PicassoNect - Kinect based Painter

By Maor Dickman, Shay Ziv & Nimrod Raifer

Introduction

Kinect is a line of motion sensing input devices by Microsoft for Xbox 360 and Xbox One video game consoles and Windows PCs. Based around a webcam-style add-on peripheral, it enables users to control and interact with their console/computer without the need for a game controller, through a natural user interface using gestures and spoken commands. Kinect sensor is based on Prime-Sense technology and there are similar sensors on the market such as Asus Xtion and PrimeSense Carmine.
(http://en.wikipedia.org/wiki/Kinect)

Windows Presentation Foundation (or **WPF**) is a graphical subsystem for rendering user interfaces in Windows-based applications by Microsoft. WPF attempts to provide a consistent programming model for building applications and separates the user interface from business logic. It resembles similar XML-oriented object models, such as those implemented in XUL and SVG.
(http://en.wikipedia.org/wiki/Windows_Presentation_Foundation)

Overview

In the project we developed a WPF application which allows the user to create a painting using his hands with minimal interactions with the mouse and keyboard.
Using the Kinect, the application follows the user's hands movements, and translates them to a brush strokes. The application allows the user to select a brush from several rendering methods.

Application requirements

- Capture user movements using Kinect sensor
- Paint the desired brush according to the user hands movements
- Enable several rendering options to choose from
- Option to save and send via email the finished painting

Project Infrastructure

The project is A WPF application, based on Microsoft Kinect SDK 1.7.
It is written in C#, under Visual Studio 2012, and relies on the following libraries:

1. **Microsoft.Kinect.Toolkit** - This is the Microsoft Kinect toolkit library which provides the means to connect to a Kinect sensor, and access to the 3 main streams a Kinect sensor provides:
   • Color Stream
   • Skeleton Stream

2. **WriteableBitmapEx** - This is an open source collection of extension methods for the WriteableBitmap class. The WriteableBitmap class is available for Windows Phone, WPF, WinRT Windows Store XAML and Silverlight and allows the direct manipulation of a bitmap and could be used to generate fast procedural images by drawing diresctly to a bitmap. (http://writeablebitmapex.codeplex.com)

Application structure

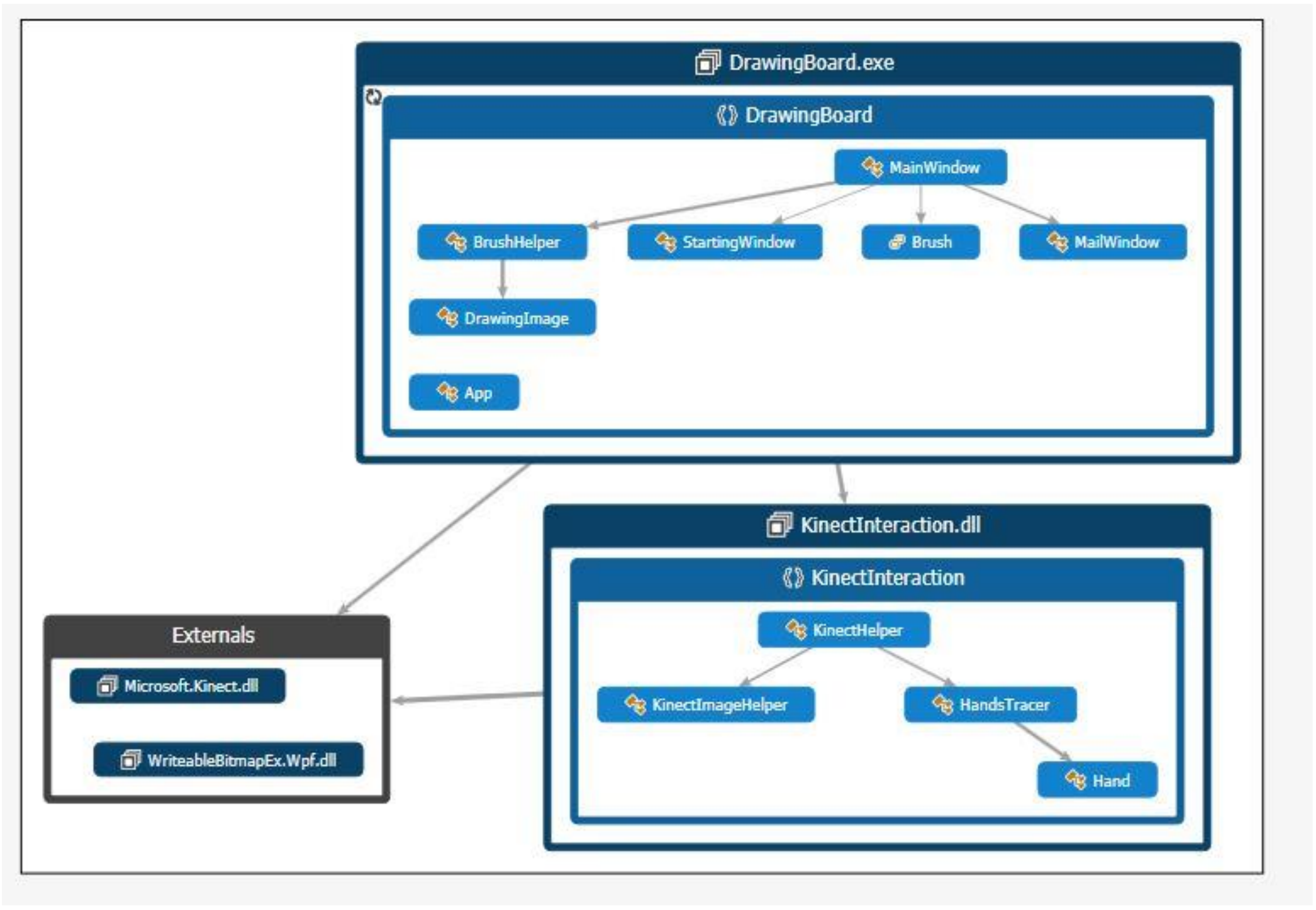


Figure 1 – application structure layout

Project Main Packages

The application is designed from two main packages:

1. Kinect interaction

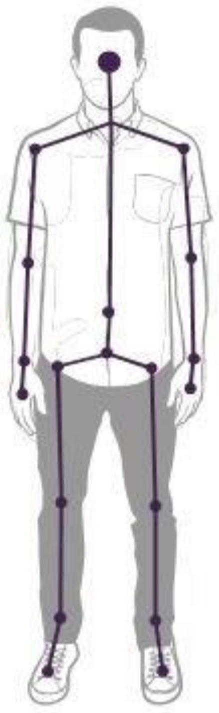


Figure 2 - skeleton

In this package we implemented the needed functions for the application to interact with the Kinect.
This package contains the following classes:

1.1 KinectHelper

The main functionalities of this Class are:

- Initializes the connection with the Kinect, looking for active Kinect sensor (if one exist)
- Initiate HandTracer instance (will be explained later)
- Initiate KinectImageHelper instance (will be explained later)
- Enables the skeleton stream and connect it to the HandTracer.
- Enables the color stream and connect it to the KinectImageHelper.

1.2 HandTracer

This Class main purpose is to track the hands locations. For this purpose this class uses the Kinect skeleton stream that return the user skeleton formation as seen in figure 2.
The main functionalities of this Class are:

- Exposing two instances of Hand class, right hand and left hand.
- Exposing HandLocationChanged event. This event is invoked when a new skeleton formation is received from the Kinect.
- Updating the hands location according to the skeleton formation.

1.3 Hand

This Class represents the user's hands and exposes the following properties:

- State – the tracking state of the hand. The state can be: tracked, inferred and not tracked.
- Position – the three dimensional position of the hand.
- oldPosition – the previous three dimensional position of the hand.

1.4 KinectImageHelper

This class exposes the image captured by the Kinect and exposes the following properties:

- Image – a byte array containing the image.
- FrameWidth – the width of the image in pixels.
- FrameHeight – the height of the image in pixels.
- kinectImageChanged event – This event is invoked when a new color frame is received from the Kinect.

2. Drawing board

In this package we implemented the application GUI, the drawing logic and other applications functions.
This package contains the following classes:

2.1 main window

This class is the main class of the program. It contains all the GUI definitions and logic in order to draw the paint on the screen.
This class holds the following main components:

- ImageSource – this object is of the type WriteableBitmap. This object holds the painted image.
- KinectHelper – used to interact with the Kinect (as explained above)
- Brush – an Enum containing the used brush.
- Color – the current used color. The color is changed at random.

This class uses the following functions:

- DrawHand - draw on the screen the selected brush according to the user hand position.
- PointToScreen – convert from kinect resolution to screen resolution.
- LoadImage – open file dialog which enables the user to choose a new pattern for the fifth brush.
- SendMail – send the image to the user's mail.

2.2 brushHelper

This class exposes functionality to draw on the screen through the WriteableBitmap.
This class holds the following main functions:

- AirBrush – draw the air brush rendering techneqe.
- Ellipse –
- DrawImage – draw an image according to a randomly pattern
- DrawBackImage – draw a circle of a random pattern
- MovingBrush – draw circles according to the user movement.
- WriteAlphaBlended – blend the old and new color in the bitmap by the new color alpha (transparency) value.

2.3 mail window

This class is window for asking the user for his email address

2.4 drawing image

This class is used to read an image from file and makes it a writeable bitmap

2.5 starting window

This class is the opening window of the project, which contains the logo of the project.

Project Brushes

In this project we implemented 4 brushes. The user can switch between the brushes and to choose different color for them at any time. The brushes are:

1. Spray

The "Spray" brush, much like the spray brush used in "Microsoft paint", draw pixels on the screen in a small area surrounding the position of the hand. In this brush, the pixels color will be thicker at the center of the pointing area, and thinner when we move away from it.

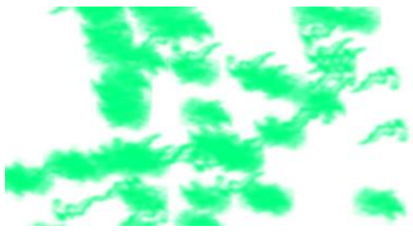
2. Ellipses

The "Ellipses" brush draws ellipses on the screen, Depending on the speed of the movement of the hands. A high speed level of hand movement - will draw large radios Ellipses.

3. Circles

The "Circles" brush draws circles on the screen, Depending on the speed of the movement of the hands. A high speed level of hand movement - will draw circles with large spaces between them.

4. Patterns



The "Patterns" brush takes a pattern from a picture and draws it on the screen. This brush can be resizable by the "Resize" scroll bar under it.