UNIVERSITY NAME

DOCTORAL THESIS

Hand Gesture Recognition via Leap Motion Sensor

Authors:
Jahangir IQBAL

Supervisor:
Dr. Irina VOICULESCU

A thesis submitted in fulfillment of the requirements for the degree of Doctor of Philosophy

in the

Research Group Name
Department or School Name

August 17, 2017

Declaration of Authorship

I, Jahangir IQBAL, declare that this thesis titled, "Hand Gesture Recognition via Leap Motion Sensor" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Date:

"Thanks to my solid academic training, today I can write hundreds of words on virtually any topic without possessing a shred of information, which is how I got a good job in journalism."

Dave Barry

University Name

Abstract

Faculty Name
Department or School Name

Doctor of Philosophy

Hand Gesture Recognition via Leap Motion Sensor

by Jahangir IQBAL

The Thesis Abstract is written here (and usually kept to just this page). The page is kept centered vertically so can expand into the blank space above the title too...

Acknowledgements

The acknowledgments and the people to thank go here, don't forget to include your project advisor...

Contents

Declaration of Authorship iii

Abstract vii

Acknowledgements ix

1 Background 1
1.1 Main Section 1 . . . 1
1.1.1 Subsection 1 . . . 1
1.1.2 Subsection 2 . . . 1
1.2 Main Section 2 . . . 1

2 Constructing Hand UI Model 3
2.1 Basic 3D Modeling . . . 3
2.2 Set Location Method . . . 4
2.3 Concurrency . . . 5

A Frequently Asked Questions 7
A.1 How do I change the colors of links? . . . 7

List of Figures

2.1 Hand Bone Model 3
2.2 JavaFx Group Node 4
2.3 UIHandSimple Constructor 5
2.4 setRotationByVector 5
2.5 setRotationByVector 6

List of Tables

List of Abbreviations

LAH	**L**ist **A**bbreviations **H**ere
WSF	**W**hat (it) **S**tands **F**or

Physical Constants

Speed of Light $c_0 = 2.997\,924\,58 \times 10^8\,\mathrm{m\,s^{-1}}$ (exact)

List of Symbols

a	distance	m
P	power	W ($\mathrm{J\,s^{-1}}$)
ω	angular frequency	rad

For/Dedicated to/To my…

Chapter 1

Background

1.1 Main Section 1

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aliquam ultricies lacinia euismod. Nam tempus risus in dolor rhoncus in interdum enim tincidunt. Donec vel nunc neque. In condimentum ullamcorper quam non consequat. Fusce sagittis tempor feugiat. Fusce magna erat, molestie eu convallis ut, tempus sed arcu. Quisque molestie, ante a tincidunt ullamcorper, sapien enim dignissim lacus, in semper nibh erat lobortis purus. Integer dapibus ligula ac risus convallis pellentesque.

1.1.1 Subsection 1

Nunc posuere quam at lectus tristique eu ultrices augue venenatis. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Aliquam erat volutpat. Vivamus sodales tortor eget quam adipiscing in vulputate ante ullamcorper. Sed eros ante, lacinia et sollicitudin et, aliquam sit amet augue. In hac habitasse platea dictumst.

1.1.2 Subsection 2

Morbi rutrum odio eget arcu adipiscing sodales. Aenean et purus a est pulvinar pellentesque. Cras in elit neque, quis varius elit. Phasellus fringilla, nibh eu tempus venenatis, dolor elit posuere quam, quis adipiscing urna leo nec orci. Sed nec nulla auctor odio aliquet consequat. Ut nec nulla in ante ullamcorper aliquam at sed dolor. Phasellus fermentum magna in augue gravida cursus. Cras sed pretium lorem. Pellentesque eget ornare odio. Proin accumsan, massa viverra cursus pharetra, ipsum nisi lobortis velit, a malesuada dolor lorem eu neque.

1.2 Main Section 2

Sed ullamcorper quam eu nisl interdum at interdum enim egestas. Aliquam placerat justo sed lectus lobortis ut porta nisl porttitor. Vestibulum mi dolor, lacinia molestie gravida at, tempus vitae ligula. Donec eget quam sapien, in viverra eros. Donec pellentesque justo a massa fringilla non vestibulum metus vestibulum. Vestibulum in orci quis felis tempor lacinia. Vivamus ornare ultrices facilisis. Ut hendrerit volutpat vulputate. Morbi condimentum venenatis augue, id porta ipsum vulputate in. Curabitur luctus tempus justo. Vestibulum risus lectus, adipiscing nec condimentum quis, condimentum nec nisl. Aliquam dictum sagittis velit sed iaculis. Morbi tristique augue sit amet nulla pulvinar id facilisis ligula mollis. Nam elit libero, tincidunt ut aliquam at, molestie in quam. Aenean rhoncus vehicula hendrerit.

Chapter 2

Constructing Hand UI Model

2.1 Basic 3D Modeling

The Leap Motion Java API's Hand class contains all the possible functions one might need to use when gaining more information about the heirarchical structure of this Java object. For example, given a Hand class object "hand", we can access the fingers objects for this hand via "hand.fingers()". Each Finger object contains four Bone objects which are indexed from 0-3. The Bone class does contain an Enum Type that allows one to easily access them via their anatomical names (distal, intermediate, proximal, metacarpal) rather than just using a numerical index. In abstract terms, the Bone object is a vector of sorts and the ends of this bone vector represent the joints at which the bone attaches to its neigboring bones. These "joints" can be accessed via the prevJoint() and nextJoint() methods which respectively return a vector position of the Bone closer to the wrist and of the Bone object endpoint closer to the tip of the finger. The figure below shows the bones and joints of the hand for which the Leap Motion sensor records data.

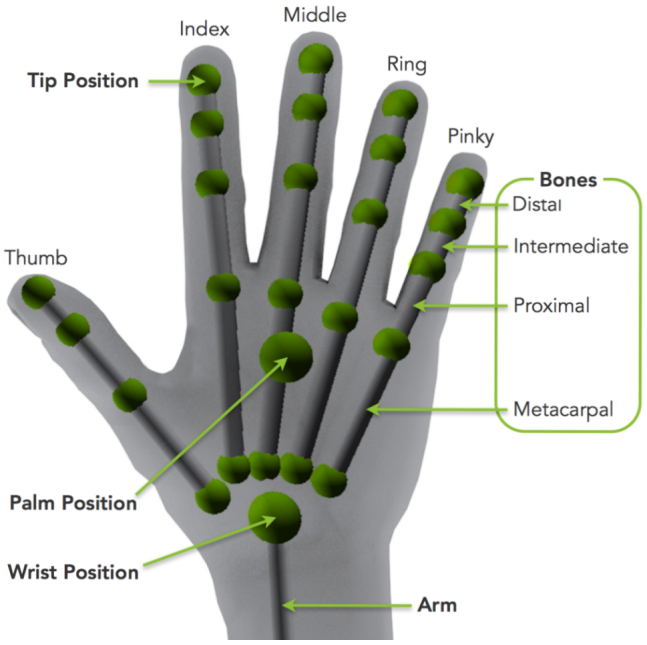

FIGURE 2.1: The Bones of the hand that Leap Motion device records data on.

A Hand object is only valid if it is detected by the Leap Motion device to be a physical object; if a Hand object is created via code using the Hand() constructor, that hand is considered "invalid" and will return true when the hand.invalid() method

is called it. The information contained within a valid Hand object read in from the device is Read Only and can not be changed or updated.

The Leap Motion device records numerical data about the hand and finger positions. Using the Hand class provided by the Leap Motion Java API and described above in the previous paragraph, a graphical model was constructed. For this GUI construction, a graphical representation of the hand was built using basic 3D geometric classes provided by the JavaFx framework. The bones of the hand model were represented by the Cylinder class and the joints were represented by the Sphere JavaFx class. This Hand model is contained inside the `UIHand_Simple`Java class. This class extends a base abstract UIHand class which itself inherits from the Javafx class called Group. Group is a type of Node in Javafx that contains an ObservableList of children Nodes that will be added to the Javafx Scene Graph in the order that they are added to the Group. An important point to note is that any transform, effect or property change applied to a Group will also be applied to all the children of that group. The figure below shows an example of how a Group Node can can contain multiple children nodes.

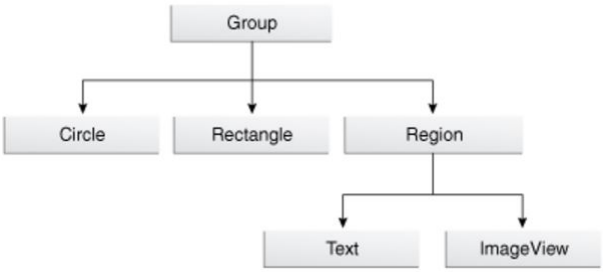

FIGURE 2.2: The Group Node structure in the JavaFx framework.

The `UIHand_Simple`class stores all the fingers bones in two dimensional array of Cylinder objects and all the respective joints in a different two dimensional array of Spheres. These arrays are of dimensions 5x3 to account for the five fingers and the three types of primary finger bones: distal, intermediate and proximal. In addition to these arrays, there is an array containing the four metacarpal bones of the hand; the thumb does not have a corresponding metacarpal bone like the other fingers. The also contains two more cylinders and a sphere to construct the palm section of the hand. To provide the hand with a uniform, well-blended color shading a PhongMaterial object is set as the hand's material property. Below is a figure showing an excerpt from code which shows the initialization of the `UIHand_Simple`and part of its constructor.

Each element of the hand, such as all the cylinders and spheres representing the various bones and joints, is added to the children of the encompassing parent group that represents the hand.

2.2 Set Location Method

The `UIHand_Simple`class also contains a method that allows for the graphical hand to be positioned according to the exact postions recorded in a Leap Motion Hand object. This method, which is called setLoc(Hand h), goes through each of the fingers and their respective bones and joints and sets the position and rotation of these these Javafx nodes based upon the Hand object passed in. This method relies on a helper class called ViewMath which contains static methods that are called to position each

```
public class UIHand_Simple extends UIHand {

    private static float fingerRadius = 14f;

    private Cylinder[][] fingerBones;
    private Sphere[][] fingerJoints;
    private Cylinder[] knuckleSpans;
    private Cylinder palmWrist;
    private Cylinder palmPinky;
    private Sphere pinkyJoint;

    public UIHand_Simple(Color color, boolean wireframe) {
        super();

        PhongMaterial dark = new PhongMaterial(color);
        PhongMaterial light = new PhongMaterial(color.brighter());

        fingerBones = new Cylinder[5][];
        for (int i = 0; i < 5; ++i) {
            fingerBones[i] = new Cylinder[3];
            for (int j = 0; j < 3; ++j) {
                fingerBones[i][j] = new Cylinder();
                fingerBones[i][j].setMaterial(dark);
                fingerBones[i][j].setRadius(fingerRadius / ViewMath.radiusScaleFactor);
                if (wireframe) fingerBones[i][j].setDrawMode(DrawMode.LINE);
            }
```

FIGURE 2.3: A snippet of code showing how a the UIHandSimple class, representing the graphical hand, is constructed.

individual cylinder representing a bone. Two of the important methods in ViewMath are setPositionByVector(Node n, Vector v) and setRotationByVector(Node n, Vector v). The method setPositionByVector sets the translate properties of the Javafx Node passed in to the xyz position recorded in the vector. The setRotationByVector method rotates the Javafx Node passed into it by the direction which is represented by the second argument vector. This method first takes the direction and "corrects" it by flipping the z-value. This is done because Javafx's coordinate system has the Z-axis increasing outward from the computer screen, while Leap Motion has the Z-axis increasing into the screen. The setRotationByVector finds the angle of rotation finding the the angle of the passed in direction to the Y-axis. In addition to the angle of rotation, the axis upon which the rotation will occur also needs to be defined. The axis of the rotation is found by taking the cross-product between the Y-axis and the "corrected" direction. Below is a snippet of code that shows the setRotationByVector method.

```
//This method rotates a given JavaFx node to point in the direction passed in
public static void setRotationByVector(Node node, Vector direction) {
    //Correct the direction to correspond to JavaFx Coordinate system
    Vector correctedDirection = new Vector(direction.getX(), direction.getY(), -direction.getZ());

    //Find the angle of the direction to the y-axis; in degrees
    double angle = correctedDirection.angleTo(Vector.yAxis()) * 180 / Math.PI;

    //Find the axis of rotation by taking the cross product of the corrected direction with the y-axix
    Point3D axis = vectorToPoint(correctedDirection.cross(Vector.yAxis()));

    //Set the axis and angle of rotation on the Node object
    node.setRotate(angle);
    node.setRotationAxis(axis);
}
```

FIGURE 2.4: A snippet of code showing how the rotation is set for an arbitrary Node object of the JavaFX Hand Model.

2.3 Concurrency

to be written..

–read some code about javafx application threading stuff. research. write it in a paragraph and explain what it means.

–look at the code that does task setting. read it a little try to understand. make some notes about what it means and how its wired. write about it.

```
// Hello.java
import javax.swing.JApplet;
import java.awt.Graphics;

public class Hello extends JApplet {
    public void paintComponent(Graphics g) {
        g.drawString("Hello, world!", 65, 95);
    }
}
```

FIGURE 2.5: A snippet of code showing how the rotation is set for an arbitrary Node object of the JavaFX Hand Model.

– also write about the frame controller methods offered by leap ui. and talk about the controller2. read code, understand, make notes. and then write about it.

Appendix A

Frequently Asked Questions

A.1 How do I change the colors of links?

The color of links can be changed to your liking using:

`\hypersetup{urlcolor=red}`, or
`\hypersetup{citecolor=green}`, or
`\hypersetup{allcolor=blue}`.

If you want to completely hide the links, you can use:

`\hypersetup{allcolors=.}`, or even better:
`\hypersetup{hidelinks}`.

If you want to have obvious links in the PDF but not the printed text, use:

`\hypersetup{colorlinks=false}`.