A touchless human-machine interface for the control of an elevator

Luca Montanaro
Vega S.R.L.
Fermo, Italy
luca.montanaro@vegalift.it

Paolo Sernani
Università Politecnica delle Marche
Ancona, Italy
p.sernani@univpm.it

Davide Calvaresi
Scuola Superiore Sant'Anna
Pisa, Italy
d.calvaresi@sssup.it

Aldo Franco Dragoni
Università Politecnica delle Marche
Ancona, Italy
a.f.dragoni@univpm.it

Abstract

We present a touchless interface based on gesture recognition for the control of an elevator. Interacting with the interface, a user can select and confirm the desired floor without touching any physical device. The interface has to guarantee a usability comparable to the habitual button panels: the users are supposed to use the elevator without any specific training, and its functionality has to be the same of traditional elevators. In addition to describing three possible implementations of the touchless interface, the paper provides two contributions: a comparison of two different technologies used as the interface inputs, and the results from 10 preliminary user tests performed to measure the perceived usability. The results are promising: two implementations out of three got an average score around 84 on the "Usability Metrics for User Experience".

1 Introduction

Touchless interfaces are useful in different application domains. An example is the Ambient Assisted Living, where voice recognition and gesture recognition are key technologies for assistive environments [Dra13]: beyond the tendency to be used for activity recognition [CCS+16], touchless interfaces enable the users to achieve a natural interaction with the available devices [AJS13]. In addition to the assistive support, a touchless interaction is useful in environments which require absolute sterility, such as operating rooms [OGS+14]. Touchless interfaces are exploited even in computer games, for pure entertainment as well as for serious purposes [KLJ04, HPKJ12].

The research described in this paper shares some features with the aforementioned domains: we present a touchless interface based on gesture recognition to control an elevator. The user can select and confirm the desired floor without touching any physical device. As in the Ambient Assisted Living, we want to provide a natural interaction with the available devices. Even if applications which require a sterile environment are not usual as in surgery, our approach could contribute to the hygiene of the elevator and its occupants, especially in places like hospitals. To this point, one could object that the button panels typical of elevators are intuitive and usable as they are. However, a touchless interface to control an elevator fits even for entertainment purposes. An example could be a scenic elevator of a skyscraper, where the touchless control could be integrated with augmented reality to let the user browse some information about the view.

In this paper, we provide two main contributions, based on a set of preliminary user tests performed with a simulated system:

- we compare two different technologies for the input device to recognize the user's gesture, one based on computer vision and one on electrical near-fields;
- we propose and evaluate three different implementations of our touchless interface.

The rest of this paper is organized as follows. Section 2 presents some related works, analyzing a sample of studies where touchless interfaces are at the service of different application domains. Section 3 describes our system, presenting the touchless interface to control an elevator. Section 4 details a preliminary evaluation of our system: after an assessment of two different input devices, we performed user tests to understand the perceived usability of three different implementations of the touchless interface. To conclude, Section 5 discusses the results and introduces future works.

2 Related works

The touchless interface presented in this paper is based on gesture recognition, i.e. the process by which gestures made by a user are the input data to devices and applications [Gee04, Tur14]. The importance of gesture recognition lies in building efficient human-machine interaction [MA07]. For such reason, we based our research on the perceived usability of our system during the touchless interaction.

As stated in the introduction, touchless interfaces and gesture recognition are common to multiple domains. Assistance, Ambient Assisted Living, and the Ambient Intelligence are relevant application domains for gesture cognition. For example, in [OCBM04] the authors presented a solution to perform the interaction with a computer system with head tracking and eye blinking, to replace the use of the mouse. In [BP11] the authors describe a gesture recognition application with the purpose to help an assisted person with activities of daily living such as interacting with appliances, switching lights on and off, and answering the door. In fact, using our interface the user executes hand gestures to interact with the surrounding ambient, i.e. the elevator: thus, our interface can be considered an Ambient Intelligence application.

In surgery, gesture recognition and touchless interfaces can be a powerful tool for patient data visualization in operating rooms [DP16, RRA+12, WSE+08], to preserve the complete sterility of such environments. In most applications, an elevator does not need absolute sterility. However, being public surfaces, button panels in elevators can be the source of bacteria colonization [KSR14, RWBG05]. Therefore, a touchless interface is useful in preserving the hygiene, especially in public places.

Gesture recognition is widely used in computer games: the video game market already includes applications which react to users' gestures. However, gesture recognition is used even in serious games, using entertainment to engage the users for the game serious purposes. Such purposes are manifold: examples are enhancing tourism [BKW08], promoting an active life [GLNM12], and supporting rehabilitation [BMC+09]. Our application could include entertainment, too. The touchless interaction might amaze technology enthusiast users. Moreover, in future, it might be integrated with augmented reality, to let the user browse some information such as the history of the building or some point of interests in scenic elevators.

3 A touchless interface for elevator control

The implemented system is a touchless interface to manage the control display inside an elevator. The only input of the interface is the movement of the user's hand to select the desired floor. Thus, the control of the elevator is based on gesture recognition. In such an environment, the interface needs to be compliant with the following requirements:

- users with no distinction of age, education level, habits, and experiences need to be able to control the elevator without a specific and deep training;
- the selection of the floor has to be based only on the user's hand movements, without any physical interface such as a button. Even buttons to turn on the recognition are excluded since the entire interaction has to be touchless;
- as in ordinary elevators, users can select more floors, and the number of false positive should be null.

In other words, the touchless interface has to be extremely intuitive, at least as much as the ordinary buttons used to control an elevator.

3.1 System interface

The interface is based on the tracking of the movements of a user's hand on the xy plane parallel to the display placed inside the elevator. We present three different interaction modes with the elevator controls:

- the first is based on two linear widgets, tracking separately the movements along the y-axis (for the floor selection) and the x-axis (for the floor confirmation);
- the second is based on a circular movement of the hand (for the selection) and on a waiting time (for the confirmation);
- the third replicates the interaction with the button panels on habitual elevators, where the user selects a button on the xy plane (for the floor selection) and confirms his selection with his finger's movement along the z-axis.

In all cases, the interaction starts when the distance of the user's hand from the display is under a fixed threshold: thus, the floor selection and confirmation are activated by the evaluation of the position along the z-axis. A normalization is necessary to track the position of the user's hand since the input devices usually return values in millimeters. By means of empirical tests, the position of the user's hand in the device sensitive area is transformed into a point on the visualization area of the interface. For example, along the x-axis, zero means extreme left and one means extreme right. The circular widget also requires a conversion in polar coordinates: the center of the elevator display is the center of the reference system.

As depicted in Figure 1, 2, and 3, some common features can be found in each widget: the interface always presents to the users the current position of the elevator, a bar to show the queue of the floor calls, and the floor the user is currently selecting.

3.1.1 Linear widgets

With the interaction mode based on linear widgets, the user selects the floor moving his hand along the y-axis (parallel to the elevator height) and confirms his selection moving his hand on the x-axis (parallel to the elevator base). The interface shown in Figure 1 gives a constant feedback to the user, through a display inside the elevator. While the user moves his hand along the y-axis, a vertical status bar at the center of the screen shows the floor that could be currently selected; when the user's hand reaches the desired floor, he can confirm the floor by moving his hand to the right, filling the label "confirm".

When the user confirms the selected floor, it might happen that he accidentally moves his hand along the y-axis. That gesture would result in an undesired behavior of the interface, hence a frustrating human-machine interaction and, in extreme cases, in a wrong action of the elevator, i.e. the selection of the wrong floor. To avoid such issues, we implemented a "debounce" mechanism: the time that the user spends with his hand on a selected floor increases a threshold which indicates the time to be spent pointing another floor to change the selection. Such threshold further increases when the user is confirming his choice by moving his hand on the x-axis. Thus, while the user is confirming a floor, the interface becomes less sensible to the change of the floor, reducing the risk of inadvertent choices.

Figure 1: The interface based on two linear widgets.

3.1.2 Circular widget

With the circular widget on the elevator display shown in Figure 2, the user selects the desired floor with a clockwise circular movement on the xy plane parallel to the display. The user confirms his choice by keeping the selection for a fixed minimum time.

To avoid inadvertent changes of the selected floor, the "debounce" mechanism for the circular widget acts by enlarging the movement needed to change the selection, proportionally to the time the user spends in the current selection. Thus, the floor selection is fluid, and during the confirmation larger movements are required to intentionally change the floor.

Figure 2: The interface based on the circular widget.

3.1.3 Button widget

The buttons widget presented in Figure 3 replicates the button panel usually available on elevators: the user has to move his finger near the display on the xy plane to select a floor, and move the finger towards the display over a certain threshold to confirm the selection. As in the previous case, the interaction is based only on the user gestures and does not require a direct touch by the user.

The buttons widget does not have a "debounce" mechanism implemented, since the area involved in the interaction is clearly shown on the elevator display.

3.2 Software Structure

The entire software system manages the recognition of user gestures, the simulation of the elevator move-

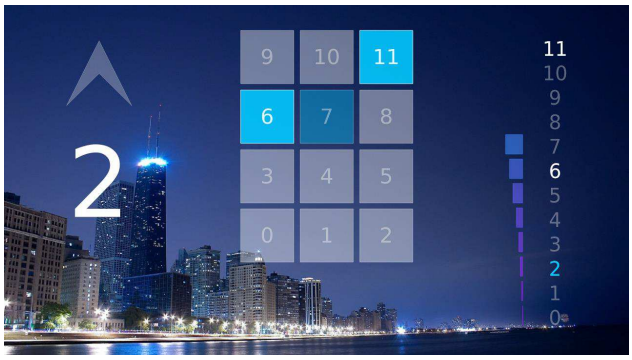

Figure 3: The interface based on the button widget.

ments (to execute the user tests), the updating of the visual interface, and the audio feedback given to the user during the interactions. The system is implemented as a multithreaded application, based on state transitions. Such structure is the same for each of the presented widgets.

Three are the possible states, i.e. "wait", "select", and "confirm"; Figure 4 depicts the state transitions.

In the "wait" state, an activation thread is continuously running and listening for an event to trigger the floor selection. As described above, the floor selection starts when the user's hand is under a threshold distance from the elevator display, along the z-axis. If such event occurs, the activation thread creates two new threads, to monitor the position of the user's hand for the floor selection and confirmation, according to the widget in use. Hence, the application goes to the "select" state. If the user cancels the operation (by moving his hand out of the threshold distance before the confirmation) the application goes back to the "wait" state and the selection and confirmation threads are killed. Otherwise, when the user confirms his selection, the application goes to the "confirm" state: the application adds the selected floor to the service queue, plays a feedback animation on the display and a confirmation sound. Then, the selection and confirmation threads are killed, and the application goes back to the "wait" state.

The elevator movements are simulated by means of a dedicated thread, which implements the elevator algorithm (the traditional SCAN algorithm known in disk scheduling [SGG12]). The interface is updated by a separated thread which shares global variables with the activation, selection, and confirmation threads, as well as with the simulation thread. Similarly, a dedicated thread manages the application audio, to give feedback to the user: the activation, selection, and confirmation threads put the references to the audio files on a queue, consumed by the audio management thread, according to the producer-consumer pattern.

The system is implemented in C++ and the visual interface is built on top of the "Qt" library[1].

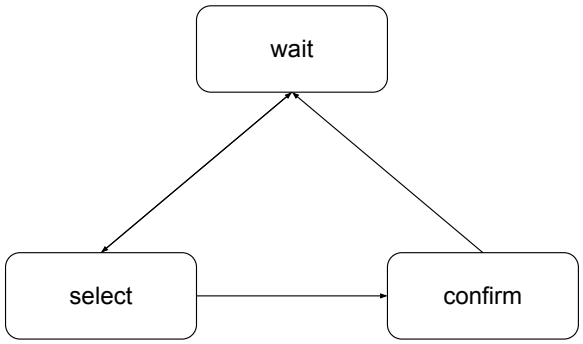

Figure 4: Application states and transitions.

3.3 Hardware Components

We implemented the system with two different technologies: the "Leap Motion Controller"[2] and the Microchip "MGC3130" based on the Microchip "GestIC" technology[3]. In both cases the interface and the gestures required to control the elevator are the same: the difference relies on the used gesture recognition techniques. With the "Leap Motion", the recognition is computer vision-based, with algorithms applied to a stereo grayscale image. With the "GestIC", the movements are detected with an electrical near-field generated by the device (the "MGC3130"), and thus no images are required.

4 Evaluation

To evaluate the system, we ran preliminary tests to compare the "Leap Motion" and the Microchip "GestIC" for the purpose of the gesture-based control of an elevator. Then, we ran usability tests with ten users, to evaluate the perceived usability during the interaction with the system interface. Such usability tests are based on the "Usability Metric for User Experience" (UMUX) [Fin10], a Likert scale used for the subjective assessment of an application's perceived usability, based on the following four items:

1. this system's capabilities meet my requirements;
2. using this system is a frustrating experience;
3. this system is easy to use;
4. I have to spend too much time correcting things with this system.

[1] https://www.qt.io/
[2] https://www.leapmotion.com/
[3] http://www.microchip.com/design-centers/touch-input-sensing/gestic-technology/overview

The user can assign a value from 1 (strongly disagree) to 7 (strongly agree) to each item. The positive items (1 and 3) are scored as $[user's\ value - 1]$, while the negative items (2 and 4) are scored as $[7 - user's\ value]$. The UMUX score of a test is a value between 0 and 100, given by the sum of the four items' scores divided by 24 and multiplied by 100.

We ran the tests in lab settings, and the users had to perform simple tasks, i.e. the selection and the confirmation of some floors with the implemented widgets, presented in a random order. Five out of ten users who performed the tests are male. Five users are between 20 and 30 years old; one is between 30 and 40 years old, two between 40 and 50, and two between 50 and 60. Four users have a secondary school education level; two users hold a bachelor degree or equivalent; four users have a master degree or equivalent.

The main threat to the validity of the tests is that the interface is simulated, and thus the users performed the task in front of a PC, instead of inside a real elevator. Nevertheless, the evaluation is focused on the perceived usability during the interaction with the system interface: the input device and the interaction area would be the same even in a real environment.

4.1 Technology comparison

The different devices used to take the user's gestures as the system input require different resources to perform the gesture recognition: the "Leap Motion" is more resource-demanding than the "MGC3130". In fact, the "Leap Motion" generates a grayscale stereo image which has to be processed with computer vision algorithms while the "GestIC" technology returns the coordinates of the hand on the sensible area thanks to the perturbation of the electrical near field generated by the device. Despite such feature makes the "MGC3130" more adequate to an embedded context, in our opinion the "Leap Motion" allows a better user experience for our application, since:

- the "GestIC" allows a maximum range of interaction of 15 cm, while the "Leap Motion" has a range of interaction around 60 cm, allowing larger gestures;

- the sensitive area of the "GestIC" is limited to a 7 inches diagonal. Therefore, the gesture-based interaction can be overlapped to displays of 7 inches at most. To mitigate such issue, we executed tests trying to separate the interaction area from the display, using a 7 inches area beside a larger monitor. However, such expedient interferes with the natural usability and the direct feedback available by moving the hand exactly in front of the display.

Hence, we decided to use only the "Leap Motion" for the usability tests described in Subsection 4.2.

4.2 Interface usability

Table 1 presents the results of the UMUX questionnaire filled by the users. The results are promising and show a high acceptance of the gesture-based interaction: the best user experience is provided by the interfaces based on the linear widgets and the buttons widget, with an average score of 84.27 and 83.40 respectively. However, as highlighted in Figure 5, the linear widgets get more homogeneous results (standard deviation 12.62): one person had a perceived usability between 61 and 70, four people between 71 and 80, one person between 81 and 90 and four people between 91 and 100. The buttons widget gets more distributed scores: in two cases the interface scored lower than 60.

During the tests with all the widgets, some users felt disoriented at the beginning of the interaction: they did not immediately understand how to control the interface and find the correct distance to interact with it. Such issues were more relevant with the interface based on the circular widget which got an average score of 70.30 (standard deviation 24.81). As shown in Figure 5, the perceived usability scored less than 50 in two tests with the circular widget. The solution to mitigate such issues could be a short demo running on the interface display with audio instructions.

Table 1: The results on the perceived usability, based on the Usability Metric for User Experience (UMUX) [Fin10].

	average score	**std dev**
Linear widgets	84.27	12.62
Circular widget	70.30	24.81
Buttons widget	83.40	19.15

5 Conclusions

We presented a touchless interface to control an elevator by means of hand gestures. We described three different modes of interaction based on three implementations of the system interface, i.e. the linear widgets, the circular widget, and the buttons widget. The linear widgets use the vertical movements of the user's hand for the floor selection and the horizontal movements for the confirmation. The circular widget uses a circular movement for the selection and a wait for the confirmation. The buttons widget replicates the conventional button panel of an elevator. The interaction starts when the user puts his hand nearer than a fixed distance from the elevator display. Then, he selects and confirms the desired floor by moving his

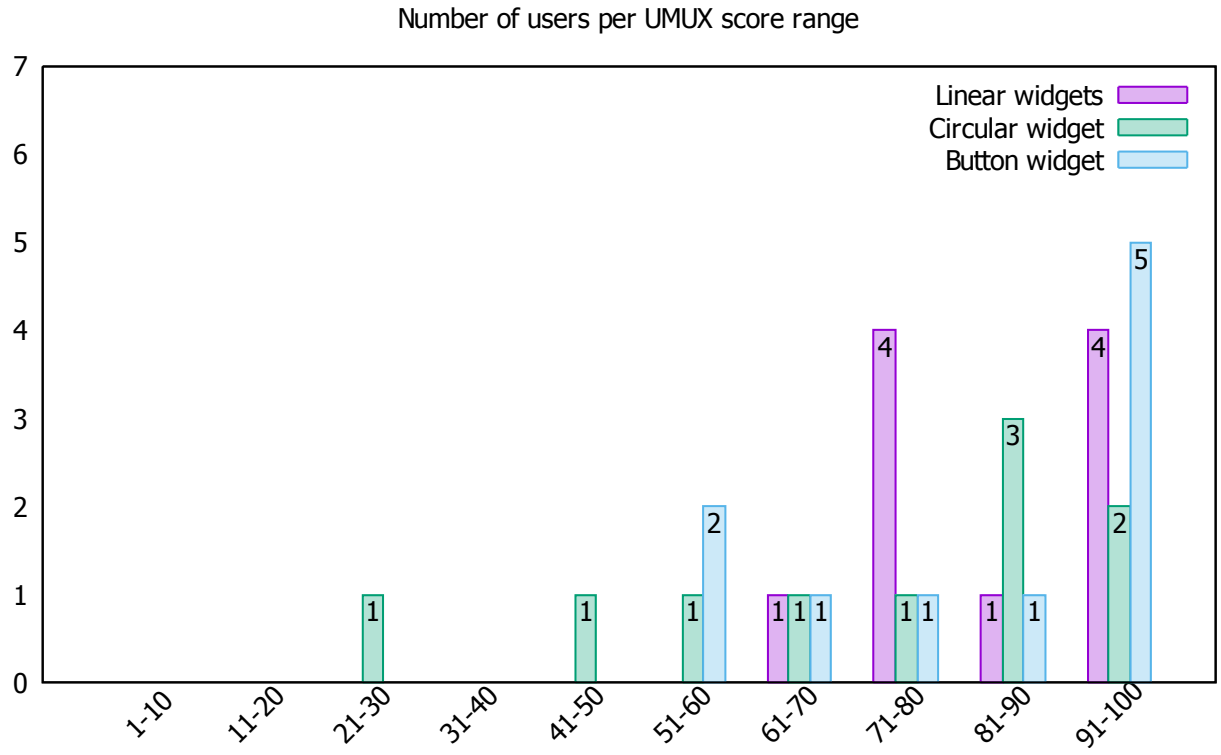

Figure 5: The distribution of the UMUX scores: the linear widgets got the best results, with 9 tests over 71.

hand, according to the widget displayed on the elevator monitor, receiving a constant feedback.

We performed preliminary tests with two different input devices:

- the "Leap Motion Controller", to recognize gestures with computer vision;
- the Microchip "MGC3130", to perform the recognition with the electrical near-field technology Microchip "GestIC".

Despite the "MGC3130" is suitable for an embedded context (such as the control of an elevator) we decided to use the "Leap Motion" for the usability tests. In fact, the "Leap Motion" has a larger sensitive area which results in larger movements allowed to the users, especially in front of displays greater than 7 inches.

We ran the usability tests with ten users using a simulated interface. The perceived usability is encouraging: the average UMUX score of the linear widgets is 84.27 with a standard deviation of 12.62. Thus, the users accepted the touchless interaction without feeling disoriented: after an initial explanation, they were all able to select and confirm a floor. The buttons widget had similar results. On the contrary, the circular widget had the most problematic usability, scoring 70.30 on the UMUX scale: the users experienced difficulties in immediately understanding how to select a floor.

The user tests also highlighted:

- the need of support tools to let the users immediately understand the flow of interaction. For example, a short demo running on a corner of the elevator display could help users in the first impact with the interface;
- the need of enough room to interact with the interface, to avoid unintended choices or even the impossibility to make the necessary gestures.

The development of support tools to make the interface easier to understand is ongoing work. However, user tests inside a real elevator are the only viable solution to fully validate the idea of a touchless control of an elevator.

References

[AJS13] Dimitra Anastasiou, Cui Jian, and Christoph Stahl. A german-chinese speech-gesture behavioural corpus of device control in a smart home. In *Proceedings of the 6th International Conference on PErvasive Technologies Related to Assistive Environments*, PETRA '13, pages 62:1–62:6. ACM, 2013.

[BKW08] R. Ballagas, A. Kuntze, and S.P. Walz. Gaming tourism: Lessons from evaluating rexplorer, a pervasive game for tourists. In *Pervasive Computing: 6th International Conference, Pervasive 2008 Sydney, Australia, May 19-22, 2008 Proceedings*, pages 244–261, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg.

[BMC+09] J.W. Burke, M.D.J. McNeill, D.K. Charles, P.J. Morrow, J.H. Crosbie, and S.M. McDonough. Optimising engagement for stroke rehabilitation using serious games. *The Visual Computer*, 25(12):1085–1099, 2009.

[BP11] M. Bhuiyan and R. Picking. A gesture controlled user interface for inclusive design and evaluative study of its usability. *Journal of software engineering and applications*, 4:513–521, 2011.

[CCS+16] D. Calvaresi, D. Cesarini, P. Sernani, M. Marinoni, A.F. Dragoni, and A. Sturm. Exploring the ambient assisted living domain: a systematic review. *Journal of Ambient Intelligence and Humanized Computing*, pages 1–19, 2016.

[DP16] L.T. De Paolis. A touchless gestural platform for the interaction with the patients data. In E. Kyriacou, S. Christofides, and C.S. Pattichis, editors, *XIV Mediterranean Conference on Medical and Biological Engineering and Computing 2016: MEDICON 2016*, pages 874–878. Springer International Publishing, 2016.

[Dra13] A.F. Dragoni. Virtual carer: A first prototype. In *Telehealth Networks for Hospital Services: New Methodologies*, pages 290–299. IGI Global, 2013.

[Fin10] K. Finstad. The usability metric for user experience. *Interacting with Computers*, 22(5):323 – 327, 2010.

[Gee04] D. Geer. Will gesture recognition technology point the way? *Computer*, 37(10):20–23, 2004.

[GLNM12] Kathrin Gerling, Ian Livingston, Lennart Nacke, and Regan Mandryk. Full-body motion-based game interaction for older adults. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '12, pages 1873–1882. ACM, 2012.

[HPKJ12] G.F. He, J.W. Park, S.K. Kang, and S.T. Jung. Development of gesture recognition-based serious games. In *Proceedings of 2012 IEEE-EMBS International Conference on Biomedical and Health Informatics*, pages 922–925, 2012.

[KLJ04] H. Kang, C.W. Lee, and K. Jung. Recognition-based gesture spotting in video games. *Pattern Recognition Letters*, 25(15):1701 – 1714, 2004.

[KSR14] C.E. Kandel, A.E. Simor, and D.A. Redelmeier. Elevator buttons as unrecognized sources of bacterial colonization in hospitals. *Open Medicine*, 8(3):e81–e86, 2014.

[MA07] S. Mitra and T. Acharya. Gesture recognition: A survey. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 37(3):311–324, 2007.

[OCBM04] R. O'Grady, C. J. Cohen, G. Beach, and G. Moody. Navigaze: enabling access to digital media for the profoundly disabled. In *Information Theory, 2004. ISIT 2004. Proceedings. International Symposium on*, pages 211–216, 2004.

[OGS+14] K. O'Hara, G. Gonzalez, A. Sellen, G. Penney, A. Varnavas, H. Mentis, A. Criminisi, R. Corish, M. Rouncefield, N. Dastur, and T. Carrell. Touchless interaction in surgery. *Communications of the ACM*, 57(1):70–77, 2014.

[RRA+12] G.C. Ruppert, L.O. Reis, P.H. Amorim, T.F. de Moraes, and J.V. da Silva. Touchless gesture user interface for interactive image visualization in urological surgery. *World Journal of Urology*, 30(5):687–691, 2012.

[RWBG05] K.A. Reynolds, P.M. Watt, S.A. Boone, and C.P. Gerba. Occurrence of bacteria and biochemical markers on public surfaces. *International Journal of Environmental Health Research*, 15(3):225–234, 2005.

[SGG12] A. Silberschatz, P.B. Galvin, and G Gagne. *Operating system concepts, 9th Ed.* John Wiley & Sons, 2012.

[Tur14] M. Turk. Gesture recognition. In K. Ikeuchi, editor, *Computer Vision: A*

Reference Guide, pages 346–349. Springer US, 2014.

[WSE+08] J.P. Wachs, H.I. Stern, Y. Edan, M. Gillam, J. Handler, C. Feied, and M. Smith. A gesture-based tool for sterile browsing of radiology images. *Journal of the American Medical Informatics Association*, 15(3):321–323, 2008.