Interactive Machine Learning for More Expressive Game Interactions

Carlos Gonzalez Diaz
Department of Computer Science
University of York
York YO10 5DD, UK
Email: cgd506@york.ac.uk

Phoenix Perry
Department of Computing
Goldsmiths, University of London
London SE14 6NW
Email: p.perry@gold.ac.uk

Rebecca Fiebrink
Department of Computing
Goldsmiths, University of London
London SE14 6NW
Email: r.fiebrink@gold.ac.uk

***Abstract*—Videogame systems incorporate varied sensors to increase the range of player interactions and improve player experience. However, implementing robust recognisers for player actions with sensors presents significant challenges to developers. Further, sensor-based controls offer little player customisation compared to traditional input interfaces (gamepads, keyboards and joysticks). Past research on motion-driven music systems has successfully used interactive machine learning (IML) techniques to facilitate the development and customisation of sensor-based interfaces, both by developers and end users. However, existing standalone software tools for IML are not ideal for use in game development and distribution. In order to support more effective and flexible use of sensors by game developers and players, we developed an integrated IML solution for Unity3D in the form of a visual node system supporting classification, regression and time series analysis of sensor data.**

I. Introduction

Videogames are increasingly incorporating a diverse variety of sensors. Examples include DIY hardware games, VR sensors, smartphones, AR systems, smart watches and more. Despite decades of research on using sensors as game controllers, there are still no standard practices for how to design sensor-based interactions. It can be difficult for developers to implement accurate and robust sensor analysis when sensors are noisy or high-dimensional, or when the goal is to sense complex movements or actions. Further, players might want to customise sensor-based interfaces—similar to how they currently customise gamepads—to reflect their preferences or expertise, or to suit their own range of motion or abilities. However, current solutions provide little such functionality.

In this paper, we describe a new interactive machine learning (IML) system designed to facilitate development and customisation of games that use unconventional input devices—including sensors, audio, and video—to capture rich information about player movements and actions. This work is informed by the success of IML in facilitating the design of new gestural musical interfaces for professional musicians [1], [2] as well as children and adults with disabilities [3], [4]. We begin by briefly explaining how IML has been used in similar domains and describing the barriers to using existing tools for game development. We then introduce the newly built Unity3D tool, InteractML.

978-1-7281-1884-0/19/$31.00

II. Interactive Machine Learning

Supervised learning algorithms are capable of building new recognisers or control systems from examples, rather than requiring a developer to write code analysing sensor data and specifying how an application (e.g., a game) should respond. Specifically, an algorithm learns from examples of human gestures or actions, each paired with the desired response. For instance, a developer (or player) could provide a few examples of an accelerometer being tilted right and left, along with information about how the colour of an on-screen game object should change in response to each tilt. A supervised learning algorithm can then build a mathematical model capable of choosing a new colour (or any other property) change in response to each new tilt observed during gameplay.

In interactive machine learning, a user (e.g., a developer or player) iteratively builds and refines the model through "cycles of input and review" [5]. IML systems for building new gestural controllers for music typically allow users to create new gesture examples on the fly, and to evaluate models by experimenting with the new controllers in realtime [1], [6]. Users can iteratively modify the training examples, sensor, and learning algorithm to improve performance. A similar approach has been used to customise virtual game characters' behaviour via physical player movements [7].

Several standalone software tools have been created to support IML creation of gestural control interfaces, such as Wekinator [1], GRT [6], and GVF [8]. Occasionally these have been used in games. For example, Schedel and Perry [9] used Wekinator to create a musical game in Unity3D controlled using a Cello K-bow and a Microsoft Kinect [9]. However, the implementation of this system was difficult and needed the writing of substantial new code to enable communication between Wekinator, Unity and the audio engine. Further, the final game could not be released as a standalone application, as it required multiple software programs (Unity3D, Wekinator, and a tailored application to extract data from sensors) running in parallel. The complexity of this toolchain limits the utility of such an IML solution for game developers.

It is worth noting that Unity3D's built-in "machine learning" tools[1] provide reinforcement learning algorithms for training

[1]https://unity3d.com/machine-learning

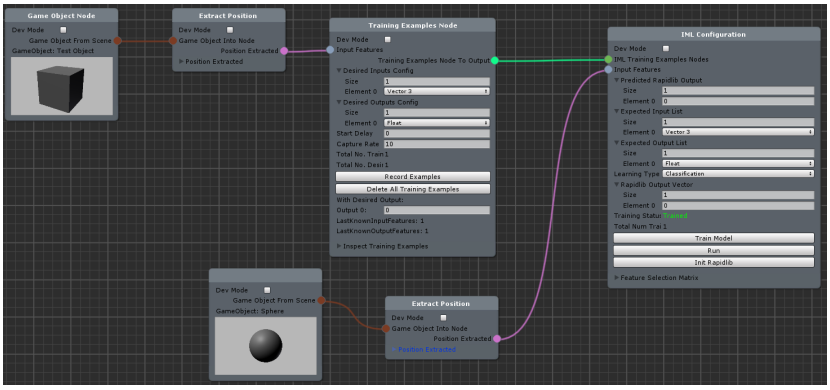

Fig. 1. InteractML, the proposed interactive machine learning solution. On the left of the image are the nodes extracting features from gameobjects in the scene. The node "Extract Position" sends data from a gameobject forward in the graph. In the center of the image is the "Training Examples" node where users can iteratively collect pairs of features extracted (positions, rotations, velocities, etc) and game outputs (sounds, colours, particles, etc). To the right is the "IML Configuration" node, which holds the properties to build the ML model, specifying the type of supervised learning algorithm (classification, regression or time series analysis), the training dataset and which gameobject will feed features during gameplay to perform the real-time ML analysis. Game creators can iteratively customise the training dataset, the features extracted or the properties affected by the ML model outputs until they reach the ML configuration that best expresses their intent (wave a hand, fly a dragon, play an instrument, etc).

agents. Supporting supervised learning with IML to train new sensor-based interactions thus requires entirely different algorithms, workflows, and interfaces.

III. Integrated IML for the Unity3D Game Engine

We have developed a Unity3D plugin, InteractML, that enables developers to configure, train, and use IML systems within the game editor. Using a visual node scripting system (Fig. 1), developers can visualise incoming sensor data, configure game inputs (e.g., specifying what data to extract from sensors or objects in the game); train and refine ML models (by iteratively adding new training examples in realtime); and connect the ML model outputs (the real-time predictions calculated based on the training data) to other objects/scripts in the game scene. In addition, since InteractML doesn't rely on external software, the ML models can be trained and/or refined by player-provided examples in the released version of the game.

InteractML currently supports classification (with k-nearest neighbour), regression (with multilayer perceptron neural networks) and time series analysis (with dynamic time warping) of sensor input. Yet developers do not need to know anything about the algorithms themselves to begin using InteractML; they must only understand (1) the interaction flow required to build a model (i.e., record data/train/run, as described in Section II) and (2) which type of machine learning algorithm (classification, regression, time series analysis) they wish to use. InteractML comes with numerous examples and tutorials to aid developers with these tasks.

IV. Conclusion

InteractML offers a simple and accessible tool to develop sensor interactions, using an in-engine visual node IML workflow that does not require prior expertise with ML techniques. This tool aims to support game creators in more easily developing interactions with sensors, and in creating more complex embodied experiences for players. Since the ML models can be iteratively trained by developers and/or players, there is also room to explore bespoke controllers tailored to people with motion impairments, and game mechanics based around player customisation of actions.

Acknowledgment

This research was supported by the Artist + Machine Intelligence (AMI) Focused Research Awards, funded by Google Arts and Culture.

References

[1] R. Fiebrink, D. Trueman, and P. R. Cook, "A meta-instrument for interactive, on-the-fly machine learning." in *Proc. of the International Conference on New Interfaces for Musical Expression*, 2009, pp. 280–285.

[2] R. Fiebrink, "Machine learning as meta-instrument: Human-machine partnerships shaping expressive instrumental creation," in *Musical Instruments in the 21st Century*. Springer, 2017, pp. 137–151.

[3] S. Parke-Wolf, H. Scurto, and R. Fiebrink, "Sound Control: Supporting custom musical interface design for children with disabilities." in *Proc. of the International Conference on New Interfaces for Musical Expression*, 2019.

[4] S. Katan, M. Grierson, and R. Fiebrink, "Using interactive machine learning to support interface development through workshops with disabled people," in *Proceedings of the 33rd annual ACM conference on human factors in computing systems*, 2015, pp. 251–254.

[5] J. J. Dudley and P. O. Kristensson, "A review of user interface design for interactive machine learning," *ACM Transactions on Interactive Intelligent Systems*, vol. 8, no. 2, pp. 1–37, Jun 2018.

[6] N. Gillian and J. A. Paradiso, "The Gesture Recognition Toolkit," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 3483–3487, 2014.

[7] A. Kleinsmith and M. Gillies, "Customizing by doing for responsive video game characters," *International Journal of Human-Computer Studies*, vol. 71, no. 7-8, pp. 775–784, Jul 2013.

[8] B. Caramiaux, F. Bevilacqua, and A. Tanaka, "Beyond recognition: Using gesture variation for continuous interaction," in *CHI'13 Extended Abstracts on Human Factors in Computing Systems*, 2013, pp. 2109–2118.

[9] M. Schedel, R. Fiebrink, and P. Perry, "Wekinating #000000Swan: Using machine learning to create and control complex artistic systems," *Proceedings of the 11th International Conference on New Interfaces for Musical Expression*, pp. 453–456, June 2011.