485(95)00079-8]

Zwicker, M., Pfister, H., van Baar, J., Gross, M., 2001. Surface Splatting. Proceedings of SIGGRAPH'01, p.371-378.