Received January 29, 2021, accepted April 7, 2021, date of publication April 20, 2021, date of current version April 27, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3074509

A New Approach to Analyzing Interactions of Two Objects in Space Based on a Specially-Tailored Local Coordinate System

HAIYAN YU[1], YUANJUN HE[2], AND WENJUN ZHANG[3], (Senior Member, IEEE)
[1]College of Mechanical Engineering, Donghua University, Shanghai 201600, China
[2]Department of Computer Science, Shanghai Jiaotong University, Shanghai 200240, China
[3]Department of Mechanical Engineering, University of Saskatchewan, Saskatoon, SK S7N5A9, Canada

Corresponding authors: Wenjun Zhang (chrismzhang@hotmail.com) and Haiyan Yu (yuhy@dhu.edu.cn)

This work was supported in part by the National Key Research and Development Program of China under Grant 2017YFB1304000.

ABSTRACT Conventionally, geometric computing problems are treated as algebraic computing problems by representing a geometric object in a global reference coordinate system. This approach has two problems. The first problem is that the intuitive view of the interaction of objects is lost, and the second problem is that algebraic computing is prone to errors for degenerated cases related to the interaction of (e.g., notable case of "divided by zero"). In this paper, we propose a new approach to geometric computing especially to analyze the interaction of two objects (e.g., two triangles). The main idea behind this new approach is a specially-tailored local coordinate system for two interacting objects is defined, which makes the projection of the objects on this local coordinate system represent the true geometry of the objects. This idea significantly departs from the conventional approach which is primarily based on the concept of the global coordinate system. Three examples are provided to illustrate the effectiveness of the proposed approach. Among them, one example is related to the robustness of methods for analyzing the relations of two interacting objects, which is still an open issue in the field of geometric computing and suggests that the proposed approach could have some benefit to robustness in analysis.

INDEX TERMS Geometric computing, geometric method, dimension reduction, projection, geometric transformation.

I. INTRODUCTION

In product development, including both aesthetic products (e.g., sculpture) and functional products [21], manipulation of a set of geometric objects is an essential task. In fact, the generic activity in the product development is nothing but to create a geometry that occupies space [22]. Without the loss of generality, this paper discusses the problem of the computer analysis of interactions or relations of two geometric objects.

A popular idea to deal with this problem is to represent this geometric problem into an algebraic problem. However, with this idea, the intuitive view of geometric objects is lost, leading to a degraded contribution to human cognitive activities (understanding, description, manipulation, and inference) for the original geometric problem. Besides, the robustness of methods to analyze relations of two geometric objects is still in question. A well-known example of the concept of robustness may refer to the problem of the so-called singularity in robotics [1], [23]. For a robot shown in Fig. 1a, there are two relative positions with two links, as indicated by the dashed line, which correspond to the relations that the link 1 and the link 2 are coincident (i.e., extended and folded). The two positions correspond that the determinant of the Jacobian matrix of the system is zero (Fig. 1b). As one can see, the visual presentation of the two relations of Fig. 1a is lost if the relations are represented algebraically (Fig. 1b), i.e., the matrix along with the singularity of the matrix. Another robustness problem with this example is related to the matrix computation; in particular the computational error surrounding the singular position can lead to the computational instability. By the way, the robustness of methods for detecting relations of two geometric objects or objects is still an open issue in geometric computing [17].

The associate editor coordinating the review of this manuscript and approving it for publication was Lei Wei.

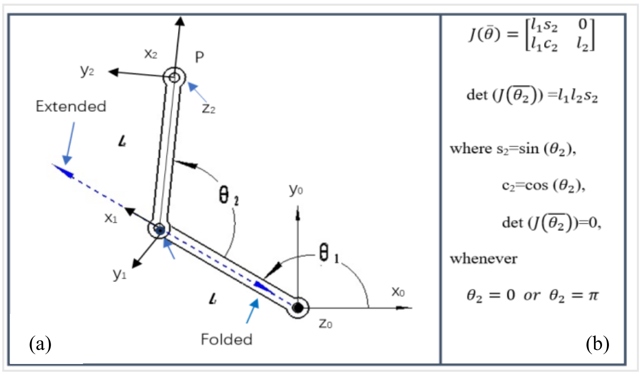
Extended
Folded
P
(a)
(b)
$J(\vec{\theta}) = \begin{bmatrix} l_1 s_2 & 0 \\ l_1 c_2 & l_2 \end{bmatrix}$
$\det(J(\overline{\theta_2})) = l_1 l_2 s_2$
where $s_2 = \sin(\theta_2)$,
$c_2 = \cos(\theta_2)$,
$\det(J(\overline{\theta_2})) = 0$,
whenever
$\theta_2 = 0$ or $\theta_2 = \pi$

FIGURE 1. Singularity in robots [1]. (a) Geometric meaning of singularity of Jacobian matri; (b) The Jacobian matrix representing the relationship between the velocity at point P and angular velocity at the two joints.

Effort has been taken in literature on finding an approach to geometric computing, particularly analysis of relations of two geometric entities in space, that can retain visual interpretation at most while at the same time can be effective. A well-known approach is the one to find a series of projection planes, where part of the dimension of a geometric entity can be truly represented, and algebra can be employed to represent these planes along with true geometric elements on these planes [9]. Fig. 2 illustrates this approach, where Fig. 2a shows a line segment AB (neither parallel nor vertical to the projection planes H, V, W). Fig. 2b shows a series of projections, completed in 2D, to find the true length of AB. Fig. 2c is its analytical solution. This approach is robust to degenerative cases but not a general one.

Another approach in literature is as this: first to find a particular feature with a 3D geometrical problem in the 2D domain [6], [7], and then to solve this 2D problem and subsequently, the original 3D problem is solved. This approach only works out for a small set of problems and besides, the projection plane is not tailored to the two objects under consideration. Yet, another approach is based on imaging technology, particularly the possibility of reconstructing 2D images of a 3D geometric entity to their origin [6], [8]. This approach involves intensive numerical computations, and its efficiency is a concern. Besides, cons associated with any numerical approach may present with this approach.

This paper presents a new approach to geometric computing. The approach is based on two ideas. The **first idea** is to establish a local projection plane (**LP** for short) for the two interacting geometric entities by tailoring it to two concerned geometric objects such that they are in parallel, vertical, or symmetrical to the LP, and subsequently establishing a local coordinate system (LCS) based on the LP. The philosophy behind this idea is that the relation between two interacting objects can be captured and represented most effectively and efficiently by the local coordinate system, which is primarily based on the geometric feature of the two objects in interaction, rather than a general-purpose local coordinate system, which is the case in other geometric computing methods in literature [28], [29]. The **second idea** is

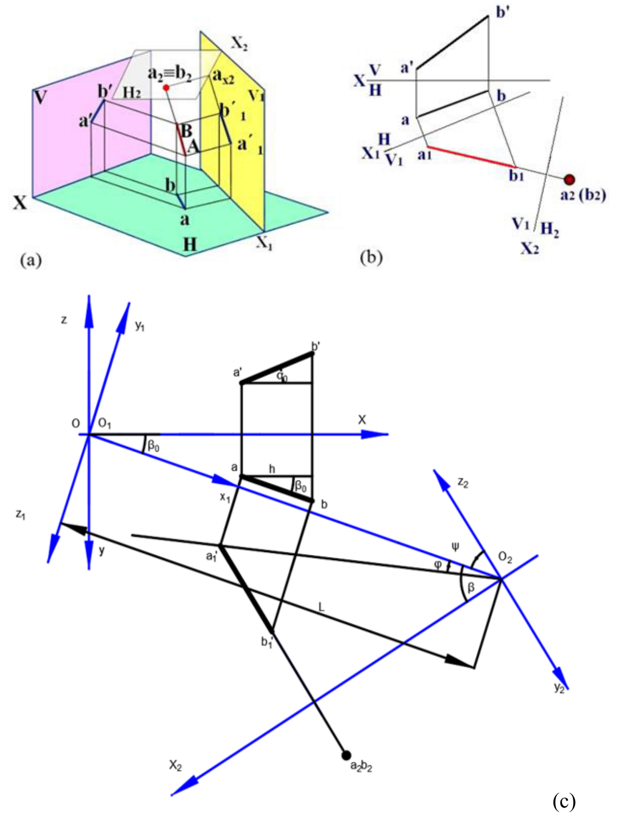

FIGURE 2. An example to show the basic projections in DG. (a) 3D; (b) 2D drafting; (c) analytical solution.

to computerize the 2D process of solving the 2D interaction problem in the LP after the interacting feature of the two concerned objects is projected on the LP. The proposed approach thus enjoys the benefits of visual intuition, efficiency, and robustness in the manipulation of geometric entity, which are fundamental in the context of human-centered computer aided design [10], [15].

It is worth to mention that the benefit of the proposed approach is mainly to the developer of the software for geometrical design rather than to the user of the software. The example of such a developer is the joint team of Europe and Israel who developed a robust software library of geometric algorithms and data structures [16] and computer integrated design and manufacturing of machines [24].

The remaining part of this paper is organized as follows. The proposed approach is presented in Section 2. In Section 3, three examples are given to illustrate the effectiveness of the proposed approach. Finally, a conclusion is given in Section 4.

II. THE PROPOSED APPROACH

The general principle behind any approach of meeting the foregoing two requirements is to convert the 3D problem to the 2D problem by a specially-tailored local coordinate system. The fundamental reason that the 3D problem is more difficult to be solved than the 2D problem is because of the limitation in the capability of the human visual system for

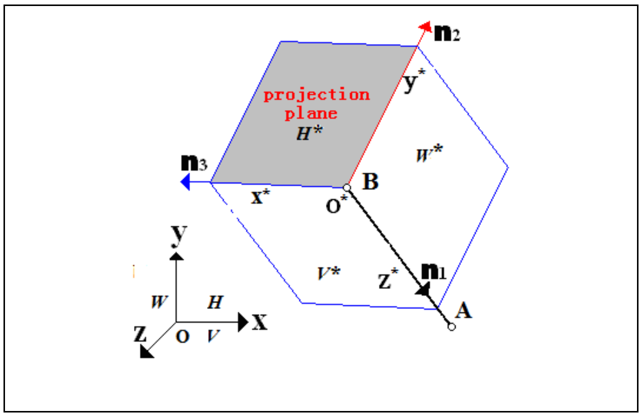

FIGURE 3. The construction of new coordinate system.

the 3D object perception. For instance, it is very difficult to visually judge whether two line segments in 3D space are in an intersection state or not. However, when two objects (e.g., two line segments) in a special plane (on which the two line segments represent their true geometry), it is easy to visually judge whether the two line segments are in an intersection state or not. To generalize this point, one can conclude that a special projection plane (and subsequently a specially-tailored coordinate system which builds on the projection planes) is needed, on which the human can visually see the true geometry of objects and then judge their relationships. The foregoing discussion has led to the two ideas as described before.

A. SPECIALLY-TAILORED LOCAL COORDINATE SYSTEM (F*)

Let us denote F as the original coordinate system and F* as the specially-tailored coordinate system between two concerned objects (e.g., one: line; the other: sphere). Let us take a line segment BA as an example to illustrate how the F* can be established (Fig. 3). Take BA as a normal vector of the specially-tailored projection plane and as one of the coordinate planes of F*. The unit vector BA is also written as BA for simplicity. This unit vector is further defined as the z*-axis of F*, and as such the plane corresponding to the z*-axis is the x*-y* coordinate plane of F* (Fig. 3). The origin of F* is set up at point B (Fig. 3). Define the x*-axis as an arbitrary vector in the x*-y* plane, and accordingly the y*-axis can be defined based on the right-hand rule. For the convenience of later discussions and in the context of projection, the specially-tailored coordinate system F* is expressed as (H*/V*/W*), where H* is the x*-y* plane, and V* and W* are the x*-z* and y*-z* planes, respectively. The projection of the line segment AB on the F* is thus as follows: (1) on the z*-x* plane (or V*), the projection of the line AB is a line $A_V B_V$ (which is the line AB itself; see Fig. 4), and (2) the projection of the line AB on the x*-y* plane (or H*) is a point $A_H \equiv B_H$ (which is also the origin O*; see Fig. 4).

In the above, F* is a specially-tailored local coordinate system established by tailoring two interacting entities such

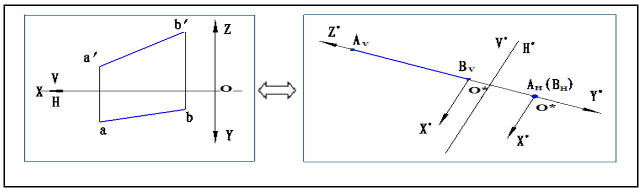

FIGURE 4. Transforming a general line to a vertical line.

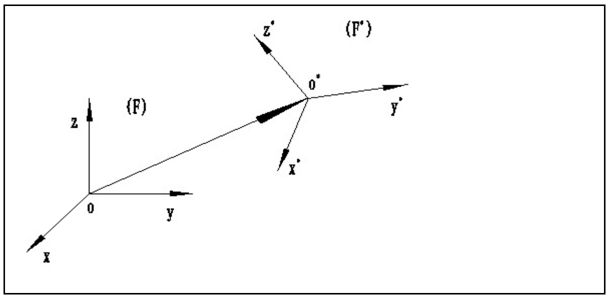

FIGURE 5. Transformation from frame F to F*.

that the corresponding two geometric entities are in a parallel or vertical or symmetrical relation to the coordinate planes of F*. Suppose F is a global reference coordinate system. ${}^{F}_{F^*}T$ (the relations matrix between {F} and {F*}). ${}^{F}_{F^*}T$ is a 4×4 matrix including a rotation matrix ${}^{F}_{F^*}R$ and a translation matrix ${}^{F}_{F^*}O$, where ${}^{F}_{F^*}R$ captures the difference of F and F* in orientation and ${}^{F}_{F^*}O$ captures the difference of F and F* in translation, particularly the coordinates of the origin of F* in the frame F. ${}^{F}_{F^*}T$ can be written as follows:

$$ {}^{F}_{F^*}T = \begin{bmatrix} {}^{F}_{F^*}R_{3\times3} & {}^{F}O^*_{3\times1} \\ 0\,0\,0 & 1 \end{bmatrix}_{4\times4} \tag{1} $$

where

$$ {}^{F}_{F^*}R = \begin{bmatrix} \cos(x,x^*) & \cos(x,y^*) & \cos(x,z^*) \\ \cos(y,x^*) & \cos(y,y^*) & \cos(y,z^*) \\ \cos(z,x^*) & \cos(z,y^*) & \cos(z,z^*) \end{bmatrix} \tag{2} $$

From ${}^{F}_{F^*}T$, we can also find ${}^{F}_{F^*}T$. Then, to any point with respect to the frame F (or F*), we can find their corresponding representation on the frame F*(or F), respectively, with Equation (2) and Equation (3), respectively.

$$ F_P = {}^{F}_{F^*}T^{F^*}P \tag{3} $$
$$ F_P = {}^{F}_{F^*}T^{F}P \tag{4} $$

where $F_P = \begin{Bmatrix} F_x \\ F_y \\ F_z \\ 1 \end{Bmatrix}$ and $\begin{Bmatrix} F_x \\ F_y \\ F_z \end{Bmatrix}$ is the coordinates of P in the frame F; $F_P = \begin{Bmatrix} F^*_x \\ F^*_y \\ F^*_z \\ 1 \end{Bmatrix}$ and $\begin{Bmatrix} F^*_x \\ F^*_y \\ F^*_z \end{Bmatrix}$ is the coordinates of P in the frame F*; and the number '1' in ${}^{F}P$ and ${}^{F^*}P$ has no meaning but for the homogeneous matrix representation only.

There are 25 basic geometric entities, e.g., line segment, sphere, etc. F* is established for each pair of the basic entities.

The total number of combinations of the 25 basic entities is 325, which is found from $\binom{25}{2} + 25$. The proposed approach includes a computer program to create F* for all these combinations (Supplemental material 1 of the present paper). For instance, for the pair of a line segment and a sphere, F* is defined as follows: (1) to have the origin of the F* be the center of the sphere, (2) to have one coordinate plane be parallel to the line segment, and (3) to have another coordinate plane be vertical to the line. As such, two coordinate planes are determined, and the third one can be naturally determined based on the right-hand rule. It is noted that there is no need to find the transformation matrix from F* to F if one is only interested in the interacting relation of the two concerned entities (e.g., whether the two entities overlap), as F* is designed exactly for facilitating the representation of this interacting relation, and this is the benefit of F* over other approaches in literature. The transformation matrix ${}^{F}_{F^*}T$ is needed when the information such as the location where the interaction occurs with respect to Frame F needs to know.

B. COMPUTERIZING THE 2D DRAFTING PROCESS

After F* is established for two interacting objects under consideration, their relation has been represented on the coordinate planes (or projection planes). Therefore, further completing the characterization of the relation will involve 2D drafting operations.

There are about 10 basic drafting (BD) operations, upon which more operation drafting operations are constructed. Examples of the BD operations are drafting a line through 2 points, finding the intersection point of 2 lines, drawing a line that is perpendicular to another line and passes through a point, and so forth. These basic operations can be computerized as computer program functions or procedures (Supplemental material 1). As such, to a particular geometric manipulation problem, say object A and object B, the following steps can be followed. The first step is to find the F* for them and subsequently the 3D interaction problem reduces to the 2D interaction problem. The second step is to analyze the 2D interaction problem by the 2D drafting operations. It may be clear that both steps are automated by the computer.

Take it as an example to draw a circle through three points (P1, P2, P3) for the illustration purpose. The whole task can be completed with the following tasks. Task 1: draw a vertical line which passes through the middle point of the line that connects two points P1 and P2. Let the computer function for this be denoted by LPPN(); in particular LPPN(A, B, L), where A and B are two points, and L is the resulting line. As such, Task 1 is expressed by LPPN(P1, P2, L1). Task 2: LPPN(P2, P3, L2). Task 3: get the intersection point of any two lines, which is expressed as PLL As such, Task 3 can be expressed as PLL(L1, L2, C), where L1 and L2 are two lines and C stores the information of the intersection point of the two lines L1 and L2. The expression of the overall task can then be expressed by CPPP() = {LPPN(); LPPN(); PLL()}, where CPPP(P1,P2,P3) is a function that finds the circle given three points. Clearly, any complicated 2D drafting can be decomposed into a set of the basic drafting operations.

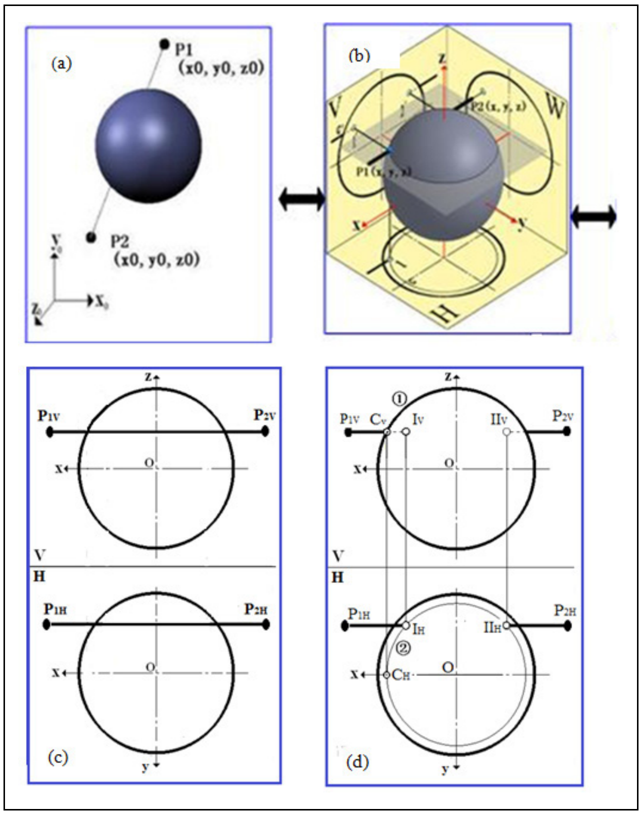

FIGURE 6. Intersection of a line with a sphere. (a) A 3D geometric problem; (b) Transformation to computing coordinates; (c) Dimension reduction of 3D geometries; (d) Construction of 2D solution.

III. EXAMPLES

Three examples are taken to show how the proposed approach works. The first example is analysis of the interaction of a line and a sphere. The second example is to check whether a point is in a bounded box. The third example is a line interacting with a view frustum (or view frustum clipping). Details of the three examples along with the computer code can be found in Supplemental Material 1.

Example 1: Finding the intersection points of a line L with a sphere S (Fig. 6a)

Given: line L is defined by point P_1 and P_2, and sphere S is defined by its center and radius R.

Find: intersection points I and II.

The entire solving process can be decomposed into two steps as follows (Fig. 6):

Step 1: to define F*. The steps of defining F* is shown in Fig. 6b: (1) to have the origin of F* be the center of the sphere, (2) to have one coordinate plane be parallel to the line segment, and (3) to have another coordinate plane be vertical to the line. As such, the projection of the sphere and the projection of the line onto F* are as follows: On V and H, respectively, (1) the projection of the 3D line is a 2D line parallel to the x-axis, and (2) the projection of the sphere is a circle (the center of the circle is the origin of F* and the radius of the circle is the radius of the sphere R), and the circle is denoted as circle (O, R) (Fig. 6c).

Step 2: to decompose the entire task to find the relation of the sphere and line into several 2D drafting tasks: Task 1: on V, to find intersection points of line$P_{1V}P_{2V}$ and circle (O, R) (marked as Circle① in Fig. 6d). The computer function for this is denoted by PLC(), in particular PLC ($P_{1V}P_{2V}$, Circle①, C_V, D_V), where C_V and D_V are the intersecting points. Task 2: on H, to draw a circle (marked as circle② in Fig. 6d), whose center is the origin, and radius is the x-coordinate of C_V. Then, PLC($P_{1H}P_{2H}$, Circle, I_H,II_H), where I_H and II_H are the intersecting points. Task 3: find I_V and II_V (according to I_H and II_H), respectively. The intersection points I (x_1,y_1,z_1) and II (x_2,y_2,z_2) can be represented by their projections; particularly, I_H is defined as (x_1,y_1), I_V is defined as (x_1,z_1); II_H is defined as (x_2,y_2), II_V is defined as (x_2,z_2).

It is noted that the aforementioned 2D computer functions can be found in Supplemental material 1.

Example 2: Detecting the location of a point P to a bounding box (Fig. 7)
Given: Point P and a box defined by its length, width and height denoted by $2h_u$, $2h_v$ and $2h_w$, respectively.
Find: whether P is in the box.

We will give both the conventional method and proposed method in this paper for a comparison in the following.

(1) The conventional method

First, the equation of each plane of a bounding box is established and is then substituted with the coordinates of point P. Finally, by detecting the sign of the substituted equation, the relation of this point and each plane is obtained. In this method, the normal vector of all the planes should all be outward or inward.

(2) Our method

Step 1: to define F*. F* is defined by O-u-v-w. O is the center of this box, and u, v and w are three normal vectors of the box (Fig. 7). Then, the coordinate of point P is transformed to F*, denoted as (P_u, P_v, P_w).

Step 2: the problem can then be expressed by inequalities (4). The point P is not in the bounded box if one of the inequalities (4) is satisfied:

$$p_u > h_u \text{ or } p_u < -h_u \text{ or } p_v > h_v \text{ or } p_v < -h_v \text{or } p_w > h_w \text{ or } p_w < -h_w \quad (5)$$

where h_u, h_v, and h_w are the distance of O to three planes or the half of the length, width, and height of this box, respectively. Our method only involves the standard transformation and simple yes-or-not judgment operation.

Example 3: View Frustum Clipping (Fig. 8)
Given: Line P_1P_2 and a View Frustum defined by P_e, P_t and P_b and four points on the bottom plane.
Find: Intersecting points.

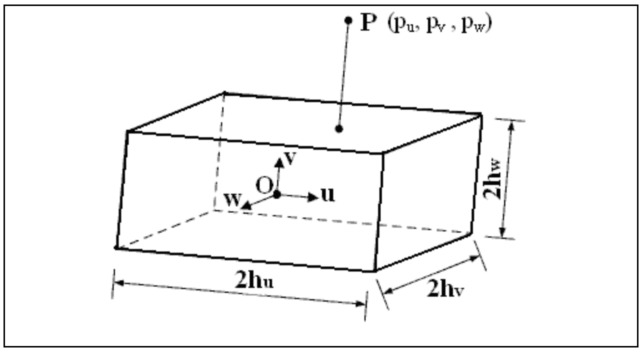

FIGURE 7. The location relation of a point P with a bounding box.

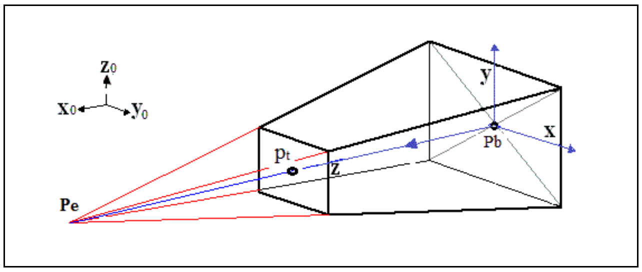

FIGURE 8. A view frustum and the construction of the frame F*.

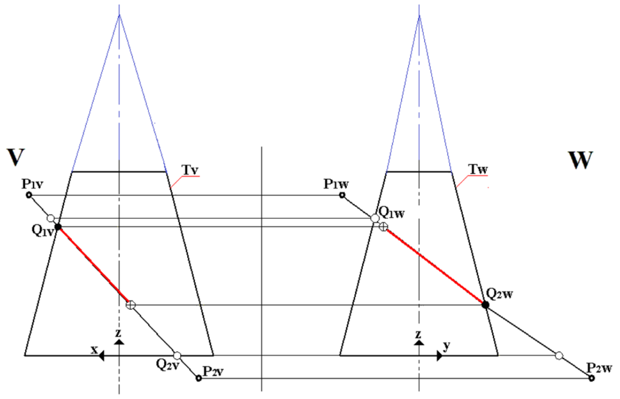

FIGURE 9. The view frustum clipping algorithm with our method.

View frustum clipping is one of the basic techniques in 3D display system [18]. View frustum is a pyramid for perspective transformations. Fig. 8 is a view frustum. The following is the solving process with our method.

Step 1: to define F*. The two symmetric planes and the bottom plane of the frustum are chosen as three coordinate planes of F*, and the vector P_bP_t is taken as the z-axis for F* (see Fig. 8). In F*, the projections of the view frustum to V and W are both isosceles trapezoids (denoted by T_v and T_v) and the projections of the 3D line P_1P_2 on V and W are denoted by $P_{1v}P_{2v}$ and $P_{1w}P_{2w}$, respectively, as shown in Fig. 9.

Step 2: to decompose the 2D drafting task into two clipping functions of LineClipTrapezoidal() (the computer function for finding the intersecting points of a isosceles trapezoid and a line in the code; see Supplemental material 1) and the intersection of the two clipping leads to the final result in F*. See the following tasks for details: Task 1: on V, find

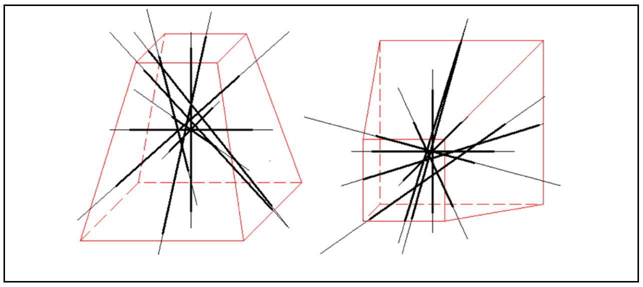

FIGURE 10. The test results of the algorithm for the view frustum clipping.

the intersecting points of T_v and $P_{1v}P_{2v}$, denoted by Q_{1v} and Q_{2V}, i.e., LineClipTrapezoidal ($P_{1v}P_{2v}$,T_v, Q_{1v},Q_{2V}). On W, LineClipTrapezoidal ($P_{1w}P_{2w}$,T_w, Q_{1w},Q_{2w}). Task 2: check the validity of the 2D intersection points. Q_{1v}, Q_{2V} and Q_{1w}, Q_{2w} may not be the projections of the 3D clipping points. The validity is checked by the following rules: Rule 1: Q_v is valid only when its corresponding Q_w on or in T_w. Rule 2: Q_w is valid only when its corresponding Q_v must be on or in T_w. For example, in Fig. 9, the Q_{1v} and Q_n2w are the valid projections of the intersecting points (marked as"●"). This operation is in fact a function of finding the intersecting set. Task 3: compose the 3D coordinate in F*. From the valid points, get their corresponding projections on the other projection plane (Fig. 9, marked as"⊕"). The view frustum clipping algorithm can be put in the supplemental material 1. Fig. 10 shows some test results of the view frustum clipping algorithm. A comparison of our algorithm with the Liang-Barsky algorithm [25] reveals that our algorithm runs slightly faster than the Liang-Barsky algorithm but apparently; besides, our method enjoys the intuitive geometrical representation.

IV. CONCLUSION WITH FURTHER DISCUSSIONS

This paper presented a new approach to geometric computing. There are two common goals with any approach to geometric computing: Goal 1: it should preserve the geometric intuition as much as possible; Goal 2: it should enable the computer processing as much as possible (i.e., automating the processing). The two goals may be conflicting. In that sense, it can be said that the proposed approach has provided a trade-off between the two goals. Specifically, the approach has two steps. The first step is to find a specially-tailored coordinate system (F*) such that the geometric elements are parallel, vertical or symmetrical to the coordinate plane of F*. A computer program was developed to automate this step. This step can be analogous to the construction of solid geometry, which decomposes a complex geometric entity into a group of standard solids such as sphere, square, rod, and so on and then, construct the 3D geometric entity from this set. It is noted that on the coordinate planes of F*, the two concerned objects have the highest possibility that their true dimensions are displayed, so the 3D problem becomes the 2D problem against this local coordinate system rather than a global coordinate system in many existing approaches. The second step of the approach was to computer-process the 2D problem (e.g., interaction of two line segments, a point on line, etc.), which is readily available. It is noted that F* is a local coordinate system defined by tailoring to the interacting relation of two objects, and this is a point of departure from the notion of the local coordinate system in literature, which ignores the interacting relation of two objects.

The main benefit of the proposed approach is that the feature of the geometrical problem is kept in F*. For instance, the problem of finding the workspace of the robot as described in Fig. 1 can be described by stating that Link 1 and Link 2 are either folded in one line or extended in one line instead of being stated as the singularity of the Jacobian matrix. Apparently, if the problem is solved by the concept of singular Jacobian matrix, there may be some numerical challenge in terms of the definition of the determinant of the Jacobian matrix being zero.

In a separate paper, we will show benefits with the proposed approach to solving some well-known geometrical interaction problems in terms of robustness over the existing approach in the literature, such as the 3D triangle-triangle interaction problem [11]. Supplemental material 2 of the present paper gives a brief description of the theory, algorithm, computational robustness and performance of a 3D triangle-triangle interaction method which is based on the approach presented in this paper.

In the future, a new concept called the resilience of computational methods for geometric computing will be studied, which is different from the robustness of computational methods. The difference of these two concepts is now clear in scheduling methods for service and manufacturing systems [26], [27].

REFERENCES

[1] J. J. Craig, *Introduction to Robotics: Mechanics and Control*. Reading, MA, USA: Addison-Wesley, 1992.

[2] J. Malkevitch. (2003). *Mathematics and Art, American Mathematical Society Feature Column*. [Online]. Available: http://www.ams.org/samplings/feature-column/fcarc-art1

[3] H. Stachel, "What is descriptive geometry for?" in *Proc. Dresden Symp. Geometry Process.*, 2003, pp. 327–336.

[4] P. Paukowitsch, "Fundamental ideas for computer-supported descriptive geometry," *Comput. Graph.*, vol. 12, no. 1, pp. 3–14, Jan. 1988.

[5] M. Kreveld, "The power of parallel projection," *Inf. Process. Lett.* vol. 46, no. 4, pp. 185–191, 1993.

[6] H. Stachel, "Descriptive geometry meets computer vision-the geometry of two images," *J. Geometry Graph.*, vol. 10, no. 2, pp. 137–153, 2006.

[7] Y. T. Lee and F. Fang, "3D reconstruction of polyhedral objects from single parallel projections using cubic corner," *Comput.-Aided Des.*, vol. 43, no. 8, pp. 1025–1034, Aug. 2011.

[8] Y. T. Lee and F. Fang, "A new hybrid method for 3D object recovery from 2D drawings and its validation against the cubic corner method and the optimisation-based method," *Comput.-Aided Des.*, vol. 44, no. 11, pp. 1090–1102, Nov. 2012.

[9] H. Zhu, *Advanced Descriptive Geometry*. Shanghai, China: Shanghai Science and Technology, 1985.

[10] S. Modi, M. K. Tiwari, Y. Lin, and W. J. Zhang, "On the architecture of a human-centered CAD agent system," *Comput.-Aided Des.*, vol. 43, no. 2, pp. 170–179, Feb. 2011.

[11] M. Held, "ERIT—A collection of efficient and reliable intersection tests," *J. Graph. Tools*, vol. 2, no. 4, pp. 25–44, Jan. 1997.

[12] L. X. Fan, M. Y. Cai, Y. Lin, and W. J. Zhang, "Axiomatic design theory: Further notes and its guideline to applications," *Int. J. Mater. Product Technol.*, vol. 51, no. 4, pp. 359–374, 2015.

[13] W. J. Zhang and J. W. Wang, "Design theory and methodology for enterprise systems," *Enterprise Inf. Syst.*, vol. 10, no. 3, pp. 245–248, Mar. 2016.

[14] W. J. Zhang, Y. Lin, and N. Sinha, "On the function-behavior-structure model for design," in *Proc. 2nd CDEN Conf.*, Alberta, CA, Canada, Jul. 2005, p. 8.

[15] Y. Zeng and I. Horváth, "Fundamentals of next generation CAD/E systems," *Comput.-Aided Des.*, vol. 44, no. 10, pp. 875–878, Oct. 2012.

[16] The Computational Geometry Algorithms Library. *The Cooperative Association for Internet*. Accessed: May 4, 2016. [Online]. Available: http://www.cgal.org

[17] D. Halperin, "Robust geometric computing in motion," *Int. J. Robot. Res.*, vol. 21, no. 3, pp. 219–232, Mar. 2002.

[18] R. Parekh, *Principles of Multimedia*, 2nd ed. New York, NY, USA: McGraw-Hill, 2013, p. 413.

[19] A. Gomez and M. Shin, "3D structure development using a three-layer self-folding technology," *J. Mech. Sci. Technol.* vol. 32, p. 2107, Oct. 2018, doi: 10.1007/s12206-018-0613-y.

[20] W. Sun, L. Gu, R. Wang, and T. Qi, "Adaptive finite element analysis of steel girder deck pavement," *J. Mech. Sci. Technol.*, vol. 32, no. 2, pp. 593–603, Feb. 2018.

[21] Y. Lin and W. Zhang, "Integrated design of function, usability, and aesthetics for automobile interiors: State-of-the-art, challenges, and solutions," *J. Syst. Control Eng.*, vol. 220, no. 18, pp. 697–708, 2006.

[22] L. Cao, A. T. Dolovich, A. Chen, and W. Zhang, "Topology optimization of efficient and strong hybrid compliant mechanisms using a mixed mesh of beams and flexure hinges with strength control," *Mech. Mach. Theory*, vol. 121, pp. 213–227, Mar. 2018.

[23] L. Cheng, Y. Lin, Z.-G. Hou, M. Tan, J. Huang, and W. J. Zhang, "Adaptive tracking control of hybrid machines: A closed-chain five-bar mechanism case," *IEEE/ASME Trans. Mechatronics*, vol. 16, no. 6, pp. 1155–1163, Dec. 2011.

[24] W. J. Zhang, "An integrated environment for CADCAM of mechanical systems," Ph.D. dissertation, Fac. Mech. Eng. Marine Technol., Delft Univ. Technol., Delft, The Netherlands, 2012, p. 263.

[25] Y.-D. Liang and B. A. Barsky, "A new concept and method for line clipping," *ACM Trans. Graph.*, vol. 3, no. 1, pp. 1–22, Jan. 1984.

[26] W. J. Zhang, "Towards a resilient manufacturing system," *Ann. CIRP*, vol. 60, pp. 469–472, May 2011.

[27] B. Han, C. L. Liu, and W. J. Zhang, "A method to measure the resilience of algorithm for operation management," *IFAC-Papers Line*, vol. 49, no. 12, pp. 1442–1447, 2016.

[28] D. Cohen, "Incremental methods for computer graphics," ARPA, London, U.K., Harvard Rep. ESD-TR-69-193, Apr. 1969.

[29] M. Elliriki, C. S. Reddy, and K. Anand), "An efficient line clipping algorithm in 2D space," *Int. Arab J. Inf. Technol.*, vol. 16, pp. 798–807, Oct. 2019.

HAIYAN YU received the Ph.D. degree from the College of Information Science and Technology, Donghua University, in 2011. Since 1998, she has been working on integrating classical descriptive geometry to modern computing. In 2017, she visited Stanford University for the collaborative research in nano imaging. She has constructed a batch of algorithms under the framework of computerized descriptive geometry. She is currently active in communications with international researchers. She has been invited as a Session Chairman for the International Conference on Geometry and Graphics (ICGG) and made presentations, in 2012, 2016, and 2018. She has published over 20 articles in the fields of geometry and graphics. Her main research interests include computer graphics, geometric computing and their applications in CAD, and engineering modeling. She is a Council Member of the China Graphics Society and the Secretary-General of the Committee of Graphics Computing.

YUANJUN HE received the degree in mathematics from Zhengjiang University, China. He is currently a Professor of computational science with Shanghai Jiao-Tong University, Shanghai, China. He has developed a CAD software called KerenCAD. He has published five books and more than 150 articles. His main research interests include computer graphics and geometric computing.

WENJUN ZHANG (Senior Member, IEEE) received the Ph.D. degree from the Delft University of Technology, in 1994. He has published over 300 articles in refereed journals or magazines, over 200 papers in refereed conference proceedings with his H-index of 55 (according to Google Scholar), and held over ten patents. His one of the focused research themes is computer aided design and informatics. His main research interest includes system science and engineering and their applications to manufacturing and service systems. He is currently a Fellow of the Canadian Academy of Engineering (CAE), a Fellow of ASME, and a Senior Member of SME. He has been very active in editorial board work for six reputed journals, including IEEE Transaction on Mechatronics (Senior Editor since 2019), IEEE System Journal, and IEEE Transactions on Systems, Man, and Cybernetics—System (currently).