AdaptiX – A Transitional XR Framework for Development and Evaluation of Shared Control Applications in Assistive Robotics

MAX PASCHER, TU Dortmund University, Germany and University of Duisburg-Essen, Germany
FELIX FERDINAND GOLDAU, German Research Center for Artificial Intelligence (DFKI), Germany
KIRILL KRONHARDT, TU Dortmund University, Germany
UDO FRESE, German Research Center for Artificial Intelligence (DFKI), Germany
JENS GERKEN, TU Dortmund University, Germany

(a) (b) (c)

Fig. 1. Setup with (**a**) a user's view in the Virtual Reality (VR) simulation environment, (**b**) an over-the-shoulder image of interaction with a physical robot, and (**c**) a combined view of physical robot and visual cues in Mixed Reality (MR).

With the ongoing efforts to empower people with mobility impairments and the increase in technological acceptance by the general public, assistive technologies, such as collaborative robotic arms, are gaining popularity. Yet, their widespread success is limited by usability issues, specifically the disparity between user input and software control along the autonomy continuum. To address this, shared control concepts provide opportunities to combine the targeted increase of user autonomy with a certain level of computer assistance. This paper presents the free and open-source *AdaptiX* XR framework for developing and evaluating shared control applications in a high-resolution simulation environment. The initial framework consists of a simulated robotic arm with an example scenario in Virtual Reality (VR), multiple standard control interfaces, and a specialized recording/replay system. *AdaptiX* can easily be extended for specific research needs, allowing Human-Robot Interaction (HRI) researchers to rapidly design and test novel interaction methods, intervention strategies, and multi-modal feedback techniques, without requiring an actual physical robotic arm during the early phases of ideation, prototyping, and evaluation. Also, a Robot Operating System (ROS) integration enables the controlling of a real robotic arm in a *PhysicalTwin* approach without any simulation-reality gap. Here, we review the capabilities and limitations of *AdaptiX* in detail and present three bodies of research based on the framework. *AdaptiX* can be accessed at https://adaptix.robot-research.de.

CCS Concepts: • **Computer systems organization** → **Robotic control**; • **Human-centered computing** → *Visualization techniques*; *Virtual reality*.

EICS '24, June 24–28, 2024, Cagliari, Italy

ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/XXXXXXX.XXXXXXX

Additional Key Words and Phrases: assistive robotics, human–robot interaction, shared user control, augmented reality, virtual reality, mixed reality, visual cues

ACM Reference Format:
Max Pascher, Felix Ferdinand Goldau, Kirill Kronhardt, Udo Frese, and Jens Gerken. 2024. AdaptiX – A Transitional XR Framework for Development and Evaluation of Shared Control Applications in Assistive Robotics. In *EICS '24: The 16th ACM SIGCHI Symposium on Engineering Interactive Computing Systems, June 24–28, 2024, Cagliari, Italy.* ACM, New York, NY, USA, 28 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

Robotic arms as assistive technologies are a powerful tool to increase self-sufficiency in people with limited mobility [32, 42], as they facilitate the performance of Activities of Daily Living (ADLs) – usually involving grasping and manipulating objects in their environment – without human assistance [47]. However, a frequent point of contention is the assistive robot's autonomy level. The reduction of user interaction to just oversight with purely autonomous systems elicits stress [48] and feelings of distrust in their users [64]. On the other side of the autonomy spectrum, manual controls can be challenging - or even impossible - to operate, depending on the significance and type of impairment. Shared control – a combination of manual user control through standard input devices plus algorithmic support through computer software adjusting the resulting motion – may have the potential to mitigate both concerns [1]. Here, both the user and the robot share a task on the operational level, enabling people with motor impairments to get involved in their assistance. As a result, such approaches can increase the feeling of independence while improving ease of use compared to manual controls [16].

A characteristic real-world scenario, motivated by our research, has an assistive robotic arm (e.g., a Kinova Jaco 2) attached to a wheelchair to support the user in ADLs. Here, the user is challenged with operating six or more Degrees-of-Freedom (DoFs), which requires complex input devices or time-consuming and confusing mode switches. This potentially results in increased task completion time and user frustration [20]. Addressing this, shared control systems can facilitate more straightforward and accessible robot operation. However, they may require well-designed communication of robot (motion) intent, so that the user retains awareness and understands the level of support they get from the system [43]. Also, different users might need distinct input devices or require multi-modal input to account for varying abilities.

Based on our experiences, we identified several challenges that currently influence and potentially impede the effective development of shared control approaches:

- Shared control systems for assistive technologies still pose open questions requiring considerable experimentation, tweaking and balancing between user and robot interaction [33].
- While much research explored robot motion intent, there is little insight into what works best in which situation and for which type of user. In assistive robotics, the visualization and feedback modality must be carefully adapted to the user's needs and abilities as there is no "one size fits all" solution [22].
- Similarly, suitable input devices may vary between users. Depending on individual preferences and capabilities, multi-modal input or the choice between different input modalities may be required [2].
- Bringing robots and humans physically together during research studies is difficult due to the laborious and costly transportation, safety concerns with robots and general availability of the user group.

Contribution. To allow researchers, designers and developers to address these challenges holistically and flexibly, we present *AdaptiX* – a free, open-source XR framework [1]. Aimed at Design and Development (D&D), AdaptiX combines a physical robot implementation with a 3D simulation environment. The simulation approach (analogous to simulations in industrial settings [36, 41, 56]) mitigates the assistive robotic arm's bulky, expensive, and complex nature. It also makes the integration of visualization feedback or different input modalities easier to explore and test, while a Robot Operating System (ROS) interface allows the direct transfer to the real robot. Testing new interaction and control options becomes much less time-consuming while simultaneously excluding potentially dangerous close-contact situations with users before glitches are managed [42]. In total, the framework facilitates the development and evaluation of assistive robot control applications *in-silico* and creates a practical and effective step between ideation, development, and evaluation, allowing HRI researchers more flexibility and facilitating efficient resource usage.

To summarize, the *AdaptiX* framework contributes the following:

- *AdaptiX* allows researchers to rapidly design and test novel visualization and interaction methods.
- The framework integrates an initial concept and implementation of a shared control approach.
- The integrated ROS interface facilitates connection to a non-simulated – physical – robotic arm to perform bidirectional interactions and data.
- The framework's concept enables a code-less trajectory programming by hand-guiding the simulated or physical assistive robotic arm to the specific location and saving the position and orientation of the Tool Center Point (TCP).
- Recording TCP data enables replaying user-controlled robot movements and results in a fully customizable system. Options include changing specific details during replaying, such as repositioning cameras or re-rendering background scenes.
- Finally, the entire continuum of Mixed Reality (MR) can be exploited in the *AdaptiX* environment. This allows applications in Virtual Reality (VR), pure screen space, Augmented Reality (AR), simultaneous simulation and reality, and pure reality (cf. the *virtuality continuum* of Milgram and Kishino [40]).

2 RELATED WORK

While robotic arms are a particularly useful and versatile subset of assistive technologies, their widespread success is limited by a number of design challenges concerning the interaction with their human user. In recent years, a growing body of research addressed these concerns and associated optimization options to increase their usability, e.g., [11, 19, 33]. During the *AdaptiX* development process, we aimed to include functionality to address the challenges of shared control optimization[18], intent communication[43], and attention guidance[45].

2.1 Shared Control for Assistive Robots

Current shared control systems operate along an autonomy continuum, respectively balancing user input and system adjustments. At one extreme, the systems tend to be heavily manual, with only minor adjustments to the user's input [53]. On the other end are systems where users primarily provide high-level commands for the robot to execute [57]. A number of different approaches – including time-optimal [20] and blended mode switching [15], shared-control-templates [49] and body-machine-interfaces [28] – are currently employed in various settings.

[1] *AdaptiX* framework. https://adaptix.robot-research.de, last retrieved October 19, 2023.

A fundamentally different approach is the shared control system proposed by Goldau and Frese [18]. Their concept combines a robotic arm's cardinal DoFs according to the current situation and maps them to a low-DoF input device. The mapping is accomplished by attaching a camera to the robotic arm's gripper and training a Convolutional Neural Network (CNN) by having people without motor impairments perform ADLs [18] – similar to the learning-by-demonstration approach for autonomous robots by Canal et al. [6]. The CNN returns a set of newly mapped DoFs, ranked by their assumed likeliness based on the CNN for the given situation, allowing users to access a variety of movements for each situation. In addition, the CNN-based approach allows the system to be easily extendable as the same system can be trained to discriminate between many different situations – making it a viable concept for day-to-day use. Goldau and Frese [18] conducted a proof-of-concept study comparing the control of a simulated 2D robot with manual or CNN-based controls. Task execution was faster with their proposed concept; however, users experienced it as more complex than manual controls [18].

Our framework *AdaptiX* is influenced by Goldau and Frese's approach, but extends it from 2D to 3D space. This increases the number of possible DoFs, which allows for an accurate representation of ADLs in the framework. By adding functionality, visualizations, and a ROS integration, *AdaptiX* can be used to develop and evaluate novel interaction control methods based on this approach for shared control, which we refer to as *Adaptive DoF Mapping Control (ADMC).*

2.2 Robot Motion Intent

Regardless of the specific interaction details, it is necessary to effectively communicate the intended assistance provided by the (semi-)autonomous system [4]. Clear communication between robots and humans enhances the shared control system's predictability, avoids accidents, and increases user acceptance.

A crucial element of the D&D process of robotic devices is, therefore, the testing of intent communication methods. *Choreobot* – an interactive, online, and visual dashboard – proposed by van Deurzen et al. [58] supports researchers and developers to identify where and when adding intelligibility to the interface design of a robotic system improves the predictability, trust, safety, usability, and acceptance. Moreover, Pascher et al. [43] provide an extensive overview of the various types of visualization and modalities frequently used in communicating robot motion intent. These range from auditory [9] and haptic [8] modalities to anthropomorphizing the robot and using its gaze [37] or gestures [17]. Their findings are substantiated by Holthaus et al. [23], who used an ethnographic approach to derive a comprehensive communication typology.

While all these intent communication modalities are viable, visual representations of future movements are often quoted as less workload-intense for the end-user [12]. AR is, therefore, unsurprisingly a frequently used tool to convey detailed motion intent [7, 21, 50, 60, 62], allowing interactions to become more intuitive and natural to humans [35]. Suzuki et al. emphasize the benefits of AR-based visualizations for communicating movement trajectories or the internal state of the robot [55].

The visual feedback employed by *AdaptiX* mimics AR in a VR environment with directional cues registered in 3D space. This approach allows the user to understand different movement directions for the actual control and the suggested DoF combinations. To streamline understanding the control methods, one of our primary approaches is the usage of arrows – a straightforward and common visualization technique to communicate motion intent[51, 52, 60].

2.3 Feedback Modalities for User Attention Guidance

When creating systems using shared control, it is crucial to guide the user's focus to the assistance the robot is offering [46]. This guidance is particularly important if either party is moving the

robot in a way that could lead to collisions or worsen the situation. To enhance the predictability of shared control systems, various feedback modalities have been proposed to guide user attention as a secondary feedback mechanism to AR. The goal is to provide a feedback solution that results in short reaction times, enabling users to quickly direct their focus to the information provided by the robot.

In the related discipline of autonomous driving systems, if the vehicle encounters a situation it was not programmed or trained to handle, it will issue a Take-Over-Request (TOR). This TOR prompts the driver to take manual control of the vehicle to prevent a collision or to drive in areas the vehicle cannot handle autonomously.

Auditory, visual, and tactile/haptic modalities are commonly used for TORs [61] – either as a single sensory input [46] or a combination of multiple variants [45]. Simulation studies, along with research on reaction times to different sensory stimuli, indicate that multi-modal feedback results in the lowest possible reaction times in shared control systems [5, 13, 30].

Implementing these feedback methods into existing assistive robot systems would be straightforward as the necessary output devices – like screens, speakers, or vibration motors – are commonly already present. To allow researchers to evaluate the benefits of the different modalities, *AdaptiX* includes three modes for attention guiding: visual, auditory, and tactile/haptic. Developers can either choose one modality or follow a multi-modal approach.

3 FRAMEWORK CONCEPT

The *AdaptiX* XR framework facilitates the development and evaluation of HRI shared control applications in an easy-to-use, high-resolution transitional MR environment. Equipped with a VR simulation environment containing a virtual *Kinova Jaco 2* and ample customization options, researchers can streamline their D&D process while simultaneously reducing overhead and boosting efficiency. Figure 2 provides an overview of the framework's architecture.

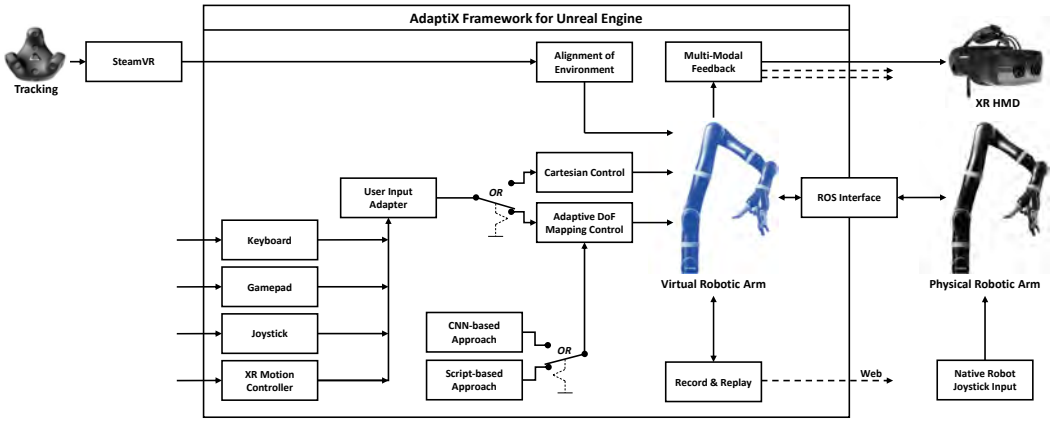


Fig. 2. Overview of *AdaptiX*' architecture, illustrating each component, their directional communication, and the crossover from and to the framework. The user input is either used for *Cartesian Control* or Adaptive DoF Mapping Control (ADMC). For ADMC, either a CNN-based or script-based rule engine can be selected.

In addition to an Cartesian robot control, we propose ADMC as an initial shared control approach, using suggestions by a rule engine (e.g., a CNN or script-based approach) to be controlled by the user. ADMCs are implemented directly into the *Unreal Engine* to enable researchers and developers

to fully customize the control methods, systems behavior, and feedback techniques by coding in *C++* or *Blueprints*.

AdaptiX supports several pre-implemented input devices and provides an adapter class for an easy development and implementation of further input devices. This supports researchers and developers to easy implement their ideas and concepts. The integrated ROS interface facilitates connection to a non-simulated – physical – robotic arm to perform bidirectional interactions and data exchange in a *DigitalTwin* and *PhysicalTwin* approach.

AdaptiX enables effortless trajectory programming by manually guiding the TCP of a simulated or physical robotic arm to a desired location and recording its position and orientation. Recorded data of user-controlled robot movements can be replayed. Offering the adjustment of specific details, such as camera positions and background scenes, results in a highly customizable system.

The aim is to provide a modular and extensible framework so that research teams do not need to start from scratch when implementing their shared control applications.

3.1 Adaptive DoF Mapping Control (ADMC)

For the adaptive DoF mapping – referred to as *ADMC* – of the robotic arm, the goal is to present a set of DoF mappings ordered based on their effectiveness in accomplishing the pick-and-place task used in the experiment. The concept of "usefulness" assumes that maximizing the cardinal DoFs of the robot assigned to an input-DoF while progressing towards the next goal is the most advantageous option.

This DoF mapping, referred to as the *optimal suggestion*, is assumed to be the best choice due to a significant reduction in the need for mode switches when multiple DoFs are combined into a single movement. The more DoFs are combined (assuming it is sensible for the given situation), the fewer mode switches are required. As a result, the DoF mappings are ordered based on the number of DoFs they combine.

In addition to the optimal suggestion, the second suggestion is a selection of an orthogonal variation of the first suggestion, which has the highest probability and most variation in spatial direction and keeps the number of combined DoFs unchanged. This secondary suggestion is likely useful to users as they can utilize it to adjust their position while maintaining a sensible orientation toward the next goal. The following DoF mappings were used (see Figure 3):

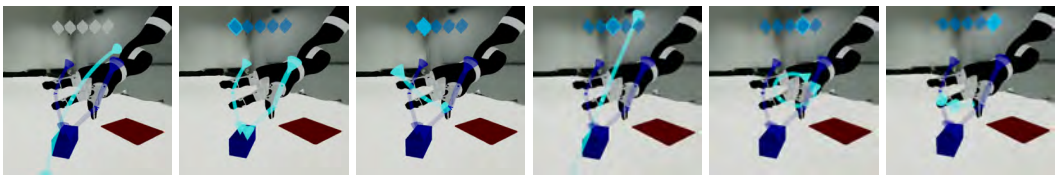


Fig. 3. Suggestions as Visualized in the ADMC, (**a**) Continue previous movement, (**b**) Optimal Suggestion, (**c**) Adjustment Suggestion, (**d**) Translation Suggestion, (**e**) Rotation Suggestion, (**f**) Gripper Suggestion. Colors: Bright cyan arrow: Currently active DoF mapping. Dark blue arrow: Next most likely DoF mapping.

(1) *Optimal Suggestion:* Combining translation, rotation, and finger movement [opening and closing] into one suggestion, causing the gripper to move towards the target, pick it up, or release it on the intended surface.

(2) *Adjustment Suggestion:* An orthogonal suggestion based on (1) but excluding the finger movement. Allows the users to adjust the gripper's position while still being correctly orientated.
(3) *Translation Suggestion:* A pure translation towards the next target, disregarding any rotation.
(4) *Rotation Suggestion:* A pure rotation towards the next target disregarding any translation.
(5) *Gripper Suggestion:* Opening or closing of the gripper's fingers.

3.1.1 CNN-based Approach. For the CNN approach, a color-and-depth camera is attached to the gripper of an assistive robotic arm. The live video feed is transmitted to a CNN, which is trained using data collected from non-impaired individuals performing ADLs using the robotic arm along with a high-DoF input device. The CNN does not need a model of the environment to provide these mappings. Principal Component Analysis (PCA) is employed to transform the CNN's output into a matrix $\hat{D}$, where each column represents a combination of cardinal DoFs along which the robotic arm can move.

Next, a subset of $\hat{D}$ is selected, containing as many columns as the number of DoFs provided by the input device. This selected subset is referred to as D, and it serves to map input-DoFs to output-DoFs. When an input-DoF is engaged, the robot's movements are determined by the values in the corresponding vector of D, which proportionally activate the robot's cardinal DoFs. A mode switch is defined as the exchange of D with a different subset of $\hat{D}$. This enables the system to switch between various mappings of input-DoFs to output-DoFs, adapting the robot's control according to the user's needs and preferences. A visual representation of this control pipeline is depicted in Figure 4a.

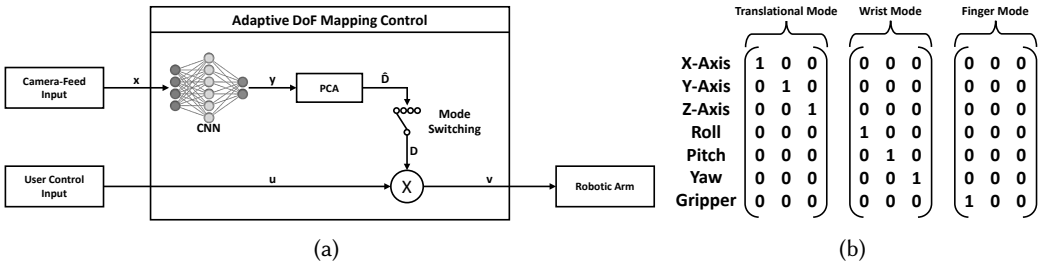


Fig. 4. Concept of adaptive DoF mapping control. **(a)** Control pipeline for proposed adaptive shared control and **(b)** matrix representation of DoF mappings: Columns represent input-DoFs. Rows represent output-DoFs. Subsets represent modes. Two empty columns were added to represent zero movement mappings in *Finger Mode*.

$\hat{D}$ is a square matrix with dimensions based on the number of cardinal DoFs available on the robot to be controlled. In the case of the *Kinova Jaco 2* [29], this results in a 7×7 matrix. This matrix represents a mapping of input-DoFs to output-DoFs when the number of DoFs in both cases is equal. The values in each column, ranging from -1 to 1, indicate the proportion with which the specific cardinal DoF is utilized when engaging the corresponding input-DoF.

By defining $\hat{D}$ as an identity matrix, each input-DoF is mapped to a single output-DoF. Selecting an equal number of columns from $\hat{D}$ to form matrix D allows for manual control with mode switching along cardinal DoFs. Moreover, this representation enables the combination of multiple cardinal movements into arbitrary output DoF mappings. For example, a (transposed) column of *(0.5, 0.5, 0, 0, 0, 0, 0)* would result in diagonal movement along the X- and Y-Axes of the robot. Such

combinations enable the offering of complex movements with different proportions depending on the situation, enhancing the control options available to users. The identity matrix for a *Kinova Jaco 2* with a 3-DoFs joystick is illustrated in Figure 4b.

3.1.2 Script-based Approach. As an alternative rule engine for our ADMC concept, we implemented a task-specific script. This approach eliminates potential biases that a more generic, but currently limited method like a CNN-based control might introduce. It is essential to note that our task-specific script is effective only in a controlled experimental environment.

The task-specific script assesses the end effector's current position, rotation, and finger position relative to a target, allowing it to adaptively calculate the matrix $\hat{D}$. This script recommends optimal movements to pick up an object and place it onto a target drop area, maximizing the combination of as many DoFs as possible. Additionally, it provides other DoF combinations that may be less beneficial to mimic the idea that each subsequent column in $\hat{D}$ has a decreasing likelihood of being useful. These additional DoF mappings are ordered by the number of combined DoFs in a decreasing manner.

To validate the effectiveness of this approach, we conducted pilot tests, comparing it to a *Wizard-of-Oz* method. In this scenario, a human "simulated a CNN" to explore user interaction with such a system.

3.1.3 Point of Time to Communicate the Suggestion. Our ADMC concept uses an adaptive DoF mapping system to recommend DoF mappings to the users depending on the current situation. The system visualizes the currently active DoF mapping as a bright cyan and the suggestion as a dark blue arrow (see Figure 3). This suggestion can be communicated – based on the the configuration – either continuously or only if the next most likely movement direction differs from the currently active DoF mapping by a certain threshold.

To calculate this threshold – the difference between the currently active and new most likely DoF mapping –, *cosine similarity* [54] is used, ranging from exact alignment [0%] to total opposite direction [100%]. The formula for cosine similarity of two n-dimensional vectors is defined as:

$$\text{cosine similarity} = \cos\left(\vec{a}, \vec{b}\right) = \frac{\vec{a}\vec{b}}{\|\vec{a}\|\|\vec{b}\|} = \frac{\sum_{i=1}^{n} a_i b_i}{\sqrt{\sum_{i=1}^{n} (a_i)^2}\sqrt{\sum_{i=1}^{n} (b_i)^2}} \tag{1}$$

To implement a difference value, the cosine similarity needs to be transformed. As a cosine similarity of -1 indicates completely opposed vectors, the difference value needs to return 1 – i.e. the maximum possible difference – for a cosine similarity value of -1. A cosine similarity of 1, indicating exact similarity, should return a difference value of 0 – i.e. no difference. Perpendicular vectors with cosine similarity 0 should return a difference value of 0.5 – i.e. a 50% difference. To calculate the difference value **d**, the following formula is used:

$$\text{difference d} = 1 - \frac{\cos\left(\vec{a}, \vec{b}\right) + 1}{2} \tag{2}$$

This difference value represents the difference between two vectors. While the user moves the robot with an active DoF mapping, the adaptive DoF mapping system reevaluates the situation and calculates new suggested DoF mappings. The default difference value is set to 0.2 (20% difference between currently active and new most likely DoF mapping).

3.2 Full Mixed Reality Continuum

In our framework, we created an environment in which the entire continuum of MR is exploitable. This extends the use of *AdaptiX* to new scenarios and environments – including the real world. The

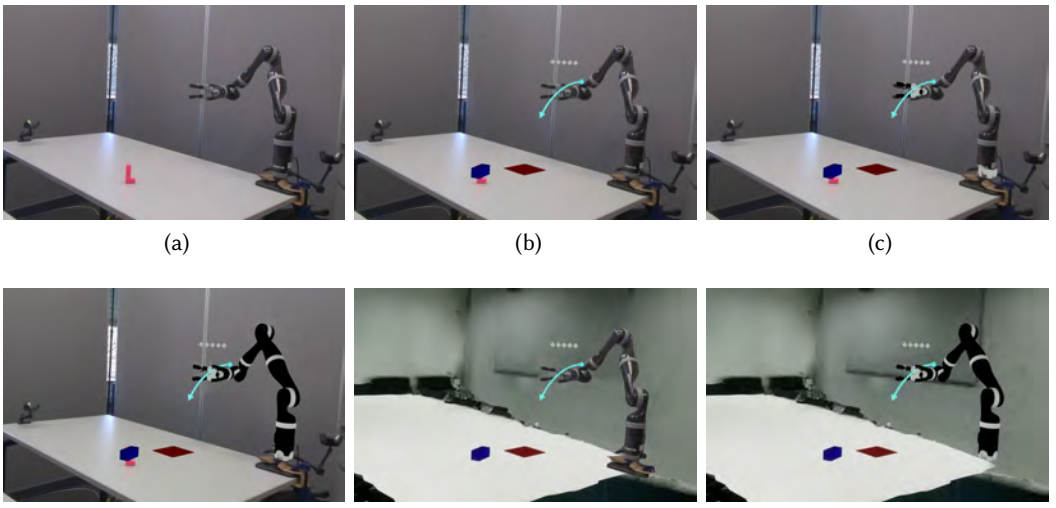


Fig. 5. MR continuum with (**a**) only the real robotic arm in real environment, (**b**) augmenting of directional cues in the real environment with the real robotic arm, (**c**) additional visualizing the gripper and base of the virtual robotic arm in the real environment, (**d**) visualizing the simulated robotic arm in the real environment, (**e**) visualizing the real robotic arm in the virtual environment, and (**f**) visualizing the simulated robotic arm in the virtual environment.

virtual and real environments of the robotic arm are aligned, allowing researchers to seamlessly switch between the user controlling the real and virtual robot. The level of MR can be adjusted in various steps (cf. the *virtuality continuum* of Milgram and Kishino [40]).

The MR environment setups include:

(1) the completely real environment with the real robotic arm,
(2) the real environment extended with visual cues,
(3) the real environment into which the virtual robot is transferred and displayed (with and without visual cues),
(4) the virtual environment into which the real robot is transferred and displayed (with and without visual cues),
(5) the completely virtual environment with the virtual robotic arm.

A comparison of the user's view in reality and simulation can be seen in Figure 5. MR continuum level **(1)** is suitable for study baseline-condition, without any multi-modal feedback to the user. In level **(2)** an AR visualization technique is mimicked, showing the whole physical setup augmented by basic cues. Especially level **(3)** and **(4)** enable customizing either the robot itself or the environment to extent/exchange the physical setup but still not loosing the context. In **(3)** users can interact with a totally new or customized robot while being in a familiar environment. World's distractions can be excluded in **(4)** while the the original robot is presented. Finally, level **(5)** provides a VR environment that can be fully customized.

3.3 Interfaces

We designed *AdaptiX* to facilitate the comparison of different interaction designs, intervention strategies, and feedback techniques for shared robot control. The initial version of the framework

includes interface types for extending user input, ROS integration, and multi-modal feedback. However, this baseline can easily be customized and extended by future development.

3.3.1 User Input. We provide a standard control approach where pressing a keyboard button moves the end effector along cardinal DoFs (x, y, z, roll, pitch, yaw, opening and closing the gripper). Using build-in functionalities, the designated keyboard input can easily be adjusted to other input devices like gamepads, joysticks, or customized assistive input appliances.

In contrast to tele-operating the robotic arm, a *follow-me* approach for any trackable object in 3D space – e.g., the user's handheld VR motion controller – was implemented. The robot's end effector directly follows the movement of the trackable object, which corresponds functionally to direct control. This can be used to generate high-dimensional input and record intended behavior quickly, providing an easy way of interacting and controlling the robot, especially for inexperienced users.

3.3.2 ROS Integration. The ROS integration allows for a bidirectional exchange of information between the simulation and a real robot, mirroring the robot's state *in-silico* and vice versa. Figure 6 shows the involved components: a ROS bridge facilitates the multi-device connection between the framework and the real robot while exchanging robot data. On the ROS side, the messages for the arm position and orientation control and the values for the angle-accurate control of the gripper fingers are read in via the ROS subscriber node. They are then processed, and the robot arm and gripper are controlled through our action client. In addition, the joint angles, the TCP, and the position of all three gripper fingers are published via ROS, which are then input by our *Unreal Engine* framework. The virtual and real robots are synchronized via ROS every 0.1 seconds.

Based on this, our framework provides – depending on the specific context – both a *DigitalTwin* and *PhysicalTwin* approach, allowing the control of either with the other.

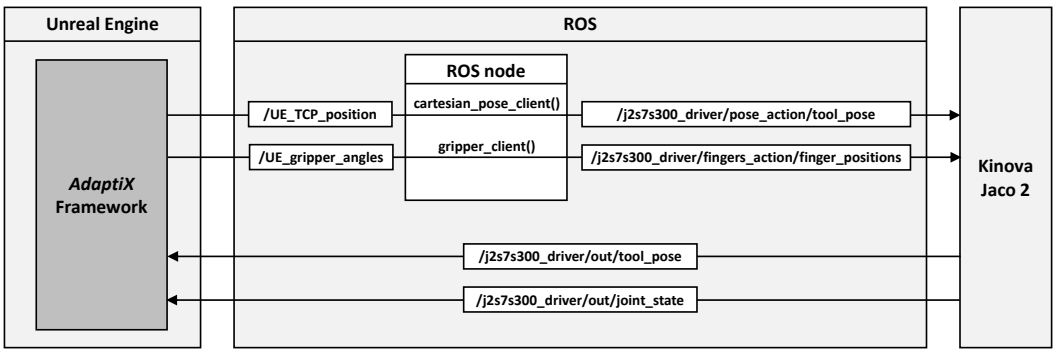


Fig. 6. Component connections of the ROS interface for mixed reality.

3.3.3 Multi-Modal Feedback. To communicate any combination of DoFs, our framework supports several visual cues to illustrate the intended movement trajectory and provides multi-modal feedback extensions via audio and haptic-tactile feedback. Visual feedback can be either provided dynamically attached to the virtual/physical robot's end effector, stationary in the world, or attached to the user's view.

AdaptiX aims to support the development of novel multi-modal interaction and feedback designs either in the pure VR simulation testbed environment or by interacting with a real robot in MR, which mimics an AR setting due to the stereoscopic video-feed. Moreover, it is also possible to show the real robot in our VR simulation environment instead of the simulated one.

Figure 7 shows three exemplary AR-style visualizations provided by the framework, including (a) a robotic ghost overlay, (b) discrete waypoints in 3D, and (c) a variety of multidimensional arrows. Though varying in design, these visualizations can effectively communicate the robot's motion intent to the user.

Ghost: A visualization of robot motion intent by showing an additional version of the robot (or specific components) registered in 3D space, in another color and/or opacity. These visualizations communicate the exact position and orientation a robot at a given time, behaving precisely as though the real robot had been moved this way.

Waypoints: This visualization technique augments the position of a robot (or in our case, the gripper of the robotic arm) in 3D space at a certain point in the future. Usually, the robot navigates linearly between these *Waypoints*, which increases predictability.

Arrow: Among visualizations arguably the most basic but certainly also the most familiar (as seen in traffic navigation systems, road signs, and on keyboards). *Arrows* are found both in straight and curved varieties, where curved arrows indicate a rotation. Given the abundance of *Arrows* in daily life, it makes sense that many robot motion intent visualizations use them.

Classic: This visualization also uses *Arrows*, but in our prototype they are used as a baseline condition to evaluate adaptive and non-adaptive controls. Here, as with the standard input device *Kinova Jaco 2*, two axes can be controlled simultaneously and the user has to choose between different translations and rotations by mode-switching.

(a) Ghost (b) Waypoints (c) Arrows

Fig. 7. Visualization examples pre-implemented in the framework.

All interfaces are modular, enabling quick adaptations and switching between variations. This flexibility allows for studies with clean methodologies and easy comparisons without additional overhead. The community is invited to extend the implementations with any interfaces or control methods desired for their research.

3.4 Recording and Replay

AdaptiX contains an easy-to-use general-purpose system to record, store and replay simulation data, including detailed information about robot states, execution times, or the states of various objects in the environment. The recording system generates Comma-separated values (CSV) text files, which can be accessed with any data manipulation software (e.g., Python or MATLAB). The added output functionality differs significantly from the replaying system provided by the underlying *Unreal Engine*, which is mainly designed for visual replays and – among other things – does not support a CSV file format.

In addition, *AdaptiX*'s recording and replaying system is entirely customizable. Camera re-positioning or re-rendering background scene options are included in the initial version. By default, the recording system tracks the user's view, the robotic arm, and all moveable actors in the virtual environment. All other objects are assumed to be stationary, thus part of the level, and ignored as such. This approach allows for the randomization of background scenes by re-rendering.

The system stores the assigned virtual meshes, scales, possible segmentation tags for each tracked object, and the complete pose data per frame. During the replay process, all objects that were initially recorded in a specific level are swapped with the corresponding data stored in the loaded recording. However, if a different scene is being loaded, the objects from that scene are used instead. In every subsequent frame, all objects are positioned at their respective position until the loaded recording has finished. The system permits custom code to be run at the end of each loaded frame, thus enabling de-bugging and data rendering during replays.

Overall, *AdaptiX* facilitates the lightweight storage of recordings as CSV files with the option to render and store complex and large-scale data (e.g., images or videos) for subsequent evaluation. This lightweight approach is particularly useful when deploying experiments on external devices or recording extensive datasets.

4 FRAMEWORK IMPLEMENTATION

The *AdaptiX* simulation environment is based on the game engine *Unreal Engine 4.27* [14]. The advanced real-time 3D photoreal visuals and immersive experiences provide a suitable foundation for our framework, and assets for future extensions are readily available. *Unreal Engine 4.27* includes integrated options for various hardware setups, thus enabling the framework to be deployed on different operating systems while utilizing most currently available VR/MR/AR headsets, gamepads, and joysticks. At the time of writing, *Unreal Engine 4.27* is free to use, has a considerable user space, and allows unrestricted publications of non-revenue generating research products like the *AdaptiX* framework. Detailed implementation descriptions can be accessed in the *README* provided in the repository at https://adaptix.robot-research.de.

Fig. 8. Example scenario provided in *AdaptiX* including a table, a virtual *Kinova Jaco 2* robotic arm and colored blocks on the tabletop.

4.1 Simulation Environment

The *AdaptiX* default scenario centers on the photogrammetry scan of an actual room that contains a table with an attached virtual robotic arm (see Figure 8). A simulated camera is mounted on the arm's gripper. We added a toggle-off option to hide the camera from the user's view.

The framework includes a straightforward testbed scenario for pick-and-place operations, mimicking the basic principles of most ADLs. The simulation centers around a red surface as a drop target and a blue block as the to-be-manipulated object. Once the object has been successfully placed, the setup randomly re-positions the blue block on the table surface, and the task can be repeated.

We optimized the robotic arm simulation for operation via a VR motion controller with an analog stick, several playable buttons, and motion capture capabilities (e.g., *Meta Quest 2* [38]). These options provide a workable foundation to implement and test diverse interaction concepts, including adaptive concepts which can be configured to match the individual physical abilities of the intended user.

By incorporating the *Varjo XR-3* [59] – a particularly high-resolution XR-Head-Mounted Display (HMD) – we implemented a transitional MR environment. Using two *HTC VIVE* trackers [25], the virtual and real worlds are synchronized so that the robots' working areas are identical. By including the *HTC VIVE* motion controller [24], it is then possible to control the physical robot directly via the *PhysicalTwin* approach of *AdaptiX* (see Figure 1).

The virtual robotic arm is designed as a modular entity, allowing easy integration to new levels following the *Unreal Engine*'s *ActorBlueprint* class structure.

4.1.1 Simulated Robotic Arm. The commercially available *Kinova Jaco 2* assistive robotic arm [29] is specifically designed as an assistive device for people with motor impairments. It is frequently used by a) the target audience and b) researchers – e.g., [3, 20] – during HRI studies, hence the suitability for inclusion in *AdaptiX*.

We designed the simulated *Kinova Jaco 2* as close as possible to the actual product, using virtual meshes generated directly from computer-aided design (CAD) files provided by the manufacturer. Much like in reality, the virtual arm consists of a series of individual links connected by angular joints as shown in the annotated rendering of the assembled model Figure 9.

As *AdaptiX* – including the operation of its simulated robotic arm – is optimized for HRI studies, it focuses on user interaction rather than low-level robot control, whilst also able to incorporate those. Hence, rather than following the standard base-up control, the simulated arm moves in reverse: the user's input directly controls the end effector's motion; the connected joints are positioned to connect the end effector with the base.Each intermediate joint is modeled as a dampened spring with the links unaffected by gravity. This also resolves the redundancy, i.e., joint angle ambiguity a 7-jointed robot has.

Fig. 9. Virtual Robotic Arm with Physics Constraints: purple capsules represent links, green discs represent angular constraints.

This approach allows for nearly arbitrary motion of the end effector and a semi-realistic interaction of the arm with the environment. As a beneficial side effect, developers can disconnect the end effector from the rest of the arm and allow the user to control a free-floating robot hand without any constraints. However, the internal physics engine to realistically handle collisions and interactions between the end effector and the environment is still active.

Likewise, we based the grasp concept on a custom interaction design for robotic grasping rather than physics. Physics-based grasping in a virtual environment is a challenging task [26] and would require substantial preparation and asset fine-tuning from future developers who use the framework. Instead, we defined a logic-based approach that we consider sufficiently realistic for shared control applications: an object is regarded as grasped once it has contact with two opposite fingers while closing the gripper until the fingers open again. The grasped object is rigidly attached to the end effector, keeping its relative position stable and moving alongside the end effector until released.

4.1.2 Simulated Camera System. Computer-aided robot control usually requires a camera system – or a comparable sensor – to measure context information about the current environment for the underlying software function. To provide a realistic equivalent in simulation, *AdaptiX* contains a virtual version of the commercially available *Intel Realsense D435* [27]. This camera system is commonly used in research applications [10, 63] and can deliver aligned color and depth images. The built-in color sensor generates depth data by applying a stereo-vision algorithm using grayscale image data of two built-in infrared (IR) imagers. To improve the texture information captured by the IR imagers, the camera also includes an IR projector, which projects a static pattern on the scene.

As with the simulated robotic arm, the virtual camera system is a modular actor that can be arbitrarily placed within the simulation environment. Its mesh and texture are derived directly from the manufacturer's CAD files to optimize authenticity. The virtual camera system includes all image sensors of the original, plus an optional virtual sensor generating a segmented image of the scene. We designed the virtual sensor parameters to be as close as possible to those of the actual sensors. They include – but are not limited to – sensor dimensions, lens structure, focal length, and aperture.

Because the framework can provide depth information directly from the 3D simulation, the virtual depth camera does not need to calculate its data using stereo-vision but instead yields perfect per-pixel depth information. If stereo-vision-generated depth data with realistic noise, errors, and other algorithm-specific effects is needed, the virtual system also delivers the IR images for a manual calculation.

Additionally, the simulated camera system supports the usage of the image data in-simulation and storing the data on disk for applications such as dataset generation or logging.

4.2 Adaptive DoF Mapping Control (ADMC)

The adaptive DoF mapping is implemented in the object *Axis Wizard*, which provides functions to calculate the optimal suggestion, as well as the other possible optimizations. The calculation relies solely on the virtual objects in the simulation environment instead of object recognition or camera data to enable development and evaluation without a physical robot setup. However, the camera feed for object recognition can be activated by developers to read positions and orientations. In addition to the positions and orientations of the *Gripper Mover* and the *Current Target* (which can be an object to pick up or a target surface to place the object on, depending on the context), two other parameters of *Axis Wizard* are important to ensure the correct calculations for the pick-and-place task – *Minimal Hover Distance* and *Hover Height*.

Disregarding the handling of edge cases, the calculation of the optimal suggestion is taken care of in three steps: 1) calculating *Translation*, 2) calculating *Rotation*, and 3) calculating the finger movement variable *Gripper*. The Blueprints for implementation details are provided in Appendix A.

4.2.1 Calculation of the Optimal Suggestion. *Minimal Hover Distance* represents the distance – projected on the XY-plane – between the *Gripper Mover* and the *Current Target*. When this distance is smaller than the *Minimal Hover Distance* (see Figure 12 in the appendix), the *Axis Wizard* uses a point above the *Current Target* for its calculations – referred to as the *Target Point*, instead of the *Current Target*'s position to prevent the robot from getting too close to the table, allowing for proper gripper rotation. Then, a vector from the *Gripper Mover*'s position towards the *Target Point* is calculated, normalized, and inversely rotated by the *Gripper Mover*'s rotation. This calculation returns a unit vector pointing from the *Gripper Mover* toward the *target point* in the *Gripper Mover*'s reference frame. This vector is then scaled by the *Vel Trans* value of the *Kinova Jaco 2* to get a translation of the size of the movement performed by the *Kinova Jaco 2* during one frame.

Hover Height determines the height of the aforementioned point above the *Current Target*. If the XY-projected distance between the *Gripper Mover* and the *Current Target* is smaller than the *Minimal Hover Distance*, the *Axis Wizard* directly uses the *Current Target*'s position for its calculations instead of the point above it.

To calculate the optimal suggestion's *Rotation*, the *Translation* – calculated in the first step – is used as input for the *Make Rot from X* node. This node returns a *rotator* representing the rotation required to make an object point toward the direction indicated by the input vector – *target point*. To mitigate an additional *roll* of *Gripper Mover*, the inverse value is added, keeping the *Gripper Mover's* orientation largely steady. Additionally, since only a small part of the rotation is performed during one frame, the *rotator* is scaled down. The calculation for the *Rotation*, excluding edge cases, is depicted in Figure 13 in the appendix.

4.2.2 Calculation of Gripper values. The *Gripper* value only depends on whether the target point is within reach of the robotic fingers, either with or without additional movement (i.e. if the fingers are almost close enough, there will be a movement towards the target point, otherwise the fingers will engage without moving the gripper) and whether or not an object is currently being grasped (i.e. if an object is grasped and the gripper is close to the target point, it suggests to open the fingers, otherwise close them).

4.2.3 Calculation of the Adjustment Suggestion. The adjustment suggestion is calculated by rotating the optimal suggestion's *Translation* by 90° around the Y-Axis, keeping the same *Rotation* and setting the *Gripper* value to 0. This results in a DoF mapping which moves roughly along the *Gripper Mover's* Z-Axis, or colloquially "up and down" between the fingers if the optimal suggestion is seen as "forward and backward". As *Rotation* is kept the same between the optimal and adjustment suggestions, the resulting movement keeps the fingers roughly facing the direction of the *Current Target*.

The translation, rotation, and gripper suggestions use much simpler calculations. The translation suggestion calculates a vector from the *Gripper Mover* towards the *Current Target*, inversely rotates it by the *Gripper Mover's* rotation to put it into the *Gripper Mover's* reference frame and uses that as the *Translation* value for the suggested *Adaptive Axis*. This vector is also what the rotation suggestion uses to calculate a *Rotator* representing a rotation towards the *Current Target*. The gripper suggestion checks whether an object is currently being grasped. If so, the suggestion is to open the fingers (*Gripper* = -1). Otherwise, the suggestion is to close the fingers (*Gripper* = 1).

4.2.4 Attention Guidance in Threshold. Both the *Continuous* and *Threshold* approaches share the same core calculation for DoF mappings. However, the *Threshold* approach has an additional task:

determining whether the optimal suggestion significantly differs from the currently active DoF mapping. This task is more related to visualization than the DoF mapping calculation itself and is managed by the *Gizmo* object.

The *Gizmo* object contains a *Realtime Threshold* variable, which represents the threshold as a value between 0 and 1. It also includes a function called *Adaptive Axes Nearly Equal*, which determines whether two *Adaptive Axes* are nearly equal by checking if their difference is below the *Realtime Threshold*. The threshold value is chosen to be between 0 and 1 to align with a percentage of difference (see Section 3.1.3), providing a more intuitive understanding of the amount of difference compared to the cosine similarity value used as the basis for the difference calculation.

As the *Unreal Engine* does not provide an arbitrarily sized vector structure, the calculations required needed to be programmed manually rather than with built-in vector operations. Therefore, two math expression nodes were defined, one calculating the dot product of two 7D vectors and the other calculating the magnitude of a 7D vector. Using these, the cosine similarity between two *Adaptive Axes* could be calculated in *Unreal Blueprints* (see Figure 14 in the appendix). To forego the transformation of the cosine similarity into a percentage difference, the *Unreal Engine's Nearly Equal* node was used to determine whether the cosine similarity was nearly equal to *1* – meaning the vectors align – with a threshold of *2 * Realtime Threshold*. The threshold needed to be multiplied by *2* as the range of the cosine similarity has a magnitude of 2. The result of this calculation is a boolean value that is true if the difference between the *Adaptive Axes* is below the threshold and false otherwise.

The resulting value is then used by the *Gizmo* to show the arrow corresponding to the optimal suggestion. It is also used to notify the *Game Mode* – an object representing the game, keeping track of study variables, etc. – that the threshold was broken. This triggers an event that causes a 1kHz sound to play and a haptic effect to occur on the motion controller. A reset variable is used to prevent the sound from constantly triggering. However, there appears to be a specific point during movement at which it is possible for users to stop their input and the software to get caught in a loop of firing the event and resetting it, causing a constant sound and vibration. If users continued their movement, the software stopped firing the event, seizing the sound and vibration. Unfortunately, this was only noticed during the experiment, which is why the problem persists in the current software version. Assuming *Threshold* is to be used in future research, a better solution for a single fire execution of the notification needs to be developed.

5 LIMITATIONS

In HRI research, the leading factor impacting user experience is usually the chosen method of (shared) control and the respective interfaces. Using frameworks like *AdaptiX* allows researchers to tweak these variables toward high user satisfaction through methodological studies and experiments.

However, like any simulation, *AdaptiX* only approximates reality and contains ingrained limitations when working with the system and evaluating generated results.

5.1 Scenario Selection

In the initial version, *AdaptiX* provides only a single level, as seen in all screenshots of this work. This scenario functions mainly as a model for simple tasks. As such, it lacks environment interactions or varying backgrounds and is not designed for a specific assistive task.

This single level might need to be revised to represent the complete application range of assistive shared control, which is why extensions are required. As such, *AdaptiX*'s modular design allows the community to generate custom levels for their specific research interests effortlessly.

5.2 Simulation Sickness caused by Head Mounted Display

HMDs are a popular tool to create immersive virtual environments, frequently used in research and industrial settings. However, using a HMD in HRI can create a significant displacement between the virtual object and the physical world through effects related to the resulting limited field of view, reduced depth perception, and distorted spatial cues.

For applications within the *AdaptiX* framework, these issues could result in users experiencing motion sickness, disorientation, discomfort, and potentially decreased performance when interacting with the simulated robotic arm or virtual objects. Researchers must consider these artifacts when designing experiments, especially when developing studies including qualitative questionnaires or when comparing different levels of MR continuum.

5.3 Simulation Environment

The simulation environment centers on the photogrammetry scan of an actual room that contains a table with an attached virtual robotic arm. Compared to a 3D modeling of a room, the photogrammetry does not provide a high resolution, leading to a partial blurred appearance.

AdaptiX does not provide a photo realistic virtual environment (yet). However, in our studies, the slightly blurred appearance never seemed to have had a negative effect. On the contrary, it has helped participants focus on the scene's relevant parts (i.e. the robot and objects). Researchers and developers are invited to create and evaluate a 3D modeled environment.

5.4 Simulated Robotic Arm

If controlled entirely in simulation, the robotic arm (as described in Section 4.1.1) does not move identically to an actual *Kinova Jaco 2* because of implementation decisions favoring physical interactions over accurate per-joint robot actions. In most other cases, the individual joints are in relatively realistic positions, even though they might not be identical to the underlying solution provided by an inverse kinematic of the real robot.

Especially in the *follow-me* approach (see Section 3.3.1), it is possible to reach outside of the mechanical range of the robotic arm. Due to the entirely physics-based connection, this results in partially disconnected joints. However, this is only an issue of visualizing the robotic arm in the simulation environment and does not affect the control or the TCP data recording.

Likewise, grasping simulated objects is based on a custom implementation, and grabbed objects are firmly attached to the end effector. Care must be taken for objects that are – in reality – too heavy for the gripper, have slippery surfaces, or have mechanical dimensions that make the object unstable when held. Theoretically, this "ideal kind of grasping" allows the virtual robot to move any arbitrarily large and heavy object. To address this, we added the object tag *Graspable* that allows developers to define permitted – and by omission – unpermitted objects.

5.5 Simulated Camera System

Although the simulated camera is based on manufacturer CAD files, comparison tests failed to deliver completely identical data to the actual recording system. These variances stem from environmental differences between simulation and reality, as light or dust/other particles in the air will cause effects in the produced image. However, these effects can be added in post-production or – if required – activated in the framework. By default, the respective settings are disabled as they would primarily introduce noise that not every developer might want.

On a technical level, the images generated by the virtual system differ slightly in terms of data types. The virtual grayscale IR images consist of three identical color channels instead of a single

channel in reality. Also, the virtual IR and color images include an additional fourth alpha channel, which is not used in our framework.

The generated depth data format also differs, as the actual camera system generates images as 16-bit unsigned integer, and the simulation provides them as 16-bit signed floats. The depth data generated by the framework is pixel-perfect, which ignores various camera system effects that occur in reality by the calculation of depth using stereo-vision.

All these technical differences are addressed within the framework through data transformation and should not noticeably affect the output of *AdaptiX*. However, researchers and developers should be aware of these adjustments for future developments and extension.

5.6 ROS Interface

The ROS interface connects the virtual with a real robot, each with its own environmentally-determined set of limitations. This results in some logical inconsistencies while using the interface. The obvious velocity limitations of the real system result in delayed execution if reality is to follow the simulation. Therefore, the maximum velocity of the virtual robotic arm is set automatically to the physical characteristics after enabling ROS. Also, as the virtual joints are not controlled by an inverse kinematics (IK) but instead based on physics, the interface sends only end effector poses to the real robot, omitting individual joint poses. This may result in differing robot configurations, with only the end effector point being aligned in some instances.

When sending pose data from the real robot to the virtual twin in simulation, most of these restrictions do not apply. The simulated robot can move arbitrarily fast, and its configuration aligns automatically with the real system. The only restriction is that, by default, no further information about the natural environment is available, resulting in a relatively empty virtual environment if relying purely on the ROS interface.

When designing expansions, developers also must be aware that ROS and *Unreal Engine* differ in handedness. ROS is based on a right-handed coordinate system, while the *Unreal Engine* uses a left-handed approach. *AdaptiX* internally does the necessary transformation for the robotic arm but will not automatically calculate this for other position and orientation data, e.g., obstacles. However, researchers can mitigate this by applying the provided coordinate transformation methods of the robotic arm to any further object.

6 FRAMEWORK EXAMPLE ADAPTIONS

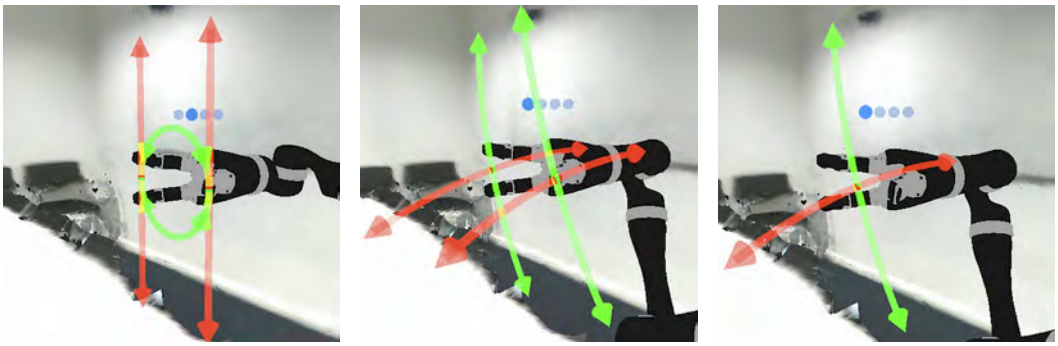
The *AdaptiX* framework has been successfully used in two case studies evaluating interaction concepts and multi-modal feedback with remote and laboratory-based focus groups. A third case study is currently in preparation.

6.1 Example Adaption 1: Adaptive Control of an Assistive Robot

In an initial study [31], the *AdaptiX* framework was used to explore the proposed ADMC control method with associated visual cues for various DoF mappings.

In particular, Kronhardt et al. analyzed how the novel adaptive control method – proposed by Goldau and Frese [18] – performs in a 3D environment compared to the standard mode-switch approach with cardinal DoF mappings. They also investigated whether changes in the visual cues' appearance impact the performance of the adaptive control method.

Three different types of control with varying visual cues and methods of mapping DoFs were compared in a remote online study. These included the *Classic* visualization, one based on *Double Arrow* using two arrows attached to the gripper's fingers, and a visually reduced variant *Single Arrow*, using only one arrow through the middle of the gripper. See Figure 10 for a graphical comparison.



(a) Classic (b) Double Arrow (c) Single Arrow

Fig. 10. Evaluated interaction design and visualizations [31].

Due to the ongoing COVID-19 pandemic, the study was conducted entirely in a VR environment created by *AdaptiX*. Non-specific participants were recruited that had access to the required hardware (an *Oculus Quest* VR-HMD) for an immersive experience.

The participants repeatedly performed a simple pick-and-place task by controlling the virtual *Kinova Jaco 2* using one of the three control types. Comparative results established that adaptive controls require significantly fewer mode switches than the classic control methods. However, task completion time and workload did not improve. Study participants also mentioned concerns about the dynamically changing mapping of combined DoFs and the 2-DoF input device.

Framework contribution: *AdaptiX* demonstrated its effectiveness in this remote study to evaluate new interaction designs and feedback techniques. The innovative advantage is that the physical robotic device does not need to be present during these preliminary studies when testing and evaluating essential design elements. The *Record & Replay* functionality of *AdaptiX* allowed a remote analysis of participants data. This VR approach significantly increases the potential to include end-users in the research and design process while at the same time decreasing cost, time involvement, and accessibility concerns.

6.2 Example Adaption 2: Communicating Adaptive Control Recommendations

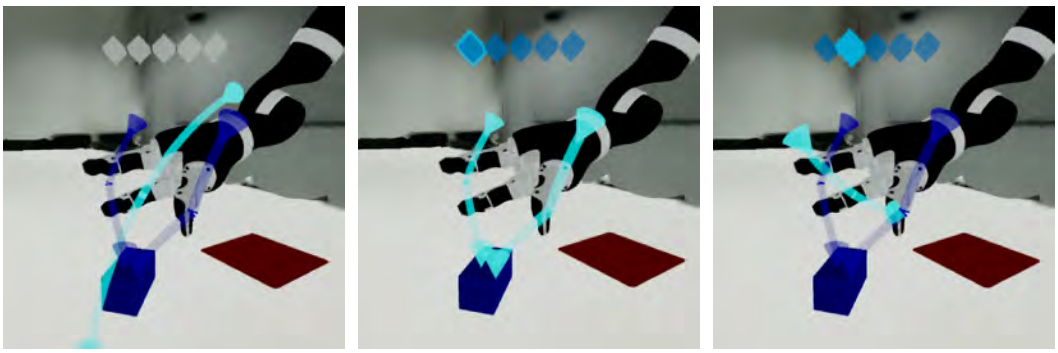
A follow-up study [44] evaluated two new adaptive control methods for an assistive robotic arm, one of which involves a multi-modal approach for attention guiding of the user.

Pascher et al. used *AdaptiX* in a laboratory study to cross-validate the initial study's findings on how participants interact with the environment. The adaptive system re-calculated the best combination of DoFs to complete the task during movement. These calculations were presented to the user as alternative control options for the current task. Users cycled through these suggestions – by pressing a button on the input device – to make a suitable selection or continue moving with the previous active DoFs (see Figure 11).

They contrasted the variants *Continuous* and *Threshold*, differing in the time at which suggestions are communicated to the user, against a non-adaptive *Classic* control method. Possible effects on task completion time, the number of necessary mode switches, perceived workload, and user opinions on each control method were compared. Further, we establish that *Continuous* and *Threshold* performed equally well in quantitative and qualitative insights. Consequently, both are promising approaches to communicating proposed directional cues effectively.

Framework contribution: The integrated multi-modal feedback is an integral feature of *AdaptiX*, capable of supporting the system's real-time suggestions by user attention guiding. Although some participants experienced the combined visual-auditory-haptic multi-modal feedback as "irritating" [44], it effectively communicated updated suggestions. One application of virtual frameworks like *AdaptiX* might be the differentiation between different modality types and corresponding user preferences in an easy-to-set-up study. Highlighting the advantage of our framework, Pascher et al. could evaluate their different visualizations and multi-modal feedback without implementing a VR environment.

Based on the successful implementation of *AdaptiX* in this laboratory study, we are confident that the framework performs well in remote and in-person studies.



(a) (b) (c)

Fig. 11. Suggested control alternatives in light blue, visualized as in case study 2: (**a**) Moving forward and downward towards the object, (**b**) Closing the fingers to grasp the object, and (**c**) Moving towards the target area.

6.3 Example Adaption 3: Comparing Input Devices for Controlling a Physical Robot in Mixed Reality

A third study is currently in preparation. Although unpublished at the time of writing, it highlights the MR capability of the framework and the integration options of different input devices.

In this study, researchers are using the *Varjo XR-3* XR-HMD to explore a similar interaction design and feedback technique to the *Threshold* approach of Pascher et al. [44]. By incorporating this XR-HMD, the prototype mimics an AR environment (see Section 3.2 to the user, seeing the physical setup augmented by visual cues. Instead of a virtual pick-and-place task as before, this study combined a physical object, a physical drop area, and a physical robotic arm with AR cues delivered via the headset.

Participants compared three assistive input techniques: 1) a head-based control by using the deflection of the head on the *pitch* axis for continuous input and on the *roll* axis for mode-switching, 2) a gamepad input by using the *Xbox Adaptive Controller* [39] extended with *Logitech Adaptive Gaming Kit* [34] buttons for a discrete input, and 3) the control-stick of a HMD motion controller – as a baseline to Pascher et al. [44].

Framework contribution: With its real-world setting augmented by virtual cues, the research moved closer to reality on the MR-continuum than the previous two case studies. *AdaptiX* successfully performed as an easy-to-use interface between the usage of a physical robot and virtual communication via a XR-HMD.

It also allowed the research team to quickly evaluate the efficiency of different input devices with the potential to control the robotic arm along the adaptive DoF mapping. The standardized *User Input Adapter* enables researchers to easily chose between different technologies – supporting continuous, discrete, and absolute user input – and further extend it to their needs by its modular nature.

7 CONCLUSION

Integrating *AdaptiX* into HRI research can streamline the development and evaluation of new interaction designs and feedback techniques for controlling assistive robotic arms. The framework is advantageous in remote and in-person studies as its usage negates the need for a physical robotic device during the initial ideation and prototyping stages, thus increasing flexibility, accessibility, and efficiency.

An initial shared control concept by adaptive DoF mapping is provided and implemented to support researchers and developers to either change, extend, or exchange methods with their ideas.

In studies using a physical robot, the integration of ROS bridges the gap to reality, by enabling a bidirectional connection between virtual and physical robotic arm. ROS allows developers and users to choose between a *DigitalTwin* and *PhysicalTwin* approach while interacting with *AdaptiX*.

Using *AdaptiX*, researchers benefit from the entire continuum of MR. As the simulated and real-world environments of the robotic arm are perfectly aligned, nearly seamless switching between controlling the real and virtual robot is possible. This functionality allows applications in pure screen space, VR, AR, simultaneous simulation/reality, and pure reality.

AdaptiX's 3D teach-in interface facilitates a code-less trajectory programming of an assistive robot by hand-guiding the simulated or real robot to the specific location and saving the position and orientation of the tool center point. These waypoints are interpolated to a combined movement trajectory.

The framework's recording/replaying system is entirely customizable. It includes options to change details during replay, such as repositioning cameras or re-rendering background scenes. A fully integrated recording of participants interacting with the robot is possible, which can be analyzed afterward to evaluate the specific research variables.

Taken together, *AdaptiX* is a free and open-source tool that enables HRI researchers to test and evaluate their shared control concepts for assistive robotic devices in a high-resolution virtual environment. The cited case studies clearly demonstrate the benefits researchers and developers can draw from using the framework. The near-endless customization options allow users to tweak the initial version to their specific research needs, resulting in practically tailor-made environments.

7.1 Framework Extensions

We invite the community to extend the *AdaptiX* framework based on their requirements needs by creating custom levels/scenarios and integrating new interfaces. *AdaptiX* can be accessed free-of-charge at https://adaptix.robot-research.de. Refer to the *README* provided in the repository for a detailed description of how to implement experiments in *AdaptiX*.

ACKNOWLEDGMENTS

This research is supported by the *German Federal Ministry of Education and Research* (BMBF, FKZ: 16SV8563, 16SV8565).

REFERENCES

[1] David A. Abbink, Tom Carlson, Mark Mulder, Joost C. F. de Winter, Farzad Aminravan, Tricia L. Gibo, and Erwin R. Boer. 2018. A Topology of Shared Control Systems—Finding Common Ground in Diversity. *IEEE Transactions on*

Human-Machine Systems 48, 5 (Oct. 2018), 509–525. https://doi.org/10.1109/thms.2018.2791570

[2] Stephanie Arévalo-Arboleda, Max Pascher, Annalies Baumeister, Barbara Klein, and Jens Gerken. 2021. Reflecting upon Participatory Design in Human-Robot Collaboration for People with Motor Disabilities: Challenges and Lessons Learned from Three Multiyear Projects. In *The 14th PErvasive Technologies Related to Assistive Environments Conference - PETRA 2021* (2021-06-29). ACM. https://doi.org/10.1145/3453892.3458044

[3] Maude Beaudoin, Josiane Lettre, François Routhier, Philippe S. Archambault, Martin Lemay, and Isabelle Gélinas. 2018. Impacts of robotic arm use on individuals with upper extremity disabilities: A scoping review. *Canadian Journal of Occupational Therapy* 85, 5 (2018), 397–407. https://doi.org/10.1177/0008417418820878 PMID: 30866682.

[4] Connor Brooks and Daniel Szafir. 2020. Visualization of Intended Assistance for Acceptance of Shared Control. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE Press, Piscataway, NJ, USA, 11425–11430. https://doi.org/10.1109/IROS45743.2020.9340964

[5] Jennifer L Burke, Matthew S Prewett, Ashley A Gray, Liuquin Yang, Frederick RB Stilson, Michael D Coovert, Linda R Elliot, and Elizabeth Redden. 2006. Comparing the effects of visual-auditory and visual-tactile feedback on user performance: a meta-analysis. In *Proceedings of the 8th international conference on Multimodal interfaces*. Association for Computing Machinery, New York, NY, USA, 108–117.

[6] Gerard Canal, Guillem Alenyà, and Carme Torras. 2016. Personalization Framework for Adaptive Robotic Feeding Assistance. In *Social Robotics*. Vol. 9979. Springer International Publishing, Cham, 22–31. https://doi.org/10.1007/978-3-319-47437-3{_}3

[7] Ravi Teja Chadalavada, Henrik Andreasson, Maike Schindler, Rainer Palm, and Achim J Lilienthal. 2020. Bi-directional navigation intent communication using spatial augmented reality and eye-tracking glasses for improved safety in human–robot interaction. *Robotics and Computer-Integrated Manufacturing* 61 (2020), 101830.

[8] Yuhang Che, Allison M. Okamura, and Dorsa Sadigh. 2020. Efficient and Trustworthy Social Navigation via Explicit and Implicit Robot–Human Communication. *IEEE Transactions on Robotics* 36, 3 (2020), 692–707. https://doi.org/10.1109/TRO.2020.2964824

[9] Tiffany L. Chen, Chih-Hung King, Andrea L. Thomaz, and Charles C. Kemp. 2011. Touched by a robot. In *Proceedings of the 6th international conference on Human-robot interaction - HRI '11*, Aude Billard, Peter Kahn, Julie A. Adams, and Greg Trafton (Eds.). ACM Press, New York, New York, USA, 457. https://doi.org/10.1145/1957656.1957818

[10] Marianna Ciccarelli, Alessandra Papetti, Cecilia Scoccia, Giacomo Menchi, Leonardo Mostarda, Giacomo Palmieri, and Michele Germani. 2022. A system to improve the physical ergonomics in Human-Robot Collaboration. *Procedia Computer Science* 200 (2022), 689–698. https://doi.org/10.1016/j.procs.2022.01.267

[11] Yann-Seing Law-Kam Cio, Maxime Raison, Cédric Leblond Ménard, and Sofiane Achiche. 2019. Proof of Concept of an Assistive Robotic Arm Control Using Artificial Stereovision and Eye-Tracking. *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 27, 12 (2019), 2344–2352. https://doi.org/10.1109/TNSRE.2019.2950619

[12] Andre Cleaver, Darren Vincent Tang, Victoria Chen, Elaine Schaertl Short, and Jivko Sinapov. 2021. Dynamic Path Visualization for Human-Robot Collaboration. In *Companion of the 2021 ACM/IEEE International Conference on Human-Robot Interaction* (Boulder, CO, USA) *(HRI '21 Companion)*. Association for Computing Machinery, New York, NY, USA, 339–343. https://doi.org/10.1145/3434074.3447188

[13] Adele Diederich and Hans Colonius. 2004. Bimodal and trimodal multisensory enhancement: effects of stimulus onset and intensity on reaction time. *Perception & psychophysics* 66, 8 (2004), 1388–1404.

[14] Epic Games. 2023. Unreal Engine. Published online (last visited on October 19, 2023). https://www.unrealengine.com Version 4.27.2.

[15] Chinemelu Ezeh, Pete Trautman, Louise Devigne, Valentin Bureau, Marie Babel, and Tom Carlson. 2017. Probabilistic vs linear blending approaches to shared control for wheelchair driving. In *2017 International Conference on Rehabilitation Robotics (ICORR)* (London, United Kingdom). IEEE Press, Piscataway, NJ, USA, 835–840. https://doi.org/10.1109/ICORR.2017.8009352

[16] F. Flemisch, D. A. Abbink, M. Itoh, M.-P. Pacaux-Lemoine, and G. Weßel. 2019. Joining the blunt and the pointy end of the spear: towards a common framework of joint action, human–machine cooperation, cooperative guidance and control, shared, traded and supervisory control. *Cognition, Technology & Work* 21, 4 (Aug. 2019), 555–568. https://doi.org/10.1007/s10111-019-00576-1

[17] Michael J Gielniak and Andrea L Thomaz. 2011. Generating anticipation in robot motion. In *2011 RO-MAN*. IEEE Press, Piscataway, NJ, USA, 449–454.

[18] Felix Ferdinand Goldau and Udo Frese. 2021. Learning to Map Degrees of Freedom for Assistive User Control: Towards an Adaptive DoF-Mapping Control for Assistive Robots. In *The 14th PErvasive Technologies Related to Assistive Environments Conference* (Corfu, Greece) *(PETRA 2021)*. Association for Computing Machinery, New York, NY, USA, 132–139. https://doi.org/10.1145/3453892.3453895

[19] Muhammad Abdul Haseeb, Maria Kyrarini, Shuo Jiang, Danijela Ristic-Durrant, and Axel Gräser. 2018. Head Gesture-Based Control for Assistive Robots. In *Proceedings of the 11th PErvasive Technologies Related to Assistive Environments*

Conference (Corfu, Greece) *(PETRA '18)*. Association for Computing Machinery, New York, NY, USA, 379–383. https://doi.org/10.1145/3197768.3201574

[20] Laura V. Herlant, Rachel M. Holladay, and Siddhartha S. Srinivasa. 2016. Assistive Teleoperation of Robot Arms via Automatic Time-Optimal Mode Switching. In *The Eleventh ACM/IEEE International Conference on Human Robot Interaction* (Christchurch, New Zealand) *(HRI '16)*. IEEE Press, Piscataway, NJ, USA, 35–42.

[21] Nicholas J. Hetherington, Elizabeth A. Croft, and H.F. Machiel Van der Loos. 2021. Hey Robot, Which Way Are You Going? Nonverbal Motion Legibility Cues for Human-Robot Spatial Interaction. *IEEE Robotics and Automation Letters* 6, 3 (jul 2021), 5010–5015. https://doi.org/10.1109/lra.2021.3068708

[22] Catherine Holloway. 2019. Disability Interaction (DIX): A Manifesto. *Interactions* 26, 2 (feb 2019), 44–49. https://doi.org/10.1145/3310322

[23] Patrick Holthaus, Trenton Schulz, Gabriella Lakatos, and Rebekka Soma. 2023. Communicative Robot Signals: Presenting a New Typology for Human-Robot Interaction. In *Proceedings of the 2023 ACM/IEEE International Conference on Human-Robot Interaction* (Stockholm, Sweden) *(HRI '23)*. Association for Computing Machinery, New York, NY, USA, 132–141. https://doi.org/10.1145/3568162.3578631

[24] HTC. 2023. VIVE Controller 2.0. Published online (last visited on October 19, 2023). https://www.vive.com/de/accessory/controller2018/

[25] HTC. 2023. VIVE Tracker 3.0. Published online (last visited on October 19, 2023). https://www.vive.com/de/accessory/tracker3/

[26] Markus Höll, Markus Oberweger, Clemens Arth, and Vincent Lepetit. 2018. Efficient Physics-Based Implementation for Realistic Hand-Object Interaction in Virtual Reality. In *2018 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*. IEEE Press, Piscataway, NJ, USA, 175–182. https://doi.org/10.1109/VR.2018.8448284

[27] Intel Corporation. 2022. Intel Realsense - Product Family D400 Series. Published online (last visited on 13.02.2023). https://www.intelrealsense.com/wp-content/uploads/2022/11/Intel-RealSense-D400-Series-Datasheet-November-2022.pdf Revision 014.

[28] Siddarth Jain, Ali Farshchiansadegh, Alexander Broad, Farnaz Abdollahi, Ferdinando Mussa-Ivaldi, and Brenna Argall. 2015. Assistive robotic manipulation through shared autonomy and a Body-Machine Interface. In *2015 IEEE International Conference on Rehabilitation Robotics (ICORR)*. IEEE Press, Piscataway, NJ, USA, 526–531. https://doi.org/10.1109/ICORR.2015.7281253

[29] Kinova inc. 2021. Jaco assistive robot - User guide. Published online (last visited on October 19, 2023). https://assistive.kinovarobotics.com/uploads/EN-UG-007-Jaco-user-guide-R05.pdf EN-UG-007-r05-202111.

[30] Robert J Kosinski. 2008. A literature review on reaction time. *Clemson University* 10, 1 (2008), 337–344.

[31] Kirill Kronhardt, Stephan Rübner, Max Pascher, Felix Goldau, Udo Frese, and Jens Gerken. 2022. Adapt or Perish? Exploring the Effectiveness of Adaptive DoF Control Interaction Methods for Assistive Robot Arms. *Technologies* 10, 1 (2022), 24 pages. https://doi.org/10.3390/technologies10010030

[32] Maria Kyrarini, Fotios Lygerakis, Akilesh Rajavenkatanarayanan, Christos Sevastopoulos, Harish Ram Nambiappan, Kodur Krishna Chaitanya, Ashwin Ramesh Babu, Joanne Mathew, and Fillia Makedon. 2021. A survey of robots in healthcare. *Technologies* 9, 1 (2021), 8.

[33] Audrey Lebrasseur, Josiane Lettre, François Routhier, Philippe S. Archambault, and Alexandre Campeau-Lecours. 2019. Assistive robotic arm: Evaluation of the performance of intelligent algorithms. *Assistive Technology* 33, 2 (May 2019), 95–104. https://doi.org/10.1080/10400435.2019.1601649

[34] Logitech. 2023. Adaptive Gaming Kit. Published online (last visited on October 19, 2023). https://www.logitechg.com/en-us/products/gamepads/adaptive-gaming-kit-accessories.943-000318.html

[35] Zhanat Makhataeva and Huseyin Atakan Varol. 2020. Augmented Reality for Robotics: A Review. *Robotics* 9, 2 (2020). https://doi.org/10.3390/robotics9020021

[36] Elias Matsas and George-Christopher Vosniakos. 2015. Design of a virtual reality training system for human–robot collaboration in manufacturing tasks. *International Journal on Interactive Design and Manufacturing (IJIDeM)* 11, 2 (Feb. 2015), 139–153. https://doi.org/10.1007/s12008-015-0259-2

[37] Alyxander David May, Christian Dondrup, and Marc Hanheide. 2015. Show me your moves! Conveying navigation intention of a mobile robot to humans. In *2015 European Conference on Mobile Robots (ECMR)*. 1–6. https://doi.org/10.1109/ECMR.2015.7324049

[38] Meta. 2023. Meta Quest 2. Published online (last visited on October 19, 2023). https://www.meta.com/de/quest/products/quest-2/

[39] Microsoft. 2023. Xbox Adaptive Controller. Published online (last visited on October 19, 2023). https://www.microsoft.com/en-us/d/xbox-adaptive-controller/8nsdbhz1n3d8

[40] Paul Milgram and Fumio Kishino. 1994. A taxonomy of mixed reality visual displays. *IEICE TRANSACTIONS on Information and Systems* 77, 12 (1994), 1321–1329.

[41] Sarah Luisa Müller, Stefan Schröder, Sabina Jeschke, and Anja Richert. 2017. Design of a Robotic Workmate. In *Lecture Notes in Computer Science*. Springer International Publishing, 447–456. https://doi.org/10.1007/978-3-319-58463-8_37

[42] Max Pascher, Annalies Baumeister, Stefan Schneegass, Barbara Klein, and Jens Gerken. 2021. Recommendations for the Development of a Robotic Drinking and Eating Aid - An Ethnographic Study. In *Human-Computer Interaction – INTERACT 2021* (2021-09-01), Carmelo Ardito, Rosa Lanzilotti, Alessio Malizia, Helen Petrie, Antonio Piccinno, Giuseppe Desolda, and Kori Inkpen (Eds.). Springer, Cham. https://doi.org/10.1007/978-3-030-85623-6_21

[43] Max Pascher, Uwe Gruenefeld, Stefan Schneegass, and Jens Gerken. 2023. How to Communicate Robot Motion Intent: A Scoping Review. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems - CHI '23*. https://doi.org/10.1145/3544548.3580857

[44] Max Pascher, Kirill Kronhardt, Felix Ferdinand Goldau, Udo Frese, and Jens Gerken. 2023. In Time and Space: Towards Usable Adaptive Control for Assistive Robotic Arms. In *2023 32th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*. IEEE.

[45] Sebastiaan Petermeijer, Pavlo Bazilinskyy, Klaus Bengler, and Joost de Winter. 2017. Take-over again: Investigating multimodal and directional TORs to get the driver back into the loop. *Applied Ergonomics* 62 (2017), 204–215. https://doi.org/10.1016/j.apergo.2017.02.023

[46] Sebastiaan Petermeijer, Fabian Doubek, and Joost de Winter. 2017. Driver response times to auditory, visual, and tactile take-over requests: A simulator study with 101 participants. In *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. 1505–1510. https://doi.org/10.1109/SMC.2017.8122827

[47] Laura Petrich, Jun Jin, Masood Dehghan, and Martin Jagersand. 2022. A Quantitative Analysis of Activities of Daily Living: Insights into Improving Functional Independence with Assistive Robotics. In *2022 International Conference on Robotics and Automation (ICRA)*. 6999–7006. https://doi.org/10.1109/ICRA46639.2022.9811960

[48] Anita Pollak, Mateusz Paliga, Matias M. Pulopulos, Barbara Kozusznik, and Malgorzata W. Kozusznik. 2020. Stress in manual and autonomous modes of collaboration with a cobot. *Computers in Human Behavior* 112 (2020), 106469. https://doi.org/10.1016/j.chb.2020.106469

[49] Gabriel Quere, Annette Hagengruber, Maged Iskandar, Samuel Bustamante, Daniel Leidner, Freek Stulp, and Jörn Vogel. 2020. Shared Control Templates for Assistive Robotics. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE Press, Piscataway, NJ, USA, 1956–1962. https://doi.org/10.1109/ICRA40945.2020.9197041

[50] Emanuele Ruffaldi, Filippo Brizzi, Franco Tecchia, and Sandro Bacinelli. 2016. Third point of view augmented reality for robot intentions visualization. In *International Conference on Augmented Reality, Virtual Reality and Computer Graphics*. Springer, 471–478.

[51] Ivan Shindev, Yu Sun, Michael Coovert, Jenny Pavlova, and Tiffany Lee. 2012. Exploration of Intention Expression for Robots. In *Proceedings of the Seventh Annual ACM/IEEE International Conference on Human-Robot Interaction* (Boston, Massachusetts, USA) *(HRI '12)*. Association for Computing Machinery, New York, NY, USA, 247–248. https://doi.org/10.1145/2157689.2157778

[52] Moondeep C. Shrestha, Ayano Kobayashi, Tomoya Onishi, Erika Uno, Hayato Yanagawa, Yuta Yokoyama, Mitsuhiro Kamezaki, Alexander Schmitz, and Shigeki Sugano. 2016. Intent communication in navigation through the use of light and screen indicators. In *2016 11th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*. 523–524. https://doi.org/10.1109/HRI.2016.7451837

[53] Joris Sijs, Freek Liefhebber, and Gert Willem R.B.E. Romer. 2007. Combined Position & Force Control for a robotic manipulator. In *2007 IEEE 10th International Conference on Rehabilitation Robotics*. IEEE. https://doi.org/10.1109/icorr.2007.4428414

[54] Amit Singhal. 2001. Modern information retrieval: A brief overview. *IEEE Data Eng. Bull.* 24, 4 (2001), 35–43.

[55] Ryo Suzuki, Adnan Karim, Tian Xia, Hooman Hedayati, and Nicolai Marquardt. 2022. Augmented Reality and Robotics: A Survey and Taxonomy for AR-Enhanced Human-Robot Interaction and Robotic Interfaces. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems* (New Orleans, LA, USA) *(CHI '22)*. Association for Computing Machinery, New York, NY, USA, Article 553, 33 pages. https://doi.org/10.1145/3491102.3517719

[56] Emmanuele Tidoni, Mohammad Abu-Alqumsan, Daniele Leonardis, Christoph Kapeller, Gabriele Fusco, Cristoph Guger, Cristoph Hintermuller, Angelika Peer, Antonio Frisoli, Franco Tecchia, Massimo Bergamasco, and Salvatore Maria Aglioti. 2017. Local and Remote Cooperation With Virtual and Robotic Agents: A P300 BCI Study in Healthy and People Living With Spinal Cord Injury. *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 25, 9 (Sept. 2017), 1622–1632. https://doi.org/10.1109/tnsre.2016.2626391

[57] Katherine M Tsui, Dae-Jin Kim, Aman Behal, David Kontak, and Holly A Yanco. 2011. "I want that": Human-in-the-loop control of a wheelchair-mounted robotic arm. *Applied Bionics and Biomechanics* 8, 1 (2011), 127–147.

[58] Bram van Deurzen, Herman Bruyninckx, and Kris Luyten. 2022. Choreobot: A Reference Framework and Online Visual Dashboard for Supporting the Design of Intelligible Robotic Systems. *Proc. ACM Hum.-Comput. Interact.* 6, EICS, Article 151 (jun 2022), 24 pages. https://doi.org/10.1145/3532201

[59] Varjo. 2023. Varjo XR-3. Published online (last visited on October 19, 2023). https://varjo.com/products/xr-3/

[60] Michael Walker, Hooman Hedayati, Jennifer Lee, and Daniel Szafir. 2018. Communicating Robot Motion Intent with Augmented Reality. In *Proceedings of the 2018 ACM/IEEE International Conference on Human-Robot Interaction* (Chicago, IL, USA) *(HRI '18)*. Association for Computing Machinery, New York, NY, USA, 316–324. https://doi.org/10.1145/3171221.3171253

[61] Hanna Yun and Ji Hyun Yang. 2020. Multimodal warning design for take-over request in conditionally automated driving. *European transport research review* 12, 1 (2020), 1–11.

[62] Mohammad Kassem Zein, Majd Al Aawar, Daniel Asmar, and Imad H Elhajj. 2021. Deep learning and mixed reality to autocomplete teleoperation. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 4523–4529.

[63] He Zhang and Cang Ye. 2019. Human-Robot Interaction for Assisted Wayfinding of a Robotic Navigation Aid for the Blind. In *2019 12th International Conference on Human System Interaction (HSI)*. IEEE. https://doi.org/10.1109/hsi47298.2019.8942612

[64] Jakub Złotowski, Kumar Yogeeswaran, and Christoph Bartneck. 2017. Can we control it? Autonomous robots threaten human identity, uniqueness, safety, and resources. *International Journal of Human-Computer Studies* 100 (2017), 48–54.

A BLUEPRINTS OF ADMC IMPLEMENTATION

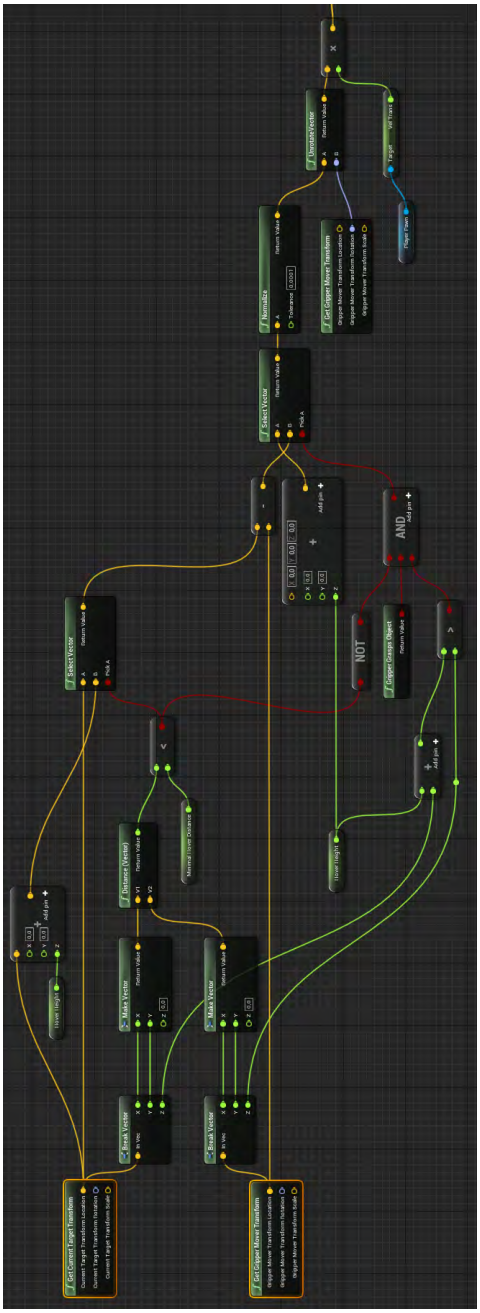


Fig. 12. Calculation of the translation for the *Optimal Suggestion*: Excerpt of *Blueprint* code calculating the *Translation* value of the *Adaptive Axis* for the *Optimal Suggestion*. Not pictured: Edge case handling for gripping an object.

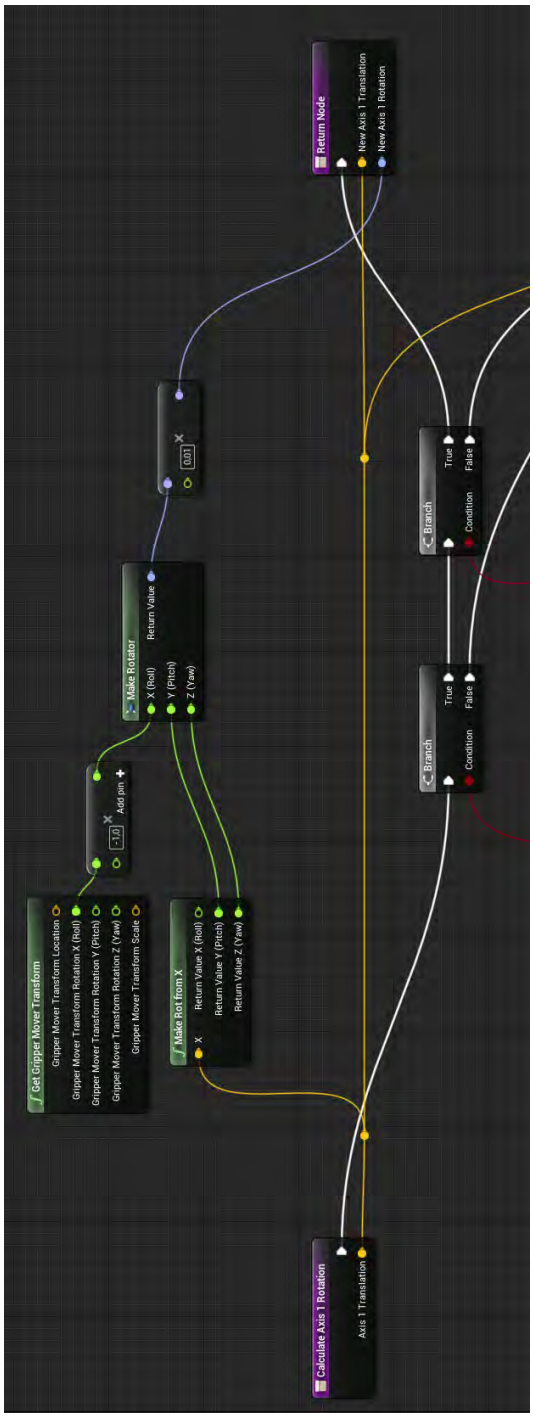


Fig. 13. Calculation of the Rotation for the *Optimal Suggestion*: Excerpt of *Blueprint* code calculating the *Rotation* value of the *Adaptive Axis* for the *Optimal Suggestion*. Not pictured: Edge case handling.

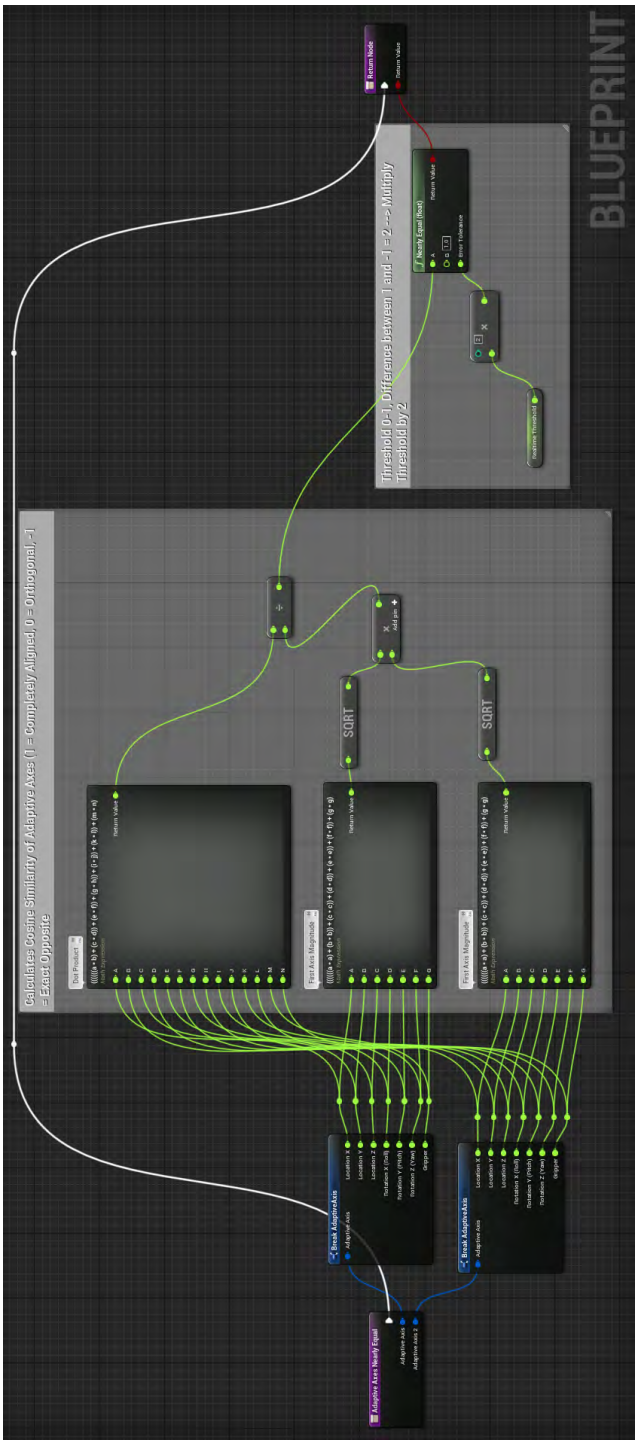


Fig. 14. *Adaptive Axes Nearly Equal* function to prepare the multi-modal attention guiding of the user.