KinecTool: a repository of open Kinect projects

Niccoló Fabbri
E-mail address
niccolo.fabbri@stud.unifi.it

Abstract

Hand gesture based Human-Computer-Interaction (HCI) is one of the most natural and intuitive ways to communicate between people and machines. Compared to the entire human body, the hand is a smaller object with more complex articulations and more easily affected by segmentation errors. It is thus a very challenging problem to recognize hand gestures. In this project, I use Candescent NUI[1] libraries for hands and fingers tracking and I use Kinect SDK[2] for hand gestures. In conclusion, I propose two demos: one about the game rock paper scissors (RPS) to demonstrate the performance of Candescent NUI libraries in a real life application and other about the control of PowerPoint or PDF program using Kinect SDK.

1. Introduction

The goals of this paper are to propose two demos: one demo of the game rock paper scissors (RPS), in particular I focus on the possibility of recognizing the number of fingers present in the hand. Other demo to demonstrate the possibility of how to use the *Kinect SDK* to do basic gesture recognition to control *PowerPoint* or *PDF* program. Hand gesture recognition is an important research issue in the field of Human-Computer-Interaction, because of its extensive applications in virtual reality, sign language recognition and computer games. Existing vision-based approaches are greatly limited by the quality of the input image from optical cameras. Variations in lighting and background clutters would only worsen the problem. Hand gesture recognition concerns two challenging problems: hand detection and gesture recognition, namely how to robustly detect the hand and how to efficiently and accurately recognize the gesture of the hand. For hand detection the depth map obtained by Kinect sensor was used to detect the hand shapes. For the gesture recognition, even with the Kinect sensor, it is still a very challenging problem. Because typically, the resolution of a Kinect sensor is only 640x480. Although it works well to track a large object, e.g. the human body, it is difficult to detect and segment precisely a small object from an image at this resolution, e.g., a human hand that occupies a very small portion of the image.

1.1. Related Works

One of the first tasks was to find a library or an open project providing information on hands and fingers tracking. The choice between libraries was very small. Lots of existing projects provide hand tracking. Unfortunately, most of them use *openNI* or *NITE* on Linux instead of the *Kinect SDK* and secondly they use the skeleton tracking engine to follow the hand's gestures. That means one point or joint in space and no finger tracking. This makes it impossible to use for hand postures using fingers. The *Kinect SDK Dynamic Time Wrapping (DTW) Gesture Recognition* project offers the possibility to record gestures in 2D and recognize them but it only works with the skeleton joints. Then there was the *Robust Hand Gesture Recognition with Kinect Sensor* [3] project from Nanyang Technological University. The project

was very interesting and promising but a source code was not available. In a first time, I tried to replicate this paper but the results obtained were not up to expectations, thus I decided not to use the project. Finally, a project stood out, the *Candescent NUI*. It provides full hand recognition with finger points, palm, depth, volume, etc. The open project was using the *OpenNI* library but had just been updated to the *Kinect SDK*. The *Candescent NUI* library offers everything this project needs to be developed so it was selected as the base for the recognition.
For the second demo, the choice between libraries is not important because *Kinect SDK* version 1.8 is enough. Most of the work concerning the *PowerPoint* controller is about gesture recognition to access a specific commands. Instead, I propose a solution that improve use of the controller, extending functionality and introducing a better accuracy. This software, in union with the Kinect $Xbox$ 360 device, offers a set of gestures to start presentation, close presentation and move slides, in addition to a set of buttons to perform the control of the skeleton tracking.

2. Tools and techniques employed

Implementation of these demos have been developed entirely in C# code. For a correct execution you need to install some additional packages:

- Kinect for Windows SDK v1.8 [2]
- C# programming language
- Microsoft Visual Studio 2012 [4]
- Candescent NUI library [1]
- OpenNI library [5]

The **Kinect for Windows Software Development Kit** (SDK) enables developers to create applications that support gesture and voice recognition, using Kinect sensor technology on computers running Windows 7, Windows 8, Windows 8.1.

C# programming language is designed for building a variety of applications that run on the .NET Framework. C# is suggest because it is full compatibility with SDK and because it is simple, powerful, type-safe, and object-oriented.

Microsoft Visual Studio that is an integrated development environment (IDE) from Microsoft. It used to develop computer programs for Microsoft Windows. It uses Microsoft software development platforms such as Windows API, Windows Forms, Windows Presentation Foundation, Windows Store.

Candescent NUI is a set of libraries created by Stefan Stegmueller. It is designed for hand and fingers tracking using Kinect depth data. It has been developed in C# with OpenNI and Microsoft Kinect SDK.

OpenNI framework is an open source SDK used for the development of 3D sensing middleware libraries and applications

3. Candescent NUI

Candescent NUI provides lots of useful information for hand and fingers tracking. It starts by detecting close objects, two at most. These objects are then treated to get hand features. Beware, if an object is actually not a hand, like a head for example, the algorithms will slow down the application because they will not be able to extract the features and might probably crash. If the objects are hands, features will be extracted. A convex hull algorithm gives the finger tips position (X, Y, Z), direction, etc. Other features like the volume of the hand, the palm position, the number of fingers, each finger's base position and id. (Figure 1)

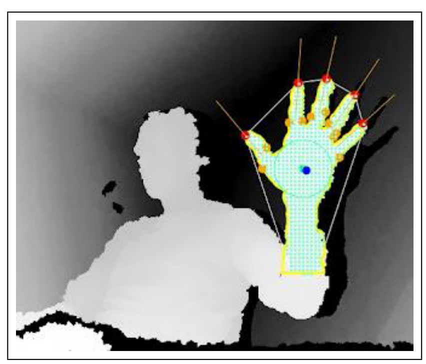

Figure 1. Candescent NUI

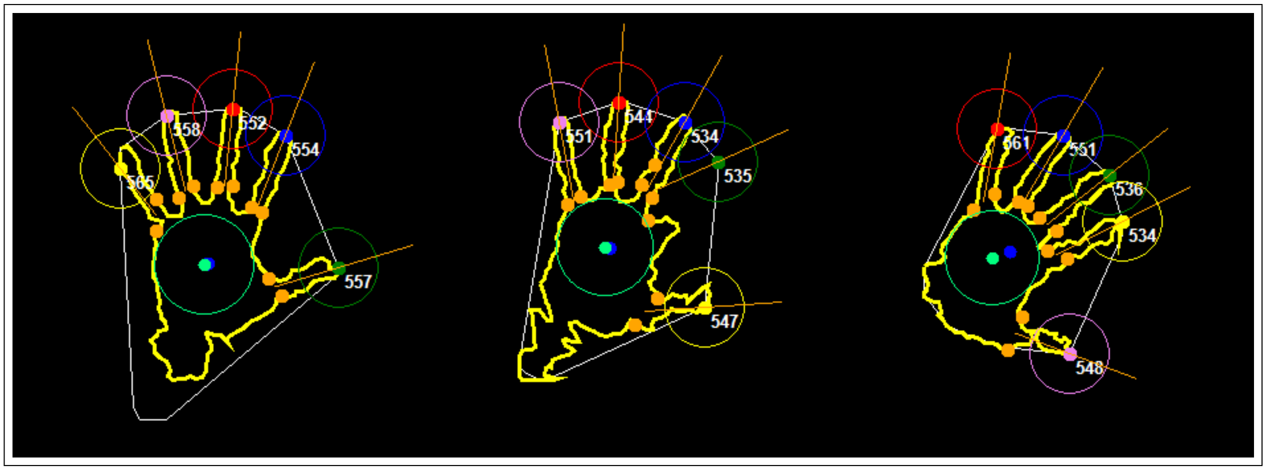

Figure 2. Fingers' order: 1: Red, 2: Blue, 3: Green, 4: Yellow, 5: Pink

3.1. Bases to start with Candescent NUI

To be able to use the Cansdescent library the project must add references of the library's dll:

- *CCT.NUI.Core*
- *CCT.NUI.HandTracking*,
- *CCT.NUI.KinectSDK*
- *CCT.NUI.Visual*

They need to be added in the references of the project. To use Candescent with the Kinect SDK instead of *openNI* as it can, the data source must be configured properly.

3.2. Kinect field of view

Kinect for Windows detects objects at 400mm in near mode. With *Candescent NUI* hands have to be at least at 500mm (minimum depth) from the camera to be detected correctly. The maximum distance is 800mm, further than this the hands are too far to be detected and tracked. The perfect distance for the tracking is around 650mm.

3.3. Hands detection

Candescent NUI can detect up to 2 hands simultaneously. These are stored in a list of *HandData*. If two hands are detected, *Hand[0]* always will be the hand on the right side and *Hand[1]* the one on the left side. Each hand has an id, a location, a volume, a palm, fingers, a contour shape and a convex hull as main data.

3.4. Fingers detection

The way the fingers are detected is special. There is no notion of thumb, little finger or middle finger. The fingers are numbered from 0 to 4 for each hand. Actually a hand can have more than 5 fingers with this library as the fingers are summits of the convex hull algorithm. The first finger is always the highest one and then the other fingers are ordered clockwise. That means if the hand rotates, the first finger and the others change position, there's no lock mode on fingers. The fingers data are in a List *FingerPoints*. There is *FingerCount* which returns the exact number of fingers at the moment, Figure 2.
The fingers are defined by the summits of the convex hull algorithm. Summits are usually above the palm of the hand. But sometimes those summits are under it, at the wrist level which makes false finger detections. To prevent that the *detectFingers* method takes into account only fingers detected above the palm of the hand. This takes away the possibility of having the thumb beneath the palm or having the hand downward but those options are rarely used.

4. Kinect SDK

From Microsoft Research labs, the Kinect development kit for Windows 7 allows developers to create their own interfaces and applications for windows. It was designed to be used with C++

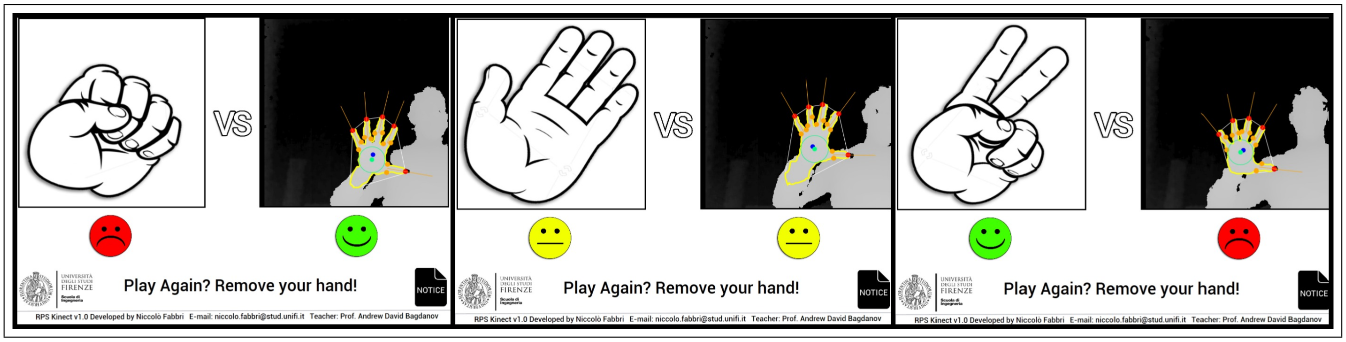

Figure 3. Rock Paper Scissors The Game - Interface.

and C#. The paper only focuses on C# with Microsoft Visual Studio.
As described in [4], Kinect SDK offers raw sensor stream to access to low-level streams from the depth sensor, color camera sensor, and four-element microphone array. Skeletal tracking: the capability to track the skeleton image of one or two people moving within the Kinect field of view for gesture-driven applications. Advanced audio capabilities: audio processing capabilities include sophisticated acoustic noise suppression and echo cancellation, beam formation to identify the current sound source, and integration with the Windows speech recognition API.

5. Application

The goals of these demos are to showcase a real-life applications. Both these demos run in real time. Unfortunately *Rock Paper Scissors The Game* is not always stable and therefore was evaluated as beta game. Nobody usability test is realised to *Rock Paper Scissors The Game*.

5.1. Rock Paper Scissors The Game

Rock-paper-scissors is a traditional game. The rule is rock breaks scissors; scissors cut paper; and paper covers rock. In this demo, I built a Rock-paper-scissors game system played between a human and a computer. Three hand gestures are defined as 3 different weapons in the game, as shown in Figure 3, which can be recognized by our system, according to the game rule, my system can decide who is the winner.

5.2. Kinect PPT/PDF Control

This program aims to build a gesture recognition system that uses natural gestures to control a *PowerPoint* or *PDF* presentation. When people give *PowerPoint* presentations, they usually have a clicker object that can control the slides remotely. However, holding onto the clicker during the presentation occupies the hand. When a person's hand is already occupied, the range of motions and gestures that can be performed is limited. This limitation is not necessarily a physical limitation; the presenter may simply be (potentially unconsciously) unwilling to perform certain gestures while their hand is occupied. The primary goal of this system is to free the user from these restraints and automatically to react to the naturally gestures throughout the presentation. In many other system gestures are contrived and unnatural. This system focuses on recognizing natural gestures that users would be likely to perform during a *PowerPoint* presentation even without the system. By focusing the system's attention on natural gestures, the user should not have to think about performing artificial and awkward gestures to control the *PowerPoint*. The gesture recognition system should not hinder the user, allowing the human-computer interaction to be as seamless and intuitive as possible.

Main Features

The *Kinect PPT/PDF Control* interface is show in Figure 4, this system improves some important features:

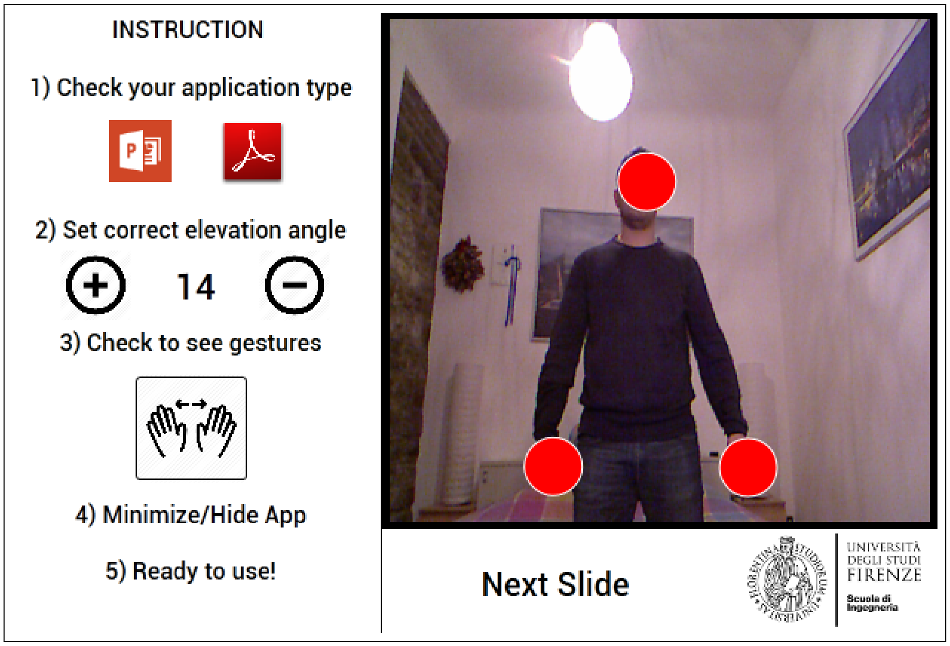

Figure 4. Kinect PPT/PDF Control - Interface.

- **Mode Switch**: Through these buttons you can change the program to control. The choice is between *PowerPoint* and a generic *PDF Reader*.
- **Set Elevation Angle**: Through plus or minus buttons, you can change the elevation angle of the device. This setting is more important if you want to present while sitting.
- **Gesture Button**: Through this button you can see the main gestures.
- **Bounding Box**: Through this text box you can see the recognized gesture.

6. Usability test - Kinect PPT/PDF Control

The users are asked to complete a series of routine tasks supervised by a moderator with the aim of estimating system:

- Learnability
- Easiness of use
- Accuracy
- Effort and Comfort
- Consistency

6.1. Methodology

Before starting the test, the controller is introduced to the user through a presentation of the hands and arms gestures while the supervisor explains briefly the main features of the device. He also gives some useful tips for the best experience and answers the user's questions, if any.

Test Protocol

The testing protocol is composed by 3 consecutive task oriented test. In each test, as the user is asked to complete a series of routine tasks, the supervisor is allowed to help the user by giving some tips.

(a) Start presentation

(b) Select a specific mode

(c) Go to a precise slide

(d) Come back to a precise slide

(e) Change elevation angle

(f) Switch from one mode to other

(g) Repeat all tasks to sitting

(h) Quit presentation

The study involved 17 participants between the ages of 18 and 68 who were unfamiliar with this project. The location of the usability evaluation was a room where the user can test the system in privacy and without too many distractions. The Kinect device was placed on the table in front of user; the user was asked to use the device standing in front, and a second time in a sitting position. The test has requested about 5 minutes for each task oriented test, totaling some 15 minutes to user.

6.2. Results

The system usability is evaluated by asking the users to answer 20 SEQ[1] over a 7-point Likert scale to assess their opinion. For negative scores I asked for an explanation in order to understand better the issue. Most of the participants (84%) are satisfied with the *Kinect PPT/PDF Control* system. The majority of participants (79%) agree they learned to use the control system after a little practice and that most people will learn to use it quickly (81%). Help given during the test is important to understand the working principle (56%). Nobody reported difficulty remembering the gestures (27%) or felt uncomfortable during operation: the position and the posture are considered good by many users (70%) also because you can use the system in both modes: standing or sitting. Most of the participants agree that the gestures work correctly: the accuracy of the application is considered sufficiently positive (58%)[2]. The majority of participants (83%) think they completed the tasks quickly. Arm is the most strained part but the effort is considered acceptable by most of the participants (60%)[3].
The essay question *Suggestions* (without percentage) was very interesting: the majority of participants suggested the possibility to change gestures or to create them custom. In particular the motion gestures, previous and next slides, it was suggested to insert new gestures that had more affinity with the scrolling of pages of a book as it concerns the field of the human computer interaction. Furthermore, as regard to the closing gesture is not appreciate more. I asked to each candidate the reasons and how to resolve: the closing gesture, cross arms, is not appreciate because the user does not consider it *suitable* to the environment in which he will expose his presentation, for example university or office. The user requires, as closing gesture, a gesture more moderate. Furthermore, it was suggested the possibility to integrate a gesture that blocks the current slide so that the presenter had a behavior as natural as possible, without fear of noises. With the same gesture, release the control of slide and continue the presentation. There is not a problem with start gesture. See table 1 for a detailed report.

[1]Single Ease Question
[2]Obtained by subtraction of question 9
[3]Obtained by subtraction of question 18

Conclusions

In this study, I tested two different ideas for interaction Human Computer: *Rock Paper Scissors The Game* and *Kinect PPT/PDF Control*.
The results of the test, only for *Kinect PPT/PDF Control*, shown that the system is good, the idea to consider hand gestures as natural interaction can be interesting. For the first demo, *Candescent NUI* provides accuracy to dynamic hand and fingers detections but result are usually noisy, so I decided not to use the project and evaluate as beta game. It has not been possible to simulate fully the RPS dynamic game.
For the second demo, *Kinect PPT/PDF Control*, the result are promising. The *Kinect SDK* provides accuracy to skeleton tracking and provides enough about the control gesture. This demo simulate correctly the clicker object that the user can use to control the slides remotely. The result highlight that posture during the operation (both modes: sitting or standing) allow a prolonged usage.
In conclusion, I can affirm that *Kinect PPT/PDF Control* may be an useful alternative in situations in which using conventional input devices (mouse, touchscreen or clicker object) is difficult, for example if you have a busy hand or other obstacle.

N	Questions	Mean rating	σ	% Agree
1	Age	27.58	10.92	/
	Task specific questions			
2	Advance to the next slide works properly	5.1	0.9	73%
3	Going to the previous slide works properly	5.0	1.1	72%
4	Plus or minus elevation angle buttons work properly	5.8	0.5	83%
5	Start gesture works properly	4.9	1.2	71%
6	Exit gesture works properly	4.8	1.8	69%
7	Gestures button works properly	5.9	0.8	84%
8	I am able to complete the tasks quickly	5.8	1.5	83%
	General questions			
9	The application makes many errors	3.4	0.5	48%
10	I feel very uncomfortable during operation	2.5	1.0	36%
11	The system is easy to use	4.9	0.8	70%
12	The gestures are difficult to remember	1.9	0.6	27%
13	I believe that the gestures are innate	4.8	1.3	69%
14	I learn to use the system better after a little time	5.5	1.2	79%
15	Position and posture are comfortable	4.9	0.7	70%
16	Help given during the test is very important to understand how the device works	3.4	3.9	56%
17	Most people will learn to use the system quickly	5.7	1.2	81%
18	Arms effort	2.8	1.0	40%
19	Overall, I am satisfied with the system	5.9	0.5	84%
20	Suggestions	/	/	/

Table 1. Detailed report. Single Ease Question on a **1** (*Strongly disagree*) to **7** (*Strongly agree*) scale. *Percent Agree: (%) Ratings greater than 4*

Future Developments

A future development may be to integrate the possibility to control video inside presentation. For this advancement, I will have suppose to do a simple tracking of the right or left hand *joint*[4]: when I have a close hand, it is the scrolling of the mouse, when I have a open hand (5 fingers), this could be a left click, that permits to start the video. Another possible scenario, that requires a little modification, may be the usage during a surgery, to browse some patients' clinical test, in a sterile operating room without physically touching a device and so destroying the sterile field.

[4]Connects two bones of a skeleton

References

[1] Candescent NUI Project, 2012 `http://candescentnui.codeplex.com/`

[2] Kinect for Windows SDK v1.8, 2013 `https://www.microsoft.com/en-us/download/details.aspx?id=40278`

[3] Z. Ren, J. Yuan, J. Meng and Z. Zhang, "Robust Part-Based Hand Gesture Recognition using Kinect Sensor". In: *ACM Multimedia (ACM MM)*, Arizona, USA, 2011

[4] Microsoft Visual Studio Professional 2012, 2012 `https://www.microsoft.com/it-it/download/details.aspx?id=30682`

[5] OpenNI Project, 2015 `http://www.openni.ru/`

[6] Kinect from Wikipedia, 2016 `http://en.wikipedia.org/wiki/Kinect`