scientific reports

OPEN

An explicit and rapid intersection algorithm of plane and NURBS surface in CNC surface machining

Shengli Wei✉, Kai Zhao, Huaiping Yan & Yuan Li

To address the challenge of plane-surface intersection in toolpath planning for CNC surface machining, a rapid intersection algorithm between a plane and a NURBS surface is proposed, which can provide an explicit Non-Uniform Rational B-Spline (NURBS) description of the intersection curve. By leveraging the dual-parameter property of NURBS surfaces, a series of fixed parameter values (usually values in the knot vector of this parameter) in one parametric direction are selected to generate NURBS curves in the other parametric direction. The bisection method is employed to compute the intersection points of these curves and the plane. Subsequently, the reverse engineering algorithm is applied to construct a NURBS curve that passes through these points, representing the desired intersection curve. The proposed algorithm significantly improves computational efficiency by reducing the problem to one of calculating the intersections of multiple NURBS curves and the plane. Furthermore, the explicit NURBS representation of the intersection curve facilitates subsequent NURBS interpolation. Simulations and experimental results validate the algorithm's effectiveness and accuracy, demonstrating its capability of deriving the intersection expression.

Keywords Intersection algorithm, Explicit expression, Reverse engineering, Bisection method, Non-uniform rational b-splines (NURBS)

In CNC surface machining, the sectioning method is widely adopted for generating machining trajectories by intersecting a series of planes with the target surface[1]. Rapid computation of intersection curves is critical for this process. While algebraic methods yield exact solutions for analytically defined surfaces, parametric surfaces such as NURBS (Non-Uniform Rational B-Splines) pose significant challenges due to their inherent complexity. Traditional approaches, including tracing methods, bounding box techniques, and triangular mesh approximations[2–9], often require extensive computational resources or fail to provide explicit NURBS descriptions of the intersection curves. In recent years, new progress has been made in the research of intersection algorithms. Recent advancements in intersection algorithms include adaptive step-size tracing[2], subdivision surface intersection[3], and point-cloud-based methods[4]. Researchers also studied the intersection of surfaces with unique shapes. Liu Xiaoming used a method based on a toroidal surface approximation to study the toroidal intersection algorithm[10]. Qian Fu et al. studied the intersection of ball B-spline curves and proposed an intersection method for ball B-spline curves[11].

However, these techniques either rely on dense sampling of surface points or approximate representations, limiting their practicality in NURBS-based CNC systems. This paper proposes a bisection-based algorithm combined with reverse engineering to derive NURBS intersection curves, enhancing both computational efficiency and compatibility with downstream NURBS interpolation processes.

Intersection of a plane and a NURBS curve

NURBS curves are expressed in the form of parameters, and it is very challenging to solve the intersection issue analytically. In practice, approximate numerical methods are usually deployed to address this issue. The most widely used methods include the bisection method, tracing method, bounding box technique, etc. Furthermore, other algorithms are used to solve the intersection issue. Rao Daojuan et al. introduced the particle swarm optimization algorithm into the intersection of NURBS curves and proposed a NURBS curve intersection algorithm based on Particle Swarm Optimization (PSO)[12]. Delint Ira Setyo Adi et al. studied the NURBS fitting by using the particle swarm optimization algorithm, which provided a basis for the NURBS curve intersection with the particle swarm optimization algorithm[13]. Jingjing Shen et al. used the matrix representation method to study the intersection of straight lines and cropped NURBS surfaces[14].

School of Computer Science and Information Engineering, Anyang Institute of Technology, Anyang, Henan, China.
✉email: weishengli401@126.com

The above research methods are highly versatile and robust but very computation-consuming. However, in the scenario of CNC surface machining, the two ends of the tool path are usually located on both sides of the plane, and there is usually only one intersection point between the tool path curve and the plane. In this case, the bisection method is highly efficient and converges at the rate of 1/2. After 20 iterations, an accuracy of 10^{-6} can be achieved. Here, the bisection method is used to solve the intersection of a curve and a plane.

A NURBS curve is defined as[15]:

$$\boldsymbol{P}(u) = \frac{\sum_{i=0}^{n} N_{i,p}(u)\,\mathbf{V}_i W_i}{\sum_{i=0}^{n} N_{i,p}(u)\,W_i} \tag{1}$$

where $N_{i,p}(u)$ is the B-spline basis function, $\boldsymbol{V}_i$ are the control points, W_i are the weights, u is the parameter, which is generally in the range of [0,1].

Assuming one end of the curve is $\boldsymbol{P}(0)$ and the other is $\boldsymbol{P}(1)$, and the curve and plane have only one intersection point, the bisection method can be used to approximate the intersection point.

For a plane defined by $\boldsymbol{N}\cdot\boldsymbol{X} + \mathrm{D} = 0$, where A is the plane normal vector, X (x, y, z) is the position, and D is the intercept. The bisection method is applied to locate the intersection point when the curve spans both sides of the plane. The algorithm proceeds as follows:

```
a = 0
b = 1
if ($\boldsymbol{N}\bullet\boldsymbol{P}(a)$+D)* ($\boldsymbol{N}\bullet\boldsymbol{P}(b)$+D) <0
    while (b − a)/2 > TOL1
        c = (a + b)/2
        if ($\boldsymbol{N}\bullet\boldsymbol{P}(c)$+D) < TOL2, stop, end
        if ($\boldsymbol{N}\bullet\boldsymbol{P}(a)$+D)* ($\boldsymbol{N}\bullet\boldsymbol{P}(c)$+D)<0
            b = c
        else
            a = c
      end
  end
```

The bisection method allows us to quickly locate the intersection point of the NURBS curve and the plane. Figure 1 is a schematic diagram of the intersection of a NURBS curve and a plane using the bisection method. For cases when the curve doesn't span both sides of the plane, the piecewise bisection method, tracking method, or box bonding method can be used to solve the problem. However, as mentioned above, in the surface machining scenario, the two ends of the curve are usually located on both sides of the plane with only one intersection point. In most situations, the bisection method can be applied to find the intersection point directly.

Intersection of a plane with multiple NURBS curves

A NURBS surface is expressed as[15]

$$\boldsymbol{s}(u,v) = \frac{\sum_{i=0}^{n}\sum_{j=0}^{m} N_{i,p}(u)\,N_{j,q}(v)\,W_{i,j}\mathbf{P}_{i,j}}{\sum_{i=0}^{n}\sum_{j=0}^{m} N_{i,p}(u)\,N_{j,q}(v)\,W_{i,j}} \tag{2}$$

where $N_{i,p}(u)$ and $N_{j,q}(v)$ are the B-spline basis functions of the surface parameters u and v respectively, $\boldsymbol{P}_{i,j}$ are the control points, $W_{i,j}$ are the weights. The two knot vectors of the two parameters are

$$\boldsymbol{U} = \left[\underbrace{0{,}0,\,0{,}0}_{p+1}, u_0, \cdots, u_{n-p-1}, \underbrace{1{,}1,\,1{,}1}_{p+1}\right]$$

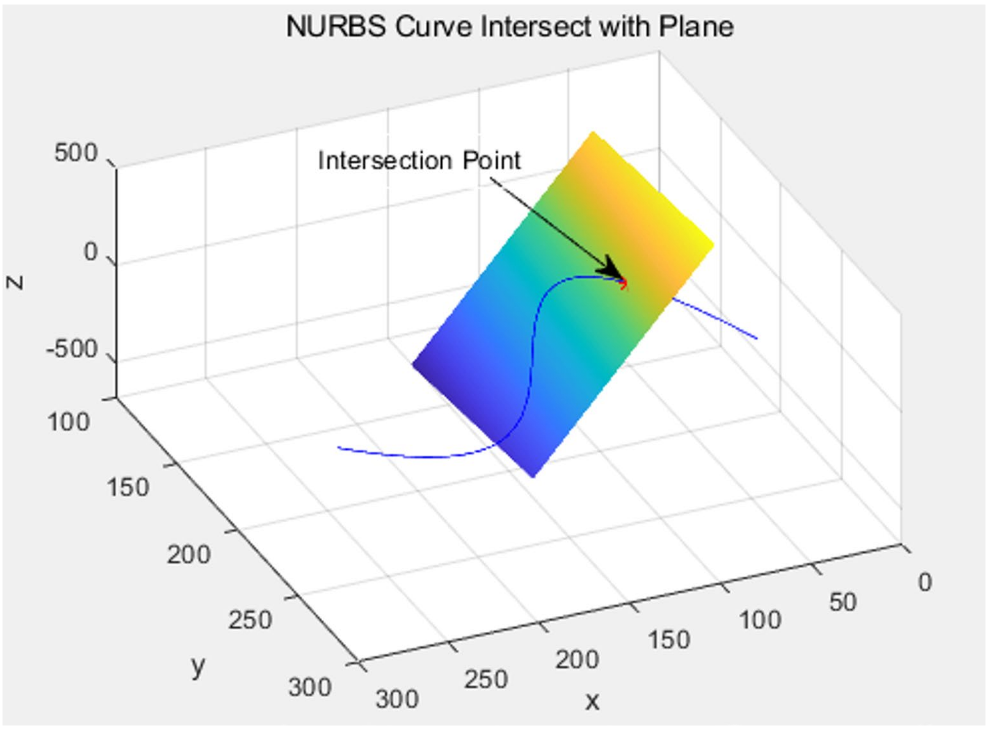

Fig. 1. Intersection of a NURBS curve with a plane.

$$\boldsymbol{V} = \left[\underbrace{0{,}0,0{,}0}_{q+1}, v_0, \cdots, v_{m-q-1}, \underbrace{1{,}1,1{,}1}_{q+1}\right]$$

.

NURBS surfaces can be rewritten as

$$\boldsymbol{s}\left(u,v\right) = \frac{\sum_{i=0}^{n} N_{i,p}\left(u\right) \sum_{j=0}^{m} N_{j,q}(v)\, W_{i,j} \boldsymbol{P}_{i,j}}{\sum_{i=0}^{n} N_{i,p}\left(u\right) \sum_{j=0}^{m} N_{j,q}(v)\, W_{i,j}} \tag{3}$$

By fixing one parameter (e.g. $v = v_k$), Eq. (3) can be rewritten as Eq. (4), which represents a NURBS curve.

$$\boldsymbol{s}\left(u,v_k\right) = \frac{\sum_{i=0}^{n} N_{i,p}\left(u\right) \boldsymbol{P}_{v_k}}{\sum_{i=0}^{n} N_{i,p}\left(u\right) W_{v_k}} \tag{4}$$

where

$$\boldsymbol{P}_{v_k} = \sum_{j=0}^{m} N_{j,q}(v_k)\, W_{i,j} \boldsymbol{P}_{i,j}$$

$$W_{v_k} = \sum_{j=0}^{m} N_{j,q}(v_k)\, W_{i,j}$$

.

By fixing the parameter v with a series of fixed values (here $v = [0, v_0, \cdots, v_{m-q-1}, 1]$ as it is in the knot vector), a family of NURBS curves in the u-direction is generated. The bisection method is used to compute the intersection points of the plane and this family of curves, just as discussed in "Intersection of a plane and a NURBS curve". Thus, $m - q + 2$ NURBS curves are intersecting with the plane, and $m - q + 2$ intersection points will be found. Figure 2 demonstrates the intersection of multiple curves with the plane.

Find the intersection curve reversely

Based on the reverse engineering and NURBS theory, the control points of a NURBS curve going through all the $m - q + 2$ intersection points will be computed reversely. According to the NURBS curve expression, without considering the weights, there is

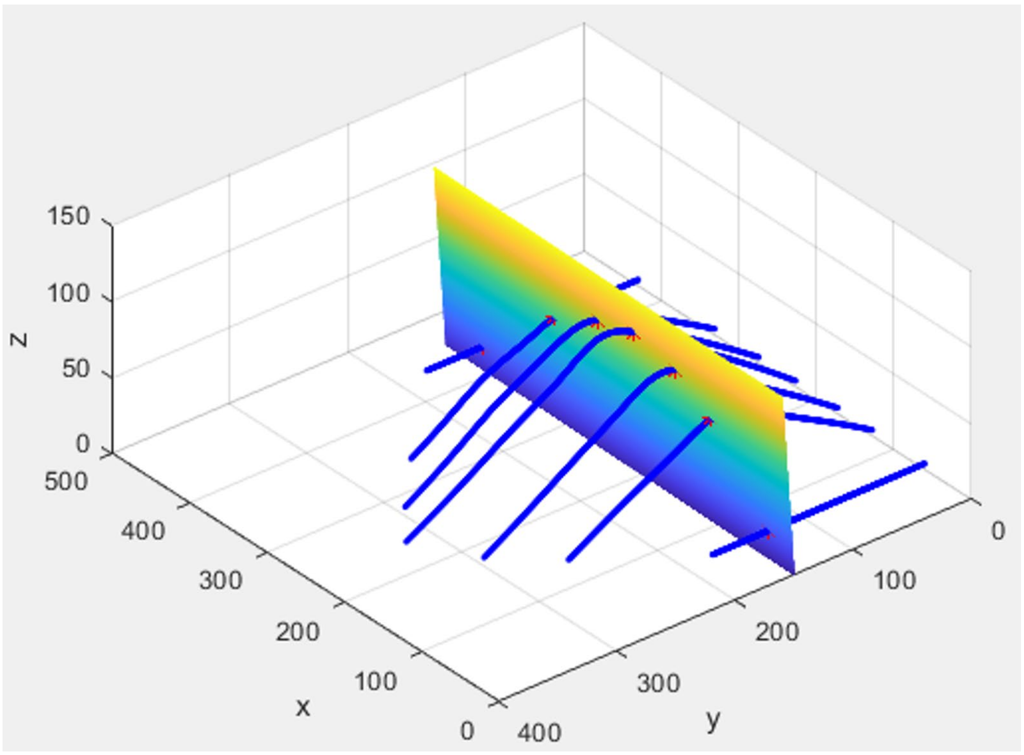

Fig. 2. Intersection of multiple curves with the plane.

$$\sum {}_{j=0}^{m} N_{j,q}\left(v\right) \boldsymbol{V}_j = \boldsymbol{P_j} \tag{5}$$

where $\boldsymbol{P_j}$ are the data points through which we want the generated curve goes. In this case, $\boldsymbol{P_j}$ include the intersection points and the complementary points which are determined by the boundary conditions. $\boldsymbol{V}_j$ are the required control points that will generate the NURBS curve.

Due to the rational properties and non-uniformity of NURBS, NURBS interpolation is extremely difficult even impossible, and here it is first simplified as a B-spline interpolation problem. The error caused by this simplification can be solved later through additional interpolation as discussed in the "Algorithm verification". According to[16], Eq. (5) can be rewritten in matrix form as follows

$$\begin{bmatrix} b_0 & c_0 & & & & & & \\ 1/4 & 7/12 & 1/6 & & & & & \\ & 1/6 & 2/3 & 1/6 & & & & \\ & & \cdots & \cdots & \cdots & & & \\ & & & \cdots & \cdots & \cdots & & \\ & & & & 1/6 & 2/3 & 1/6 & \\ & & & & & 1/6 & 7/12 & 1/4 \\ & & & & & & a_m & b_m \end{bmatrix} \begin{bmatrix} V_0 \\ V_1 \\ V_2 \\ \cdots \\ V_{m-2} \\ V_{m-1} \\ V_m \end{bmatrix} = \begin{bmatrix} d_0 \\ P_0 \\ P_1 \\ \cdots \\ P_{m-3} \\ P_{m-2} \\ d_{m-1} \end{bmatrix} \tag{6}$$

For cubic NURBS curves, the following boundary conditions are added to Eq. (6).

$$b_0 = 9\ c_0 = -3\ d_0 = 6P_0$$

$$a_m = -3\ b_m = 9\ d_{m-1} = 6P_{m-2}$$

In Eq. (6), the matrix is a tridiagonal matrix that can be solved using the pursuit method. Therefore, control points are computed by Eq. (6) reversely. The NURBS curve generated by these control points will go through the intersection points. Figure 3 illustrates the generated NURBS curve. The generated curve is used to approximate the actual intersection curve of the NURBS surface and the plane. Figure 4 demonstrates that the generated curve fits the actual intersection curve nearly perfectly. In the CNC surface machining process, this generated curve can be used as the tool path.

To strengthen the effectiveness of the algorithm, we have tested more NURBS surfaces with the proposed algorithm. Two of these surface are Figs. 5 and 6.

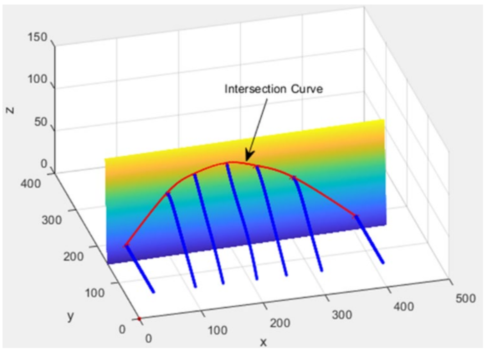

Fig. 3. Intersection curve generated by the control points.

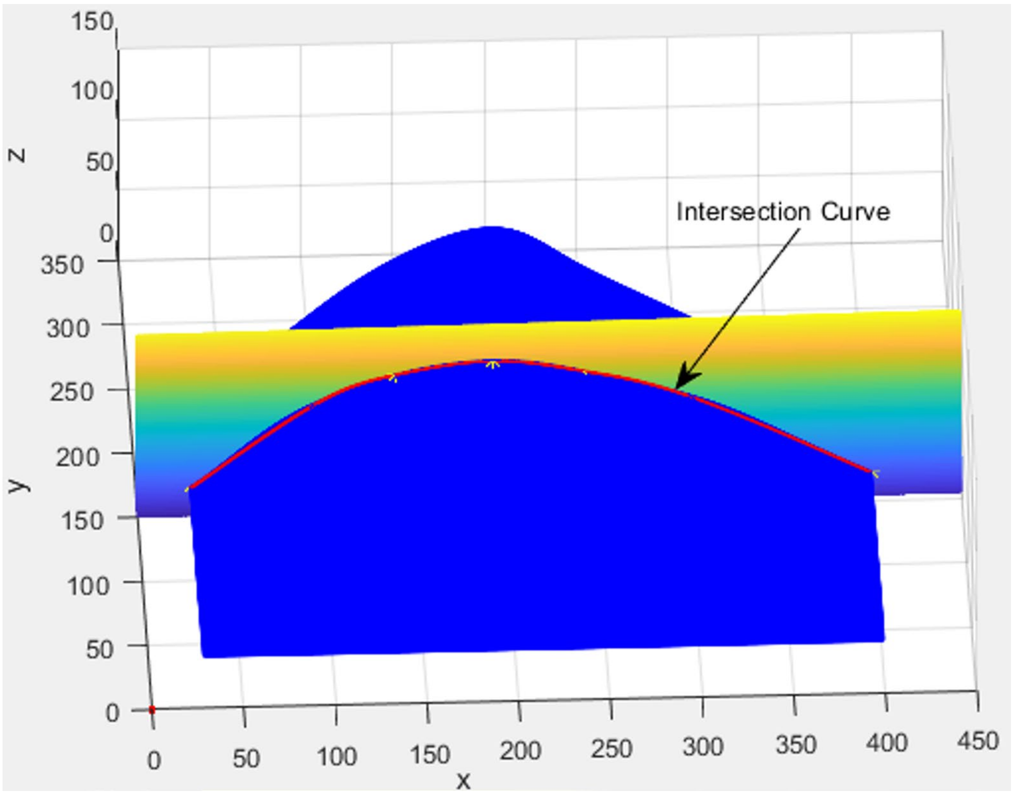

Fig. 4. The generated curve can fit the actual intersection curve well.

Algorithm verification

Since the intersection curve is a curve coming from the NURBS surface intersecting with the plane, the curve must be a planar curve, and it must be on the plane. That is, the intersection points and the control points obtained reversely must be on the plane, too. To verify the correctness and accuracy of the proposed algorithm, first, we should verify if the intersection points and the control points obtained reversely satisfy the plane expression. In one of the experiments, the cubic NURBS surface has $9\times9=81$ control points. Each of the two parameters u and v has $9-3=6$ segments. The expression of the plane intersecting with the NURBS surface is $0.01\mathrm{x}+12\mathrm{y}-\mathrm{z}-1800=0$. Usually, the projection of the control points of the NURBS surface onto one coordinate plane, i.e., the x-y plane of the coordinate system, is in a rectangular area. The reason that $0.01\mathrm{x}+12\mathrm{y}-\mathrm{z}-1800=0$ is chosen as the expression of the plane is that we want to ensure that the plane is as parallel as possible to two opposite edges of the projection rectangle and perpendicular to the other two opposite edges (For the sake of simplicity, we will refer to it as ideal intersection later). In this situation, the generated curve used to approximate the intersection curve is closest to the intersection curve. Otherwise, if the plane is far from parallel and perpendicular to the edges of the rectangle, the approximation is not satisfying, and other improvement measurements should be taken, as discussed later. By fixing the parameter v as it is in

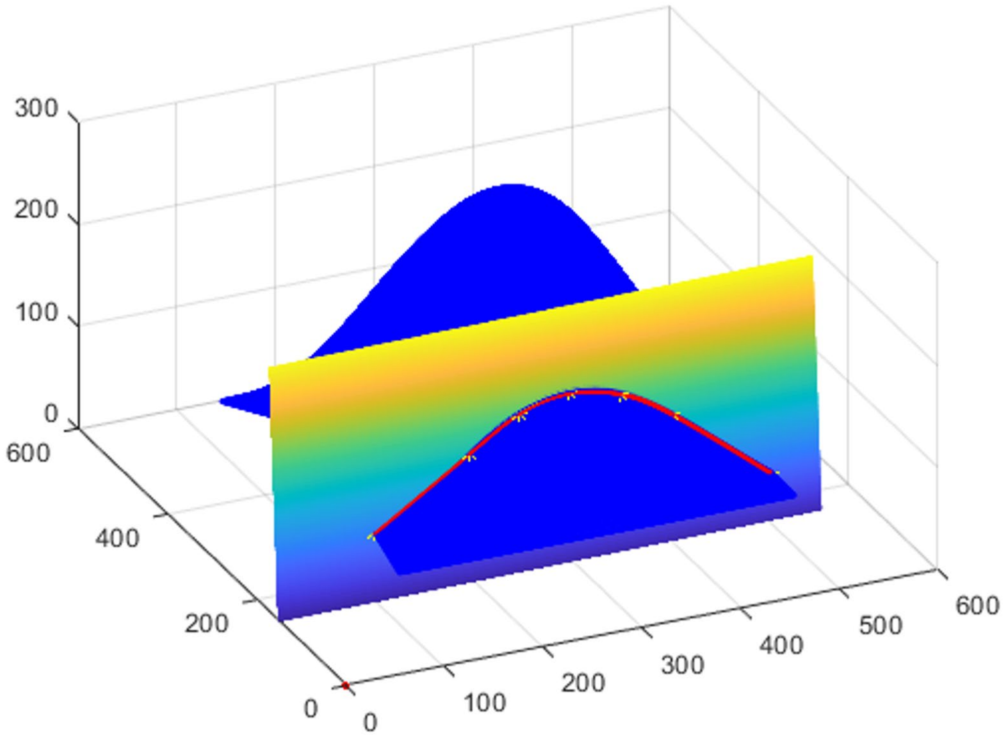

Fig. 5. NURBS surface and plane intersection test 1.

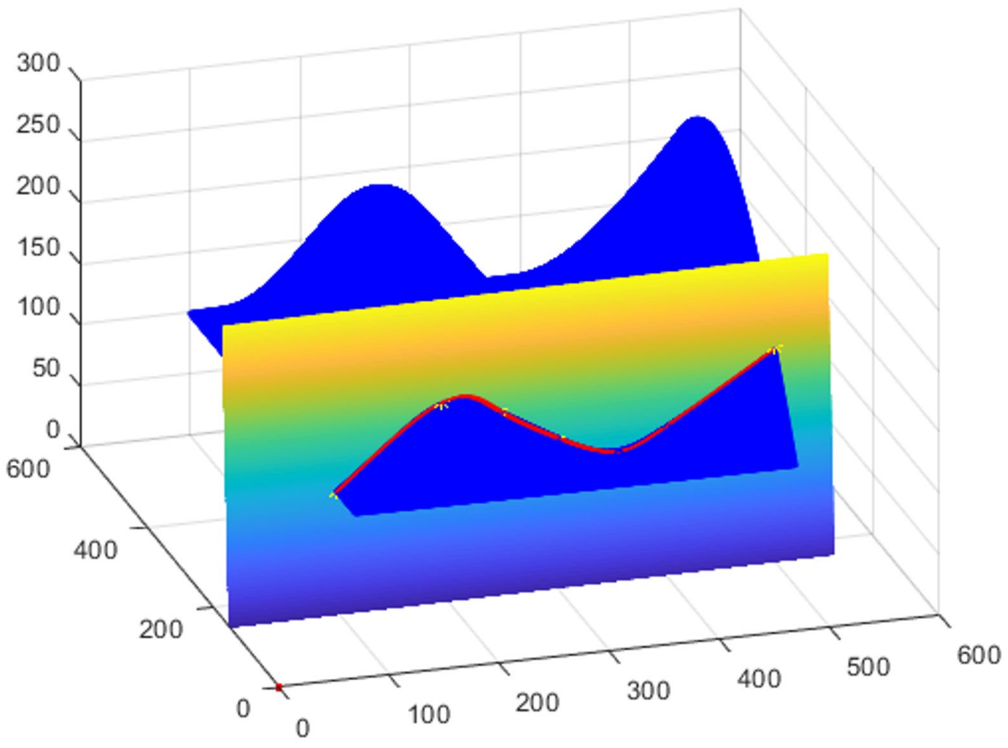

Fig. 6. NURBS surface and plane intersection test 2.

No.	Intersection points	Error (*10^{-4})
1	30.000000 151.641674 20.000000	0.880000
2	97.602938 155.997906 72.950913	− 0.116200
3	141.938504 157.488822 91.285298	− 0.489600
4	194.400707 158.088789 99.009415	0.600700
5	242.284335 157.403372 91.263243	0.643500
6	300.780479 155.831411 72.984723	0.137900
7	400.000000 151.333340 20.000000	0.800000

Table 1. Intersection points verification.

No.	Control points	Error (*10^{-4})
1	30.000000 151.641674 20.000000	0.880000
2	54.837038 153.336323 40.584198	0.483800
3	104.511115 156.725622 81.752593	− 0.178500
4	137.573169 157.443274 90.695106	− 0.863100
5	196.827231 158.434216 103.178772	0.923100
6	241.522149 157.352595 90.646294	0.674900
7	290.790181 156.575633 81.815511	− 0.131900
8	363.596727 153.080771 40.605170	0.492700
9	400.000000 151.333340 20.000000	0.800000

Table 2. Control points verification.

its knot vector, seven u-direction NURBS curves are formed, which are used to intersect with the plane to get seven intersection points. Then, nine control points are computed reversely.

Tables 1 and 2 show that the intersection points and the control points are on the plane, and the errors meet the error control requirement (error requirement being 10^{-4}).

To further verify the proposed algorithm, the actual intersection curve is compared with the generated curve, as shown in Fig. 5. In the test, the actual intersection curve is obtained via dense sampling. As can be seen from Fig. 7, the two curves have a high degree of agreement. The RMSE (Root Mean Square Error) between the two curves is only about 1.87.

The above results show that the algorithm is effective and has high accuracy when the plane is nearly parallel to the two opposite edges of the projection rectangle and perpendicular to the other two opposite edges (ideal intersection). If this is not the case, there will be obvious errors between the generated and actual curves, as shown in Fig. 8. The RMSE is 9.23, and it is obviously larger than that of the ideal intersection in which the RMSE is 1.87. This obvious error is mainly because the original knot vector of the NURBS surface is used as the knot vector of the generated curve in the proposed algorithm. This is a little unreasonable, but the purpose of doing so is to significantly reduce the complexity of the calculations. When a plane ideally intersects with a NURBS surface, the proposed algorithm brings about an ideal effect. However, when the plane doesn't intersect with the surface ideally, the proposed algorithm will cause large errors. To address this problem is quite complex, and some studies have used complex algorithms such as particle swarm optimization (PSO) to solve this problem. However, the CNC system has real-time requirements, and complex algorithms cannot meet this requirement. At the same time, the increase in accuracy brought about by these complex algorithms is not always satisfactory. According to reference[12] and our estimation, it takes about 20 ms to find just one intersection point of a NURBS curve when PSO algorithm is used. This is unacceptable for a CNC system. In the proposed algorithm, a NURBS calculation takes only several nanoseconds, and the total cost of finding the intersection curve is about several microseconds. Furthermore, for the PSO algorithm, the computational cost will increase exponentially if high accuracy is required. On the other hand, the computational cost of the proposed algorithm will only increase at a square root rate.

To improve the accuracy with the least increase in computational complexity, a strategy adopted here is to interpolate the generated curve so as to improve the intersection effect. By interpolating the generated curve, the shape of the curve can be controlled more flexibly, and the generated curve can fit the actual intersection curve better. In general, large errors are prone to occur at the place where the curvature of the intersection curve is large and at both ends of the intersection curve, so it is important to interpolate at these places. With curve interpolation, the accuracy of the approximation is significantly improved, and the RMSE reduces from 9.23 to 2.23. Figure 9 shows a comparison of the improved intersection curve after interpolation and the actual intersection curve, and it can be seen that the two curves have a high degree of agreement. Comparing Figs. 8 and 10, one can see that the accuracy has improved significantly.

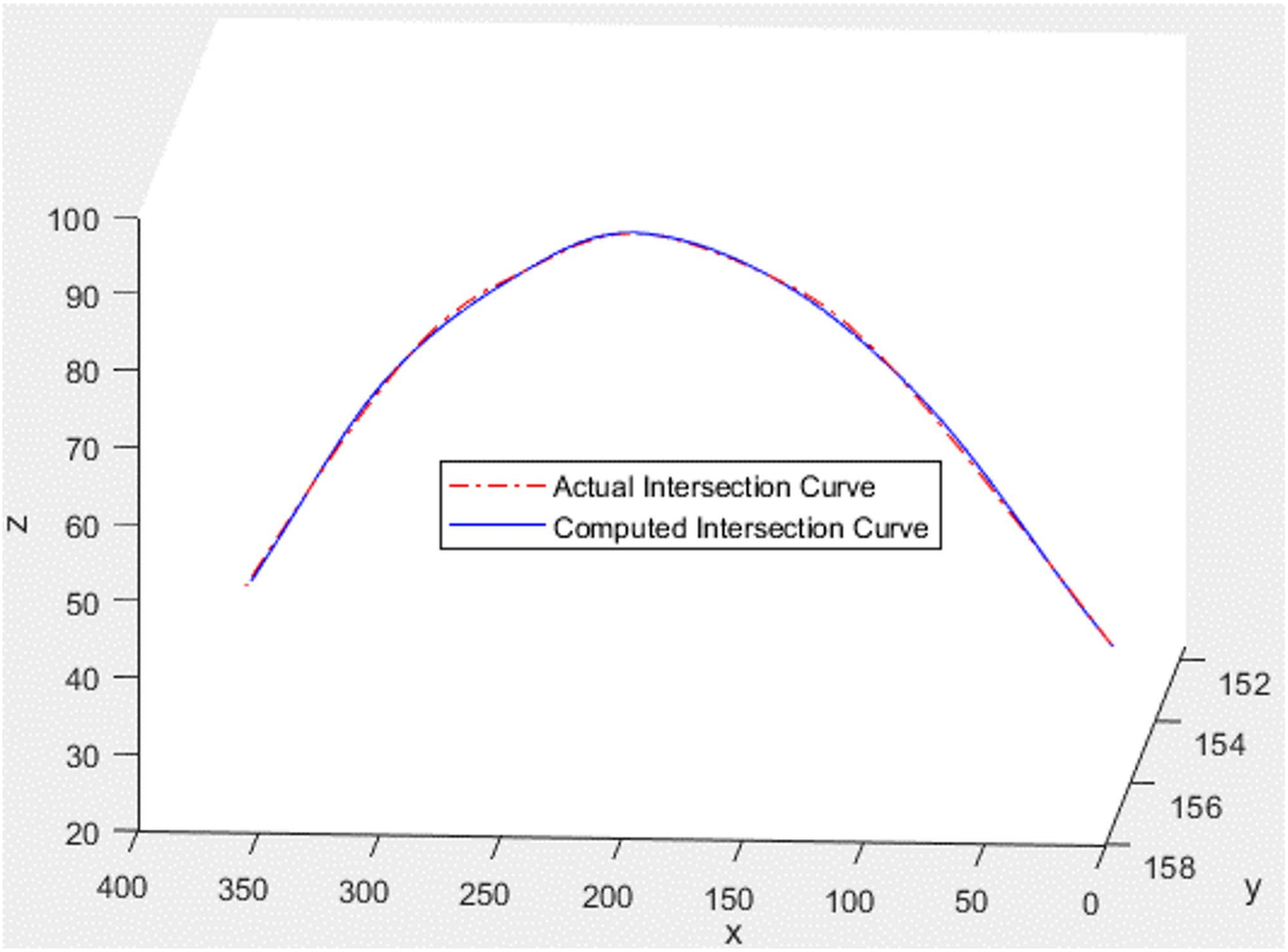

Fig. 7. Comparison of the actual intersection curve and the generated curve.

Conclusion

The proposed algorithm can find the intersection curve of a plane and a NURBS surface rapidly and give the NURBS description of the intersection curve, which can be used as a tool path in the CNC surface machining process. The NURBS description is convenient for storing and transmitting the tool path in the subsequent machining process in which the NURBS curve interpolation algorithm is usually needed. The explicit NURBS description benefits a lot to the NURBS interpolation algorithm. The proposed algorithm only needs to solve the intersection points of several NURBS curves with the plane. In the abovementioned experiment, 7 NURBS curves with 9 control points in each curve are first obtained, and only 63 NURBS calculations are required. When the control accuracy is 10^{-6}, it takes about 140 NURBS calculations to find the intersection points of the 7 curves and the plane with the bisection method. Equation (6) can be solved using the Thomas algorithm. The computational cost of the reverse engineering is about $5 * n$ floating-point operations, where n is the order of the matrix in Eq. (6). Its computational cost is relatively small compared to that of the NURBS, and so the computational cost of the reverse engineering can be neglected. Without considering the computation of the reverse engineering, the total computation is just a little more than 200 NURBS calculations. On the contrary, the traditional triangular patch approximation algorithm that uses a large number of small triangular patches to fit the original surface may require thousands or even hundreds of thousands of NURBS calculations to form triangular patches. Also, the traditional triangular patch approximation algorithm needs to judge whether each of these massive patches intersects with the plane, which is very computation-consuming. Therefore, the proposed algorithm has high efficiency. This is of great significance for CNC systems with high real-time requirements. The shortcoming of the proposed algorithm is that it needs ideal intersection conditions, just as described previously. However, in the path planning process of surface machining, people have opportunities to set how the plane intersects with the NURBS surface. In some special situations where the ideal intersection condition is not satisfied, curve interpolation techniques can be adopted to reduce the approximation error.

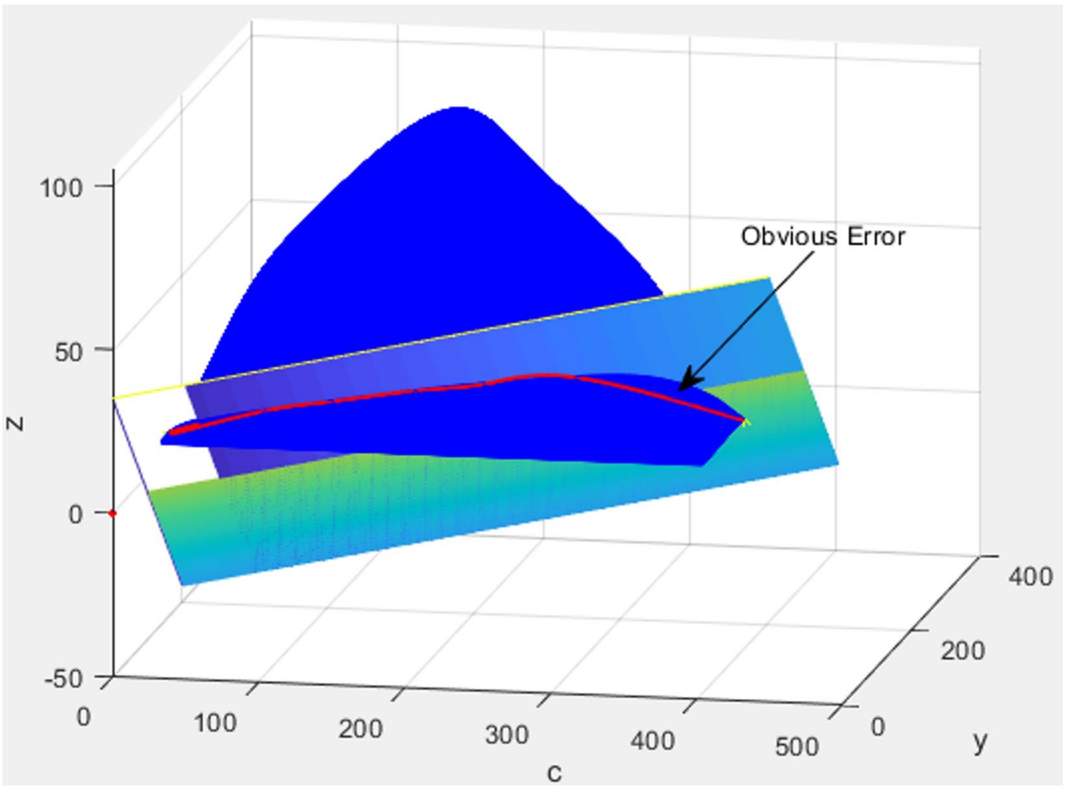

Fig. 8. Obvious error occurs in non-ideal intersection.

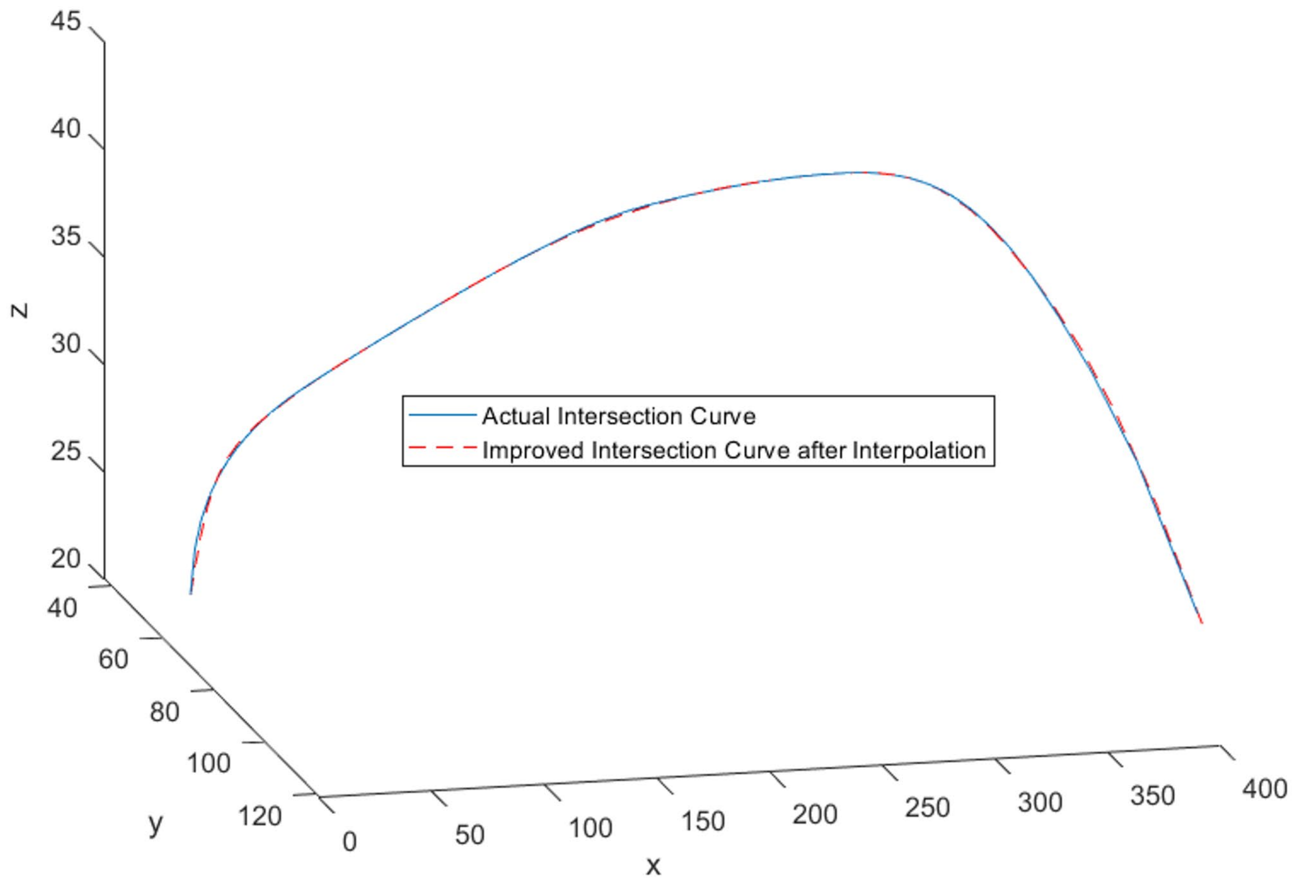

Fig. 9. The improved intersection curve after interpolation fits the actual intersection curve well.

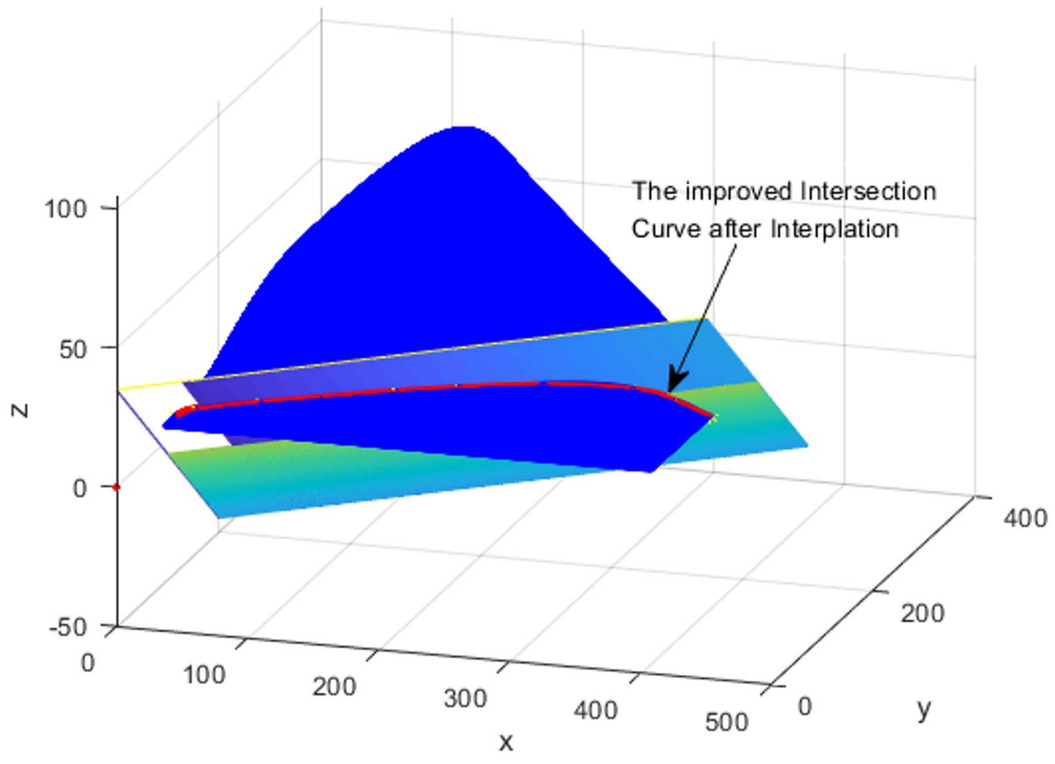

Fig. 10. The improved intersection curve after interpolation is fairly accurate.

Data availability

The data, code, and algorithm that support the findings of this study are available from the corresponding author, Shengli WEI, weishengli401@126.com, upon reasonable request. Alternatively, they are available in the repository: https://github.com/aygxywlw/Surface-Intersection, DOI: https://doi.org/10.5281/zenodo.15278443.

Received: 17 April 2025; Accepted: 24 October 2025
Published online: 25 November 2025

References

1. Xu, D. D. *Research on Tool Path Optimization of Five Axis NC Machining Based on Equal Residual Height Method* 3 (Shenyang Jianzhu University, 2020).
2. Liu, P. & He, X. M. Research on surface intersection technology based on step-size adaptive tracking method. *Comput. Eng. Appl.* **56** (21), 253–259 (2020).
3. Zhu, J. N. et al. Efficient algorithm for computing intersection curve between plane and subdivision. *Comput. Integr. Manuf. Syst.*, **20**(6), 1322–1329.
4. Zheng, P. F. et al. Intersection algorithm of point cloud surface by Spatial mesh and refinement. *J. Zhejiang Univ. (Eng. Sci.)* **52**(3), 605–612.
5. Zhang, J. X. & Wu, J. Intersection algorithm for discrete and complex surfaces of spare parts in engineering machine. *J. Chang. Univ. (Natural Sci. Ed.)*. **27**(5), 116–119.
6. Shi, Y. F. et al. Surface segmentation algorithm based on Spatial polygon triangulation. *J. Graph.* **40**(3), 447–451.
7. Wang, C. et al. An improved algorithm for the intersection of NURBS surface and implicit surface and its application to hull surface intersection. *Shipbuild. China*. **54**(3), 43–49.
8. Song, H. X., Han, Y. & Wu, C. N. An intersection algorithm of NURBS surface and plane based on the method of isoline. *Digit. Technol. Appl.* https://doi.org/10.19695/j.cnki.cn12-1369.2011.07.076
9. Yao, W. W. *Research and Implementation on Bézier Surface/ Surface Intersection Algorithm Based on Hierarchical Bounding Box* 12 (Dalian University of Technology, 2007).
10. Liu, X. M. *Point Projection Methods Using Torus Patch Approximation and Torus/torus Intersection Algorithms* 6 (Tsinghua University, 2011).
11. Fu, Q., Wu, Z. K. & Wang, X. C. etc. An algorithm for finding intersection between ball B-spline curves. *J. Comput. Appl. Math.* 327, 260–273 (2018).
12. Rao, D. J., Liu, X. F. & Mu, G. W. Intersection of NURBS curves based on PSO algorithm. *J. Hebei Univ. Technol.*, **43**(3), 87–91.
13. Adi, E. I. S., Shamsuddin, S. M. & Ali, A. Particle Swarm Optimization for NURBS curve fitting. In *Sixth International Conference on Computer Graphics, Imaging and Visualization*, 259–263. https://doi.org/10.1109/CGIV.2009.64 (2009).
14. Shen, J. J. et al. A Line/Trimmed NURBS surface intersection algorithm using matrix representations. *Comput. Aided. Geom. Des.* **48**, 1–16 (2016).
15. Piegl, L. & Tiller, W. The NURBS Book, 2nd edn.
16. Hearn *Computer Graphics with OpenGL*, 3rd edn (Publishing House of Electronics Industry, 2005).

Acknowledgements

This work was partly supported by Henan Province Science and Technology Research Project (242102220038), Key Research Projects of Higher Education Institutions in Henan Province (23B520021), Research and Cultivation Fund Project of Anyang Institute of Technology (YPY2022011), Key Disciplines of Henan Province, Key Disciplines of Anyang Institute of Technology.

Author contributions

S.W. contributed to the design of the research, and the manuscript was mainly written by S.W. Other authors have validated the research from their own professional perspectives and provided valuable suggestions for revisions.

Declarations

Competing interests

The authors declare no competing interests.

Additional information

Correspondence and requests for materials should be addressed to S.W.

Reprints and permissions information is available at www.nature.com/reprints.