VIRTUAL REALITY AS A STEPPING STONE TO REAL-WORLD ROBOTIC CAREGIVING

A Dissertation
Presented to
The Academic Faculty

By

Yijun Gu

In Partial Fulfillment
of the Requirements for the Degree
Master of Science in Computer Science in the
School of Interactive Computing

Georgia Institute of Technology

May 2021

VIRTUAL REALITY AS A STEPPING STONE TO REAL-WORLD ROBOTIC CAREGIVING

Thesis committee:

Prof. Charles C. Kemp
Dept. of Biomedical Engineering
Georgia Institute of Technology

Prof. Greg Turk
School of Interactive Computing
Georgia Institute of Technology

Prof. C. Karen Liu
Dept. of Computer Science
Stanford University

Date approved: April 28, 2021

ACKNOWLEDGMENTS

I thank Prof. Charles C. Kemp and Zackory Erickson for serving as my mentor during my time at the Healthcare Robotics Lab, inspiring me with various ideas and helping me with research work. I thank Prof. Greg Turk and Prof. C. Karen Liu for serving on my thesis reading committee. I thank Henry Clever and Patrick Grady for feedback and advice about this work. I am also grateful for the love and the support of my family and friends.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

Versatile robotic caregivers could benefit millions of people worldwide, including older adults and people with disabilities. Recent work has explored how robotic caregivers can learn to interact with people through physics simulations, yet transferring what has been learned to real robots remains challenging. By bringing real people into the robot's virtual world, virtual reality (VR) has the potential to help bridge the gap between simulations and the real world. In this thesis, we present Assistive VR Gym (AVR Gym), which enables real people to interact with virtual assistive robots. We also provide evidence that AVR Gym can help researchers improve the performance of simulation-trained assistive robots with real people. Prior to AVR Gym, we trained robot control policies (*Original Policies*) solely in simulation for four robotic caregiving tasks (robot-assisted feeding, drinking, itch scratching, and bed bathing) with two simulated robots (PR2 from Willow Garage and Jaco from Kinova). With AVR Gym, we developed *Revised Policies* based on insights gained from testing the Original policies with real people. Through a formal study with eight participants in AVR Gym, we found that the Original policies performed poorly, the Revised policies performed significantly better, and that improvements to the biomechanical models used to train the Revised policies resulted in simulated people that better match real participants. Notably, participants significantly disagreed that the Original policies were successful at assistance, but significantly agreed that the Revised policies were successful at assistance. Overall, our results suggest that VR can be used to improve the performance of simulation-trained control policies with real people without putting people at risk, thereby serving as a valuable stepping stone to real robotic assistance.

CHAPTER 1

INTRODUCTION

Robotic assistance with activities of daily living (ADLs) could increase the independence of people with disabilities, improve quality of life, and help address pressing societal needs, such as aging populations, high healthcare costs, and shortages of healthcare workers [1, 2].

Physics simulations provide an opportunity for robots to safely learn how to physically assist people. Yet, the reality gap between physics simulations and real assistance poses several challenges. By bringing real people into the robot's virtual world, virtual reality (VR) has the potential to serve as a stepping stone between simulated worlds and the real world. For assistive robots, the person receiving assistance is an extremely important and complex part of the environment. As we show through a formal study, enabling real people to interact with a virtual robot can quickly reveal deficiencies through both objective and subjective measures of performance.

To facilitate the use of VR in the development of assistive robots, we present Assistive VR Gym[1] (AVR Gym) an open source framework that enables real people to interact with virtual assistive robots within a physics simulation (see Figure 1.1). AVR Gym builds on Assistive Gym, which is an open source physics simulation framework for assistive robots that models multiple assistive tasks [3]. AVR Gym enables people to interact with virtual robots without putting themselves at risk, which is especially valuable when evaluating controllers that have been trained in simulation with virtual humans.

As confirmed through a formal study with eight participants, our use of AVR Gym enabled us to identify significant, unexpected shortcomings in the simulation-trained baseline control policies that were originally released with Assistive Gym. Moreover, we were able

[1] https://github.com/Healthcare-Robotics/assistive-vr-gym

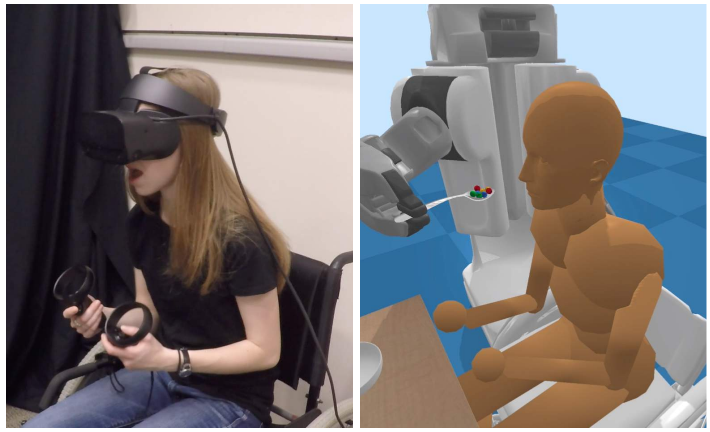

Figure 1.1: A participant using VR to interact with a PR2 robot that has learned how to provide feeding assistance through physics simulation.

to use AVR Gym to improve the control policies by discovering that the simulated humans did not adequately represent the biomechanics of real people. Improving the simulated humans used to train control policies, resulted in Revised control policies that dramatically outperformed our Original policies.

In Chapter 2, we discuss related work focused on physically assistive robotics and VR. In Chapter 3, we present AVR Gym. Chapter 4 describes baseline control policies and revised policies. Chapter 5 details our evaluation of virtual assistive robots with two experimental methods, including objective and subjective measures and summarizes the results of the evaluation. We prove that biomechanical models significantly impact policy performance and can be assessed and improved via VR. In Chapter 6, we discuss the limitation of the work and list out possible future work. In Chapter 7, we present our conclusion. The majority part of the thesis has been published as a paper to the IEEE International Conference on Robot & Human Interactive Communication (RO-MAN) 2020 [4].

CHAPTER 2

RELATED WORK

2.1 Physically Assistive Robotics

A significant amount of research has been conducted on real robotic systems for providing physical assistance to people. For example, researchers have demonstrated how table-mounted robotic arms can provide drinking assistance to people using a force sensing smart cup [5] and a brain-machine interface for shared-autonomy [6]. Similarly, robots have shown promising results for providing feeding assistance [7, 8, 9, 10, 11, 12]. Prior research has also investigated robotic assistance for bed bathing [13, 14], which can be especially valuable for people who are confined to a bed due to disabilities or injuries. Robots also present an opportunity to help individuals in getting dressed with a variety of garments. Robot-assisted dressing has been demonstrated on several robotic platforms for assisting both able-bodied participants and real people with disabilities [15, 16, 17, 18, 19, 14, 20, 21, 22, 23].

Despite these promising results in physical robotic assistance, it remains challenging to design robotic systems that can assist with multiple tasks across a wide spectrum of human shapes, sizes, weights, and disabilities. This is due in part to difficulties and costs associated with evaluating assistive robotic systems across a large distribution of people. Physics simulation presents an opportunity for robots to learn to safely interact with people over many tasks and environments. For instance, researchers have demonstrated the use of physics simulation for learning robot controllers for robot-assisted dressing tasks [24, 25, 26, 27]. Clegg et al. has also used Assistive Gym to learn control policies for a real PR2 robot to dress an arm of a humanoid robot with a hospital gown [28].

2.2 Virtual Reality

We use virtual reality (VR) to evaluate and improve simulation-trained assistive robots with real people. Studies have provided evidence that VR can offer people both a sense of presence and embodiment [29, 30]. Accordingly, VR provides an opportunity to assess important attributes for human-robot interaction (HRI), including proxemics, legibility of motion, and embodiment [31, 32].

VR has been widely used as a tool to improve robot performance across an assortment of tasks. A common use for VR is for teleoperation of robots for dexterous manipulation tasks [33, 34]. This VR teleoperation has also been used to collect high-quality robotic manipulation demonstrations [35, 36, 37]. Within robotic rehabilitation, VR has been used to enhance rehabilitation training and assess rehabilitation metrics in real time [38, 39, 40, 41, 42].

Across these works, VR has often been used for people to take control of robots; to provide demonstrations and to train robots for performing various tasks. In this thesis, we show that VR can be used to safely evaluate and improve simulation-trained assistive robot controllers with real people.

CHAPTER 3

ASSISTIVE VR GYM

AVR Gym builds on Assistive Gym and is implemented using the open source PyBullet physics engine [43]. We connect assistive environments from Assistive Gym into VR with vrBullet, a VR physics server that uses the OpenVR API. We use the Oculus Rift S VR head-mounted display, which uses inside-out tracking, and two Oculus Touch controllers, each of which provides 3-DoF position and 3-DoF orientation tracking. Figure 3.1 depicts the VR setup with 3rd and 1st person perspectives of the virtual environment.

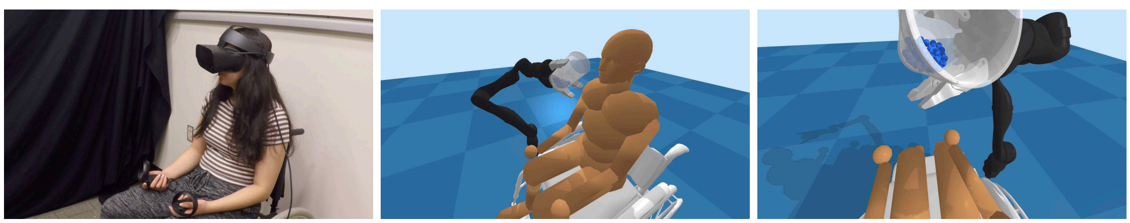

Figure 3.1: (Left) Human participant using VR interface to receive drinking assistance from a simulated robot. (Middle) 3rd person perspective of the simulated human model that the human participant is controlling. (Right) 1st person perspective of what a participant observed in VR.

The simulated human model in VR has 20 controllable joints including two 7-DoF arms, a 3-DoF head, and a 3-DoF waist. Using the Oculus' inside-out tracking, we estimated each human participant's torso height, the distance from hipbone to center of the head. We observed that people often tilt their head and twist their body leading to an inaccurate measurement. We designed a calibration process that helps human participants to turn their head straight ahead to the front. The summarized Python code below walks through the major steps of the calibration process[1].

```
def calibration():
    while True:
```

[1] https://github.com/Healthcare-Robotics/assistive-vr-gym/blob/master/assistive_gym/envs/env.py#L114

```
    # Query the headset for its 3D Cartesian position and 3D orientations
    headset_pos, headset_orient = getVREvents(deviceType=VR_DEVICE_HMD)
    roll_diff, pitch_diff, yaw_diff = np.rad2deg(headset_orient) - front_orient # (90, 0, 0)

    # For roll, pitch, and yaw, check whether the degree of the angle is within the range of (-5, 5) and draw the arrow and text with red/green color.
    roll_color, roll_direction = assign_color_direction(roll_diff, check_range(roll_diff, range=5))
    draw_arrow(roll_color, roll_direction)
    draw_text(roll_color, roll_diff)

    pitch_color, pitch_direction = assign_color_direction(pitch_diff, check_range(pitch_diff, range=5))
    draw_arrow(pitch_color, pitch_direction)
    draw_text(pitch_color, pitch_diff)

    yaw_color, yaw_direction = assign_color_direction(yaw_diff, check_range(yaw_diff, range=5))
    draw_arrow(yaw_color, yaw_direction)
    draw_text(yaw_color, yaw_diff)

    # Measure torso height if orientations are within the range of (-5, 5)
    if check_range(roll_diff, range=5) and check_range(pitch_diff, range=5) and check_range(yaw_diff, range=5):
        HMD_distance = headset_pos[1] - bed_height if task in BedBathing else wheelchair_height
```

```
break
```

Listing 3.1: summarized Python code for calibration algorithm

We defined the front $\boldsymbol{\phi} = (\phi_r, \phi_p, \phi_y)$ in global space. In AVR Gym, ϕ_r controls the head to move up and down and is set to 90° here; ϕ_p controls the head to tilt and is set to 0° here; ϕ_r controls the head to move left and right and is set to the direction of the wheelchair or the bed faces toward, 0° here.

As shown in Listing 3.1 and Figure 3.2, we queried the headset for its 3D Cartesian position, $\boldsymbol{\chi}$, and 3D orientation, $\boldsymbol{\theta} = (\theta_r, \theta_p, \theta_y)$ in global space. Let $\boldsymbol{\psi} = (\psi_r, \psi_p, \psi_y) = \boldsymbol{\theta} - \boldsymbol{\phi}$ be the 3D vector from the center of a person's head to the front. With the 3D vector, we drew an arrow pointing from head to the front and drew the text about the degree next to the arrow. We checked whether ψ_r, ψ_p, ψ_y are within the range of (-5°, 5°). If the orientation is within the range, we used green color to draw the arrow and text; otherwise, we used red color. Human participants adjusted the direction their head following the arrows and texts until all of them turned to green. We then measured the torso height as the difference from the headset to the wheelchair or the hospital bed.

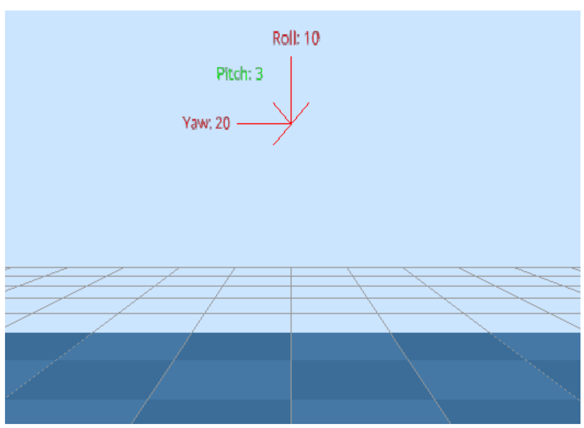

Figure 3.2: Calibration system to help human participant to turn his/her head to the front.

Given this height estimate, we modified the height of the simulated human body to match that of the real human participant. We then used the VR headset to align the head and waist joints of a virtual human model to the person's pose. The summarized Python code below walks through the major steps of the head and waist alignment algorithm[2].

```python
def head_waist_alignment(self):
    # Query the headset for its 3D Cartesian positions and orientations
    headset_pos, headset_orient = getVREvents(deviceType=VR_DEVICE_HMD)

    # Compute position and roll, pitch, yaw joints of the simulated human head to align with the position and orientation of the headset
    roll, pitch, yaw = headset_orient
    target_head_orient = [-roll + np.deg2rad(90), -pitch, yaw - np.deg2rad(180) if yaw > human.head_yaw else yaw + np.deg2rad(180)]
    target_head_pos = multiplyTransforms(headset_pos, target_head_orient, [0, 0.08, 0], [0, 0, 0, 1])[0]

    # Get the 3D vector from the center of a person's waist to the center of their head
    psi_x, psi_y, psi_z = target_head_pos - get_cartesian_pos_quaternion_orient(human.waist)

    # Compute roll, pitch, yaw joints of the simulated human waist
    r_psi = atan2(psi_y, psi_z)
    p_psi = atan2(x*cos(r_psi), psi_z)
    y_psi = atan2(cos(r_psi), sin(r_psi)*sin(p_psi))
    target_waist_orient = [r_psi, p_psi, y_psi]

    # Distribute part of the measured head yaw orientation to the waist
    target_head_orient[2] = 80 * (target_head_orient[2] - y_psi) / 110
    target_waist_orient[2] = 30 * (target_head_orient[2] - y_psi) / 110

    # Control the simulated human to reach the target head joint values within the maximum motor forces
    target_head_orient -= [r_psi, p_psi, 0]
```

[2] https://github.com/Healthcare-Robotics/assistive-vr-gym/blob/master/assistive_gym/envs/env.py#L189

```
    PD_control(human.head_joints, target_head_orient, head_gains, head_forces)

    # Control the simulated human to reach the target waist joint values within the maximum motor forces
    PD_control(human.waist_joints, target_waist_orient, waist_gains, waist_forces)
```

Listing 3.2: summarized Python code for head and waist alignment algorithm

As shown in Listing 3.2, at each time step, we first queried the headset for its 3D Cartesian position, $\boldsymbol{\chi}$, and 3D orientation, $\boldsymbol{\theta} = (\theta_r, \theta_p, \theta_y)$ in global space. We computed the position and roll, pitch, yaw joints of the simulated human head to align with the position, $\boldsymbol{\chi}$, and orientation, $\boldsymbol{\theta}$, of the headset. To align the waist pose, we computed the appropriate angles needed such that the simulated human had the same 3D Cartesian head position as the real person. We first observed the fixed 3D position for the center of a person's waist, $\boldsymbol{W}$, given as a fixed offset above the wheelchair or hospital bed. Let $\boldsymbol{\psi} = (\psi_x, \psi_y, \psi_z) = \boldsymbol{\chi} - \boldsymbol{W}$ be the 3D vector from the center of a person's waist to the center of their head. We then computed the roll, pitch, and yaw $(r_{\boldsymbol{\psi}}, p_{\boldsymbol{\psi}}, y_{\boldsymbol{\psi}})$ orientations for the waist of the simulated human according to:

$$
\begin{aligned}
r_{\boldsymbol{\psi}} &= \text{atan2}(\psi_y, \psi_z) \\
p_{\boldsymbol{\psi}} &= \text{atan2}(\psi_x \cdot \cos(r_{\boldsymbol{\psi}}), \psi_z) \\
y_{\boldsymbol{\psi}} &= \text{atan2}(\cos(r_{\boldsymbol{\psi}}), \sin(r_{\boldsymbol{\psi}}) \cdot \sin(p_{\boldsymbol{\psi}}))
\end{aligned}
$$

We observed that people often prefer to use some waist motion when looking left or right. To account for this, we distributed some of the measured head yaw orientation to the waist. Specifically, we let the yaw orientation of the head to be $0.7(\theta_y - y_{\boldsymbol{\psi}})$, and the yaw orientation of the waist joint to be $0.3(\theta_y - y_{\boldsymbol{\psi}})$. Finally, we controlled the physics engine to simulate the head and waist joint motors to reach the target value we compute within the maximum motor forces.

As shown in Figure 3.1, participants also held two Oculus Touch controllers, which we used to control both arms of the simulated human in VR. The summarized Python code below walks through the major steps of arm alignment algorithm[3].

```python
def arm_alignment(self):
    # Query the controllers for their 3D Cartesian positions and orientations
    controllers_pos, controllers_orient = getVREvents(deviceType=VR_DEVICE_CONTROLLER)

    # Control the simulated human left and right arm.
    arm_control(controllers_pos[0], controllers_orient[0], human.left_arm, human.left_arm_joints)  # Left arm
    arm_control(controllers_pos[1], controllers_orient[1], human.right_arm, human.right_arm_joints)  # Right arm

def arm_control(self, controller_pos, controller_orient, arm, arm_joints):
    #  Compute position and roll, pitch, yaw joints of the simulated human hand to align with the position and orientation of the controller
    roll, pitch, yaw = controller_orient
    target_hand_orient = [-roll, -pitch, yaw - np.deg2rad(180)]
    target_hand_pos = multiplyTransforms(controller_pos, target_hand_orient, [0, 0, 0.08], [0, 0, 0, 1])[0]

    # Compute the 7-DoF arm joint angles using inverse kinematics
    hand_pos, hand_orient = get_cartesian_pos_quaternion_orient(human.hand)

    if l2_norm(hand_pos, target_hand_pos) > 0.001 and l2_norm(
```

[3] https://github.com/Healthcare-Robotics/assistive-vr-gym/blob/master/assistive_gym/envs/env.py#L222

```
hand_orient, target_hand_orient) > 0.01:
        # Sample 100 IK solutions for the arm joints and attempts to find one necessary for the hand to reach the target position and orientation
        target_angles = arm.ik(human.hand, target_hand_pos, target_hand_orient, arm.all_joint_indices, max_iterations=100)

        # Control the simulated human to reach the target arm joint values within the maximum motor forces.
        PD_control(arm_joints, target_angles, arm_gains, arm_forces)
```

Listing 3.3: summarized Python code for the arm alignment algorithm

As shown in Listing 3.3, at each time step, we queried each handheld controller for its 3D Cartesian position and orientation in global space. We computed the position and roll, pitch, yaw joints of the simulated human hand to align with the position, χ, and orientation, θ, of the controller. Then using inverse kinematics with a damped least squares method, we computed the 7-DoF arm joint angles necessary for the hand of the simulated person to have the same position and orientation as measured by the controller. Since we are optimizing for a 7-DoF arm pose using a 6-DoF controller pose measurement, this inverse problem is ill-posed, which implies that stable solutions are not guaranteed. However, we have found this inverse kinematics optimization to work well in practice, with only minor offsets in the estimated pose of a person's full arm. Finally, we controlled the physics engine to simulate the arm joint motors to reach the target value we compute within the maximum motor forces.

3.1 Test Environments

AVR Gym builds upon four physics-based assistive environments from Assistive Gym. A key aspect of achieving realism for users was to match the simulated wheelchair and bed to a real wheelchair and bed. Each of these environments, shown in Figure 3.3, is associated

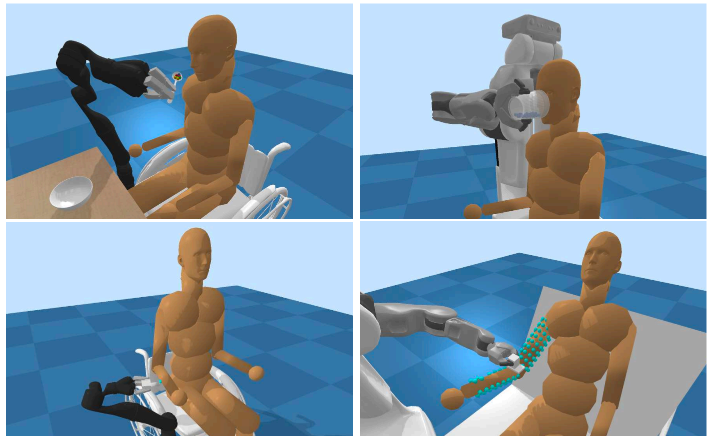

Figure 3.3: Assistive environments in physics simulation with the Jaco and PR2 robots. (Top row) Feeding and drinking assistance. (Bottom row) Itch scratching and bed bathing assistance.

with activities of daily living (ADLs) [44], including:

- **Feeding**: A robot holds a spoon and aims to move the spoon to a person's mouth. We place small spheres on the spoon, representing food. The person sits in a wheelchair during assistance.
- **Drinking Water**: A robot holds a cup containing small particles that represent water. The robot aims to move the cup toward the person's mouth and tilt the cup to help the person drink the water. The person sits in a wheelchair during assistance.
- **Itch Scratching**: A robot aims to scratch a randomly generated location along the person's right arm. The robot grasps a scratching tool in its left end effector. The person sits in a wheelchair during assistance.
- **Bed Bathing**: The robot holds a simulated washcloth tool while the person lies on a hospital bed in a randomly generated resting pose. The robot aims to move the washcloth around the person's right arm in order to clean the arm.

These four assistive environments in AVR Gym are nearly identical to the original environments presented in [3], including the same male and female human models, realistic human joint limit models, robot base pose optimization, reward functions with human preferences, and task goals.

CHAPTER 4

SIMULATION-TRAINED ROBOT CONTROL POLICIES

Prior to developing AVR Gym, we trained eight robot control policies solely in simulation for the four caregiving tasks (robot-assisted feeding, drinking, itch scratching, and bed bathing) and two collaborative robots (the PR2 from Willow Garage and the 7-DoF Jaco (Gen2) arm from Kinova). These policies were the baseline policies released with Assistive Gym and described in the corresponding paper [3]. We slightly modified the policies to run in AVR Gym, and trained the modified version of policies, which we refer to as the *Original Policies*.

4.1 Original Policies

For each assistive task and robot, we follow the same training procedure presented in [3]. At each time step during simulation, the robot executes an action and then receives a reward and an observation based on the state of the world. Actions for a robot's 7-DoF arm are represented as changes in joint angles, $\Delta P \in \mathbb{R}^7$. The PR2 uses only its right or left arm depending on the assistive task. The observations for a robot include the 7D joint angles of the robot's arm, the 3D position and orientation of the end effector, and the forces applied at the end effector. The robot's observation also includes details of task-relevant human joints, including 3D positions of the shoulder, wrist, and elbow during the bed bathing and itch scratching assistive tasks, and the position and orientation of the person's head during feeding and drinking assistance. Both the Jaco and PR2 robots use the same observation and reward functions during training.

Our Original policies were trained using proximal policy optimization (PPO) [45], which is an actor-critic deep reinforcement learning approach that has recently shown success in assistive robotic contexts [46, 28, 3]. These policies are modeled using a fullyconnected neural network with two hidden layers of 64 nodes and tanh activations. We train policies for 50,000 simulation rollouts (trials), where each rollout consists of 200 consecutive time steps (20 seconds of simulation time at 10 time steps per second). Prior to each simulation rollout, we randomly initialize the simulated person's arm pose for itch scratching and bed bathing tasks, and head orientation for feeding and drinking tasks. Once initialized, the human holds a static pose throughout the entire rollout. Each policy is trained with default male and female human models, with heights, body sizes, weights, and joint limits matching published 50th percentile values [47]. Note that these control policies are not temporal models and do not consider observations from previous time steps in a rollout. As such, these models are not affected by the specific nature of human motion, allowing us to evaluate these policies with real people without needing to model realistic human motion in simulation.

4.2 Revised Policies

During early pilot studies in AVR Gym with lab members, we observed that the Original policies exhibited unexpected deficiencies and poor performance when providing assistance to human participants in VR. Most notable were failures in the itch scratching and bed bathing assistance tasks, where the robots would fail to move their end effectors closer to the person's body. Through iterative investigation and development with real people in AVR Gym, we developed the *Revised Policies*, based on the key discovery that the biomechanical models of simulated people we used to train the Original policies significantly differed from the biomechanics of real people. The two leading factors were variation among human heights and waist bending movements, wherein the Original policies were trained on male and female models with fixed heights and were not trained on any variation among the human waist joints.

Given these findings, we modified each simulation environment such that the simulated human biomechanics better match people in VR. The summarized Python code below shows the major modification.

```
# randomize the simulated human torso height
torso_height = random.uniform(0.6 - 0.1, 0.6 + 0.1) if gender == male else random.uniform(0.54 - 0.1, 0.54 + 0.1)

# randomize the simulated person's initial waist joints to angles within the range of -10 to 10.
joints_positions = [(j, random.uniform(-10, 10)) for j in human.waist_joints]
setup_human_joints(joints_positions)
```

Listing 4.1: summarized Python code for the modification

As shown in Listing 4.1, we randomized the simulated human torso height and initial waist orientation before each training trial began. The person's torso height, measured from hipbone to center of the head, was uniformly sampled from 50 cm to 70 cm for male models and 44 cm to 64 cm for female models. We then randomized the simulated person's initial three kinematic waist joints to angles within the range of (-10°, 10°). With these improved biomechanical simulations, we trained a new set of eight *Revised Policies*, one for each robot and assistive task, using the same training process discussed in section 4.1.

When evaluated with simulation humans, these Revised policies achieve similar rewards and task success rates as the Original policies. However, as confirmed by our formal study (chapter 5), the Revised policies overcame the limitations of the Original policies and performed significantly better across both subjective and objective metrics when evaluated with real people in VR.

CHAPTER 5
FORMAL STUDY

We conducted a formal study with eight participants to evaluate the performance of the Original policies and the Revised policies in terms of objective and subjective measures. In addition, we performed a posthoc analysis of biomechanical differences between the simulated humans used to train the Original policies, the simulated humans used to train the Revised policies, and the real humans who participated in our study.

5.1 Experimental Procedure

We conducted an experiment with eight able-bodied human participants (four females and four males). We obtained informed consent from all participants and approval from the Georgia Institute of Technology Institutional Review Board (IRB). We recruited participants to meet the following inclusion/exclusion criteria: 18 years of age or older; able-bodied; no cognitive or visual impairments; fluent in spoken and written English; and not diagnosed with epilepsy. Participant ages ranged from 19 to 24 years old with torso heights varied between 0.51 and 0.58 meters.

For each participant, we conducted a total of 16 trials in VR with four trials for each of the four assistive tasks. The four trials for each task were organized such that we evaluated both the Original and Revised policies on both the PR2 and Jaco robots. We randomized the ordering of assistive tasks, robots, and control policies for each participant according to a randomized counterbalancing design.

During the study, participants wore the VR headset and held both controllers. Participants sat in a wheelchair for the itch scratching, feeding, and drinking tasks and laid in a hospital bed for bed bathing assistance. Similar to the simulation environments used for training control policies, each trial in VR lasted 20 seconds for a total of 200 time steps.

The robot executed an action at each time step in VR, once every 0.1 seconds. Participants were instructed that they could move their arms, upper body, and head to interact with the simulated robots towards the goal of successfully receiving assistance for each task. Prior to each assistive task, we gave participants an unscripted practice trial to familiarize themselves on interacting with the robot and how to accomplish the task in VR. We randomly selected either the PR2 or Jaco robot and used the Revised control policies (see section 4.2) for each practice trial. In order to elicit a wide range of human motion and interactions throughout our study, we instructed participants to accomplish a task (e.g. drink water from the cup), but we did not provide instructions on how to interact with the robot or how to appropriately complete the task.

5.2 Objective Measures

We used the reward functions and success percentages defined in Assistive Gym as objective measures of performance [3].

- **Feeding**: The robot is rewarded for moving the spoon closer to the person's mouth and when food enters the person's mouth. The robot is penalized for dropping food or applying large forces to the person. Task success is defined by the robot feeding at least 75% of all food particles to the person's mouth.
- **Drinking Water**: The robot is rewarded for moving the cup closer to the person's mouth, tilting the cup for drinking, and when water particles enter the person's mouth. The robot is penalized for spilling water or applying large forces to the person. Task success is defined by the robot pouring at least 75% of all water into the person's mouth.
- **Itch Scratching**: The robot is rewarded for moving the scratching tool closer to the itch location and for performing scratching motions around the itch. The robot is penalized for applying large forces to the person, or more than 10 N of force [26] near the itch.

Task success is defined by the robot performing at least 25 scratching motions near the itch.

- **Bed Bathing**: The robot is rewarded for moving the washcloth closer to the person's arm and for using the bottom of the washcloth to wipe off markers uniformly distributed (3 cm apart) along the person's right arm. The robot is penalized for applying large forces to the person. Task success is defined by the robot wiping off at least 30% of all markers along a person's arm)

5.3 Subjective Measures

In order to assess participants' perceptions of the control policies, we used a questionnaire with four statements pertaining to perceptions of the robot's performance, safety, comfort and speed. For each statement, participants were asked to record how much they agreed with the statement on an interval scale from 1 (strongly disagree) to 7 (strongly agree) with 4 being neutral. We based this 7-point scale questionnaire on past work on robot-assisted feeding [48].

The four statements follow:

- L1: The robot successfully assisted me with the task.
- L2: I felt comfortable with how the robot assisted me in VR.
- L3: I felt I would be safe if this were a real robot.
- L4: The robot moved with appropriate speed.

5.4 Experimental Results

5.4.1 Performance of the Original Policies

Table 5.1 depicts the average reward and task success when both robots used the Original policies in simulation with static human models and in AVR Gym with the eight human

Table 5.1: Average reward and task success between simulation and VR using the Original control policies.

Task	Simulation		VR	
	PR2	Jaco	PR2	Jaco
Feeding	103 (77%)	82 (75%)	122 (100%)	37 (63%)
Drinking	380 (59%)	144 (32%)	502 (75%)	-92 (0%)
Scratching	62 (35%)	218 (48%)	25 (25%)	-69 (0%)
Bathing	85 (13%)	118 (28%)	-87 (0%)	-126 (0%)
Avg. Success	46%	46%	50%	16%

participants.

When evaluating the Original policy for a given robot and assistive task in VR, we averaged rewards and task success over trials from all eight participants. For simulation, we averaged rewards and task success over 100 simulation rollouts with a random initial human pose for each rollout. Since the simulated human holds a static pose throughout an entire trial, we take an average over a larger number of simulation trials to evaluate performance over multiple human poses. We note that each task uses a slightly different reward function due to task-specific reward elements and hence reward values are not directly comparable across different tasks.

For the feeding assistance task, the Original policies achieved similar average reward and task success with real people as they achieved in simulation with fixed human models. However, performance was more varied for the drinking, itch scratching, and bed bathing tasks. The most noticeable errors occurred with the itch scratching and bed bathing tasks, where the Original policies for both the Jaco and PR2 robots frequently failed to move their end effectors closer to a person's body. These errors can be visually seen in Figure 5.1 where the PR2 actively moved away from the person during bed bathing assistance and in Figure 5.2 where the Jaco robot actuated itself to behind the wheelchair rather than scratching an itch on the person's arm. Differences in results between the two robots can be partially attributed to differences in robot kinematics and base positioning. The Jaco

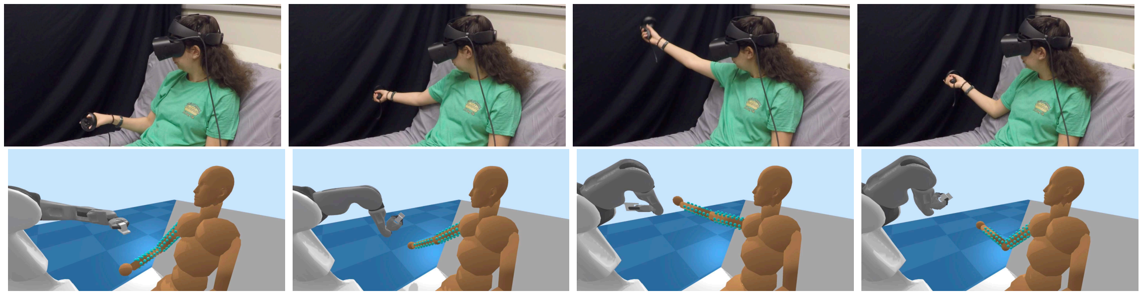

Figure 5.1: The PR2 robot fails to provide assistance with bed bathing in VR when using the Original control policy. Blue markers are placed uniformly around the person's right arm, which the robot can clean off with the bottom of the wiping tool.

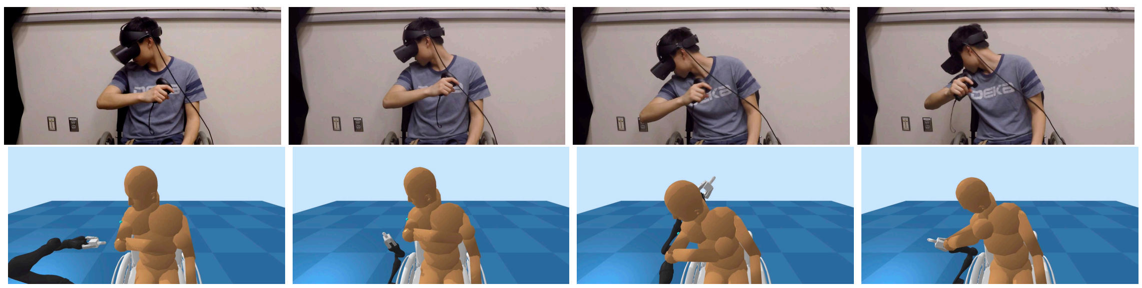

Figure 5.2: The Jaco fails to provide itch scratching assistance in VR with the Original control policy. The itch is represented by a blue marker.

is mounted to a fixed position on the wheelchair, whereas we optimize the PR2's base pose near a person according to joint-limit-weighted kinematic isotropy (JLWKI) and task-centric manipulability metrics [3, 49].

Except for responses about the robot's speed (L4), participants' average responses were negative or neutral (L1, L2, and L3) (see Figure 5.5). Most notably, participants tended to perceive the robots as being unsuccessful at assistance, as evidenced by responses to L1 being significantly below neutral (4) with $p < 0.05$ using a Wilcoxon signed-rank test.

Responses did vary by task. When the robots used the Original policies for itch scratching assistance, participants slightly disagreed with the statement on successful assistance (L1), with an average response of 3.3 across all participants. When using the Original policies for bed bathing assistance, participants reported an even lower average rating of 1.2, where almost all participants strongly disagreed that the robots provided successful bathing assistance.

Table 5.2: Average reward and task success for the Original and Revised control policies in VR with real people.

Task	Original Policies		Revised Policies	
	PR2	Jaco	PR2	Jaco
Feeding	122 (100%)	37 (63%)	113 (100%)	36 (45%)
Drinking	502 (75%)	-92 (0%)	458 (75%)	199 (42%)
Scratching	25 (25%)	-69 (0%)	36 (45%)	95 (62%)
Bathing	-87 (0%)	-126 (0%)	-18 (0%)	-16 (0%)
Avg. Success	50%	16%	**55%**	**37%**

5.4.2 Performance of the Revised Policies

Table 5.2 compares the average reward and task success between the Original and Revised control policies during our VR study. For a given task and robot, we average the rewards and task success rates over trials from all eight participants.

From the table, we observe that the Revised policies for both robots outperformed the Original policies when providing itch scratching and bed bathing assistance. Figure 5.3 demonstrates the Jaco robot using a Revised policy to provide bed bathing assistance to a participant in VR. While the Original policies for the Jaco robot failed to reliably assist human participants with drinking, we observe that the Revised policies for the Jaco exhibited a greater level of success. Figure 5.4 depicts an image sequence of the Jaco robot using a Revised control policy to provide drinking assistance. Based on these success metrics and qualitative image sequences, we find that the Revised polices exhibited reasonable performance for all of the assistive tasks.

For all four statements (L1, L2, L3, and L4), participants' average responses were positive or neutral (see Figure 5.5). Most notably, participants tended to perceive the robots as being successful at assistance, as evidenced by responses to L1 being significantly above neutral (4) with $p < 0.001$ using a Wilcoxon signed-rank test.

Figure 5.5 displays a comparison of questionnaire responses between the Original and Revised policies when responses are averaged over all participants, tasks, and robots for

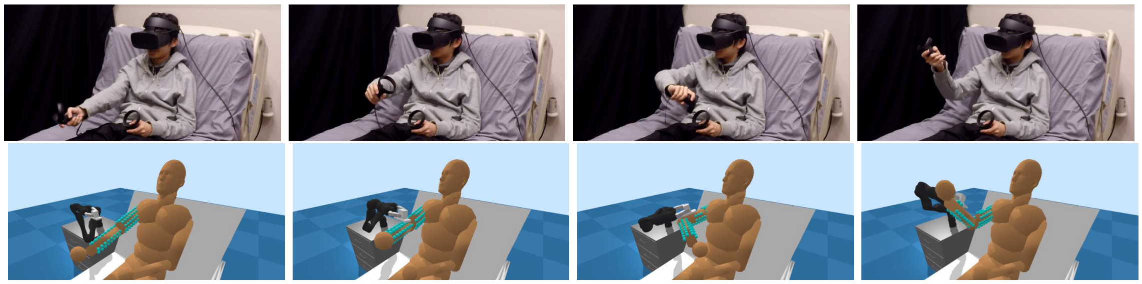

Figure 5.3: Image sequence of a table-mounted Jaco providing bed bathing assistance using the Revised control policy.

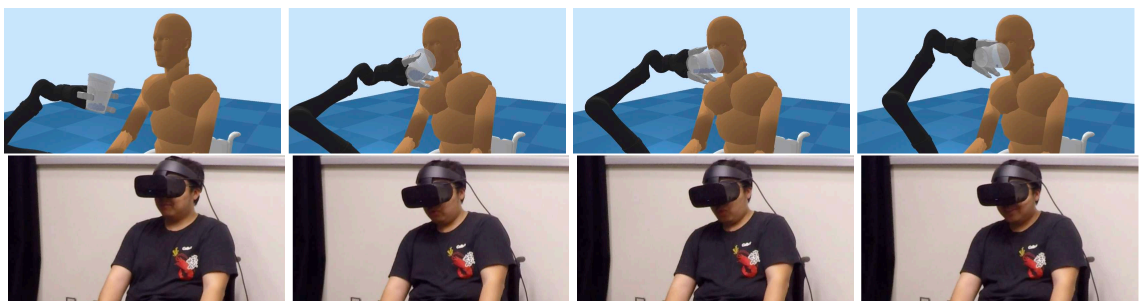

Figure 5.4: Image sequence of drinking assistance with the Jaco and the Revised control policy.

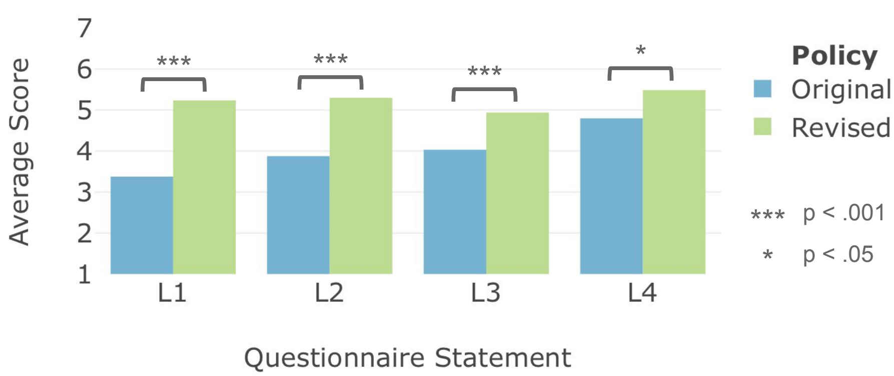

Figure 5.5: Comparison of 7-point scale questionnaire responses for both Original and Revised policies. Responses are averaged over all participants, tasks, and robots. p-values are computed with a Wilcoxon signed-rank test.

each questionnaire statement. On average, participants provide higher responses for trials in which a robot used a Revised policy to provide assistance. To determine a statistical difference between the Original and Revised policies, we apply a Wilcoxon signed-rank test, a non-parametric test, between participant responses from trials using the Original control policies and responses when using the Revised policies. The computed p-values are depicted in Figure 5.5, where we observed a statistically significant difference at the $p < .001$ level for the first three questions relating to task success, comfort, and safety. These results indicate that participants perceived a noticeable improvement in a robot's performance and safety when using Revised control policies that were trained based on biomechanics feedback from VR.

Overall, we found that that our Revised policies significantly outperformed the Original policies, which indicates that VR can be used to efficiently improve simulation-trained controllers for physical human-robot interaction.

5.4.3 Posthoc Biomechanical Analysis

Our iterative research and development using AVR Gym, discussed in section 4.2, resulted in Revised policies trained with new distributions of human torso heights and waist orientations. Here, we provide a posthoc statistical analysis of these biomechanical parameters for our simulated humans and our real participants in VR.

During our human study, we aligned the human model in VR with a real participant's pose, as described in chapter 3. This alignment enabled us to record the full state, height, and pose of a participant at each time step during the VR trials.

To determine whether a statistical difference exists between human torso heights, we applied a Wilcoxon signed-rank test between human torso heights in the simulation environments used to train the Original policies and torso heights of human participants in VR. This test returned a p-value of $p < 0.001$, indicating that the original simulation environments did not properly model the distribution of real human heights observed during VR.

We then applied the same test to a set of torso heights from the updated simulation environments used to train the Revised policies. When compared to torso heights of human participants, the test returns a p-value of $p = 0.29$, which indicates that there is a less significant difference between the distributions of human heights used to train the Revised control policies and human heights observed in VR.

In order to evaluate differences in waist orientations between simulation and VR, we consider the 3D position of a person's head, averaged over time. Since the human model in VR is aligned with a real participant's pose, we track the 3D head position of a participant by querying the 3D head position of the virtual human model. In order to separate biomechanical differences in torso height from waist orientations, we modified the original simulation environments to include random variation among human torso heights, but no variation among waist orientations. Using a Wilcoxon signed-rank test, we found a statistically significant difference in head position (averaged over all time steps for a trial) between data from these new simulation environments and the head positions of participants in VR, with a p-value of $p < 0.01$. However, by performing this same test between head positions from the revised simulation environments (section 4.2) and from VR, we found a p-value of $p = 0.26$, indicating that the improved simulation environments used to train our Revised policies better matched the biomechanics of our participants in VR.

CHAPTER 6
DISCUSSION AND LIMITATIONS

6.1 Trained control policies

As shown in Table 5.2, we noted that even the revised controllers can not perfectly assist human participants with tasks and still exhibit some errors. This is especially true for hard to reach locations, such as scratching an itch underneath a person's arm. On the other hand, these control policies also assume that the person receiving assistance acts collaboratively with the robot. Due to the uncertain and unstructured environment, however, human may appear uncooperative or irrational. We observed that these control policies sometimes acted wildly and might put people at risk in the real world. Therefore, additional research is required to explore how robot controllers learned in simulation can act appropriately when exposed to uncooperative or irrational human motion.

6.2 Transfer simulation-trained robot policies to real world

Our current work has shown that VR can serve as a stepping stone to real-world robotics assistance. There are still essential challenges remaining that prevent safely and effectively transferring learned robot capabilities to real robots. One of the big challenges is the generalisation from a deterministic training environment to a noisy and uncertain testing environment for example in form of real sensor noise. Future work can benchmark how much observation noise standard robot control policies can handle when transferred to real robots. Furthermore, we could explore various generalisation techniques, such as machine learning-based domain adaptation and randomization of simulated environments, to improve the robustness of robot control policies to noise.

CHAPTER 7
CONCLUSION

We presented Assistive VR Gym (AVR Gym), which enables real people to interact with virtual assistive robots. We also provided evidence that AVR Gym can help researchers improve the performance of simulation-trained assistive robots. Overall, our results suggest that VR can be used as a stepping stone between physics simulations and the real world by enabling real people to interact with virtual robots.

Appendices

APPENDIX A
AVR GYM INSTALLATION AND TESTING

A.1 Install

A.1.1 Setup

AVR Gym is only supported on Windows due to the VR dependency.

- Install Python 3.6.8
- Install git
- Install Microsoft visual c++ build tools
- Install c++ build tools for visual studio. During installation, click on the ”C++ build tools” checkbox
- Download Oculus Rift S Software and install SteamVR

A.1.2 Install AVR Gym

- Setup Bullet:

```
mkdir git
cd git
python -m pip install --user virtualenv
python -m venv env
env\Scripts\activate
pip install --upgrade setuptools
git clone -b vr https://github.com/Zackory/bullet3.git
cd bullet3
pip install .
```

Navigate to the bullet3 directory, click on `build_visual_studio_vr_pybullet_double.bat` and open `build3/vs2010/0MySolution.sln`. When asked, convert the projects to a newer version of Visual Studio. If you installed Python in the C:\root directory, the batch file should find it automatically. Otherwise, edit this batch file to choose where Python include/lib directories are located. Build `App_SharedMemoryPhysics_VR` project in Release/optimized build.

- Install AVR Gym and all the required dependencies:

```
git clone https://github.com/Healthcare-Robotics/assistive-vr-gym.git
cd assistive-vr-gym
pip install .

```

A.2 Testing AVR Gym with Policies

A.2.1 Install Reinforcement Learning Library

```
# Install pytorch RL library
pip install git+https://github.com/Zackory/pytorch-a2c-ppo-acktr --no-cache-dir
# Install OpenAI Baselines 0.1.6
pip install git+https://github.com/openai/baselines.git
```

A.2.2 Train Policies for Testing

AVR Gym provides 4 assistive tasks (ScratchItch, BedBathing, Feeding, Drinking), 2 commercial robots (PR2, Jaco) and 3 human states for training (Static, Active or New (Revised Biomechanics)):

- Static human environment names: `[task][robot]-v0`

- Active human environment names: `[task][robot]Human-v0`
- New human environment names: `[task][robot]New-v0`

You can train the control policies for new human environment using the command below:

```
python -m ppo.train --env [Train env name] --num-env-steps 10000000 --
    save-dir ./trained_models/
```

A.2.3 Robot assisting a static person in VR

Now we are going to evaluate our pretrained policy. AVR Gym provides three states for VR testing: Static VR, Active VR or New VR (Revised Biomechanics):

- Static VR human environment names: `[task]VR[robot]-v0`
- Active VR human environment names: `[task]VR[robot]Human-v0`
- New VR human environment names: `[task]VR[robot]New-v0`

To launch the simulated environment in VR, and align the simulated human model to your real body pose as determined by your Oculus Rift S, you can run the script using the command below:

```
python enjoy_vr_trial.py --gender [gender] --vr-env [VR env name] --env
    [Train env name]
```

The `enjoy_vr_trial.py` script will also record the entire VR trial (robot and human state at each time step), which can then be replayed.

A.2.4 Replay a recorded trial from VR

You can replay the entire interaction that occurred in VR using the command below:

```
python replay_vr.py --env [Train env name] --replay-dir <fill in replay
    directory here>
```

The `--replay-dir` argument should be set to the directory created from `enjoy_vr_trial.py`.

REFERENCES

[1] D. M. Taylor, "Americans with disabilities: 2014," *United States Census Bureau*, 2018.

[2] S. for Enhancing Geriatric Understanding, E. A. Surgical, and A. G. S. Medical Specialists (SEGUE), "Retooling for an aging america: Building the healthcare workforce," *Journal of the American Geriatrics Society*, vol. 59, no. 8, pp. 1537–1539, 2011.

[3] Z. Erickson, V. Gangaram, A. Kapusta, C. K. Liu, and C. C. Kemp, "Assistive gym: A physics simulation framework for assistive robotics," *IEEE International Conference on Robotics and Automation (ICRA)*, 2020.

[4] Z. Erickson, Y. Gu, and C. C. Kemp, "Assistive vr gym: Interactions with real people to improve virtual assistive robots," in *2020 29th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*, IEEE, 2020, pp. 299–306.

[5] F. F. Goldau, T. K. Shastha, M. Kyrarini, and A. Gräser, "Autonomous multi-sensory robotic assistant for a drinking task," in *2019 IEEE 16th International Conference on Rehabilitation Robotics (ICORR)*, 2019.

[6] S. Schröer, I. Killmann, B. Frank, M. Völker, L. Fiederer, T. Ball, and W. Burgard, "An autonomous robotic assistant for drinking," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2015, pp. 6482–6487.

[7] D. Park, Y. K. Kim, Z. M. Erickson, and C. C. Kemp, "Towards assistive feeding with a general-purpose mobile manipulator," *arXiv preprint arXiv:1605.07996*, 2016.

[8] G. Canal, G. Alenyà, and C. Torras, "Personalization framework for adaptive robotic feeding assistance," in *International Conference on Social Robotics*, Springer, 2016, pp. 22–31.

[9] C. J. Perera, T. D. Lalitharatne, and K. Kiguchi, "Eeg-controlled meal assistance robot with camera-based automatic mouth position tracking and mouth open detection," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2017, pp. 1760–1765.

[10] D. Park, Y. Hoshi, and C. C. Kemp, "A multimodal anomaly detector for robot-assisted feeding using an lstm-based variational autoencoder," *IEEE Robotics and Automation Letters*, 2018.

[11] T. Rhodes and M. Veloso, “Robot-driven trajectory improvement for feeding tasks,” in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2018, pp. 2991–2996.

[12] T. Bhattacharjee, E. K. Gordon, R. Scalise, M. E. Cabrera, A. Caspi, M. Cakmak, and S. S. Srinivasa, “Is more autonomy always better? exploring preferences of users with mobility impairments in robot-assisted feeding,” in *Proceedings of the 2020 ACM/IEEE International Conference on Human-Robot Interaction*, 2020, pp. 181–190.

[13] C.-H. King, T. L. Chen, A. Jain, and C. C. Kemp, “Towards an assistive robot that autonomously performs bed baths for patient hygiene,” in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, 2010, pp. 319–324.

[14] Z. Erickson, H. M. Clever, V. Gangaram, G. Turk, C. K. Liu, and C. C. Kemp, “Multidimensional capacitive sensing for robot-assisted dressing and bathing,” in *2019 IEEE 16th International Conference on Rehabilitation Robotics (ICORR)*, IEEE, 2019, pp. 224–231.

[15] S. D. Klee, B. Q. Ferreira, R. Silva, J. P. Costeira, F. S. Melo, and M. Veloso, “Personalized assistance for dressing users,” in *International Conference on Social Robotics*, Springer, 2015, pp. 359–369.

[16] R. P. Joshi, N. Koganti, and T. Shibata, “Robotic cloth manipulation for clothing assistance task using dynamic movement primitives,” in *Proceedings of the Advances in Robotics*, 2017, pp. 1–6.

[17] G. Canal, E. Pignat, G. Alenyà, S. Calinon, and C. Torras, “Joining high-level symbolic planning with low-level motion primitives in adaptive hri: Application to dressing assistance,” in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2018.

[18] A. Jevtić, A. F. Valle, G. Alenyà, G. Chance, P. Caleb-Solly, S. Dogramadzi, and C. Torras, “Personalized robot assistant for support in dressing,” *IEEE transactions on cognitive and developmental systems*, vol. 11, no. 3, pp. 363–374, 2018.

[19] Z. Erickson, M. Collier, A. Kapusta, and C. C. Kemp, “Tracking human pose during robot-assisted dressing using single-axis capacitive proximity sensing,” *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 2245–2252, 2018.

[20] G. Chance, A. Jevtić, P. Caleb-Solly, G. Alenya, C. Torras, and S. Dogramadzi, “ elbows out”—predictive tracking of partially occluded pose for robot-assisted dressing,” *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3598–3605, 2018.

[21] E. Pignat and S. Calinon, “Learning adaptive dressing assistance from human demonstration,” *Robotics and Autonomous Systems*, 2017.

[22] N. Koganti, T. Tamei, K. Ikeda, and T. Shibata, “Bayesian nonparametric motor-skill representations for efficient learning of robotic clothing assistance,” in *Workshop on Practical Bayesian Nonparametrics, Neural Information Processing Systems, 2016*, 2016, pp. 1–5.

[23] Y. Gao, H. J. Chang, and Y. Demiris, “Iterative path optimisation for personalised dressing assistance using vision and force information,” in *2016 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, IEEE, 2016, pp. 4398–4403.

[24] Z. Erickson, A. Clegg, W. Yu, G. Turk, C. K. Liu, and C. C. Kemp, “What does the person feel? learning to infer applied forces during robot-assisted dressing,” in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2017, pp. 6058–6065.

[25] W. Yu, A. Kapusta, J. Tan, C. C. Kemp, G. Turk, and C. K. Liu, “Haptic simulation for robot-assisted dressing,” in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2017, pp. 6044–6051.

[26] Z. Erickson, H. M. Clever, G. Turk, C. K. Liu, and C. C. Kemp, “Deep haptic model predictive control for robot-assisted dressing,” in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2018, pp. 1–8.

[27] A. Kapusta, Z. Erickson, H. M. Clever, W. Yu, C. K. Liu, G. Turk, and C. C. Kemp, “Personalized collaborative plans for robot-assisted dressing via optimization and simulation,” *Autonomous Robots*, 2019.

[28] A. Clegg, Z. Erickson, P. Grady, G. Turk, C. C. Kemp, and C. K. Liu, “Learning to collaborate from simulation for robot-assisted dressing,” *IEEE Robotics and Automation Letters*, 2020.

[29] R. M. Baños, C. Botella, M. Alcañiz, V. Liaño, B. Guerrero, and B. Rey, “Immersion and emotion: Their impact on the sense of presence,” *Cyberpsychology & behavior*, vol. 7, no. 6, pp. 734–741, 2004.

[30] K. Kilteni, R. Groten, and M. Slater, “The sense of embodiment in virtual reality,” *Presence: Teleoperators and Virtual Environments*, vol. 21, no. 4, pp. 373–387, 2012.

[31] L. Takayama and C. Pantofaru, “Influences on proxemic behaviors in human-robot interaction,” in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, 2009, pp. 5495–5502.

[32] J. Wainer, D. J. Feil-Seifer, D. A. Shell, and M. J. Mataric, "The role of physical embodiment in human-robot interaction," in *ROMAN 2006-The 15th IEEE International Symposium on Robot and Human Interactive Communication*, IEEE, 2006, pp. 117–122.

[33] V. Kumar and E. Todorov, "Mujoco haptix: A virtual reality system for hand manipulation," in *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, IEEE, 2015, pp. 657–663.

[34] J. I. Lipton, A. J. Fay, and D. Rus, "Baxter's homunculus: Virtual reality spaces for teleoperation in manufacturing," *IEEE Robotics and Automation Letters*, vol. 3, no. 1, pp. 179–186, 2017.

[35] V. Kumar, A. Gupta, E. Todorov, and S. Levine, "Learning dexterous manipulation policies from experience and imitation," *arXiv preprint arXiv:1611.05095*, 2016.

[36] T. Zhang, Z. McCarthy, O. Jow, D. Lee, X. Chen, K. Goldberg, and P. Abbeel, "Deep imitation learning for complex manipulation tasks from virtual reality teleoperation," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2018.

[37] H. Liu, Z. Zhang, X. Xie, Y. Zhu, Y. Liu, Y. Wang, and S.-C. Zhu, "High-fidelity grasping in virtual reality using a glove-based system," in *2019 International Conference on Robotics and Automation (ICRA)*, IEEE, 2019, pp. 5180–5186.

[38] A. Frisoli, M. Bergamasco, M. C. Carboncini, and B. Rossi, "Robotic assisted rehabilitation in virtual reality with the l-exos," *Stud Health Technol Inform*, vol. 145, pp. 40–54, 2009.

[39] K. Brütsch, A. Koenig, L. Zimmerli, S. Mérillat-Koeneke, R. Riener, L. Jäncke, H. J. van Hedel, and A. Meyer-Heim, "Virtual reality for enhancement of robot-assisted gait training in children with neurological gait disorders," *Journal of rehabilitation medicine*, 2011.

[40] B. R. Ballester, L. S. Oliva, A. Duff, and P. Verschure, "Accelerating motor adaptation by virtual reality based modulation of error memories," in *2015 IEEE International Conference on Rehabilitation Robotics (ICORR)*, IEEE, 2015, pp. 623–629.

[41] X. Huang, F. Naghdy, G. Naghdy, H. Du, and C. Todd, "The combined effects of adaptive control and virtual reality on robot-assisted fine hand motion rehabilitation in chronic stroke patients: A case study," *Journal of Stroke and Cerebrovascular Diseases*, 2018.

[42] F. Bernardoni, Ö. Özen, K. Buetler, and L. Marchal-Crespo, "Virtual reality environments and haptic strategies to enhance implicit learning and motivation in robot-

assisted training," in *2019 IEEE 16th International Conference on Rehabilitation Robotics (ICORR)*, IEEE, 2019.

[43] E. Coumans and Y. Bai, "Pybullet, a python module for physics simulation for games, robotics and machine learning," *http://pybullet. org*, 2016–2019.

[44] M. P. Lawton and E. M. Brody, "Assessment of older people: Self-maintaining and instrumental activities of daily living," *The gerontologist*, vol. 9, no. 3_Part_1, pp. 179–186, 1969.

[45] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[46] A. Clegg, "Modeling human and robot behavior during dressing tasks," Ph.D. dissertation, The Georgia Institute of Technology, 2019.

[47] A. R. Tilley, *The measure of man and woman: human factors in design*. John Wiley & Sons, 2002.

[48] D. Park, Y. Hoshi, H. P. Mahajan, H. K. Kim, Z. Erickson, W. A. Rogers, and C. C. Kemp, "Active robot-assisted feeding with a general-purpose mobile manipulator: Design, evaluation, and lessons learned," *Robotics and Autonomous Systems*, vol. 124, p. 103 344, 2020.

[49] A. Kapusta and C. C. Kemp, "Task-centric optimization of configurations for assistive robots," *Autonomous Robots*, vol. 43, no. 8, pp. 2033–2054, 2019.