Embedding Segmented Humans into 2D Games for Gamified Exercises

Christopher Schultze
christopher.schul1@ucalgary.ca
University of Calgary
Calgary, Alberta, Canada

Diwij Dev
diwij.dev@ucalgary.ca
University of Calgary
Calgary, Alberta, Canada

Monir Imamverdi
monir.imamverdi1@ucalgary.ca
University of Calgary
Calgary, Alberta, Canada

Ricardo Cruz
ricardo.cruzmendez@ucalgary.ca
University of Calgary
Calgary, Alberta, Canada

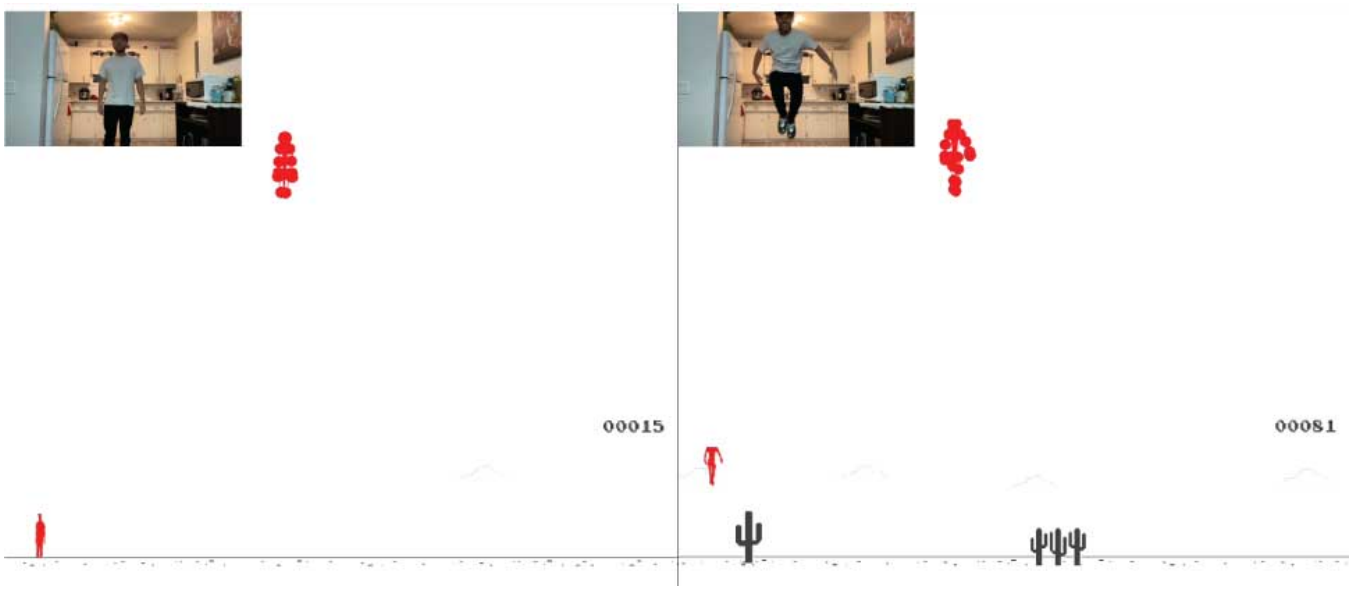

Figure 1. Jump detection in Dino game

Abstract

This paper proposes a novel approach for embedding segmented humans into a 2D game environment to facilitate gamified exercises. The proposed approach utilizes computer vision techniques to extract the segmented human from a real-world image or video, which is then overlaid onto a 2D game environment. The game environment is designed to provide a fun and engaging platform for users to perform workouts, such as jumping jacks, squats, push ups, etc., which are tailored to their specific needs. The game also provides real-time feedback on bio-mechanical form and encourages users to achieve their exercise goals through various game mechanics.

ACM Reference Format:
Christopher Schultze, Diwij Dev, Monir Imamverdi, and Ricardo Cruz. 2023. Embedding Segmented Humans into 2D Games for Gamified Exercises. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

The goal for our project is twofold: to make a low cost mixed reality application and an to increase realism through body segmentation. The current paradigm of mixed reality fitness applications is generally based on expensive equipment with

limited flexibility. For example, smart fitness mirrors [16], a category that has been gaining popularity with offerings from the likes of Tonal and LuluLemon, are cost prohibitive, offer a narrow field of view, and have a fixed vantage point for the user. Costing $1000 and up, we see this as a major barrier for entry for those interested in at-home fitness solutions. Arora et al. say that "a pose estimation-based gaming interaction system would require a media capturing device, a processing unit, and a display device to show the output. [11] With this in mind, we opted to make our application based on the browser, and to take advantage of modern web browsers' capability to access the user's webcam, display, and processing unit. Since modern web browsers are available on most operating systems, it makes our application simple and cheap to access.

People are to engage more when visualizing a mirror image of themselves rather than a 3D avatar made to resemble their characteristics. By incorporating themselves into the game as the moving character, both the player and the game can critique movement and form while providing far more valuable and accurate feedback to the player. One of the main objectives of our application is to provide feedback on form, since research has shown time and again that proper technique is imperative for correct muscle utilization to reach one's desired fitness outcomes. The problem we see is that 3D avatars interpolate, and sometimes even misrepresent a player's movements, and cannot provide meaningful physiological feedback.

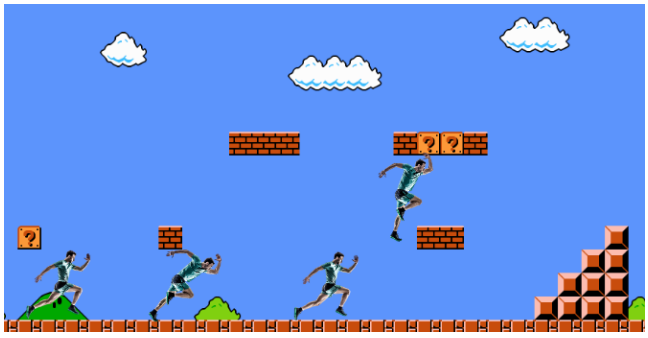

Figure 2. Teaser figure

Another factor is the simplicity in implementation. Unlike complex game development practices, this type of application can be developed using JavaScript and HTML, making it relatively easy to develop, test, and iterate.

2 RELATED WORK

Below we discuss prior work with regards to design considerations and methods for applications that utilize human body, programming interfaces that enable this design to occur, and how these programs can be represented and automatically synthesized.

One similarity between our work and HybridTrak [20] is the usage of an off the shelf webcam. In HybridTrak, full-body tracking in virtual reality improves presence, allows interaction via body postures, and facilitates better social expression among users. However, full-body tracking systems today require a complex setup fixed to the environment (e.g., multiple lighthouses/cameras) [7] and a laborious calibration process, which goes against the desire to make VR systems more portable and integrated. HybridTrak [20] provides accurate, real-time full-body tracking by augmenting inside-out1 upper-body VR tracking systems with a single external off-the-shelf RGB web camera. HybridTrak [20] uses a full-neural solution to convert and transform users' 2D full-body poses from the webcam to 3D poses leveraging the inside-out upper-body tracking data. The paper shows that HybridTrak [20] is more accurate than RGB or depth-based tracking methods [13]. HybridTrak [20] was tested with users in the popular VRChat app and showed that body postures presented by HybridTrak [20] are more distinguishable and more natural than a solution using an RGBD camera [14].

Our work extends this integration by removing the VR requirement and, instead, embedding the human segmented model into 2D environments. This creates further flexibility in hardware, where now all that is needed is an off-the-shelf webcam, a monitor, and a lightweight JavaScript game. As well, fundamentally, the inherit benefit of seeing one's real body embedded into the game, as we have demonstrated, as opposed to an interpolated 3D model is critical in applications like fitness.

To further highlight the proliferation of 3D abstracted player models in the current status quo of human embedded game applications, we discuss ControllerPose[9]. ControllerPose integrates cameras into handheld controllers, where batteries, computation and wireless communication already exist. By virtue of the hands operating in front of the user during many VR interactions, the controller-borne cameras[9] can capture a superior view of the body for digitization and enable body tracking applications [3]. The pipeline composites multiple camera views together, performs 3D body pose estimation [18], uses this data to control a rigged human model with inverse kinematics, and exposes the resulting user avatar to end user applications. The system, however, is limited by low frame rates due to the computational demand of the machine learning model as well high latency due to the nature of the wireless communications. This affects the workloads that can be comfortably used by users.

Again, like with HybrikTrak, seeing one's real body in a game is essential to applications like fitness which require accurate tracking. As it stands, HybridTrak cannot adequately support such implementations due to the 3D model which creates a disconnected sensation for the players. In addition, the technical limitations of wireless computation creates

frame rate liabilities which cannot match the pace of a fitness or game application, hence our motivation to seek a simpler, faster, and more accurate solution.

A more affordable solution in the market is Pose-on-the-Go [8], which is a full-body pose estimation system that uses sensors already found in today's smartphones [10]. This stands in contrast to prior systems, which require worn or external sensors. This result is achieved via extensive sensor fusion, leveraging a phone's front and rear cameras, the user-facing depth camera, touchscreen, and IMU. Even still, data about a user's body (e.g., angle of the elbow joint) is missing, where inverse kinematics are to estimate and animate probable body poses. The system is most accurate with arm holding the phone, as well as the torso's movements. The rear camera is used to keep track of the environment and the body's relative position. However, a lot of applications are "face-centric" [12] and don't need full body tracking, which is where a technology like this would excel.

This implementation highlights nicely the trade offs of affordability and usability. With Pose-on-the-Go, applications that uniquely focus on a players face stand to benefit greatly from this implementation. In our application, however, games that require a fixed player model, such as side scrollers, can be implemented to embed a player's real body form.

And like we've discussed with previous papers, we believe there is an inherent advantage to implementing a player's real body as opposed to an interpolated 3D rendering when it comes to tracking body movements accurately for fitness applications as we've demonstrated.

3 DESIGN AND IMPLEMENTATION

For the implementation part, we opted for Google's MediaPipe Pose library [15], a high-fidelity body pose tracking library that infers 33 3D landmarks and allows background segmentation masking on the human body via a live video stream. This library enabled us to achieve our two primary objectives: to track and classify body movements, and to isolate the body's image. This will allow us to place the segmented image of the player's body directly in the game, and the pose estimation techniques allow us to map body movements to game controls.

As mentioned, one of our main objectives was to place a player's body directly into the game in order to facilitate better immersion as well as accurate visual feedback of a player's performance of certain exercises. MediaPipe is able to generate a segmentation mask around a player's body, and using that image, we're able to isolate the body from its surroundings, crop that part of the camera feed, and place it into the game as a playable character.

MediaPipe Pose tracks 33 distinct landmarks on the human body. Typically, MediaPipe implementations use machine learning [17] to classify poses by training the model on static images of different body poses, for example the "up" and "down" position of a push-up, and subsequently classifying the images coming out of a video feed with a specific degree of confidence. We have yet to test this method as its applicability to a constantly moving player's body performing movements with a high degree of variability is not clear. For example, a "jump" can have many different forms, say, with one, or both, or neither knees bent, and everything in between. We have currently opted with two-dimensional tracking of the players joints, and the angles between them, to classify movements. The machine learning classification route may be worth further investigation for its performance benefits, but for now we're achieving a high degree of accuracy with our tracking methodology. For now, we have programmed our application to detect jumps, squats, and push-ups, using various calculations based on the landmarks provided by MediaPipe, which we will detail below.

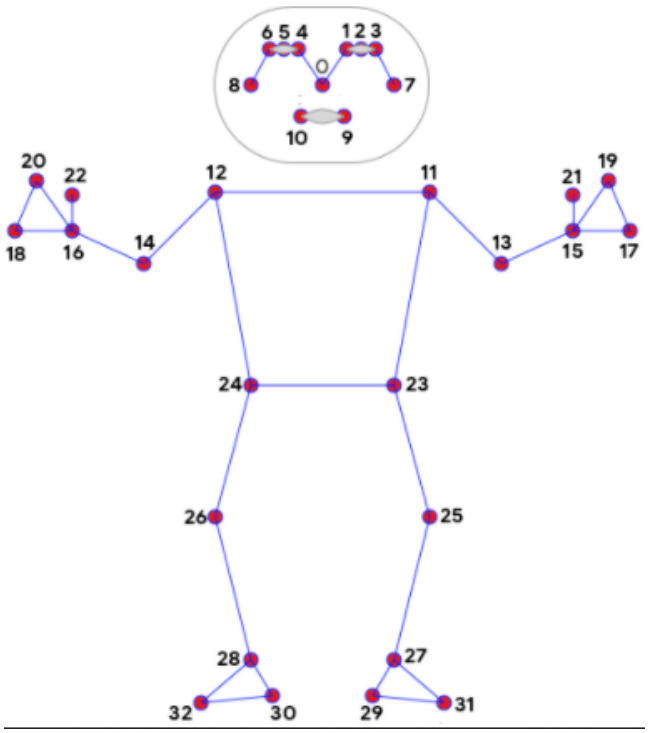

Figure 3. Mapping used in MediaPipe

Since we are solely using a webcam and a browser, our application somewhat lends itself to 2D environments as we are not considering depth of field in the camera stream. This allows for a unique experience where the player can see themselves in the game, and see how they are performing certain movements, and navigate through the level in real time. The simplicity of a 2D game is, too, an advantage here, where we can direct the player's focus on execution of the exercises themselves as opposed to game mechanics and strategy. The simple games are straightforward to grasp and

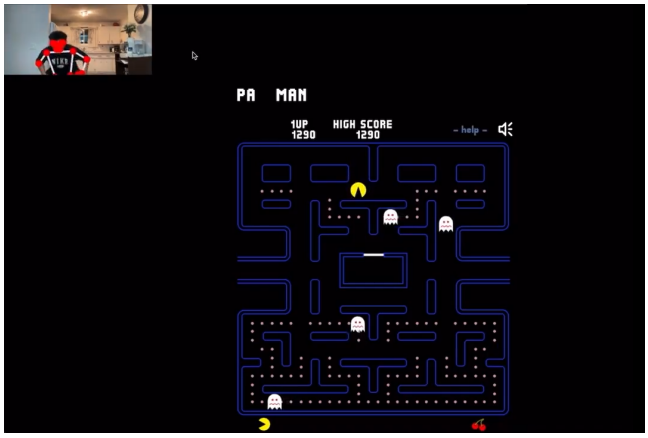

Figure 4. Body segmentation as an external view

we believe that most people can jump right in and have a great workout!

Central to this experience is a fun, guided, and well-structured game design that guides the players through a workout for each level. The levels will be designed with the workout's specific structure and rhythm in mind, where players need to perform specific movements at specific intervals so that it closely mimics an interval training experience all the whilst playing through a game.

There were two functional requirements that were aimed to be fulfilled by the prototype. The first was to insert an image of the player into the game that would update in real-time from a video stream. This would enhance the player's immersion into the computer game as they would see themselves embedded into the 2D environment. The second requirement was to enable the user of a single-player game to send inputs by making specific movements in physical reality. This would facilitate our motivation to 'gamify' exercises.

In order to detect specific player movements, we used the generated pose landmarks and calculated the movements of specific joints as it pertains to specific movements. For example, for a jump, we calculated the delta of the y (vertical) coordinate of both the right and left knees from the standing position in order to register a jump. Since the coordinates on the screen are normalized, top left being (0,0) and bottom right being (1,1), we settled on a difference of -0.2 for a jump to be registered. This was satisfactory for the purpose of our prototype, but will certainly need tuning to cover natural variances of the movement, such as one-legged jumps, etc.

To detect a push up, we looked at the position of the feet, hips, and shoulders, as well as the angle at the elbow joints in order to register the "down" and the "up" position. The baseline is recorded at the "up" position, and subsequently the position is continuously compared until the players elbow angle cross a specific threshold and the shoulders have moved vertically down.

Similarly, to detect a squat, we captured a baseline standing reference, and compared subsequent join coordinates to it. For the squat, we looked at the angles at both the hips and knees.

For example, we calculated the angle at the knee using the following formula:

$$\theta = \cos^{-1}\left(\frac{AB^2 + AC^2 - BC^2}{2(AB * AC)}\right)$$

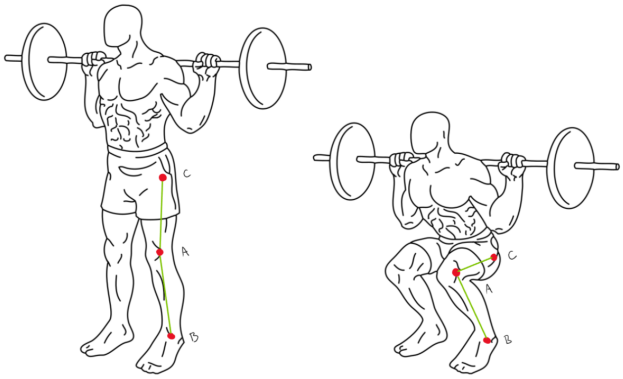

Figure 5. Calculating the angle of the squat posture

With all these movements, we've designated some ideal reference points for the user to achieve. Utilizing full range of motion is imperative for exercise, both from a hypertrophy as well as an injury-prevention standpoint. We aim to provide live feedback to players on specific to achieve said goals. For example, and ideal squat will hit a minimum 90 degrees at the hips, so we'll tell the player whether or not they've reached the specified threshold for a successful squat.

The other goal was to embed the player's body into the game as the playable character and update it in real-time. The Pose API provides a way to extract the user's pose into an image using its body segmentation masking. We used this technique to replace the dinosaur character in our prototype, and scaled it appropriately to match the game's environment. This portion of the project will need some deliberate design choices to allow the player to feel a coherent and inclusive experience as part of the game's 2D world. Considerations such as game scale, level pace, sounds design, among others, will have to be curated to achieve a cohesive gaming experience. Games that we created prototypes of for our purposes include Chromium's dino runner easter egg[5], a browser-based version of Doodle Jump[1], and a browser-based version of Pacman[4].

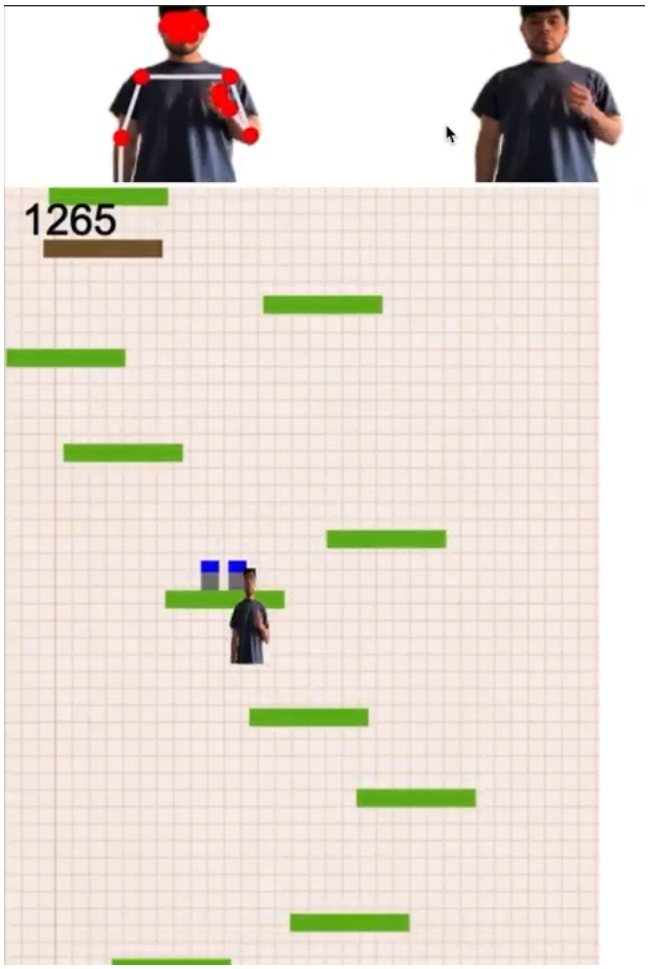

Figure 6. Example of body segmentation in a vertical scroller game such as Doodle Jump

4 CONTRIBUTIONS

We present the collaborative efforts of a team of researchers to address a particular research problem. Each member of the team has made significant contributions to the project, bringing their unique expertise and perspectives to the table.

- Monir has been responsible for all graphic work, presenting and back-end development.
- Ricardo has brought video production skills for the demo, along with code development for the MediaPipe Pose integration.
- Chris has been the project manager, managing project timelines and establishing shared coding environments.
- Diwij's expertise in the research side was contributed towards this project. Writing the paper and formatting while also some minor parts of developments in the back-end side of code.

Together, the contributions of each team member have resulted in a comprehensive and rigorous study that addresses an important research question.

5 LIMITATIONS AND FUTURE WORK

In the duration of this project, we encountered some noteworthy limitations that may warrant future design considerations as well as future work:

Retrofitting vs. Building from Scratch

Our project was centered around retrofitting existing open source JavaScript games with a body-segmented character model. Needless to say, this workflow is fundamentally opposite to a game development process that would be built around said incorporation, rather than embedding it after the fact. While our project was effective in demonstrating the concept, our development approach did not mirror a real-world game development process which would centre around an established character model at a much earlier stage in the game development. In the following section, we explore the challenges of our approach as well as some methodologies that would benefit embedding body segmentation models into games.

The Need for Standardization

One of the things that became increasingly apparent as we progressed in the project was the fact that each game required custom implementation. The bespoke nature of our work resulted in additional complexity with each game that we tried to adapt by incorporating a body segmented character model. Some of the considerations that needed to be accounted for were in fact simple, but nonetheless amounted for a substantive amount of the workload. For example, character models are scaled differently in each game, and scaling that to the body segmentation model we were processing through Pose API [2] meant that we needed to hard code coordinates relative to where the user stood in camera frame. In addition, we need to ensure that the user stood in the same spot each time. For example, in our Dino Runner implementation, the user must stand far enough so that knees positions are roughly at the y=0.6 so that the height of the player matches the original dinosaur's hitbox.

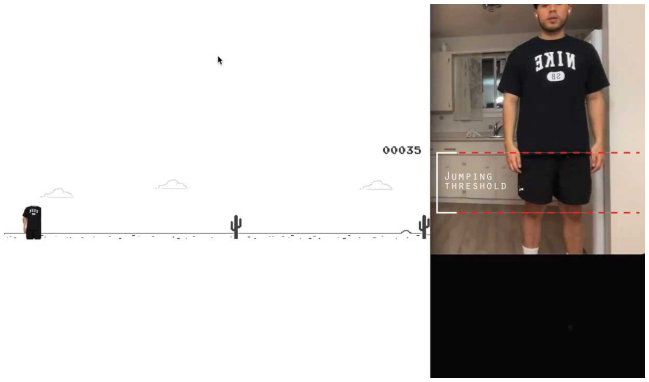

Figure 7. Image showing exact coordinates of knees

A standardized method of capturing and scaling player models would be beneficial in this regard.

Gleaning from Industry

When we look at the industry for existing standardizations for other applications, such as VR, one can find useful and interesting ideas and implementations. For example, Steam's OpenVR API [19] provides a standardized way of interfacing with VR hardware from multiple vendors without requiring that applications have specific knowledge of the hardware they are targeting. OpenVR is a standard which encourages a common baseline experience for users and easier development for programmers. To give an illustrative example: without OpenVR, if a developer wants to program "when the left-hand trigger is pressed… do X", the question becomes: "are you referring to Oculus Rift or Razer OSVR…" and so on. OpenVR is a one-size-fits-all approach, and something similar would be beneficial in the body segmentation space. The Kinect library is a close technical relative in that it abstracts the skeletal physics of a player's body movement. The current gap in the market, and what this project had aimed to explore, is a graphical embedding of a player's body to be the playable character.

Animation-heavy Applications Are Prohibitive

As part of our demo, we had the ambitious intention of embedding a body-segmented player into a Street Fighter-like game[6]. We quickly came to face the realization that scripting the complex movements of a fighting game was a herculean task that was far beyond the scope of our project. Street Fighter, and most fighting games, have complex player animations such as flips, rolls, etc. This remains to be a limitation of body segmented applications, since, presumably, no human can perform Ryu's Tatsumaki Senpukyaku (aka. hurricane kick).

6 CONCLUSION

Embedding segmented humans into games for a fun, inexpensive and easily accessible AR experience for exercises has the potential to revolutionize the way we approach physical fitness. By leveraging technology to create engaging into interactive games that incorporate real-life movements and exercises, individuals may be more likely to participate in physical activity and maintain a consistent exercise routine. Through the use of pose tracking technology and simplistic programming narratives, developers can create personalized workouts that adapt to the user's skill level and progress over time. Furthermore, the incorporation of social elements and competition can increase motivation and accountability for individuals participating in gamified exercise programs. Overall, the potential benefits of embedding segmented humans into 2D games for gamified exercises are numerous, and further research and development in this area could have a significant impact on public health and physical fitness.

References

[1] [n.d.]. Doodle Jump GitHub Repository. https://github.com/omackenzie/Doodle-Jump-in-JS

[2] [n.d.]. Google - Pose Detection API. https://developers.google.com/ml-kit/vision/pose-detection

[3] [n.d.]. Gypsy Motion Capture System. http://metamotion.com/gypsy/gypsy-motion-capture-system.htm

[4] [n.d.]. Pacman GitHub Repository. https://github.com/luciopanepinto/pacman

[5] [n.d.]. T-Rex-Runner GitHub Repository. https://github.com/wayou/t-rex-runner

[6] [n.d.]. TekkenJS GitHub Repository. https://github.com/Archiehere/TekkenJS

[7] Karan Ahuja, Chris Harrison, Mayank Goel, and Robert Xiao. 2019. Mecap: Whole-body digitization for low-cost vr/ar headsets. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*. 453–462.

[8] Karan Ahuja, Sven Mayer, Mayank Goel, and Chris Harrison. 2021. Pose-on-the-Go: Approximating User Pose with Smartphone Sensor Fusion and Inverse Kinematics *(CHI '21)*. Association for Computing Machinery, New York, NY, USA, Article 9, 12 pages. https://doi.org/10.1145/3411764.3445582

[9] Karan Ahuja, Vivian Shen, Cathy Mengying Fang, Nathan Riopelle, Andy Kong, and Chris Harrison. 2022. ControllerPose: Inside-Out Body Capture with VR Controller Cameras. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems* (New Orleans, LA, USA) *(CHI '22)*. Association for Computing Machinery, New York, NY, USA, Article 108, 13 pages. https://doi.org/10.1145/3491102.3502105

[10] Raphael Anderegg, Loïc Ciccone, and Robert W Sumner. 2018. PuppetPhone: puppeteering virtual characters using a smartphone. In *Proceedings of the 11th ACM SIGGRAPH Conference on Motion, Interaction and Games*. 1–6.

[11] Akul Arora, Harsh Gupta, Naman Jain, and A Shanthini. 2022. Prospects of pose estimation as an input in gaming. In *AIP Conference Proceedings*, Vol. 2519. AIP Publishing LLC, 030035.

[12] Teo Babic, Florian Perteneder, Harald Reiterer, and Michael Haller. 2020. Simo: Interactions with distant displays by smartphones with simultaneous face and world tracking. In *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems*. 1–12.

[13] Federica Bogo, Angjoo Kanazawa, Christoph Lassner, Peter Gehler, Javier Romero, and Michael J Black. 2016. Keep it SMPL: Automatic estimation of 3D human pose and shape from a single image. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part V 14*. Springer, 561–578.

[14] Lubomir Bourdev and Jitendra Malik. 2009. Poselets: Body part detectors trained using 3d human pose annotations. In *2009 IEEE 12th International Conference on Computer Vision*. IEEE, 1365–1372.

[15] Camillo Lugaresi, Jiuqiang Tang, Hadon Nash, Chris McClanahan, Esha Uboweja, Michael Hays, Fan Zhang, Chuo-Ling Chang, Ming Yong, Juhyun Lee, Wan-Teh Chang, Wei Hua, Manfred Georg, and Matthias Grundmann. 2019. MediaPipe: A Framework for Perceiving and Processing Reality. In *Third Workshop on Computer Vision for AR/VR at IEEE Computer Vision and Pattern Recognition (CVPR) 2019*. https://mixedreality.cs.cornell.edu/s/NewTitle_May1_MediaPipe_CVPR_CV4ARVR_Workshop_2019.pdf

[16] Hye Sun Park, Gun A Lee, Byung-Kuk Seo, and Mark Billinghurst. 2021. User experience design for a smart-mirror-based personalized training system. *Multimedia Tools and Applications* 80, 31159–31181.

[17] Yadira Quiñonez, Carmen Lizarraga, and Raquel Aguayo. 2022. Machine Learning solutions with MediaPipe. In *2022 11th International Conference On Software ocess Improvement (CIMPS)*. 212–215. https://doi.org/10.1109/CIMPS57786.2022.10035706

[18] Denis Tome, Thiemo Alldieck, Patrick Peluse, Gerard Pons-Moll, Lourdes Agapito, Hernan Badino, and Fernando De la Torre. 2020. Selfpose: 3d egocentric pose estimation from a headset mounted camera. *arXiv preprint arXiv:2011.01519* (2020).

[19] Wei Wang, Yuki Suga, Hiroyasu Iwata, and Shigeki Sugano. 2011. OpenVR: A software tool contributes to research of robotics. In *2011 IEEE/SICE International Symposium on System Integration (SII)*. 1043–1048. https://doi.org/10.1109/SII.2011.6147593

[20] Jackie (Junrui) Yang, Tuochao Chen, Fang Qin, Monica S. Lam, and James A. Landay. 2022. HybridTrak: Adding Full-Body Tracking to VR Using an Off-the-Shelf Webcam *(CHI '22)*. Association for Computing Machinery, New York, NY, USA, Article 348, 13 pages. https://doi.org/10.1145/3491102.3502045