Universite de Caen Normandie

Master 2 Imalang

Annual project

Skeleton tracking using Kinect

Author:
Vaishali Chaudhari

Supervisor:
Dr. Youssef Chahir

2016-2017

Contents

List of Figures

1 Introduction

In this annual project, we are interested to track complete skeleton of human body. Skeleton tracking is not just about tracking the joints by reading player (object) information; rather it tracks the complete body movement.The complete skeleton-tracking feature is built on the depth depth data processing, internal machine learning, and color vision algorithms.

We tracked human skeleton in 2 modes: standing and seated. In standing mode 20 bone joints are tracked , while in seated mode 10 joints are tracked. The tracked skeletons are recored into frames. The skeleton frames are stored temporarily. Maximum 1000 skeleton frames are stored. In the future, we can use this frames for the analysis of skeleton.

1.1 Development tools and software

- Microsoft Visual Studio 2010 Express or higher editions of Visual Studio
- Microsoft .NET Framework 4.0 or higher
- Kinect for Windows SDK 1.x

1.2 Hardware

The Kinect for Windows sensor

We can use the Kinect for Windows or the Kinect for Xbox sensor for your development. The Kinect for Windows sensor comes with an external power supply, which supplies the additional power, and a USB adapter to connect with the system. For the latest updates and availability of the Kinect for Windows sensor, we can refer to `http://www.microsoft.com/en-us/kinectforwindows/site` .

1.3 Features of the Kinect for Windows SDK

Kinect provides a **Natural User Interface (NUI)** for interaction using body motion and gesture as well as spoken commands.

- Capturing and processing the color image data stream
- Processing the depth image data stream
- Capturing the infrared stream
- Tracking human skeleton and joint movements
- Human gesture recognition
- Capturing the audio stream
- Enabling speech recognition
- Adjusting the Kinect sensor angle
- Getting data from the accelerometer

- Controlling the infrared emitter

Real-time human pose recognition is difficult and challenging because of different body poses, sizes, dresses, heights and so on. In the next part, we describe the methodology of skeleton tracking using kinect sensor.

2 Skeleton tracking

2.1 Kinect sensor

Kinect sensor returns raw depth data from which we can easily identify the pixels that represent the skeleton (tracking object) . Using Kinect SDK, we can track up-to 6 skeletons and up-to 20 joints of each skeleton. The following Figure 1 shows the overall process flow that creates joint points from raw depth data:

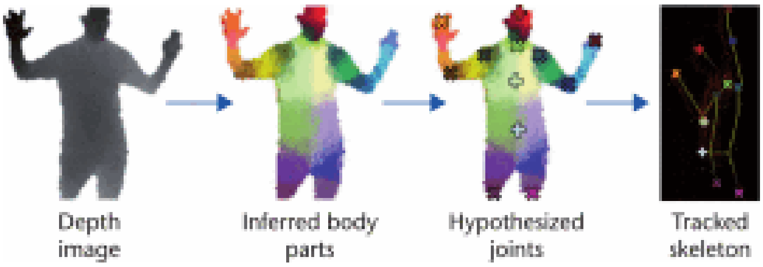


Figure 1: Kinect skeletal tracking process [3]

1. Depth data is processed in the rendering pipeline process and matches with decision forest labeled data and generates the inferred body segments.
2. Once all parts are identified based on the labeled data, the sensor identifies the body joints.
3. The sensor then calculates the 3D view from the top, front, and the left of the proposed joints.
4. Then the sensor starts tracking the human skeleton and body movement based on the proposed joint points and the 3D view.

Kinect uses trained tree structure (known as **decision tree**) to match the data for a specific type of human body. This tree is known as **Decision Forest**.

Tracking modes

The Kinect for Windows SDK provides us with a set of APIs that allow easy access to the skeleton joints. There are 2 modes to track the skeleton: standing and seated .

Standing

The SDK uses multiple channels to detect the skeleton. The default channel tracks all 20 skeletal joint positions with the *Tracked, Not Tracked, or Inferred* tracking mode. The SDK supports the tracking of up to 20 joint points. Each and every joint position is identified by its name (head, shoulders, elbows, wrists, arms, spine, hips, knees, ankles, and so on), and the skeleton-tracking state is determined by either Tracked, Not Tracked, or Position Only. By considering the Hip Center as root, there could be five different sequences as shown in the following Figure 2. As shown in Figure 2 we can have a total five sequences of bones (1 to 5) . We can move and orient the entire skeleton in the skeleton space by translating and rotating the root joint.

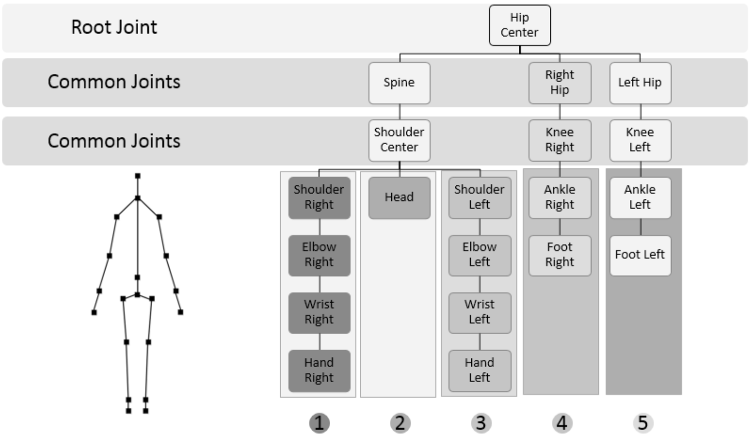


Figure 2: Bone sequence of default or standing mode

Seated

The Kinect for Windows SDK also supports tracking of a seated skeleton. You can change the tracking mode to detect a seated human body that returns up to 10 joint points. The following diagram 3 shows the hierarchical bone sequence for a seated skeleton with Shoulder Center as root joint:

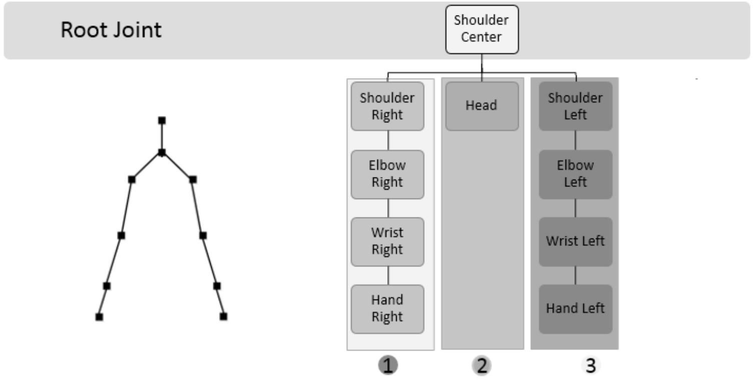


Figure 3: Bone sequence seated mode

2.2 Windows Presentation Foundation(WPF)

WPF , which stands for Windows Presentation Foundation, is Microsoft's latest approach to a GUI (Graphical User Interface) framework, used with the .NET framework [1] .

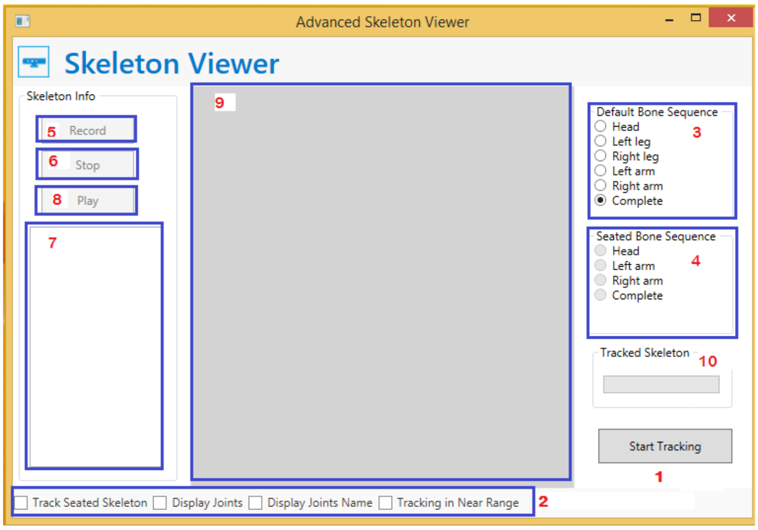
WPF advantages

- It's newer and thereby more in tune with current standards
- Microsoft is using it for a lot of new applications, e.g. Visual Studio
- It's more flexible, so you can do more things without having to write or buy new controls
- When you do need to use 3rd party controls, the developers of these controls will likely be more focused on WPF because it's newer
- XAML makes it easy to create and edit your GUI, and allows the work to be split between a designer (XAML) and a programmer (C#, VB.NET etc.)
- Data-binding, which allows you to get a more clean separation of data and layout Uses hardware acceleration for drawing the GUI, for better performance
- It allows you to make user interfaces for both Windows applications and web applications (Silverlight/XBAP)

3 Experiments and results

3.1 Introduction

We designed GUI shown by Figure 4 using WPF framework and C# programming language. Let's see the description of different components denoted by numbers in Figure 4. We can say this is an integrated solution to explore skeleton tracking in better way.



Default Bone Sequence
○ Head
○ Left leg
○ Right leg
○ Left arm
○ Right arm
● Complete

Seated Bone Sequence
○ Head
○ Left arm
○ Right arm
○ Complete

□ Track Seated Skeleton □ Display Joints □ Display Joints Name □ Tracking in Near Range

Figure 4: Skeleton tracking GUI

- **Start tracking (1)**
  Button to start tracking

- **Joints (2)**
  Different checkboxes to display extra parameters of skeleton e.g., *Display Joints, Display Joints Name*. *Tracking Seated Skeleton* checkbox is for to track skeleton in seated mode (only 10 joints are tracked) . We can enable near range tracking using *Tracking in Near Range* checkbox.

- **Default Bone sequence (3)**
  Standing mode of sequence tracking of bones. We can track complete skeleton or part of skeleton like head, left leg, right arm etc. We can track complete skeleton (20 joints) using *Complete*.

- **Seated Bone sequence (4)**
  Track skeleton with sequence of seated mode.

- **Record (5)**
  Record skeletons.

- **Stop (6)**
  Stop recording.

- **Frameset (7)**
  Frame of tracked skeletons. At a time, maximum 1000 frames stored.

- **Play (8)**
  Played selected frames.

- **Tracking window (9)**
  In this window, we can see tracked skeleton.

- **Tracked Skeleton (10)**
  Notification of sensor tracking skeleton.

This application has the following features:

- Enable near range while tracking the skeleton in both default and seated mode

- Display the tracked joints, joint names, and drawing bones

- Visualize different bone sequences for both tracking modes

- Visual indication for total tracked skeleton

- Record and play a fixed set of skeleton collections

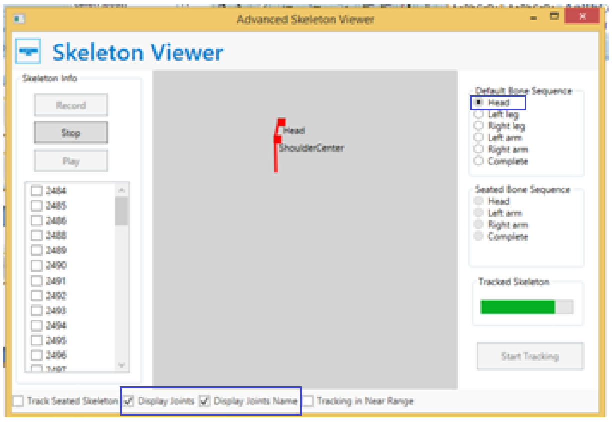
- Select a specific frame and display

3.2 Experiments

By using Skeleton viewer we track skeleton in standing mode with 20 joints. Below Figures describes different joint tacking. Selected parameters are displayed in blue square.

1. Standing or default mode

   - Head: Figure 5



□ 2484
□ 2485
□ 2486
□ 2488
□ 2489
□ 2490
□ 2491
□ 2492
□ 2493
□ 2494
□ 2495
□ 2496
□ 2497

Default Bone Sequence
● Head
○ Left leg
○ Right leg
○ Left arm
○ Right arm
○ Complete

Seated Bone Sequence
○ Head
○ Left arm
○ Right arm
○ Complete

□ Track Seated Skeleton ☑ Display Joints ☑ Display Joints Name □ Tracking in Near Range

Figure 5: head

   - Left leg: Figure 6

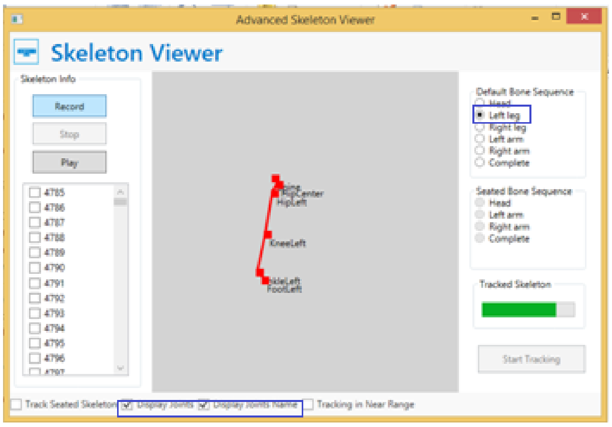


Figure 6: Left leg

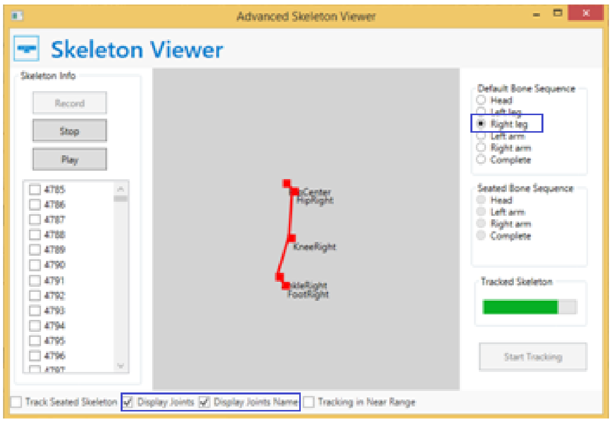


Figure 7: Right leg

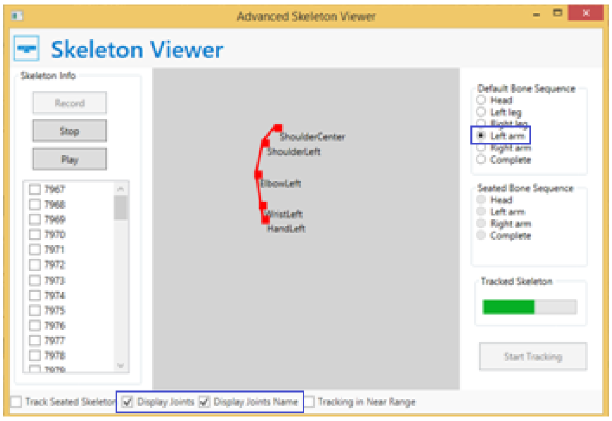
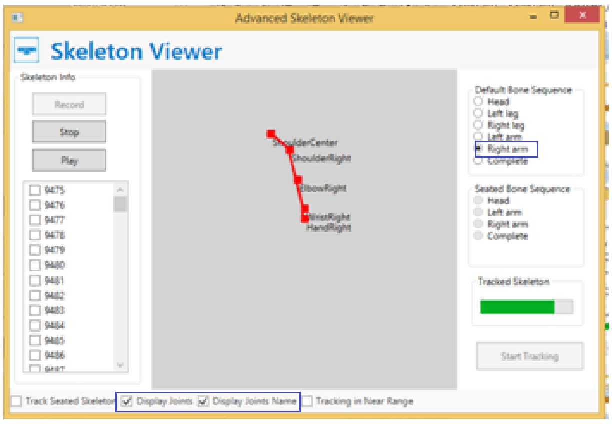


Figure 8: Left arm

- Right leg: Figure 7

- Left arm: 8
- Right arm: Figure 9

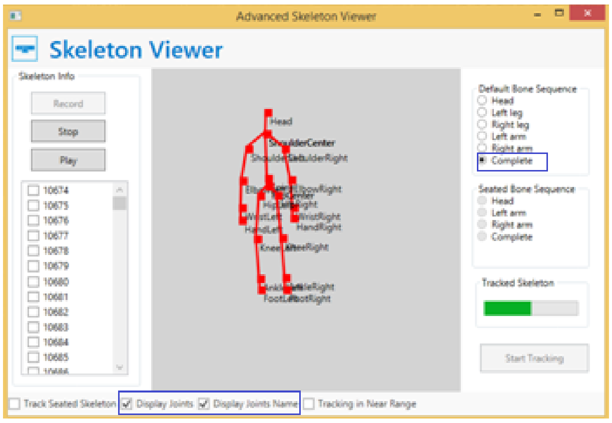


● Right arm

☑ Display Joints ☑ Display Joints Name

Figure 9: Right arm

- Complete skeleton: Figure 10



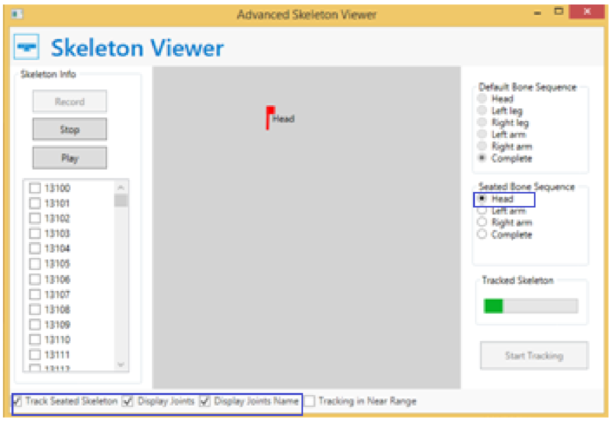
● Complete

☑ Display Joints ☑ Display Joints Name

Figure 10: Complete skeleton

2. Seated mode

- Head (seated mode): Figure 11



Default Bone Sequence
○ Head
○ Left leg
○ Right leg
○ Left arm
○ Right arm
● Complete

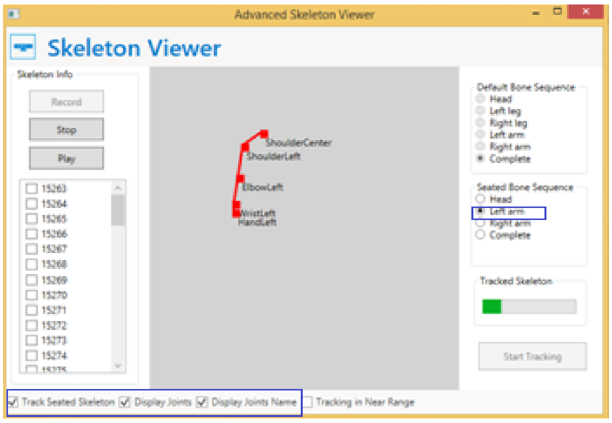
Seated Bone Sequence
● Head
○ Left arm
○ Right arm
○ Complete

Tracked Skeleton

☑ Track Seated Skeleton ☑ Display Joints ☑ Display Joints Name □ Tracking in Near Range

Figure 11: Head tracking (seated mode)

- Left arm (seated mode): Figure 12



Default Bone Sequence
○ Head
○ Left leg
○ Right leg
○ Left arm
○ Right arm
● Complete

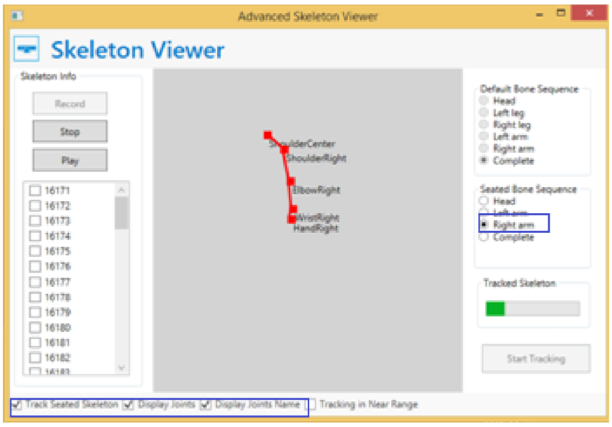
Seated Bone Sequence
○ Head
● Left arm
○ Right arm
○ Complete

Tracked Skeleton

☑ Track Seated Skeleton ☑ Display Joints ☑ Display Joints Name □ Tracking in Near Range

Figure 12: Left arm (seated mode)

- Right arm (seated mode): Figure 13
- Complete skeleton (seated mode): Figure 14

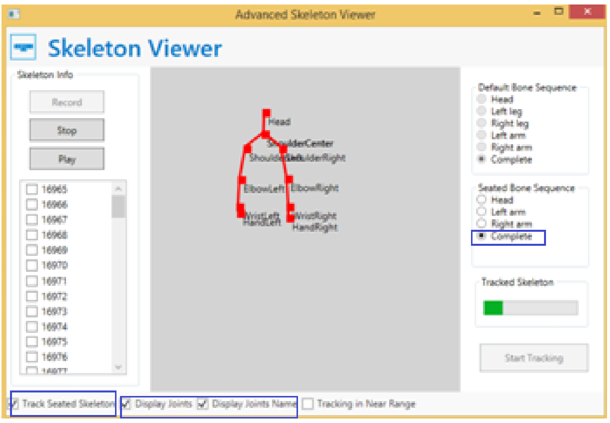


Default Bone Sequence
○ Head
○ Left leg
○ Right leg
○ Left arm
○ Right arm
● Complete

Seated Bone Sequence
○ Head
○ Left arm
● Right arm
○ Complete

☑ Track Seated Skeleton ☑ Display Joints ☑ Display Joints Name □ Tracking in Near Range

Figure 13: Right arm (seated mode)



Default Bone Sequence
○ Head
○ Left leg
○ Right leg
○ Left arm
○ Right arm
● Complete

Seated Bone Sequence
○ Head
○ Left arm
○ Right arm
● Complete

☑ Track Seated Skeleton ☑ Display Joints ☑ Display Joints Name □ Tracking in Near Range

Figure 14: Complete skeleton (seated mode)

4 Conclusion

We detected skeleton in several manner. Skeleton detection is useful in many applications such as object tracking, object recognition, military applications etc. Kinect sensor is very easy to use and set.

Programming using Kinect for Windows SDK and WPF is reduce sapce and complexity of coding. The results obtained in this project are very useful in evaluation of tracking joint points in virtual environment.

References

[1] Chris Sells and Ian Griffiths. *Programming Windows presentation foundation.* " O'Reilly Media, Inc.", 2005.

[2] Jarrett Webb and James Ashley. *Beginning Kinect Programming with the Microsoft Kinect SDK.* Apress, 2012.

[3] Zhengyou Zhang. Microsoft kinect sensor and its effect. *IEEE multimedia*, 19(2):4–10, 2012.