BMC Research Notes

RESEARCH NOTE

Open Access

A toolkit for haptic force feedback in a telerobotic ultrasound system

Reza Fotouhi[1*], Atieh Najafi Semnani[1], QianWei Zhang[1], Scott J. Adams[2] and Haron Obaid[2]

Abstract

Objective: To develop a collision engine (haptic force feedback simulator) compatible with a 5-degrees-of-freedom (DOF) haptic wand. This has broad applications such as telerobotic ultrasound systems. Integrating force feedback into systems is critical to optimize remote scanning. A collision engine compatible with a 5-DOF haptic wand was developed based on the Gilbert–Johnson–Keerthi algorithm. The collision engine calculated force during collision between the wand and a virtual object based on code developed using MATLAB. A proportional force was subsequently returned to a user via the haptic wand, thereby simulating the collision force for the user. Three experiments were conducted to assess the accuracy of the collision engine on curved and flat surfaces.

Results: The average errors in calculation of distances between the wand and virtual object were 2.1 cm, 3.4 cm, and 4.2 cm for the model of the human hand, cylinder, and cuboid, respectively. The collision engine accurately simulated forces on a flat surface, though was less accurate on curved surfaces. Future work will incorporate haptic force feedback into a telerobotic ultrasound system. The haptic force simulator presented here may also be used in the development of ultrasound simulators for training and education.

Keywords: Haptic simulator, Gilbert–Johnson–Keerthi algorithm, Ultrasound, Tele-sonography

Introduction

Patients in remote and rural communities often have limited access to ultrasound imaging due to the lack of specialized sonographers in these communities. These challenges could be tackled with introducing a telerobotic ultrasound system that would allow specialists to remotely operate and control an ultrasound probe and remotely perform ultrasound scanning [1–3].

Most telerobotic ultrasound systems currently lack feedback of applied force on the patient during the examination. Haptic force feedback simulators is to replicate, physical interaction between an ultrasound probe and patient body to give the operator the opportunity to adjust the forces applied during telerobotic scanning [4, 5].

To support integration of haptic force feedback during interaction with a remote object or a virtual object, a new software is required. Many haptic packages which are currently available are proprietary rather than open source. Open source packages, such as V-COLLIDE and SOLID, operate on Linux and are not compatible with haptic devices, such as the 5-DOF haptic wand developed by Quanser which is used in this study [6].

To address this limitation, the objective of this study was to develop a collision engine (haptic force feedback simulator) compatible with a 5-DOF haptic wand and a Windows operating system. When the 5-DOF haptic wand collides with a virtual object, a force is sensed by a user via the haptic wand, providing force feedback. In this study we assess the performance of this haptic simulator on flat and curved surfaces of three different virtual objects.

*Correspondence: reza.fotouhi@usask.ca
[1] Department of Mechanical Engineering, University of Saskatchewan, 57 Campus Drive, Saskatoon, SK S7N 5A9, Canada
Full list of author information is available at the end of the article

Literature review

Telerobotic ultrasound systems

A commercially available telerobotic ultrasound system is MELODY, developed by AdEchoTech [7–9]. The MELODY system currently does not allow the sonographer to control the amount of force applied by the ultrasound probe (head) and could benefit from haptic technology. *Haptics* is defined as "touch interactions that occur for the purpose of perception and manipulation of objects" [10–12]. Haptic rendering is a functionality that adds certain haptic properties to an object to give it a realistic feel [11].

Haptic rendering-collision detection

Collision detection algorithms detect collisions between objects and an avatar in a virtual environment [4]. In order to detect a collision, the position of the end-effector (avatar) in the virtual environment must be determined. If the avatar is in free space and not colliding or touching a virtual object, then the calculated contact forces on the interface will be zero. However, if the avatar is touching a virtual object, forces are felt by the user manipulating the end-effector.

Haptic rendering

Collision response, or force-response, algorithms compute interaction force between avatars and virtual objects. It calculates an appropriate amount of force to be passed onto the haptic interface device. This force, F, is calculated using the following equation:

$$F = k\,x + C\,v \tag{1}$$

Here k is stiffness of object, C is viscous damping, x is penetration depth between the object and avatar (e.g. haptic wand), and v is the linear velocity of the avatar.

Collision detection engine

To develop a virtual reality haptic simulator, a software package (collision detection engine) and a haptic interface are required. The purpose of a collision detection engine is to detect collisions between two objects, calculate the distance between colliding objects, calculate the collision force, and generate an equal force via the haptic device. Commonly used algorithms include the Gilbert–Johnson–Keerthi (GJK) algorithm, bounding volume algorithm, and virtual spring algorithm, among others [10, 13–16].

In this toolkit, the GJK algorithm is utilized. This algorithm is a method to detect collisions between two convex objects. The algorithm is based on calculation of the Minkowski difference between two convex objects and determining whether or not the origin included is different. If the origin is included, the two objects collide with each other and have some points in common [17].

Haptic interface

Several algorithms have been used for training or simulation using haptic interface [13, 18–21].

Evaluation of haptic interfaces

Haptic interfaces are devices that enable manual interaction with virtual environments. They are employed for tasks that are usually performed using hands in the real world [10, 19, 22]. The haptic device used for this research is a 5-DOF haptic wand. The interface has five degrees of freedom, consisting of three degrees of translation and two degrees of rotation (roll and yaw).

Main text

Materials and methods

The collision engine which is developed in this study is based on collision detection using the GJK algorithm [23]. To calculate the approximate amount of force during the collision of the lower tip of a haptic wand and a virtual object, we developed MATLAB code which calculates the force based on Eq. (1) [12]. An overview of the methodology is shown in Figure 4 and described in further detail as follows:

1. 3D objects

Three 3D objects were imported into SolidWorks and saved in STL (stereolithography) format. To compare results between flat and curved surfaces of objects imported into the collision engine, STL files of the following objects were imported into MATLAB and Simulink (see Figure 5):

a. A human hand reconstructed from a CT scan [24];
b. A cylinder with a 0.025 m radius and a 0.1 m height; and
c. A cuboid measuring $0.1 \times 0.1 \times 0.2$ m^3.

2. Conversion from STL to X3D format

Using Blender (v. 2.49), we converted the STL files to X3D format, which is the required format for Quanser 3D Viewer. The converted human hand model in X3D format is shown in Figure 6. Considering that visualization and force calculation are two parallel and separate processes (Figure 4), a transformation matrix between MATLAB and Blender coordinate systems was used to calculate applied forces from object positions.

3. Import into 3D Viewer

The 3D Viewer was used for visualization of an avatar of the wand's tip and three objects imported into the virtual environment. Objects represented in X3D format were imported into 3D Viewer. The wand and an object in X3D format in virtual environment are shown in Figure 7.

4. Convert STL files to object vertices

Using a variation of MATLAB code [25], we converted STL files to object vertices (see Fig. 1a, Fig. 2a, and Fig. 3a). Vertices of an object were used as inputs for the GJK algorithm. These vertices form an $n \times 3$ matrix (where n is the number of vertices, each labeled by 3 coordinates), which define the object in the X, Y, and Z planes. The cuboid, cylinder, and human hand were represented by 36, 456, and 3660 vertices, respectively.

5. GJK algorithm in Simulink and MATLAB

Vertices of the imported objects produced in the previous step, as well as stiffness and damping ratios of each object and real time position vectors of the haptic interface, were used as inputs for the main MATLAB code [25]. For all objects, stiffness in the X, Y and Z directions was manually set to (160, 160, 160) N/m, and damping factors were set to (0.8, 0.8, 0.8) N·s/m, respectively.

The position of the haptic interface was defined by a 1×3 vector. "1" represents a single point (haptic interface), and "3" represents the X, Y, and Z coordinates of this point. The position vector is output of a specific QUARC block called "Haptic 5 DOF Wand Cartesian Plane" (Figure 5).

The center of mass of each object was set to (0, 0, −0.075). We simplified the wand to be comprised of a single point with one vertex and three degrees of freedom. Initial position of the wand was set at (0, 0, 0).

6. Simulating and generating haptic force feedback

A "CollisionFlag" was deployed so that when a collision was detected, the flag went to "1"; otherwise, it stayed at "0". If "CollisionFlag" was "1", the code calculated the force based on Eq. (1). The 5-DOF haptic wand generated force feedback when the wand collided with an object in the virtual environment.

Three virtual experiments were conducted in a 3D-setting to assess the accuracy of the collision engine: dorsal aspect of the human hand, curved surface of a cylinder, and flat surface of a cuboid. In these experiments, over a 10-s period, a user manipulated the haptic wand so that it would collide with the surface of the virtual object. Distance between the wand and the virtual object was recorded and contact forces between the virtual objects and the wand were measured by the collision engine based on Eq. (1) in X, Y and Z directions.

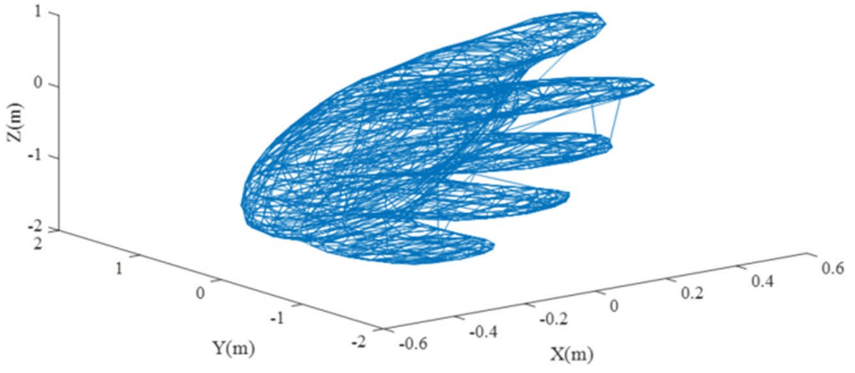
Results and discussion

Experiment with a model of a human hand

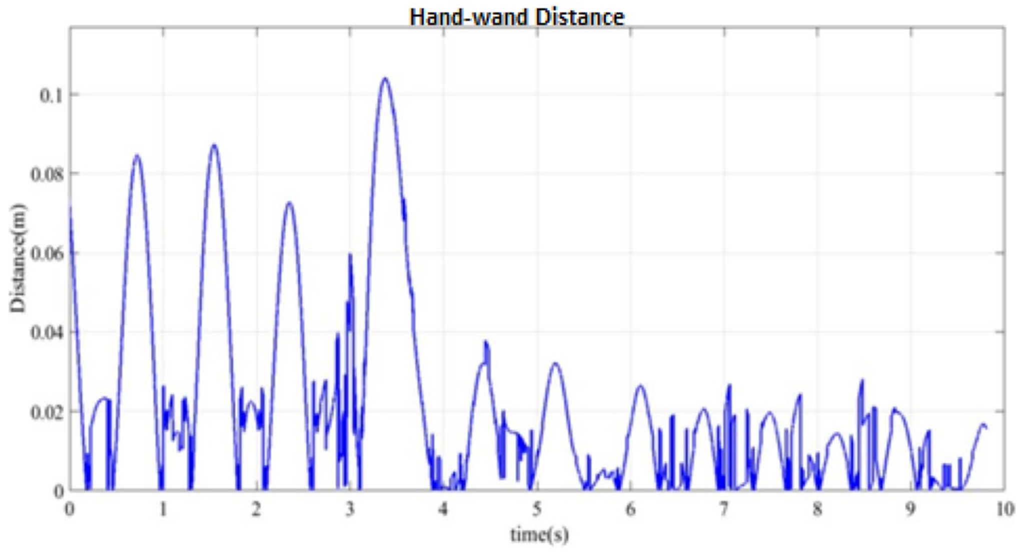
Figure 1b shows the distance between the wand and the dorsal surface of the hand in the first virtual experiment in which the wand was collided with a model of a hand over a 10-s period. At collision, the calculated distance between the wand and hand is zero. For this experiment, there is no record of false value for calculated distance from the engine. The simulated contact forces between the hand and wand are presented in Fig. 1c. The average of these forces during contact were 0.998 N, 0.996 N and 0.345 N in X, Y and Z directions, respectively. The average of the forces for all 11 contact periods is given in Table 1.

Experiment with a cylinder

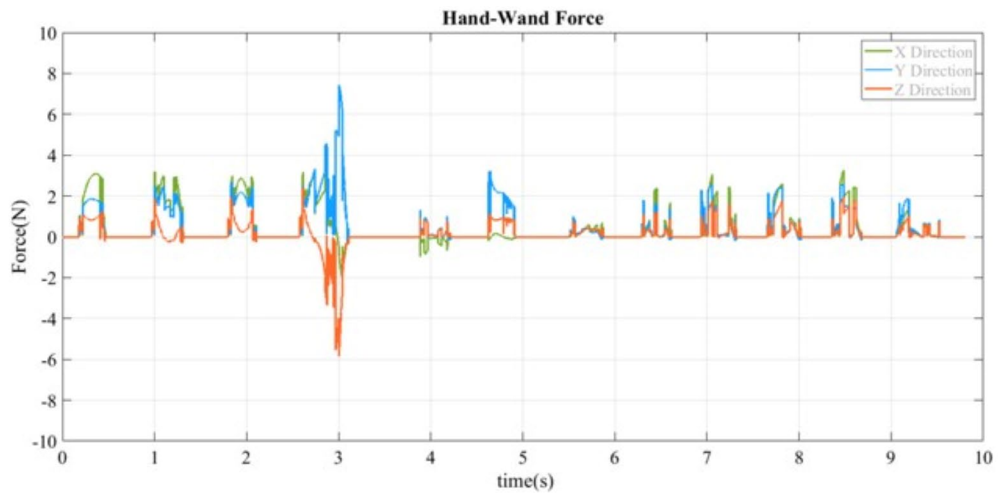
Figure 2b shows distance between the wand and curved surface of a cylinder in the second experiment where the wand was collided with the cylinder over a 10-s period. In some instances the engine miscalculated distances, which caused a noisy collision response. These miscalculated distances occurred at 6 instances: 3.8, 4.0, 4.2, 6.5 6.9, and 9.0 s, with an average error of 3.44E-2 m. The simulated contact force between the cylinder and wand are presented in Fig. 2c. The average of these forces during contact were 0.3561 N, 1.711 N and 0.6248 N in X, Y and Z directions, respectively, as shown.



Part a.



Part b.



Part c.

Fig. 1 **a** Plot of human hand model vertices; **b** distance sequence for human hand model and wand collision; **c** force sequence for human hand model and wand collision

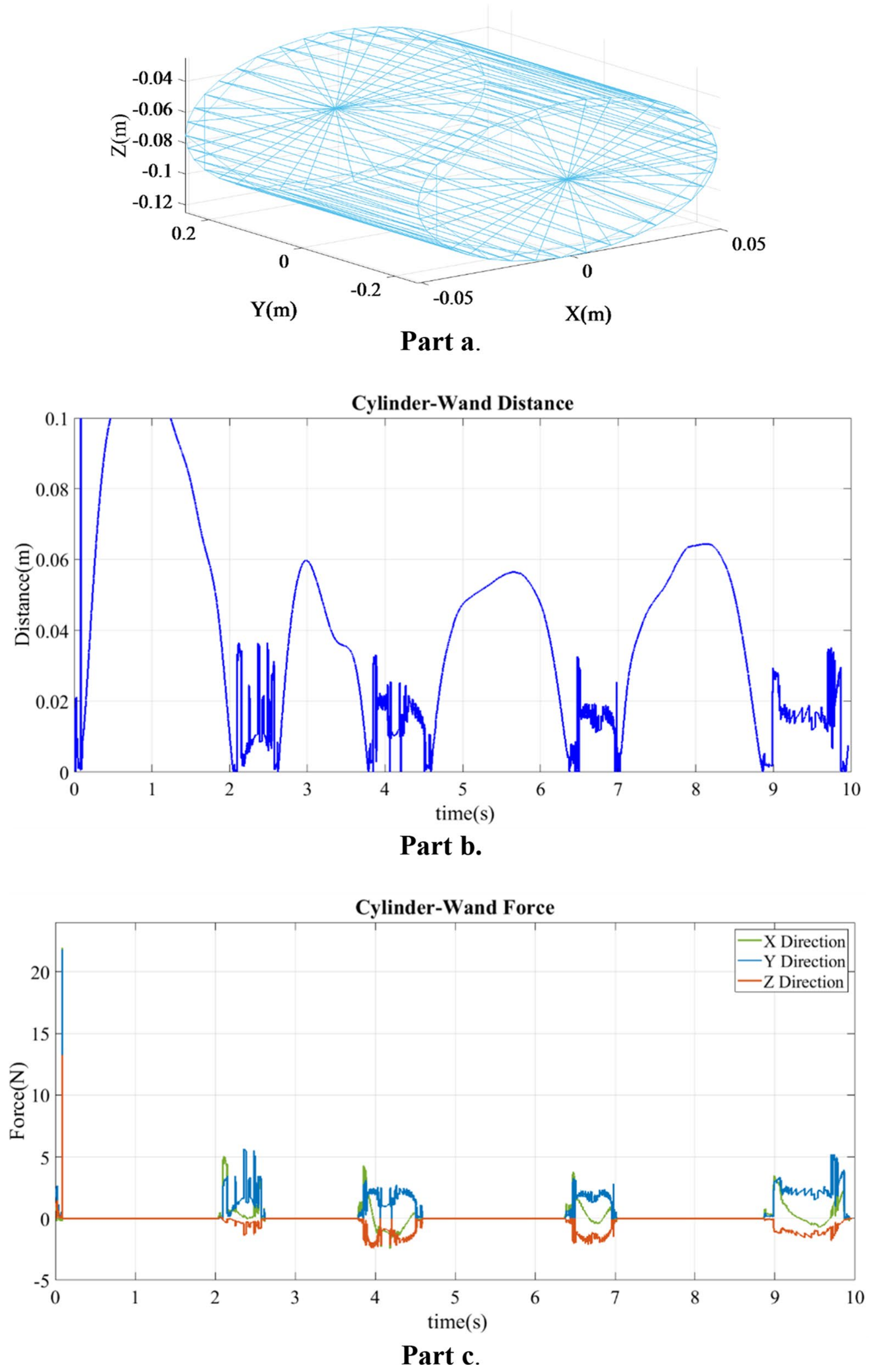
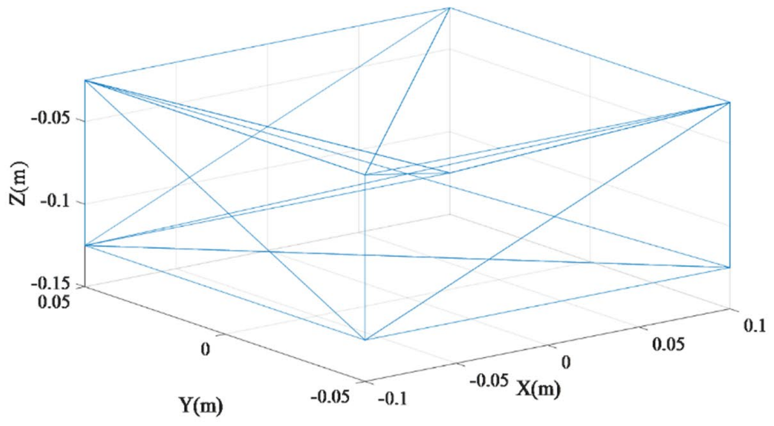
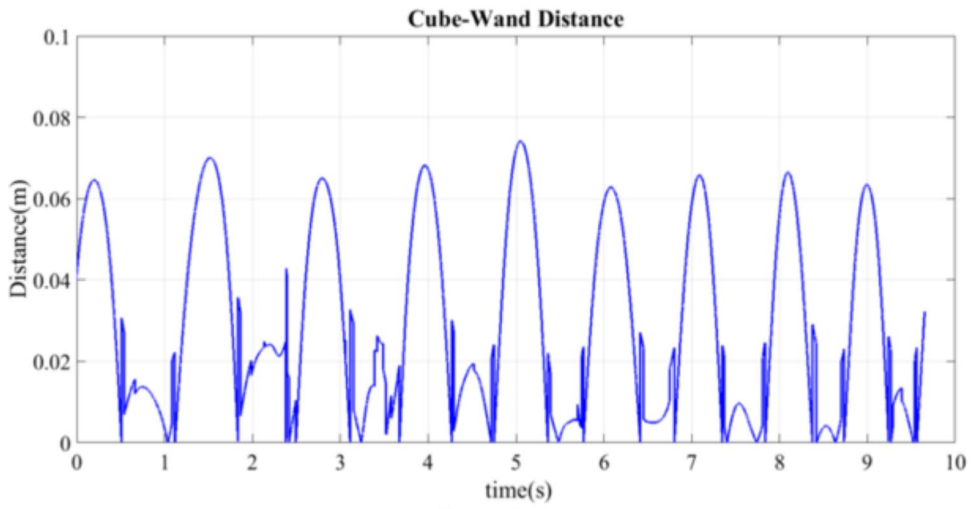


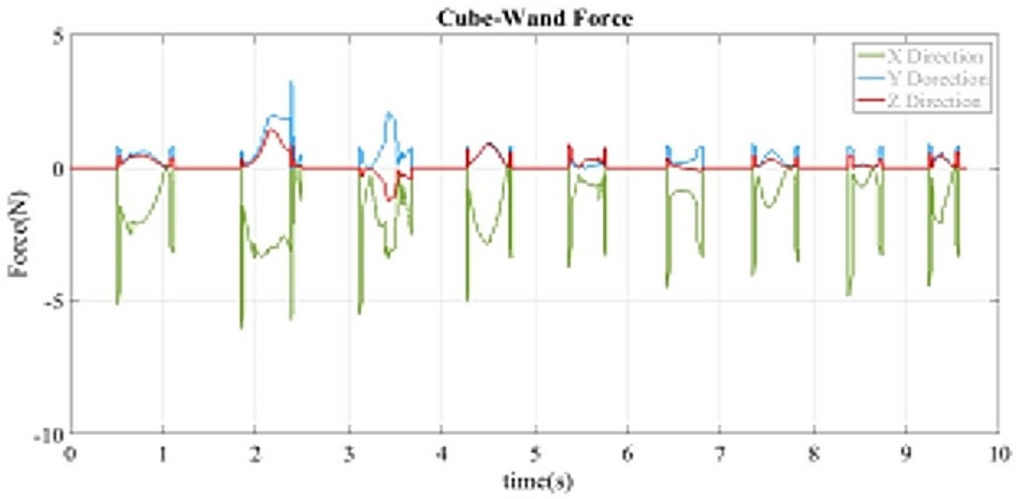
Fig. 2 **a** Plot of cylinder model vertices; **b** distance sequence for cylinder model and wand collision; **c** force sequence for cylinder model and wand collision



Part a.



Part b.



Part c.

Fig. 3 a Plot of cube model vertices; **b** distance sequence for cube model and wand collision; **c** force sequence for cube model and wand collision

Experiment with a cuboid

Figure 3b shows distance between the wand and surface of a cuboid in the third experiment where the wand was collided with the surface of a cuboid. In this experiment, the engine returned a false value for the calculated distance at instance of 2.4 s, which led to a noisy collision response, with an average distance error of 4.176E-2 m. The simulated force during contact between the cuboid and the wand in *X*, *Y* and *Z* directions is shown in Fig. 3c. The average contact forces in *X*, *Y* and *Z* directions were 1.62 N, 0.477 N, and 0.217 N, respectively, as shown. The average of the forces for all 9 contact periods is given in Table 4. By comparing the results of the two experiments, as shown in Table 2, it can be observed that the collision engine can detect the collision between the wand and a flat surface more efficiently than with a curved surface. Reported results are experiments between the haptic wand, manually controlled by an operator, and different virtual objects; the speed of wand motion, and shape of each object affects contact intervals.

To verify the generated force based on Eq. (1) from Simulink, we manually calculated these forces for two random instances at 8.098 s and 8.436 s in the cuboid experiment. The distance between the cuboid and wand at 8.098 s was 6.645E−2 m, with an expected force value of zero at that time. Figure 3c shows that the related force at 8.098 s is equal to zero in *X*, *Y*, and *Z* directions. Figure 3b shows that the distance at instant 8.436 s is −9.685E−4 m, which means the wand collided with the cuboid. Table 3 shows the wand velocity, distance vector from the wand toward the center of mass of the cuboid, and force value for *X*, *Y*, and *Z* directions at instant 8.436 s. The calculated force based on Eq. (1) at this instant ($F_x = 0.3194\ N$) is the same as the mean value of the contact forces as shown in Fig. 3c ($F_x = 0.3\ N$). The mean values of contact forces are reported in Table 4.

It should be noted that contact forces are not constant for each collision; as seen from Eq. (1); forces are dependent on penetration depth (x) and wand velocity (v); thus, contact forces may change for each contact; since the damping values used for each virtual objects are small, damping of contact forces are not noticeable.

Conclusion

In this paper, a collision engine compatible with a 5-DOF haptic wand to simulate haptic force feedback is presented. Through virtual experiments with cuboid, cylinder, and human hand models, we have shown that our approach can work on several surfaces. The collision engine was able to accurately simulate haptic force feedback on a flat surface, though was less accurate on curved surfaces. One of the limitations of the GJK algorithm [16] which was utilized in this research is that it can only detect collisions between convex objects. The haptic force feedback simulator presented here will have applicability in development of ultrasound simulators for training and education, and in incorporating force feedback for telerobotic ultrasound system.

Limitations

Haptic packages which are utilized in the majority of research works include Phantom Omni and Open Haptics [18]; however, neither are open source. V-COLLIDE and SOLID are two open source software packages commonly implemented as collision engines [12], both on Linux.

V-COLLIDE is an n-body processor used for multiple object collision detection. Once a user assigns the position of objects to V-COLLIDE, the software will return contact status of the objects including collision or any possible contact. Although V-COLLIDE is a powerful tool for dynamic collision detection and recalling the position of objects, this software is unable to calculate distance between objects [26]. SOLID is another collision detection library utilized for multiple three-dimensional polygonal objects going through rigid motion. SOLID is comparable with V-COLLIDE in terms of performance and application, and it determines interface between objects [27]. In comparison with V-COLLIDE and SOLID, the collision engine presented in this paper can be used on a Windows operating system and facilitates measurement of distance between colliding objects.

The proposed algorithm utilized in the collision engine is limited since it can only detect collisions between convex objects. In the future, the convex decomposition method [18] can be incorporated into the engine to remedy this. The program currently does not record the trajectory of the wand, although this feature may be added in the future. To contact from different direction, as shown in above examples, we can rotate them (e.g. cuboid) in virtual setting; because of physical construction of the haptic wand, it has limited range of motion, which is about ±30° about Z-axis.

Appendix

See Figs. 4, 5, 6, 7; Tables 1, 2, 3, 4.

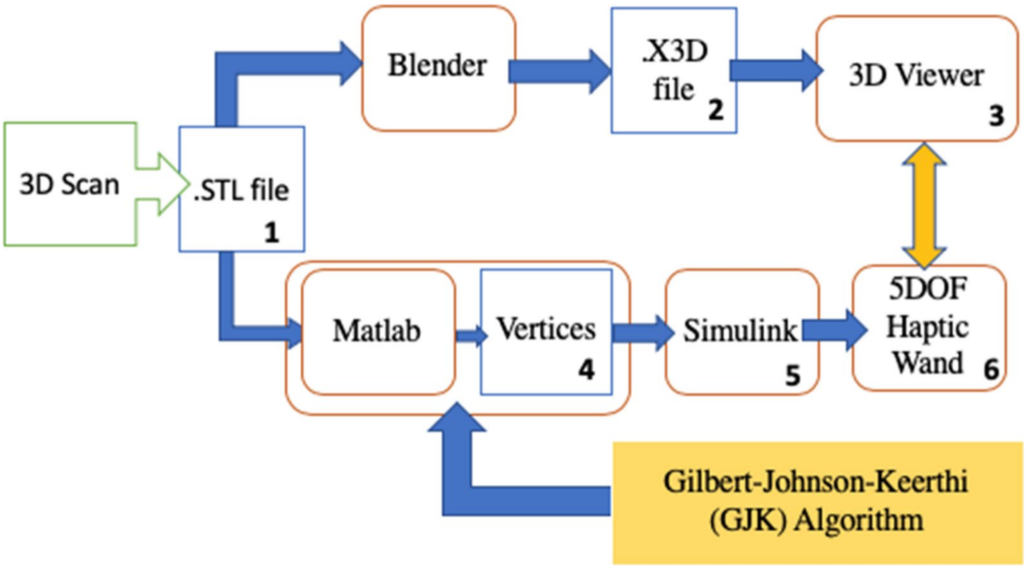


Fig. 4 Flowchart of the collision engine. Numbers in boxes refer to steps in Methodology

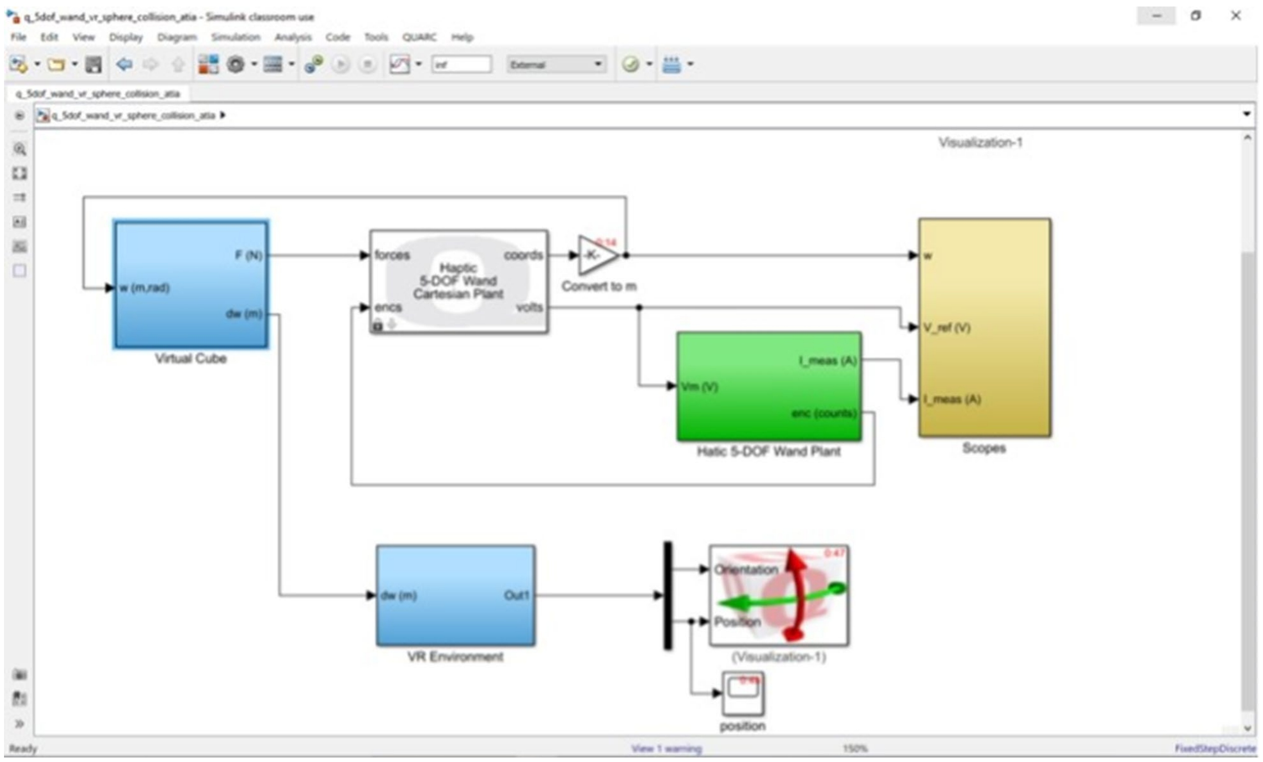


Fig. 5 Simulink model for the virtual reality haptic simulator, including QUARC blocks

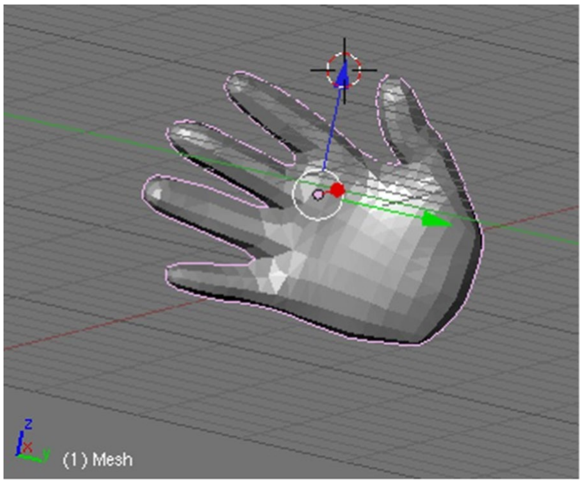


Fig. 6 Converting the object from STL to X3D format

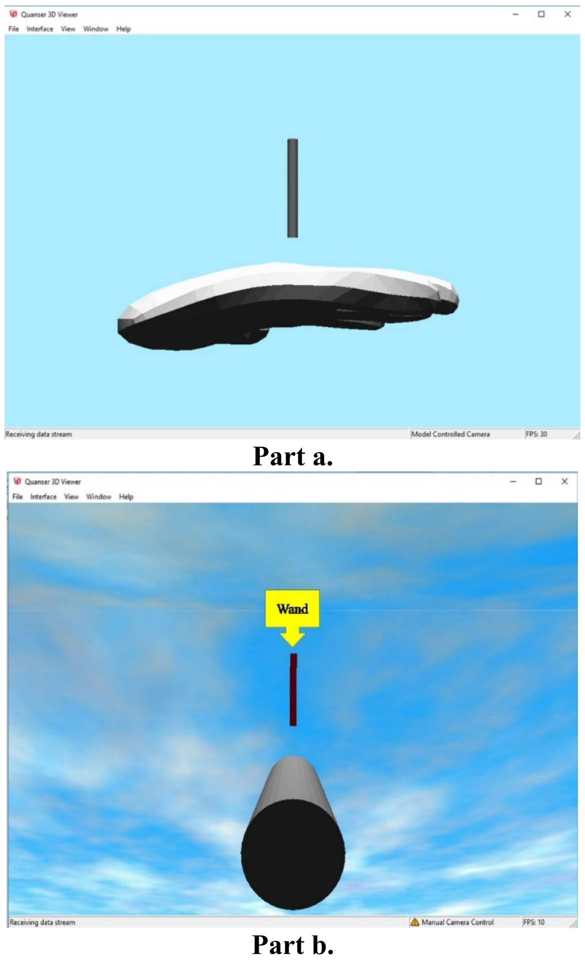


Fig. 7 X3D format of the imported models in the Virtual Reality Environment (3D Viewer), with the wand above each object; **a** human hand model, **b** cylinder model

Table 1 Contact intervals with mean values of measured forces in *X*, *Y*, and *Z* directions, Human Hand Model

Time period (s)	F_x (N)	F_y (N)	F_z (N)
0.2–0.45	2.434	1.542	0.8417
1–1.3	1.799	1.407	0.1984
1.8–2.1	2.204	1.647	0.5779
2.65–3.05	1.191	2.791	− 1.133
3.88–4.25	− 0.2231	0.2434	0.2146
4.6–4.95	0.003	0.1909	0.7829
5.5–5.9	0.3367	0.2689	0.2665
6.3–6.65	0.4804	0.3912	0.2679
6.9–7.3	0.7628	0.6750	0.4729
7.65–8.1	0.8455	0.8792	0.5574
8.35–8.7	1.154	0.9189	0.7470

Table 2 Comparison of experiments performed for the flat surface of the cuboid and curved surface of the cylinder

	Number of vertices	Average of distance error (m)	Number of errors	Total collisions
Cuboid	36	4.176E-2	1	18
Cylinder	456	3.44E-2	6	9

Table 3 Force calculation at instant 8.436 s, Cuboid

	Wand velocity (m/s)	Distance vector (m)	Force (N)
X	0.2055	− 9.685E−4	0.3194
Y	0.0087	1.967E−4	0.0245
Z	0.0069	1.268E−4	0.0148

Table 4 Contact intervals with mean values of measured forces in X, Y, and Z directions, cuboid model

Time period (s)	F_x (N)	F_y (N)	F_z (N)
0.50–1.10	− 2.1	0.3	0.3
1.80–2.40	− 2.9	1.1	0.9
3.10–3.70	− 1.8	0.9	− 0.3
4.25–4.75	− 1.9	0.5	0.5
5.40–5.75	− 0.5	0.1	0.2
6.45–6.80	− 1.1	0.2	0.1
7.40–7.80	− 0.8	0.3	0.2
8.40–8.75	− 0.3	0.1	0.05
9.25–9.60	− 1.4	0.3	0.43

Abbreviations
3D: 3-Dimensional; DOF: Degrees-of-freedom; GJK: Gilbert–Johnson–Keerthi; QUARC: Real-time control software from Quanser; SOLID: An acronym for the first five object-oriented design principles; STL: Standard Template Library; V-COLLIDE: A collision detection library for large environments (of polygonal objects); X3D: A royalty-free ISO/IEC standard for declaratively representing 3D computer graphics.

Acknowledgements
This Project is collaborative research between Robotics lab in Mechanical Engineering Department, and Department of Medical Imaging at the University of Saskatchewan. People involved in this research are co-authors here. We gratefully acknowledge assistance from Mr. Doug Bitner, Department of Mechanical Engineering, University of Saskatchewan, in setting up and troubleshooting the haptic wand.

Authors' contributions
RF made contributions to the conception of the idea/work, analysis and interpretation of data, extensive editing and redrafting of the paper, and co-supervision of the research work of ANS. ANS modified a Simulink model from Quanser to enable a 5-DOF haptic wand to generate haptic force feedback responses for imported objects. She contributed to developing the work, analysis and interpretation of data, and writing the first draft of the paper. QZ contributed to data analysis, working with the equipment, and helped to revise the paper. SJA contributed to interpretation of data and helped to revise the paper. HO co-supervised the research work of ANS, made contributions to the conception of idea/work, and helped to revise the paper. All authors read and approved the final manuscript.

Funding
This research was supported through funding from the Natural Sciences and Engineering Research Council of Canada, and from the College of Medicine, University of Saskatchewan.

Availability of data and materials
Data are codes available publicly at HF2MS in https://sourceforge.net/projects/hf2ms/files/; and in https://github.com/atiandjf/HF2TS).

Declarations

Ethics approval and consent to participate
Not applicable.

Consent for publication
Not applicable.

Competing interests
The authors declare that they have no competing interests.

Author details
[1]Department of Mechanical Engineering, University of Saskatchewan, 57 Campus Drive, Saskatoon, SK S7N 5A9, Canada. [2]Department of Medical Imaging, University of Saskatchewan, Saskatoon, SK, Canada.

Received: 17 April 2021 Accepted: 11 October 2021
Published online: 24 October 2021

References
1. Fotouhi R, Oraji R, Mondragon C, Berryman B. Development of a remote ultrasound imaging system. In: ASME international design engineering technical conferences and computers and information in engineering conference; Boston, MA; 2015. V003T14A008.
2. Boman K, Olofsson M, Forsberg J, Boström S-A. Remote-controlled robotic arm for real-time echocardiography: the diagnostic future for patients in rural areas? Telemed J E Health. 2009;15(2):142–7. https://doi.org/10.1089/tmj.2008.0079.
3. Avgousti S, Christoforou EG, Panayides AS, et al. Medical telerobotic systems: current status and future trends. Biomed Eng Online. 2016;15(1):96. https://doi.org/10.1186/s12938-016-0217-7.
4. Salisbury K, Conti F, Barbagli F. Haptic rendering: introductory concepts. IEEE Comput Graphics Appl. 2004;24(2):24–32.
5. Blum T, Rieger A, Navab N, Friess H, Martignoni M. A review of computer-based simulators for ultrasound training. Simul Healthc. 2013;8(2):98–108.
6. Quanser. Haptics; 2020. http://www.quanser-update.azurewebsites.net/quarc/documentation/quarc_using_devices_haptics.html. Accessed 14 July 2020.
7. AdEchoTech. Products; 2016. www.adechotech.com/products/. Accessed 4 Dec 2020.
8. Adams SJ, Burbridge BE, Badea A, Kanigan N, Bustamante L, Babyn P, et al. A crossover comparison of standard and telerobotic approaches to prenatal sonography. J Ultrasound Med. 2018;37(11):2603–12.
9. Adams SJ, Burbridge BE, Badea A, et al. Initial experience using a telerobotic ultrasound system for adult abdominal sonography. Can Assoc Radiol J. 2017;68(3):308–14. https://doi.org/10.1016/j.carj.2016.08.002.
10. Kraut J, Hochman JB, Unger B. Temporal bone surgical simulation employing a multicore architecture. In: 26th IEEE Canadian conference on electrical and computer enginerring; Regina, SK; 2013. p. 1–6.
11. Singapogu R, Sander ST, Burg TC, Cobb WS. Comparative study of haptic and visual feedback for kinesthetic training tasks. Stud Health Technol Inform. 2008;132:469–71.
12. Laycock SD, Day AM. A survey of haptic rendering techniques. Comput Graph Forum. 2007;26(1):50–65.
13. Ericson C. Chapter 9—convexity-based methods. In: Ericson C, editor. Real-time collision detection. San Francisco: Morgan Kaufmann; 2005. p. 383–412.
14. Liu L, Wang Z, Xia S. A volumetric bounding volume hierarchy for collision detection. In: 2007 10th IEEE international conference on computer-aided design and computer graphics; Beijing, China; 2007. p. 485–8.
15. Bryan J, Stredney D, Wiet G, Sessanna D. Virtual temporal bone dissection: a case study. In: Proceedings of the Conference on Visualization '01. San Diego, CA; 2001. p. 497–500.
16. Barbieri E. Improving the GJK algorithm for faster and more reliable distance queries between convex objects. ACM Trans Graph. 2017;36(4):151a.
17. Ericson C. Chapter 3—A math and geometry primer. In: Ericson C, editor. Real-time collision detection. San Francisco: Morgan Kaufmann; 2005. p. 23–73.
18. Noborio H, Sasaki D, Kawamoto Y, Tatsumi T, Sohmura T. Mixed reality software for dental simulation system. In: 2008 IEEE international workshop on haptic audio visual environments and games; Ottawa, ON; 2008. p. 19–24.
19. 3D Systems. Scanners and haptics; 2020. https://www.3dsystems.com/scanners-haptics. Accessed 14 July 2020.
20. Kim L, Sukhatme GS, Desbrun M. A haptic-rendering technique based on hybrid surface representation. IEEE Comput Graph Appl. 2004;24(2):66–75.
21. Virtual Reality Dental Training System. http://www.novint.com/VRDTS.htm. Accessed 14 July 2020.
22. Unger B, Sepehri N, Rampersad V, Pisa J, Gousseau M, Hochman JB. Elements of virtual temporal bone surgery: manipulandum format may be more important to surgeons than haptic device force capabilities. Laryngosc Investig Otolaryngol. 2017;2(6):358–62.
23. Talasaz A, Trejos AL, Perreault S, Bassan H, Patel RV. A dual-arm 7-degrees-of-freedom haptics-enabled teleoperation test bed for minimally invasive surgery. J Med Devices. 2014;8(4):041004.
24. Hand V1 3D Model. http://www.free3d.com/3d-model/hand-v1--675788.html, Accessed 10 July 2020.
25. Lebel P. GJK algorithm distance of closest points in 3D. MathWorks file exchange; 2018. https://www.mathworks.com/matlabcentral/fileexchange/62429-gjkalgorithm-distance-of-closest-points-in-3d. Accessed 10 July 2020.
26. Hudson TC, Lin MC, Cohen J, Gottschalk S, Manocha D. V-COLLIDE: accelerated collision detection for VRML. In: Proceedings of the second symposium on virtual reality modeling language, Feb 26; 1997. p. 117–23.
27. Lin M, Gottschalk S. Collision detection between geometric models: a survey. In: Proceedings of IMA conference on mathematics of surfaces May 1998. vol. 1. p. 602–8.