Gesture Recognition Using Leap Motion: A Comparison Between Machine Learning Algorithms

Ivo Aluízio Stinghen Filho
Universidade Federal do Amazonas
iasf@icomp.ufam.edu.br

Estevam Nicolas Chen
Universidade do Estado do Amazonas
enc.eng@uea.edu.br

Jucimar Maia da Silva Junior
Universidade do Estado do Amazonas
jucimar.jr@gmail.com

Ricardo da Silva Barboza
Universidade do Estado do Amazonas
rsbarboza@gmail.com

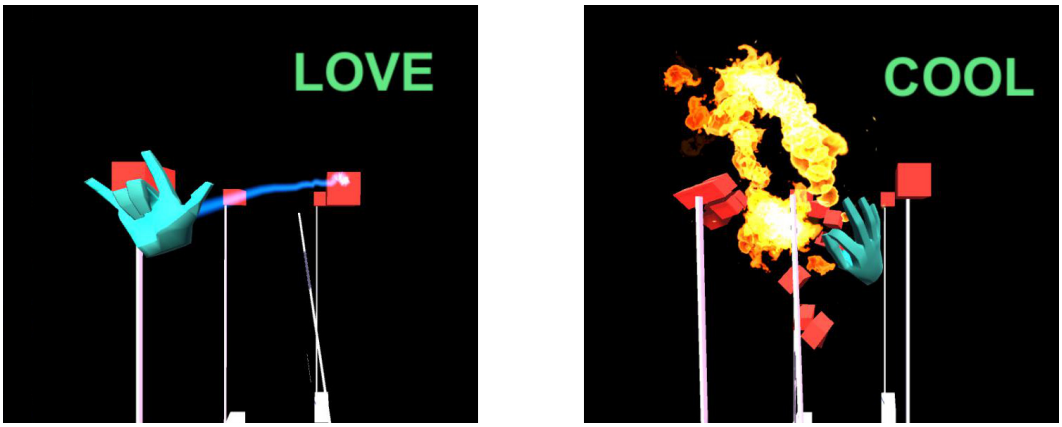

Figure 1: Hand gesture recognition in Unity

ABSTRACT

In this paper we compare the effectiveness of various methods of machine learning algorithms for real-time hand gesture recognition, in order to find the most optimal way to identify static hand gestures, as well as the most optimal sample size for use during the training step of the algorithms.

In our framework, Leap Motion and Unity were used to extract the data. The data was then used to be trained using Python and scikit-learn. Utilizing normalized information regarding the hands and fingers, we managed to get a hit rate of 97% using the decision tree classifier.

CCS CONCEPTS

• **Human-centered computing** → **Gestural input**; • **Computing methodologies** → **Supervised learning by classification**;

KEYWORDS

Virtual Reality, Leap Motion, Motion Capture, Machine Learning

SIGGRAPH '18 Posters, August 12-16, 2018, Vancouver, BC, Canada

ACM ISBN 978-1-4503-5817-0/18/08.
https://doi.org/10.1145/3230744.3230750

ACM Reference Format:
Ivo Aluízio Stinghen Filho, Estevam Nicolas Chen, Jucimar Maia da Silva Junior, and Ricardo da Silva Barboza. 2018. Gesture Recognition Using Leap Motion: A Comparison Between Machine Learning Algorithms. In *Proceedings of SIGGRAPH '18 Posters.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3230744.3230750

1 INTRODUCTION

Humans interact with machines in a variety of ways. As such, many forms of HCI(Human-Computer interaction) has developed. Although the use of the mouse and keyboard is widespread, new methods of HCI are also developed. An example is through gesture recognition, a topic that received great attention in the field of HCI due to the development of VR (Virtual Reality) technology, as well as a method to control robots.

A gesture is a form of non-verbal expression. It includes hands, face and other parts of the body. Hand gestures can be divided in two categories: Static and dynamic gestures. This paper will focus on static hand gestures.

Depending on the type of the input data, the hand gesture recognition can be also divided in two categories: appearance-based and 3D model-based algorithms. Appearance-based algorithms use the data acquired from the silhouette or contour of the input images, meanwhile 3D model-based algorithms use volumetric or skeletal data, or even a combination of the two.

In this work, we present a comparison of three different machine learning algorithms, including Decision Trees[Yao et al. 2014], KNN (K-Nearest Neighbors algorithm) and SVM (Support Vector Machines)[Gunn et al. 1998].

2 FRAMEWORK

In order to train the machine learning algorithms, it is necessary to acquire the data regarding the fingers. To do so, it was chosen to record a series of hand signs using Leap Motion[Ribeiro et al. 2016][Shao 2016], and to ease the recording process, the Leap Motion API (Application Programming Interface) in Unity was used. It was possible to get many different combinations of variables regarding the fingers. In this work the main variables chosen were the combination of the normalized spatial positions of the tip of the 5 fingers and the 4 angles between adjacent fingers.[Chen et al. 2018]

To generate the necessary data, it is necessary to do at least one record session for each machine learning class, each representing a different hand sign. For each record session, while positioning a hand in different ways to represent the same hand sign, a script in Unity records the necessary data based on Leap Motion to a .csv file, along with the class name. A line containing the data and the class name is recorded every 0.05s, varying in total recording time with necessity.

On this .csv file 30% of the lines were used to train, while the remaining 70% lines were used to evaluate accuracy. A Python algorithm then returns the percentage of correct predictions.

In this work seven classes were used: "Open", "Close", "Thumb", "Two", "Four", "Cool", and "Love", each representing a hand gesture. The .csv files were analyzed, each containing the 7 gestures, with the same proportion. Overfitting started occurring when over than 8400 samples were utilized (1200 per hand gesture), as such, that particular file was used. Figure 2 shows the accuracy of gesture predictions using different classifier algorithms using a file 1200 learning samples per gesture.

Overall, the decision tree classifier gave better results, as such, it was used in further tests. Noticeably, the worst results were that of the "Close" class, with 97% hit rate, and the "Four" class was able to be correctly predicted 100% of the time.

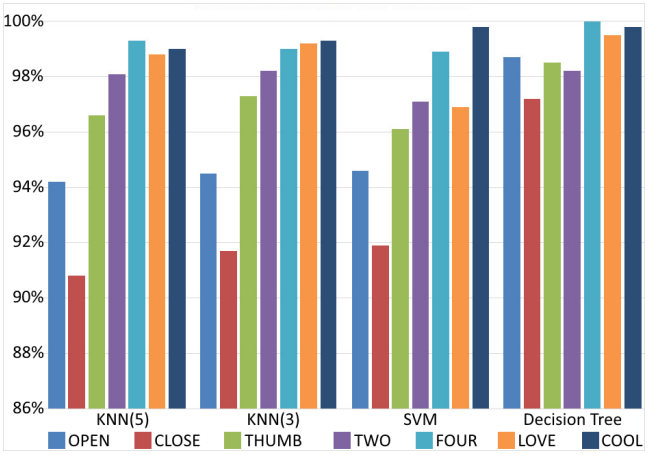

Figure 2: Comparison between KNN (with 3 and 5 neighbours as parameters), SVM and Decision Tree classifiers, respectively, for each of the 7 classes.

3 SOCKET COMMUNICATION AND REAL-TIME TESTING IN UNITY

Taking in consideration the results, the decision tree classifiers was used for offline base classification and training with real-time sample input. By having offline training, it's possible to make real-time predictions.

It was done by forming a loop between Unity and a socket created by Python, running on localhost, as shown in figure 3. As such, the current state is updated every 0.02 seconds in Unity's interface, Unity then sends a vector as input through the socket, to the decision tree classifier in scikit-learn (Python). Having received the data through the socket, the predict() function is run, (using the decision tree classifiers) and returns the result. Ex: "Open". Finally, Unity receives the result from the socket and creates effects as needed.

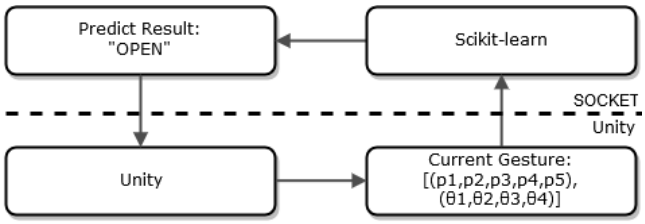

Figure 3: Data loop between Python and Unity.

In the examples shown in figure 1, Unity receives the result of the prediction and shows special effects accordingly. If the socket response is "Cool", it conjures up a fire spell, while if it is 'Love', it conjures up a lightning spell.

4 CONCLUSIONS AND FUTURE WORK

In this paper, we used 1200 samples per class and normalized data regarding the fingers and the angles between them. With this, we achieved a hit rate of over 97% using the decision tree classifier.

Our studies pointed out that such application in conjunction with the Unity engine and the Leap Motion API can be used in real-time applications of arbitrary gestures.

All things considered, further study of how dynamic gestures behave through normalization might be interesting, as both dynamic and static gestures can be used for various applications, such as VR games.

ACKNOWLEDGMENTS

The authors would like to thank UFAM, UEA Ludus Lab, Samsung Ocean and FAPEAM for supporting the development of this work.

REFERENCES

Feiyu Chen, Jia Deng, Zhibo Pang, Majid Baghaei Nejad, Huayong Yang, and Geng Yang. 2018. Finger Angle-Based Hand Gesture Recognition for Smart Infrastructure Using Wearable Wrist-Worn Camera. *Applied Sciences* 8, 3 (2018), 369.

Steve R Gunn et al. 1998. Support vector machines for classification and regression. *ISIS technical report* 14, 1 (1998), 5–16.

Ramos Ribeiro et al. 2016. *Framework for registration and recognition of free-hand gestures in digital games.* SBGames.

Lin Shao. 2016. *Hand movement and gesture recognition using Leap Motion Controller.* Stanford University, Stanford, CA.

Dengfeng Yao, Minghu Jiang, Abudoukelimu Abulizi, and Xu You. 2014. Decision-tree-based algorithm for 3D sign classification. In *Signal Processing (ICSP), 2014 12th International Conference on.* IEEE, 1200–1204.