2020 IEEE International Conference on Artificial Intelligence and Virtual Reality (AIVR) | 978-1-7281-7463-1/20/$31.00 ©2020 IEEE | DOI: 10.1109/AIVR50618.2020.00026

MoveBox: Democratizing MoCap for the Microsoft Rocketbox Avatar Library

Mar Gonzalez-Franco[1], Zelia Egan[1], Matthew Peachey[2],
Angus Antley[1], Tanmay Randhavane[1], Payod Panda[1], Yaying Zhang[1],
Cheng Yao Wang[4], Derek F. Reilly[2], Tabitha C Peck[3], Andrea Stevenson Won[4], Anthony Steed[5] and Eyal Ofek[1]

[1]*Microsoft Research* Redmond, WA, USA margon@microsoft.com
[2]*Dalhousie University* Halifax, Canada
[3]*Davidson College* Davidson, NC, USA
[4]*Cornell University* Ithaca, NY, USA
[5]*University College London* London, UK

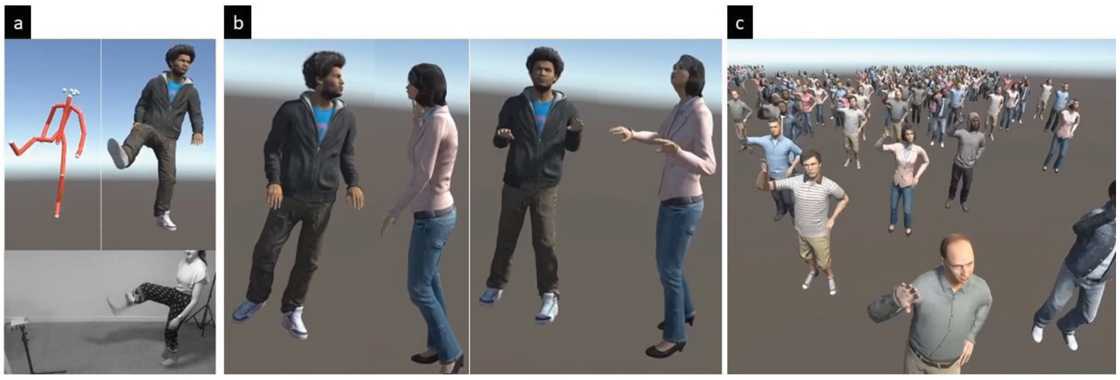

Fig. 3. A selection of avatars animated with the MoveBox system. a) Person doing live MoCap from Azure Kinect onto a Microsoft Rocketbox avatar. b) Playback of two avatar animations created to represent a social interaction with MoveBox. c) A crowd of avatars dancing powered by Mocap.

***Abstract*—This paper presents MoveBox an open sourced toolbox for animating motion captured (MoCap) movements onto the Microsoft Rocketbox library of avatars. Motion capture is performed using a single depth sensor, such as Azure Kinect or Windows Kinect V2. Motion capture is performed in real-time using a single depth sensor, such as Azure Kinect or Windows Kinect V2, or extracted from existing RGB videos offline leveraging deep-learning computer vision techniques. Our toolbox enables real-time animation of the user's avatar by converting the transformations between systems that have different joints and hierarchies. Additional features of the toolbox include recording, playback and looping animations, as well as basic audio lip sync, blinking and resizing of avatars as well as finger and hand animations. Our main contribution is both in the creation of this open source tool as well as the validation on different devices and discussion of MoveBox's capabilities by end users.**

***Index Terms*—Avatars, Animation, Motion Capture, MoCap, Rigging, Depth Cameras, Lip Sync**

I. Introduction

Avatars are of increasing importance for social research and social Virtual Reality (VR) applications [1]. Avatars are needed to interact with others inside social virtual environments (VEs), and are the basis for creating realistic stories inside VEs [2]. While some avatars may represent intelligent agents that synthesize their motion from prerecorded motion-capture (MoCap) [3], others may be a real-time virtual representation of real people [4]. These *self-avatars* are a representation of a user within head mounted display (HMD) worn VR and are fundamental to social-science embodiment research [5]–[8].

One of the main challenges in creating realistic avatars and self-avatars is the need to animate them so that their motions replicate the user's motions [9]–[11]. Previous research and development has develop a variety of sensors, tools and algorithms for animating avatars. A hindrance to this work has been a common set of open source data and tools. This is partly addressed by the Microsoft Rocketbox library [12].

Animation of avatars is often performed through motion capture (MoCap) systems that capture user motions in VR. MoCap is the processes of sensing a person's pose and movement. Real-time motion capture of users is usually limited to measuring the position and orientation of the user's head, measured by the HMD, and the position and orientation of the left and right palms, measured by hand-held trackable controllers. Controllers may include a set of input buttons and triggers, as well as additional capacitive sensors for recovery of the finger motions in the vicinity of the controllers. The tracking may be done using grounded sensors or beacons such as light emitters, cameras, or magnetic sensors [9].

A common technique to extrapolate limited sensing to a full body animation is through Inverse Kinematics (IK) [13], [14]. IK works best for joints between known locations and thus poses a challenge for joints that are far from the head and the hands, such as hips, legs or feet. To enable animation of the feet, IK usually applies heuristics such as generating walking steps based on the global movement of the user's head, or using pre-recorded animations.

High quality, non real-time animations are captured using professional motion capture systems. In systems such as Optitrack and Vicon multiple high frame rate cameras are used to capture a professional actor's performance. The resulting motions recorded by such systems are of high quality and can be used in projects such as motion picture productions and AAA games. However these systems are expensive and require complex setup, which makes them inaccessible for many researchers, artists, small content creators, and users.

Furthermore, once a motion is captured, there are still difficulties in applying the motion to the desired avatar. Different motion capture and avatar systems may represent the skeletal structure in different ways, with variations in the number of joints and the topology of bones [11]. Large productions employ artists and technicians familiar with each system to ensure the pipeline from capturing to rendering will perform as needed. However there is a need for a simpler pipeline for a range of other use cases, for example by researchers in fields such as psychology or sociology wishing to use avatars as tools to test new theories (e.g. [3], [15]).

In this paper, we present an open source toolbox, MoveBox, that uses a single sensor—specifically a depth camera such as the Microsoft Azure Kinect or Kinect V2—to record the full body motion of a user. The captured motion can be applied in real-time to an avatar in Unity 3D. Envisioning the needs of researchers, artists, and others for such a tool box, we have added capabilities beyond the main skeletal animation using a Kinect depth camera, to help those that want to create convincing animated avatars without a sophisticated MoCap production studio. This includes using audio input to generate approximate lip motion on the avatars, synchronized voice recording, seamless recording and playback of animations, and scaling avatars to fit the body proportions of the user. We support the Microsoft RocketBox avatar library [16], which is freely available to the research community. It contains humans of different genders, races, and everyday occupations.

II. Related Work

A. Motion Capture

1) Body Capture: Most existing motion capture (MoCap) systems are based on capturing the actor performance using multiple sensors to enable robust capturing of all the skeleton joints' positions regardless of joint occlusion from some of the sensors [9]. The accuracy of the system is dependent on precise calibration of the relative position and orientation of all the sensors. An alternative is suits with Inertial Measurement Unit (IMU) arrays, but these suffer from translational drift [17] and require more setup time i.e. putting the suit on.

Different companies and researchers have attempted to use low cost trackers, such as a multitude of Vive Trackers as a base for motion capture [18]. However, these systems are targeted to the professional market with larger budgets. Liu et al. [19] developed a system that uses six wearable sensors, however even this minimal number is still cumbersome for smaller productions that need to maintain them charged and wear them in precise locations on each actor's body. Additionally, their system uses proprietary built hardware and the software used is not available to the public. Ahuja et al. [20] suggested a set of spherical mirrors and cameras that are mounted on the user's HMD, to capture the user's motion. While limited only to people that wear HMDs, the first person view of the body reduces the quality of the tracking.

Compared to other lightweight body tracking solutions, we have found Azure Kinect to be more robust to occlusion partially due to the underlying Deep Neural Network (DNN) body model that the data is being fit to. Although it is possible to integrate multiple Azure Kinects in a Unity application, for this toolbox we aim for a simple system that can be easily set up. We, therefore, simplified the system to use one single depth camera; either the Azure Kinect or the older model Microsoft Kinect V2. While limited to a single point of view, the camera does not need to be calibrated, be rigged to other cameras, or maintain temporal synchronization with other cameras, thus enabling a quick and easy setup.

2) Facial Capture: There are several approaches to construction of facial performance capture for virtual models. A common approach is camera-based solutions that employ computer vision algorithms to calculate the three-dimensional mesh of the face. These might require a camera system with a depth sensor [21], multi-view cameras [22], or binocular stereo [23]. Some solutions also employ deep learning techniques to estimate the mesh [24].

In addition to camera-based solutions, some audio-based facial animation solutions also exist. Lip-sync libraries typically estimate the visemes from the phonemes detected in the audio stream [25] [26]. Some solutions like the SALSA suite [25] can also synthesize the speech to predict emotions, and apply appropriate facial animations from this prediction.

It is important to note that if the user of the MoveBox system wears a VR headset, then their facial features might be obscured.

B. Character Engines

Motion capture systems that interface with avatars for end users are critical to the adoption of more human-like avatars for research and social VR [27]. Previous work in the area of character engines that support MoCap, like HALCA [28] and HUMAN [11], [29], have aimed at converting MoCap from different sources into avatars and robot skeletons. Work by

Gillies and Spanlang comparing real-time character engines for virtual environments proposed four metrics by which to evaluate character engines: 1) photo-realism 2) movement realism 3) interaction and 4) control. [30].

Currently, the most widely used game engines, Unity and Unreal, both provide industry quality for these four features if interaction is limited to the triggering of transitions in a finite state machine. Each game engine has plugins that allow streaming live motion capture data into an application. Such data can serve multiple purposes. First, it could be used to drive a character in a social mixed reality application. Second, it could be captured and used to implement motion synthesis algorithms such as motion matching [31]. Third, it could be used in a data set for Deep Learning [32] such as learning motions through reinforcement learning.

However, for social mixed-reality applications, live body-tracking data can be used to drive IK. IK targets can overlay a base animation generated by an animation graph that is implemented with a finite state machine such as the Mecanim Humanoid animation system in Unity. However, this setup can also be complicated to configure, and it is a drawback for recording from real-time motion capture systems that plugin to Unity, specially if they do not readily integrate with the Mecanim Humanoid animation system.

A common implementation for the situation where there is only head and hand tracking, is to blend the user motions with a base animation for the full body. One problem with this approach is that when the movement of the head triggers an animation, for example the walk animation, the result can be a unnatural scuttling of the avatar's feet. And additional fixes are needed to have both leaning and walking, for example having additional feet trackers [33].

III. MoveBox: the toolbox

MoveBox aims to provide several resources to users of the Microsoft Rocketbox avatars. On their own, avatars are only a list of vertices and meshes attached to joints. Without simple methods for implementing animations, it becomes challenging for users to bring them into their projects. Therefore, the main goal of this toolbox is to enable users to create animations for Microsoft Rocketbox avatars on their own. This will help democratize the use of the library and of avatars in general, and ensures that the use of avatars is not limited to those with professional grade animation equipment.

Figure 4 provides a schematic of the functions and features of MoveBox. MoveBox runs on the input of depth sensing cameras that provide skeletal tracking to perform the full body animation of the Microsoft Rocketbox avatars. It also uses audio input for creating a basic lip sync speech animation, and procedural control for minimal facial idling such as eye blinking [34]. Additional MoveBox features include recording and playback of MoCap animations and resizing of avatars.

A. Implementation

1) Real-time Motion Capture: MoCap data is input from either the Azure Kinect or Kinect V2, both of which are

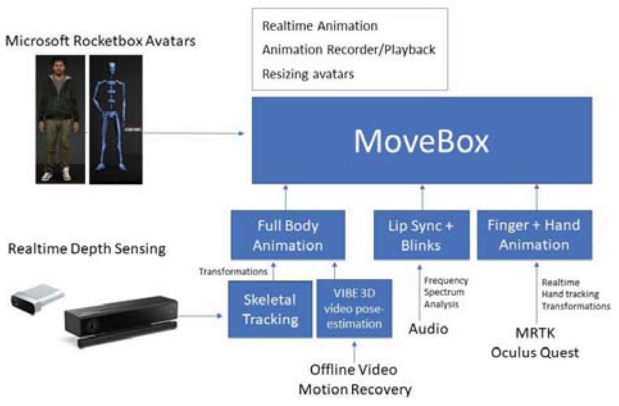
Microsoft Rocketbox Avatars
Realtime Animation
Animation Recorder/Playback
Resizing avatars
MoveBox
Full Body Animation
Lip Sync + Blinks
Finger + Hand Animation
Realtime Depth Sensing
Transformations
Skeletal Tracking
VIBE 3D video pose-estimation
Frequency Spectrum Analysis
Audio
Realtime Hand tracking Transformations
MRTK
Oculus Quest
Offline Video Motion Recovery

Fig. 4. Schematic of the functions and features of the toolbox.

capable of tracking, at at 30Hz, the three-dimensional data via their respective depth cameras. These cameras were selected mainly because their software development kits (SDKs) provide body tracking features. However, each of them have different systems of coordinates and bone structures, which can make the conversion and calibration between the tools and the avatars somewhat complex.

Therefore, once the body-tracked data is read into our system, we process that data in order for it to be compatible with the skeletal structure of the Rocketbox avatars (Figure 5).

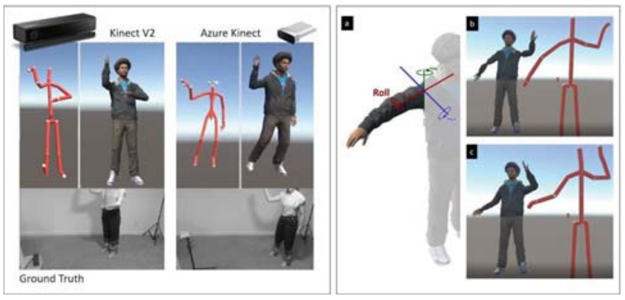

Fig. 5. LEFT Real-time body animation. In red the input (either in Azure Kinect or Kinect V2). On the side the reconstruction of the motion in the Microsoft Rocketbox. RIGHT: improvement of the Kinect V2 animation based on the original roll data of the avatar. a) Pitch and yaw rotations are correctly recovered by Kinect V2. b) however by default there is no consistent roll information. c) MoveBox recovers the original roll of the avatar to at least ensure an stable and valid reconstruction. See supplementary video.

To do so we extract each body frame of data from Azure Kinect or Kinect V2 at a rate of 30 frames per second. Each instance of a body frame contains data about several joints in the human body. Poses of avatars are represented by their skeletal structure, where each joint has its 3 or 6 degrees of freedom (DOF) transformation (rotations and sometimes position) from its parent in the skeletal structure. In total the Microsoft Rocketbox avatars have 53 body bones and 28 facial bones. However, neither Kinect V2 nor Azure Kinect provide that much information (see more details on the apparatus section). Hence the need to convert the different skeletal structures and poses.

The data measurements that we are most interested in are position and orientation, which are represented as a three dimensional (x, y, z) coordinate and a four dimensional quaternion (w, x, y, z) coordinate respectively.

Position simply measures where in the 3D space each joint is located during a particular frame, whereas the orientation measurement returns the relative rotation of each joint.

On system awake the position and orientation of every tracked joint in the avatar (25 in Kinect V2, and 32 in Azure Kinect) is stored for later reference by the animation rendering script. The skeletal structure of avatars and tracking systems are rarely the same. This is true for the RocketBox avatars and the Kinect trackers. Each frame the Kinect joint data is captured and then mapped to the corresponding joint in the RocketBox avatar skeleton in real-time.

After mapping the data to a Microsoft Rocketbox compatible format, the animation is rendered by looping through every joint and updating its position and orientation based on the current frame's values. In order to ensure that proper orientation of avatar limbs is maintained, a reference is kept for each joint's parent joint so that absolute position and rotation $(q_{absolute})$ values can be calculated via quaternion arithmetic:

$$\mathrm{q}_{absolute} = \prod_{\mathrm{i=root}}^{k} \mathrm{q}_i$$

Finally, the rotations between frames are interpolated using exponential smoothing. This smoothing is necessary to account for the difference in frame rate between the input cameras, often at only 30Hz, and the display, possibly at the 90Hz rendering frame rate of most VR HMDs.

2) Motion Recovery from Existing Videos: Movebox includes an external tool for 3D multi-person human pose estimation from RGB videos. We utilized a deep-learning based approach open sourced as VIBE [35], which trains a temporal model to predict the parameters of the SMPL body model [36] for each frame while a motion discriminator tries to distinguish between real and regressed sequences. The output of the model is a sequence of pose and shape parameters in the SMPL body model format. To animate RocketBox avatars with predicted 3D poses, the toolbox first extracted the joints data from pose parameters, computed the transformation between SMPL and RocketBox skeleton structures, and then mapped to the corresponding joint in the RocketBox avatar skeleton as shown in Figure 6.

Despite this video extraction does not work in real-time, it provides a good alternative and can demonstrate to be a critical feature to allow the recreation of scenes from archival footage or other content that has been recorded in real life, helping to transfer them to VR. This type of footage usage can be of great advantage to researchers on Immersive Journalism [37], where they want to recreate scenes. But also for anthropological and social studies, in which for example a bystander effect to a bar fight needs to be recreated [15].

3) Lip Sync: One common challenge with avatar animations is the incorporation of facial animation. Even in professional settings facial MoCap is often done separately from body capture.

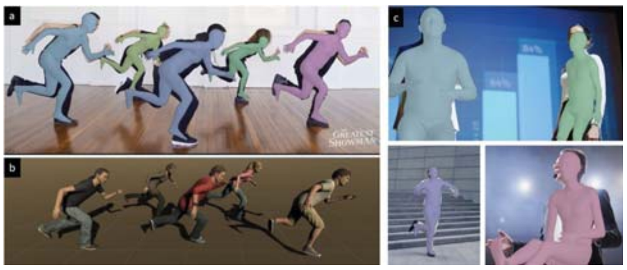

Fig. 6. 3D Multi-person pose estimation from RGB videos. (a) Predicted SMPL Models overlay on the video. (b) Animating Microsoft Rocketbox with predicted 3D pose. (c) non optimal results in videos where 1/3 partial body occluded, or half body occluded.

Some solutions for general users leverage RGB data from regular cameras and computer vision tools that already have facial expression recognition built-in, such as OpenPose [38]. This could be translated to the bone structure of RocketBox, or to drive additional blendshapes.

However, the use of cameras for facial animation is challenged when users are wearing an HMD in VR. There have been solutions to instrumentalize HMDs [39], but these are not universal for end users of the library. In that line a much simpler solution is to use audio input to generate less accurate but meaningful lip synchronization [34], that will nonetheless work even if users are wearing an HMD.

The Lip Synchronization feature in MoveBox retrieves audio information from the microphone, or an audio file, and animates the mouth of the avatar to open and close in real-time. It uses the spectral frequency of the microphone input to proportionally move two joints on the rigged face: the jaw and the upper lip. The result, although less impressive than using blendshapes with phonemes, gives a compelling experience.

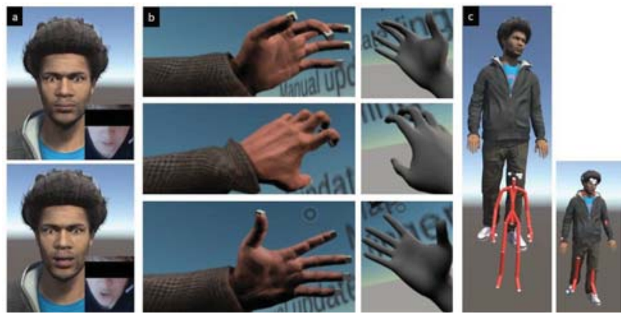

Fig. 7. (a) Real-time lip sync can be enabled using Movebox. (b) Movebox add-on to use real-time hand tracking with Oculus Quest. (c) Avatars can be resized to match the real size of the user. See supplementary video.

4) Hand and finger motions: Enabling avatar's hand movement can increase fidelity of expression. New devices such as the Oculus Quest and HoloLens 2 are providing finger tracking information in real-time. The expectation is that more devices will eventually include such finger tracking enabling better hand interactions. Our toolbox has an ad-on feature to apply device's hand tracking info on Rocketbox avatar's hand bone and 15 finger bones with few easy drag and drop actions on Unity editor, and hence animate the hands while in VR (Figure 7). This is an important feature as fingers and hands are very important for social interaction.

5) Joint Comparison : For many extended MoveBox features, such as enhanced animation looping or masking, we implemented a correlation algorithm between body parts of the same skeleton in two different frames or rotation states.

A similarity index is calculated between two body states, based on the array of quaternion rotations for every joint in the body.

The product of a quaternion q and its inverse q^{-1} is the identity $(1, 0, 0, 0)$, which has a vector part with a magnitude of zero.

$$\mathrm{qq}^{-1} = (1, 0, 0, 0)$$

The magnitude of the vector part of the quaternion product of q_1 and q_2^{-1} is zero when q_1 and q_2 are identical, and increases as the similarity between the two quaternions decreases.

Based on this equation, a similarity index (S_i) for a pair of body positions (with a body position defined by its set of joint rotations) can be defined as the sum of the magnitude of the quaternion product of one joint rotation quaternion and the inverse of the other for each joint or

$$S_i = \sum_{j=0}^{n} \sqrt{x_j^2 + y_j^2 + z_j^2}$$

where j is the joint and x_j, y_j, and z_j are the second, third and fourth parts of the quaternion $q_{Pj} = (w_j, x_j, y_j, z_j)$ and $q_{Pj} = q_{1j} * q_{2j}^{-1}$ and quaternions q_{1j} and q_{2j} are the pair of rotations for the joint j.

This ability lowers the barrier for a range of applications: an autocorrelation of positions among a set of frames in a recorded animation can enhance looped playback so that a loop appears more continuous; comparison of prerecorded body positions to user body positions captured on camera could enable the smooth integration of real-time movements with prerecorded animations, effectively creating a mask of motions on certain bones. For example, a creator might want to use a recorded body motion but implement a brand new lip sync.

This implementation of comparisons can enable a set of key features in the toolbox, including masking animations and the improved looping tool, as described below.

B. Features

1) Recording and Playback of Animations: An important feature of the toolbox is that it enables users to record the animations they create via motion capture and replay them across any of the Microsoft Rocketbox avatars. This will make creating many different characters that rely on similar motions much easier, as a user can simply create one animation and have it replayed on multiple avatars. It also helps speed up the creation of scenes with avatars as it is very easy to visualize the final output and/or record an alternative animation.

2) Animation Looping: The recording feature enables control, cropping and looping of animation clips upon playback. The applications of animation recording include the creation of multi-avatar scenes by a single creator, which are then improved by the ability to loop recordings which allows continued playback for an undetermined duration of time.

To enhance playback quality, a similarity looped animation is made available for more continuous loops. This is done by trimming the clip at a pair of frames, one near the beginning and one near the end, with similar body positions. An optimal pair of frames can be found by implementing the Joint Comparison technique discussed in Section III-A5 along the series of recorded frames.

3) Masking: The use of motion masking already available in Unity is also streamlined in MoveBox, so that users can use multiple animations for the same avatar. For example a body animation and an independent lip sync animation can both be applied to the same avatar. Automatic masking is also discussed in Section III-A5.

4) Resizing Avatars: The Rocketbox avatars can be scaled to a custom size to match the user's size. The ability to use an appropriately sized avatar improves accuracy of body positions and decreases the risk of avatar limb crossover. When triggered, the resizing feature takes the three dimensional positions of the user's foot and head bones, which are captured by the Azure Kinect or Kinect v2 camera, to measure the user's height. The Rocketbox avatar is then resized, by its scale component, to match the user's height (Figure 7).

IV. VALIDATION

To demonstrate the result of utilizing MoveBox for the Microsoft Rocketbox avatars, we performed a series of MoCaps with Kinect V2 and Azure Kinect inside Movebox, as well as compared the system to other tools such as Cinema4D.

A. Apparatus

1) Kinect V2: Kinect V2 is RGB-D camera based on Time-of-Flight (ToF) that provides dense depth estimations (up to 2 million points per frame) together with color images at a high frame rate (30Hz). This sensor considerably pushed forward several research fields such as: 3D reconstruction [40], [41] or gestures [42]. The natural frame rate of Kinect is only 30Hz, which may blur very fast motions but is quite good for most natural motions of users.

Body tracking SDK calculates the position of 25 joints and can count up to 6 people simultaneously. To compensate for single point of view and resulting occlusion, Kinect implemented a data-driven approach based on a large data-set of motions, and fits the most plausible full body motion to the visual part of the body [43].

However, Kinect V2 recovers the skeleton only with 2 DOF per joint, without roll rotations. This device was originally aimed at consumer use with the Xbox console, in which the users are always facing frontally to the TV. Either seating in the sofa in or standing in front. Avoiding roll rotations allowed the system to deal with a large variation of cloths. But it has undesirable effects when trying to animate 3D avatars, as we will see in this section.

2) Azure Kinect: The current Azure Kinect is also a ToF camera, designed to deal a wider range of motions than Kinect V2. With a 12MP CMOS rolling shutter sensor it can generate 12 million points every 66 milliseconds.

The body tracking SDK implements a more advanced Deep Neural Network (DNN) algorithm that includes the raw color and depth data to estimate the pose of the person from segmentation of body parts and skeletal tracking. It also includes temporal consistency checks. All in all, this post processing provides better joint orientation information as well as reconstruction of body parts that are off-view. In total it provides information on 32 joints.

B. MoCap Validation

We ran our toolbox with Kinect V2 and Azure Kinect and tested how well it performed with upper and lower body motions as well as 360 rotations of participants, occlusions and stand downs.

1) Upper Body and Lower Body: Both devices recover the joints and bones position well. However, Kinect v2 did not have stable recovery of the roll angle of bones around their own axis. While this is not visible when animating avatars with revolution bones (such as a stick figure) it generates erroneous mapping of the mesh and textures along the roll axis for anthropomorphic avatars (See Figure 5). By contrast, the Azure Kinect had better roll reconstruction of the joints along its own axis.

We implemented a model based solution to the roll rotations in Kinect V2 to always default the orientation of the joints around themselves to that on the original avatar. As in the original avatars the relations of rolls between joints are correct. This at least ensures more stability and valid mesh deformations at the joints (Figure 5). This improvement is important because this type of errors on the avatar can reduce the experience dramatically [44].

Lower Body capture performed well on both devices. And after doing the resize of avatars to match the participant the feet were nicely grounded as long as they were on camera sight. However, further collision systems could enhance even more the grounding and make it more robust.

2) Facial and Hand animation: Both Kinect V2 and Azure Kinect suffered from unwanted revolutions on the wrists. To give stability, our solution was to fix the wrist rotations and set them to match that of the lower arm. However, MoveBox also provides hand and finger animation for HMDs that support hands tracking, such as Oculus Quest or HoloLens 2. This works well when the user's hands face toward or away from the HMD. When the hands face sideways or the fingers are tangled, hand tracking may lose accuracy. As a fix, we apply less impact on the avatar's hands when hand-tracking confidence level is low.

For facial animation, currently, MoveBox only implements basic blinking idling and lipsync. But the bone structure of the faces could result in a much larger data set of facial expressions for animations.

3) Occlusions and Full body rotations: The two models of Kinect differ in their resistance for occlusions. While Kinect v2 may react to a joint occlusion by generating an impossible pose, Azure Kinect tend to returns a feasible position.

This difference in robustness is due to the design considerations of both devices. Kinect V2 was designed to sense the pose of a user facing the sensor, which limits the possible motions that could be captured by the user. Whereas Azure Kinect, was designed with a general pose recovery based on a 3D human model that handles such poses with no problem (see supplementary video).

4) Evaluating the reliability 3D Pose from videos: As stated above, currently estimating the 3D pose from videos is designed for offline usage, but it can achieve 30 fps on a RTX2080Ti using [35]. In addition, since the current trained model used videos with visible full body as training data, the predicted 3D pose is reliable only when the full body is visible as shown in Figure 6 (a,b) (b). However, if the lower legs only are occluded, then results of upper body may still be usable (see Figure 6(c)). But if half or less of the body is visible, the results will be unreliable, as seen in Figure 6 (c).

C. End User Validation

A collaborating university lab is conducting research in storytelling authoring tools for AR/VR. As part of their research, several authors are developing stories involving numerous characters that interact with each other and with the viewer. They have been using the Microsoft Rocketbox avatars since their release, first with a Unity asset called Cinema Mocap 2 and now with MoveBox. Working during the COVID-19 pandemic, a single lab member with a both a Kinect 2 and an Azure Kinect at home records action sequences for all characters, while collaborators contribute dialogue and creative direction. In this section we detail their experiences using MoveBox.

1) Animation Quality: The lab does not have sophisticated 3D content creation or animation expertise. As relative novices, their experiences using Cinema Mocap 2 vs. MoveBox are markedly different. The animation smoothness using MoveBox was much higher than they could achieve using Cinema Mocap 2. The quality difference was noticeable when creating animations using Kinect 2 for each tool, but the Azure Kinect with MoveBox produced the highest quality content.

An additional limitation the lab faced was their inability to record motions that were not directly facing the camera, something that is available with Azure Kinect and MoveBox.

2) Lip Sync: Similarly, the researchers did not have support for mapping avatar mouth movement to audio files before switching to MoveBox. Being able to add MoCap animations as well as lip-sync animations from a single toolbox provided a simplified workflow and removed the need for custom development.

3) All in one Tool Box: A major benefit noted was the all in one nature of MoveBox. With the other tool that they had previously used, the user would have to record their motion, rig the exported avatar to a compatible skeletal structure, create an animation controller, and attach the animation clip to the Rocketbox avatar just to view the output. This workflow was very inefficient, especially when creating groupings of multiple avatars each with a unique animation. As shown in figure nine, MoveBox enables the user to record and playback animations in very little steps and without the need for any complicated operations.

4) Creating Stories: Using MoveBox, the creator was able to record an animation for one character, and view that animation while recording additional motion for a second character. This process was repeated numerous times in order to create larger and more complicated segments. The option to view previous animations while recording new segments removes guesswork when synchronizing a multi-avatar sequence.

This mode of recording significantly reduced the amount of time it took to create multi-avatar. An example of a ten second animation with three different avatars waving at one another was measured. It took the lab 7 minutes to record the three avatar motions with Cinema Mocap 2 and it required several recording attempts to achieve pseudo-synchronization due to their inability to view previous animations. With MoveBox, the lab was able to get similar quality motions in 2 minutes, with a single attempt needed per avatar to achieve a higher quality of synchronization across the animation group.

V. Discussion and Limitations

The presented MoveBox system represents a first step in making real-time MoCap technology for the hands of many users. There are many future additions that we hope to add. Starting with support of more sensors, from stereo depth cameras, to even just using a simple color camera, and use computer vision for recovery of the plausible pose.

The current system was tested recording one person in a time. While both Kinect cameras are designed to handle more than one actor, it may increase the complexity and occlusions in the scene. The current system allows a person to record a performance in sync with watching a pre-recorded performance, so many scenes involving multiple users can be captured this way, but we hope to extend the system for multi actors, or maybe multiple actors online which can help multi-user production in these days of social distancing.

By leveraging existing RGB video content, this toolbox can also be used for motion banking. This also opens the possibility of using millions of existing videos as a source for animations and bringing archival footage into virtual environments.

While the MoveBox system provides good body tracking, the facial performance capture is limited. Currently MoveBox has a simplistic lipsync feature that maps the audio spectral amplitude to the mouth movement. As an immediate next step, we would like to implement lip sync capability that applies phoneme information from audio to viseme blendshapes. In addition to lip sync, we would also like to add expressiveness through facial muscles.

With a similar approach the toolbox could also incorporate facial expressions. This should be relatively similar to the rest of the animation pipeline, as the Microsoft Rocketbox avatars are equipped with facial rigging. Since the user's face is obscured by the VR headset, we cannot use solutions that rely on facial video capture [24], and must instead rely on idling animations [34] or facial animation through speech sentiment analysis [45]. Using RGB information for facial expressions would limit the use of this toolbox for users of VR. Hence a more general solution is to have a good idling system for the face combined with lipsync. This approach has been shown to enhance the experience and create enfacement on users [34]. Idling animations add randomly generated animations to the model in order to counter the "robot stare" that puts many models in the uncanny valley.

The idling animation could be further improved by performing sentiment analysis on speech and applying facial animations based on sentiment.

VI. Conclusions

Up until now, motion capture was one of the hallmarks for high budget productions. To generate motions that look realistic while easy to capture, productions invested in high-end motion capture equipment and spaces, or rented expensive hours at motion capture studios. Low budget productions, hobbyists, artists, and researchers, usually had to generate motions using their animation capability, using code to generate procedural animation, or hoping to find useful MoCap recordings that may fit their needs, withing a few data sets of animations containing past recorded motions.

In this paper we present a publicly available open toolset, MoveBox, that is aimed to enable every user, even with a humble budget, to capture as many motions as she needs, acted according to the project needs. Although the generated quality is not equal to the motions generated by high-end equipment, it already enables many more users to access to motion recordings and to start using avatars into their experiments.

We envision a future where researchers of many fields ranging from computer science to psychology, sociology or other may use MoveBox to power interactive avatars during VR experiments.

References

[1] J. N. Bailenson and A. C. Beall, "Transformed social interaction: Exploring the digital plasticity of avatars," in *Avatars at work and play*. Springer, 2006, pp. 1–16.

[2] M. Slater, "Place illusion and plausibility can lead to realistic behaviour in immersive virtual environments," *Phil Trans of the Royal Society B: Biological Sciences*, vol. 364, no. 1535, pp. 3549–3557, 2009.

[3] M. Gonzalez-Franco, M. Slater, M. E. Birney, D. Swapp, S. A. Haslam, and S. D. Reicher, "Participant concerns for the learner in a virtual reality replication of the milgram obedience study," *PloS one*, vol. 13, no. 12, p. e0209704, 2018.

[4] N. Yee, J. N. Bailenson, and N. Ducheneaut, "The proteus effect: Implications of transformed digital self-representation on online and offline behavior," *Communication Research*, vol. 36, pp. 285–312, 2009.

[5] M. Gonzalez-Franco and T. C. Peck, "Avatar embodiment. towards a standardized questionnaire," *Frontiers in Robotics and AI*, p. 74, 2018.

[6] T. C. Peck and M. Gonzalez-Franco, "Avatar embodiment. a standardized questionnaire," *Frontiers in Virtual Reality*, 2020.

[7] 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.

[8] M. Gonzalez-Franco, B. Cohn, E. Ofek, D. Burin, and A. Maselli, "The self-avatar follower effect in virtual reality," in *IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*. IEEE, 2020, pp. 18–25.

[9] B. Spanlang, J.-M. Normand, D. Borland, K. Kilteni, E. Giannopoulos, A. Pomés, M. González-Franco, D. Perez-Marcos, J. Arroyo-Palacios, X. N. Muncunill *et al.*, "How to build an embodiment lab: achieving body representation illusions in virtual reality," *Frontiers in Robotics and AI*, vol. 1, p. 9, 2014.

[10] M. Gonzalez-Franco, D. Perez-Marcos, B. Spanlang, and M. Slater, "The contribution of real-time mirror reflections of motor actions on virtual body ownership in an immersive virtual environment," in *IEEE virtual reality conference (VR)*. IEEE, 2010, pp. 111–114.

[11] B. Spanlang, X. Navarro, J.-M. Normand, S. Kishore, R. Pizarro, and M. Slater, "Real time whole body motion mapping for avatars and robots," in *Proceedings of the 19th ACM Symposium on Virtual Reality Software and Technology*, 2013, pp. 175–178.

[12] M. Gonzalez-Franco, E. Ofek, Y. Pan, A. Antley, A. Steed, B. Spanlang, A. Maselli, D. Banakou, N. Pelechano, S. Orts-Escolano, V. Orvalho, L. Trutoiu, M. Wojcik, M. Sanchez-Vives, J. Bailenson, M. Slater, and J. Lanier, "The rocketbox library and the utility of freely available rigged avatars for procedural virtual humans and embodiment." *Frontiers in Virtual Reality*, 2020.

[13] Z. Tan, Y. Hu, and K. Xu, "Reality based immersive telepresence system for remote conversation and collaboration," in *Int. Workshop on Next Generation Computer Animation Techniques*, 2017, pp. 234–247.

[14] F. Jiang, X. Yang, and L. Feng, "Real-time full-body motion reconstruction and recognition for off-the-shelf vr devices," in *Proc 15th ACM SIGGRAPH Conference on Virtual-Reality Continuum and Its Applications in Industry*, vol. 1, 2016, pp. 309–318.

[15] M. Slater, A. Rovira, R. Southern, D. Swapp, J. J. Zhang, C. Campbell, and M. Levine, "Bystander responses to a violent incident in an immersive virtual environment," *PloS one*, vol. 8, p. e52766, 2013.

[16] "Microsoft rocketbox," https://github.com/microsoft/Microsoft-Rocketbox, 2020.

[17] C. Xu, J. He, X. Zhang, X. Zhou, and S. Duan, "Towards human motion tracking: Multi-sensory imu/toa fusion method and fundamental limits," *Electronics*, vol. 8, p. 142, 01 2019.

[18] "Motionshadow," https://www.motionshadow.com/, 2020.

[19] H. Liu, X. Wei, J. Chai, I. Ha, and T. Rhee, "Realtime human motion control with a small number of inertial sensors," in *Symposium on Interactive 3D Graphics and Games*, ser. I3D '11, 2011, p. 133–140.

[20] K. Ahuja, C. Harrison, M. Goel, and R. Xiao, "Mecap: Whole-body digitization for low-cost vr/ar headsets," in *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*, ser. UIST '19, 2019, p. 453–462.

[21] J. Thies, M. Zollhöfer, C. Theobalt, M. Stamminger, and M. Niessner, "Headon: Real-time reenactment of human portrait videos," *ACM Trans. Graph.*, vol. 37, no. 4, Jul. 2018.

[22] T. Beeler, F. Hahn, D. Bradley, B. Bickel, P. Beardsley, C. Gotsman, R. W. Sumner, and M. Gross, "High-quality passive facial performance capture using anchor frames," *ACM Trans. Graph.*, vol. 30, no. 4, 2011.

[23] L. Valgaerts, C. Wu, A. Bruhn, H.-P. Seidel, and C. Theobalt, "Lightweight binocular facial performance capture under uncontrolled lighting," *ACM Trans. Graph.*, vol. 31, no. 6, Nov. 2012.

[24] A. Tewari, M. Zollhofer, H. Kim, P. Garrido, F. Bernard, P. Perez, and C. Theobalt, "Mofa: Model-based deep convolutional face autoencoder for unsupervised monocular reconstruction," in *The IEEE International Conference on Computer Vision (ICCV) Workshops*, Oct 2017.

[25] "Salsa lip-sync, emoter, and eyes - for unity," 2020. [Online]. Available: https://crazyminnowstudio.com/unity-3d/lip-sync-SALSA/

[26] "Oculus Lipsync for Unity Development," 2020. [Online]. Available: https://developer.oculus.com/documentation/unity/audio-ovrlipsync-unity/

[27] J. Lee, J. Chai, P. S. Reitsma, J. K. Hodgins, and N. S. Pollard, "Interactive control of avatars animated with human motion data," in *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, 2002, pp. 491–500.

[28] B. Spanlang, "Halca hardware accelerated library for character animation," *Barcelona: Universitat de Barcelona, EVENT Lab*, 2009.

[29] B. Spanlang, D. Corominas, and M. Slater, "A virtual whole body system," in *CHI Whole Body Interaction Workshop*, 2010.

[30] M. Gillies and B. Spanlang, "Comparing and evaluating real time character engines for virtual environments," *Presence: Teleoperators and Virtual Environments*, vol. 19, no. 2, pp. 95–117, 2010.

[31] D. Holden, T. Komura, and J. Saito, "Phase-functioned neural networks for character control," *ACM Trans. Graph.*, vol. 36, no. 4, Jul. 2017.

[32] X. B. Peng, P. Abbeel, S. Levine, and M. van de Panne, "Deepmimic," *ACM Transactions on Graphics*, vol. 37, no. 4, p. 1–14, Aug 2018.

[33] P. Abtahi, M. Gonzalez-Franco, E. Ofek, and A. Steed, "I'm a giant: Walking in large virtual environments at high speed gains," in *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, 2019, pp. 1–13.

[34] M. Gonzalez-Franco, A. Steed, S. Hoogendyk, and E. Ofek, "Using facial animation to increase the enfacement illusion and avatar self-identification," *IEEE Transactions on Visualization and Computer Graphics*, vol. 26, no. 5, pp. 2023–2029, 2020.

[35] M. Kocabas, N. Athanasiou, and M. J. Black, "Vibe: Video inference for human body pose and shape estimation," in *IEEE Conf on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 5252–5262.

[36] M. Loper, N. Mahmood, J. Romero, G. Pons-Moll, and M. Black, "Smpl: A skinned multi-person linear model," vol. 34, no. 6, Oct. 2015.

[37] N. De la Peña, P. Weil, J. Llobera, E. Giannopoulos, A. Pomés, B. Spanlang, D. Friedman, M. V. Sanchez-Vives, and M. Slater, "Immersive journalism: immersive virtual reality for the first-person experience of news," *Presence: Teleoperators and virtual environments*, vol. 19, no. 4, pp. 291–301, 2010.

[38] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh, "Openpose: realtime multi-person 2d pose estimation using part affinity fields," *arXiv preprint arXiv:1812.08008*, 2018.

[39] H. Li, L. Trutoiu, K. Olszewski, L. Wei, T. Trutna, P.-L. Hsieh, A. Nicholls, and C. Ma, "Facial performance sensing head-mounted display," *ACM Transactions on Graphics (ToG)*, vol. 34, no. 4, 2015.

[40] J. Smisek, M. Jancosek, and T. Pajdla, "3d with kinect," in *Consumer depth cameras for computer vision*. Springer, 2013, pp. 3–25.

[41] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison *et al.*, "Kinectfusion: real-time 3d reconstruction and interaction using a moving depth camera," in *Proceedings of the 24th annual ACM symposium on User interface software and technology*, 2011, pp. 559–568.

[42] Z. Ren, J. Yuan, J. Meng, and Z. Zhang, "Robust part-based hand gesture recognition using kinect sensor," *IEEE transactions on multimedia*, vol. 15, no. 5, pp. 1110–1120, 2013.

[43] J. Shotton, A. Fitzgibbon, A. Blake, A. Kipman, M. Finocchio, B. Moore, and T. Sharp, "Real-time human pose recognition in parts from a single depth image," in *CVPR*. IEEE, June 2011.

[44] G. Padrao, M. Gonzalez-Franco, M. V. Sanchez-Vives, M. Slater, and A. Rodriguez-Fornells, "Violating body movement semantics: Neural signatures of self-generated and external-generated errors," *Neuroimage*, vol. 124, pp. 147–156, 2016.

[45] R. Hoegen, D. Aneja, D. McDuff, and M. Czerwinski, "An end-to-end conversational style matching agent," in *Proceedings of the 19th ACM International Conference on Intelligent Virtual Agents*, 2019, pp. 111–118.