An HCI Project Report

on

Kinect Photo Gallery

Submitted by

Ashish Kumar Singh (13IT107)
Rishu Anand (13IT134)
Sugandh Kumar (13IT240)

VI Sem B.Tech (IT)

Under the guidance of

Prof. G R M Reddy, Mr. Ashwin T S and Mr. Pragnesh Thaker
Dept of IT, NITK Surathkal

in partial fulfillment for the award of the degree
of

BACHELOR OF TECHNOLOGY

In

INFORMATION TECHNOLOGY

At

Department of Information Technology
National Institute of Technology Karnataka, Surathkal.
April 2016

Abstract

The aim of this project is to to leverage on Kinect's physical gesture capabilities to create a photo gallery application. Various functionalities have been implemented like swipe right, swipe left, zoom in, zoom out, panning through the image, switching to different folders for images and changing the background of the application. This is to done to ease the interaction between the user and the photo gallery application. The native gallery app requires a lot of mouse and keyboard button clicks and there is difficulty in choosing options like zooming. navigation to a different image folder etc. Our application uses Kinect to offer hands-free interaction which is built using WPF(Windows Presentation Framework) and C#.

Contents

List of Figures

1 Introduction

The new technologies development in the area of human-computer interaction are very encouraging. These developments will help in integrating technological solution in everyday live. Progress in human computer interaction field has also opened a wide opportunity for researchers and scholars to develop new technology. These concepts are affecting the fields of human-computer interaction. Research on human machine interaction is very important in improving the ways for human to interact with the machine. Current technology trends are towards interacting with machines using gestures. Users might no longer need to use keyboard and mouse as input devices. Tools such as Microsoft Kinect, Wii Remote, PlayStation Move and PlayStation Eye have opened a new chapter in this field.

This project attempts to leverage the physical gesture detection capabilities of Microsoft Windows Kinect. Kinect's Sensor is able to uniquely and accurately detect objects using ColorFrameReader, BodyFrameReader, InfraredFrameReader and DepthFrameReader. It is able to track different bodies at once and 56 unique joints in the body, including those at the fingertips. The gesture detection is accurate and gestures have been trained using a wide variety of machine learning algorithms, mainly random forest regressor. Because of the huge advantages of Kinect, it has been used to develop a Picture Viewing Application with hands-free interaction.

The application itself has been built using Windows Presentation Framework(WPF) for the GUI and Visual Studio for development. It uses a wide variety of gestures for providing features like move to next/previous image, zoom in/out, panning through the image, changing backgroung, browsing different folders and finally quitting the application. The frames indicating the current active positions of the various joints in the body are obtained continuously at a very high fps, so there is no chance of missing a gesture of any kind.

Further, the application also provides the Kinect User Viewer and the Kinect Hand Cursor to make the use of the application even more intuitive.

2 Related Work

There has been a lot of research work in the area of hand gesture detection based image viewer.

- **Hand Detecting and Positioning Based on Depth Image of Kinect Sensor: Van Bang Le, Anh Tu Nguyen, and Yu Zhu.**
  - This paper[1] presents a method of hand segmentation based on Kinect sensor depth image. When the Kinect infrared camera output imaging is 1 channel video, the gray level of images and the distance between body-camera have a proportional relationship. With this feature, this article is under OPENCV-PYTHON programming environment.
  - This paper puts forward a palm center detected based on contours moment.
  - The proposed method has many advantages such as: high precision, high stability, small computation time and the influence of the key points is very small.
  - The problem is that segmentation of the hand isnt always accurate. Also, 3-D gestures cannot be mapped.
- **TouchLight: An Imaging Touch Screen and Display for Gesture-Based Interaction: Andrew D. Wilson.**
  - TouchLight[2] uses two cameras in combination with a commercially available projection screen technology which allows projection onto an otherwise transparent surface. This arrangement allows for certain novel applications and flexibility which go beyond previous related technologies.
  - Image processing techniques have been presented to produce a touch image useful for many gesture-based and perceptual computing scenarios.
  - It's adavantage is that it is very efficient and intuitive to use. It is a technology of the future and can have a wide variety of applications.
  - As of now, the resources used to develop it are expensive, hence it is not economically feasible. Also, it is difficult to set up.
- **Gesture Based Interface Using Motion and Image Comparison: Shany Jophin, Sheethal M.S, Priya Philip, T M Bhruguram.**
  - This system[3] is based on image comparison and motion detection technology to do mouse pointer movements and selection of icon.
  - It is simple and easy to use.
  - Though, it is difficult to get stable results because of the variety of lighting and detection of the same color in any other location in the background. Most algorithms used have illumination issues.

  - The performance of the software can be only improved by small percentage due to the lack of a powerful camera and a separate processor for this application.

- **Control Windows Photo Viewer by Hand Gesture Recognition using MATLAB: Manoj Makhija, Kapil Goyal.**
  - Shape based approach[4] is proposed for hand gesture recognition to run multiple applications with just movement of hand. This paper first pre-processed the image and then find centroid and check where the centroid lies.
  - On the basis of the centroid gesture is recognized and according to the gesture function of the photo viewer application are to be controlled by gesture recognition.
  - It is simple, quick and effective to use.
  - However, only a minimal number of gestures can be performed as gestures are primarily dependent on thumb movements. Also, the camera used may fail to capture gestures if gestures take place at a much faster frame rate.

3 Proposed Methodology and Framework

3.1 Flowchart

Microsoft's Kinect v2 SDK has been used to detect and track gestures. The application has been built using Windows Presentation Framework(WPF) and the coding has been done in Visual Studio. The workflow of the application is richly described the flowchart below:

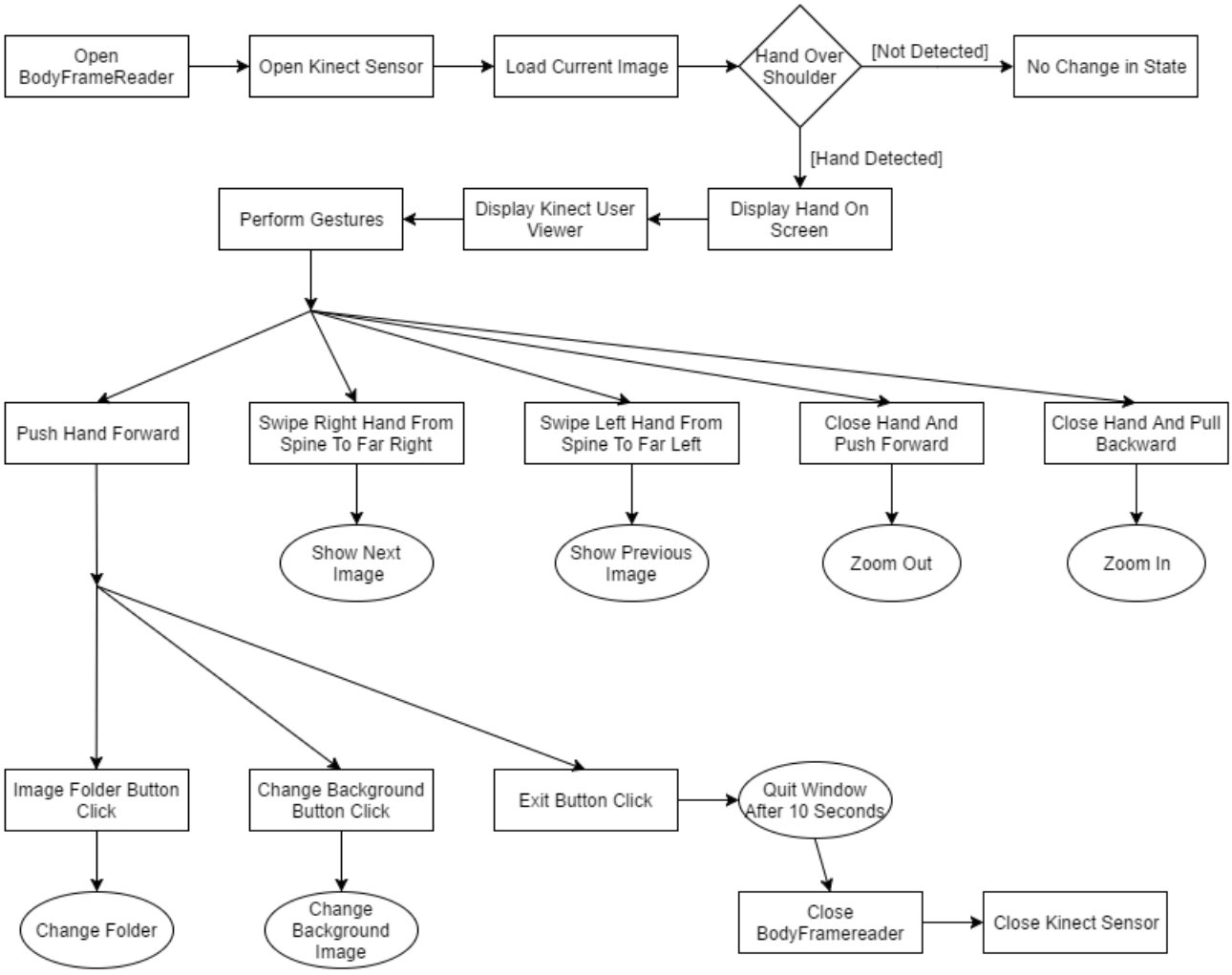

Figure 1: Workflow of the Application

3.2 Open BodyFrameReader Source

Represents a reader for body frames. The BodyFrameSource property gets the source of the body frames and the acquireLatestFrame() is used to get the most recent frame. The frames are obtained at a rate more than 30 fps. These frames can be used to track the various joint positions.

3.3 Hand Over Shoulder Engagement

Our application requires hand to be placed over the shoulder level for the first engagement of the Kinect Cursor. This displays the hand cursor on the screen and the kinect user view is activated.

3.4 Detection of Gestures

Various gestures have been implemented in our application for the following functionalities:

1. **View Next/Previous Image.** This is done using right/left hand swipes starting from the middle spine position to the right/left sides for moving to next/previous image respectively. The left swipe, for eg, has been implemented in the following way:

```
private bool leftSwipe(Body body, JointType jointType)
{
    var shoulderSpine = body.Joints[JointType.SpineShoulder];
    var hand = body.Joints[jointType];
    var leftShoulder = body.Joints[JointType.ShoulderLeft];

    bool isDetected = hand.Position.Y > shoulderSpine.Position.Y && (leftShoulder.Position.X - hand.Position.X) > 0.3;

    if (isDetected == true)
    {
        leftBoolCounter += 1;
        if (leftBoolCounter > 1)
        {
            return false;
        }
        else
            return true;
    }
    else {
        if (shoulderSpine.Position.Y > hand.Position.Y || hand.Position.X > shoulderSpine.Position.X)
        {
            leftBoolCounter = 0;
        }
        return false;
    }
}
```

Figure 2: Left Swipe Implementation

2. **Zoom In/Out.** The gesture to be performed for this needs the fist to be closed on the image view. Once the fist is closed, i.e., the image has been grabbed, move the fist forward(+z coordinate) to zoom out and backward(-z coordinate) to zoom in.

3. **Panning The Image.** Panning means to move around the image. If the image size is greater than the image holder, then panning can be done by grabbing the image, i.e., closing the fist and moving it in the vertical plane.

4. **Button Clicks.** Various buttons exist for providing options like changing background, browsing through different folders and to exit the application. For this, the gesture used is pushing on the button, i.e., moving the open palm in the +z coordinate direction. The hand cursor also gets highlighted for a while to indicate the button is being pressed.

3.5 Screenshots

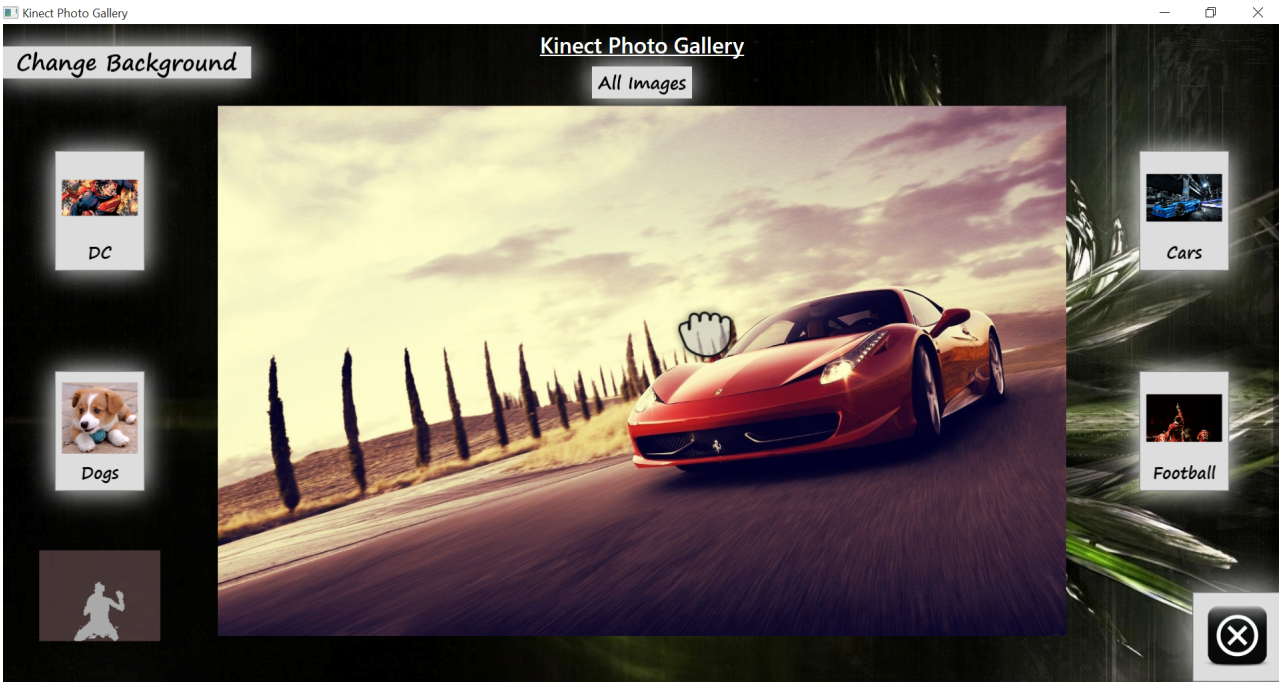

Figure 3: Grabbing the Image

4 Results and Discussion

4.1 Accuracy

- Bodyframereader frames obtained via Kinect Senor can track a huge number of joints(namely 56) in the torso, right arm, left arm, right leg and left leg. These are very precise to begin with and reflect the exact positions in the three dimensional space of the body to be tracked.

- The gestures are mapped according to the joint positions and their movements in the entire 3-D space. This makes them extremely precise and reflect perfect fucntionality.

- The hand cursor of KinectRegion is very accurate in detecting open hand and fist. It is already implemented in Microsoft Kinect v2 SDK and has been rigourously trained using a variety of machine learning algorithms, primarily random forest regressor.

4.2 Ease of use

- The gestures used for various functionalities of a Kinect Photo Gallery are very intuitive. Consider the swipe gestures for example. Left swipe, i.e., moving to the previous image is done by hand whereas the right swipe is done by right hand. Pulling the image towards oneself implies zooming the image in since we are getting closer to the image. Similarly, pushing the image away implies zooming out the image, i.e., moving away from the image. Button clicks are simulated by performing the pushing action.

- Also, the kinect user viewer gives a shadow image of the user on the screen. Therefore, the user is always aware what actions he/she is performing. This also helps in performing gestures.

- The hand cursor used is part of the KinectRegion(acts as a container and coordinator for other KinectInteraction Controls), which simulates the open hand and fist. This gives the user an idea about what state the mouse cursor is in.

4.3 Performance

- The current, previous and the next image are already loaded when a new image is loaded. The previous and the next image is hidden while only the current image is shown. Hence, the time taken to move to a new image is very less.

- There is no tearing observed while zooming into and zooming out an image. Moreover, the time between performing a gesture and the resultant action on the WPF window is almost negligible. Hence, it allows for quick interaction.

- No heavy image processing is done as the body frames are continuously read using BodyFrameReader Class of WindowsPreview.Kinect and hence the joints are continuously tracked. Therefore, the position of all body joints is known at all times and no computation is involved. Hence, it is very fast and efficient.

5 Future Work

The Kinect Photo Gallery can be further enriched in functionalities by adding gestures for cropping, controlling brightness, saving the image etc. Speech recognition can also be integrated into the application to further simplify the interaction between the user and the application. Moreover, the gallery app can be generalised to include images from the entire file directory with appropriate sorting features.

References

[1] Van Bang Le, Anh Tu Nguyen, Yu Zhu: Hand Detecting and Positioning Based on Depth Image of Kinect Sensor. In International Journal of Information and Electronics Engineering, Vol. 4, No. 3, May 2014.

[2] Andrew D. Wilson: TouchLight: An Imaging Touch Screen and Display for Gesture-Based Interaction. In Microsoft Research, 2014.

[3] Shany Jophin, Sheethal M.S, Priya Philip, T M Bhruguram: Gesture Based Interface Using Motion and Image Comparison. In International Journal of Advanced Information Technology (IJAIT) Vol. 2, No.3, June 2012.

[4] Manoj Makhija, Kapil Goyal: Control Windows Photo Viewer by Hand Gesture Recognition using MATLAB. In International Journal of Advance Engineering and Research Development(IJAERD), Volume 1, Issue 8, August 2014.