frontiers in Robotics and AI

ORIGINAL RESEARCH
published: 31 August 2016
doi: 10.3389/frobt.2016.00051

FPGA-Based High-Performance Collision Detection: An Enabling Technique for Image-Guided Robotic Surgery

*Zhaorui Zhang[1,2], Yao Xin[2], Benben Liu[2], Will X. Y. Li[3], Kit-Hang Lee[4], Chun-Fai Ng[4], Danail Stoyanov[5], Ray C. C. Cheung[2] and Ka-Wai Kwok[4]**

[1] Department of Computer Science, The University of Hong Kong, Pokfulam, Hong Kong, [2] Department of Electronic Engineering, City University of Hong Kong, Kowloon Tong, Hong Kong, [3] School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing, China, [4] Department of Mechanical Engineering, The University of Hong Kong, Pokfulam, Hong Kong, [5] Department of Computer Science, Centre for Medical Image Computing, University College London, London, UK

Collision detection, which refers to the computational problem of finding the relative placement or configuration of two or more objects, is an essential component of many applications in computer graphics and robotics. In image-guided robotic surgery, real-time collision detection is critical for preserving healthy anatomical structures during the surgical procedure. However, the computational complexity of the problem usually results in algorithms that operate at low speed. In this paper, we present a fast and accurate algorithm for collision detection between Oriented-Bounding-Boxes (OBBs) that is suitable for real-time implementation. Our proposed *Sweep and Prune* algorithm can perform a preliminary filtering to reduce the number of objects that need to be tested by the classical *Separating Axis Test* algorithm, while the OBB pairs of interest are preserved. These OBB pairs are re-checked by the *Separating Axis Test* algorithm to obtain accurate overlapping status between them. To accelerate the execution, our *Sweep and Prune* algorithm is tailor-made for the proposed method. Meanwhile, a high-performance scalable hardware architecture is proposed by analyzing the intrinsic parallelism of our algorithm and is implemented on FPGA platform. Results show that our hardware design on the FPGA platform can achieve around 8× higher running speed than the software design on a CPU platform. As a result, the proposed algorithm can achieve a collision frame rate of 1 kHz and fulfill the requirement for the medical surgery scenario of *Robot-Assisted Laparoscopy*.

Keywords: OBB, collision detection, *Sweep and Prune*, FPGA, CPU, robotic-assisted laparoscopy, computer vision

OPEN ACCESS

Edited by:
Hongbin Liu,
King's College London, UK

Reviewed by:
Paolo Fiorini,
University of Verona, Italy
Luis Gomez,
University of Las Palmas de Gran Canaria, Spain

****Correspondence:***
Ka-Wai Kwok
kwokkw@hku.hk

Specialty section:
This article was submitted to Biomedical Robotics, a section of the journal Frontiers in Robotics and AI

Received: *30 March 2016*
Accepted: *09 August 2016*
Published: *31 August 2016*

Citation:
Zhang Z, Xin Y, Liu B, Li WXY, Lee K-H, Ng C-F, Stoyanov D, Cheung RCC and Kwok K-W (2016) FPGA-Based High-Performance Collision Detection: An Enabling Technique for Image-Guided Robotic Surgery. Front. Robot. AI 3:51. doi: 10.3389/frobt.2016.00051

1. INTRODUCTION

Current state-of-the-art surgical systems provide stereo vision and tele-operated robotic control of instruments to enable highly dexterous surgical actions in small spaces. However, soft-tissue deformation, restricted workspace, and limited field-of-view impose significant challenges on safe, accurate, and effective surgical interventions. To encounter the problems, there is an increasing

demand for augmenting and enhancing the visual and force-based (haptic) feedback. Advances in computer vision techniques (Stoyanov, 2012; Maier-Hein et al., 2013, 2014) have enabled the real-time 3D surface reconstruction from laparoscopic images to generate improved visualization of the surgical site (Sridhar et al., 2013). Despite the promising information that real-time 3D geometry provides during surgery, the technology has not been utilized to provide haptic feedback due to the lack of real-time collision detection techniques to provide low-level information of interaction between multiple instruments and anatomical structures.

Achieving *in vivo* haptic sensing is difficult during robot-assisted laparoscopic surgery, due to the complex interactions between the surgical instruments and many objects, such as trocar, anatomical structures, and other instruments. It is technically challenging to embed multiple miniaturized force sensors onto different parts of the instrument (Okamura, 2004) due to difficulties in meeting the biocompatibility and surgical sterilization requirements. A promising alternative to hardware sensing is to provide simulated haptic feedback based on potential collision/contact between image-based anatomical models and surgical instruments (Gibson et al., 1997; Bethea et al., 2004; Meijden and Schijven, 2009; Okamura, 2009; Kwok et al., 2013). The simulated haptic feedback can raise surgeon's instant and distinct awareness on the relative configurations between the surgical instruments and the tissue of interest. To achieve this, fast and robust collision detection is the prerequisite. The virtual complex-shape objects at the surgical site will have to be decomposed and generalized into numerous primitives, such as spheres and boxes. After that, these uniform geometric representations are effectively parallelized in order to maintain smooth and continuous haptics at a rate above 1 kHz (Basdogan et al., 2004). This 1 kHz rate is the recognized requirement for providing smooth and continuous haptics during the medical surgery.

Collision detection is the process of determining whether two or more bodies are colliding at one or more points. It is a fundamental technique used in a wide spectrum of applications including computer graphics, virtual prototyping, gaming, haptics (Peterlik and Filipovic, 2011), robot motion planning (Vadakkepat et al., 2008; Sudha and Mohan, 2011), robotic control with constraints (Vachhani et al., 2009), molecular modeling, etc. Such detection is usually employed to prevent virtual objects from penetrating each other by identifying their geometric interface. The penetration prevention can also be important for the surgical safety, as it reduces unnecessary tissue perforation. Spheres, axis aligned bounding boxes (AABBs) and oriented bounding boxes (OBBs) are typically used primitives to represent objects for collision detection. **Figure 1** illustrates the state of space representations of their geometries. The data structure of a sphere is relatively simple, comprising of the radius and center coordinate, for which Chow et al. (2011) has already proposed a Field-programmable gate array (FPGA)-based computational scheme to detect potential collisions. AABB is a cuboid, where the edges are parallel to XYZ coordinates, respectively. The OBB is also a cuboid, but flexibly oriented in 3D. Altomonte et al. (2008) also demonstrates a full force feedback chain in an anatomical environment with a good performance.

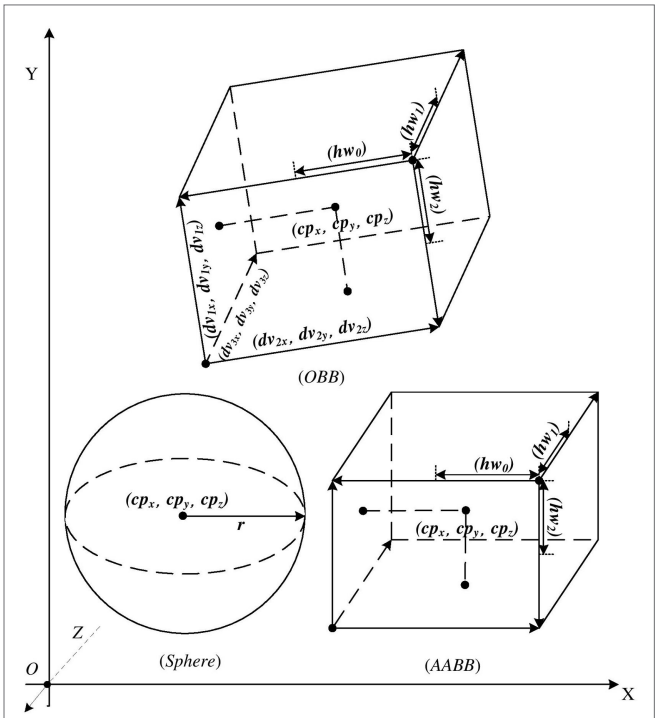

FIGURE 1 | Geometric modeling of three bounding volumes.

During collision detection, the *Sweep and Prune* (SAP) algorithm is an efficient method for preliminary filtering among a large number of cuboids and reducing the amount of objects to be tested. However, *SAP* can only be used in the format of AABBs. Thus, its application is highly restricted on free-form objects in 3D, which their longitudinal axes are seldom aligned with the coordinate axes. Therefore, fast and accurate computation schemes are proposed to process collision detection among OBBs. OBB is a widely used bounding volume in many applications because of its flexible and tight-bound characteristics, despite of its relatively high computational complexity compared to other primitives due to its flexibly oriented edges. For checking the intersection between two OBBs, the most classical algorithm is the *Separating Axis Test* (SAT). However, if the amount of OBB is large, using brute force method to check all overlap statuses between OBBs consume large amount of computing resources and become inefficient. The complexity of brute force method is $O(n^2)$, where n is the number of OBBs, which exceeds the threshold in our scenario. To solve this, we derive a fast and accurate algorithm that allows us to reduce the computational complexity from $O(n^2)$ to sorting algorithm-dependent, which the computational complexity is controlled by our choice of sorting algorithm.

For execution of the algorithm, hardware platform is essential. Field-Programmable Gate Arrays (FPGA) is a hardware platform for engineers to design a special processor to execute their work. An FPGA mainly consists of six components, including programmable I/O units, configurable logic block that consists of LUTs and registers, embedded RAM, rich routing resources, hard IP cores. All these components support the engineers to

design a customized "computer" to execute a special task. The architecture will be designed and loaded into FPGA chip, and the compilation takes several minutes to several hours, depends on the complexity of architecture. Generally, the customized hardware platforms would generally have better performance than the routine hardware platforms in executing a specific task. However, the architectures of most hardware platforms only allow to be written once, and no redesigning of the architecture can be done. FPGA platform, in contrast, has the advantage to supports architecture redesigning features without losing too much performance comparing with other non-redesignable hardware. Moreover, reconfigurable feature of FPGA platform also supports better scalability than other hardware platforms, like ASIC. At the same time, FPGA platform support highly customized processing architectures to fulfill the algorithm-specific requirements, like the other hardware platforms.

General-purpose GPU (Graphics Processing Unit) and FPGA (Field-Programmable Gate Array) have been widely used to accelerate computationally intensive algorithms in a wide range of areas, such as machine learning, image processing, bioinformatics, etc. Such HPC (High-Performance Computing) systems are usually equipped with a CPU and one or more coprocessors as accelerators to cooperate with CPU. As an implementation of SIMD (Single instruction, multiple data), GPUs analyze data as if it is in graphic form, and process using thousands of cores in parallel. One example is CUDA (Compute Unified Device Architecture), which is a parallel computing architecture and programing model developed by Nvidia (Kestur et al., 2010). On the other hand, unlike GPUs, FPGAs do not have any fixed instruction-set architecture. They provide an array of programmable logic blocks and reconfigurable interconnects, which can be routed to perform any complex logic or functions. They also include fast on-chip memory and DSPs for dedicated use (Papadonikolakis et al., 2009). FPGAs are programed using HDL (hardware description language), such as Verilog. Compared with FPGAs, GPUs are easier to program and debug with the high level language and APIs and require less time for design and implementation. However, this implementation easiness comes at a cost. FPGA, which consists of high density arrays of uncommitted logic, is very flexible in hardware infrastructure, and developers can have choice to trade-off resources and performance by adjusting the hardware architecture. With fine-grained bit-wise parallelism, FPGAs can achieve higher performance and lower latency for most algorithms (Brost et al., 2014). The reconfigurable nature of FPGA enables the developers to create a high-performance functional prototype that can emulate and verify solutions. The prototype could also be integrated into the final system (Brost et al., 2014). Besides, FPGA outperforms GPU in terms of less energy consumption and higher performance per joule (Che et al., 2008). Nevertheless, FPGA design can be transformed to ASIC (Application-Specific Integrated Circuit) design with less effort. In our study, the proposed algorithm requires the adjustment of the parallelism in any level to customize its precision or performance. Therefore, FPGA is the platform that meets our requirements and is chosen in our study as the hardware platform.

This paper generally focuses on collision detection, also known as broad phase proximity query, between OBBs. A fast and accurate algorithm is developed to be executed on FPGA platform with a customized hardware architecture for collision detection among numerous OBBs. The algorithm is designed and optimized such that its computational processes can be fully pipelined. The data independence parts are designed in parallel, so as to maximize the advantages of FPGA platform. Temporary memory units are adopted to reduce the data dependence. Besides, the precision of data does not always require the use of the floating point in the algorithm. In this paper, mixed-precision techniques are used to define the data format. In the sorting process, reduced precision data format, which consumes fewer logic resources than high precision one, is used to increase the datapath parallelism; meanwhile, the data precision is appropriately defined based on the flexible input requirement of accuracy. To our knowledge, this is the first work to exploit a reconfigurable computing architecture for collision detection. Overall, the key contributions of this paper include:

1. Development of a fast and accurate algorithm for collision detection among OBBs. It can reduce the complexity from $O(n^2)$ to sorting complexity, n is defined as the number of OBBs. As a result, the ratio of acceleration increases with a growth of the number of objects.
2. Design of a scalable, parallel, and fully pipeline hardware architecture, which accelerates the presented algorithm on an FPGA platform.
3. Implementation of FPGA-based mixed-precision techniques to reduce the complexity of number sorting required for *Improved Sweep and Prune* algorithm.
4. Comparison of the speedup and energy efficiency between our *Improved Sweep and Prune* algorithm and the conventional *Sweep and Prune* algorithm on various computing platforms. It validates the feasibility and practical value of our algorithm in real-time haptic interaction of tele-operated laparoscopic surgery.

2. MATERIALS AND METHODS

2.1. Configuration of the FPGA and CPU Platforms

The FPGA platform was built on a Xilinx Virtex-6 XC6VHX565T FPGA platform with 16,384 datasets. It contains 864 DSP slices (with 25×18 multipliers and 48-bit adder/subtracter/accumulator), 912 intern Block RAMs and 88,560 configurable logic blocks (CLBs), which can configured for logic, arithmetic, ROM, RAM, or data register functions. Xilinx ISE design suite 14 was used for the programing processes on the platform. The code was written in Verilog. The design process using Xilinx ISE consisted of five steps: Design Entry, Synthesis, Implementation, Verification, and Download. The programing model was firstly designed on the ISE design suite, and then the clock cycle was obtained after synthesis. After that, the programing model went through the implementation stage, which involved translation, mapping, placing, and routing. In the step of verification, we checked for the

errors of our code through the waveforms between the input data and output data. Lastly, we downloaded our code into real demo-board of FPGAs. The download time was around a few seconds. The total compiling and loading time of the design process was about 30 min. The price of FPGA was around $6000. The software platform was executed on a CPU platform, with the configuration of Intel *Core*^{TM} i7 CPU 860 @ 2.8 GHz with 4 GB RAM, and the compiler was Visual Studio 2012. No multi-threading was used in the Visual Studio.

2.2. Fast Algorithm for Collision Detection between OBBs

In this section, a collision detection algorithm, which OBBs are generated to represent the tissue and medical devices, is introduced for applications in robot-assisted medical surgery. With the aim to be implemented on FPGA efficiently, its computation was parallelized. The method involved three steps: (i) projection and transformation of the geometric model of OBBs into the state-space representation of AABBs; (ii) reduction of number of candidates in a preliminary selection using the *SAP* algorithm; (iii) acquisition of the overlapping status among the selected OBBs with *SAT*, so as to prescribe the corresponding precision required for further process. **Figure 2** shows the work flow of the presented algorithm.

The related information about Sphere, AABB, and OBB are shown in **Figure 1** and defined as follow:

(1) The coordinate of center point of *Sphere*: (cp_x, cp_y, cp_z). The radius of *Sphere*: r.
(2) The coordinate of center point of *AABB*: (cp_x, cp_y, cp_z). Half length of three sides of *AABB*: (hw_0, hw_1, hw_2).
(3) The coordinate of center point of *OBB*: (cp_x, cp_y, cp_z). Three unit direction vectors along each side of *OBB*: (dv_{00}, dv_{01}, dv_{02}), (dv_{10}, dv_{11}, dv_{12}), (dv_{20}, dv_{21}, dv_{22}) or define them as $dv^{0,1,2}_{0,1,2}$. Half length of three sides of *OBB*: (hw_0, hw_1, hw_2).

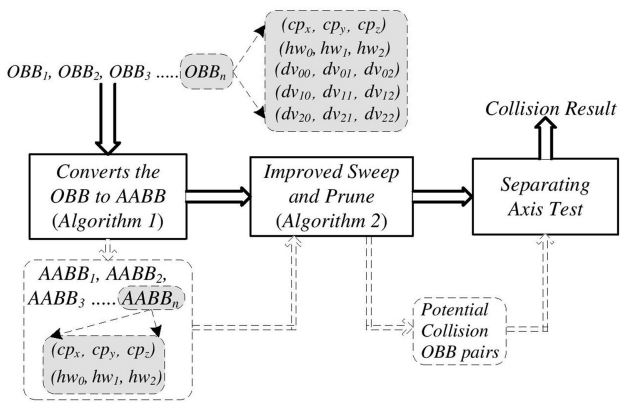

FIGURE 2 | The work flow of proposed algorithm. The dotted arrows indicate the output of one module and input of the next module, while the solid arrows show the data flow of algorithm.

2.2.1. Converting OBB to AABB

Bounding volume was used to bound tissue and medical devices to check the collision status between them in the robot-assisted medical surgery. The *SAP* was an efficient algorithm to reduce the number of bounding box candidates during collision detection. It was defined base on the bounding volume whose edges parallel to the axis, such as the AABB. In order to use *SAP* algorithm to reduce the number of OBB candidates, OBBs were converted to AABBs. This processing and the detailed information about how to convert OBB to AABB were defined in pseudo code of **Algorithm 1**.

2.2.2. Improved Sweep and Prune

In the robot-assisted medical surgery, the collision detection between tissue and medical devices was the crucial part. In robot-assisted laparoscopic surgery, the tissue and laparoscopic grippers were transformed to OBBs for the convenience of overlap-checking between them. *SAP* was a broad phase algorithm for limiting the number of pairs of bounding boxes candidates during the collision detection. This was achieved by sorting the starts *min-bound* and the ends *max-bound* of the bounding volume of each object along a number of arbitrary axes. When the bounding volumes of two objects overlap in all axes, they were flagged to be tested by more precise and time consuming algorithm. **Figure 3** shows the principle of *SAP* algorithm in 1-dimension. *SAP* algorithm was an efficient algorithm to filter highly separated objects and reduces the complexity of collision detection from $O(n^2)$ (*Brute Force*) to the complexity depends on the choice of sorting algorithm.

The *SAP* algorithm can be considered as four steps.

(1) Data initialization: the first step is to store the OBB's maximum and minimum values on *x, y, z* axis into three structure arrays respectively, and tag with an OBB-index and flag. The OBB-index specifies the OBB, which the data belongs to, while the flag specifies if the value is a maximum or a minimum.

Algorithm 1 | Converting OBB to AABB.

1 ***Input:*** (1) $cp_{x,y,z}$: the coordinate of center point of OBB; (2) $dv^{0,1,2}_{0,1,2}$: three direction vectors of OBB; (3) $hw_{0,1,2}$: three half width of sides of OBB.
2 ***Output:*** Data struct ***Point***(*obbname, flag, data,SP*).

```
for 0 ≤ i < obbnumber do
    (X,Y,Z)_{V1-8} ← f(cp_{x,y,z}, dv^{0,1,2}_{0,1,2}, hw_{0,1,2})
    Max(Min)_{x,y,z} ← Max(Min){(X,Y,Z)_{V1-8}}
end for
for 0 ≤ i < obbnumber do
    Point_{x,y,z}[i].data ← Max_{x,y,z}
    Point_{x,y,z}[i].obbname ← i
    Point_{x,y,z}[i].flag ← 1
    SP ← P
end for
for obbnumber ≤ i < (2 × obbnumber) do
    Point_{x,y,z}[i].data ← Min_{x,y,z}
    Point_{x,y,z}[i].obbname ← (i – obbnumber)
    Point_{x,y,z}[i].flag ← 0
    SP ← P
end for
```

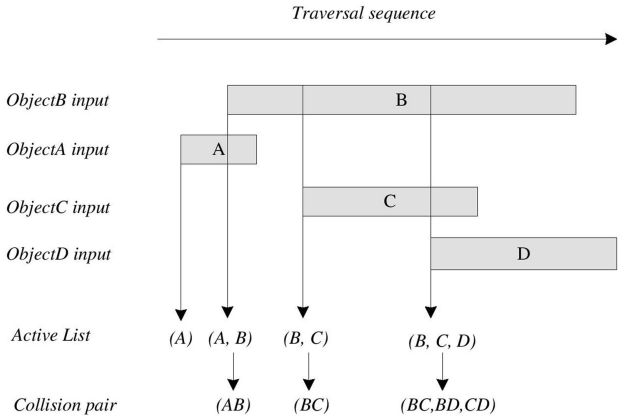

FIGURE 3 | The principle of *Sweep and Prune* in 1-dimension.

Algorithm 2 | Proposed *Sweep and Prune*.

1 Let parameters of $cp_{x,y,z}$ $dv_{0,1,2}^{0,1,2}$, $hw_{0,1,2}$ $max/min_{x,y,z}$, S be a Data Struct: **OBB**.
2 ***Input:*** The Data Struct ***Point***.
3 ***Output:*** OBB pairs that need to rechecked by *SAT*.

```
Sort for three structure Point_{x,y,z} according to their data
for 0 ≤ i < (2 × obbnumber) do
  if point_{x,y,z}[i].flag == 0 then
    j_{x,y,z} ← 1
    while
    point_{x,y,z}[i + j_{x,y,z}].obbname ≠ point_{x,y,z}[i].obbname and point_{x,y,z}[i + j_{x,y,z}].SP
    ≠ point_{x,y,z}[i].SP
    do
      a_{x,y,z} ← point_{x,y,z}[i].obbname
      b_{x,y,z} ← point_{x,y,z}[i + j_{x,y,z}].obbname
      SAT(OBB[a_{x,y,z}],OBB[b_{x,y,z}])
      j_{x,y,z} ← j_{x,y,z} + 1
    end while
  end if
end for
```

(2) Sorting: the second step is to sort the data values in the three arrays separately.
(3) Traversing: the third step is to traverse the sorting result sequence according to its flag. When a minimum value was identified, its OBB-index is stored into an active list. Then, the active list is traversed, and the collision pairs are returned. When a maximum value was identified, the active list is traversed to find the OBB-index, which is the same as the current OBB-index and remove it from active list. After this step, three groups of collision pairs are obtained.
(4) Searching same pair: the fourth step is to search the same pairs among above three groups.

By analyzing the *SAP* algorithm, the following parts were identified and considered as bottlenecks.

(1) In the third step of the *SAP* algorithm, using a linked list to store current OBBs, traverse, insert new members and delete members were time and resources consuming on FPGA platform.
(2) In the fourth step of the *SAP* algorithm, finding the same pairs among three groups was a repetitive work. It consumed a lot of time and resources for searching, especially on FPGA platform.

Since FPGA platform was not efficient for sequential or conditional work, these steps did not execute smoothly on FPGA platform. In this paper, a novel method was proposed to avoid above time and source consuming parts. **Algorithm 2** showed the detail information about the *Improved Sweep and Prune* algorithm.

The first bottleneck was substituted by identifying all OBBs between the maximum value and the minimum value of each OBB. This approach avoided frequent access to the active linked list, which was not efficient on FPGA and CPU platform. When traversing the sorting result sequence, if a minimum value was identified, its OBB-index was printed. The sequence continued to be traversed and the OBB-indices were printed one-by-one before the maximum value of the initial OBB-index was identified. If a maximum value was identified, the sorting result sequence continued to be traversed. In short, for an OBB, all OBB-indices between its minimum value and maximum value were checked. The algorithm considered all OBBs close to it, while ignored ones those are far away. The modification can reduce the traversal complexity considerably.

The second bottleneck was solved by performing the *SAT* algorithm for all pairs on three axes, respectively. Actually, the same pairs among three axes were considered to be checked by the *SAT* algorithm. However, searching for same pairs on three axes was time and resources consuming, which had been replaced in this method. Even more, these three axes were independent in this method and could be executed in parallel to save time.

2.2.3. Separating Axis Test

Tissue and medical devices were generated into OBBs for the sake of using OBBs to substitute them to check their intersection status in the robot-assisted medical surgery. OBB was a bounding box with flexible edge orientations. *SAT* was a classical algorithm for various shapes of bounding boxes to implement the intersection test between them. A number of researches used this algorithm to check the overlapping status between bounding boxes (Redon et al., 2004; Akenine-Moller, 2005). It was developed from the *Separating hyperplane theorem*, a basic theorem of convex analysis. This theorem referred that given two convex objects *A* and *B*, either these two objects were overlapping or there was a separating hyperplane *H*, with *A* on one side of *H* and *B* on the other side.

According to the above description, for overlap test for OBB *A* and OBB *B*, the overlapping status between them should be checked in at most 15 axes. These axes were three coordinate axes of *A*, three coordinate axes of *B* and nine axes perpendicular to an axis from each. If two OBBs were separated on all these 15 axes, they would be considered as separated. If there was an axis they were overlapping on it, these two OBBs would be regarded as intersecting. On each potential separating axis *L*, two OBBs were considered as separated on an axis when the sum of their projected radii was less than the distance between the projections

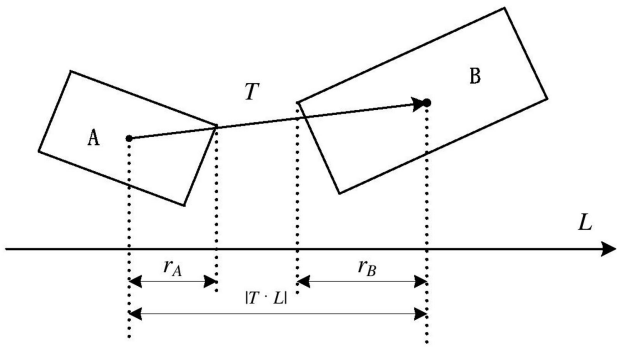

FIGURE 4 | The principle of *Separating Axis Test*. Overlapping between two OBBs was tested by comparing ($r_a + r_b$) with |T·L| on each axis. Operator '·' was defined as dot product.

of their center points. This is illustrated in equation (1). **Figure 4** shows the principle about the algorithm *SAT* on one potential separating axis.

$$|T \cdot L| > r_A + r_B \tag{1}$$

2.2.4. Analyzing Parallelism and Pipelining

In the interest of accelerating the data update rate about the collision detection in the robot-assisted medical surgery, according to the above introduction of proposed algorithm, it was obvious that this algorithm could be speeded up by designing a parallel hardware architecture. In this section, a scalable hardware architecture was developed for the proposed robot-assisted medical surgery collision detection algorithm. The architecture took the advantages of intrinsic parallelism on an FPGA platform to accelerate the algorithm. This acceleration of the algorithm was implemented to fulfill the data update rate requirement of Robot-Assisted Laparoscopy surgery. Programmable hardware devices, such as FPGAs, nowadays provided advanced features and resources for rapid modeling of system-on-chips space (Monmasson and Cirstea, 2007). In order to fully utilize the flexibility of FPGAs, we explored the hardware architecture design in two levels of parallelism, task-level parallelism and pipeline (Li et al., 2011). The hardware design mainly included three parts: *convert OBB to AABB*, *Improved Sweep and Prune*, and *Separating Axis Test*. The structure of each part will be introduced separately in this section. **Figure 5** shows the overview of the whole system architecture, which describes the computing logic within this hardware architecture design. In the practical implementation, registers were implemented between two adjacent steps of calculation to increase the working frequency of the system.

2.2.5. Parallel Hardware Design for Converting OBB to AABB

OBBs were converted to AABBs for the convenience of using *SAP* to reduce the number of bounding box candidates in the proposed robot-assisted medical surgery collision detection algorithm. The hardware architecture took advantages of intrinsic parallelism on an FPGA platform to accelerate this part. There were two major parts included in this architecture. The first part was the hardware design for proposed algorithm to calculate the projection of 8 vertices of OBB on three axes. The second part was the hardware design for finding the max-projection and the min-projection among 8 vertices of OBB on three axes. Both parts took full advantage of the FPGA platform by efficiently pipelined structural design. Furthermore, the processes of calculating the projection for 8 vertices of OBB and the processes of finding the max-projection and min-projection on three axes were performed in parallel.

2.2.6. *Improved Sweep and Prune* Using Adjustable Data Format

The *Improved Sweep and Prune* algorithm in Section 2.2.2 was used to reduce the computation complexity of collision detection algorithm from *Brute Force* method $O(n^2)$ to linear computation complexity in robot-assisted medical surgery. A mixed-precision hardware architecture on FPGA platform was proposed in this part to further reduce the number of OBB candidates, which were far from each other.

The data from Section 2.2.5 was sorted together with their relevant information. Because sorting of floating-point data was time and resources consuming, the fixed-point data representation (reduced precision) was adopted to reduce the complexity of sorting part. The reconfigurability of FPGAs could provide customized trade-off between accuracy and performance. In the proposed algorithm, 3 max-projections and 3 min-projections (X_{max}, X_{min}, Y_{max}, Y_{min}, Z_{max}, Z_{min}) for sorting were converted from floating point to fixed point, of which the bit-width depended on the precision requirement of collision detection result and the speed requirement of hardware architecture running on FPGA platform.

In the hardware design, the *Radix Sort* algorithm was employed to perform sorting because of its non-comparative characteristic and low time and resource consumption. The complexity of *Radix Sort* was $O(kn)$, where k was the number of radix and n was the amount of data to be sorted.

Collision pairs of interest were obtained after this primary selection. After that, these collision pairs were processed using the *SAT*. The processes of data sorting on three axes were designed to work in parallel and fully pipelined. Thus, result could be obtained in each clock cycle on FPGA platform, and the clock cycle could reach *ns* level.

2.2.7. Separating Axis Test

The *SAT* was final step of medical surgery collision detection algorithm to check the intersection status among OBBs, which used to bind the tissue and medical devices. A hardware architecture, which took advantages of intrinsic parallelism on FPGA platform, was proposed to accelerate this part to fulfill the requirement of data update rate of 1 kHz in surgery.

In this part, the overlapping statuses between the remaining OBB candidate pairs on 15 different axes as L_{1-15} were checked. These axes were three coordinate axes of OBB *A*, three coordinate axes of OBB *B* and nine axes perpendicular to an axis from each. These 15 different tests were independent. They were designed to

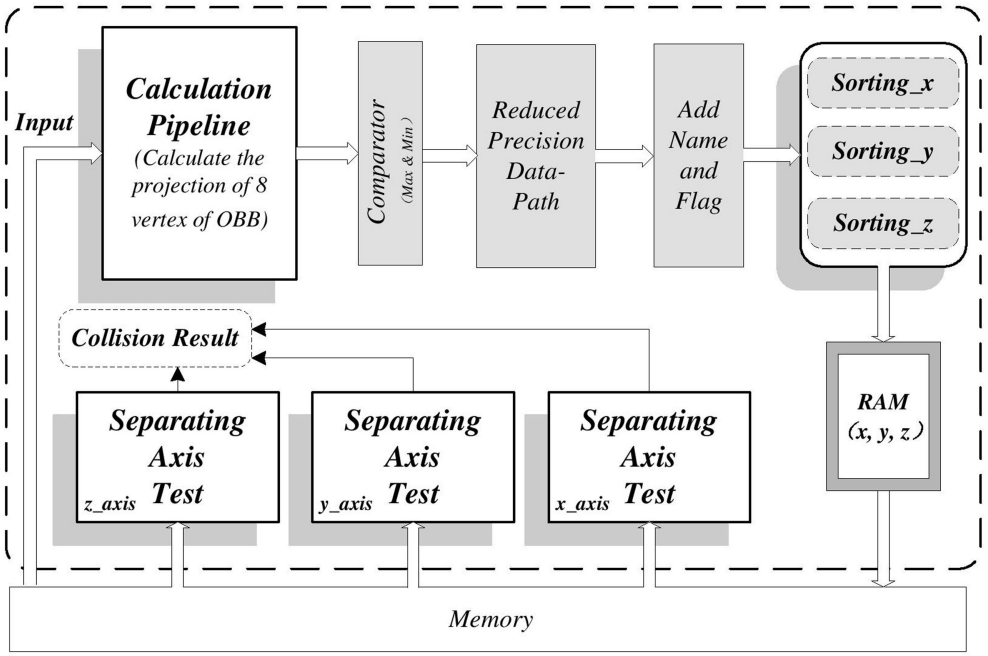

FIGURE 5 | Top-level structure for the hardware architecture includes three major components. (1) Calculation pipeline part: calculate the projection of 8 points of OBB on three axes. (2) Sorting part: sort for the data in algorithm *Sweep and Prune*. (3) Separating Axis Test: test overlapping status between two OBBs on 15 different axes.

run in parallel and were fully pipelined in hardware architecture. This part mainly consisted of arithmetic and comparisons between the related information defined for OBB, especially add, subtract, and multiply. These arithmetic functions were mostly independent, and thus were designed in parallel and fully pipelined.

2.2.8. Analysis for Proposed Algorithm

The *SAP* algorithm was an efficient way to filter far separated OBB pairs in medical surgery collision detection algorithm, but it became less significant as the collision gets dense. The Minimum Period for *Improved Sweep and Prune* algorithm and Minimum Period for the *SAT* algorithm were defined as P_s and P_o, respectively. The number of OBB candidates of original and reserved by *Improved Sweep and Prune* was defined as n and m, respectively. Only when the inequality (2) was true, *Improved SAP* algorithm would be used to perform a preliminary filtering to reduce the number of OBB candidates. The result of inequality showed that if collision was dense, *Brute Force* algorithm would be more efficient.

$$m < (n^2 \times P_o - 32 \times n \times P_s) / P_o \quad (2)$$

2.3. Vision-Based Haptic Feedback in Robotic Assisted Surgery

In robot-assisted surgery, stereo laparoscopes are available and often used by the surgeons to perceive the surgical environment inside the body in 3D. With the stereoscopic video, reliable 3D tissue reconstruction provides detailed geometric information of the surgical site, which can be used to generate haptic feedback or guidance for improving surgical accuracy, consistency, and safety. There are studies demonstrated that haptics can provide cues to the surgeon, allowing protection and avoiding damage to critical structures, such as blood vessels (Carter et al., 2005; Abolhassani et al., 2007; Schostek et al., 2009; Wang et al., 2013).

We used the video from a robotic prostatectomy surgical procedure with the da Vinci surgical robot to recover a 3D surface reconstruction of the surgical site based on stereo vision. The 3D surface of prostate tissue was reconstructed by establishing correspondence of points between the left and right images (Stoyanov, 2012). This allows estimation of the depth of each image points based on triangulation principle, thus forming a set of 3D cloud points consistent with the tissue morphology. The raw 3D point cloud was transformed into a mesh, and we separated the triangle mesh figure of the tissue and two grippers. In the algorithm, two triangles were combined as a rectangle, which could be identified as a face of OBB. After reconstruction, automatic cube tessellation (Barequet and Har-Peled, 2001; Huebner et al., 2008) was applied on the instruments, and the 3D tissue points to construct OBBs for further detection of potential collision, which was considered the prerequisite information for rendering instantaneous guidance to surgical manipulation.

3. RESULTS

3.1. Implementation and Results

In this paper, we mainly focused on the collision detection between OBBs, which applied to the robot-assisted medical surgery. In

order to accelerate the algorithm to satisfy the requirement of data update rate in surgery, a fast and accurate algorithm was proposed and utilized to design a hardware architecture on FPGA platform. The efficiency was compared with its performance on CPU platform. We took advantages of FPGA platform to implement the hardware design in two levels of parallelism. The first was task-level parallelism, which was based on the independence of each task in algorithm. The second was pipelining, which was a deep parallel. In this parallel level, we could obtain the collision result in each digital clock cycle. This digital clock cycle could reach *ns* level.

Our proposed algorithm reduced computation complexity from $O(n^2)$ (n is the number of objects) of brute force method to the complexity of sorting method. In our proposed algorithm, radix sorting algorithm with the complexity of $O(k \times n)$ was used. In order to reduce computation complexity, OBBs were converted to AABBs, and *Improved SAP* algorithm was used to perform a primary filtering to culling OBBs that were far away from each other. This part was designed as a pipelined hardware architecture on FPGA platform. A pipelined hardware architecture for the *SAT* algorithm was also designed, and a minimum clock period of 8.110 ns was achieved. Therefore, a collision result between two OBBs can be obtained in 8.110 ns. **Table 1** shows the resource usages utilization of our FPGA platform hardware design. The *LUTs* were the abbreviation of Look-Up-Tables. They were memory unit and were responsible for storing the value we defined into related memory address. The *Register* was a single memory unit and was responsible for storing a bit data. The *Occupied Slices* was a logic unit, which consist of several *LUTs* and *Register*. The *DSP* was the abbreviation of Digital Signal Processing and defined as a processing unit. The *Wordsize* was used to define the bit-width of data. The *Minimum Period* was defined as the digital clock cycle.

Figure 6 shows the time consumption of proposed medical surgery collision detection algorithm on a CPU platform in a scenario with a large number of objects. The time consumptions for collision detection between different numbers of OBBs using the *SAP* and the *Improved Sweep and Prune* on a CPU platform are shown in **Figure 7**. It demonstrated that the acceleration ratio increased with the growth of the number of objects, and the average speedup for proposed algorithm over original algorithm was about 10×. Next, time consumption for proposed algorithm to process different number of objects with floating-point data on a CPU platform and with 36 bit-width of data on an FPGA platform are compared in **Figure 8**. The results suggested that the hardware design about the proposed algorithm on an FPGA platform could further speedup the algorithm when comparing with the software design on a CPU platform, which the speedup was about 8×. Moreover, **Figure 9** shows the time consumptions with different numbers of OBBs and different bit-widths of data on FPGA platform. The result indicated a scalable performance of our hardware architecture. In addition, low power consumption and highly portable are very important features in medical surgery, which reduce the system footprint and improve the surgical ergonomics inside operating theater. Compare to the size of PC or other larger medical devices, FPGA platform composed of *cm* level chips and was more portable. Nevertheless, the power consumption of our FPGA platform was 7.249 w. Comparing with the 95 w power consumption of software design on CPU, our FPGA platform design was extremely energy efficient. **Table 2** shows the resource utilization of hardware implementation of the proposed algorithm with 16384 datasets on our Xilinx Virtex-6 FPGA platform.

TABLE 1 | Resource utilization of *SAT*.

Name	Used	Available	Utilization (%)
LUT	31,504	354,240	8
Register	30,543	708,480	4
Occupied Slices	15,491	88,560	17
DSPs	525	864	60
Wordsize		Single precision	
Minimum period		8.110 ns	

3.2. Qualitative Example Results for Robotic Surgery

The positions and orientations of both left and right laparoscopic instruments relative to the coordinate frame of camera were computed given the robot kinematics along with its joint configuration and hand-eye calibration. The prostate gland was visualized with image pair obtained by a stereo laparoscope. **Figures 10A,B** illustrate the surgical site for a typical robot-assisted laparoscopy, radical prostatectomy. **Figure 10C** shows a surface mesh of such cloud points textured with the tissue image, of which its anatomical margin is of interest for the surgeon to carry out a dissection, while preserving nearby blood vessels, canals and nerves. After reconstruction and automatic cube tessellation, two triangles were combined into rectangle, and three sets of OBBs were constructed based on these rectangles to represent the tissue and two laparoscopic grippers (**Figure 10D**). As a result, number of OBBs equaled with the number of rectangles. Three sets of OBBs were

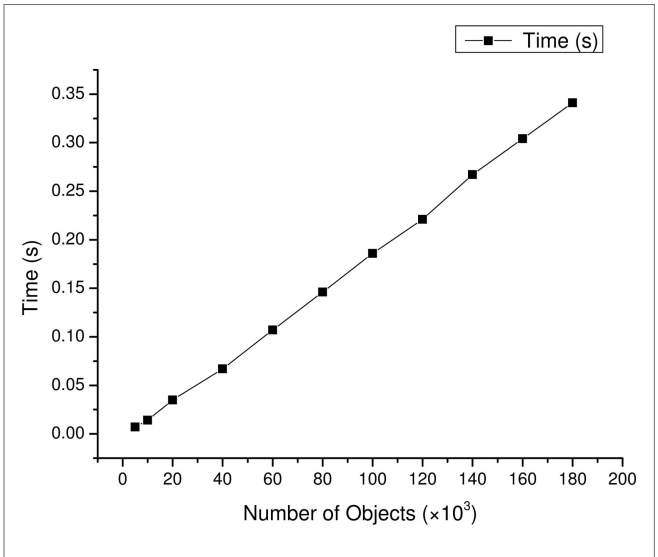

FIGURE 6 | The time consumption for collision detection on a CPU platform with varying number of objects.

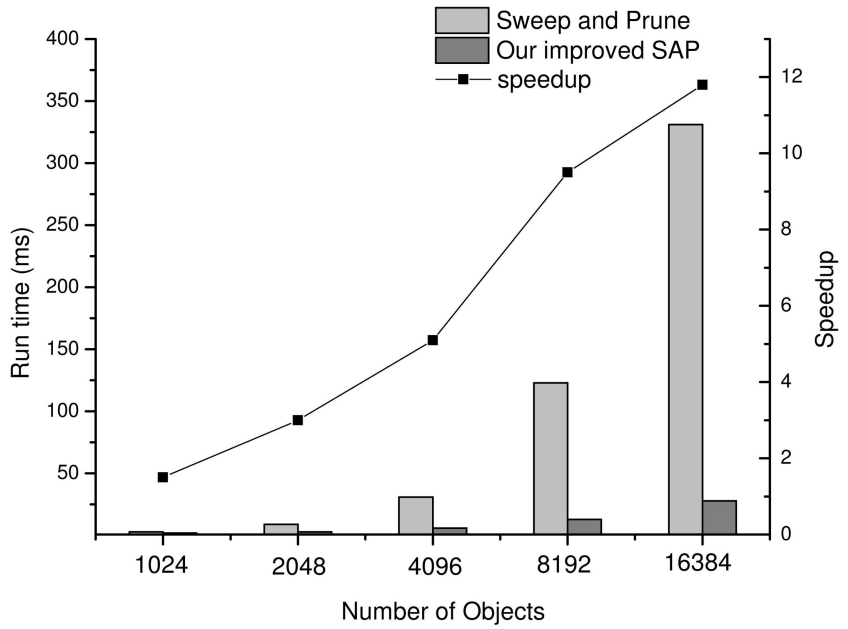

FIGURE 7 | The comparison between the algorithm *Sweep and Prune* and the *Improved Sweep and Prune* on time consumption for CPU platform-based collision detection.

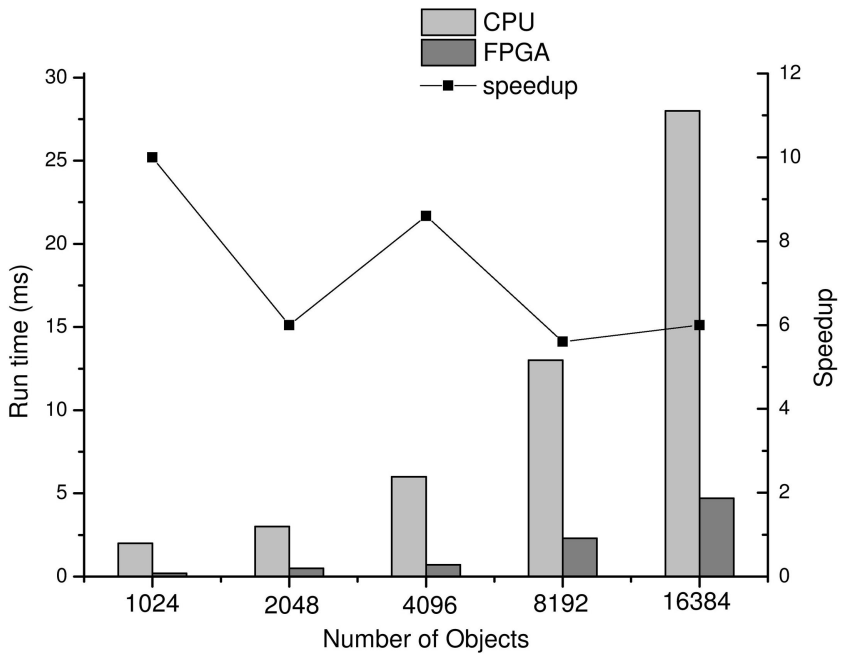

FIGURE 8 | The comparison between our FPGA platform and CPU platform on time consumption for collision detection using our proposed algorithm.

generated to represent the tissue and two laparoscopic grippers for the purpose of checking the overlapping status between them, which the tissue and the grippers consisted of 17.5 k (10 k + 7.5 k) OBBs. We applied our collision detection method on these 17.5 k OBBs using a data bit-width of 20. According to the experimental results in Section 3, we could achieve around 1 kHz rate feedback update. This update rate was able to meet the requirements of robotic assisted surgery and robotic control, which usually

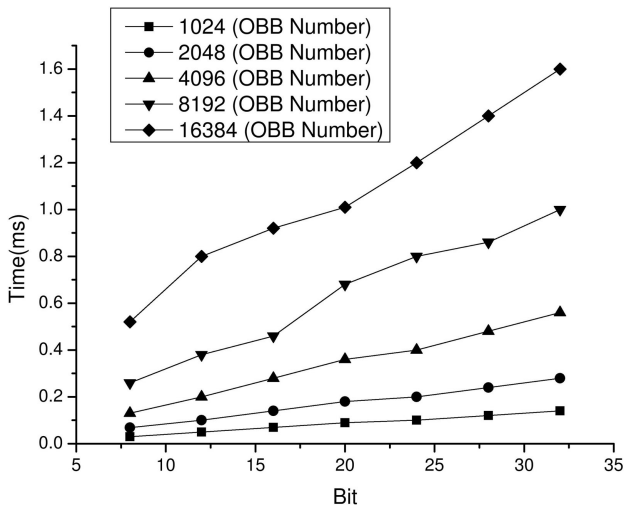

FIGURE 9 | The time consumption of our FPGA platform in processing different number of OBBs with different input data bit-width.

TABLE 2 | Resource utilization of proposed algorithm.

Name	Used	Available	Utilization (%)
LUT	66,061	354,240	18
Register	47,634	708,480	7
Occupied Slices	31,833	88,560	36
DSPs	630	864	72
RAM	705	912	77
Minimum period		8.509 ns	

operated at high frame rates for smooth perception. Our result indicated that the proposed algorithm and the parallel, fully pipelined architecture on our FPGA platform are very promising for applications in image-guided, haptic-enabled robotic surgery.

4. DISCUSSION

In this study, we analyzed the *Sweep and Prune* algorithm and found out the time and resources consuming parts. Based on the findings, we proposed modifications on the *Sweep and Prune* algorithm and tested its performance in the CPU platform. Our results demonstrated that the *Improved Sweep and Prune* algorithm can speed up the collision detection on computing platform when comparing with the conventional *Sweep and Prune* algorithm. The speedup rate would increase proportionally with the number of objects. These evidences indicated that we successfully spotted out the bottleneck on Traversing and Searching the Same Pair steps. Moreover, our novel modifications on the *Sweep and Prune* algorithm, as mentioned in Section 2.2.2, solved these bottlenecks effectively, through minimizing the traversal complexity and elimination of the repetitive works.

After the validation of our *Improved Sweep and Prune* algorithm, we analyzed and parallelized the execution of the different components of our collision detections algorithms. In **Algorithm 1**. *Converts the OBB to AABB*, it consisted of several components. The components of calculating the projection of 8 vertices, finding maximum value and minimum value on three axes, writing the data and its relative information into structure *Point* on three axes, the loop from 0 to *obbnumber* and from *obbnumber* to $2 \times obbnumber$ were independent, respectively, and could be executed in parallel. In **Algorithm 2** *Improved Sweep and Prune*, the components of traversing the sorting result sequence on three axes were independent and could be executed in parallel. In *Separating Axis Test* algorithm, the components of the 9 loops of calculation for rotation matrix R_{ij} and using *Separating Axis Test* to check the overlapping status between two OBBs on 15 different axes were independent, respectively, and could be executed in parallel. The effects of parallelization were analyzed by comparison between the execution of our *Improved Sweep and Prune* algorithm on the CPU platform and our FPGA platform with the customized parallelization architecture. Our FPGA platform largely reduced the run time for the collision detection. This indicated that our parallel and fully pipeline hardware architecture on FPGA platform effectively accelerated the execution of presented algorithms.

Nevertheless, in contrast to the rigid instruction set of GPUs and CPUs, FPGA platform provides the freedom for bit-width optimization for specific function (Lee et al., 2006). We analyzed the performance of our algorithms with different bit-width of data on our FPGA platform and found that the reduction of data bit-width proportionally reduced the execution time of our collision detection algorithm. The novel use of adjustable bit-width of FPGA platform on collision detection provides a scalable performance based on of requirement on precision and update rate, which is important for optimizing its functionality on robot-assisted Minimally Invasive Surgery (MIS).

In actual generation of haptics interaction, a fast, accurate, and reliable collision detection is the prerequisite. Virtual Fixtures, also known as active constraints (Bowyer et al., 2014), is one of the typical control approaches in robot-assisted surgery, which assists operators by providing manipulation guidance through force feedback. In particular, it can generate resistive force to prevent surgical tools from entering a virtual forbidden region (Abbott et al., 2007), thus reducing the chance of undesired damage to delicate tissue. The haptics interaction relies on reliable collision detection between surgical tools and the complicated patient anatomy. Such process is commonly regarded as the computational bottleneck because of the tight requirement on updating rate (>1 kHz) as well as the resolution. By utilizing the highly reconfigurable architecture of FPGA, our novel computation technique can resolve this bottleneck by efficient pipelining and parallelization, hence enables large-scale collision detection at high frequency.

In robot-assisted MIS, recovering the underlying 3D structure of the operating field *in vivo* is of importance to the registration of pre-operative imaging data with the surgical field-of-view for providing dynamic active constraints and motion compensation (Stoyanov et al., 2008). The recovery of 3D tissue structure and morphology during robot-assisted surgery is a crucial step toward accurate deployment of surgical guidance and control

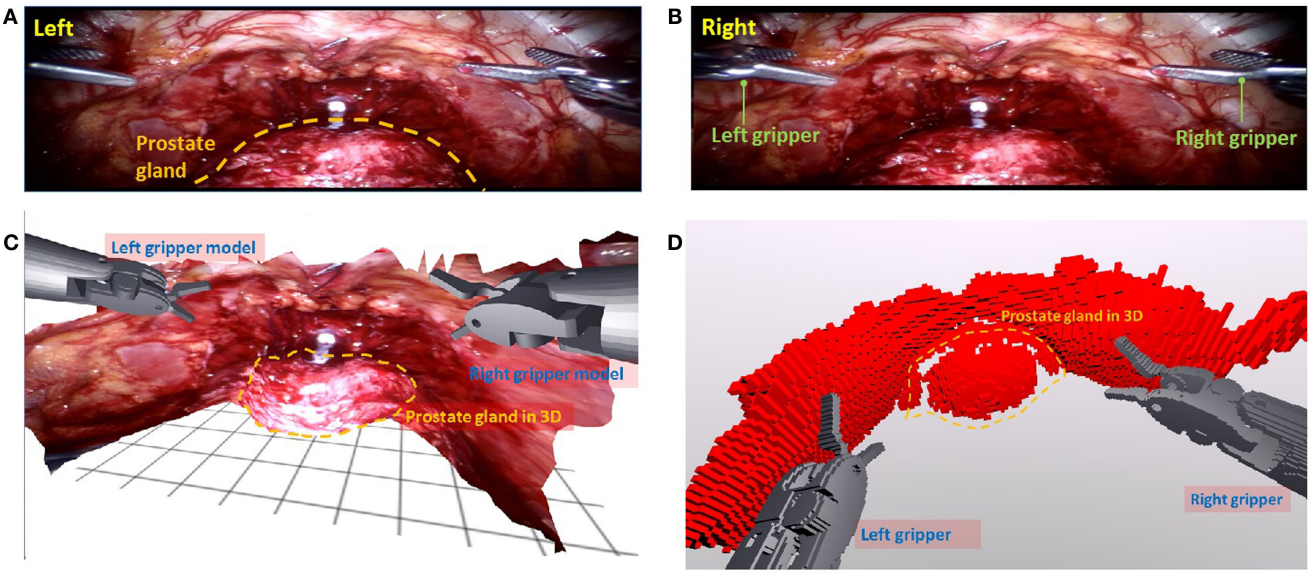

FIGURE 10 | Transformation of the surgical site into OBBs based on stereo vision from the da Vinci surgical robot. The left **(A)** and right **(B)** camera images were obtained using a stereo laparoscope of da Vinci Surgical System Si. **(C)** A 3D mesh outlining the tissue morphological structure. CAD models of both instruments were rendered based on their known kinematic configuration. **(D)** Three sets of OBBs were generated by a rapid tessellation of the mesh and CAD models. The tissue and two laparoscopic grippers were represented by 11,227 and 39,732 OBBs, respectively.

techniques in minimally invasive therapies (Stoyanov et al., 2010). The recovery of 3D information from stereo images remains the classic problem in computer vision, and many research focuses in this area. A number of researches were done to find solution for identification of the unique correspondence of image primitives across the stereo image pair using a calibrated stereo rig, and there were reviews providing summaries of progress in this area (Scharstein and Szeliski, 2002; Brown et al., 2003; Stoyanov et al., 2004). Our qualitative example only required an update rate of around 25 Hz for the 3D reconstruction, which was far lower than update rate of many application algorithms. The update rate of 25 Hz was just for showing the 3D reconstruction and letting the surgeon to check through the 3D display. However, in the practical application, such as vision-based haptic feedback in robotic assisted surgery, the requirement of update rate will be above 1 kHz (Siciliano and Khatib, 2008), which is much higher than 25 Hz. Thus, the significance of the speedup in the update rate of application algorithm will be much higher and essential.

In order to accelerate the collision detection rate, OBBs were used as the bounding box for detection of the collision status, as they were regularly shaped and consumed less computing resources. Although the geometries of the organ and the instruments were covered completely by the OBBs and their collisions were avoided, the limitation of the current model was the generation of a very coarse surface, which did not precisely follow the organ surface. In the practical situation, medical specifications on resolution will be flexible depends on surgical requirements in terms of precision and updating rate. To meet the requirements, FPGA is readily run in cascade to improve its parallel processing performance, according to the previous study (Chen and Dinavahi, 2013; Sano et al., 2014; Njiki et al., 2016). Based on this, if higher updating rate and precision are required, our FPGA platform will be readily scalable with more FPGA boards to run in cascade for fulfilling a better precision requirement, and hence, a better resolution. Moreover, a study introduced a rapid collision prediction technique to detect likely to collide region (Kim et al., 2002). Our algorithm may also exploit the technique to refine the radius for collision detection. After that, the resolution of our collision detection algorithm would be even further improve in the interested region.

Another important feature of the FPGA platform is that the pipelining and parallel architecture of FPGA can grantee the collision detection module to execute independently. Therefore, the speed bottleneck of other components will not affect the 1 kHz execution speed of our collision detection module. Theoretically, when integrated into a complete force-feedback generation process, our algorithm can maintain the 1 kHz speed, as long as the data input rate equal or high than 1 kHz. There are studies that proof the feasibility of implementing the FPGA-based architecture for acceleration of CPU platform performance on different purposes (Smach et al., 2007; Fons et al., 2013). Nevertheless, we can also exploit the scalability of our FPGA platform by adjusting the OBB numbers and bit-width that used to define the OBBs in our algorithm to further improve the execution speed.

Previously, some studies attempted to fulfill a certain real-time requirement for collision detection among the bounding boxes using GPU. Avril et al. (2011) presented a technique that dynamically adapt the first step (broad phase) of the collision detection process on GPU platform during simulation. Pabst et al. (2010) proposed a hybrid CPU/GPU collision detection

technique for rigid and deformable objects based on spatial subdivision. Liu et al. (2010) demonstrated an algorithm of SAP for collision detection between very large numbers of moving bodies using GPUs. Govindaraju et al. (2006) provided a collision culling implemented on a GPU platform. Nevertheless, fixed bit-width will consume more resources, and branch divergence may kill the performance. Compared to FPGA, however, GPU-based methods do not perform well in power consumption and achieve the real-time processing rate requirement of 1 kHz with a lower number of tetrahedra (1300 tetrahedra) (Courtecuisse et al., 2010), comparing with our study (17.5 k OBBs). Another study presented a GPU-based approach to detect collision of items of 14.4 k, 2 k, and 1.7 k triangles with a processing rate of 153 Hz, which cannot meet the medical specification of 1 kHz (Mainzer and Zachmann, 2015). Nevertheless, Altomonte et al. (2008) demonstrated the realization of a surgical simulation with haptic feedback at over 1 kHz rate, under a very high resolution. To achieve the comparable resolution, the previously mentioned FPGA cascading method has to be investigated and applied.

In contrast to GPU computation, FPGA platforms allow highly customized processing architectures according to algorithm specific requirements. FPGAs deploy a reconfigurable computing architecture so that both parallelism and precision of data structure can be customized to enable independent data flow and reduce the processing time. For scenarios involving the collision detection of a large number of objects, FPGA platform is a suitable implementation choice to construct a complete algorithm pipeline. As a result, collision detection can be performed once at a clock cycle, which is a single electronic pulse of the CPU. The clock cycle of the FPGA platform can reach the *ns* level faster than the GPU platform. Each cycle can process one computing operation, such as computing, accessing memory, or writing data to memory. With the bit-wise operations on FPGA, the FPGA data format can be flexibility defined upon the requirement of the algorithm, while GPUs and CPUs are disadvantaged for algorithms amenable to bit-width modifications (Cope et al., 2005). Mixed-precision (Chow et al., 2011) computation is also a technique to save computing resources and speedup algorithms by reducing precision below the floating-point bit-width, particularly for objects that are distant to each other and for which approximate computation is sufficient. In addition, FPGA platforms can be self-contained embedded systems without the need of PCI Express connectivity like GPU boards. In our study, the hardware architecture of our FPGA platform was further designed to be scalable. With more resources on the FPGA platform, the proposed medical surgery collision detection algorithm could process a larger number of collision bounding volume candidates with little loss of data update rate. The design could be scaled-up and scaled-down with the utilization of different number of BRAMs. The data processing power could be increased by a larger number of BRAMs. For example, implementing the hardware architecture on a Virtex-7 platform would increase the amount of processing data with little degradation of frame rates, because the main action of sorting part was the data transfer between BRAMs, and data transfer was not a complex operation on an FPGA platform. Our study also demonstrated that the power consumption of FPGA platform (8 W) was also much less than that of a GPU in a computer platform in another study (Collange et al., 2009), which the power consumption of GPU would be about 110 W during operation. When adding up the power consumption of CPU (about 90 W), the total power consumption of GPU platform would be as high as 200 W. The low power consumption property of FPGA allows the use of portable computation and reduces surgical robotics system footprint, thus improving the surgical ergonomics inside operating theater.

4.1. Conclusion and Future Work

The current study has proposed a novel fast and accurate collision detection algorithm among OBBs with potential applications in medical surgery. The inherited computational complexity of collision detection has been a major bottleneck for many practical applications that require real-time feedback, such as robot-assisted surgery. Our algorithm efficiently reduces the computation complexity comparing with the brute force method. Moreover, a novel mixed-precision hardware architecture on FPGA platform, designed based on *SAP* and *SAT* algorithm, is also built to further accelerate the data update rate of the proposed algorithm for the purpose of robot-assisted medical surgery. Our study illustrates that the combination of our proposed algorithm and our hardware architecture on FPGA platform is about 8× more efficient than its execution on CPU platform. As a result, the frame update rate can reach up to 1 kHz, even in a large-scale scenario, which certificated by the data extracted from robot-assisted laparoscopic surgery. This frame update rate can also satisfy the requirement and extend its use to many other practical applications, like camera, robot-assisted minimally invasive surgery, medical imaging, and interoperative, etc. In the next stage, we will also explore the method for high-performance collision detection of deformable objects. Prior to implementation on surgical systems, integration of the proposed fast collision detection system with a haptic rendering engine used in computer surgical simulation will be essential to conduct user experiment and understand the role of our developed system in a computational approach. The computer surgical simulation can help surgeons to learn through virtual surgical tasks and rehearsals involving interaction between instruments and tissues.

AUTHOR CONTRIBUTIONS

In this work, all the authors contributed to the conception of FPGA-based high-performance collision detection, the analysis and interpretation of the data acquired, and the preparation of the manuscript.

FUNDING

This work is supported in parts by the Croucher Foundation, the Research Grants Council (RGC) of Hong Kong.

REFERENCES

Abbott, J. J., Marayong, P., and Okamura, A. M. (2007). *Haptic Virtual Fixtures for Robot-Assisted Manipulation*. Berlin, Heidelberg: Springer, 49–64.

Abolhassani, N., Patel, R., and Moallem, M. (2007). Needle insertion into soft tissue: a survey. *Med. Eng. Phys.* 29, 413–431. doi:10.1016/j.medengphy.2006.07.003

Akenine-Moller, T. (2005). "Fast 3D triangle-box overlap testing," in *ACM SIGGRAPH 2005 Courses* (New York, NY: ACM). SIGGRAPH '05.

Altomonte, M., Zerbato, D., Botturi, D., and Fiorini, P. (2008). "Simulation of deformable environment with haptic feedback on GPU," in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems* (Nice: IEEE), 3959–3964.

Avril, Q., Gouranton, V., and Arnaldi, B. (2011). "Dynamic adaptation of broad phase collision detection algorithms," in *2011 IEEE International Symposium on VR Innovation (ISVRI)* (Singapore: IEEE), 41–47.

Barequet, G., and Har-Peled, S. (2001). Efficiently approximating the minimum-volume bounding box of a point set in three dimensions. *J. Algorithms* 38, 91–109. doi:10.1006/jagm.2000.1127

Basdogan, C., De, S., Kim, J., Muniyandi, M., Kim, H., and Srinivasan, M. (2004). Haptics in minimally invasive surgical simulation and training. *IEEE. Comp. Graphics Appl.* 24, 56–64. doi:10.1109/MCG.2004.1274062

Bethea, B. T., Okamura, A. M., Kitagawa, M., Fitton, T. P., Cattaneo, S. M., Gott, V. L., et al. (2004). Application of haptic feedback to robotic surgery. *J. Laparoendosc. Adv. Surg. Tech.* 14, 191–195. doi:10.1089/1092642041255441

Bowyer, S. A., Davies, B. L., and Rodriguez y Baena, F. (2014). Active constraints/virtual fixtures: A survey. *IEEE Trans. Robot.* 30, 138–157. doi:10.1109/TRO.2013.2283410

Brost, V., Yang, F., and Meunier, C. (2014). Flexible vliw processor based on fpga for efficient embedded real-time image processing. *J. Real Time Image Proc.* 9, 47–59. doi:10.1007/s11554-012-0321-2

Brown, M., Burschka, D., and Hager, G. (2003). Advances in computational stereo. *IEEE Trans. Pattern Anal. Mach. Intell.* 25, 993–1008. doi:10.1109/TPAMI.2003.1217603

Carter, T. J., Sermesant, M., Cash, D. M., Barratt, D. C., Tanner, C., and Hawkes, D. J. (2005). Application of soft tissue modelling to image-guided surgery. *Med. Eng. Phys.* 27, 893–909. doi:10.1016/j.medengphy.2005.10.005

Che, S., Li, J., Sheaffer, J. W., Skadron, K., and Lach, J. (2008). "Accelerating compute-intensive applications with GPUs and FPGAs," in *Symposium on (IEEE) SASP 2008. Application Specific Processors, 2008* (California: Application Specific Processors, 2008), 101–107.

Chen, Y., and Dinavahi, V. (2013). Multi-FPGA digital hardware design for detailed large-scale real-time electromagnetic transient simulation of power systems. *IET. Gen. Transmiss. Distrib.* 7, 451–463. doi:10.1049/iet-gtd.2012.0374

Chow, G., Kwok, K. W., Luk, W., and Leong, R. (2011). "Mixed precision processing in reconfigurable systems," in *2011 IEEE 19th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM)* (Salt Lake City: IEEE), 17–24.

Collange, S., Defour, D., and Tisserand, A. (2009). "Power Consumption of GPUs from a Software Perspective," in *Computational Science – ICCS 2009: 9th International Conference Baton Rouge, LA, USA, May 25–27, 2009 Proceedings, Part I* (Berlin, Heidelberg: Springer), 914–923.

Cope, B., Cheung, P. Y. K., Luk, W., and Witt, S. (2005). "Have GPUs made FPGAs redundant in the field of video processing?" in *Proceedings. 2005 IEEE International Conference on Field-Programmable Technology, 2005* (Singapore: IEEE), 111–118.

Courtecuisse, H., Jung, H., Allard, J., Duriez, C., Lee, D. Y., and Cotin, S. (2010). Gpu-based real-time soft tissue deformation with cutting and haptic feedback. *Prog. Biophys. Mol. Biol.* 103, 159–168. doi:10.1016/j.pbiomolbio.2010.09.016

Fons, F., Fons, M., Cantó, E., and López, M. (2013). Real-time embedded systems powered by fpga dynamic partial self-reconfiguration: a case study oriented to biometric recognition applications. *J. Real Time Image Proc.* 8, 229–251. doi:10.1007/s11554-010-0186-1

Gibson, S., Samosky, J., Mor, A., Fyock, C., Grimson, E., Kanade, T., et al. (1997). "Simulating arthroscopic knee surgery using volumetric object representations, real-time volume rendering and haptic feedback," in *CVRMed-MRCAS'97, Vol. 1205 of Lecture Notes in Computer Science*, eds J.Troccaz, E.Grimson, and R.Msges (Berlin Heidelberg: Springer), 367–378.

Govindaraju, N. K., Lin, M. C., and Manocha, D. (2006). Fast and reliable collision culling using graphics hardware. *IEEE Trans. Vis. Comput. Graph* 12, 143–154. doi:10.1109/TVCG.2006.29

Huebner, K., Ruthotto, S., and Kragic, D. (2008). "Minimum volume bounding box decomposition for shape approximation in robot grasping," in *IEEE International Conference on Robotics and Automation, 2008. ICRA 2008* (California: IEEE), 1628–1633.

Kestur, S., Davis, J., and Williams, O. (2010). "Bias comparison on FPGA, CPU and GPU," in *VLSI (ISVLSI), 2010 IEEE Computer Society Annual Symposium on* (Lixouri: IEEE), 288–293.

Kim, J., De, S., and Srinivasan, M. A. (2002). "Computationally efficient techniques for real time surgical simulation with force feedback," in *Haptic Interfaces for Virtual Environment and Teleoperator Systems, 2002. HAPTICS 2002. Proceedings. 10th Symposium on* (Washington, DC: IEEE), 51–57.

Kwok, K.-W., Tsoi, K. H., Vitiello, V., Clark, J., Chow, G., Luk, W., et al. (2013). Dimensionality reduction in controlling articulated snake robot for endoscopy under dynamic active constraints. *IEEE Trans. Robot.* 29, 15–31. doi:10.1109/TRO.2012.2226382

Lee, D.-U., Gaffar, A. A., Cheung, R. C., Mencer, O., Luk, W., and Constantinides, G. A. (2006). Accuracy-guaranteed bit-width optimization. *IEEE Trans. Comp. Aided Design Integr. Circ. Syst.* 25, 1990–2000. doi:10.1109/TCAD.2006.873887

Li, Y., Yao, Q., Tian, B., and Xu, W. (2011). "Fast double-parallel image processing based on FPGA," in *2011 IEEE International Conference on Vehicular Electronics and Safety (ICVES)* (Beijing: IEEE), 97–102.

Liu, F., Harada, T., Lee, Y., and Kim, Y. J. (2010). "Real-time collision culling of a million bodies on graphics processing units," in *ACM SIGGRAPH Asia 2010 Papers* (New York, NY: ACM), 154:1–154:8. SIGGRAPH ASIA '10.

Maier-Hein, L., Groch, A., Bartoli, A., Bodenstedt, S., Guillaume, B., Chang, P., et al. (2014). Comparative validation of single-shot optical techniques for laparoscopic 3d surface reconstruction. *IEEE Trans. Med. Imaging* 33, 1913–1930. doi:10.1109/TMI.2014.2325607

Maier-Hein, L., Mountney, P., Bartoli, A., Elhawary, H., Elson, D., Groch, A., et al. (2013). Optical techniques for 3d surface reconstruction in computer-assisted laparoscopic surgery. *Med. Image Anal.* 17, 974–996. doi:10.1016/j.media.2013.04.003

Mainzer, D., and Zachmann, G. (2015). "Collision detection based on fuzzy scene subdivision," in *GPU Computing and Applications* (Singapore: Springer Singapore), 135–150.

Meijden, O., and Schijven, M. (2009). The value of haptic feedback in conventional and robot-assisted minimal invasive surgery and virtual reality training: a current review. *Surg. Endosc.* 23, 1180–1190. doi:10.1007/s00464-008-0298-x

Monmasson, E., and Cirstea, M. (2007). FPGA design methodology for industrial control systems-a review. *IEEE Trans. Ind. Electron.* 54, 1824–1842. doi:10.1109/TIE.2007.898281

Njiki, M., Elouardi, A., Bouaziz, S., Casula, O., and Roy, O. (2016). A multi-FPGA architecture-based real-time TFM ultrasound imaging. *J. Real Time Image Proc.* 1–17. doi:10.1007/s11554-016-0563-5

Okamura, A. M. (2004). Methods for haptic feedback in teleoperated robot-assisted surgery. *Ind. Robot Int. J.* 31, 499–508. doi:10.1108/01439910410566362

Okamura, A. M. (2009). Haptic feedback in robot-assisted minimally invasive surgery. *Curr. Opin. Urol.* 19, 102. doi:10.1097/MOU.0b013e32831a478c

Pabst, S., Koch, A., and StraBer, W. (2010). Fast and scalable CPU/GPU collision detection for rigid and deformable surfaces. *Comp. Graphics Forum* 29, 1605–1612. doi:10.1111/j.1467-8659.2010.01769.x

Papadonikolakis, M., Bouganis, C.-S., and Constantinides, G. (2009). "Performance comparison of GPU and FPGA architectures for the SVM training problem," in *Field-Programmable Technology, 2009. FPT2009. International Conference on* (Sydney: IEEE), 388–391.

Peterlik, I., and Filipovic, J. (2011). Distributed construction of configuration spaces for real-time haptic deformation modeling. *IEEE. Trans. Ind. Electron.* 58, 3205–3212. doi:10.1109/TIE.2009.2032438

Redon, S., Kim, Y. J., Lin, M. C., and Manocha, D. (2004). Fast continuous collision detection for articulated models. 145–156. SM '04.

Sano, K., Hatsuda, Y., and Yamamoto, S. (2014). Multi-fpga accelerator for scalable stencil computation with constant memory bandwidth. *IEEE Trans. Parallel Distributed Syst.* 25, 695–705. doi:10.1109/TPDS.2013.51

Scharstein, D., and Szeliski, R. (2002). A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *Int. J. Comput. Vis.* 47, 7–42. doi:10.1023/A:1014573219977

Schostek, S., Schurr, M. O., and Buess, G. F. (2009). Review on aspects of artihcial tactile feedback in laparoscopic surgery. *Med. Eng. Phys.* 31, 887–898. doi:10.1016/j.medengphy.2009.06.003

Siciliano, B., and Khatib, O. (2008). *Springer Handbook of Robotics*. Springer Science & Business Media.

Smach, F., Miteran, J., Atri, M., Dubois, J., Abid, M., and Gauthier, J.-R. (2007). An FPGA-based accelerator for Fourier descriptors computing for color object recognition using SVM. *J. Real Time Image Process.* 2, 249–258. doi:10.1007/s11554-007-0065-6

Sridhar, A. N., Hughes-Hallett, A., Mayer, E. K., Pratt, P. J., Edwards, P. J., Yang, G.-Z., et al. (2013). Image-guided robotic interventions for prostate cancer. *Nat. Rev. Urol.* 10, 452–462. doi:10.1038/nrurol.2013.129

Stoyanov, D. (2012). Surgical vision. *Ann. Biomed. Eng.* 40, 332–345. doi:10.1007/si0439-011-0441-z

Stoyanov, D., Darzi, A., and Yang, G. (2004). "Dense 3d depth recovery for soft tissue deformation during robotically assisted laparoscopic surgery," in *Medical Image Computing and Computer-Assisted Intervention-MICCAI 2004, Vol. 3217 of Lecture Notes in Computer Science*, eds C.Barillot, D.Haynor, and P.Hellier (Berlin, Heidelberg: Springer), 41–48.

Stoyanov, D., Mylonas, G., Lerotic, M., Chung, A., and Yang, G.-Z. (2008). Intraoperative visualizations: Perceptual fidelity and human factors. *J. Display Technol.* 4, 491–501. doi:10.1109/JDT.2008.926497

Stoyanov, D., Scarzanella, M., Pratt, R., and Yang, G.-Z. (2010). "Real-time stereo reconstruction in robotically assisted minimally invasive surgery," in *Medical Image Computing and Computer-Assisted Intervention MICCAI 2010, Vol. 6361 of Lecture Notes in Computer Science*, eds T.Jiang, N.Navab, J.Pluim, and M.Viergever (Berlin, Heidelberg: Springer), 275–282.

Sudha, N., and Mohan, A. (2011). Hardware-efficient image-based robotic path planning in a dynamic environment and its fpga implementation. *IEEE Trans. Ind. Electron.* 58, 1907–1920. doi:10.1109/TIE.2010.2054051

Vachhani, L., Sridharan, K., and Meher, P. (2009). Efficient fpga realization of cordic with application to robotic exploration. *IEEE Trans. Ind. Electron.* 56, 4915–4929. doi:10.1109/TIE.2009.2026225

Vadakkepat, P., Lim, P., De Silva, L., Jing, L., and Ling, L. L. (2008). Multimodal approach to human-face detection and tracking. *IEEE Trans. Ind. Electron.* 55, 1385–1393. doi:10.1109/TIE.2007.903993

Wang, X., Sliker, L. J., Qi, H. J., and Rentschler, M. E. (2013). A quasi-static model of wheel-tissue interaction for surgical robotics. *Med. Eng. Phys.* 35, 1368–1376. doi:10.1016/j.medengphy.2013.03.008

Conflict of Interest Statement: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.