Interactive Machine Learning for Movement Interaction in VR

Tom Lawrence Tianyuan Zhang Clarice Hilton Marco Gillies*

Department of Computing
Goldsmiths, University of London
London, UK

ABSTRACT

Full body movement is a powerful way of interacting with virtual reality experiences. Not only does it reproduce real world interactions, it can also have positive effects on emotions. However, designing effective movement interaction can be hard, as our knowledge about how we move is tacit and embodied, meaning that we can move without knowing exactly how we make those movements. This makes it hard to explicitly program movement interaction. Interactive Machine Learning (IML) is an alternative approach in which movement interaction is designed and implementing by providing examples of movement. This paper presents IntearctML, a movement interaction design platform based on IML, as well as a case study of using it to create a VR experience called Dolittle VR.

Index Terms: Virtual Reality, Movement Interaction, Interactive Machine Learning.

1 INTRODUCTION

Mel Slater's theory of Virtual Reality[10] places movement at the core of the VR experience. The two core illusions, Place Illusion and Plausibility Illusion, are based on reproducing the sensorimotor contingencies[9] that relate our movement and perception in the real world. This means that our interactions with VR should mirror our interactions with the real world, i.e. they should use the movement of our full bodies. This is already common place in VR where head movement is key to viewing the world[10], walking to how we navigate it[11], and our hands to how we interact with objects[1]. There are multiple other benefits of using movement interaction[5], such as making use of real world skills[7] or embodied cognition[8].

One of the challenges of developing movement interaction techniques is that much of our movement knowledge is embodied and tacit[4], meaning that we can perform a task such as riding a bicycle without being able to put into words exactly how we do it. If we cannot put our movement knowledge into worlds, we are unlikely to be able to express it in program code. Machine learning methods can address this issue, because they make it possible to implement movement interaction by giving examples of movements, rather than explicit coding. However, traditional machine learning is a batch process requiring the gathering of large amounts of data in a long process that conflicts with the rapid prototyping typical of interaction design. The use of Interactive Machine Learning[2] techniques that aim to reproduce the iterative workflow of design process, have also been shown to benefit the designers of movement interaction by allowing them to design in more embodied ways[3].

This paper presents InteractML, a platform for interactive machine learning based movement interaction design. It also presents a case study of using InteractML to develop DolittleVR a virtual reality experience that relies on movement as a way of interacting with abstract virtual animals.

*m.gillies@gold.ac.uk

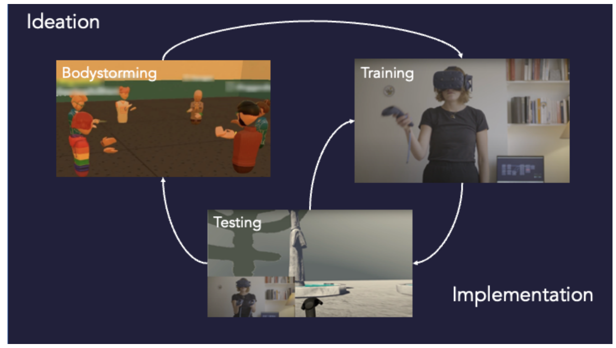

Figure 1: The InteractML process.

2 INTERACTML

InteractML[6] is an interactive machine learning library built for the 3D game engine Unity. It can build interaction using any real time data however it is specifically tailored for movement interaction. It allows users to design movement interaction by actually performing the movements. Real time movement data can be processed by the machine learning model to produce responsive effects in the environment, this is particularly useful for virtual reality applications where user motion is a key feature. It is built for iterative and collaborative interaction design, where users can ideate as they train the model. This should produce expressive movement interaction owing to the creative freedom in the workflow. Another strength of InteractML is that interaction is built from real movement data which is sensitive to the nuances of complex movement. This is often too elaborate to feasibly replicate through rule-based methods.

The workflow supported by InteractML is shown in figure 1. It begins with an ideation phase using "unplugged" bodystorming phase in which designers come up with new movement ideas by moving their own bodies (this can also be done immersively, the image in figure 1 shows people bodystorming in the social VR platform RecRoom). This is followed by an iterative implementation phase moving between training a machine learning model and testing it (and sometimes returning to ideation). Both of these are done through movement. During training, designers perform examples of a movement, which they then test by performing the movement interaction.

InteractML is configured via a visual node-based graph (Figure 2 which makes the tool accessible to both technical and non-technical designers. The node based graph enables choosing learning algorithms, including classification and regression algorithms and selecting feature representations, which include positional and rotation features as well as velocities of both.

3 DOLITTLE VR

In Dolittle VR the user encounters various animals as they explore a forest. The user can interact with these animals with a language of body gestures and dance with them. The result is stimulating game that is inspired by dance and movement therapy and produces

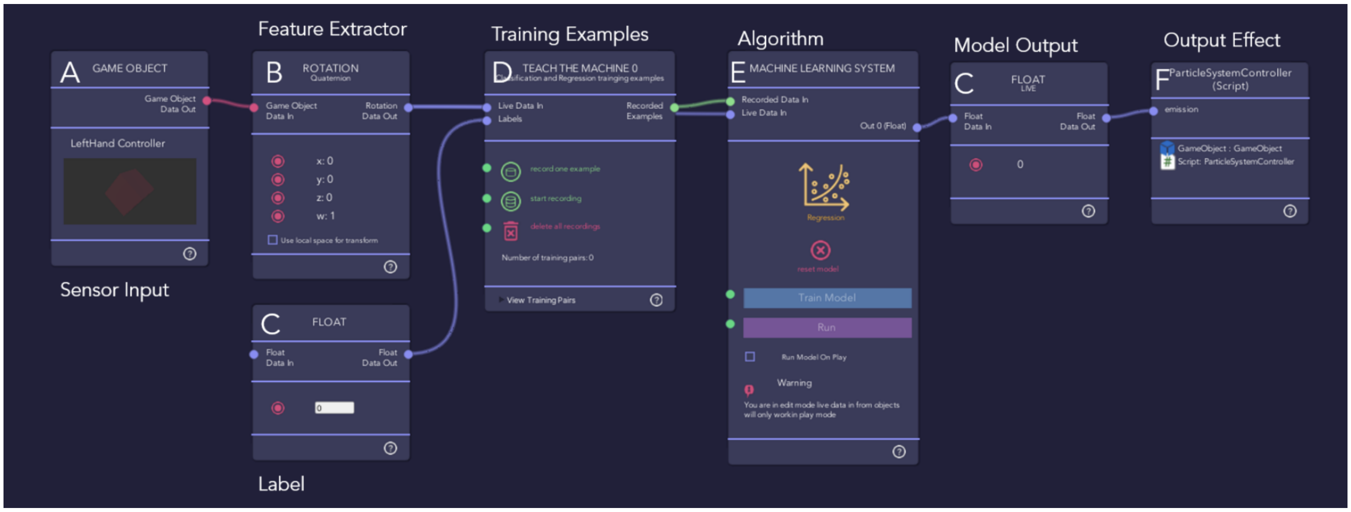

Figure 2: Example of an InteractML graph showing flow from input data, and training to output effects. The InteractML interface consists of a series of nodes representing different aspects of the machine learning pipeline such as feature extraction, training data, the learning algorithms used and the way the output is piped to other functions in Unity.

multi-sensory feedback through the animal's dynamic response. As the user explores the virtual world they will come across different animals wandering. The user can then move the arms and body in a specific gesture whilst vocally calling out to the animal this will beckon the animal to run towards them. After the animal has run over to the user the animal dances with the user. The user can move their arms and body which triggers the animal to dance. Each animal has their own unique dance in response. After a minute the animal becomes tired and runs away from the user.

3.1 Movement Interaction

The movement interaction design is split into two part first the gesture to call the animal over and then the dancing to play with the animal.

3.1.1 Gestures

Each animal has an associated gestures that participants use to call the animal (Figure 3):

- Bird: stand tall and flap your arms stretched wide to your side.
- Kangaroo: role your hands in front of your chest
- Snake: shake your body with your hands above your head
- Firefly: crouch down and flap your arms to your side.

3.1.2 Dancing

The animals dance with users by responding to the users arm and body movements. The faster the user moves their arms and body the faster the animal moves in response. The response of the animal to user movement is immediate giving the user a sense of agency as they feel they can directly effect the animal. Each animal's dance is a unique animation. The speed and intensity of the animation increases with greater user activity.

3.2 Animals

The animals' behaviour was designed to be simple but produce a characterful and stimulating responses. To give the impression that the user was interacting with a real animal.

3.2.1 Bird

The bird flies high above the ground. When the user calls the bird, it starts circling the user. If the user flaps their arms the bird nose dives in front of them; flying up and down following the speed of user motion. This produces dramatic motion as the bird performs aerial acrobatics.

3.2.2 Kangaroo

The kangaroo hops around the environment. When the user calls the kangaroo it hops towards them. It then bounces around the space in front of the user with a speed dependent on the speed of user hand and body movement. This stimulates the user to move with the kangaroo.

3.2.3 Snake

The snake slithers around on the floor. When the user calls the snake it slivers over to the user. The snake then puts its head and neck up like it is facing a snake charmer. The speed of the snakes wiggling depends on the speed of the user's hand and body wiggling. The height of the snake follows the height of the user's hands. This produces a hypnotic effect as the snake mirrors the user's movements.

3.2.4 Firefly

The fireflies swarm as a cloud of sparkling lights. When the user calls the fireflies the cloud of light surrounds the user. The sparkling lights flow and spiral around the user's hands. The spiraling becomes more dramatic the more the user moves and flaps their arms. The effect is scintillating as the cloud of light flows around the user's body and arms.

The animals' appearances were kept deliberately simple and abstract in order to focus participants on the expressiveness of their movement and interaction (Figure 4). The bird, kangaroo and snake were all represented as collections of untextured geoemtric primitives. The fireflies were represesed as glowing particles.

4 Implementation

The application was built in Unity using Meta's OVR SDK for virtual reality integration and InteractML for movement recognition. To register hand movements hand controllers were used, allowing for a full range of motion and reliably recorded hand position.

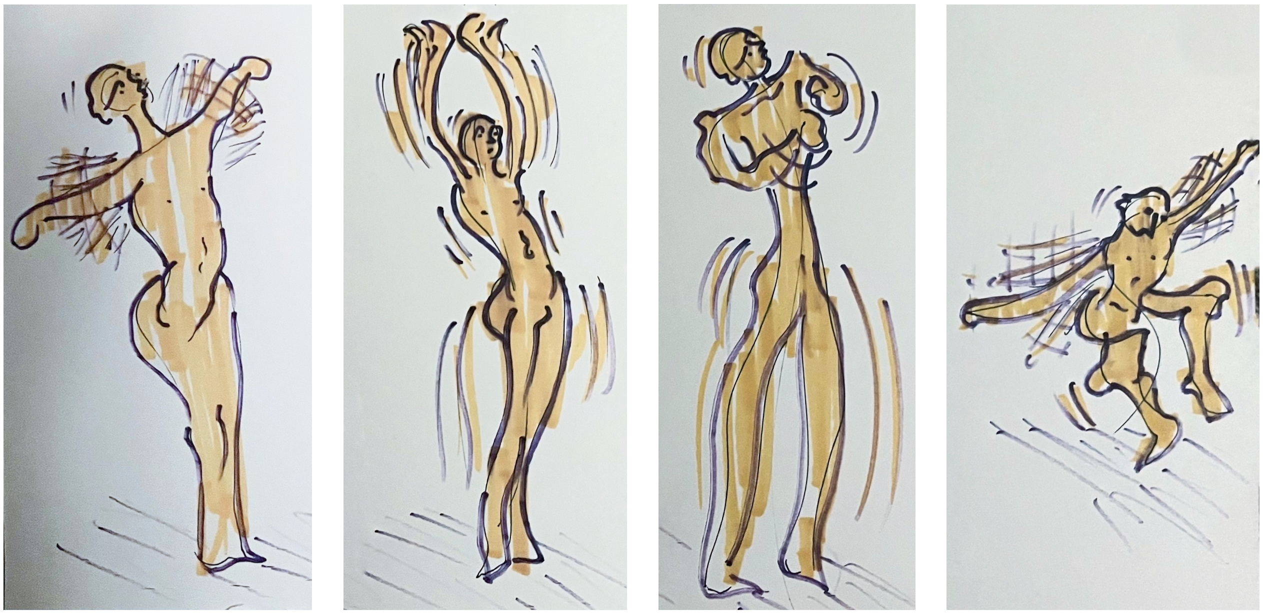

Figure 3: The original sketches showing the different gestures for each animal. Left to right: bird (flapping arms), snake ("slithering" with the whole body), kangaroo (jumping with arms raised in front of the body), firefly (crouching down and flapping arms)

4.1 Machine Learning Gesture Detection

4.1.1 Ideation Process

The process of designing the IML graph, brainstorming gestures and training the models was both collaborative and iterative. This is because to train the model the gestures have to be physically performed. Therefore to ensure that the gesture detection is compatible with many different users we had to train the model with many individuals' input. Multiple people performed the gestures throughout the development process, each individual naturally had their own ideas and interpretation of the gestures. The final set of gestures was based on ideas and feedback from all participants. In particular they helped select gestures that they thought were most intuitive, comfortable and fun.

4.1.2 Data Features

The first step in creating interaction with InteractML is to select the features used to represent the data when training the machine learning model. For example if you want to a model to respond to your hand controller motion the features could be the hand controller position or velocity. The selection of the features which suited the application took numerous iterations and much experimentation. We used scaled hand controller positions vectors relative to the reference frame of the users headset. The vectors were scaled by dividing the position by the stretched arm length recorded in registration, allowing gesture detection to remain consistent between individuals with different body sizes. In addition, we used the scaled distance between the hand controllers and the distance between each hand controller and the headset were used. These features were also scaled by arm length. These measurements added further stability to gesture detection complimenting the hand controller positions. Finally, the scaled distance between the headset and the ground was added. This was the y-component of the headset position scaled by dividing the value by the full standing height. This allows us to add interaction with the full motion of the body and detect if a user is crouching or crawling. This was used to differentiate the firefly pose where the user has to be crouching.

4.1.3 Machine Learning Model

The training algorithm used for gesture detection was the inbuilt classification algorithm of InteractML. Classification was chosen because each gesture needed to be considered a distinct class as each animal would be responsive to one distinct gesture. The classification model uses a standard k-nearest neighbors algorithm with a vector input. The algorithm outputs the detected class as an integer. The integer had a range between 0 and 4. Where 0 is when no particular gesture has been detected and 1-4 represent one of the animal gesture classes. This model could be run in real-time when the application was playing to detect the current gesture of the user.

4.1.4 Final Training Process

For the final training process we recorded examples of each gesture from 6 different individuals. We then tested whether the gesture detection was effective from this dataset on 2 further individuals. Each person recorded 300 examples of each gesture included a classification for when the users hands were in a default position as a null class. This resulted in 12000 data points recorded for the entire dataset. After training the resultant model was sufficient for smooth and reliable game play.

5 Conclusion

The design and implementation of Dolittle VR has shown the potential of Interactive Machine Learning for movement interaction design for VR. It enabled rapid and intuititve design and implementation of complex body gestures. In a small user study (out of scope of this paper), our participants found that the movement interaction was relaxing and helped them build a rapport with the animals, showing that this type of interaction is effective and produces positive emotions.

Acknowledgments

This research was partially supported by the UK Engineering and Physical Sciences Research Council grant "4i: Immersive Interaction Design for Indie Developers with Interactive Machine Learning" (EP/S02753X/1).

References

[1] D. A. Bowman and L. F. Hodges. An evaluation of techniques for grabbing and manipulating remote objects in immersive virtual environments. In *Proceedings of the 1997 symposium on Interactive 3D graphics - SI3D '97*, pp. 35–ff. ACM Press, New York, New York, USA, 4 1997. doi: 10.1145/253284.253301 1

[2] J. A. Fails and D. R. Olsen. Interactive machine learning. In *Proceedings of the 8th international conference on Intelligent user interfaces - IUI '03*, IUI '03, p. 39. ACM Press, New York, New York, USA, 2003. doi: 10.1145/604045.604056 1

[3] R. Fiebrink, P. R. Cook, and D. Trueman. *Human model evaluation in interactive supervised learning*. CHI '11. ACM Press, New York, New York, USA, 2011. doi: 10.1145/1978942.1978965 1

[4] M. Gillies. Understanding the Role of Interactive Machine Learning in Movement Interaction Design. *ACM Transactions on Computer-Human Interaction*, 26(1):1–34, 2 2019. doi: 10.1145/3287307 1

[5] M. Gillies, B. Lee, N. D'Alessandro, J. Tilmanne, T. Kulesza, B. Caramiaux, R. Fiebrink, A. Tanaka, J. Garcia, F. Bevilacqua, A. Heloir, F. Nunnari, W. Mackay, and S. Amershi. Human-Centred Machine Learning. In *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems - CHI EA '16*, vol. 07-12-May-, pp. 3558–3565, 2016. doi: 10.1145/2851581.2856492 1

[6] C. Hilton, N. Plant, R. Fiebrink, P. Perry, C. G. Diaz, R. Gibson, B. Martelli, M. Zbyszynski, and M. Gillies. InteractML: Making machine learning accessible for creative practitioners working with movement interaction in immersive media. In *Proceedings of the ACM Symposium on Virtual Reality Software and Technology, VRST*, 2021. doi: 10.1145/3489849.3489879 1

[7] R. J. K. Jacob, A. Girouard, L. M. Hirshfield, M. S. Horn, O. Shaer, E. T. Solovey, and J. Zigelbaum. Reality-based interaction: a framework for post-WIMP interfaces. In *CHI '08: Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, pp. 201–210. ACM, New York, NY, USA, 2008. doi: 10.1145/1357054.1357089 1

[8] D. Kirsh. Embodied cognition and the magical future of interaction design. *ACM Trans. Comput.-Hum. Interact.*, 20(1):3:1–3:30, 4 2013. doi: 10.1145/2442106.2442109 1

[9] J. K. O'Regan and A. Noë. A sensorimotor account of vision and visual consciousness. *The Behavioral and brain sciences*, 24(5):939–973, 2001. 1

[10] M. Slater. Place illusion and plausibility can lead to realistic behaviour in immersive virtual environments. *Philos Trans R Soc Lond B Biol Sci*, 364(1535):3549–3557, 12 2009. doi: 10.1098/rstb.2009.0138 1

[11] M. Usoh, K. Arthur, M. C. Whitton, R. Bastos, A. Steed, M. Slater, and F. P. Brooks. Walking ¿ walking-in-place ¿ flying, in virtual environments. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques - SIGGRAPH '99*, pp. 359–364. ACM Press, New York, New York, USA, 7 1999. doi: 10.1145/311535.311589 1

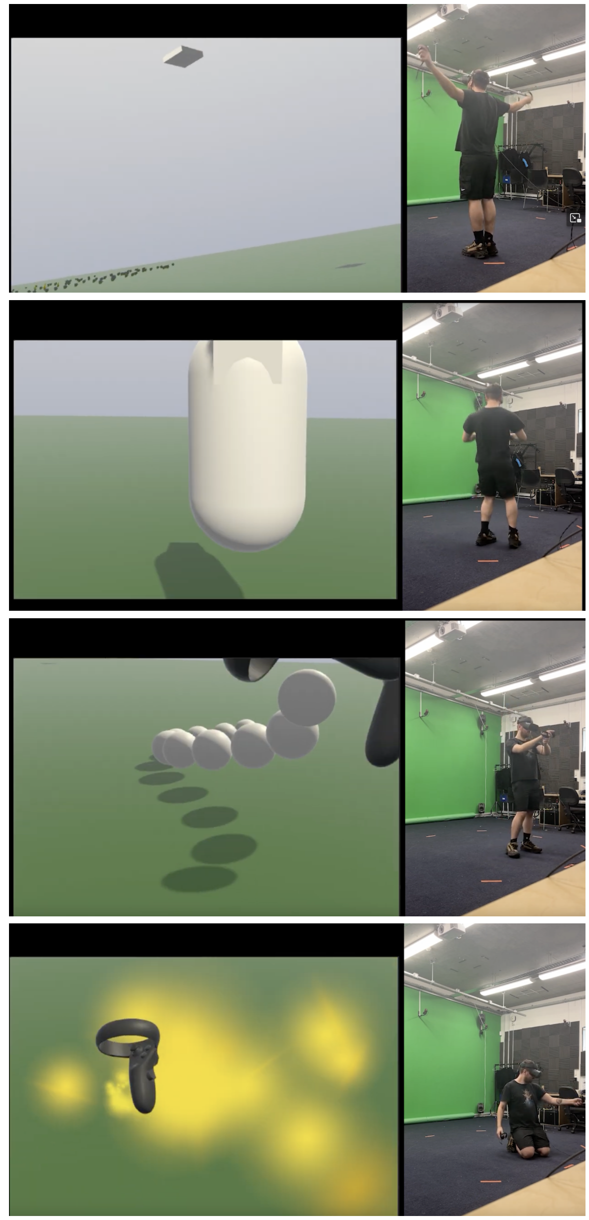

Figure 4: Top to bottom: (1) Bird flying in front of user. Bird is the cuboid in the sky on the left; (2) Kangaroo hopping in front of user. Kangaroo is the grey capsule on the left; (3) Snake wiggling in front of user on the left; (4) Firefly gold particles surrounding user and spiralling around the user's hand on the left.