Real-time Expressive Avatar Animation Generation based on Monocular Videos

Wenfeng Song
Computer School, Beijing Information Science and Technology University

Xianfei Wang
Computer School, Beijing Information Science and Technology University

Yang Gao*
State Key Laboratory of Virtual Reality Technology and Systems, Beijing Advanced Innovation Center for Biomedical Engineering, Beihang University

Aimin Hao
State Key Laboratory of Virtual Reality Technology and Systems, Beihang University Beijing, Peng Cheng Laboratory, Shenzhen, China

Xia Hou
Computer School, Beijing Information Science and Technology University

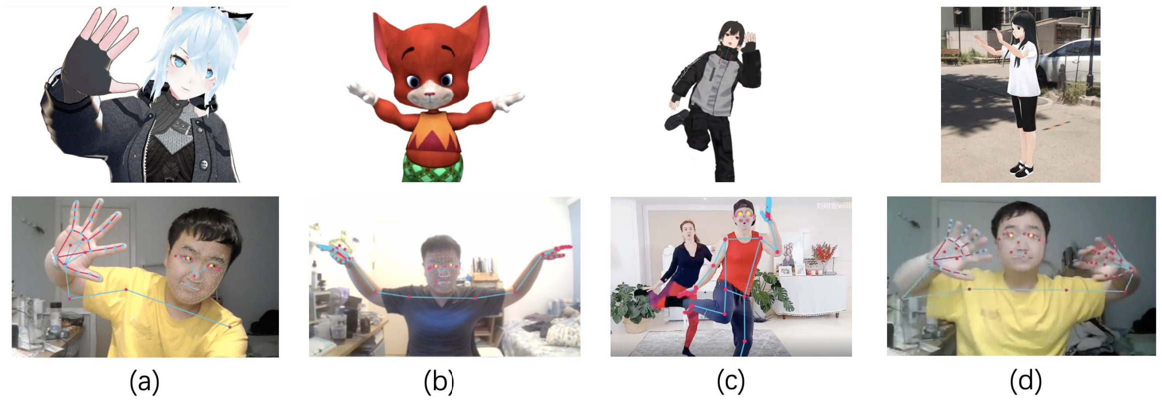

Figure 1: Virtual avatar animation (top) and corresponding user's video frames (bottom). (a) & (b): Facial expression and gesture animations; (c) & (d): Full body tracking animations in VR and AR modes.

Abstract

The technologies for generating real-time animated avatars are very useful in the fields of VR/AR animation and entertainment. Most of the existing studies, however, always require the technology of time-consuming motion capture at high cost. This paper proposes an efficient lightweight framework of dynamic avatar animation, which can generate all the facial expressions, gestures, and torso movements properly in real time. The entire technique is driven only by monocular camera videos. Specifically, the 3D posture and facial landmarks of the monocular videos can be calculated by using Blaze-pose key points in our proposed framework. Then, a novel adaptor mapping function is proposed to transform the kinematic topology into the rigid skeletons of avatars. Without the dependency of a high-cost motion capture instrument and also without the limitation of the topology, our approach produces avatar animations with a higher level of fidelity. Finally, animations, including lip movements, facial expressions, and limb motions, are generated in a unified framework, which allows our 3D virtual avatar to act exactly like a real person. We have conducted extensive experiments to demonstrate the efficacy of applications in real-time avatar-related research. Our project and software are publicly available for further research or practical use (https://github.com/xianfei/SysMocap/).

Index Terms: Computing methodologies—Computer graphics—Animation—Motion capture; Human-centered computing—Human computer interaction (HCI)—Interactive systems and tools

*E-mail: gaoyangvr@buaa.edu.cn

1 Introduction

Expressive avatar animation has benefited from the advancement of data-driven deep learning and motion capture technologies, which have a wide range of applications in virtual reality (VR) fields. In particular, human-driven animated avatars with expressive motions are in high demand to act as virtual characters in telecommunication, virtual try-on, 3D games, or entertainment production, replacing time-consuming and expensive manual animation production [1].

To precisely capture the motion, existing methods have designed a variety of sensors, including facial capture [2,3], expensive 3D scanners [4], digital hand gloves [5], multiple view camera settings [6], wearable IMU [7], and depth cameras [8,9], etc. These methods could accurately capture the motion-related signals beyond RGB appearance, with high-quality human model reconstruction. However, the motion capture instruments are expensive and time-consuming. Meanwhile, these settings require a complex setup environment, which limits their utility. To tackle these problems, monocular video-driven motion capture is used for animating the avatar to look like a real person. Some methods [10,11] are based on parameterized models to reconstruct geometry and texture from a single image, which can recover human shapes with fixed topology and a coarse skinning human model. However, they are unable to generate both detailed geometrical and realistic textures. Furthermore, the reconstructed unseen regions tend to be unfaithfully smooth due to the limited observation. On the other hand, some recent works proposed estimating human motions via skeleton [12]. This work could efficiently get the motions and even meet real-time efficiency. However, the skeletons lack the details of the human characters, and far from the high fidelity requirement, these cannot be employed to represent the human avatar characters. To summarize, a lightweight framework is urgently required to accurately drive arbitrary rigid avatars that do not rely on high-performance systems or complex algorithms to predict individualized avatar animations.

However, this task is fraught with difficulties. First, the variety of standard motion capture datasets leads to different kinematic topologies. In the course of avatar animating, the operations required to make different virtual characters perform the same action are different. However, the widely utilized mixamo [13] motion has 14 key points without the hands and face. And Kinect-based datasets have 16 key points, so the critical points and topology cannot be unified. Furthermore, avatar animation applications necessitate real-time interaction and environmental immersion. However, due to the vast scale of the predicted parameters, the estimation of avatars and animations is unavoidably expensive and time-consuming. In addition, to improve the immersion feeling of the participant avatar, the driven method should provide expressive human motions in terms of facial expressions, detailed gestures with fingers, and sufficient body keypoints. But these three parts are usually calculated separately, which makes it hard to show animations in a single picture.

In order to unify the various standards in motion representation, we propose a novel instantaneous topology that encompasses the main control points of human movements and expressions. This innovative topology provides a proxy between the video-driven keypoints and avatars. Meanwhile, to tackle the proxy difficulty of transformation between the matrix and Eula angles, we propose a matrix transformation algorithm to match the topology. Furthermore, we propose a plug-in module to connect the Blazepose keypoints [12], which promotes avatar animation to improve efficiency and reduce computation cost. Finally, we combine the entire process, including motion estimation and avatar re-targeting, into a unified lightweight framework that can be used in AR, VR, and MR applications, as shown in Fig. 1. To unify and simplify the entire process, we present a lightweight yet efficient adapter with a well-designed proxy skeleton. Our salient contributions can be summarized as follows:

- We reveal the relationship between the avatars and the video Blazeposes and formulate the transition between the two typologies, which gives rise to a simple yet effective driven method for obtaining real-time animation dynamics.

- We propose a lightweight adaptor based on four different types of the widely used standards and a proxy topology of human motion that can adapt to commonly used avatars through single-camera video accurately and in real time.

- We design a flexable unified video-driven framework for animated avatars, which can provide expressive animated keypoints, including facial expressions, gestures, and body movements. Comprehensive experiments have demonstrated the superiority of our framework.

2 Real Time Expressive Avatar Animation Generation Framework

The goal of our work is to animate the high-fidelity avatar in real-time from a single RGB camera. Fig. 2 shows the pipeline of our framework. There are three main components to our work: (1) Monocular video motion capture. We track the human's motion through multiple key points estimation models, including face, hands, and body. We aim to reduce the computation cost via low-cost tracking methods, e.g., BlazePose. Different from Blazepose, we unify the hands, face, and body positions via three off-the-shelf models [12, 14, 15], which has the advantage of high efficiency and robustness. (2) An adaptor to convert the raw motion skeletons into a proxy topology, which is consistent with the standard rigid characters. In this component, we first design a simplified proxy kinematic skeleton. The skeleton is extracted from standard VRM without the hands and head keypoints. After that, we further convert the rotation matrix of the motion data to Euler angles. In this manner, the BlazePose keypoints are converted to those of mediate kinematic skeletons, which can then be fed with avatars. (3) Kinematic keypoints driving the virtual character.We provide a proxy skeleton map of critical points that is compatible with the name scheme and coordinate system standards of VRM, Mixamo, and other traditional format bones.

2.1 Motion Capture from Monocular Videos

To achieve real time efficiency, we employ the BlazePose [12], which is a widely used and highly comprehensive algorithm on consumer-end computers. We briefly revisit the models in BlazePose. The algorithm has three different types of convolutional neural network (CNN) models: the lite model, the full model, and the heavy model.

In addition, the BlazePose method can also be used with the real-time hand detection algorithm [14] and the real-time face recognition algorithm BlazeFace [15] to generate virtual character facial and hand movements. The BlazePalm algorithm detects information about each hand joint in real time, while the BlazeFace algorithm detects the facial contours and joint points. Usually, an expression recognition algorithm is also required to determine the expression of the actor in the video to achieve better facial effects of the virtual character [16].

Since BlazePose only employs rotation information while driving the virtual character skeleton, even if the motion capture accuracy is good enough, the virtual character can only be driven to recover its approximate position. This is because the model does not match the height, arm length, or leg length of the character in the video. As a result, using position data to control the virtual character may result in improper deformation and distortion. To this end, we unify the facial, hand and body pose models as a whole pipeline.

2.2 Proxy Skeletons Design

In this part, we make a proxy skeleton in a mediate topology, taking into account Blazepose and the way avatars move. Our proxy skeletons are inspired by the 3D model file skeletal architecture of most human avatars. Theoretically, our work resembles the skeletal structure of the human body, which adheres to or can be easily adjusted to the skeletal structure of most 3D models of humanoids in experiments. We illustrate the differences between Blazepose and Avatar skeleton topologies in Fig. 3.

On the one hand, the BlazePose provides a skeleton style that does not contain a spine, but the majority of virtual image skeletons contain one. Furthermore, only the backbone nodes store location data; the remainder of the nodes just keep track of rotations relative to the parent node. To deduce the spine information, we must use the four spots on the torso. The raw motion from BlazePose is difficult because the wrist has just four points. This could result in an inability to identify fine details, such as whether the back is straight when leaning over.

On the other hand, virtual character 3D models usually have different formats (e.g. VRM, FBX, GLTF, etc.), coordinate systems, and other parameters. In addition, to further feed the skeletal mapping module in the next stage, we specify that this step generates an intermediate skeletal form. The right of Fig. 3 represents this skeletal form, which contains 12 key bones, namely Hips, Neck, Chest, Spine, RightUpperArm, RightLowerArm, LeftUpperArm, LeftLowerArm, LeftUpperLeg, LeftLowerLeg, RightUpperLeg, RightLowerLeg. The typical VRM is simplified to 12 bones for the following reasons: (1) These key bones could control most of the human body's animations, including the hands, legs, and torso. (2) The proxy skeleton connects the BlazePose and avatar skeletons, allowing the former topology to be converted with minimal transformation

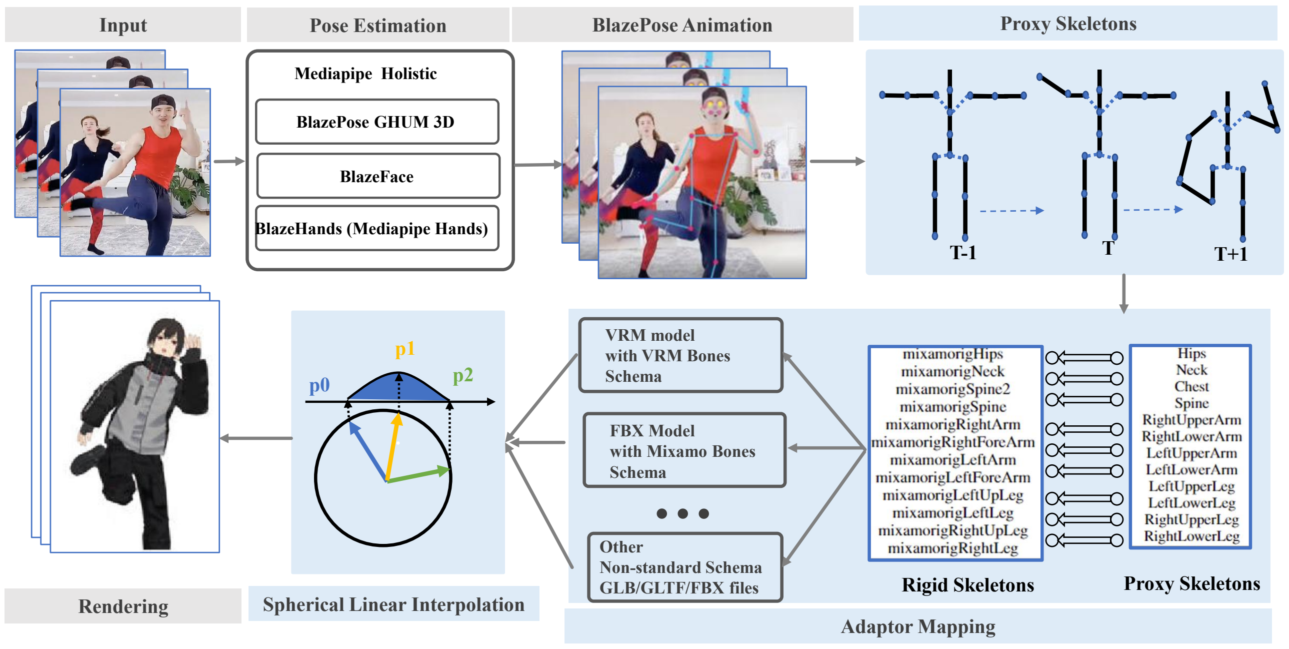

Figure 2: An overview of our unified avatar animation framework. (1) Motion capture in monocular video. This module is realized through a unified face, hand, and body keypoint estimation model. (2) An adaptor that converts the raw motion skeletons into a proxy topology that matches the rigid characters. In this component, we first create a simplified proxy kinematic skeleton by deleting the hand and head keypoints from the standard VRM. After that, we convert the motion data's rotation matrix to euler angles. In this way, the keypoints of BlazePose are converted to mediate kinematic skeletons, which will be further fed with avatars. (3) Virtual character driven by a kinematic key points map. We provide a proxy skeleton map of keypoints, which is suitable for VRM, Mixamo, and other classical format bones' naming schema and coordinate system standards.

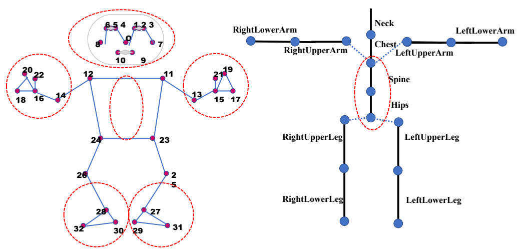

Figure 3: Comparisons between the BlazePose (left) and our proxy skeleton (right). We simplify the hands, face and feet structures, and improve the torso to the standard of rigid animation.

loss errors.

2.3 Adaptor Mapping for Avatar Animations

Through the above measurements, models with different skeletal naming styles can be unified into a consistent naming rule. We will discuss how to drive different formats of virtual character 3D models. Our proxy skeleton form is based on the VRM coordinate system. There is no need to convert this type of model. Here, we calculate the rotation matrix to Euler angles. The following formula is used to calculate the rotation matrix M to Euler angles.

$$M(x,y,z) = \begin{cases} f1(x,y,z) \\ f2(x,y,z) \\ f3(x,y,z) \end{cases} \tag{1}$$

f1(x, y, z), f2(x, y, z), and f3(x, y, z) denote mapping the proxy skeletons with three different types of avatars, such as VRM, Mixamo, and so on. These mapping functions can be easily extended to more rigid avatars with similar typologies.

Moreover, different standards of virtual characters show different coordinate systems. For example,when the users plan to make virtual characters with different skeleton kinematic typologies in T-pose to a novel pose, extending their right forearms forward simultaneously, the virtual characters need different adjustments. Hence, our adaptor calculates the rotation angle of the target bone using the function expression. We also use the function to control the rotation of the virtual character's bones. Afterward, inter-frame interpolation in the time domain (spherical linear interpolation) can solve the problem that lower motion capture speeds cannot match the high frame rates for rendering.We employ the (q_0, q_1) as the two frame's motion quaternion format. which is converted from Euler angles.

After the foregoing procedures, to drive different types of avatars, we first need to traverse the entire table to find the driving bones by name matching to find the driving skeleton. Then we find the corresponding bones according to their corresponding bone names and use the expression parsing function to parse the set function expressions ($f1, f2, f3$) for the coordinate system transformation.

2.4 Implementation Details

To simplify the further use of our contributions, we design a desktop application to address virtual character management, motion capture, driving virtual characters, forwarding motion data, etc. It can lay the groundwork for virtual live broadcasting, AR/VR development, and other scenarios.

For the sake of development convenience and system portability, we split the software into three modules: the motion capture module; the user graphical interface module (including renderer and virtual character management); and the real-time avatar-driven module.

The motion capture module uses an event-driven design optimized for real-time performance, with Mediapipe [17] to execute the BlazePose GHUM 3D algorithm. The camera and video file decoder act as event sources, generating an event for each frame acquired and sending the event to the neural network to calculate and extract

Table 1: Technical Specifications of Test Groups

	Group 1	Group 2	Group 3	Group 4
CPU	Intel Core i7-4790K	Intel Core i7-4790K	Intel Core i7-12700H	Intel Core i7-12700H
GPU	NVIDIA GTX770	NVIDIA GTX770	NVIDIA RTX3060	Intel Iris Xe
RAM	16GB DDR3	16GB DDR3	16GB DDR5	16GB DDR5
OS	macOS 11.6	Windows 11	Windows 11	Windows 11

Table 2: Robustness Evaluation of Our Method

Scene	Group 1	Group 2	Group 3	Group 4
Dance	14.9fps	8.6fps	22.1fps	13.7fps
Aerobics	15.2fps	7.8fps	22.3fps	13.9fps
Singing	18.1fps	12fps	25.6fps	15.2fps

the position of each skeletal point in it. Because the neural network needs to work in real time, it only reads the most recent frame. If the video frames can't be processed because of a lack of performance, they will be thrown away.

The renderer uses three.js, which is based on WebGL technology, and takes the corresponding loader to load 3D models in different formats (VRM, GLB/GLTF, and FBX).

Our system meets the real-time level performance requirements of low latency and high reliability, and it has some scalability for the following real-time live streaming solutions that can be used on both mainstream live streaming platforms and independent live streaming platforms. It lays the foundation for the following real-time live streaming solutions applicable to mainstream live streaming platforms and the self-developed AR/VR virtual image real-time live streaming function.

2.5 Performance Assessment

In this section, we conduct experiments to evaluate the efficiency of our system compared with other state-of-the-art methods. Because virtual character forwarding and related applications such as live streaming necessitate real-time processing and face and hand detection algorithms for improved presentation, the model's combined performance (with face and hand detection algorithms) must be evaluated in this topic. In this study, three groups of computers with different hardware configurations and operating systems are selected to compare the performance of this software, and a higher complexity model is used for the model complexity with the technical specifications shown in Table 1.

In the program designed in our study, dance videos (which contain a large number of complex movements, and the performance of the model is usually very demanding), aerobics videos (which are simpler than dance movements but still contain a large number of movements, and the performance of the model is usually demanding), and singing videos (which mainly focus on the facial movements of the upper body and do not contain the lower body, and most of these are used by most virtual anchors) as test data. The program's FPS performance monitor is used to record the frame rate, and an average of ten samples is used to figure out the frame rate.

The test results in Table 2 show that there is no significant difference in performance on the full-body video (calisthenics and dancing). However, the performance on half-body video is significantly better than that on full-body video. And the runtime performance is higher on Windows than on macOS. This may be related to the fact that NVIDIA graphics cards on macOS also need to use Apple's Metal API and cannot use the official NVIDIA drivers and CUDA. The overall results are generally satisfactory when combined with

Table 3: CPU Utilization of Our Method

Scene	Group 1	Group 2	Group 3	Group 4
Dance	15%	12%	8%	8%
Aerobics	14%	11%	8%	7%
Singing	14%	12%	9%	7%

Table 4: GPU Utilization of Our Method

Scene	Group 1	Group 2	Group 3	Group 4
Dance	51%	47%	29%	61%
Aerobics	52%	47%	30%	60%
Singing	54%	46%	28%	59%

Table 5: Performance Comparison with SOTA Methods on 3DPW

Algorithm	PA-MPJPE	MPJPE	MPVPE
TCMR [18]	55.8	95.0	111.3
PyMAF [19]	58.9	92.8	110.1
VIBE [11]	56.5	93.5	113.4
HMMR [20]	72.6	116.5	139.3
Doersch et al. [21]	74.7	-	-
Sun et al. [22]	69.5	-	-
Ours	57.5	104.6	118.3

the time-domain intra-frame interpolation algorithm (spherical linear interpolation) used for output rendering. Our software has a relatively small overhead on system performance, Table 3 shows the CPU usage and Table 4 shows the GPU usage.

2.6 Comparisons with State-of-the-art Methods

To compare with other high performance methods, we have chosen state-of-the-art methods including TCMR [18] and PyMAF [19] (not real time methods), MocapNET [23–25] and BlazePoze [12] (real time methods). Since the TMCR and PyMAF algorithms are non-real-time algorithms, the action information can only be obtained after inputting a video file, while the MocapNET and BlazePoze 3D algorithms (ours) are real-time algorithms that can use a camera or a video file as an input source.

In the algorithm selection test, dance videos and fitness videos are selected for testing because the movements appearing in the dance and fitness videos are more complex and can better test the accuracy of the algorithms. The testing environment for TMCR and PyMAF is the Google Colab online machine learning platform, and the testing environment for BlazePose 3D and MocapNET is the local machine, on which BlazePose 3D runs the Heavy model. The hardware configuration of this machine includes: Intel Core i7-12700H CPU, 16GB RAM, Intel Iris Xe Graphics GPU, Windows 11 or Ubuntu 18.04 on WSL (Windows Subsystem for Linux). A dance video is used as the test because it usually contains more complex movements, which can better test the performance of the algorithm. The total elapsed time is recorded as the total time for the non-real-time algorithm, the frame rate is recorded as the frame rate for the real-time algorithm, and the test is conducted five times to take the average as the result.

The quantity results in Table 5 demonstrate our method could achieve comparable performance in terms of accuracy measurement. To be noted that, our system is not the best in accuracy but it is the most efficient. There should be a trade-off between accuracy and the efficiency in practical system. Moreover we have compared results in the highly complex gestures or finger poses in Fig. 4, the results show our method achieved reasonable poses.

The efficiency results are shown in Table 6, from which it can be seen that the non-real-time algorithms TCMR and PyMAF consume much more time than the real-time algorithms (this test result may also be related to the performance of the free version of Colab, however, both of them have NVIDIA Tesla professional GPU acceleration), but even if these two algorithms consume more time, in the complex movements in the dance for the front-back relationship between the limbs and the torso , there are some problems with the inference of the right hand in relation to the body, as the left hand is behind the body in the video, while the left hand of the generated character model is in front of the body there are still some problems with the inference of the left hand position, as the left arm of the character is behind the body in the original video. All four

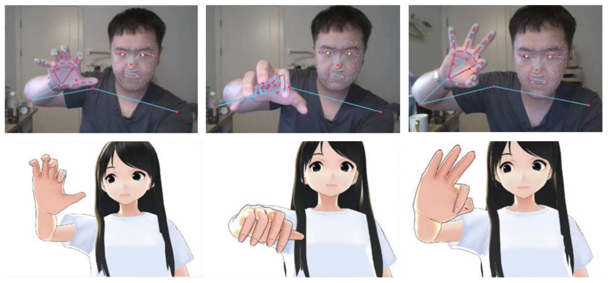

Figure 4: Results on hand gestures and finger poses.

Table 6: Performance Comparison with State-of-the-art Methods.

Algorithm	TCMR	PyMAF	MocapNet	Ours
Environment	Google Colab		Local Machine	
Time(min)	19.82	23.26	-	-
FPS	-	-	2.3	12.8

algorithms encounter such problems to varying degrees.

Typically, algorithm accuracy, time efficiency, and hardware requirements are the three aspects that cannot be combined. Recognizing more accurately usually requires more processing time, and accurate real-time algorithms require better hardware.

2.7 User Study

We have conducted a user study with 40 participants to evaluate the avatar animations compared with other state-of-the-art methods. We design the user study in terms of realistic animation, fluency, and the user's preference of the animated avatars. We model the evaluations and experience quantitatively as 5 levels, from 1 ("No satisfaction") to 5 ("High satisfaction"), following the previous work [1].

Participants. The user study is conducted online using survey sites in order to reach a wide and diverse population. We solicited the help of 40 volunteers from a social networking site and from college, including 20 males and 20 females, they are VTuber users without high-level professional animation knowledge background. As shown in Table 7, avatar users come in a variety of shapes and sizes. Each participant took an average of 10 minutes to complete the survey. Of the participants, 47.5% are youth from 21 to 30 years old, 30% are pioneers ranging from 11 to 20 years old. Most of them are university students, engineers, and VTuber users.

Results and Discussion. We invite the participants to evaluate 20 avatars, including facial expressions, body and hand details, of which 10 avatars are ours and 10 are from existing works. The avatars can do things like talk, sing, dance, and so on, which are the most common animations. The application scenes include the indoor, outdoor, and light-changing environments. Table 7 shows that our avatar animations get obviously higher scores, the majority of participants selected our animated avatars as the most engaging natural avatar.

2.8 Applications

Our framework can be used for live streaming, film and television production, etc., and can be applied to live streaming of 3D virtual characters on VR/AR devices.

Table 7: Evaluation of User Study

	Methods	Realness	Fluency	Preference
Male	**Our System**	**4.5**	**4.2**	**4.0**
	ThreeDPoseTracker	3.8	4.1	3.8
	K Tuber	3.7	3.9	3.7
Female	**Our System**	**4.2**	**4.1**	**3.9**
	ThreeDPoseTracker	3.9	3.6	3.7
	K Tuber	3.5	3.2	3.5

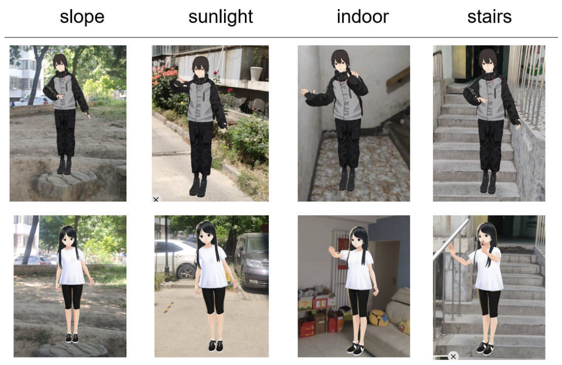

Figure 5: AR-based avatar animation on mobile phone via WebXR

2.8.1 Live Streaming and Recording with OBS

We employ the open broadcaster software (OBS), a free, open-source, and cross-platform screen-casting and streaming app. It can stream videos to any RTMP-supporting destination, including YouTube, Twitch, Instagram, and Facebook.

Because we use Web technology the application development, virtual characters and their actions can be seen in real time on the Web with just a browser. Developed from OBS, our system is user-friendly with simple operations.

2.8.2 Display on VR/AR devices

There is no corresponding live platform for VR/AR virtual characters' live streaming yet, but with the popularity of the metaverse concept, VR/AR live streaming will become a future trend. To this end, we also design a platform for our application to achieve AR/VR action live streaming with the help of the WebXR interface. Since Three.js currently supports the WebXR API, only a slight modification of the Three.js renderer is needed. We make relevant modifications to Three.js to make it can support the WebXR API, so it can be used in an AR and VR environment respectively.

In our framework, we have designed a protocol for forwarding motion capture data to VR/AR devices over the network. Since the motion capture process is done on the computer and the rendering is only done on the VR/AR device, the performance requirements of the VR/AR device are very low.

Instead of transferring the rendered screen, we used a way to send the model file and action data, which reduces the amount of data sent to other devices such as AR/VR, enabling us to obtain higher performance and network communication stability without sacrificing fluency. The motion capture frame rate is typically between 20–30 fps. The rendering frame rate on VR/AR devices can reach 60+ fps with interpolation (depending on the refresh rate of devices).

In order to make the virtual character better displayed on the real ground, it is also necessary to calibrate the ground height by adjusting the y-axis offset and calculating the lowest point of the model to make the virtual character always stand on the ground. In this paper, we use the Android phone with ARCore to observe the virtual character and its movement in different environments. In Fig. 5, we choose different terrains (e.g. tarmac, stairs, dirt slope) and different light environments (e.g. shade, sunlight, indoors) to test the virtual character and have good results.

Our framework also supports VR devices, such as HTC Vive, Oculus Rift, Google Cardboard, HoloLens, etc. In order to further reduce the user threshold, we select one of the lowest-cost cardboard HMD to describe. The cardboard HMD consists of cheap and common materials such as cardboard, convex lens, leather straps, and velcro [26], and requires the phone to be placed into the HMD.

3 Conclusion

This research provides a lightweight framework for dynamic avatar animation that is economical and produces higher-fidelity avatar animations. The entire technique is driven only by monocular camera videos without the dependency of a high-cost motion capture instrument and also without the limitation of the topology, allowing our 3D virtual avatar to behave exactly like a real person. Our system is compatible with VR/AR platform applications, and we provide the open source to interested researchers. The potential improvement of our method mainly lie in the pose estimation module. When the input video stream is severely obstructed, our technique will fail. Also, to improve performance and overall smoothness, some movements, complex gestures, and finger poses aren't as accurate as they would be with other motion capture algorithms. However, they can still show the streamer's movements. In our future work, we will improve the efficiency of model compression. In the meantime, further improving the precision and animation effect is our unremitting pursuit.

Acknowledgments

This paper is supported by National Natural Science Foundation of China (62102036,62002010), Beijing Natural Science Foundation (4222024), R&D Program of Beijing Municipal Education Commission (KM202211232003), Open Project Program of State Key Laboratory of Virtual Reality Technology and Systems, Beihang University (No.VRLAB2022A02), Beijing Natural Science Foundation under Grant (4214066), Beijing Advanced Innovation Center for Biomedical Engineering under Grant ZF138G1714, Research Unit of Virtual Body and Virtual Surgery Technologies, Chinese Academy of Medical Sciences, 2019RU004.

References

[1] Man To Tang, Victor Long Zhu, and Voicu Popescu. Alterecho: Loose avatar-streamer coupling for expressive vtubing. In *2021 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 128–137. IEEE, 2021.

[2] John P Lewis, Ken Anjyo, Taehyun Rhee, Mengjie Zhang, Frederic H Pighin, and Zhigang Deng. Practice and theory of blendshape facial models. *Eurographics (State of the Art Reports)*, 1(8):2, 2014.

[3] Chen Cao, Yanlin Weng, Shun Zhou, Yiying Tong, and Kun Zhou. Facewarehouse: A 3d facial expression database for visual computing. *IEEE Transactions on Visualization and Computer Graphics (TVCG)*, 20(3):413–425, 2013.

[4] Daniel Cudeiro, Timo Bolkart, Cassidy Laidlaw, Anurag Ranjan, and Michael J. Black. Capture, learning, and synthesis of 3d speaking styles. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 10101–10111, June 2019.

[5] Breannan Smith, Chenglei Wu, He Wen, Patrick Peluse, Yaser Sheikh, Jessica K Hodgins, and Takaaki Shiratori. Constraining dense hand surface tracking with elasticity. *ACM Transactions on Graphics (TOG)*, 39(6):1–14, 2020.

[6] Yuxiang Zhang, Zhe Li, Liang An, Mengcheng Li, Tao Yu, and Yebin Liu. Lightweight multi-person total motion capture using sparse multi-view cameras. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5560–5569, 2021.

[7] Abhishek Sharma and Eric Rombokas. Improving imu-based prediction of lower limb kinematics in natural environments using egocentric optical flow. *IEEE Transactions on Neural Systems and Rehabilitation Engineering (TNSRE)*, 30:699–708, 2022.

[8] Ruizhi Shao, Hongwen Zhang, He Zhang, Mingjia Chen, Yanpei Cao, Tao Yu, and Yebin Liu. Doublefield: Bridging the neural surface and radiance fields for high-fidelity human reconstruction and rendering. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[9] Tao Yu, Zerong Zheng, Kaiwen Guo, Pengpeng Liu, Qionghai Dai, and Yebin Liu. Function4d: Real-time human volumetric capture from very sparse consumer rgbd sensors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5746–5756, 2021.

[10] Zhe Li, Tao Yu, Zerong Zheng, Kaiwen Guo, and Yebin Liu. Posefusion: Pose-guided selective fusion for single-view human volumetric capture. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14162–14172, 2021.

[11] Muhammed Kocabas, Nikos Athanasiou, and Michael J Black. Vibe: Video inference for human body pose and shape estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5253–5263, 2020.

[12] Valentin Bazarevsky, Ivan Grishchenko, Karthik Raveendran, Tyler Zhu, Fan Zhang, and Matthias Grundmann. Blazepose: On-device real-time body pose tracking. *CoRR*, abs/2006.10204, 2020.

[13] Sue Blackman. Rigging with mixamo. In *Unity for Absolute Beginners*, pages 565–573. Springer, 2014.

[14] Fan Zhang, Valentin Bazarevsky, Andrey Vakunov, Andrei Tkachenka, George Sung, Chuo-Ling Chang, and Matthias Grundmann. Mediapipe hands: On-device real-time hand tracking, 2020.

[15] Valentin Bazarevsky, Yury Kartynnik, Andrey Vakunov, Karthik Raveendran, and Matthias Grundmann. Blazeface: Sub-millisecond neural face detection on mobile gpus, 2019.

[16] Vasileios Choutas, Georgios Pavlakos, Timo Bolkart, Dimitrios Tzionas, and Michael J. Black. Monocular expressive body regression through body-driven attention. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *European Conference on Computer Vision (ECCV)*, pages 20–40, Cham, 2020. Springer International Publishing.

[17] Camillo Lugaresi, Jiuqiang Tang, Hadon Nash, Chris McClanahan, Esha Uboweja, Michael Hays, Fan Zhang, Chuo-Ling Chang, Ming Guang Yong, Juhyun Lee, Wan-Teh Chang, Wei Hua, Manfred Georg, and Matthias Grundmann. Mediapipe: A framework for building perception pipelines, 2019.

[18] Hongsuk Choi, Gyeongsik Moon, Ju Yong Chang, and Kyoung Mu Lee. Beyond static features for temporally consistent 3d human pose and shape from a video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1964–1973, June 2021.

[19] Hongwen Zhang, Yating Tian, Xinchi Zhou, Wanli Ouyang, Yebin Liu, Limin Wang, and Zhenan Sun. Pymaf: 3d human pose and shape regression with pyramidal mesh alignment feedback loop. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2021.

[20] Angjoo Kanazawa, Jason Y Zhang, Panna Felsen, and Jitendra Malik. Learning 3d human dynamics from video. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pages 5614–5623, 2019.

[21] Carl Doersch and Andrew Zisserman. Sim2real transfer learning for 3d human pose estimation: motion to the rescue. *Advances in Neural Information Processing Systems (NIPS)*, 32, 2019.

[22] Yu Sun, Yun Ye, Wu Liu, Wenpeng Gao, Yili Fu, and Tao Mei. Human mesh recovery from monocular images via a skeleton-disentangled representation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5349–5358, 2019.

[23] Ammar Qammaz and Antonis A Argyros. Towards holistic real-time human 3d pose estimation using mocapnets. In *British Machine Vision Conference (BMVC)*. BMVA, November 2021.

[24] Ammar Qammaz and Antonis A. Argyros. Occlusion-tolerant and personalized 3d human pose estimation in rgb images. In *IEEE International Conference on Pattern Recognition (ICPR), (to appear)*, January 2021.

[25] Ammar Qammaz and Antonis A Argyros. Mocapnet: Ensemble of snn encoders for 3d human pose estimation in rgb images. In *British Machine Vision Conference (BMVC)*, Cardiff, UK, September 2019. BMVA.

[26] Ramakrishna Perla and Ramya Hebbalaguppe. Google cardboard dates augmented reality : Issues, challenges and future opportunities, 2017.