Hand movement and gesture recognition using Leap Motion Controller

Lin Shao*
Stanford EE 267, Virtual Reality,Course Report, Instructors: Gordon Wetzstein and Robert Konrad

Abstract

The novel device Leap Motion Controller provides an informative representation of hands. We utilize tracking data through the API of Leap Motion Controller to recognize hand movement and gestures. Our experiment shows that our method based on Leap Motion Controller tracking data can recognize hand gesture accurately when no occlusion happens.

1 Introduction

Gesturing is a natural part of human communication and becomes more and more important in AR/VR interaction. The Leap Motion Controller is a new device developed for gesture interaction by Leap Motion (https://www.leapmotion.com/). The device has a small dimension of 0.5x1.2x3 inches. To use the Leap Motion Controller, the user need to connect it to a computer by USB. Then the users put hands on top of the Leap Motion Controller. Figure 1 [1] gives an example of how to use the Leap Motion Controller.

Figure 1: *Leap Motion Usage. The small object in the middle is Leap Motion Controller connecting to the Mac on the right. Hand on top of the Leap Motion is tracked and interacted with virtual objects.*

The Leap Motion Controller could detect palm and fingers movements on top of it. The tracking data which contains the palm and fingers' position, direction, velocity could be accessed using its SDK. According to [Weichert et al. 2013], the Leap Motion Controller provides a detection accuracy of about 200 μm.The newest version of Leap Motion Controller Orion currently do not provide gesture recognition. We are trying to implement the gesture recognition by ourselves.

2 Related Work

After Leap Motion first releases the Leap Motion Controller in 2013, researchers have started to analyze its performances. The performance in selection task of leap motion controller is compared with a common mouse device [Bachmann et al. 2015]. Fitts' law is introduced in the evaluation system. It indicates the Leap Motion Controller's performance in general computer pointing tasks is limited compared with a mouse device given the error rate of 7.8% for Leap motion controller and 2.8% for the mouse device.

The Leap Motion Controller has a wide range of application. It has been used for stoke rehabilitation by people from The Intelligent Computer Tutoring Group in the UniversIty [Bracegirdle et al. 2014]. Another important application is hand gesture recognitions. Many gesture recognition methods have been put forward under difference environments.

Marin et.al [Marin et al. 2015] works on hand gestures recognition using Leap Motion Controller and kinect devices. Ad-hoc features are built based on fingertips positions and orientations. These features are then fed into a multi-class SVM classifier to recognize their gestures. Depth features from the Kinect are also combined with features from Leap Motion Controller to improve the recognition performances. They only focus on static gestures rather than dynamic gestures.

Hand Motion Understanding system developed by [Cooper et al. 2011] utilize colour-coded glove to track hand movement. The tracking system requires users to wear gloves which reduces the user experiences.

We developed more complicated hand gestures recognition systems which not only provide static gestures recognitions but also dynamic gesture recognitions. Only the Leap Motion Controller is required. The users do not need other types of sensors to put on their hands.

3 Approach

3.1 Device

Leap motion controller is new interactive devices mainly aiming at hand gestures and finger position detection developed by Leap Motion. Figure 2 [2] shows the internal structure of Leap Motion Controller. There are three Infrared Light emmitters and two cameras which received the IR lights.

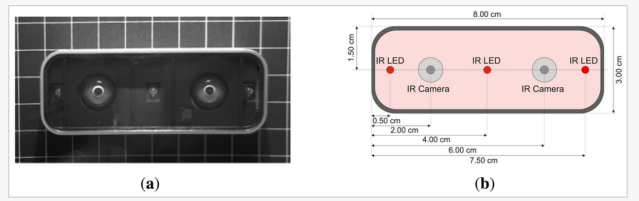

Figure 2: *Leap Motion Controller internal structure.*

3.2 Data

Leap motion has kept updating their SDK after the first release. Currently the newest version is Orion Version 3.1.2. Leap motion provides preprocessed data through their Application Programming Interface. This data is accessed by a Frame object querying by per frame. There are several properties a frame object contains in the Orion version.

*e-mail:lins2@stanford.edu
[1] source:https://www.leapmotion.com/product/desktop?lang=en
[2] source:http://www.mdpi.com/1424-8220/13/5/6380/htm

- Palm position P_{pos}, normal P_N and velocity P_v.
- Hand direction P_D.
- Fingertips position F^i_{pos}, direction F^i_D and velocity F^i_v where i starts from 0 to 4 representing thumb, index, middle,ring and pinky respectively.
- Arm direction

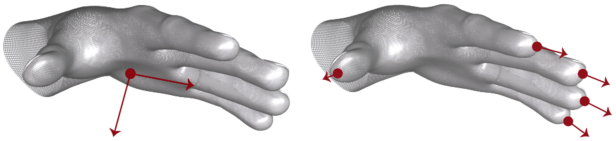

Figure 3: *Left: Palm tracking data. Right; Fingertips tracking data*

Figure 3 [3] shows the tracking data which will be used in our experiments. The palm's position, normal and hand directions are shown in left picture. The right picture shows the fingertips positions and directions.

The origin tracking data is calculated in Leap Motion coordinate systems. The Leap Motion Controller uses a right handed system and millimeters as the unit. We implemented our demo based on Unity(https://unity3d.com/). The Unity uses a left hand system and meters as the unit. So the z-coordinates are opposite in Leap Motion Controller coordinate system and Unity system. The Orion helps solves these problems. The frame object accessed through LeapProvider is in unity coordinate systems and use meters as the unit.

3.3 Features

Based on previous raw data we collected from Leap Motion Controller API, we starts to build features to recognize hand gestures. These features could mainly be divided into two parts. One parts are associated with static gestures containing the positions and directions. The others are used to identify dynamic gestures.

3.3.1 Static Gesture Features

Features for static gestures are mainly built based on palm and fingers relative distances. We calculated two types of distances. One type is distances between fingertips F^i_{pos} and palm center P_{pos} denoted by D_i. The other type is distances between two fingers which are adjacent. For example distance between thumb and index, distance between index $D_{01}$and middle denoted by D_{12} Figure 5 shows examples of static gestures we could effectively recoginize. We have two standard gestures shown in Figure 4 where the distance values are used as parameters to distinguish different static gestures

The other gesture features are built based on distances between fingers and palms. The distance between thumb and index is used to identify the OK gestures. The distance between index and middle finger is used to distinguish V gesture and Index and Middle pointing gesture. The rest gestures simply combined these two standard gestures. For example the index L gestures on the top most in Figure 5 are index and thumb extended and the rest fingers bent.

[3] pictures source:https://developer.leapmotion.com/documentation

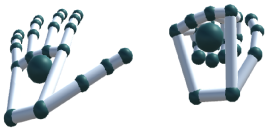

Figure 4: *Left: fingers all extended. Right: fist gesture*

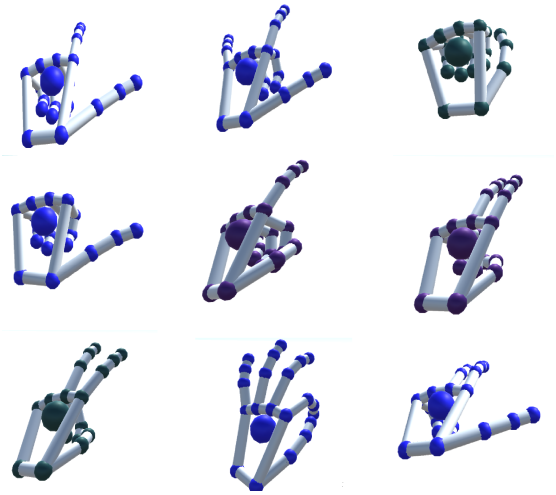

Figure 5: *Examples of static gestures. The first line gestures are index L gestures, ILY gestures, fist. The second line gestures are Thumb up, Index pointing, Index and middle pointing. The third line gestures are V gesture, Ok gesture, Index and Middle L gesture.*

3.3.2 Dynamic Gesture Feature

The dynamic gesture features are easily distinguished from static gestures features. We calculate the total value of velocity magnitude among fingers and palm. If the total movement value is greater than a user-defined threshold, we believe the hand is moving. Otherwise, we starts to recognize the static hand gestures.

Dynamic Gesture features mainly use the velocity of fingertips and palm to detect the movement patterns. Compared with the static gestures, dynamic gestures are much more complicated. We starts from the global movement and then go through the details of the fingers' movement. From the global movement, we try to detect hand translation movement, hand rotation movement, hand circle movement. Then we consider the fingers' movement. Since there are so many possible movement and will focus on the movement of index finger which is very useful in communication and interactions.

Hand Translation Feature
Translation feature indicates fingers and palm are moving together straightly without rotation. We calculate the cross correlation of velocity vectors between fingers F^i_v and palm P_v for all fingers. If the absolute values of these cross correlations are greater than 0.95, we recognize that the hands are moving straightly.

Hand Rotation Feature
Palm rotation features contains two parts. One is the difference of current palm normal P^t_N and previous palm normal P^{t-1}_N defined by DP_N. The other parts is the angle between difference of current palm DP_N and hand direction P_D. We then calculate the cross correlation of DP_N and hand direction P_D

Hand Circle Features
Hand circe feature indicates the palm is drawing a great circle. Same to hand rotation features, we calculate the first order difference between palm normals. Also make sure the hand is not rotating.

Index Key Tapping and Index Swipe
Figure 6 [4] shows examples of index key tapping and index swipe. Index key tapping and index swipe gestures are built based on index pointing static gesture. The difference between key tapping and swipe is the index only moves vertically for key tapping and moves horizontally for swipe. We then calculate the cross correlation between the direction of index finger velocity F_v^1 and the palm normal P_N. If the absolute cross correlation is greater than a threshold for example 0.94 we believe that the dynamic gesture is index key tapping. It indicates the movement of index is parallel with the palm norm. If the absolute cross correlation is smaller than a threshold for example 0.2, we believe that the dynamic gestures is index swipe. Because it indicates the movement of index is orthogonal to the palm normal.

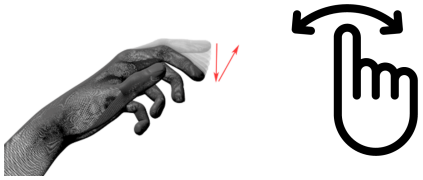

Figure 6: *Left: index key tapping. Right : index swipe*

Index Circling Direction Features
Figure 7[5] We also try to predict the circle direction whether it is clockwise or counter clockwise when the index is moving along a circle. We first calculate the first order difference of the index finger velocity between F_{vt}^1 and $F_{v(t-1)}^1$ denoted by DF_{vt}^1 denoted by CP. Then the cross product of F_{vt}^1 and DF_{vt}^1 is generated. The direction of cross product are different between clockwise circle and counter clockwise circle. We simply calculated the sign of the cross correlation result between CP and the direction of hand. If the sign is positive, circle moves clockwise, otherwise circle direction is counter clockwise.

Figure 7: *Example of index circling*

4 Evaluation and Failure Cases Analysis

When hands are correctly tracked, our methods could accurately detect the hand gestures. However there are some cases when the Leap Motion Controller fails to detect all the fingers. We analysis the failure cases to find out factors leading to misclassification.

[4]pictures source:https://developer.leapmotion.com/documentation
[5]pictures source:https://developer.leapmotion.com/documentation

Self-Occlusion

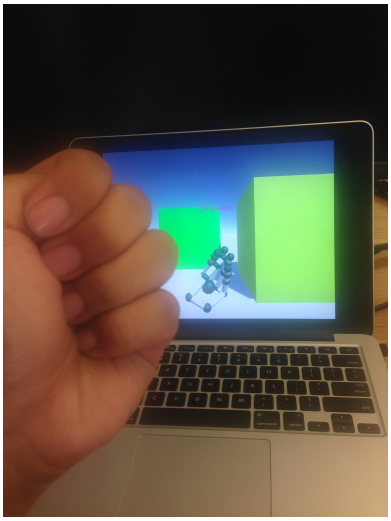

Figure 8: *Example of self occlusion*

Figure 8 shows an example where the fingers are self occluded by the palm. In that picture the Leap Motion Controller is put below the palm and can not get full data about the fingers. It predicts a gesture which is not correct. The Leap Motion Controller use IR to gather hand information in the space. When important fingers or regions are self-occluded by other hand parts, tracking data quality will be greatly reduced.

Distal phalanges

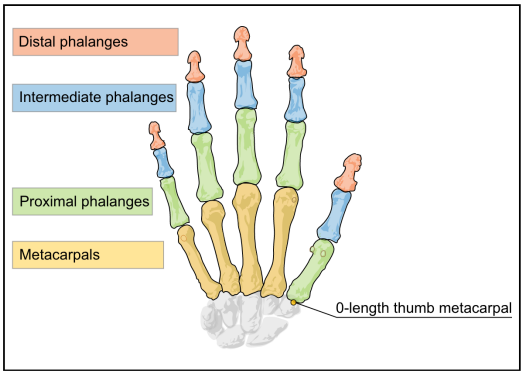

Figure 9: *Example of hand skeleton*

Skeletal Tracking Model is a standard hand model provided by Leap Motion. It simulates the actual hand skeleton.Figure 9[6] shows the picture of hand skeleton.

In our experiments, we find out the tracking data of distal phalanges of middle finger and pinky finger is not stable. Figure 10 gives two examples of wrong tracking of distal phalanges. Left image shows the orientation of middle finger's distal phalanges not bent while in the real hand it it straight. Right image shows the pinky is bent in

[6]pictures source:https://developer.leapmotion.com/documentation/objc/devguide/Intro_Skeleton_API.html?proglang=objc

real hand while it is straight in the Leap Motion Controller tracking data. Distal phalange of index finger is much stable. Therefore we should avoid use the tracking data of middle and pinky finger's distal phalanges.

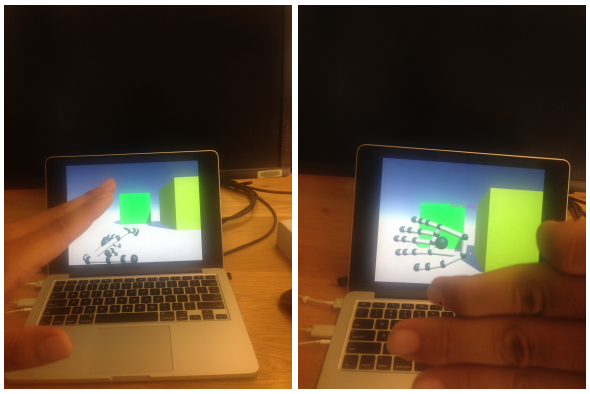

Figure 10: *Example of wrong distal phalanges tracking*

Detection Region
Currently the detection region for Leap Motion Controller is still small. Hand tracking data becomes unstable when hands are near the detection region boundaries.

Parameters
In the above feature descriptors, some parameters are associated with real hand sizes. If the hand sizes and corresponding parameters are not matching, failure cases happen.

Error accumulation
For hand movement gestures, we also use first order differences. These values are less accurate and less robust due to error accumulation.

5 Applications

One of the applications is to use these gestures to navigations in the Virtual Reality Environment. In our demo, we adopt three different gestures with left and right hands to represent moving forward/backward, turn left/right, roll clockwise/counter-clockwise. Using these gestures the users could control the movement instead of using some extra control devices for example control panel. Figure 11 shows examples of navigation in our demo.

6 Hand interaction

Hand interaction is implemented based on unity5. In unity5 the interaction between objects require at least one object is set to be rigidbody. In the interaction process, the first part is to detect when the two objects begin to touch each others. Unity 5 provides different collide region. We could choose the most accurate type when hand are touching the object we will receive a touching signal. However it requires a large amounts of calculation per frame and will reduce the speed and user experiences. We use simple cubes as the rigid body objects in our demo. Basic physical laws are already implemented to rigid body objects in unity5.

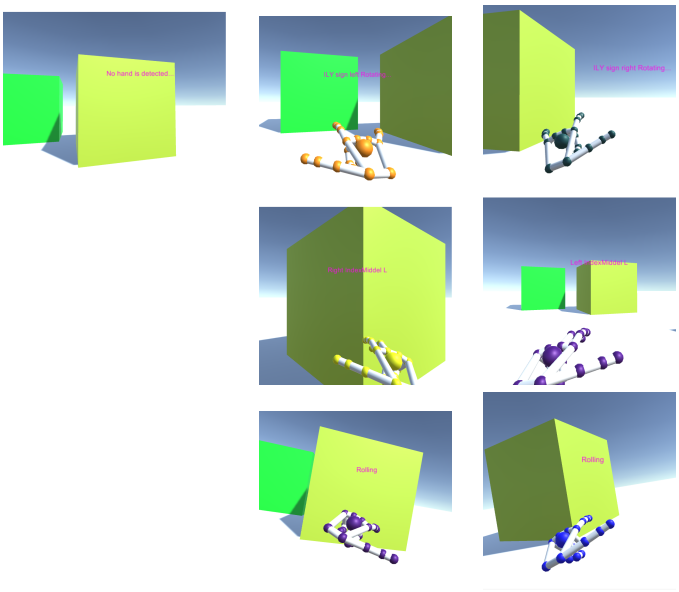

Figure 11: *Top left: Starting situation. The first line: Turn left, Turn right. The second line: Move forward. Moving backward. The last line: Roll clockwise, Roll counter clockwise*

7 Discussion

In our experiments, self occlusion usually leads to misclassification. The self occlusion is a common problem in all hand recognition systems for example the Kinect detection system which provides the depth information.

We therefore try to deal with such problems. We could use more than one Leap Motion Controllers to provide more accurately and continuously hand tracking data. Two Leap Motion Controllers could be put far from each others with nearly orthogonal angle. Then tracking data from two different Leap Motion Controllers is transformed into the same world space coordinates.

We assign different priorities to these two types of tracking data. When a hand is self-occluded from a Leap Motion Controller's view, we reduce the priority of the tracking data from that Leap Motion Controller and trust the other Leap Motion Controller. Due to the orthogonal angle between these two Leap Motion Controller, the other Leap Motion Controller could recognize the current hand since the hand is no longer self-occluded from the new Leap Motion Controller's view. In this environment, we could implement more complicated and accurate hand movement tracking system.

Acknowledgements

I would like to thank the wonderful teaching team of the course. Many thanks to Gordon Wetzstein and Robert Konrad for amazing lectures and helpful project discussion.

References

BACHMANN, D., WEICHERT, F., AND RINKENAUER, G. 2015. Evaluation of the leap motion controller as a new contact-free pointing device. *Sensors 15*, 1, 214.

BRACEGIRDLE, A., MITROVIC, S. T., AND MATHEWS, M., 2014. Investigating the usability of the leap motion controller: Gesture-based interaction with a 3d virtual environment.

COOPER, H., HOLT, B., AND BOWDEN, R. 2011. Sign language recognition. In *Visual Analysis of Humans: Looking at People*, T. B. Moeslund, A. Hilton, V. Krüger, and L. Sigal, Eds. Springer, Oct., ch. 27, 539 – 562.

MARIN, G., DOMINIO, F., AND ZANUTTIGH, P. 2015. Hand gesture recognition with jointly calibrated leap motion and depth sensor. *Multimedia Tools and Applications*, 1–25.

WEICHERT, F., BACHMANN, D., RUDAK, B., AND FISSELER, D. 2013. Analysis of the accuracy and robustness of the leap motion controller. *Sensors 13*, 5, 6380.