IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 18 January 2023, accepted 17 February 2023, date of publication 27 February 2023, date of current version 6 March 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3249182

RESEARCH ARTICLE

ASIC Design and Implementation of the Real-Time Collision Detection for Machine Tool Automation

TSUNG-HSIEN LIU[1], (Member, IEEE), PO-YI CHEN[2], AN-HONG LI[3], YU-YANG FANG[4], RONG-SHINE LIN[5], AND YUAN-SUN CHU[6]

[1]Department of Communications Engineering, Advanced Institute for Manufacturing with High-tech Innovations (AIM-HI), National Chung Cheng University, Chiayi 62102, Taiwan
[2]Novatek Microeletronics Corporation, Hsinchu 300092, Taiwan
[3]Nuvoton Technologies Inc., Hsinchu 30078, Taiwan
[4]Himax Technologies Inc., Tainan 74148, Taiwan
[5]Department of Mechanical Engineering, Advanced Institute for Manufacturing with High-tech Innovations (AIM-HI), National Chung Cheng University, Chiayi 62102, Taiwan
[6]Department of Electrical Engineering, Advanced Institute for Manufacturing with High-tech Innovations (AIM-HI), National Chung Cheng University, Chiayi 62102, Taiwan

Corresponding author: Yuan-Sun Chu (chu@ee.ccu.edu.tw)

This work was supported in part by the Ministry of Science and Technology (MOST), Taiwan, under Contract MOST 109-2221-E-194-032-MY2; and in part by the Advanced Institute of Manufacturing with High-Tech Innovations (AIM-HI) through the Featured Areas Research Center Program within the framework of the Higher Education Sprout Project by MOST.

ABSTRACT Collision detection of two objects plays an essential role for the machine tool automation. Although the collision detection of two objects has been studied in applications like virtual reality, the collision detection for the machine tool requires high precision to avoid overcut damage to the high-cost machine tools. The current collision detection for machine tools is under the soft-ware based computer numerical control (CNC), the low computation capability of which refrains the CNC based approach from real-time collision detection. In this paper, we consider the design of application specific integrated circuit (ASIC) to enhance the collision detection for machine tool automation. Because the bounded objects are represented by meshed triangles, we consider the separating axis theorem (SAT) based detection algorithm. Furthermore, by considering high precision required by machine tool applications, the proposed algorithm includes collision detection of either non-coplanar or coplanar triangles. Following the collision detection algorithm, we design hardware architecture with parallel processing to provide higher throughput rate over the architecture reported in our conference paper. The VLSI implementation results under the TSMC TN40G (45nm) CMOS technology reveal that our architecture requires 1,212K gates and provides detection throughput 38.46M per second for collision detection of two triangles, while operating at 500 MHz. For two objects represented by 400 and 400 meshed triangles, respectively, our hardware architecture can provide collision detection in 0.96 ms, which is smaller than the 1 ms required for real-time processing of collision detection of two objects.

INDEX TERMS Application specific integrated circuit, collision detection, coplanar triangles, low power consumption, machine tool, meshed triangles, non-coplanar triangle, separating axis theorem.

I. INTRODUCTION

Machine tool automation plays an important role to enhance the value and efficiency of machine tools. For the automation of the five-axis sculpturing machine tool (SMT) in Fig. 1, collision detection of the sculptured object and cutter is an essential technique. The collision detection of two objects is necessary not only in machine tool automation [1], [2], [3], [4], [5], [6] but also in the film animation, computer games, robots [7], etc.

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

FIGURE 1. The 5-axis sculpturing machine tool [8].

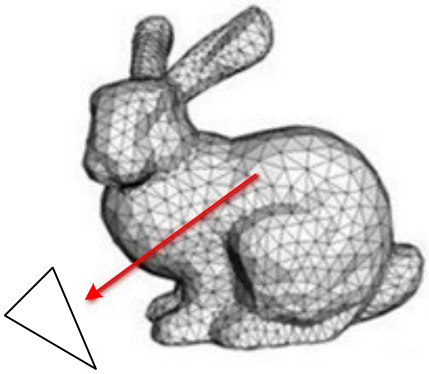


FIGURE 2. Mesh of triangles for representing a bounded object [11].

Representation of the 3-dimensional (3D) objects [9], [10], [11] is the first step for studying the collision detection of two objects. Among the multiple representations, the use of meshed polygons to represent a 3D object (see Fig. 2) simplifies the problem of collision detection. The collision detection of two bounded objects is reduced to the collision detection of two polygons from two objects [12], [13], [14]. Furthermore, there exist practical algorithms for collision detection of two polygons, e.g., the Gilbert-Johnson-Keerthi distance algorithm [15] and those depending on the separating axis theorem (SAT) [1], [9], [16], [17].

Nonetheless, the collision detection of two objects requires large amount of computational work. For example, if either of the two objects is represented by 200 polygons, the computational burden for collision detection of two objects is equivalent to the burden for 40,000 times of collision detection of two polygons. To realize the collision detection of two objects for the SMT, the soft-ware based computer numerical control (CNC) approach [18] has been developed. Due to the low-computing capability, the soft-ware based approach makes the SMT unable to achieve real-time collision detection [3], [7], [19], [20].

To overcome the above-mentioned drawbacks of the soft-ware based approach, we study the design of application specific integrated circuit (ASIC) for collision detection of two objects in this paper. The considered objects are

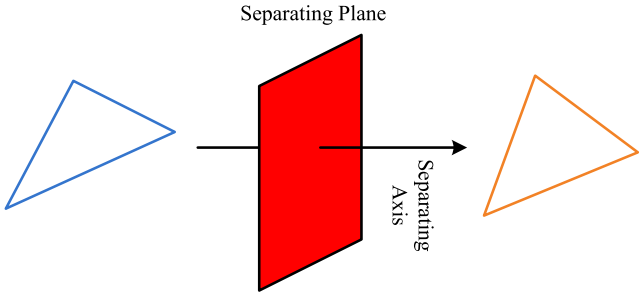


FIGURE 3. Separating plane and separating axis.

represented by meshed triangles. The problem of collision detection of two objects is converted to the problem of collision detection of two triangles from the two objects. Thus, by following the SAT based algorithm, we propose a hardware architecture for collision detection of two objects. Due to the parallel structure, the designed architecture performs collision detection of two objects in less than 1 ms and meets the requirement for the automation of 5-axis SMT. Furthermore, multi-threshold technique is employed to reduce the power consumption and leads our architecture to be applicable in a low power consumption environment.

Fundamentally, the contribution in this paper is an expansion to the work in our conference paper [20]. In addition to the more elegant technical presentation, we will propose an improved algorithm and hardware design over the one in [20]. We will show that the improved architecture requires less silicon area, works at higher frequency, consumes less power, and performs the collision detection in shorter time interval. Furthermore, to our knowledge, there is no journal paper studying the ASIC design for collision detection applied to machine tool automation. As machine tool automation is essential for the future precision machinery, our contribution about the state-of-the-art hardware design for collision detection of two objects here is therefore valuable.

The organization of this paper is as follows. In Section II, we introduce the SAT and our proposed SAT based algorithm for collision detection of two triangles. Based on the proposed algorithm, a new hardware architecture is proposed in Section III. We report the implementation results for the proposed architecture in Section IV. Concluding remarks about our architecture and implementation results are delivered in Section V.

II. THE COLLISION DETECTION ALGORITHM

Our following implementation algorithm for collision detection of two 3D triangles relies on the SAT [16], [21]. In general, the SAT is applied to the collision detection of two convex sets. But here, we only present those results related to the collision detection of two triangles in the 3D or 2D space.

Let us consider the scenario in Fig. 3. Two 3D triangles are said to be collision-free if and only if there exists a separating plane between the two triangles. A vector that is orthogonal to the separating plane is the associated separating axis. The

TABLE 1. Potential separating axes for two non-coplanar triangles A and B with vertices $\{A_1, A_2, A_3\}$ and $\{B_1, B_2, B_3\}$, respectively.

Index	Axis	Index	Axis	Index	Axis
#1	$\overrightarrow{A_1A_2} \times \overrightarrow{B_1B_2}$	#2	$\overrightarrow{A_1A_2} \times \overrightarrow{B_2B_3}$	#3	$\overrightarrow{A_1A_2} \times \overrightarrow{B_3B_1}$
#4	$\overrightarrow{A_2A_3} \times \overrightarrow{B_1B_2}$	#5	$\overrightarrow{A_2A_3} \times \overrightarrow{B_2B_3}$	#6	$\overrightarrow{A_2A_3} \times \overrightarrow{B_3B_1}$
#7	$\overrightarrow{A_3A_1} \times \overrightarrow{B_1B_2}$	#8	$\overrightarrow{A_3A_1} \times \overrightarrow{B_2B_3}$	#9	$\overrightarrow{A_3A_1} \times \overrightarrow{B_3B_1}$
#10	$\overrightarrow{N_A}$	#11	$\overrightarrow{N_B}$		

separating plane and associated separating axis may not be unique.

Each triangle A is characterized by its three vertices $\{A_1, A_2, A_3\}$. The associated normal vector $\overrightarrow{N_A}$ to the triangle A can be computed from the coordinates of the three vertices. The notation $\overrightarrow{OA_1}$ denotes the directional vector that starts from the origin O of the coordinate system and ends at vertex A_1. The three edges of triangle A are $\overrightarrow{A_1A_2}$, $\overrightarrow{A_2A_3}$, and $\overrightarrow{A_3A_1}$. The cross product of two vectors $\vec{a}$ and $\vec{b}$ produces vector [22]

$$\underbrace{\begin{bmatrix} a_2b_3 - a_3b_2 \\ a_3b_1 - a_1b_3 \\ a_1b_2 - a_2b_1 \end{bmatrix}}_{\vec{a}\times\vec{b}} \triangleq \underbrace{\begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix}}_{\vec{a}} \times \underbrace{\begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}}_{\vec{b}}. \tag{1}$$

The vector $\vec{a} \times \vec{b}$ is orthogonal to both vectors $\vec{a}$ and $\vec{b}$. To determine that two non-coplanar triangles A and B are collision-free, one only needs to find one separating axis vector from the 11 potential separating axes [12], [21], [24] in Table 1. These potential separating axes include the normal vectors and the cross product vectors of edge vectors associated with the two triangles.

A potential separating axis $\vec{c}$ becomes a separating axis to two collision-free triangles if and only if the projections of the two triangles onto the potential separating axis $\vec{c}$ do not overlap. The projection of vertex A_1 of triangle A on the potential separating axis $\vec{c}$ is [22]

$$\text{proj}\left(\overrightarrow{OA_1}, \vec{c}\right) = \frac{\overrightarrow{OA_1} \cdot \vec{c}}{||\vec{c}||^2} \vec{c}, \tag{2}$$

where the inner product of two vectors $\vec{a}$ and $\vec{b}$ is defined by

$$\underbrace{a_1b_1 + a_2b_2 + a_3b_3}_{\vec{a}\cdot\vec{b}} \triangleq \underbrace{\begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix}}_{\vec{a}} \cdot \underbrace{\begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}}_{\vec{b}}. \tag{3}$$

Two vectors $\vec{a}$ and $\vec{b}$ are orthogonal if and only if $\vec{a} \cdot \vec{b} = 0$. The $||\vec{c}||^2 = \vec{c} \cdot \vec{c}$ in (2) denotes the norm squares of vector $\vec{c}$. The projection in (2) denotes the amount of projection $\overrightarrow{OA_1} \cdot \vec{c}$ along the directional vector $\vec{c}/||\vec{c}||^2$. Instead of testing the projection vectors, we can simply test

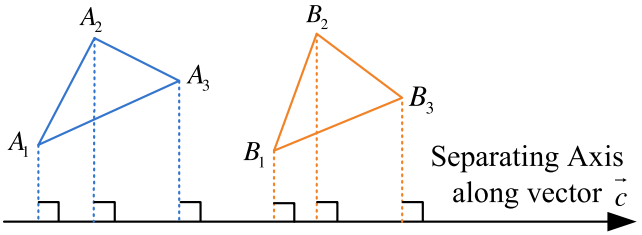


FIGURE 4. The projections of triangles A and B on the separating axis.

TABLE 2. Potential separating axes for two coplanar triangles A and B.

Index	Axis	Index	Axis	Index	Axis
#1	$\overrightarrow{N_A} \times \overrightarrow{A_1A_2}$	#2	$\overrightarrow{N_A} \times \overrightarrow{A_2A_3}$	#3	$\overrightarrow{N_A} \times \overrightarrow{A_3A_1}$
#4	$\overrightarrow{N_B} \times \overrightarrow{B_1B_2}$	#5	$\overrightarrow{N_B} \times \overrightarrow{B_2B_3}$	#6	$\overrightarrow{N_B} \times \overrightarrow{B_3B_1}$

the amount of projections to determine whether the projections of the two triangles overlap. Thus, as shown by Fig. 4, a vector $\vec{c}$ is a separating axis for triangles A and B if and only if

$$\min\left\{\overrightarrow{OA_1} \cdot \vec{c}, \overrightarrow{OA_2} \cdot \vec{c}, \overrightarrow{OA_3} \cdot \vec{c}\right\} > \max\left\{\overrightarrow{OB_1} \cdot \vec{c}, \overrightarrow{OB_2} \cdot \vec{c}, \overrightarrow{OB_3} \cdot \vec{c}\right\} \tag{4}$$

or

$$\max\left\{\overrightarrow{OA_1} \cdot \vec{c}, \overrightarrow{OA_2} \cdot \vec{c}, \overrightarrow{OA_3} \cdot \vec{c}\right\} > \min\left\{\overrightarrow{OB_1} \cdot \vec{c}, \overrightarrow{OB_2} \cdot \vec{c}, \overrightarrow{OB_3} \cdot \vec{c}\right\}. \tag{5}$$

Additionally, when the two triangles are coplanar, the potential separating axes need to be considered differently. Observe that when the two triangles are coplanar, the beginning 9 potential separating axes in Table 1 are along the same direction. For such two coplanar triangles, only 6 potential separating axes needs to be considered and they are listed in Table 2.

The SAT based collision detection algorithm is now listed in Algorithm 1. The algorithm is to test whether each potential separating axis in Table 1 or 2 is a separating axis. When a separating axis is found, the indicator CollisionOccurs $= 0$ is returned; otherwise, the indicator CollisionOccurs $= 1$ is returned. The pseudo codes in lines 3-6 of Algorithm 1 are for computing the projections (inner products) of triangles A and B onto the normal vectors $\overrightarrow{N_A}$ and $\overrightarrow{N_B}$, respectively. The codes in lines 7-11 are for testing whether $\overrightarrow{N_A}$ or $\overrightarrow{N_B}$ is a separating axis. Note that $\overrightarrow{N_A}$ and $\overrightarrow{N_B}$ are the 10- and 11-th potential separating axes in Table 1. The computed values t_k and v_k, $\forall k$, are also used in line 12 of Algorithm 1 to test whether the two triangles are coplanar. Therefore, the computed t_k and v_k, $\forall k$, are tested twice as in lines 7 and 12, a fact that leads the proposed Algorithm 1 to achieve complexity reduction. Finally, the algorithm tests each potential separating axis in lines 14-22 or 25-33 of Algorithm 1 to determine whether it

Algorithm 1 The SAT Based Algorithm to Detect Collision of Two Triangles A and B

input : $\overrightarrow{OA_1}$, $\overrightarrow{OA_2}$, $\overrightarrow{OA_3}$, $\overrightarrow{N_A}$, $\overrightarrow{OB_1}$, $\overrightarrow{OB_2}$, $\overrightarrow{OB_3}$, $\overrightarrow{N_B}$
output: CollisionOccurs

1 Compute the edge vectors $\overrightarrow{A_1A_2}$, $\overrightarrow{A_2A_3}$, $\overrightarrow{A_3A_1}$, $\overrightarrow{B_1B_2}$, $\overrightarrow{B_2B_3}$, $\overrightarrow{B_3B_1}$
2 CollisionOccurs $= 1$
3 **for** $k = 1$ **to** 3 **do**
4 | Compute $t_k = \overrightarrow{OA_k} \cdot \overrightarrow{N_A}$ and $s_k = \overrightarrow{OB_k} \cdot \overrightarrow{N_A}$
5 | Compute $u_k = \overrightarrow{OA_k} \cdot \overrightarrow{N_B}$ and $v_k = \overrightarrow{OB_k} \cdot \overrightarrow{N_B}$
6 **end**
7 **if** $((\min_k t_k > \max_k s_k)$ *or* $(\max_k t_k > \min_k s_k))$ *or* $((\min_k u_k > \max_k v_k)$ *or* $(\max_k u_k > \min_k v_k))$ **then**
8 | CollisionOccurs $= 0$
9 | return;
10 **else**
11 **end**
12 **if** $((t_1 == v_1)$ *and* $(t_2 == v_2)$ *and* $(t_3 == v_3))$ **then**
13 | % Coplanar triangles
14 | **for** $k = 1$ **to** 6 **do**
15 | | Assign $\overrightarrow{c}$ to be the k-th axis in Table 2
16 | | Compute the 6 inner product terms in (4)
17 | | **if** *(*inequality *(4)* or *(5)* holds*)* **then**
18 | | | CollisionOccurs $= 0$
19 | | | return;
20 | | **else**
21 | | **end**
22 | **end**
23 **else**
24 | % Non-Coplanar triangles
25 | **for** $k = 1$ **to** 9 **do**
26 | | Assign $\overrightarrow{c}$ to be the k-th axis in Table 1
27 | | Compute the 6 inner product terms in (4)
28 | | **if** *(*inequality *(4)* or *(5)* holds*)* **then**
29 | | | CollisionOccurs $= 0$
30 | | | return;
31 | | **else**
32 | | **end**
33 | **end**
34 **end**
35 return;

is a separating axis for the coplanar or non-coplanar triangles, respectively.

One important property possessed by Algorithm 1 is that the computations in the algorithm includes multiplications and comparisons only. Recall that the projection in (2) requires division by $||\cdot||^2$. To avoid the division operation, we replace the projection by the inner product in (4) and (5) without sacrificing the performance. This division-free algorithm leads our derived architecture in the next section to be divisor-free and numerically stable.

The number of real-valued multiplications (RMULs) required by the proposed Algorithm 1 is counted. The computations in lines 3-6 to determine whether coplanar triangles exist require 36 RMULs, because each inner product requires 3 RMULs. For the case of coplanar triangles, the number of RMULs required by the operations in lines 14-22 may be as small as 18 or as large as 6*18=108. The 18 RMULs is required when the first potential separation axis is determined to be a separation axis, and, the 108 RMULs is required when none of 6 potential separation axis is a separation axis. Similarly, for the case of non-coplanar triangles, the number of RMULs required by the algorithm for the operations in lines 25-33 varies from 18 to 9*18=152 In totality, the algorithm is of variable computational complexity, i.e., it may require as few as 36+18=54 RMULs for the coplanar triangles or as large as 36+152=188 RMULs for the non-coplanar triangles.

Our algorithm here requires fewer RMULs than our previous conference work in [20]. The major advantage of this improved algorithm is revealed by the pseudo codes in lines 4 and 5 of Algorithm 1, where t_k, s_k, u_k, and v_k, $k = 1, 2, 3$ are computed. These values are used to test whether the two triangles are coplanar and also to test whether the potential separating axes $\overrightarrow{N_A}$ and $\overrightarrow{N_B}$ are really separating axes. Thus, in the worst case of non-coplanar triangles, neither $\overrightarrow{N_A}$ nor $\overrightarrow{N_B}$ is a separating axis; and, the algorithm only needs to test the remaining 9 potential separating axes in lines 25-33. However, in [20], to determine whether the two triangles are coplanar and to determine whether normal vectors $\overrightarrow{N_A}$ and $\overrightarrow{N_B}$ are the separating axes are computed separately. The algorithm in [20] needs to test, for the worst case of non-coplanar triangles, 11 potential separating axes. For this reason, our algorithm here requires fewer RMULs than the one in [20]. Note that the algorithm in [20] also outperms the in [12] and other several algorithms. In later sections, we will also show that the designed hardware architecture from Algorithm 1 has higher hardware efficiency than the proposes architecture in [20].

III. THE PROPOSED HARDWARE ARCHITECTURE

The precision for each object may be as small as 2 μm; and, the size of each object may be as large as 200 mm or $2 \cdot 10^5$ μm in the x-axis. Under such condition, the word length for representing each fixed-point number to realize Algorithm 1 is determined to be 18 bits. Besides, the condition to test the coplanarity of two triangles in line 12 of Algorithm 1 is obtained from the mathematically results. The condition is inappropriate for the fixed-point numbers here, and, it is modified as

$$|t_k - v_k| < \epsilon, \quad k = 1, 2, 3, \tag{6}$$

The ϵ allows the proposed architecture to tolerate the numerical quantization error, truncation error after the multiplication of two numbers, and other noisy effects. Thus, the condition in (6) with $\epsilon = 2$ μm is used in our proposed hardware architecture.

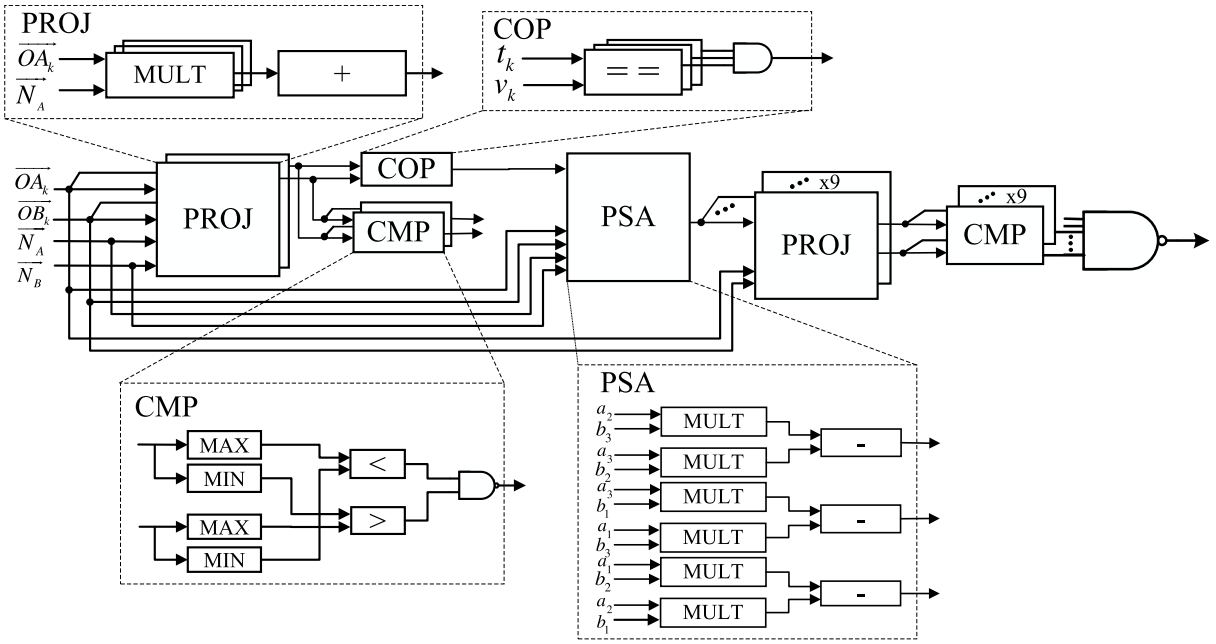


FIGURE 5. The proposed hardware architecture.

Clock	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	Input signal			PROJ		COP	PSA			PROJ		CMP/OUT			

FIGURE 6. Time schedule for the computation by the proposed architecture.

We now introduce the proposed hardware architecture for implementing Algorithm 1 in Fig. 5. The architecture is designed for the worst computational complexity case, which requires 188 RMULs to determine whether two non-coplanar triangles collide or not. For practical consideration, the information of meshed triangles to represent two objects is saved in a data file in STereoLithography (STL) file format [9]. Each triangle from the meshed triangles for representing an object is characterized by the coordinates of the three vertices and the associated normal vector. These coordinates and associated normal vectors for two triangles are inputs to our hardware architecture. Each PROJ module is for computing the inner product of two vectors. Each CMP module that follows a PROJ module performs the comparisons in (4) and (5) to yield indication about whether a separating axis is found. Two PROJs on the left-hand side are associated with the operations in lines 4 and 5 of Algorithm 1. The COP module produces an indicator to indicate whether the two input triangles are coplanar or not. The PSA module produces the remaining 9 or 6 potential separating axes for the non-coplanar or coplanar triangles, respectively. Parallel 9 PROJs are associated with the inner product operations in line 27 of Algorithm 1. When coplanar triangles are detected, only six of the 9 PROJs are activated to compute the inner products. Since the two input triangles are either coplanar or non-coplanar, only the maximum number of PROJs are required in the architecture to save silicon area. The very last NAND operation delivers indicator CollisionOccurs $= 0$ when a separation axis is found. The time schedule for the computation by the proposed architecture is illustrated in Fig. 6. The proposed architecture is divisor-free and may be numerically stable.

TABLE 3. Power consumption by our architecture after the application of the multi-Vt technique.

	Leakage Power Consumption	Total Power Consumption
low-Vt	26.02 mW (100.00%)	100.16 mW (100.00%)
low-Vt + high-Vt	13.76 mW (52.88%)	81.14 mW (81.01%)

Hardware parallelization is employed in the proposed architecture. The Algorithm 1 is of variable complexity, because it exits in line 9, 19, or 30 when a separating axis is found. However, for the architecture in Fig. 5, parallel computation is designed and there are 9 PROJ modules to process the 9 or 6 potential separating axes for non-coplanar or coplanar triangles, respectively. Such design allows the proposed architecture to work with high throughput rate and to perform with fixed complexity.

IV. THE IMPLEMENTATION RESULTS

We design the proposed architecture to consume low power using the multi-threshold (or multi-Vt) technique [23]. The TSMC TN40G (45nm) CMOS technology allows standard

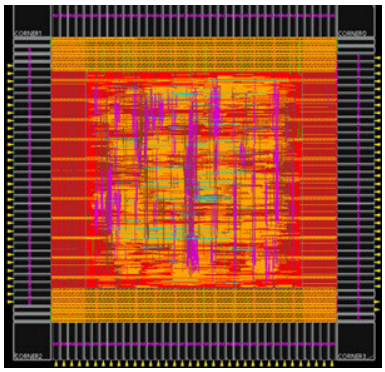


FIGURE 7. Chip layout of the proposed hardware architecture.

TABLE 4. Hardware implementation results.

	[20]	This work
CMOS Technology	TN40G (45 nm)	TN40G (45 nm)
Processing cycles	20	13
Frequency (Hz)	400M	500M
Core Area (mm^2)	2.41	1.21
Gate Equivalence (GE)	1,961K	1,212K
Throughput	20M	38.46M
Power (mW)	unavailable	81.14

cells to work at three different voltage thresholds, i.e., low threshold (low-Vt), regular threshold (regular-Vt), and high threshold (high-Vt). We set the cell with long data path to work at low-Vt, and other cells to work at high-Vt. Working at frequency 500 MHz, our proposed architecture consumes leakage power and total power listed in Table 3. Clearly, this multi-Vt trick leads our hardware architecture to consume less leakage power and total power consumption by 47.12% and 18.86%, respectively.

The implementation statistics and chip layout of our architecture are listed in Table 4 and illustrated in Fig. 7, respectively. The throughput rate for performing collision detection of two triangles is defined by

$$\text{(throughput rate)} = \text{frequency}/\text{(processing cycles)}. \quad (7)$$

Compared with the architecture in [20], the architecture here tests whether the coplanar triangles exist at the early stage. Such design leads the proposed architecture to have shared SA module for computing either 6 or 11 potential separating axes for the coplanar or non-coplanar triangles and achieves gate count reduction. The new design also demonstrates less processing cycles and higher throughput rate over the old design.

The time schedule of the proposed architecture is listed in Fig. 6. Three clock cycles are required for the 3D coordinates of the three vertices of triangle A or B to be input to the architecture. Working at frequency 500 MHz, our architecture can finish the collision detection of two triangles every 3 clock cycles or 6 ns due to the parallel and pipeline structure. For two objects represented by 400 and 400 meshed triangles, respectively, our hardware architecture can finish the collision detection of two such objects, or equivalently 160,000 times of collision detection of two triangles, in 0.96 ms. This time interval is smaller than the required 1 ms for real-time processing of collision detection of two objects. Thus, our architecture can conduct the collision detection of two such objects with real-time processing.

V. CONCLUSION

ASIC realization of collision detection of two objects for the machine tool automation is considered. For the 3D objects represented by meshed triangles in the STL file format, we develop one SAT based algorithm to process the multiple times of collision detection of two triangles. To achieve the required 1 ms for real-time processing of collision detection of two objects, we design hardware architecture with pipeline and parallel processing to determine the separating axes from the multiple potential separating axes. We demonstrate that the improved architecture in this paper outperforms our previous architecture in [20] in finishing the collision detection of two objects. The improved architecture features smaller silicon area, less power consumption, and less time interval to finish the collision detection of two objects. As machine tool automation is essential for the future precision machinery, the first hardware design of collision detection of two objects in this paper is therefore valuable.

ACKNOWLEDGMENT

The authors would like to thank the Taiwan Semiconductor Research Institute (TSRI) for providing technical support over the duration of this work.

REFERENCES

[1] T. D. Tang, "Algorithms for collision detection and avoidance for five-axis NC machining: A state of the art review," *Comput.-Aided Des.*, vol. 51, pp. 1–17, Jun. 2014.

[2] K.-J. Mei and R.-S. Lee, "Collision detection for virtual machine tools and virtual robot arms using the shared triangles extended octrees method," *Int. J. Comput. Integr. Manuf.*, vol. 29, no. 4, pp. 355–373, Apr. 2016.

[3] T. Rudolf, C. Brecher, and F. Possel-Dolken, "Contact-based collision detection—A new approach to avoid hard collisions in machine tools," in *Proc. Int. Conf. Smart Machining Syst.*, 2007, pp. 1–4.

[4] T. D. Tang, "The sweep plane algorithm for global collision detection with workpiece geometry update for five-axis NC machining, *Comput.-Aided Des.*, vol. 39, no. 11, pp. 1012–1024, Nov. 2007.

[5] T. D. Tang, "A new collision avoidance strategy and its integration with collision detection for five-axis NC machining, *Int. J. Adv. Manuf. Technol.*, vol. 81, pp. 1247–1258, Nov. 2015.

[6] J. X. Tian, Z. Liao, and L. Xiao, "A fast collision detection algorithm in virtual NC machine tool, *Appl. Mech. Mater.*, vols. 34–35, pp. 497–501, Oct. 2010.

[7] I. Lee, K. K. Lee, O. Sim, K. S. Woo, C. Buyoun, and J.-H. Oh, "Collision detection system for the practical use of the humanoid robot," in *Proc. IEEE-RAS 15th Int. Conf. Humanoid Robots (Humanoids)*, Nov. 2015, pp. 972–976.

[8] Lih Woei Carpentry Machine Co. *Precision Mini 5-Axis CNC Machine Center*. Accessed: Jan. 7, 2023. [Online]. Available: https://www.lihwoei.com.tw/products-en

[9] R. Kelly and D. Chakravorty. *The Most Common 3D File Formats in 2022-All3DP*. Accessed: Jan. 7, 2023. [Online]. Available: https://all3dp.com/2/most-common-3d-file-formats-model

[10] W. Zhao and J. Sun, "Collision detection research for deformable objects," in *Proc. Int. Conf. Comput. Sci. Electron. Eng.*, Mar. 2012, pp. 557–561.
[11] H.-R. Pakdel and F. Samavati, "Incremental subdivision for triangle meshes," in *Proc. Int. J. Comput. Sci. Eng.*, vol. 31, no. 1, pp. 80–92, Feb. 2007.
[12] O. Tropp, A. Tal, and I. Shimshoni, "A fast triangle to triangle intersection test for collision detection, *Comput. Animation Virtual Worlds*, vol. 17, no. 5, pp. 527–535, Dec. 2006.
[13] D. Olivier and G. Philippe. (2002). *Faster Triangle-Triangle Intersection Tests*. Accessed: Jan. 7, 2023. [Online]. Available: https://hal.inria.fr/inria-00072100/document
[14] L.-Y. Wei, "A faster triangle-to-triangle intersection test algorithm, *Comput. Animation Virtual Worlds*, vol. 25, nos. 5–6, pp. 553–559, Sep. 2014.
[15] M. Montanari, N. Petrinic, and E. Barbieri, "Improving the GJK algorithm for faster and more reliable distance queries between convex objects, *ACM Trans. Graph.*, vol. 36, no. 3, pp. 1–17, Jun., 2017.
[16] S. Gottschalk, "Separating axis theorem," Dept. Comput. Sci., Univ. North Carolina Chapel Hill, Chapel Hill, NC, USA, Tech. Rep., TR96-024, 1996.
[17] S. Gottschalk, M. C. Lin, and D. Manocha, "OBBTree: A hierarchical structure for rapid interference detection," in *Proc. 23rd Annu. Conf. Comput. Graph. Interact. Techn. (SIGGRAPH)*, 1996, pp. 171–180.
[18] K. Erkorkmaz and Y. Altintas, "High speed CNC system design. Part II: Modeling and identification of feed drives, *Int. J. Mach. Tools Manuf.*, vol. 41, no. 10, pp. 1487–1509, Aug. 2001.
[19] A. Raabe, S. Hochgurtel, J. Anlauf, and G. Zachmann, "Space-efficient FPGA-accelerated collision detection for virtual prototyping," in *Proc. Design Autom. Test Eur. Conf.*, Munich, Germany, 2006, pp. 1–6.
[20] Y.-H. Chang, I.-C. Chang-Chien, C.-H. Huang, and Y.-S. Chu, "Design and implementation of real-time collision detection ASIC for machine tools," in *Proc. Int. Conf. Eng., Sci., Ind. Appl. (ICESI)*, Aug. 2019, pp. 1–5.
[21] D. Eberly. *Intersection of Convex Objects: The Method of Separating Axes*. Geometric Tools, Redmond. [Online]. Available: https://www.geometrictools.com/Documentation/MethodOfSeparatingAxes.pdf
[22] S. Friedberg, A. Insel, and L. Spence, *Linear Algebra*. Upper Saddle River, NJ, USA: Prentice-Hall, 1979.
[23] B. Wang, J. Zhou, and T.-H. Kim, "SRAM devices and circuits optimization toward energy efficiency in multi-V_{th} CMOS, *Microelectron. J.*, vol. 46, no. 3, pp. 265–272, 2015.
[24] H. Sjberg and O. Ylinenp, "Collision detection for haptic rendering," M.S. thesis, Dept. Comput. Sci., Umea Univ., Umea, Sweden, 2009.

TSUNG-HSIEN LIU (Member, IEEE) received the B.S. degree in electrical engineering from National Taiwan University, Taiwan, in 1988, and the Ph.D. degree in electrical engineering from the University of Southern California, Los Angeles, CA, USA, in 1998.

He joined as a Faculty Member with the Department of Communications Engineering, National Chung Cheng University, Chiayi, Taiwan, in August 1999, where he is currently a Professor. He has been the Chairperson with the Department of Communications Engineering, National Chung Cheng University, since August 2020. His research interests include applying signal processing techniques to wireless communications, with emphasis on developing fast implementation algorithms, designing baseband hardware architectures, and analyzing statistical performance for the multiple antenna baseband receivers. He has served as the Chairperson for the Tainan Chapter, IEEE Communication Society, from 1998 to 1999.

PO-YI CHEN received the B.S. and M.S. degrees in electrical engineering from National Chung Cheng University, Chiayi, Taiwan, in 2019 and 2021, respectively.

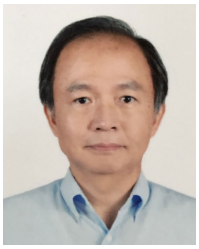
He is currently a Digital IC Design Engineer with Novatek Microeletronics Corporation.

AN-HONG LI received the B.S. degree in electronic engineering from I-Shou University, Kaohsiung, Taiwan, in 2019, and the M.S. degree in electrical engineering from National Chung Cheng University, Chiayi, Taiwan, in 2022.

He is currently working as a Digital IC Design Engineer with Nuvoton Technologies Inc., Hsinchu, Taiwan.

YU-YANG FANG received the B.S. degree in mechanical and electro-mechanical engineering from National Ilan University, Ilan, Taiwan, in 2017, and the M.S. degree in electrical engineering from National Chung Cheng University, Chiayi, Taiwan, in 2022.

He is currently working as a Digital IC Design Engineer with Himax Technologies Inc., Tainan, Taiwan.



RONG-SHINE LIN received the Ph.D. degree in mechanical engineering from the University of Michigan, Ann Arbor, MI, USA, in 1994.

He was a Process Engineer with the Department of Advanced Manufacturing Engineering, Chrysler Corporation, from 1994 to 1995. He was an Assistant Professor with the Department of Industrial Engineering, Rutgers University, New Brunswick, NJ, USA, from 1995 to 1997. He is currently an Associate Professor with the Department of Mechanical Engineering, National Chung Cheng University, Taiwan. His research interests include five-axis CNC machining, additive manufacturing, and robot motions.

YUAN-SUN CHU received the B.S. degree in electrical engineering from Feng Chia University, Taichung, Taiwan, in 1978, and the M.S. and Ph.D. degrees in electrical engineering from Katholieke Universiteit Leuven, Leuven, Belgium, in 1986 and 1991, respectively.

He is currently a Professor with the Department of Electrical Engineering, National Chung Cheng University, Chiayi, Taiwan. His research interests include computer architecture, communication IC design, and low-power IC design.