DOI: 10.1109/ARSO60199.2024.10557907

Programming Robot Animation Through Human Body Movement

Helena Anna Frijns[1], Darja Stoeva[2], Oliver Schürer[3], and Margrit Gelautz[2]

***Abstract*— The aim of our work is to make it easier for non-programmers to program robot motion. We designed a system that translates human motion to motion of a (virtual and physical) humanoid Pepper robot. The system enables programming robot animations using pose-matching imitation of human motion. We developed a prototype that implements recording functionality, evaluated the prototype, and revised the system architecture based on feedback from participants with expertise in dance or programming the Pepper robot. We extend a conceptual design space for computer animation by including a physical robot. On the basis of this design space and the prototype development, we describe interaction design considerations for robot animation systems that implement human-humanoid imitation systems.**

I. Introduction

Human-Robot Interaction (HRI) is a multidisciplinary research field, necessitated by the complexity of developing interactions with robotic systems taking place in human social space. It has been argued that HRI research has thus far been dominated by laboratory studies with simplified views on social interaction [1], [2], [3]. Inclusion of domain experts on social and nonverbal behavior such as interaction designers, social scientists [2], [4], and expert dancers [5] has been proposed as promising for developing robotics applications for social settings [2], [4]. Researchers have advocated for inclusion of animators and dancers in robot motion design, due to their expertise on making characters seem lifelike through motion [6], [7], audience perception, and human movement features [8]. Sirkin and Ju [9] argue that embodied design improvisation can help designers of physical interactions with everyday objects (including robots) bring out tacit knowledge, that is, knowledge arising from the body and its situated interactions. However, experts with knowledge required to develop social interactions may lack robotics and programming expertise [2], [4].

To enable robot programming by domain experts such as dancers, systems and input methods with a high ease of use are beneficial. Our focus is on systems that produce robot motion for social HRI and dance, using imitation of human motion as a programming tool. We developed a prototype for animating robot motion that implements a human-humanoid pose-matching imitation system, as described in Sec. III. The prototype uses Kinect v2 to realize imitation of human motion by the Pepper robot, with a graphical user interface (GUI) to start and stop the motion recording process. An evaluation of the prototype led to a revision of the system architecture. In Sec. IV, we outline interaction design considerations for robot animation systems, especially in relation to the development of our prototype. We extend a design space for computer animation [10] for robot animation. The contributions of this paper are this extended design space and interaction design considerations for human-humanoid pose-matching imitation systems, along with an implementation and evaluation of a prototype.

[1]H.A. Frijns is with the Institute of Management Science, Faculty of Mechanical and Industrial Engineering, TU Wien, 1040 Vienna, Austria helena.frijns@tuwien.ac.at

[2]D. Stoeva and M. Gelautz are with the Institute of Visual Computing and Human-Centered Technology, Faculty of Informatics, TU Wien, 1040 Vienna, Austria {darja.stoeva, margrit.gelautz}@tuwien.ac.at

[3]O. Schürer is with the Institute of Architectural Sciences, Faculty of Architecture and Planning, TU Wien, 1040 Vienna, Austria schuerer@attp.tuwien.ac.at

II. Related work

Reasons for animation of robot motion include supporting human-robot communication, generating convincing behavior, suggesting emotion, animacy [11], or personality [12], expressing information such as internal state, intentions, and attitudes, or coordinating joint activity [13]. Our focus is on expressive gestural motion. Expressive motion serves purposes such as communication, and its meaning is influenced by contextual factors [14]. Gestural motion involves gestures and behaviors that indicate a robot's state and convey information, and can be performed by executing a series of poses [15]. Techniques for robot motion design include 3D animation studies, developing a skeleton prototype of the robot, Wizard-of-Oz (WoZ) exploration of movement possibilities, and video prototyping [13]. Tools for realizing expressive robot motion need to guarantee consistent playback, safety, and scalability, and balance the complexity of information presentation while enabling access to useful information [15]. Saerbeck and van Breemen [16] distinguish three classes of robot motion design methods: trajectory design methods, motion editing methods, and high-level behavior design methods. Examples of input methods for robot motion with high ease of use can be found in the domains of Programming by Demonstration (PbD) (also referred to as Learning from Demonstration (LfD) or imitation learning [17]) and robot teleoperation (e.g., [18], [19]). PbD is a technique in which a human shows a robot example behaviors [20] that are used to generate a robot program. Bravo et al. [20] describe PbD strategies: manipulation of a robot body (kinaesthetic teaching or teleoperation), manipulation of a physical object or a (virtual or physical) robot representation, demonstration using a GUI or user body movements, and multimodal demonstrations. Similarly, input methods for LfD include kinaesthetic demonstrations, teleoperation, and passive observation [17]. These methods differ in terms of their ease of use, possibility of application

to systems with a high number of DOF (degrees of freedom), and ease of mapping demonstrations to the robot.

Motion imitation systems using Kinect have been developed for the NAO [21], [22] and Pepper [23], [24], [25]. Our work differs from this, as our focus is not on the imitation, but rather on the interaction design of a system for robot animation that implements imitation. Several authors describe choreographing or designing robot motion with a GUI, e.g., for the HRP-4C robot [26], enabling dancers to give feedback to CoBot robot motion [7], or observing how animated motions combine with procedural layers [12]. Our work differs from GUIs for robot animation, as we focus on the interaction design of systems that are distributed in space, involving, besides a GUI, a humanoid robot, sensor(s) for detecting human joint positions, and an end user controlling the robot with body movement. Depth sensors such as Kinect have been used for computer animation of virtual characters [27], [28], [29]. Our work focuses on motion for a physical, co-located robot instead, which has consequences for the spatial setup of the system and how user attention is distributed across this space. Several works implement puppeteering a robot using a GUI, e.g., for the desk-light shaped AUR robot [30], the customizable Blossom robot [31], or a WoZ interface for controlling Pepper [32]. Balit et al. [6] propose software for editing robot motion and use the robot Poppy Torso to record poses. Wölfel et al. [33] developed the ToolBot system for making reproductions of handicraft with the KUKA LWR IV robot arm and a GUI for editing motions recorded with motion capture. Our work differs, as in our prototype the user's whole body is a form of input for animating a humanoid robot. Work that is most similar is that by Porfirio et al. [4] and work implementing Extended Reality (XR, an umbrella term for methods such as VR, augmented reality, and mixed reality) for HRI software tools as outlined by [34], [35]. Porfirio et al. [4] developed the system *Synthé* to program social interactions for the NAO robot. This system enables designers to use bodystorming, a technique in which they use their bodies and props to brainstorm about the interaction. We similarly focus on prototyping using the human body, but their focus is on programming interaction scenarios and only pre-programmed animations are used. Coronado et al. [35] review research on HRI software tools that implement XR devices, game engines, physical robots, and sensors for human input. For example, Alonso et al. [34] outline a system for VR teleoperation of the NAO robot. Here the user views a virtual robot using a VR headset, which differs from our setting in which the physical robot can be observed by the human interaction partner at all times.

III. Recording motion for animation

Our focus is on systems for robot animation using a human-humanoid pose-matching imitation system. A prototype was developed that enables programming the robot Pepper by means of human motion demonstrations with Kinect v2 input. The system's imitation functionality translates a human's body pose to joint angles for Pepper, so that it directly and continuously imitates the human's body pose. The system can be used to develop robot animations: the user starts the recording, demonstrates a motion that is directly imitated by the robot (Fig. 1), stops the recording, and can then replay the recording for execution by the robot. During replay (Fig. 2), the robot is not imitating the current body pose of the user, but executing recorded poses. The system needs to implement functionalities to store and replay recordings. Moreover, the user has to be able to infer system status and give commands to change it. Recording human motion for replay on the robot can be seen as a form of robot programming. Target groups include users with little to no programming skills, for instance in the fields of education and entertainment, or HRI researchers and others who want to quickly prototype robot motions for use in interaction scenarios. An iterative design process was followed in which the prototype was evaluated (Sec. III-B) and then revised, resulting in the system architecture in Fig. 3.

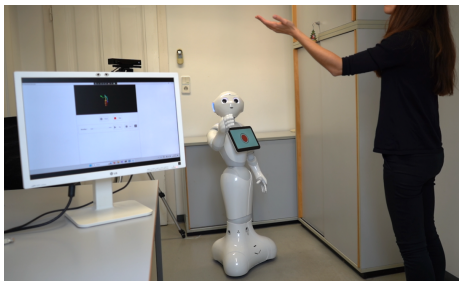

Fig. 1. User interaction with the system during recording (after revision, see Sec. III-C). The screen (left) displays the recording GUI. Pepper's tablet displays an icon indicating current system state.

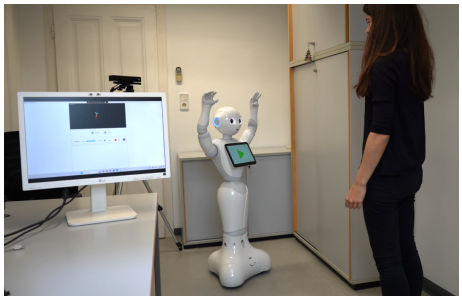

Fig. 2. Interaction during replay, when the robot executes recorded motion.

A. Technical implementation

The prototype was developed for the humanoid robot Pepper by Aldebaran, which has 20 DOF [36], and Kinect v2, with code for human-humanoid motion imitation running on a laptop. A GUI provided functionality such as making recordings and replaying them. The input to the program consists of 3D human joint positions; output consists of robot motion commands. During imitation, joint angles for Pepper are calculated in near real time, and during replay the robot executes motion based on recorded human joint position data.

Performing visual demonstrations of user body movements suffers the drawback of the correspondence problem, as there is a substantial difference between human and robot

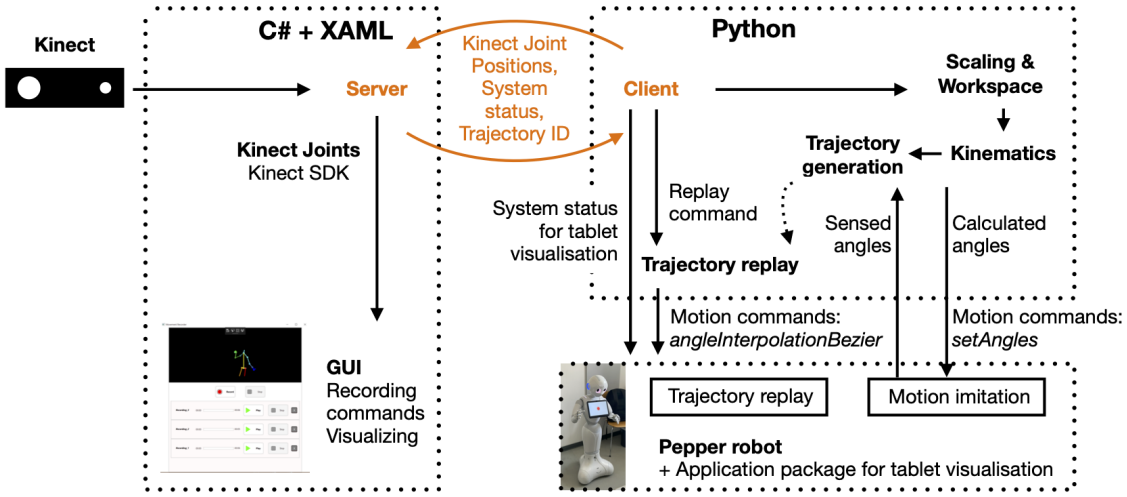

Fig. 3. Overview of the revised system architecture. Kinect input data is processed and sent to a Python program via a websocket. On the Python side, human joint positions are mapped to Pepper's workspace and a kinematics module is used to calculate Pepper's joint angles. During direct motion imitation, these joint angles are sent to the Pepper robot as motion commands. When recording, joint angles are sensed and stored for later execution.

bodies [20], [17]. Human bodies have different motion possibilities and constraints compared to robots, which means that devising a mapping is required; Pepper's arms have fewer DOF than human arms and the robot does not have separate legs but instead one 'leg' attached to a wheeled base. We chose a mapping to achieve pose similarity rather than a mapping between end-effector positions.

The imitation was realized by capturing human joint positions using Kinect v2 skeletal tracking with a C# Visual Studio project. The imitation system allows for controlling Pepper's ShoulderPitch, ShoulderRoll, ElbowYaw and ElbowRoll joints of both arms, the HipRoll, HeadPitch and HeadYaw joints, and the opening and closing of both hands. In our system only the person closest to the sensor is tracked and only a subset of Kinect frames is processed. Using C# with an XAML project and the Kinect SDK, human joint positions were visualized on the recording GUI. A websocket was implemented that sent human joint positions to a Python program. Human joint positions were scaled to fit Pepper's limb lengths and mapped to its workspace, to ensure a reachable position while maintaining the pose configuration. We developed an inverse and forward kinematics module [37] to calculate the required joint angles for Pepper to achieve a human's pose. The NAOqi Python SDK [36] was used to generate motion commands for Pepper.

B. Intermediate evaluation for system development

A user evaluation was conducted with six participants who either had dance/movement expertise or experience programming Pepper. Pepper programmers are familiar with the robot and can compare the use of our system to their experiences programming Pepper. Movement experts are one of the target groups, and the system needs to be easy to use for them. Movement expert participants (2 women, 1 man, age M=35.3 years, SD=3.0 years) had occupations such as movement or dance teacher, choreographer, and dancer.

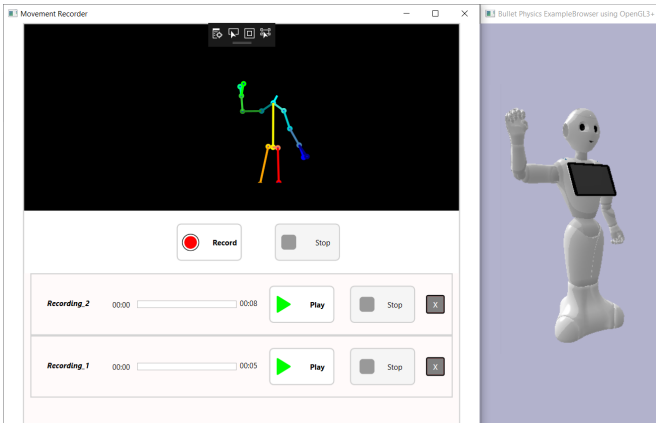

Fig. 4. GUI and qiBullet simulator executing the same pose.

Self-reported programming experience scores (on a scale from *1=very inexperienced* to *10=very experienced*, question based on [38]) were 2, 3 and 4. Self-reported familiarity with robots scores (on a scale from *1=not at all familiar* to *5=very familiar*, question based on [39]) were 1, 2 and 3. Three participants, students in the fields of informatics and electrical engineering, were invited who had experience programming Pepper (2 women, 1 man, age M=27.3 years, SD=3.2 years). Self-reported programming experience scores were 8, 9 and 9. All three rated their familiarity with robots to be 5 out of 5 and reported having programmed Pepper using Choregraphe, Python, C++ and/or ROS wrappers.

Participants were given informed consent and data consent forms to sign. Then, participants interacted with the imitation system for 5-10 minutes to explore its functionality. Participants were asked to record three motions that were developed by a dance professional and that covered different kinematic chains and gaze directions. The robot would imitate the participant's motions continuously, except

when replaying a recording. Participants were interviewed on their experience of the system. All participants completed the task of recording and replaying the gestures using the GUI. The System Usability Scale (SUS) rating of the system with recording GUI by the movement experts was M=72.5 (SD=10.9), the programmers' rating was M=87.5 (SD=11.5). According to Bangor et al. [40], systems with a SUS rating above 70 are acceptable in terms of usability while better products score in the high 70 to 80 range.

Interviews were transcribed and analyzed using a thematic analysis approach [41] using MAXQDA [42]. The convention we use in the following is that P1 indicates *participant 1 with experience programming Pepper robots* and D1 indicates *participant 1 with dance/movement expertise*. Across both groups, participants commented on the recorded motion, the leaning motion that was part of the recording, the motion imitation, motion speed and other motion qualities, and responded to the whole system and participant attention during the task. Regarding imitation accuracy, responses included that fast motion looks fine but slow motion may need to be smoothed as it looked robotic or jittery (P1, P3), and that imitation worked very well (P2, P3), experiencing imitation as intuitive (P2). D2 had the impression that it seemed easier for the system to replicate fast movements than slow movements. There were some differences between the responses of both groups. Programming participants responded to the Kinect and perceived sensor inaccuracies, responded to individual features on the GUI, and suggested potential improvements for the GUI. Movement experts responded to the robot, compared its motion to human motion possibilities, remarked on a sense of togetherness, compared the relation between human and robot in this setup to a pedagogue-pupil relationship, and indicated feeling a sense of empathy.

C. Revised System Architecture

After the evaluation, the robot's motion in response to the user, the method of trajectory generation, and the GUI were changed. Fig. 3 shows an overview of the revised system architecture. Recorded motion can be translated into a trajectory by either using calculated robot angles that are sent as motion commands or robot angles that are sensed during motion imitation (angles that are actually reached), and choices need to be made regarding the sampling rate. To prevent recalculating angles for every replay, and for personal data protection considerations, the system architecture was adapted to store robot trajectory data. Specifically, the trajectory is now based on stored sensed robot angles, so the trajectory during replay of motion is more predictable; the sensed angles have been seen by the user during recording. To realize smoother trajectory replay, a trajectory was generated on the basis of sensed robot joint angle data using the `angleInterpolationBezier` function from the NAOqi ALMotion module for robot joint control. This is a blocking call, as opposed to the non-blocking `setAngles` command that was used previously.

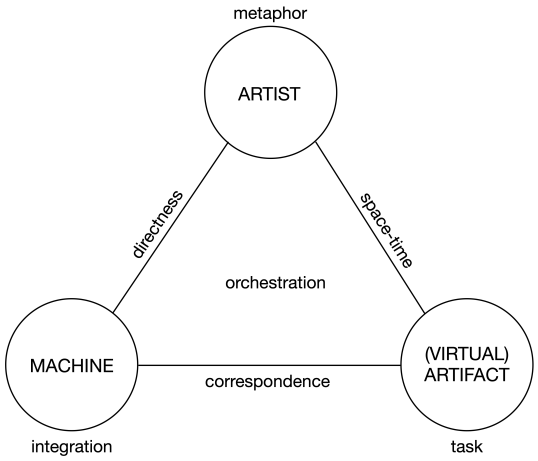

Fig. 5. Design space for computer animation from Walther-Franks and Malaka [10].

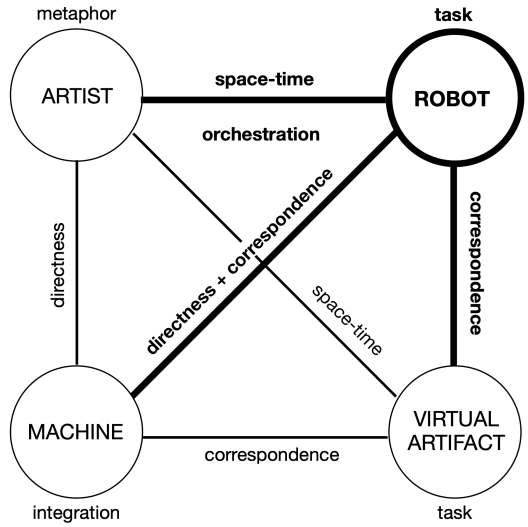

Fig. 6. Extended design space for robot animation based on [10] with the additional item *Robot*.

The GUI was adapted to have boxes for separate recordings, and a virtual robot was added using qiBullet [43], which implements a physics simulation of the Pepper robot (Fig. 4) to enable programming without direct execution by a physical robot. We added visual feedback on the tablet regarding the robot's status (e.g., a recording icon).

IV. Interaction Design Considerations

In this section, we outline interaction design considerations, using Walther-Franks and Malaka's design space for computer animation [10] (Fig. 5). We use this design space and extend it, to emphasize how the introduction of a physical robot platform complicates a computer animation task, requiring consideration of additional interaction design aspects. We extend the design space (Fig. 6) to describe systems for animating humanoid robot motion using imitation of human body movement. The physical presence of the robot and other devices that are distributed in space affect the way the user will divide their attention across these devices.

A. Design space for computer animation

In the design space for computer animation by Walther-Franks and Malaka [10] (Fig. 5), the entities involved in interaction include the *Artist* (who designs the motion), the *Machine* (combination of hardware and software), and the *Artifact* (the moving virtual character). The *Task* refers to what the tool is used for, namely animation. *Integration* refers to the way the input device is used as a means of control of character DOFs, and involves making decisions regarding which joints are mapped. *Correspondence* refers to the match between input entering the input device and the resulting animation of the artefact. *Metaphors* mentioned in [10] include the conversation metaphor, manipulation metaphor, and embodiment metaphor. *Directness* refers to the spatio-temporal distance or offset between user and animation, and includes issues such as precision and occlusion. *Orchestration* refers to how information is presented to the user and which actions the user can take in a particular order. *Space-time mapping* refers to the mapping of the dimensions of time and space from input to output. For instance, video playback involves a mapping from time to time, while computer puppetry involves a mapping from space-time to space-time [10].

B. Extended design space for robot animation

In this section, we highlight interaction design considerations for robot animation systems that implement a human-humanoid imitation system, based on our prototype, using the extended design space in Fig. 6. In systems for robot animation, the *Artifact* is split into a *Virtual artifact* (the recording data and the animation of a virtual robot) and animation of the physical *Robot*. Introducing a *Robot* adds additional *Correspondence* relations and affects *Orchestration*.

1) Task: Main animation tasks include motion creation, motion editing, and motion viewing [10]. For motion creation, choices are to be made regarding the type of demonstration (trajectory demonstration or keyframe demonstration), which data to use to generate a trajectory during replay, the sampling rate, and speed of the robot's motion. These choices have several implications for the experience of the interaction, and should be made in relation to the aim of the system. Choices may affect, for example, the required interaction during motion demonstration. A different user interaction is required when the user can demonstrate a motion fluently, as compared to an interaction in which the user demonstrates a series of poses for keyframe demonstration (which the system will interpolate between). Choices may also affect if timing information for the motion is recorded by default or not, how predictable motion will be during replay, and how detailed recorded motion will be. The sampling rate has consequences for the smoothness and precision of the motion. The sampling rate can be made dependent on the motion speed of the robot, meaning that for fast motions, more frames per second are stored to preserve detail. Recorded motion can be observed on a physical robot and/or a virtual robot. There are different ways to view the motion; viewing the motion while it is connected to the human demonstrator, versus viewing the motion when replaying a recording, disconnected from the human demonstrator. One consequence is that the robot should safely be moved from its current configuration into the first recorded pose when executing a recorded motion. For motion viewing, the system needs to take the starting pose into account, and move the robot from its current configuration into the first recorded pose. This is especially relevant for working with a physical robot platform, as the robot may otherwise jolt into the first pose. Potential extensions of the prototype for motion editing include trajectory editing capabilities (for individual joints). For generating motion that is both expressive and goal-oriented, it may be necessary to combine different types of motion, parametrize motions, or integrate additional objectives when the trajectory is generated (see [15], [44], [45], [12]).

Part of the *Task* involves interaction with the GUI, which can include, e.g., visual representations of system status, sensor information, existing recordings, feedback to user action and feedback regarding mode changes. Providing a stick figure that visualized human joint positions detected with Kinect gave participants information regarding tracking errors and accuracy (P1, P2). A virtual robot can be included for replay of the motion, troubleshooting in case errors arise, and working without the physical robot present. Providing options to associate additional data to recorded motions can be useful for organizing and reusing recorded motion. For instance, recordings could be named, labeled with various metadata, and visual representations of recorded trajectories may be stored alongside. As suggested by P3, motion commands could be exported to a Python script, to give end users the option to use the code file with other software

2) Directness: Directness depends on the sensors, calculation speed, network delays, and speed of robot motion execution. Executing motion commands by a wireless connection to the Pepper robot may add latency as compared to a virtual robot [25]. Use of (a single) Kinect can lead to occlusions and inaccuracies in the tracking data. Other possible input methods include, e.g., different depth sensors, using a combination of RGB(D) cameras to reconstruct the 3D scene, or use of on-body sensors. Each method has its (dis)advantages. Combining several sensors from different viewpoints may make tracking more accurate, but may complicate the work process and introduce costs for additional sensors.

3) Correspondence: We can consider the aspect of *Correspondence* on several levels, for instance similarity between human and robot motion in pose configuration or end-effector position (see Sec. III-A for discussion of the correspondence problem), similarity in movement style, and choices regarding spatial orientation/location change. Choices have effects on the type of motion that can be realized on the robotic platform, as well as the user's perception of the motion.

For devising a mapping, the first choice that needs to be made regards whether it is more important to devise a mapping between end-effector positions (e.g., similarity between human hand location and the gripper of a robot

arm) or a mapping to achieve pose similarity (as in our prototype). For human-humanoid imitation systems, the robot has a limited workspace. Imitation is convincing only in a particular human motion range. Consider also motion speed: if the robot has a lower speed of motion than human users, it will not be able to closely match a human's motion at high speeds. Dancers in the evaluation reported exploring how the robot's motion responded to theirs (D2, D3), including how the robot would respond to motion it cannot reproduce, such as jumping or lifting the shoulders. Moreover, they reported that the robot moves in a way that is not human-like, as it can isolate joints and it does not have a spine (D1). For the prototype described here, only in-place motion is considered, no location change. Locomotion and translation involve movements that result in a change of the robot's position in space (whereas configuration change consists of in-place body movements) [11]. Such motions could be considered for inclusion, but introduce complications for our prototype as a single RGBD sensor such as Kinect assumes a frontal position. The user is required to stay in the sensor's field of view in order to be sensed.

4) Orchestration: Regarding *Orchestration*, user attention is a factor of importance. Our prototype required changing the orientation of the body towards the Kinect, robot or the screen, visually attending to the screen and the robot, and ensuring body data can be captured with Kinect. Sensors that need to frontally capture human joint position data should be placed so that the user can orient themselves toward the robot. Mirroring of human motion by the robot (rather than imitating) while the human and the robot are facing each other was considered more familiar in the user evaluation (from viewing oneself in the mirror). If a user whose motion is imitated also needs to give commands using a GUI, the interaction design relating to the choice of input modes requires consideration, to avoid recording motion that should not be reproduced. For instance, during the evaluation, the user's leaning motion towards the computer to give start and stop commands was also reproduced. This should be avoided, for example by using alternative ways of giving commands to start and stop recording, e.g., with gesture control, speech commands, a physical button, or working with an additional person who starts and stops the recording. In the revised prototype, a countdown timer was used to allow the user to get into position prior to starting the recording.

The prototype required changing the orientation of the body (towards the Kinect or the screen), visually attending to parts of the system (the screen and the robot), and ensuring body data can be captured with Kinect. To minimize the need to continuously monitor all devices that are located in different places, the interaction could be focused on the robot, while an external screen is used for representing recorded motion, viewing recorded motion (e.g., in absence of a physical robot), and for setting the recording settings. To what extent an external screen is necessary depends on the chosen way of giving commands.

5) Integration: Decisions need to be made regarding which joints are mapped. P1 reflected on adding the possibility to the GUI to select individual limbs for recording. For *Integration*, one important consideration is that such systems make assumptions regarding bodies of users (see [46]) and their movement capabilities. In the prototype, assumptions arise due to the use of the Kinect and the mapping that is made from human to the robot, for example that users have two arms and are able to move them, which is not the case for everyone. Ideally, options should be given to customize which joints are detected and the robot's motion in response to human motion. But perhaps more importantly, other ways of programming the robot should remain available (e.g., providing code or specifying joint angle values).

6) Metaphor: Regarding *Metaphor*, dancers in the evaluation reported having the experience of embodying the limitations of the robot, or of the robot's limitations restricting their movement (D1). Some compared the interaction to the practice of being a movement teacher (teacher-pupil metaphor, D2), or reported experiencing the interaction as a conversation (D2, D3), communication or duet. A sense of connection or empathy was also reported.

7) Space-time mapping: Our prototype has a space-time to space-time mapping, from human movement in 3D space to robot movement in 3D space preserving timing. Generally, if there is both a physical robot platform and a virtual animation of a robot, this means that there are two space-time mappings in the system. Systems can deviate from each other. The program simulating the virtual robot can include a physics simulator, but that is not necessarily the case.

V. Conclusion

We developed a prototype for robot animation of the Pepper robot, which was revised after evaluation with dancers and programmers. The goal of the system is to facilitate robot animation by users who may not have sufficient programming skills but who do have other relevant expertise for developing contextually appropriate human-robot interactions. We extend a design space for computer animation so that it becomes applicable for describing robot animation systems that implement a human-humanoid pose-matching imitation system. We identify interaction design considerations connected to system architecture choices.

ACKNOWLEDGMENT

Supported by TU Wien Doctoral College TrustRobots, FWF #ConnectingMinds grant to the project Caring Robots // Robotic Care (CM 100-N), and a LINK-Masters project grant by Stiftung Niedersachsen to *"AI Tool for Dance and Choreography: It's embodied and personalized, dare improvise?"*

References

[1] M. Jung and P. Hinds, "Robots in the wild: A time for more robust theories of human-robot interaction," *ACM Transactions on Human-Robot Interaction*, vol. 7, no. 1, p. 5, 2018.

[2] E. Coronado, F. Mastrogiovanni, B. Indurkhya, and G. Venture, "Visual Programming Environments for End-User Development of intelligent and social robots, a systematic review," *Elsevier Journal of Computer Languages*, vol. 58, no. 100970, 2020.

[3] S. Šabanović, "Robots in society, society in robots: Mutual shaping of society and technology as a framework for social robot design," *Int. J. of Soc. Robot.*, vol. 2, no. 4, pp. 439–450, 2010.

[4] D. Porfirio, E. Fisher, A. Sauppé, A. Albarghouthi, and B. Mutlu, "Bodystorming Human-Robot Interactions," in *Proceedings of UIST'19*. New Orleans LA USA: ACM, 2019, pp. 479–491.

[5] E. Jochum and J. Derks, "Tonight we improvise!: Real-time tracking for human-robot improvisational dance," in *MOCO '19*. ACM, 2019, pp. 1–11.

[6] E. Balit, D. Vaufreydaz, and P. Reignier, "Integrating animation artists into the animation design of social robots an open-source robot animation software," in *Proceedings of HRI'16*, 2016, pp. 417–418.

[7] H. Knight and R. Simmons, "An intelligent design interface for dancers to teach robots," in *RO-MAN 2017)*, 2017, pp. 1344–1350.

[8] A. LaViers, C. Cuan, C. Maguire, K. Bradley, K. Brooks Mata, A. Nilles, I. Vidrin, N. Chakraborty, M. Heimerdinger, U. Huzaifa, R. McNish, I. Pakrasi, and A. Zurawski, "Choreographic and somatic approaches for the development of expressive robotic systems," *Arts*, vol. 7, no. 2, p. 11, 2018.

[9] D. Sirkin and W. Ju, "Using embodied design improvisation as a design research tool," in *International Conference on Human Behavior in Design*, 2014, p. 7.

[10] B. Walther-Franks and R. Malaka, "An interaction approach to computer animation," *Entertainment Computing*, vol. 5, no. 4, pp. 271–283, 2014.

[11] T. Schulz, J. Torresen, and J. Herstad, "Animation techniques in human-robot interaction user studies: A systematic literature review," *ACM Transactions on Human-Robot Interaction*, vol. 8, no. 2, pp. 1–22, 2019.

[12] T. Ribeiro and A. Paiva, "The practice of animation in robotics," in *Modelling Human Motion*, N. Noceti, A. Sciutti, and F. Rea, Eds. Springer International Publishing, 2020, pp. 237–269.

[13] G. Hoffman and W. Ju, "Designing robots with movement in mind," *Journal of Human-Robot Interaction*, vol. 3, no. 1, p. 89, 2014.

[14] G. Venture and D. Kulić, "Robot expressive motions: A survey of generation and evaluation methods," *ACM Transactions on Human-Robot Interaction*, vol. 8, no. 4, pp. 1–17, 2019.

[15] J. Gray, G. Hoffman, S. O. Adalgeirsson, M. Berlin, and C. Breazeal, "Expressive, interactive robots: Tools, techniques, and insights based on collaborations," in *HRI 2010 Workshop: What do collaborations with the arts have to say about HRI*, 2010, p. 8.

[16] M. Saerbeck and A. J. N. v. Breemen, "Design guidelines and tools for creating believable motion for personal robots," in *RO-MAN 2007*, 2007, pp. 386–391.

[17] H. Ravichandar, A. S. Polydoros, S. Chernova, and A. Billard, "Recent Advances in Robot Learning from Demonstration," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 3, no. 1, pp. 297–330, 2020.

[18] G. Adamides, C. Katsanos, Y. Parmet, G. Christou, M. Xenos, T. Hadzilacos, and Y. Edan, "HRI usability evaluation of interaction modes for a teleoperated agricultural robotic sprayer," *Applied Ergonomics*, vol. 62, pp. 237–246, 2017.

[19] P. Rouanet, J. Bechu, and P.-Y. Oudeyer, "A comparison of three interfaces using handheld devices to intuitively drive and show objects to a social robot: the impact of underlying metaphors," in *RO-MAN 2009*. IEEE, 2009, pp. 1066–1072.

[20] F. A. Bravo, A. M. González, and E. González, "A Review of Intuitive Robot Programming Environments for Educational Purposes," in *2017 IEEE 3rd Colombian Conference on Automatic Control (CCAC)*, Cartagena, Colombia, 2017, pp. 1–6.

[21] M. Alibeigi, S. Rabiee, and M. N. Ahmadabadi, "Inverse kinematics based human mimicking system using skeletal tracking technology," *Journal of Intelligent & Robotic Systems*, vol. 85, no. 1, pp. 27–45, 2017.

[22] F. Zuher and R. Romero, "Recognition of human motions for imitation and control of a humanoid robot," in *2012 Brazilian Robotics Symposium and Latin American Robotics Symposium*, 2012, pp. 190–195.

[23] FraPorta, "Pepper teleoperation using OpenPose," 2022. [Online]. Available: https://github.com/FraPorta/pepper_openpose_teleoperation

[24] ketchart, "Pepper-robot-controlled-by-kinect-in-ubuntu," 2020, https://doi.org/10.5281/zenodo.3828158.

[25] M. Hirschmanner, C. Tsiourti, T. Patten, and M. Vincze, "Virtual Reality Teleoperation of a Humanoid Robot Using Markerless Human Upper Body Pose Imitation," in *Humanoids 2019*, Oct. 2019, pp. 259–265.

[26] S. Nakaoka, "Choreonoid: Extensible virtual robot environment built on an integrated GUI framework," in *2012 IEEE/SICE International Symposium on System Integration (SII)*, 2012, pp. 79–85.

[27] L. Leite and V. Orvalho, "Shape your body: control a virtual silhouette using body motion," in *CHI '12 Extended Abstracts on Human Factors in Computing Systems*. ACM, 2012, pp. 1913–1918.

[28] D. Parmar, J. Isaac, S. V. Babu, N. D'Souza, A. E. Leonard, S. Jörg, K. Gundersen, and S. B. Daily, "Programming moves: Design and evaluation of applying embodied interaction in virtual environments to enhance computational thinking in middle school students," in *2016 IEEE Virtual Reality (VR)*, 2016, pp. 131–140.

[29] A. Sanna, F. Lamberti, G. Paravati, and F. D. Rocha, "A kinect-based interface to animate virtual characters," *Journal on Multimodal User Interfaces*, vol. 7, no. 4, pp. 269–279, Dec. 2013.

[30] G. Hoffman, R. Kubat, and C. Breazeal, "A hybrid control system for puppeteering a live robotic stage actor," in *RO-MAN 2008*, 2008, pp. 354–359.

[31] M. Suguitan and G. Hoffman, "Blossom: A handcrafted open-source robot," *ACM Transactions on Human-Robot Interaction*, vol. 8, no. 1, pp. 1–27, 2019.

[32] F. Rietz, A. Sutherland, S. Bensch, S. Wermter, and T. Hellström, "WoZ4u: An open-source wizard-of-oz interface for easy, efficient and robust HRI experiments," *Frontiers in Robotics and AI*, vol. 8, p. 668057, 2021.

[33] K. Wölfel, J. Müller, and D. Henrich, "ToolBot: Robotically reproducing handicraft," in *Human-Computer Interaction – INTERACT 2021*, C. Ardito, R. Lanzilotti, A. Malizia, H. Petrie, A. Piccinno, G. Desolda, and K. Inkpen, Eds. Springer International Publishing, 2021, vol. 12934, pp. 470–489.

[34] R. Alonso, A. Bonini, D. Reforgiato Recupero, and L. D. Spano, "Exploiting virtual reality and the robot operating system to remote-control a humanoid robot," *Multimedia Tools and Applications*, vol. 81, no. 11, pp. 15 565–15 592, 2022.

[35] E. Coronado, S. Itadera, and I. G. Ramirez-Alpizar, "Integrating virtual, mixed, and augmented reality to human–robot interaction applications using game engines: A brief review of accessible software tools and frameworks," *Applied Sciences*, vol. 13, no. 3, p. 1292, 2023.

[36] SoftBank Robotics, "Softbank robotics documentation," 2021. [Online]. Available: http://doc.aldebaran.com/2-5/index_dev_guide.html

[37] D. Stoeva, H. A. Frijns, M. Gelautz, and O. Schürer, "Analytical solution of pepper's inverse kinematics for a pose matching imitation system," in *2021 30th IEEE International Conference on Robot Human Interactive Communication (RO-MAN)*, 2021, pp. 167–174.

[38] J. Feigenspan, C. Kästner, J. Liebig, S. Apel, and S. Hanenberg, "Measuring programming experience," in *2012 20th IEEE International Conference on Program Comprehension (ICPC)*, June 2012, pp. 73–82.

[39] L. Bishop, A. van Maris, S. Dogramadzi, and N. Zook, "Social robots: The influence of human and robot characteristics on acceptance," *Paladyn, Journal of Behavioral Robotics*, vol. 10, no. 1, pp. 346–358, Oct. 2019.

[40] A. Bangor, P. T. Kortum, and J. T. Miller, "An empirical evaluation of the system usability scale," *International Journal of Human-Computer Interaction*, vol. 24, no. 6, pp. 574–594, 2008.

[41] A. Bladford, "Semi-structured qualitative studies," in *The Encyclopedia of Human-Computer Interaction*, 2nd ed. The Interaction Design Foundation, 2013.

[42] MAXQDA, VERBI GmbH, "MAXQDA | All-In-One Qualitative & Mixed Methods Data Analysis Tool," 2020. [Online]. Available: https://www.maxqda.com/

[43] M. Busy and M. Caniot, "qiBullet, a Bullet-based simulator for the Pepper and NAO robots," *arXiv preprint arXiv:1909.00779*, 2019.

[44] J. Kim, K. S. Chun, and D.-S. Kwon, "Gesture motion programming by applying robot motion hierarchy structure for the educational/entertainment robot engkey," in *2012 IEEE Workshop on Advanced Robotics and its Social Impacts (ARSO)*, 2012, pp. 36–39.

[45] E. Saad, J. Broekens, and M. A. Neerincx, "An iterative interaction-design method for multi-modal robot communication," in *RO-MAN 2020*, 2020, pp. 690–697.

[46] K. Spiel, "The bodies of TEI – investigating norms and assumptions in the design of embodied interaction," in *TEI '21*, 2021, pp. 1–19.