



Optimal Quadrangulation of a Strip for Flank Milling

Wei-Lun Tsai^{1,2}, Charlie C. L. Wang¹, Chih-Hsing Chu² and Kai Tang³

¹The Chinese University of Hong Kong, cwang@mae.cuhk.edu.hk

²National Tsing Hua University, chchu@ie.nthu.edu.tw

³The Hong Kong University of Science and Technology, mektang@ust.hk

ABSTRACT

This paper presents a modeling algorithm that helps automatically generate a quadrangulation strip for flank milling which minimize the cutting error to a designed reference surface. Two curves are given as the boundary of a ruled surface to be machined. We discrete two curves into sample points and try to connect them to form quadrangles by rule lines, where these rule lines simulate the cutter's moving trajectory (i.e. tool path). Not only the surface error but also some other engineering constraints are considered in our optimal quadrangulation algorithm.

Keywords: quadrangulation, tool path, flank milling, graph.

DOI: 10.3722/cadaps.2008.307-315

1. INTRODUCTION

Flank milling is an important machining technique, especially in machining of turbine blades or propeller. For these applications, the performance of a designed product relies very much on the shape of surfaces. Therefore, it is extremely important to develop the strategies for flank milling that lead to a minimal error between the designed surface and the machined surface. There are many works in literature focuses on point milling of doubly curved surfaces [1]. Flank milling cuts with the shaft of the tool rather than milling with the tip of tool. The problem we are going to solve is as below.

Problem Definition For a designed ruled surface

$$S(u, v) = (1 - v)P_u(u) + vP_d(u),$$

$P_u(u)$ and $P_d(u)$ are two boundary curves, we need to compute a tool sliding along P_u and P_d to let that the error between the machined surface M and the designed surface S is minimized. It have to notice that our designed surface S was produced by some triangulation method that achieve some property [12], for instance, minimize the area of triangulation surface or the twist property of surface. The ruled surface would be approximated by several triangles as shown in Fig.1, and the two curves P_u and P_d are named as *generating rails*.

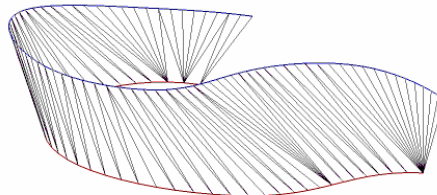


Fig. 1: Triangulation surface.

In this paper, we present a novel approach for the path planning of 5-axis flank milling for twisted ruled surface approximated by triangulation surface. Because of the nature of twisted ruled surfaces, difference always exists between the designed and machined surfaces. To machine a designed surface S , we assume that the machined surface M (simulated by the envelope surface) is also presented by a ruled surface which has the same boundary curves as S but different rule lines.

The rest of the paper is organized as follows. After reviewing the related works in section 2, the problems about using triangulation in path planning for flank milling will be discussed in section 3. The engineering constraints on quadrangulation for flank milling are presented in section 4. The global optimal quadrangulation is converted into a shortest path computation on a directed acyclic graph, which can integrate both the surface error minimization and the engineering constraints under the same framework. After giving several experimental results in section 5, our paper ends with the conclusion and discussion section.

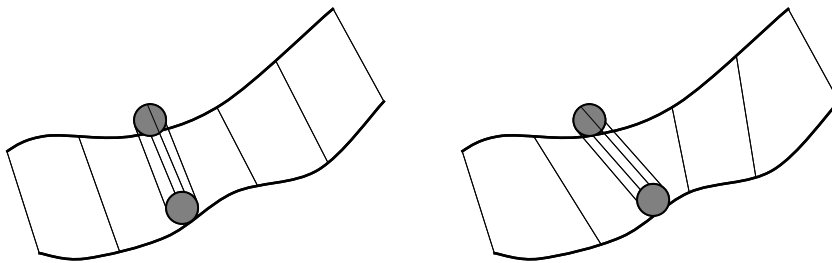


Fig. 2: Different envelope surfaces can be generated by the same generating rails.

2. RELATED WORKS

The research of 5-axis flank milling has fewer papers published in the literature than other machining related research. In [2], Tsay and Her used cutter radius, the length of rule line, and the normal vectors at the end points of rule line to approximate cutting error. The cutter axis was then determined according to the approximated cutting error. Tsay et al. also provided a method in [3] using coordinate transformation to adjust cutter axis, and a B-spline surface was machined to show their method's result.

Liu [4] presented a heuristic method that offsets two points on the rule line at parametric values of 0.25 and 0.75 along the surface normal by the cutter radius. The two offset points were then used to define the cutter axis orientation. However, this method always leads to significant cutting error.

Lartigue et al. [5] simulated the envelope surface of the tool trajectory and calculated the geometric deviation between the envelope surface and the designed surface. They employed the sum of distance between the discrete points on the envelop surface and the deigned surface along the surface normal as the cutting error between two surfaces. According to this cutting error, the axis of cutter was adjusted to make the new envelope surface more fit to the original surface, and then generated a new tool path to minimize the cutting error. The drawback of this approach is twofold: 1) the error evaluation is time-consuming and 2) the adjustment of cutter orientation is local – can hardly give a global optimum.

Robio et al. [6] presented a method by first aligning the tool axis along a rule line and then offsetting it. The error is later reduced by changing the orientation of cutter axis. Their work is further developed into the method presented in [7] that positioned the cylindrical cutter so that it touches the rule line and the two boundary curves. The method results in seven transcendental equations that should be solved for each position of cutter.

Menzel et al. [8] provided a method to let cutter tangentially touch two boundary curves and gave the accurate formula for computing the axis direction for this strategy. In order to reduce cutting error when flank milling, they provide method to adjust the orientation of cutter in their after research [9]. In [9], a three-step algorithm is developed to adjust the orientation of cutter to reduce the cutting error. Recently, several methods for error measurements have been compared in [10], and the author proposed using the error calculation as an approximation to the distance from the designed surface along the direction of the surface normal. The cutting error is reduced locally by this new error metric. However, all these approaches concern about the local optimum but not the global minimum cutting error.

Gong et al. [11] derived the proposition about the deviation at extremum point between the designed surface and the envelope surface of cylindrical cutter. A three-point offset strategy was proposed to approximate the offset surface, and a least-square approximation is conducted to further reduce cutting errors by deforming the tool axis trajectory surface. However, the deformation via least-square approximation does not guarantee the engineering constraints like the length of cutter, the variation limitation of velocity, etc. Furthermore, the deformation relies on a good initial trajectory which in general is not easy to give. This is still a local optimal solution.

Therefore, in this paper, we will discretize the problem by sampling both the generating rails with many sample points, and then compute an optimal quadrangulation on the points. As illustrated in Fig.2, different envelope surfaces can be generated on the same generating rails with different rule lines connecting the sample points. Different envelope surface leads to different tool paths for flank milling.

3. PROBLEMS OF TRIANGULATION FOR FLANK MILLING

The optimal strip triangulation method presented in [12] defined two operators: P -succeed and Q -succeed to advance the rule line from one side of two rails to the other side. During the advancement of rule line, new triangles are created. The solution based on triangulation works fine with some applications (e.g., the developable surface approximation [13, 14]). Nevertheless, it has several deficiencies when using the triangulation to simulate a ruled surface for flank milling.

When using the triangulation result to simulate a ruled surface, every triangle edge across the rails is considered as a rule line corresponding to a particular parameter (i.e., $w = w_0$ for the machined ruled surface $M(w, v)$). The result of triangulation will have several rule lines joining at one sample point at rails. For example, if there are three rule lines corresponding to $w = w_{i-1}$, $w = w_i$ and $w = w_{i+1}$ connecting to a same point $p \in P_u$, and we conduct the piecewise linear representation for the curve P_u , the first derivative of P_u computed at p by numerical difference will be zero at p .

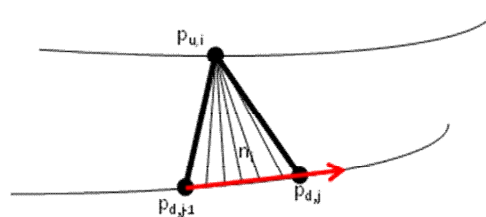


Fig. 3: Trajectory of milling tools by triangulating the generating rails.

In the flank milling, the first derivative of the generation rail relates to the velocity of cutter's ends. If the first derivative is zero, the velocity of cutters is also zero. In above example, the velocity of cutter at p is zero when $t = t_i$ (where the cutter is placed along rule line $P_u(w_i) - P_d(w_i)$ at $t = t_i$). The situation becomes even worse if more than three rule lines joining at p . If there are five rule lines that

$$P_u(w_{i-2}) = P_u(w_{i-1}) = P_u(w_i) = P_u(w_{i+1}) = P_u(w_{i+2}) = p,$$

the cutter is then stopped at p from the time t_{i-1} to t_{i+1} while the other end of cutter is still sliding along the generating rail P_d (as illustrated in Fig.2). This scenario of cutter movement has the following drawbacks:

- This brings difficulties to keep the cutter's movement smooth in the velocity (or angular velocity) and the acceleration (or angular acceleration);
- The velocity (or acceleration) jitter of cutter's movement may lead to unpredictable vibration that can damage the cutter;
- When letting the cutter static at one point on the machined surface for a long time, the machined surface will be damaged since the tool will keep rotating and removing material from the surface.

To overcome these deficiencies, we work out a novel quadrangulation algorithm in this paper to simulate a twisted ruled surface approximated by triangulation surface whose two rails have been sampled into m and n points. However, generally $m \neq n$, some sample points need be crossed without linking any rule line.

Definition 1 The sample points that are crossed during the quadrangulation of two rails are called *crossed points*, and the maximal allowed number of consecutive crossed points along one rail is named as the *crossable point number* (ab., *cp*).

The final quadrangulation can only be constructed when $cp \neq 0$. In this paper, we simply use $cp = 1$ in all our tests. Therefore, as shown in Fig.4, starting from the rule line $p_{u,i}p_{d,j}$, there are in total four possible ways to create the next quadrangle:

- *p*-succeed: moving forward along both P_u and P_d by one sample to the rule line $p_{u,i+1}p_{d,j+1}$;
- *s*-succeed: moving forward along P_u by one sample but along P_d by two samples to $p_{u,i+1}p_{d,j+2}$;
- *t*-succeed: moving along P_u by two samples but along P_d by only one sample to $p_{u,i+2}p_{d,j+1}$;
- *q*-succeed: moving along both P_u and P_d by two samples to the rule line $p_{u,i+2}p_{d,j+2}$.

Similarly, there are also four possible methods to reach this rule line $p_{u,i}p_{d,j}$. The problem now to be solved is that how to organize these operators optimally to result a quadrangulation M with minimal surface difference to the design surface S .

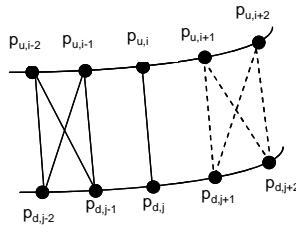


Fig. 4: The relationship between quadrangle and the four possible operators.

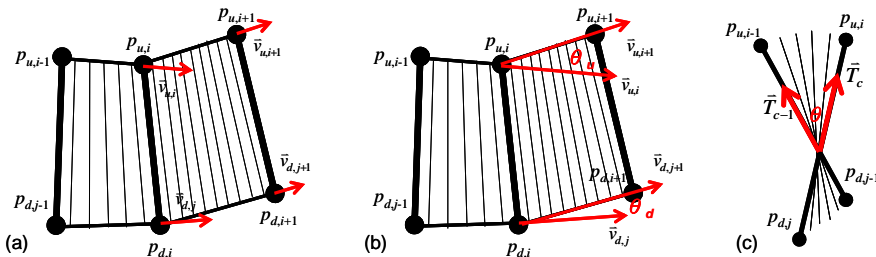


Fig. 5: Illustration for the constraints for flank milling: (a) the velocity difference constraint, (b) the velocity turning constraint, and (c) the rule line twist constraint.

4. CONSTRAINTS FOR FLANK MILLING

Four constraints for flank milling will be integrated into the optimal quadrangulation algorithm.

Cutter Length Constraint

First of all, the cutter length constraint is the most obvious but also an easy to forget constraint. We must examine the length between two sample points on different generating rails before connecting them by a rule line. This is because that the cutters in flank milling have their own length limitation. Note that here the length limitation means the length of working region on a cutter instead of its geometric length.

Velocity Difference Constraint

When the cutter is sliding along the generating rails, we can calculate the velocity of the cutter along the upper rail as

$$\bar{v}_{u,i} = \frac{p_{u,i} - p_{u,i-1}}{t}, \tag{1}$$

where t is the time for moving the cutter from one rule line to the next one (see Fig.5(a)). The velocity $\bar{v}_{d,j}$ along the lower rail can be computed in a similar way. Considering about the velocities at two consecutive sample point on the rail (e.g., $\bar{v}_{u,i}$ and $\bar{v}_{u,i+1}$), the relation between them can be approximated by

$$\bar{v}_{u,i} = \bar{v}_{u,i+1} + t\bar{a}_{u,i}. \quad (2)$$

If there is significant change between $\bar{v}_{u,i}$ and $\bar{v}_{u,i+1}$, a great acceleration (therefore force) should be given to the cutter. This should be avoided as it will generate unwanted vibration and easily damage both the cutter and the machined surface.

Thus, for the velocities at two consecutive sample point on a generating rail P_u , $\bar{v}_{u,i}$ and $\bar{v}_{u,i+1}$, the magnitude of their difference should not exceed a threshold δ_v , i.e., $\|\bar{v}_{u,i} - \bar{v}_{u,i+1}\| < \delta_v$. The velocities $\bar{v}_{d,j}$ and $\bar{v}_{d,j+1}$ on P_d should also satisfy $\|\bar{v}_{d,j} - \bar{v}_{d,j+1}\| < \delta_v$.

Velocity Turning Constraint

The above constraint controls the speed variation of cutter feeding; however, its control to the angular acceleration is implicit. To have a better control of angular acceleration, we define the following constraint on the turning of velocity vectors. For the velocities $\bar{v}_{u,i}$ and $\bar{v}_{u,i+1}$ on the generation rail P_u and $\bar{v}_{d,j}$ and $\bar{v}_{d,j+1}$ on P_d , the angle between them can be computed by (see the illustration in Fig.5(b))

$$\theta_u = \arccos \frac{\bar{v}_{u,i} \cdot \bar{v}_{u,i+1}}{\|\bar{v}_{u,i}\| \|\bar{v}_{u,i+1}\|}, \quad \theta_d = \arccos \frac{\bar{v}_{d,j} \cdot \bar{v}_{d,j+1}}{\|\bar{v}_{d,j}\| \|\bar{v}_{d,j+1}\|}. \quad (3)$$

Therefore, the turning of consecutive velocities along generating rails should be less than a threshold δ_θ , i.e.,

$$\arccos(\bar{v}_{u,i} \cdot \bar{v}_{u,i+1}) / (\|\bar{v}_{u,i}\| \|\bar{v}_{u,i+1}\|) < \delta_\theta \quad \text{and} \quad \arccos(\bar{v}_{d,j} \cdot \bar{v}_{d,j+1}) / (\|\bar{v}_{d,j}\| \|\bar{v}_{d,j+1}\|) < \delta_\theta. \quad (4)$$

Rule Line Twist Constraint

The last constraint we concerned is the twist inside a quadrangular patch. Quadrangulation of two sampled generating rails doesn't ensure that the neighboring rule lines are coplanar, so there will have twist between them as shown in Fig.5(c). Fig.5(c) is the side view of a patch, where \bar{T}_{c-1} and \bar{T}_c denote the direction vectors of two rule lines. Thus, the twist between them can be evaluated by the angle below.

$$\theta_t = \arccos \frac{\bar{T}_{c-1} \cdot \bar{T}_c}{\|\bar{T}_{c-1}\| \|\bar{T}_c\|} \quad (5)$$

During the movement of cutter from the rule line \bar{T}_{c-1} to \bar{T}_c , if $\theta_t \neq 0$ the cutter actually rotates around a center within the tool flank. Such a special motion may lead to significant machining error when the twist is large. In order to avoid this, the twist between two neighboring rule lines must be controlled.

Thus, the twisting angle θ_t between two neighboring rule lines must satisfy $\theta_t < \delta_t$ on all pairs of neighboring rule lines.

5. OPTIMAL QUADRANGULATION

In this section, the method about how to compute an optimal quadrangulation will be presented. The requirements from above four engineering constraints will also be integrated into the algorithm. Our basic idea is to convert the quadrangulation problem into a single-source shortest path problem on a weighted graph. The Dijkstra's algorithm [15] can then be utilized to obtain the shortest path which uniquely determines an ordered sequence of p -, s -, t -, or q -succeed operators that generate a global optimum. We start describe the construction algorithm for a graph Γ , which consists of following three steps.

Graph Construction:

Step 1) For every possible rule line $p_{u,i}p_{d,j}$, a node $V_{i,j}$ is constructed in Γ ;

Step 2) For every node $V_{i,j}$, several directed link $\langle V_{i,j}, V_{i+k,j+l} \rangle$ are established pointing from $V_{i,j}$ to $V_{i+k,j+l}$ with $k \in \{1,2\}$ and $l \in \{1,2\}$;

Step 3) Every directed edge in the graph is assigned a weight.

In step 1), $((m-2)(n-2)+2)$ nodes will be constructed in total. The reason why the number is not mn is that there are some graph nodes (linking to the end points of rails) cannot give any valid quadrangle – only triangles can be formed by them. In step 2), every node $V_{i,j}$ will in general have four directed links pointing out except the one near $V_{m,n}$.

By taking $V_{1,1}$ as source and $V_{m,n}$ as target, a shortest path h linking them can be determined by the well-known Dijkstra’s algorithm. The shortest path has the smallest summation of the weights of the links among all the possible path linking $V_{1,1}$ and $V_{m,n}$ in Γ . Considering about the problem of flank milling we are going to solve here, we wish to construct a quadrangulation M whose L^2 error to the designed surface S is minimized. Therefore, the weight defined on a link $\langle V_{i,j}, V_{i+k,j+l} \rangle$ pointing from $V_{i,j}$ to $V_{i+k,j+l}$ is the L^2 surface error between the quadrangle $P_{u,i+k}P_{u,i}P_{d,j}P_{d,j+l}$ and the surface S . The detail method about evaluating this error will be described below. Then, the shortest path determined by the Dijkstra’s algorithm gives the quadrangulation with minimal L^2 surface error to S . Note that, as the L^2 surface error depends on the area of quadrangles, it will not be affected by the number of quadrangles on the final quadrangulation.

Without loss of generality, we assume that the designed surface S has been tessellated into a piecewise linear triangular mesh surface. As S is a ruled surface, the vertices of triangles are either on the generating rail P_u or on the rail P_d (as the blue ones shown in Fig.6). The triangles on S have been stored in a list as $\{f_1, f_2, \dots, f_{m+n}\}$ in the order from one side of the ruled surface to the other side. A method is developed to compute the surface error between $P_{u,i+k}P_{u,i}P_{d,j}P_{d,j+l}$ and S by using the L^2 error definition given in [16]. For a given triangle f with three vertices (v_1, v_2, v_3) , if the shortest distances from (v_1, v_2, v_3) to the surface S are (d_1, d_2, d_3) , the L^2 error between f and S is calculated by

$$L^2(f, S) = \frac{1}{6}(d_1^2 + d_2^2 + d_3^2 + d_1d_2 + d_1d_3 + d_2d_3)|f| \tag{6}$$

where $|f|$ is the area of triangle f . To evaluate the L^2 error between $P_{u,i+k}P_{u,i}P_{d,j}P_{d,j+l}$ and S , we tessellate the quadrangle $P_{u,i+k}P_{u,i}P_{d,j}P_{d,j+l}$ into triangles uniformly (as illustrated in Fig.6). Then, the L^2 error between these triangles and S are computed by Eq.(6). The surface error between the quadrangle $Q = P_{u,i+k}P_{u,i}P_{d,j}P_{d,j+l}$ and S is then

$$L^2(Q, S) = \sum_{f \in Q} L^2(f, S) \tag{7}$$

This is the weight for the directed edge linking $V_{i,j}$ and $V_{i+k,j+l}$. Also, the summation of weights on the edges of a path from the source node to the target node is the L^2 surface error of its corresponding quadrangulation to S .

In here, we used a triangulation surface to approximate ruled surface in order to get L^2 error from quadrangles we concerned; however, we ignore the error that made from contiguous triangles to the ruled surface of the same rails. It likes to simulate a triangulation surface approximating the ruled surface by a quadrangulation surface.

The computation in Eq.(6) and (7) is trivial and straightforward. However, the search of (d_1, d_2, d_3) usually takes a long time. The designed surface S has been tessellated into a triangular mesh surface, thus the shortest distance from (v_1, v_2, v_3) to S can be obtained by checking the shortest distances from them to all triangles on S . To reduce the number of triangles to be searched, we first find that the triangles holding $P_{u,i}, P_{u,i+k}, P_{d,j}$ and $P_{d,j+l}$ as f_s, f_t, f_u and f_v . We assume that only the triangles whose indices between $\min\{s, t, u, v\}$ and $\max\{s, t, u, v\}$ are the ones possible to have the closest point to the triangles on $P_{u,i+k}P_{u,i}P_{d,j}P_{d,j+l}$. Thus, only these triangles are adopted to search the distances, (d_1, d_2, d_3) , from a triangle f of $P_{u,i+k}P_{u,i}P_{d,j}P_{d,j+l}$ to the designed surface S .

For the constraints listed in section 4, some evaluations relate not only to the one current rule line $P_{u,i}P_{d,j}$ but also to its previous rule line $P_{u,i-k}P_{d,j-l}$. Therefore, if we simply define one node $V_{i,j}$ in the graph Γ for every rule line candidate, it is hard to integrate the above constraints into our graph based algorithm for the optimal quadrangulation. To solve this problem, we decompose every node $V_{i,j}$ (except the source node and the target node) into four configuration-dependent nodes as: p -node, s -node, t -node and q -node, which indicate whether the rule line corresponding to this node is constructed by a p -succeed, an s -succeed, a t -succeed, or a q -succeed. By this

decomposition, the value of θ_i can be evaluated on each configuration-dependent node, and $\|\bar{v}_{u,i} - \bar{v}_{u,i+1}\|$, $\|\bar{v}_{d,j} - \bar{v}_{d,j+1}\|$, θ_u , and θ_d can be effectively computed on the directed edge pointing out from the configuration-dependent nodes. Fig.7 shows an example for such decomposition. Therefore, the optimal quadrangulation satisfying all constraints for flank milling can be computed on the graph through the shortest path algorithm.

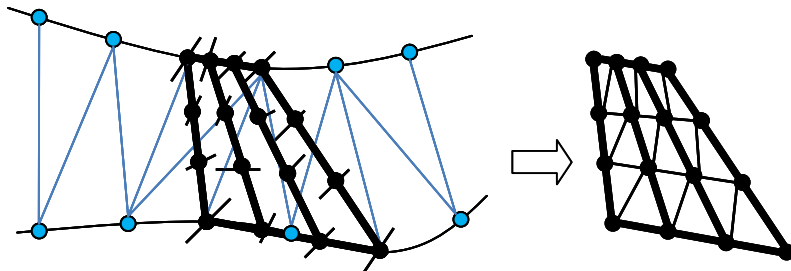


Fig. 6: To evaluate the surface error, the candidate quadrangle is tessellated into triangles.

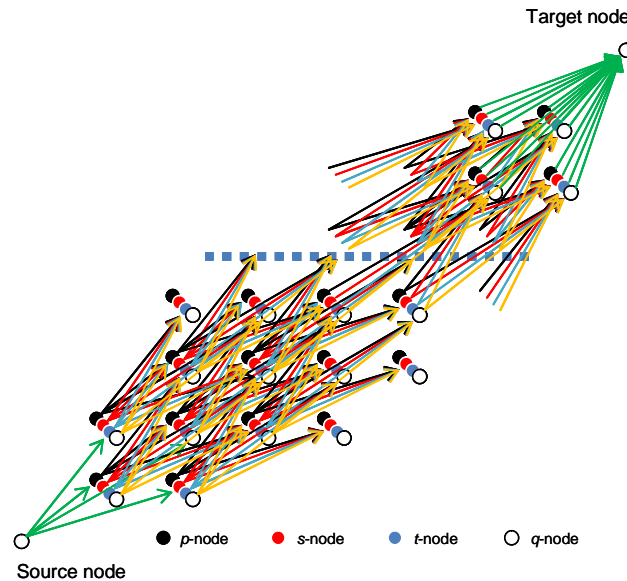


Fig. 7: An example graph with the configuration-dependent nodes.

6. EXPERIMENTAL RESULTS

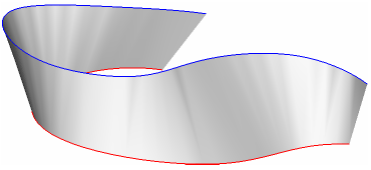
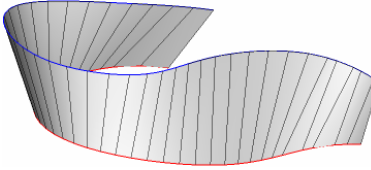
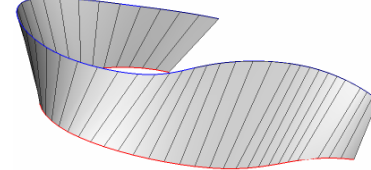
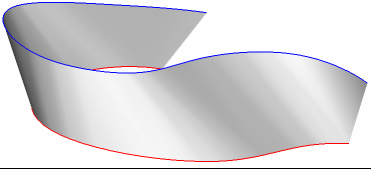
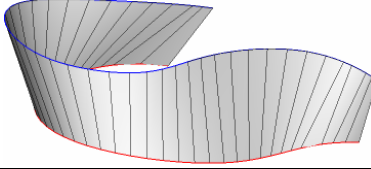
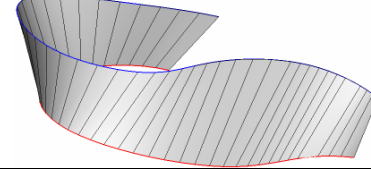
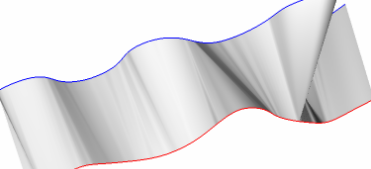
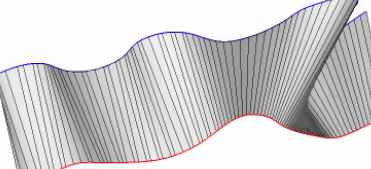
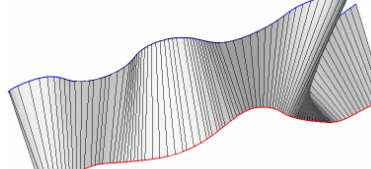
We have implemented the above algorithm by Visual C++ on a PC with Intel Core 2 T6600 CPU at 2.40GHz + 2GB RAM. The program has been tested on four designed ruled surface S_A , S_B , S_C and S_D , where S_A and S_B are two different designed surfaces on the same generating rails. Also S_C and S_D are on the same rails sampled into the same sets of points. The designed surfaces are given in the left column in Fig.8. The results from our optimal quadrangulation are listed in the middle column of Fig.8. In order to compare with other quadrangulations, we compute those quadrangulations with minimal area by weighting the directed edge in the graph Γ with the area of constructed quadrangular faces. The minimal area quadrangulations are listed in the right column of Fig.8. For comparison, the surface difference error in terms of L^2 is also listed. It is not difficult to find that the optimal quadrangulation for flank milling has less error than the minimal area quadrangulation.

7. CONCLUSION AND DISCUSSION

This paper presents a method that can generate an optimal quadrangulation to simulate the envelope surface of flank milling with minimal surface error between the machined surface and the designed surface. According to the different reference surfaces on the same generating rails, different quadrangulation can be found. Moreover, the engineering constraints have been integrated into the framework of quadrangulation by decomposing nodes into those configuration-dependent ones. In summary, our approach presented in this paper has the following advantages:

- Concerning about several engineering constraints that may happen, the constraints could be changed by different CNC machines employed – however, it is easy for our approach to vary to satisfy the change.
- Using all quadrangles instead of triangles so that avoid the problem which will occur in the strip triangulation.
- We convert the global optimal quadrangulation problem into a single source shortest path problem on a directed acyclic graph. The final optimal quadrangulation can be determined by the famous Dijkstra’s algorithm, which is a global optimum.

For the future research, there are several works to improve our method. In the error estimation part, we’ll change the error estimation method from the L^2 surface error into the direct distance from quadrangles to the ruled surface. Because our error estimation method now just let us to minimize the error to a triangulation surface but ignore the error from a triangulation surface to a ruled surface. In programming part, we are trying to employ the GPGPU (General-Purpose computation on GPUs) technique to accelerate the speed of evaluating the L^2 surface error between a quadrangle and the reference surface. GPGPU is a powerful technique in floating operation and parallel computing, through using GPGPU’s great computing power, we can not only parallel the computation of data set but also make the discrete points on quadrangle more density to more fit the real situation which could make the result better. Besides, we would compare our method to general flank milling tool path planning method, and even process a real experiment to confirm the attribute of our research. Lastly, we would try to add more engineering constraint and let $cp > 1$ to improve the flexibility of this method.

		
S_A	L^2 Surface Error: 1.029 Computing Time: 24.1(s)	L^2 Surface Error: 1.950
		
S_B	L^2 Surface Error: 0.4459 Computing Time: 45.5(s)	L^2 Surface Error: 8.498
		
S_C	L^2 Surface Error: 8.647 Computing Time: 23.4(s)	L^2 Surface Error: 10.79

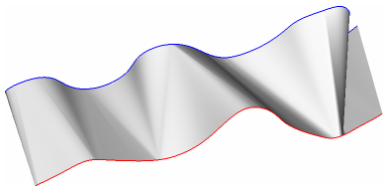
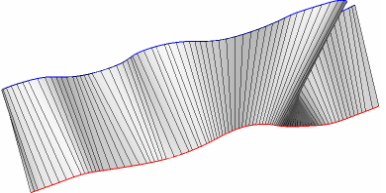
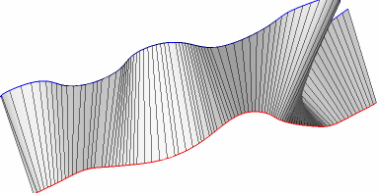
		
S_D	L^2 Surface Error: 20.93 Computing Time: 24.7(s)	L^2 Surface Error: 29.06
Designed Surface	Optimal Quadrangulation for Flank Milling	Minimal Area Quadrangulation

Fig. 8: Experimental results on different designed surfaces, where S_A and S_B are two different designed surfaces on the same generation rails. S_C and S_D are also on the same generating rails.

8. REFERENCES

- [1] Dragomatz, D.; Mann, S.: A classified bibliography of literature on NC mill path generation, *Computer-Aided Design*, 29(3), 1997, 239-247.
- [2] Tsay, D. M.; Her, M. J.: Accurate 5-axis machining of twisted ruled surfaces, *ASME Journal of Manufacturing Science and Engineering*, 123(4), 2001, 731-738.
- [3] Tsay, D. M.; Chen, H. C.; Her, M. J.: A study on five-axis flank machining of centrifugal compressor impellers, *ASME TURBO EXPO 2000*, Munich, Germany, 2000.
- [4] Liu, X.-W.: Five-axis NC cylindrical milling of sculptured surfaces, *Computer-Aided Design*, 27(12), 1995, 887-894.
- [5] Lartigue, C.; Duc, E.; Affouard, A.: Tool path deformation in 5-axis flank milling using envelop surface, *Compute-Aided Design*, 35(4), 2003, 375-382.
- [6] Rubio, W.; Lagarrigue, P.; Dessein, G.; Pastor, F.: Calculation of tool paths for a torus mill on free-form surfaces five-axis machines with detection and elimination of interference, *International Journal of Advanced Manufacturing Technology*, 14, 1998, 13-20.
- [7] Redonnet, J. M.; Rubio, W.; Dessein, G.: Side milling of ruled surfaces: optimum positioning of the milling cutter and calculation of interference, *International Journal of Advanced Manufacturing Technology*, 14, 1998, 459-465.
- [8] Menzel, C.; Bedi, S.; Mann, S.: Flank milling with flat end milling cutters, *Computer-Aided Design*, 35(3), 2003, 293-300.
- [9] Menzel, C.; Bedi, S.; Mann, S.: Triple tangent flank milling of ruled surfaces, *Computer-Aided Design*, 36(3), 2004, 289-296.
- [10] Li, C.; Mann, S.; Bedi, S.: Error measurements for flank milling, *Computer-Aided Design*, 37, 2005, 1459-1468.
- [11] Gong, H.; Cao, L. X.; Liu J.: Improved positioning of cylindrical cutter for flank milling ruled surfaces, *Computer-Aided Design*, 37, 2005, 1205-1213.
- [12] Wang, C. C. L.; Tang, K.: Optimal boundary triangulations of an interpolating ruled surface, *Journal of Computing and Information Science in Engineering*, ASME Transactions, 5(4), 2005, 291-301.
- [13] Wang, C. C. L.; Tang K.: Developable triangulations of a strip, *Computer-Aided Design and Applications*, 2(1-4), 2005, 233-242.
- [14] Chu, C.-H.; Wang, C. C. L.; Tsai C.-R.: Strip approximation using developable Bézier patches: a local optimization approach, *Computer-Aided Design and Applications*, 4(6), 2007, 807-816.
- [15] Cormen, T. H.; Leiserson, C. E.; Rivest, R. L.; Stein, C.: *Introduction to Algorithms*, 2nd ed., MIT Press, Cambridge, 2001.
- [16] Cohen-Steiner, D.; Desbrun, M.; Alliez, P.: Variational shape approximation, *ACM Transactions on Graphics*, 23(3), 2004, 905-914.