Smartphone Teleoperation for Self-Balancing Telepresence Robots

Antti E. Ainasoja[1], Said Pertuz[2] and Joni-Kristian Kämäräinen[3]

All authors are with Laboratory of Signal Processing, Tampere University of Technology, Finland

[1] *https://orcid.org/0000-0003-2016-3870* [2] *https://orcid.org/0000-0001-8498-9917*
[3] *https://orcid.org/0000-0002-5801-4371*

Keywords: Teleoperation, Telepresence, User Interface.

Abstract: Self-balancing mobile platforms have recently been adopted in many applications thanks to their light-weight and slim build. However, inherent instability in their behaviour makes both manual and autonomous operation more challenging as compared to traditional self-standing platforms. In this work, we experimentally evaluate three teleoperation user interface approaches to remotely control a self-balancing telepresence platform: 1) touchscreen button user interface, 2) tilt user interface and 3) hybrid touchscreen-tilt user interface. We provide evaluation in quantitative terms based on user trajectories and recorded control data, and qualitative findings from user surveys. Both quantitative and qualitative results support our finding that the hybrid user interface (a speed slider with tilt turn) is a suitable approach for smartphone-based teleoperation of self-balancing telepresence robots. We also introduce a client-server based multi-user telepresence architecture using open source tools.

1 INTRODUCTION

Telepresence can provide substantial aid in certain applications of welfare services, such as remotely visiting homes of self-living elderly (Apostolopoulos et al., 2012; Leeb et al., 2015). In this scope, it is desirable that these telepresence-based welfare services allow for a teleoperation via mobile devices since their users, e.g. nurses and health care providers, should have access to quick and effective interaction. Due to the flexibility and ease of access of mobile devices, the combination of novel user interfaces and smartphone-based teleoperation arise as a potential alternative for these applications (Fong et al., 2001).

In the literature, several researchers have proposed different approaches for the teleoperation of telepresence platforms, such as haptic interfaces, or touch screens (Ogata et al., 2015), and tilt user interface (Findlater et al., 2013; Baldauf et al., 2015). However, conclusive results on identifying the advantages and disadvantages of each approach in the context of self-balancing platforms are lacking and, therefore, further research is warranted. In this work, we first develop an architecture for the remote control of a self-balancing "Segway-type" telepresence robot. Subsequently, we compare three different smartphone teleoperation strategies: touchscreen button user interface, tilt user interface, and hybrid touchscreen-tilt user interface.

Telepresence platforms have shown their potential for human assistance in healthcare environments (Boissy et al., 2007; Tiberio et al., 2012). In turn, self-balancing robotic platforms are gaining momentum in the community due to their slim and light-weight build (Tsai et al., 2010). However, these platforms pose certain challenges in term of stability and control. The teleoperation through wireless networks introduce delays in robot response to commands. Furthermore, the self-balancing mechanism prevents the robot for speeding up, slowing down or stopping instantaneously. Teleoperation user interfaces must be comfortable to the operators and feel responsive even with delayed execution of commands. This work provides an experimental baseline for the assessment of the effectiveness of teleoperation user interface strategies for self-balancing telepresence platforms.

The main contributions of this work can be summarized as follows:

- We develop a client-server architecture for teleoperation using open source tools.
- We develop three intuitive but distinctly different user interfaces for operating a self-balancing telepresence robot based on three different strategies: 1) touchscreen button user interface, 2) tilt user interface and, 3) hybrid touchscreen-tilt user interface.
- We propose a navigation task and evaluation crite-

ria to compare the implemented teleoperation user interface strategies.

Experimental results from both quantitative and qualitative evaluation with random users indicate that the hybrid interface provides best performance in terms of steadiness of driving and user preference. Our source code and examples will be made publicly available to facilitate further development of new features and applications of telepresence for supporting self-living elderly.

2 RELATED WORK

The use of smartphone-based user interface for telepresence robots can be traced back to (Fong et al., 2001). In that work, they focused on the motivation and design criteria for different user interface strategies. More recently, improved user interface by robot autonomy was proposed by Vaughan *et al.* (Vaughan et al., 2016), who adopted the "reaching through the screen" concept (Foote et al., 2004). In their system, the telepresence bot is equipped with a fisheye lens that provides broader view where a remote user clicks the path and indicates the goal. ROS-based visual navigation controls the movement towards the goal location using the path points. Bohren *et al.* (Bohren et al., 2016) take autonomy to a higher level by adopting a prediction model where robot operation is based on predicted user controls. A formal definition of prediction and user assistance in teleoperation was given by Dragan and Srinivasa (Dragan and Srinivasa, 2012). This strategy is particularly important in the presence of long network delays and a similar simulation model has been proposed for multi-player game client-server systems (Bernier, 2001). The aforementioned works have focused on the development of novel user interface strategies without a direct comparison of different approaches.

In the literature, a lot of efforts have been devoted to the improvement of the user experience in the operation of telepresence platforms either by incorporating new features or by providing autonomous capabilities. Kisevel *et al.* (Kiselev et al., 2015) studied the usage of semi-autonomous features by novice telepresence users. Tee *et al.* (Tee et al., 2014) showed improved telepresence experience in meetings by using audio-visual gesture and attention recognition. Schwarz *et al.* (Schwarz et al., 2014) provide adjustable autonomy for teleoperation where manual operation is requested when autonomy fails. Cosgun *et al.* (Cosgun et al., 2013) incorporate autonomous person following as a feature for telepresence platform.

Most efforts in the comparison of user interface strategies have been performed in the context of gaming or virtual reality. In this scope, the results by Findlater *et al.* (Findlater et al., 2013) indicate that modern touchscreen interfaces are easier to use for non-expert users than traditional "gaming setups" with mouse and keyboard for simple pointing and clicking and tracking task. Baldauf *et al.* (Baldauf et al., 2015) did an analysis on different mobile phone game controller interfaces and found that, while navigation buttons and virtual joystick are easier to use and more accurate than tilt interfaces, they also needed more attention from users in order to touch the controls precisely. These findings were based on 2D games on large screen where the view point was third person or from the bird eye perspective and suggested that tilt user interface may be more suitable for certain first person perspective games, such as flight simulators.

Most strategies considered in the state-of-the-art are mainly based on touchscreen user interfaces. In this work, we decided to include the tilt-based user interfaces, as suggested by Baldauf *et al.* (Baldauf et al., 2015) for first-person perspective games due to their similarities with telepresence. Remarkably, self-balancing telepresence is distinctly more challenging to operate than the traditional standing (three/four wheel) platforms (Shimada and Hatakeyama, 2008).

3 SELF-BALANCING TELEPRESENCE SYSTEM

The motivation for this work is the development of *a social robot platform with adjustable autonomy* to support self-living elderly. This particular application poses specific requirements from both the user and system perspective. Specifically, the platform must be based on affordable commodity hardware and open source software to allow agile development of new features and services by the open source community. In addition, the platform should guarantee that adjustable autonomy is available.

Adjustable autonomy in our case means that control can always be given to human users either when requested by active peers or when the platform cannot perform requested tasks autonomously. The lowest level of autonomy is *teleoperation* and, for the needs of different peers (nurses, relatives, elderly themselves and possible third parties), the teleoperation interface must be easy-to-use, affordable and intuitive. The teleoperation system developed for this purpose in this work is illustrated in Fig. 1. The system is comprised of four main components:

- Android client

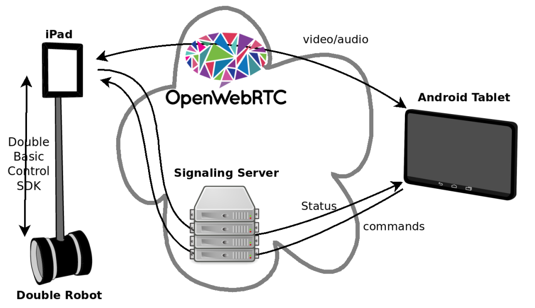

Figure 1: The most important parts of the telepresence system. The system includes a self-balancing Double telepresence robot controlled by a system that comprises an iPad, an Android tablet device and a Communication server.

- Communication server
- Telepresence robot[1]
- iPad control platform

Each one of the components of the system are detailed below.

Android Client: All the interactions between the user and the system are managed through a tablet running the *Android client*. In order to allow for a flexible operation of the telepresence robot, an Nvidia Shield K1 Android tablet was used. The table is equipped with a capacitive multi-touch screen, accelerometer and gyroscope. An application with several graphical user interfaces (GUI's) was developed in order to implement the operation strategies under study (section 4).

The Android client is connected to Internet through an LTE network. In order to have access to the video and audio streams, as well as for sending commands to the robot, the Android client is connected to the Communication server (see below) through an HTTP application running on a Linux host.

Communication Server: The job of the communication server (CS) is establishing the connection between the Android client (user) and the iPad used to control the Double robot upon request by the user. Once the connection is established, the CS manages the communication between the user and the iPad.

Depending on the type of signal, the communication goes through two different paths. On the one side, status signals reported by robot, as well as commands sent by the user from the Android client, go through the CS. The CS logs and sends the teleoperation commands received from the user. On the other side, video and audio signals from the iPad are transferred to the Android server using the OpenWebRTC[2] implementation of WebRTC[3] standard. The WebRTC protocol was chosen due to its flexibility, modularity and cross-platform interoperability. A sequence diagram illustrating connection initialization between clients and server is presented in Figure 2.

[1]https://www.doublerobotics.com/

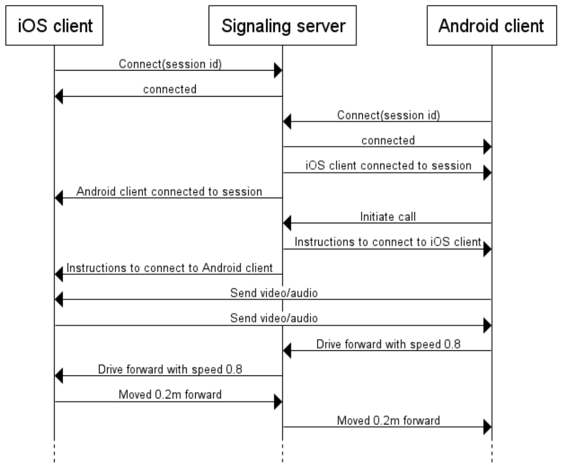
iOS client
Signaling server
Android client
Connect(session id)
connected
Connect(session id)
connected
iOS client connected to session
Android client connected to session
Initiate call
Instructions to connect to iOS client
Instructions to connect to Android client
Send video/audio
Send video/audio
Drive forward with speed 0.8
Drive forward with speed 0.8
Moved 0.2m forward
Moved 0.2m forward

Figure 2: A sequence diagram of iOS and Android clients making connection through signaling server (CS).

iPad Control Platform: An iPad tablet is used for direct control of the Double robot and for capturing the video and audio signal on the robot end. In order to facilitate navigation, the camera was equipped with a Black Eye full-frame fisheye lens with 180-degree field of view[4]. This was found to be a good compromise between viewing the surroundings of the robot and being able to see the objects in front of the robot in detail. As stated previously, the video and audio signals are sent to the user using WebRTC. For motion control of the telepresence platform, a custom application was developed based on the Double SDK for the iPad to delegate the commands to the robot.

4 SMARTPHONE USER INTERFACES

or comparison purposes, we implemented three user interfaces for the self-balancing two-wheel telepresence navigation system described in previous section: 1) *touchscreen user interface* (found best in (Baldauf et al., 2015) for third person interfaces in 2D games),

[2]https://www.openwebrtc.org

[3]https://www.w3.org/TR/webrtc/

[4]https://blackeyelens.com/product/full-frame-fisheye/

2) *Tilt user interface* (simplest interface requiring the least amount of user attention), and 3) *Hybrid* (tilt for turning and a slider for forward/backward). In all of our interfaces it is assumed that a user holds the tablet horizontally with both hands. The interfaces have a view of the remote video stream at the center and the control interfaces are on left and right sides of the stream view. Examples of the user interfaces are shown in Fig. 3. A more detailed explanation of the features of each implemented user interface strategy can be found below.

4.1 Touchscreen Button UI (B-GUI)

The navigation buttons interface consists of four buttons marked with arrows on each side of the remote video view. By pressing the buttons the robot moves forward or backward and turns left or right. Both the left and right hand side buttons worked identically and the intention was that they could be used together to accelerate/decelerate with one hand and steer with other, or use one hand for both turning and speeding. The buttons only have binary states so when the button is pressed the robot moves according to the direction at full speed. When no buttons are pressed, the robot stops. Due to the self-balancing nature of the robot, the acceleration is never immediate. Using linear or exponential acceleration over a short period of time after the button is pressed made the interface feel more sluggish and delayed.

4.2 Tilt Interface (T-GUI)

The tilt user interface uses the pitch and roll of the tablet relative to real world coordinates to operate the robot. The pitch is used to control the speed of the robot; tilting away (forward) makes the robot move forward and tilting back backwards. Likewise, the roll turns the robot left and right, respectively.

During the early experiments we noticed that some mechanism was needed to tell when the user interface was used – active – and when not. For this reason we placed dead man's switch-type buttons on the interface. In the initial setting, pressing only one of the buttons made the system active, but this was not sufficient since users, when distracted, kept holding the tablet touching a switch with one hand and subsequently lost the control of the robot. In the final interface, both dead man's switches need to be pressed simultaneously.

The tilt interface measured the angular displacement of the tablet from the neutral position. Displacement angles were mapped to the velocity (pitch) and turning speed (roll) of the robot. In order to make it easier to hold the robot still, displacement angles proportional to the full calibration scales and less than the threshold $\tau_\sigma = 10$ % from the neutral position were mapped to zero. It was noticed that, in order to have better control of the robot at lower velocities, linear mapping was not sufficient. We squared the angles to have more fine control at the lower velocities. For turning, linear mapping seemed more comfortable.

4.3 Hybrid Hnterface (BT-GUI)

The third implemented user interface was a hybrid of the two above. Steering of the robot is the same as in T-GUI and the velocity was controlled similar to B-GUI, but the button was replaced with a "spring slider" (right hand side of the GUI). A user drags the slider up and down to move the robot forward and backwards. Due to the spring effect the switch returns immediately to the neutral position when released. The position of the switch was linearly mapped to the robot velocity. One dead man's switch was placed on the left hand side to allow turning the robot without moving it backwards or forwards. Turning works only when either the dead man's switch or the slider switch are touched.

5 ASSESSMENT METHODOLOGY

In order to assess the performance of each operating strategy, users were asked to perform a simple task using one or more approach in random order and different performance indicators were measured. The details about the performance indicators are presented in Section 5.1 and Section 5.2. A summary of the performance indicators can be found in Fig. 4 The description of the task and testing environment assigned to users are presented in Section 5.3.

5.1 Quantitative Performance Indicators

Quantitative performance indicators were divided in two groups: indicators aimed at measuring the effectiveness of the assigned task, and indicators aimed at measuring the driving style and comfort of the user.

Driving Effectiveness: The driving effectiveness is aimed at assessing how effective the teleoperation strategies were in the completion of the assigned task, regardless of the driving style or perceived comfort.

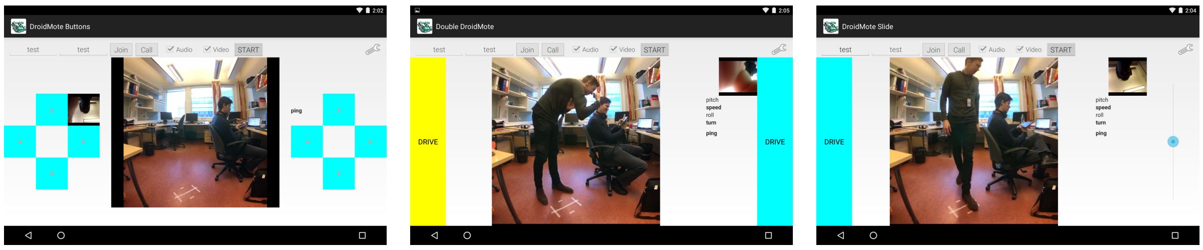

Figure 3: Developed interfaces for the implemented teleoperation strategies. (a) touchscreen interface with navigation buttons, (b) tilt interface and, (b) Hybrid touchscreen-tilt interface. The two tilt-based interfaces (b)-(c) include a dead man's switch ("Drive" button in the GUI's) that turns the tilt interface on/off.

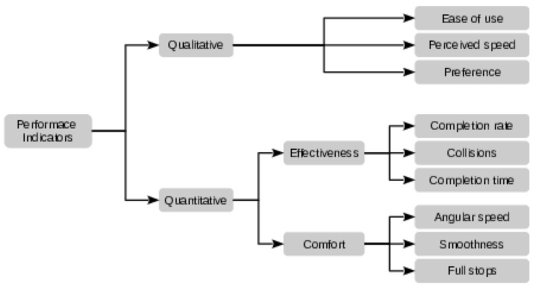

Figure 4: Performance indicators. Quantitative indicators were measured by analyzing the logged signals from the telepresence platform. Qualitative indicators were assessed by means of user surveys.

Indicators aimed at measuring the effectiveness were three:

- Completion rate
- Number of collisions
- Completion time

From these indicators, the most important one is the completion rate; whether or not the user could maneuver the robot safely, or at all. The amount of collisions was measured by observing the amount of crashes during each test drive. The completion time indicates how fast the operators are able to complete the task.

Driving Comfort: The indicators for driving comfort are aimed at assessing how smooth and steady is the operation of the platform. For this purpose, we defined three performance indicators:

- Average angular speed
- Smoothness of the trajectory
- Number of full stops

Both the average angular speed and the smoothness of the trajectory are aimed at measuring how stable is the trajectory of the platform. Ideally, the robot should move following a smooth trajectory without sudden changes in direction or speed. For this purpose, given a parametric trajectory $\{x(t), y(t)\}$ as a function of time t, the instantaneous steering angle is given by:

$$\theta(t) \approx \tan^{-1}\left(\frac{x(t+\Delta t)-x(t)}{y(t+\Delta t)-y(t)}\right), \tag{1}$$

where Δt is the time step used when recording the data of the telepresence platform.

The average angular speed, ω_{mean}, is then measured as:

$$\omega_{mean} = \frac{1}{N}\sum_{n=1}^{N}\frac{d\theta(t)}{dt}dt, \tag{2}$$

where N is the number of time stamps.

The smoothness of the trajectory is measured by estimating the Laplacian in the x and y directions, as:

$$\text{Smoothness} = |\nabla^2 x(t)| + |\nabla^2 y(t)| \tag{3}$$

Finally the number of full-stops was measured in order to record how confidently the users could perform the task. It was assumed that stopping was done when the operators felt they were about to lose control of the robot. A full stop was assumed when the operator sent no commands to the robot and robot was not moving. Specifically, the amount of robot motion was computed as:

$$\text{Robot motion} = |\frac{v}{v_{max}}| + |\frac{\omega}{\omega_{max}}|, \tag{4}$$

where v and ω are the instantaneous velocity and angular velocity, v_{max} is the maximum velocity (0.5 m/s) and ω_{max} is the maximum angular velocity (0.586 rad/s) of the robot.

Due to the self-balancing mechanism, the robot is constantly swaying back and forth, even when stationary. This was accounted by setting a threshold level on the motion. The robot was considered fully stopped if the amount of motion was below the threshold of 0.05. The robot motion and the mechanism for the detection of full stops is illustrated in Figure 5.

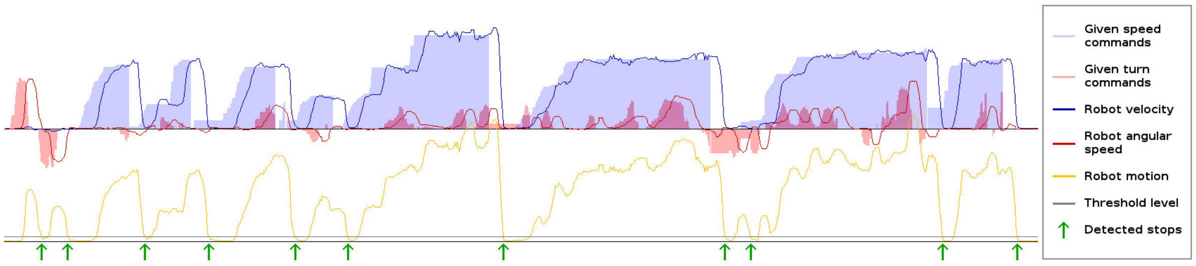

Figure 5: An example of sent commands in relation to the robot motion over time. The robot is considered to be in full stop when the yellow line is below the threshold level.

5.2 Qualitative Performance Indicators

An important performance indicator is how the operators subjectively felt about each operation strategy. A short survey was created for this purpose in order to assess three different aspects:

- Ease of use
- Perceived speed
- User preference

On the survey, operators were asked to give feedback on how they felt about the *ease of use* of each user interface approach. Each approach was assigned a qualification of 0, 1, 2 or 3, corresponding to four levels of difficulty: very easy, easy, difficult and very difficult. Users were also surveyed on how fast they perceived that they could perform the given task. This was to assess the consistency between objective quantitative performance indicators, such as the completion time, and the perceived user experience. Finally, operators who tested more than one user interface approach were asked which one they preferred. If a n user interface feels too difficult and stressful, it would not be used by the operators even if it was safe and fast.

5.3 Navigation Task and Environment

In order to collect quantitative and qualitative indicators, users were to maneuver the robot around an approximately 50 m long route in well-lit, populated lobby area (Fig. 6). Due to people having different ways of holding a tablet, the tilt and hybrid interfaces had initially a calibration tool for configuring the neutral (no movement) and extreme positions (full forward, full backward and extreme left and right) for each user. In preliminary tests, this was found to be too time consuming and complicated. Therefore the experiments reported in this work were done with a common pre-calibration. For controlling speed, the neutral position was set to 30°with 0°and 60°extreme

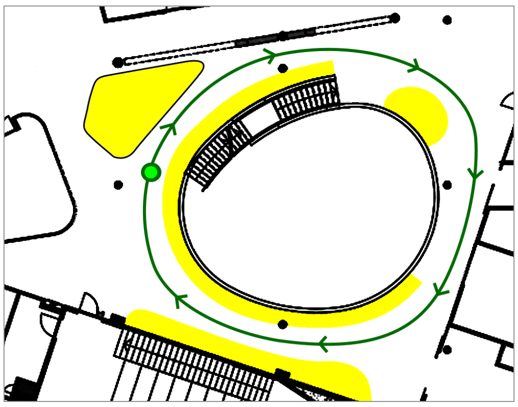

Figure 6: Map of the experiment area. Yellow denotes working areas with non-fixed furniture. Green dot is the start/finish location and green line is the ideal route of the task.

positions. For turning, the neutral position was set to 0°with -45°and 45°extreme positions. During initial experiments users reported the fast acceleration and speed to be difficult when using the touchscreen button interface. For the touchscreen button interface, velocity was therefore limited to 75% of the robot's maximum velocity.

6 RESULTS AND DISCUSSION

We recruited 21 participants (18 males, 3 females) with ages ranging from 21 to 47 (mean 27.1). 13 participants completed the navigation task described in section 5.3 using all the three user interfaces. Reported quantitative indicators were collected from the logged navigation signals and robots internal measurement unit (received by the platform), whereas qualitative results were obtained from user surveys after completing the task.

Table 1: Quantitative indicators for driving effectiveness.

UI	*Completion rate*	*Avg. collisions*	*Avg.time (std)*	*N*
B-GUI	0.94	**0.13**	220 (42) s	15
T-GUI	**1.0**	0.23	**198** (55) s	13
BT-GUI	0.91	0.33	211 (71) s	10

Table 2: Quantitative indicators for driving comfort.

UI	*stops (std)*	*angular speed (std)*	*Smoothness (std)*	*N*
B-GUI	8.2 (6.8)	0.1641 (0.1035)	0.7668 (1.1819)	15
T-GUI	**7.7** (6.5)	0.1385 (0.0180)	0.4907 (0.1105)	13
BT-GUI	7.8 (5.1)	**0.1337 (0.0158)**	**0.3596 (0.0876)**	10

6.1 Quantitative Indicators

The average indicators for the different interfaces are shown in Table 1 (Driving effectiveness) and Table 2 (Driving comfort). Tests that failed to be completed due to network failures (1 for T-GUI, 5 for BT-GUI) were omitted from quantitative data. Cases where users fatally crashed the robot (task could not be completed) are included in the completion rate, but not omitted from other qualitative data.

The tilt interface (T-GUI) is the most effective in the average task completion time. In driving comfort, however, the hybrid interface is clearly superior in smoothness and also the best in average angular speed (the lower values in table 2 indicate a better performance). The number of full stops are similar for all methods. The quantitative indicators do not provide a clear winner, but the tilt and hybrid interfaces seem to perform better than the plain touchscreen interface.

6.2 Qualitative Indicators

Results based on subjective feedback are summarized in Tables 3 and 4. Users found controlling the robot on all of the user interfaces to be easy. The two interfaces using tilt (T-GUI and BT-GUI) had a perceived speed faster than the touchscreen-only interface (Table 3). The user preference in Table 4 support this observation.

Slide interface was preferred by users with tilt being the second favorite. Some users found it difficult to use tilting for both turning and acceleration. This was partly due to the interface's sensitivity to calibration and the position in which the table was held. Some users found it difficult to hold the tablet in a position they were not accustomed to. Those preferring the buttons reported that it was easier to perceive how the robot moves using them.

Table 3: The mean ease of use and perceived speeds reported for each user interface on a scale from very easy/very slow = 0 to very difficult/very fast = 3.

UI	*Ease of use*	*Perceived speed*
B-GUI	**0.93**	1.64
T-GUI	1.00	**1.16**
BT-GUI	0.96	1.21

Table 4: Teleoperation interface preferences of people who tested all three interfaces.

B-GUI	*T-GUI*	*BT-GUI*
15%	35%	**50%**

Haptic feedback and better indication on turning speed was suggested for the tilt and slide interfaces. Not directly related to the interface schemes, the user generally found estimating the robot dimensions in relation to the surroundings difficult. This was due to the robot not being visible to the operator. An augmentation to cast a shadow on the robot's projected path could make it easier to avoid obstacles and go through tight spots.

6.3 Other Findings

The communication with the telepresence platform relied on good wireless networks. The delay between giving a command and robot reaction was usually around 300 ms, but occasionally it could go up to several seconds or connection was lost altogether.

7 CONCLUSIONS

In this paper we developed and implemented an architecture for the teleoperation of self-balancing telepresence platform using open source tools. The goal of our architecture is to foster open community development of social robotics applications for independent living elderly.

The first obvious requirement is user-friendly teleoperation user interface for the cases an operator is needed. To study teleoperation of a self-balancing bot we implemented three different user interface strategies: touchscreen, tilt and hybrid (a combination of tilt and touchscreen). Quantitative and qualitative measurements on a real navigation task suggest that the hybrid interface yields the best performance in terms of both driving comfort and user preference, tilt being the second best with small margin and touchscreen buttons being the worst with clear margin.

ACKNOWLEDGEMENTS

This work was funded by Academy of Finland project: "Robots and the Future of Welfare Services", decision number: 314180.

REFERENCES

Apostolopoulos, J. G., Chou, P. A., Culbertson, B., Kalker, T., Trott, M. D., and Wee, S. (2012). The road to immersive communication. *Proceedings of the IEEE*, 100(4):974–990.

Baldauf, M., Fröhlich, P., Adegeye, F., and Suette, S. (2015). Investigating on-screen gamepad designs for smartphone-controlled video games. *ACM Trans. Multimedia Comput. Commun. Appl.*, 12(1s):22:1–22:21.

Bernier, Y. (2001). Latency compensating methods in client/server in-game protocol design and optimization. In *Game Developers Conference*.

Bohren, J., Paxton, C., Howarth, R., Hager, G., and Whitcomb, L. (2016). Semi-autonomous telerobotic assembly over high-latency networks. In *ICRA*.

Boissy, P., Corriveau, H., Michaud, F., Labonté, D., and Royer, M.-P. (2007). A qualitative study of in-home robotic telepresence for home care of community-living elderly subjects. *Journal of Telemedicine and Telecare*, 13(2):79–84. PMID: 17359571.

Cosgun, A., Florencio, D., and Christensen, H. (2013). Autonomous person following for telepresence robots. In *ICRA*.

Dragan, A. and Srinivasa, S. (2012). Formalizing assistive teleoperation. In *RSS*.

Findlater, L., Froehlich, J. E., Fattal, K., Wobbrock, J. O., and Dastyar, T. (2013). Age-related differences in performance with touchscreens compared to traditional mouse input. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '13, pages 343–346, New York, NY, USA. ACM.

Fong, T., Conti, F., Grange, S., and Baur, C. (2001). Novel interfaces for remote driving: gesture, haptic, and pda. In *Proc. SPIE 4195, Mobile Robots XV and Telemanipulator and Telepresence Technologies VII.*

Foote, J., Liu, Q., Kimber, D., Chiu, P., and Zhao, F. (2004). Reach-through-the-screen: A new metaphor for remote collaboration. In *Advances in Multimedia Information Processing (PCM).*

Kiselev, A., Kristoffersson, A., Melendez, F., Galindo, C., Loutfi, A., Gonzalez-Jimenez, J., and Coradeschi, S. (2015). Evaluation of using semi-autonomy features in mobile robotic telepresence systems. In *Robotics, Automation and Mechatronics (RAM).*

Leeb, R., Tonin, L., Rohm, M., Desideri, L., Carlson, T., and d. R. Millán, J. (2015). Towards independence: A bci telepresence robot for people with severe motor disabilities. *Proceedings of the IEEE*, 103(6):969–982.

Ogata, M., Teramura, R., and Imai, M. (2015). Attractive telepresence communication with movable and touchable display robot. In *2015 24th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, pages 179–184.

Schwarz, M., Stuckler, J., and Behnke, S. (2014). Mobile teleoperation interfaces with adjustable autonomy for personal service robots. In *ACM/IEEE International Conference on Human-Robot Interaction*.

Shimada, A. and Hatakeyama, N. (2008). Movement control of two-wheeled inverted pendulum robots considering robustness. In *2008 SICE Annual Conference*, pages 3361–3366.

Tee, K., Yan, R., Chua, Y., Huang, Z., and Liemhetcharat, S. (2014). Gesture-based attention direction for a telepresence robot: Design and experimental study. In *ROS*.

Tiberio, L., Cesta, A., Cortellessa, G., Padua, L., and Pellegrino, A. R. (2012). Assessing affective response of older users to a telepresence robot using a combination of psychophysiological measures. In *2012 IEEE RO-MAN: The 21st IEEE International Symposium on Robot and Human Interactive Communication*, pages 833–838.

Tsai, C. C., Huang, H. C., and Lin, S. C. (2010). Adaptive neural network control of a self-balancing two-wheeled scooter. *IEEE Transactions on Industrial Electronics*, 57(4):1420–1428.

Vaughan, J., Kratz, S., and Kimber, D. (2016). Look where you're going: Visual interfaces for robot teleoperation. In *Int. Symp. on Robot and Human Interactive Communication (RO-MAN).*