Robot Simulation in Mixed Reality

Dominik Schulte Giuliano Albanese Carlo Retali ETH Zurich

Abstract

A new Mixed Reality (MR) application for the seamless interaction and control of a simulated mobile robot and a manipulator was created by leveraging the possibilities offered by ROS#. This library provides a communication layer between ROS (Robot Operating System) and .NET applications. The user control is enhanced by the use of an intuitive MR interface where path and manipulation planning can be freely used. The system performance was also tested on potential users alongside the perceived workload of this application. The NASA Task Load Index and the System Usability Scale were used to describe the quality of the system and both robots achieved decent scores in both of the questionnaires. The results of the study have shown that the application is easy and intuitive, but lag makes it difficult to operate successfully.

1. Introduction

Development and Research of Robotics can be assisted with the use of Mixed Reality (MR), where instead of having to build a prototype each time, the testing can be done in a virtual environment. MR describes the interaction between physical and virtual data. So far, it was mostly used to display additional information of the environment to the user. This can also increase safety, as the virtual interaction does not endanger humans. This is beneficial for training workers on a simulation of the real robot while displaying additional information [4, 2, 11].

There are many possibilities to control a robot in MR, such as using a controller. However, to decrease the amount of hardware the Microsoft HoloLens 1 can be used instead [5]. With this, the user can control the robot with gestures, leaving him without a controller and therefore without limitations in his work.

In their recent work, Ostanin et al. [9] presented an interactive control in MR. Their approach was to use the KUKA iiwa manipulator and the PLATO mobile platform and control the robot with virtual inputs. This control can be used for robotic manipulators and mobile platforms.

The goal of this work is to introduce a mixed-reality based controller to interact with a manipulator and a moving platform in an intuitive way without the use of physical robots. This will be achieved by developing a virtual interface on the Microsoft HoloLens 1 with which the robots can be controlled. Additionally, some safety related information is displayed, such as the path the robot is going to take. This application can then be used for training with virtual robots as well as controlling real robots, for example in a construction environment.

2. Architecture

The system architecture (Figure 1) was created for this project and can be divided in three interconnected blocks, namely ROS (Robot Operating System) [10], HoloLens, and the User. The user can interact with the simulated world through a MR device, in this case a Microsoft HoloLens 1. In the diagram, the green dashed boxes represent the type of robot for which the enclosed components are used, i.e. either a mobile robot or a robotic manipulator. The synthetic sensor data generated by the physics simulation of the robots, which may take place either in Unity [13] or Gazebo [8] (a robotic simulator), is processed by a collection of ROS nodes. To send and receive data via the nodes, the server and client subscribe to a set of topics. These topics are shown in the architecture on the connecting lines and always start with a "/". The MR app keeps track of the user inputs and translates them into commands for the ROS nodes, while Unity scripts are used to control the simulated robots. Information about the robot status and its current task, such as the planned navigation path of a mobile robot, can also be visualized in the app.

The communication between the Unity MR app and ROS is handled by ROS# [12], an open source software library in C#. The details of each part of the architecture are discussed in the following sections.

2.1. Hardware

As mentioned before, the MR device used for this project was a Microsoft HoloLens 1. Most of the implemented functionalities heavily rely upon its visual sensors and surface reconstruction capabilities. The HoloLens 1 is also

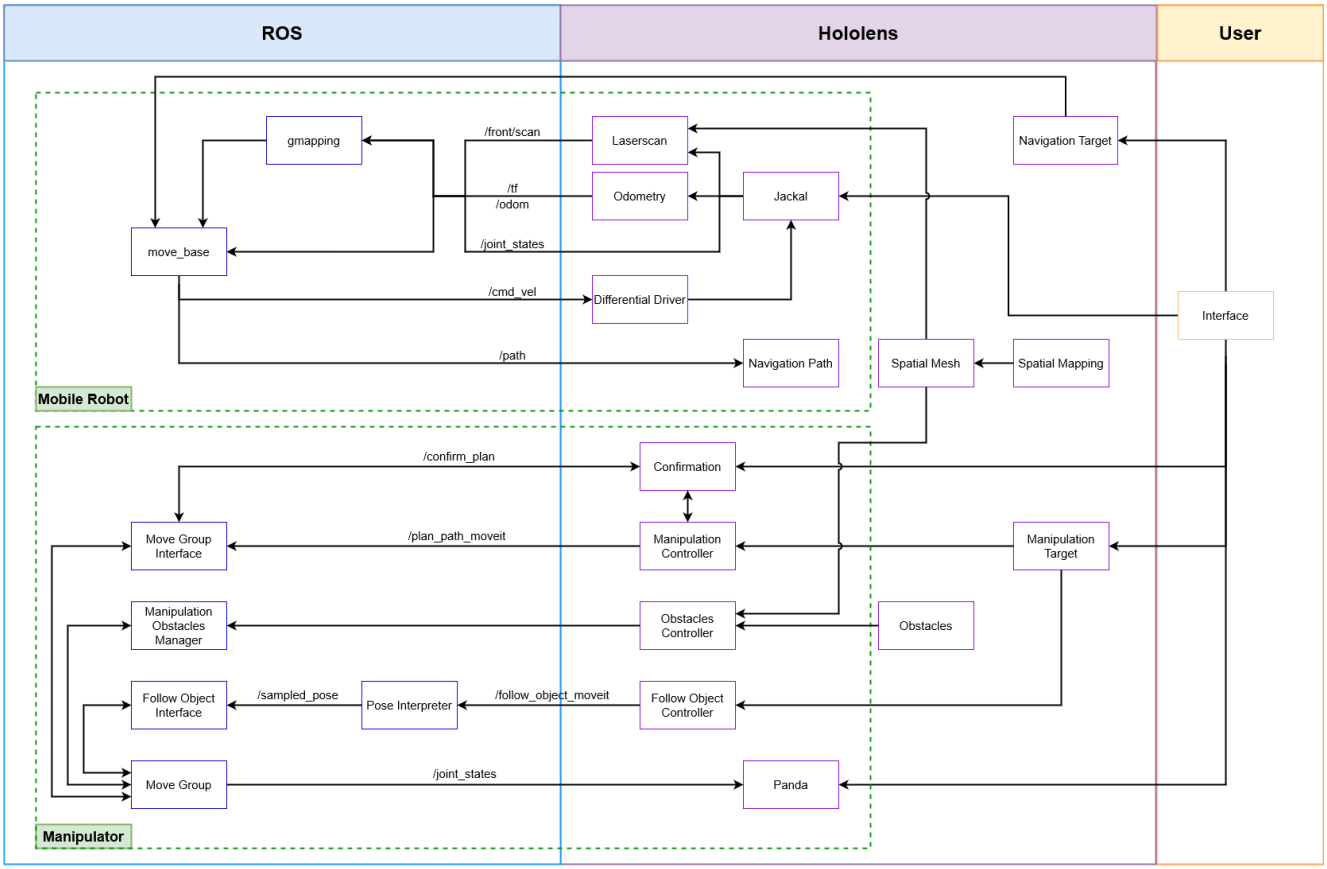

Figure 1. Overview of the System Architecture

used to display the simulation to the user as well as reading the user input.

2.2. Mixed Reality interface

The MR interface was developed in Unity with the packages ROS# and MRTK (Mixed Reality Toolkit) [6].
When the application is started, the HoloLens creates a spatial mesh and sets its current position as the origin of the virtual space. This spatial mesh is a detailed representation of the real-world surfaces around the HoloLens in the form of a surface mesh with which other virtual objects can interact. In order to improve the feeling of immersion for the user, this mesh is used to hide holograms that are behind a real life object.
The choice in robots was mainly based on the availability of complete packages and compatibility with ROS and Unity.
To test the navigation-related parts of our system, the Jackal mobile robot was used, whose physics simulation takes place in Unity. The spatial mesh only gets built once the application is started, so if the floor is not mapped in time, the robot might fall through on startup. To counteract this effect, the Jackal robot is placed on a small virtual platform (Figure 2). The user can then grab the robot and place it on the floor after waiting a few seconds. Then, the scripts to send the messages are started by pressing the ”Start Connection” button and the app begins to send odometry data to the gmapping and move_base node. Once the robot has been put on the floor, the user may not move it anymore, as the robot can not measure this movement with the odometry, which in turn would make the map generated by the navigation no longer usable.
The Panda robot, which is used for manipulation tasks, is not influenced by gravity and is not equipped with depth sensors, thus the robot can currently be placed anywhere in space without colliding with any walls or falling towards the floor.
To control the robots and start the connection to ROS, a set of buttons is used (Figure 3), which follows the user in his peripheral view. Not all of the buttons are visible at all times: a button is shown only when the corresponding action is available.
With the touch of the ”Set Navigation Target” button the navigation targeting mode is activated. The user can look around and place the target for the navigation wherever he wants with a so-called air-tap. The navigation target is only sent to ROS via ROS# upon pressing the ”Navigation Exe-

cution" button, such that the user can adjust the position of the target before the movement is actually started. This also reduces the computational load in computing the path due to less changes in navigation targets.
The same principle is used for manipulation. The target end-effector position can be placed anywhere in space. The press of button starts the calculation in ROS to check if a path exists to the target. If not, a red text is displayed in the middle of the users view. If a path is available, the movement can be executed. Furthermore, if the manipulator is in "Follow" mode, the user can enable continuous tracking with a button. In this case, the visual marker representing the target pose will be continuously followed by the manipulator.

Figure 2. Jackal Startup Platform

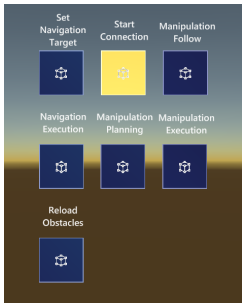

Figure 3. Mixed Reality Application Buttons

2.3. Navigation

A MR-simulated mobile robot is only able to "perceive" the obstacles in its simulation environment. However, a MR-simulated robot which aims to interact with a user in a realistic way has to be aware of its surrounding physical environment. In order to achieve this, the Hololens spatial mapping function was leveraged. The spatial mapping mesh provides a proxy for a simulated robot to "sense" real-world obstacles.
The mobile robot we used in our implementation, the Jackal robot, is equipped with a simulated 2D LIDAR, which continuously scans the environment and generates 2D point clouds. The point clouds and the fake odometry from Unity are received by the gmapping node in ROS, which uses a Rao-Blackwellized particle filter [3] to build and continuously update a 2D grid map of the surroundings.
Navigation commands issued by the user through the MR interface are sent via the corresponding topic to an ad-hoc action server in the move_base node in the form of target positions for the robot to reach. The map and the odometry data are then used by the planner within move_base to find the shortest viable path to the destination. If such a path exists, the planner starts sending a sequence of velocity commands to the wheels via the differential controller ("Differential Driver" in Figure 1). As the robot moves, the planned path is visualized in the MR app to keep the user aware of the robot's intentions.

2.4. Manipulation

In our MR app, the user can interact with a simulated robotic manipulator (the Panda robot) and move it around the virtual environment. The interface allows the user to position a visual marker indicating the position the end-effector should reach. New manipulation goals are sent upon pressing the "Manipulation Planning" button through ROS# to the MoveGroupInterface node, which forwards them to MoveGroup. This is a ROS node which provides high level functions to send commands to Moveit, the ROS package which handles manipulation tasks. The inverse kinematics solver in Moveit will then verify if a feasible path exists. If such a path exists, the user can confirm the command with the button "Manipulation Execution" and the manipulator will execute it. If the specified goal pose is not within the reachable space of the robot, the inverse kinematics solver will fail to find a viable path and an error message will be displayed in the MR app to notify the user.
Robotic manipulators are often required to work in a constrained environment, especially in collaborative tasks, and thus need to be aware of the obstacles in their workspace. MR devices could help achieve this kind of awareness and add constraints to the motion of the manipulator without additional sensors. To study how that could be done in practice, a simple obstacle avoidance feature was added to the manipulation stack. By pressing the button "Reload Obstacles" in the MR interface, all the (virtual) obstacles added to the scene – in our demo, those in red – will be registered as collision objects; the IK solver in moveit will then take them into account during motion planning. The Spatial Mapping mesh generated by the Hololens can also be registered in the same way. During testing, however, we found that the size of the mesh would slow down the solver significantly, so this feature was disabled in the demo. One possible solution to this problem could be to reduce the Spatial Mapping mesh to a simpler geometry to facilitate its loading.
A further implementation has been created, where the user can choose to test and interact with the manipulator in a more dynamic way by selecting the "Follow" mode. In this case, the user can place and drag the marker around the virtual environment and, as long as the goal position remains within reach, the manipulator will continuously follow it, without restricting the interaction to further actions on the user interface, as before. The performance in this case is limited by the time required by the inverse kinematics solver to assess the path existence and by sampling the incoming set of positions.

3. User Study Methodology

A user study was conducted to evaluate the application. A total of 13 subjects were tested, 9 male and 4 female subjects, aged between 22 and 27. The user study consisted of two tasks, to control the mobile and the manipulation robot with a randomized order to eliminate the learning effects and fatigue.

For the mobile robot task, the subject was first instructed to take the robot off the platform and place it on the floor, close to one corner of the room. After starting the odometry perception pipeline, the subject was then tasked to make the robot move to another corner in the room by placing a target and submitting it to the robot. The supervisor observed the interaction with the system.

For the manipulation robot task, the subject was tasked to place the robot on the floor in the middle of the room and perform three different manipulations with the robot, by placing the target in reach of it. After each manipulation the robot and/or the target could be moved.

After each task, the NASA Task Load Index (TLX) questionnaire [7] had to be filled out. At the end of both tasks, the System Usability Scale (SUS) [14] questionnaire had to be filled out by the subjects as well.

The NASA TLX was created for operators working with different human-machine interface systems and measures the subjective workload. The six elements of the questionnaire are compared to each other by the subjects in order to decide the importance of them. This is done with 15 comparisons where the subject decides which of the two elements is more important to him in each instance. The score for each of the six elements is multiplied with how often each element got chosen in the comparisons. All these scores are then added to result in a value between 1 and 20.

The System Usability Scale (SUS) Plus is a scale used to evaluate a wide variety of hard and software. It consists of ten statements, half of them are positive and half are negative. The subject can rate any statement with a five-point Likert Scale [1], with which the user can "Strongly Agree" or "Strongly Disagree" with the respective steps in between for each of the ten statements. The score is calculated by subtracting 1 off the score for each odd statement, subtracting the score from 5 for each even statement and then adding them all up. By multiplying the score by 2.5, a percentage is calculated which can be used to describe the quality of the system (Table 1).

SUS Score	Adjective Rating
$80.3 <$	Excellent
$68 - 80.3$	Good
68	OK
$51 - 67$	Poor
< 51	Awful

Table 1. SUS Scoring Scale

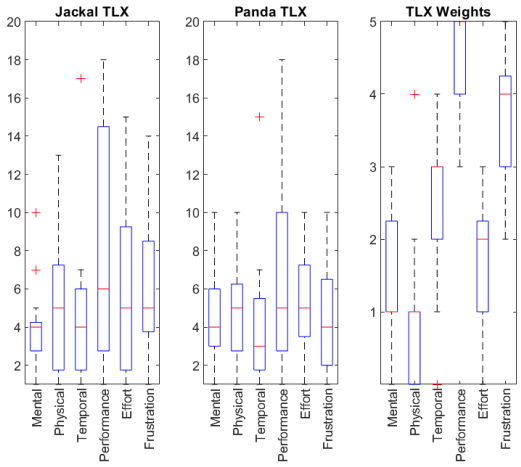

Figure 4. TLX Scores

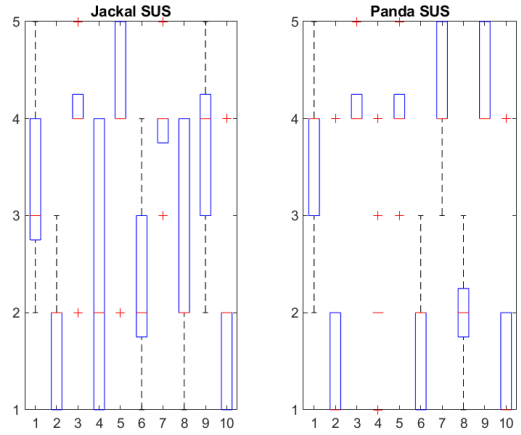

Figure 5. SUS Scores

4. User Study Results

The TLX questionnaire results (Figure 4) are very similar for both robots. *Mental* and *temporal* demand have the lowest median score, while the subjects were most dissatisfied with their *performance*. *Physical* demand was about the same as the *effort* and *frustration* for both tasks as well. The most important property is the *performance*, followed by *frustration* and *temporal* demand. *Effort* is rated only slightly more important than *mental* and *physical* demand. The average TLX score from the questionnaire was 6.36 for the Jackal and 5.52 for the Panda robot.

The results for the SUS questionnaire (Figure 5) for the two robots differ a bit more. The same general tendency can be observed for both as the odd statements received a higher score than the even statements. However, the standard de-

viation for the scores is a lot bigger for the Jackal robot. The average SUS score for Jackal is 71.54 and 79.23 for the Panda.

5. Challenges

Different challenges came up for each part of the system. Setting up the environments for the HoloLens and the ROS/Unity duality throughout ROS#, proved difficulties due to outdated tutorials and many compatibility issues with MRTK and ROS. Moreover, hurdles inherited from using VMs on computationally limited and not fully compatible hardware accounted for a lot of amount of time. Some of these compatibility issues were not obvious at first, either the app was not able to start on the HoloLens after deploying or Unity was not able to display settings for different elements, such as the buttons, due to a system reflection error. A big issue came from the Windows Subsystem for Linux (WSL), as it caused driver issues and compiler errors, preventing the connection between ROS and the MR application. WSL was initially chosen because it is comfortable to use, considering the hardware in use. As it was not possible to solve these problems, other approaches either used a native Linux machine or a Linux VM on a Windows Bootcamp partition were then used. However, in the last case further incompatibilities prevented an implementation (drivers, compiler errors, working memory and storage limitation).
Further, setting up ROS and finding robots that were fully compatible with all the packages provided different challenges, from installation problems to missing packages and incompatibilities.
ROS# also proved to be challenging, due to a bug in the newest version of the library, which changed the contents of the sent messages to all zeros without displaying any error. This again prevented the connection to be able to send and receive the correct data.
Finally, there is still an issue with deploying the application on the HoloLens, as the app does not start, probably due to a further incompatibility.
There were also a lot of problems during the user study. The application had to be restarted many times due to different errors or connection issues. As the application was not deployed on the HoloLens, there were a lot of connection issues, resulting in lag or disconnecting. On some occasions not all of the topics subscribed correctly, which was only fixed by a complete restart of the whole system. In almost every experiment Unity crashed at least once as well.

6. Discussion

The user study was developed with the main goal of understanding the impact that such applications have on the user, regarding the usability and workload. This is of critical importance in MR since the user needs to be able to address the real time world and its stimulus while also interacting with the additional interface. This is particularly true in the construction field, where MR is a further tool that workers can use to enhance their performance. In this field it is very important not to get distracted or frustrated, as distraction can be very dangerous while frustration only means that the device is not used.
According to the study results, the Panda robot and its manipulation task has been perceived as more usable by most of the users. In particular, this is shown by the SUS metric (Figure 5), that as previously stated is higher for the Panda robot, while still being above average for the Jackal, rating both robots as "good". Panda also received a better evaluation when considering the perceived workload, as computed in the TLX score from the NASA-TLX assessment tool.
Another interesting insight from the user study comes from the TLX weights as shown in figure 4. Here, users gave the higher weighting to their *performance* among the possible six components of the NASA-TLX. This could have a root cause in some technical problems that came up during the user study process. However, as *frustration* was rated as the second most important, these technical problems did not seem to frustrate the subjects too much, even though it had a big impact on the performance. However, this could also be due to the experiment being very short.
Particularly, one user had the issue of his hands not being recognized by the HoloLens at all. Moreover, some network issues caused the connection to fail, which led to the application having to be restarted, consequently affecting the scores. However, subjects generally experienced more errors for Jackal than Panda. This is due to the different way the systems work, since Jackal relies on the map construction using the spatial mesh that is prone to change over time, causing the robot to misinterpret the built map.
The SUS results also reveal another issue that probably influenced the scores: As none of the subjects work in the construction business, they were not sure if they would use this system frequently. For the jackal robot, the scores for question 4, 6 and 8 were noticeably worse, meaning subjects might need help in utilizing the application as it is now by a technical person (question 4), thought the system was too inconsistent (question 6) and found the system cumbersome to use (question 8).

7. Future Work

In future work, the stability of the application needs to be improved first. The connection issues can be reduced by deploying the application onto the HoloLens. Currently, two different ROS connectors are used, as the joint_states topic is used for both robots. Creating a unique topic for each robot then would allow the operation of both robots at the same time.

Another interesting future improvement could be to not only run simulated robots, but work with real ones, this would add an extra degree of complexity. Moreover, this working pipeline could be adapted for further educational scopes with some changes.

8. Conclusion

This project provides a mixed-reality application for the virtual interaction with two different types of robots, a mobile robot, Jackal, and a manipulator, Panda, through a set of intuitive steps and commands. The application has been developed for the Microsoft HoloLens. Given the safety context of this work, the visualization of the navigation path has also been added in order to deploy this application for workers in the construction field in the future. The architecture is composed of three main building blocks, whereas two are required in order to simulate the robots in ROS, and visualize and interact with them in Unity. The communication between ROS and Unity is enhanced by ROS#, while several scripts and programs make sure that all visualized movements, information and robots behaviours are coherent. Further, a user study was carried out in order to test if the working principles were clear to a possible user, and to assess the workload that such an approach imposes, since this is of crucial importance in Mixed-Reality. The study results show that the application is intuitive and perceived clearly by most of the users. Overall, the manipulation received higher score than the navigation. However, some technical problems took place, this could have potentially changed the user perception towards the applications.

9. Contributions

Work packages	Giuliano	Dominik	Carlo
Navigation	100%	0%	0%
Manipulation	70%	0%	30%
Interface	0%	100%	0%
Integration	60%	20%	20%
Further Features	0%	0%	100%
User Study	10%	55%	35%
Presentations	10%	65%	25%
Report	33%	33%	33%

Table 2. Contributions to the final code in percentage

The contribution percentages were defined according to the amount of existing code in the final product. The main issue in assigning the percentages was that the Manipulation work package was improved and developed further by Giuliano. Carlo spent a lot of time on it as well, however there were a lot of problems with setting up the environment. The work package "Further Features" consists of the "Follow" capability of the manipulator.

References

[1] G. Albaum. The likert scale revisited. *Market Research Society. Journal.*, 39(2):1–21, 1997. 4

[2] C. Coutrix and L. Nigay. Mixed reality: a model of mixed interaction. In *Proceedings of the working conference on Advanced visual interfaces*, pages 43–50, 2006. 1

[3] G. Grisettiyz, C. Stachniss, and W. Burgard. Improving grid-based slam with rao-blackwellized particle filters by adaptive proposals and selective resampling. In *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, pages 2432–2437, 2005. 3

[4] W. Hönig, C. Milanes, L. Scaria, T. Phan, M. Bolas, and N. Ayanian. Mixed reality for robotics. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5382–5387. IEEE, 2015. 1

[5] Microsoft. HoloLens (1st gen) Hardware. Available at https://docs.microsoft.com/en-us/hololens/hololens1-hardware. 1

[6] Microsoft. Mixed Reality Toolkit for Unity. Available at https://microsoft.github.io/MixedRealityToolkit-Unity/. 2

[7] NASA. Task Load Index. Available at https://humansystems.arc.nasa.gov/groups/TLX/. 4

[8] Open Source Robotics Foundation. Gazebo - Robot Simulation Made Easy. Available at http://gazebosim.org/. 1

[9] M. Ostanin, R. Yagfarov, and A. Klimchik. Interactive robots control using mixed reality. *IFAC-PapersOnLine*, 52(13):695–700, 2019. 1

[10] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng. Ros: an open-source robot operating system. In *ICRA workshop on open source software*, volume 3, page 5. Kobe, Japan, 2009. 1

[11] Á. Segura, A. Moreno, G. Brunetti, and T. Henn. Interaction and ergonomics issues in the development of a mixed reality construction machinery simulator for safety training. In *International Conference on Ergonomics and Health Aspects of Work with Computers*, pages 290–299. Springer, 2007. 1

[12] Siemens. ROS#. Available at https://github.com/siemens/ros-sharp. 1

[13] Unity Technologies. Unity. Available at https://unity.com/. 1

[14] usabilitiTest. System Usability Scale (SUS) Plus. Available at https://www.usabilitest.com/system-usability-scale. 4