Building an Exoskeleton Glove on Virtual Reality Platform

Quazi Irfan*, Calvin Jensen*, Zhen Ni§, Steven Hietpas§
Electrical Engineering and Computer Science Department
South Dakota State University
Brookings, SD, USA 57007
Email: {quazi.irfan, calvin.kielasjensen, zhen.ni, steven.hietpas}@sdstate.edu
*undergraduate students; §supervisor

Abstract - Recent advancements in virtual reality (VR) technologies allow users to experience virtual environment in lifelike fidelity. But users still have a very limited level of interactivity with virtual objects in these virtual environments. Current generation input/output devices are not sophisticated enough to emulate physical interactivity between the user and the virtual environment. In this paper, the construction method of building a new type of input output (I/O) device, a VR glove, that allows users to physically interact with the constituent virtual objects in the virtual environment, is presented. The VR glove tracks the user's physical finger movement and translates the movement to virtual fingers in a game environment, and if the virtual finger touches a virtual object, the motors attached at finger joints of the VR gloves spin up or down to generate haptic feedback to emulate the physical interactivity with the virtual object. The prototype glove covers a single finger. To test the glove, two different test environments, where the virtual finger interacts with soft and hard virtual object, were built.

Keywords – virtual reality haptic, force feedback, motion capture

I. INTRODUCTION

Virtual environments are digitally mediated artificial spaces. Users interact with these virtual environments through a flat display (i.e. a computer monitor, a television or a projector surface [1], [2]. But in case of 3D virtual environments, these flat displays have limited capacity to retain depth information for binocular vision. Eyes fail to infer depth information of the scene viewed on the flat panel. In recent years, virtual reality systems have solved this problem by introducing head mounted displays (HMD). These HMDs place separate screens in front of the eyes. These two displays are used to show the user the virtual environment from two slightly different viewpoints and this system allows the binocular vision to reconstruct the depth information of the scene [1].

Realistic methods to interact with the virtual objects are not yet widely available. Most VR systems come with a game controller that tracks a user's hand movement. To interact with an object, the user points the game controller towards the object and presses a button [3]. Many glove-based systems have appeared in the literature or market place over the past 30 years that track the finger movement and allow the player to see their individual finger movement in the virtual world, but only very few of these gloves can generate haptics when the virtual finger interacts with a virtual object [4]. One of the first examples of VR gloves able to generate haptics was developed at University of Tsukuba. This glove produced force feedback by pulling back the tip of the finger with strings attached with motors - placed on the dorsal side of the hand [5]. CyberGrasp is the first commercially available force feedback glove, but it weighs approximately 450g and its control unit weighs 20kg, making is difficult to transport [6]. Since then, different technologies have been emerged to miniaturize the haptic technology. Gloves employing magneto-rheological (MR) brakes, to generate force feedback, are small enough to place on the back of the finger [7]. Other VR gloves, such as Dexmo and Robotics and Mechatronics Lab (RML) gloves, have simplified the construction by using a link rod structures and cable that pulls the tip of the finger using actuators and motors, respectively, to generate force feedback [8], [9]. These approaches simplify the construction of the gloves at a cost of not being able to generate force feedback in all possible finger configurations.

In this study, a VR glove design is proposed that generates haptic feedback by employing independent motors at each finger joint. This VR glove tracks joint movement using incremental encoder attached with each motor. In contrast to other gloves, this hardware configuration allows generation of haptic feedback in all finger configurations since every joint has a dedicated motor. This encoder data is sent to the game environment to synchronize the virtual finger and the physical finger. If the movement in the virtual finger results in a collision with a virtual object, the game system sends a torque value to the motor hardware to apply required torque to restrict the real finger movement, thereby emulating the physical presence of the virtual object. The torque produced by the motor varies depending on the type of object the user is interacting with [10]. If the user is interacting with a non-elastic object, high torque is

978-1-5386-5398-2/18/$31.00 ©2018 IEEE

applied when collision occurs. But when interacting with an elastic object, the user would experience a gradient of force applied to the finger by the motor.

The remainder of the paper is organized as follows. Section II describes the proposed method and outlines the objective of the paper. Section III describes the physical setup of the gloves. Section IV describes the software and hardware firmware used in the system. Section V highlights the gloves interacting with hard and soft virtual object, and finally Section VI summarizes the paper and describes future work.

II. PROPOSED METHOD

An exoskeletal VR glove will be built that the user would wear like a regular glove. Fig. 1 is an artist's rendition of the VR gloves on one finger. At each finger joint on the gloves, motors will be attached to generate haptic feedback. Motors are

Fig. 1. One finger wearing the VR glove.

highlighted as a red cylinder in Fig. 1. The blue segments of the gloves are to accommodate the variable joint length in different finger postures. Finger movements will be read from the encoders attached to each motor. The encoder data would be used to transform the virtual finger in synchronicity with the real finger. The player would be wearing a HMD provided by the VR system to see the virtual fingers. The virtual game environment system will continuously check if there is a collision between the virtual fingers with any other virtual objects in the virtual environment. If there is a collision, the game will send appropriate torque value to the motor, and the motor will spin up or down to generate the required haptic feedback to emulate the physical presence of the virtual object.

The proposed method here is for a single glove finger. Once the software and hardware for one finger works, the same setup can be repeated to track all fingers. The study is split into two phases. In the first phase, the hardware and software for a single joint has been completed, and in the second phase, two additional joints to cover a full finger will be completed. Using a 3D printed joint that sits on top of a finger joint, any movement of the finger joint causes rotation on the motor shaft, which in turn generates encoder ticks. The virtual environment uses the encoder ticks to keep the virtual joint and the 3D printed joint in sync. In the virtual environment, if a virtual finger collides with a virtual object, the game engine will send a command for the motor to generate the appropriate haptic feedback.

III. PHYSICAL SETUP

To generate haptic feedback, a brushed DC motor was chosen for its affordability. A magnetic quadrature encoder is attached at the extended motor shaft of each motor. The encoder consists of a disc with magnets embedded in it, and two Hall Effect latches. The final configuration of the motor attached with the encoder is shown in Fig. 2. For every full rotation on the extended motor shaft, 12 ticks are generated from the Hall Effect sensors. On the other end of the motor shaft a 5:1 gear box is attached. One full revolution of the gearbox output shaft causes 15 full revolutions on the extended shaft, which in turns

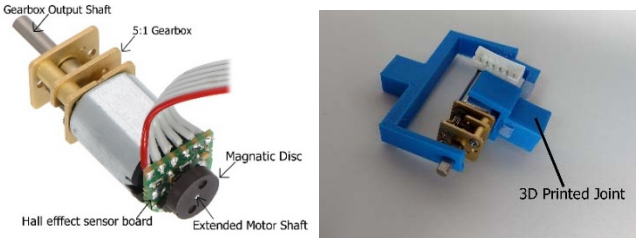

Fig. 2. DC motor with encoder enclosed in 3D printed joint.

produces 60 ticks on the encoder. These encoder counts are sent to the game system running on a laptop via USB.

The proximal interphalangeal joint (PIJ) of the index finger was chosen to test with the VR glove's joint. The active range of this joint is -7 to 101 degrees [11]. A joint was 3D printed to house the motor, which is shown in Fig. 2. During testing, the 3D-printed joint represents the PIJ in the VR gloves, which is shown in Fig. 3. In this setup, the angular movement occurred at PIJ is transferred to the motor's gearbox output shaft. For 0 to 90 degrees rotation occurring at PIJ, the hardware generates 15 encoder ticks. 15 ticks do not provide sufficient resolution to allow smooth synchronization between the virtual and real finger, but this is sufficient to demonstrate the system. To keep the design simple, it was decided not to build strap to attach it with the finger. During testing, any one of the blue extensions representing the middle phalanx or the proximal phalanx are rotated to emulate finger flexion/extension caused by the user at PIJ.

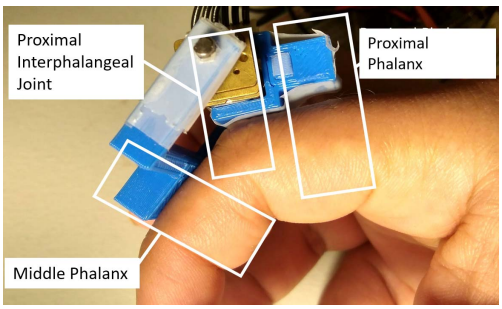

Fig. 3. 3D printed joint sitting on top of finger joint.

The A-Star 32U4 microcontroller was chosen to control the motor. It was selected for four main reasons. First, it is Arduino compatible, which means that the development would go quickly due to the large online community that supports Arduino. Second, the clock speed for the 32U4 is sufficiently high, at 16 MHz. Even if 1000 instructions were to run every loop, the control system would have still run at 16 kHz, which was larger than the desired frequency of 1 kHz. 1 kHz was chosen as a baseline frequency of the control system to test upon. Third, the 32U4 has 7 pulse width modulation (PWM) pins, required to run the motor controller. Finally, the microcontroller has 8 analog input pins, necessary for reading the motor driver's current sensor pin.

The motor is driven using the DRV8801 Single Brushed DC Motor Driver. This motor driver supports 100% PWM, which is required for operating the DC motor at full load. It supports a maximum PWM frequency of 50 kHz, which is higher than the maximum frequency a human can hear. This allows the motor to be run without producing a high-pitched sound. The driver current sense output allows the measurement of motor current. The motor driver is rated at 1 A continuous and 2.8 A peak current, which is larger than the 800 mA stall current of the DC motor. The motor driver also supports forward and backward directions, 3.3 V to 6.5 V logic, and a motor supply voltage of 8 V to 36 V.

Since the motor driver only outputs 500 mV per 1 A on the current sense pin, it was necessary to amplify the signal to utilize the full range of the 5 V analog-to-digital converter (ADC) on the 32U4. To do this, an LM324 operational amplifier was chosen. Used in a non-inverting configuration, the gain of the operational amplifier was calculated to be 12.5. At this gain, the bandwidth of the LM324 is 120 kHz, which far surpasses the frequency of the control system.

The final physical setup of the single joint of the VR glove is shown in Fig. 4.

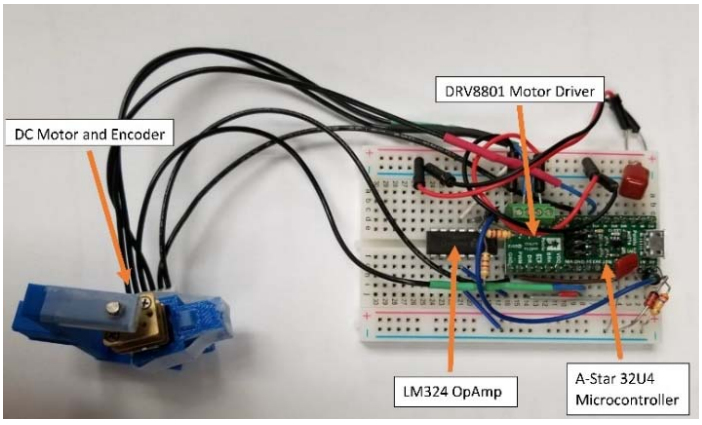

Fig. 4. Final physical setup.

IV. SOFTWARE SETUP

a. Hardware firmware setup: The flowchart for the firmware is shown in Fig. 5. The blocks in the white occur once on startup, whereas the blocks in gray occur in an infinite loop. The startup code initializes the digital and interrupt pins on the microcontroller. It also initializes values in the PID controller and removes the bias present in the current measurement.

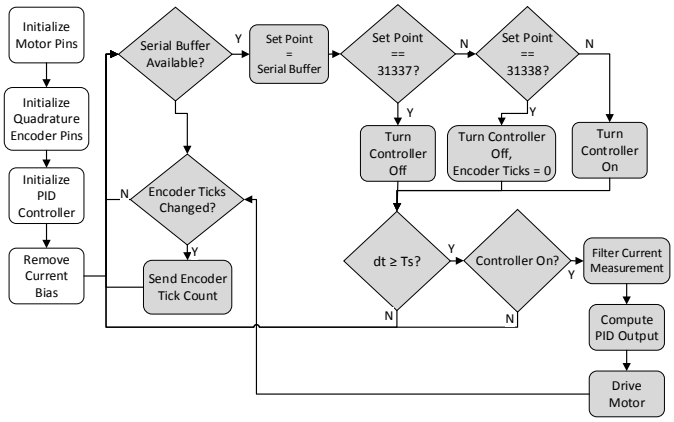

Fig. 5. Flowchart of hardware firmware

The continuously running code executes the control loop, drives the motor, and communicates with the game engine. An "if" statement is used to force the control loop to run at 100 Hz. A constant frequency is useful in control loops and helps avoid unexpected behavior. The control loop has an input torque value which is sent from the game engine. It also monitors the current sense pin on the motor driver. Since torque is proportional to current in a DC motor, the torque of the motor can be estimated and used to feed back to the controller.

A PID controller was designed using Simulink [12]. The control system, shown in Fig. 6, was designed by first modeling the motor and comparing the real output with the model output. As shown in the Fig. 7, the model was found to closely match the real system. The model was used to run simulations of different motor controllers. The PID controller was chosen for its performance and ease of implementation.

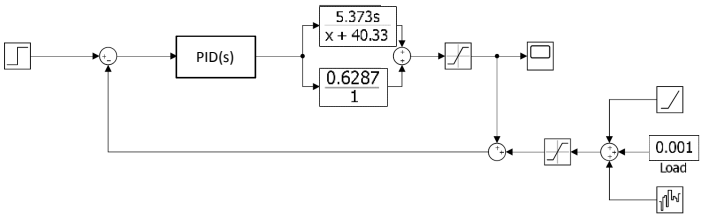

Fig. 6. Simulink control loop.

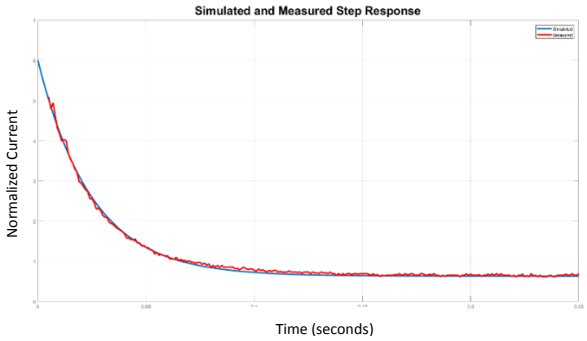

Fig. 7. Comparing the simulated and measured step responses.

Serial communication is used to relay information between the microcontroller and the game engine. To avoid slowing down the control system and sending a continuous stream of data, the game engine and microcontroller only send data when it a change occurs. A signed, two-byte value is sent from the microcontroller as encoder counts to the game engine to inform it of the motor position. Similarly, the game system sends a two-byte value for what the torque should be, which is fed to the PID controller. A special value of 31337 is sent from the game engine to set the controller to idle and stop powering the motor. A value of 31338 is sent to do the same thing while also resetting the tick count. This value is used primarily to zero the initial position of the motor. Acceptable torque range for this system is -1024 to 1023.

b. Virtual Environment Setup: The A-Star 32U4 microcontroller is connected to a laptop over a USB connection. The laptop runs a virtual environment, also called game environment, using jMonkey game engine [13]. The game is connected to the A-Star 32U4 microcontroller on a serial port using Java simple serial connector (JSSC) library . Fig. 8 shows the startup screen of the game environment. The game represents the PIJ with a red cylinder, and middle phalanx is represented by the top blue cubes, and the proximal phalanx is represented by the bottom blue cubes. In the game environment, the joint remains stationary. Encoder values only affect the rotation of the top blue box representing the middle phalanx.

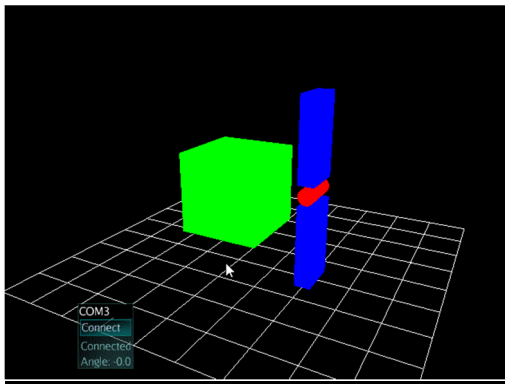

Fig. 8. Startup environment of the game.

The control flow of the game environment is shown in Fig. 9. At the beginning, the game connects with the hardware on a specified COM port number. The user is then asked to fully extend a finger, so the virtual joint and real joint can start communicating from a known posture. After calibration, the game engine uses the incoming encoder value to keep the virtual joint in sync with the 3D printed joint sitting on top of PIJ.

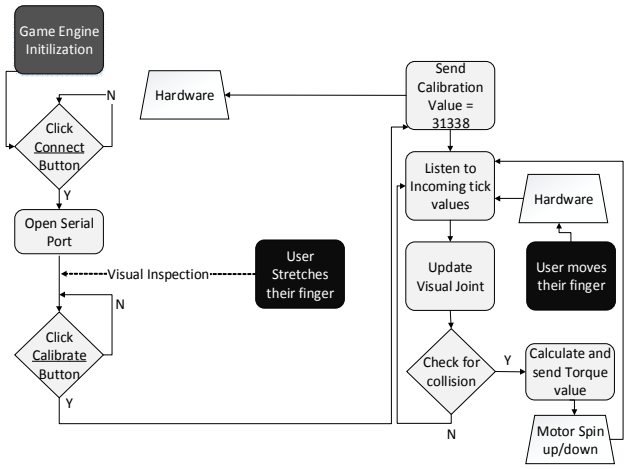

Fig. 9. Flowchart of game engine logic.

The green box in the game represents a virtual object for the virtual finger to interact with. The game environment constantly checks for a collision between the blue box and the green box, which represents the user trying to interact with a virtual object. If there is a collision, the game environment sends an appropriate torque value to the hardware to generate the desired haptic feedback. If the user moves a finger away from the box a torque value 0 is sent.

c. Hardware Design Considerations: The physical design of the system was sufficient to provide a proof of concept, however, the motor used in the joint proved to be both too large and not provide sufficient torque for realistic physical feedback. Fortunately, there are some potential design improvements that could be made. Moving the motor away from the joint would be one option. Instead of having a motor at every joint, bring the motors up to the forearm, bicep, or shoulder. The motors would then pull on wires producing forces similar to how muscles work. Another option would result in removal of motors in general and using a pneumatic system. This is known as soft robotics and is accomplished by controlling the volume of air inside small pockets strategically placed in molded rubber.

V. RESULTS AND ANALYSIS

a. Interaction with Hard Object: Two demonstrations were developed to outline the interaction with soft and hard objects. In the first demo, the green box represents a non-elastic obstacle, and in the second demo, a green sphere is used to represent an elastic object. In the first demo, when the virtual joint collides

with a solid object the game environment sends a high torque to emulate the physical presence of the virtual object. The graph in Fig. 10 shows the changes in the virtual joint angle and the different torques applied by the motor with respect to time, while interacting with a non-elastic object.

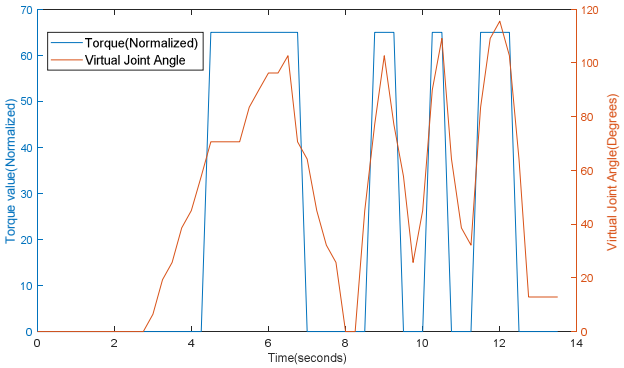

Fig. 10. Interaction with a solid object.

At 4.25 seconds, when there is a collision between the virtual joint with the green box, the game sends a torque value 65. The value 65 was found to be strong enough to emulate sudden resistance caused by the obstacle. At 6.9 seconds, when the joint moves away from the obstacle, the game sends a torque value of 0.

b. Interaction with Soft Object: In the second demo, the obstacle is an elastic object represented by a green sphere. Fig. 11 represents the virtual finger lightly touching an elastic ball (left), and virtual finger pressing hard on the elastic ball (right).

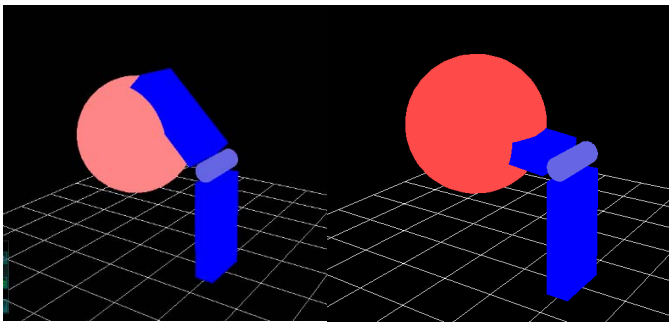

Fig. 11 In-game view of interaction with soft object

The amount of torque sent to the hardware is dependent on how deep the virtual joint goes into the ball. The game sends a small torque value at the point of contact, since in real life an exercise ball provides little resistance when held lightly. As the joint moves deeper into the ball, higher torque is applied, emulating difficulty in compressing an already compressed exercise ball. A lower torque value is sent when the finger is slowly pulled away, emulating the release of a compressed exercise ball. The following graph, Fig. 12, shows the changes in virtual joint angles and different torques applied by the motor with respect to time when interacting with an elastic object.

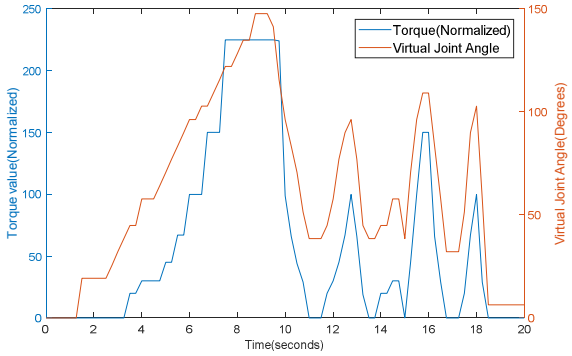

Fig. 12. Interaction with a soft object.

At 3.1 seconds the game engine detects contact between the virtual joint and the obstacle. At the beginning, the exercise ball exerts little resistive force on the finger, so a very small torque is applied to emulate the small resistive force noticeable from 3.1 to 6 seconds. As the joint goes deeper into the exercise ball a higher torque is applied by the motor which is noticeable between 6 to 8 seconds. This emulates the scenario of exercise ball resisting large deformations. As the finger is pulled from the ball at 10 seconds, the motor stops applying force on the finger. The following peaks in the chart represents finger lightly compressing the exercise ball and pulling out quickly. Torque values were limited to 250 to prevent vibration on the DC motor. In this demo, the ball changes color from pink to dark red depending on how hard the player is pressing the ball.

VI. CONCLUSION

In this paper, the construction of a single joint of a force feedback glove has been presented. Brushed DC motor with attached encoder provides joint movement information, which allows the game engine to track the physical finger with high accuracy. When the virtual finger collides with a virtual object, motors attached to the physical joint spin up to prevent the fingers from penetrating into the virtual object.

Compared to other VR gloves, our design provides a collection of desired features without requiring a lot of additional hardware. Transferring the finger joint rotation though the gear shaft of the motor to the incremental encoder allows capturing the joint motion very fast with high granularity. Also, instead of providing only on/off force feedback, controlling the torque produced by the DC motor allows one to experience interaction with virtual objects of different elasticity. The interaction of the gloves with hard and soft object has been tested by multiple participants, and they were able to discern if the object being interacted with was a hard or soft object.

In the next phase of this study, two more additional joints will be developed to track and provide feedback for a complete finger. Hand tracking will also be implemented so a game system can track the person's hand in a 3-dimensional space.

ACKNOWLEDGMENT

This research project is funded by Bennett Undergraduate Electrical Engineering Summer 2017 Research Fellowship at South Dakota State University, SD, USA.

REFERENCES

[1] L. W. Stark, "How virtual reality works: illusions of vision in 'real' and virtual environments," in *IS&T/SPIE's Symposium on Electronic Imaging: Science & Technology*, 1995, vol. 2411, pp. 277–287.

[2] J. Eddings, P. D. Wattenmaker, Linda/Illustrator-Wattenmaker, and P. Drury, *How virtual reality works*. Ziff-Davis Press, 1994.

[3] Rachel Metz, "The Step Needed to Make Virtual Reality More Real - MIT Technology Review," *technologyreview.com*, 2016. [Online]. Available: https://www.technologyreview.com/s/543316/the-step-needed-to-make-virtual-reality-more-real/. [Accessed: 15-Apr-2018].

[4] L. Dipietro, A. M. Sabatini, and P. Dario, "A Survey of Glove-Based Systems and Their Applications," *IEEE Trans. Syst. Man, Cybern. Part C (Applications Rev.*, vol. 38, no. 4, pp. 461–482, Jul. 2008.

[5] T. N. H. I. T. Nakagawa, "Force display for presentation of rigidity of virtual objects," *J. Robot. Mechatronics*, vol. 24, no. 1, pp. 39–42, 1992.

[6] "CyberGrasp Manual, version 2.0, Immersion Corporation, San Jose, CA." 2007.

[7] J. Blake and H. B. Gurocak, "Haptic Glove With MR Brakes for Virtual Reality," *IEEE/ASME Trans. Mechatronics*, vol. 14, no. 5, pp. 606–615, Oct. 2009.

[8] X. Gu, Y. Zhang, W. Sun, Y. Bian, D. Zhou, and P. O. Kristensson, "Dexmo," in *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems - CHI '16*, 2016, pp. 1991–1995.

[9] Z. MA and P. Ben-Tzvi, "RML Glove—An Exoskeleton Glove Mechanism With Haptics Feedback," *IEEE/ASME Trans. Mechatronics*, vol. 20, no. 2, pp. 641–652, Apr. 2015.

[10] T. Tarn, A. Bejczy, A. Isidori, and Y. Chen, "Nonlinear feedback in robot arm control," in *The 23rd IEEE Conference on Decision and Control*, 1984, pp. 736–751.

[11] G. I. Bain, N. Polites, B. G. Higgs, R. J. Heptinstall, and A. M. McGrath, "The functional range of motion of the finger joints," *J. Hand Surg. (European Vol.*, vol. 40, no. 4, pp. 406–411, May 2015.

[12] P. Rocco, "Stability of PID control for industrial robot arms," *IEEE Trans. Robot. Autom.*, vol. 12, no. 4, pp. 606–614, 1996.

[13] R. Kusterer, *jMonkeyEngine 3.0 Beginner's Guide*. Packt Publishing Ltd, 2013.