Introduction to Interactive Space SDK

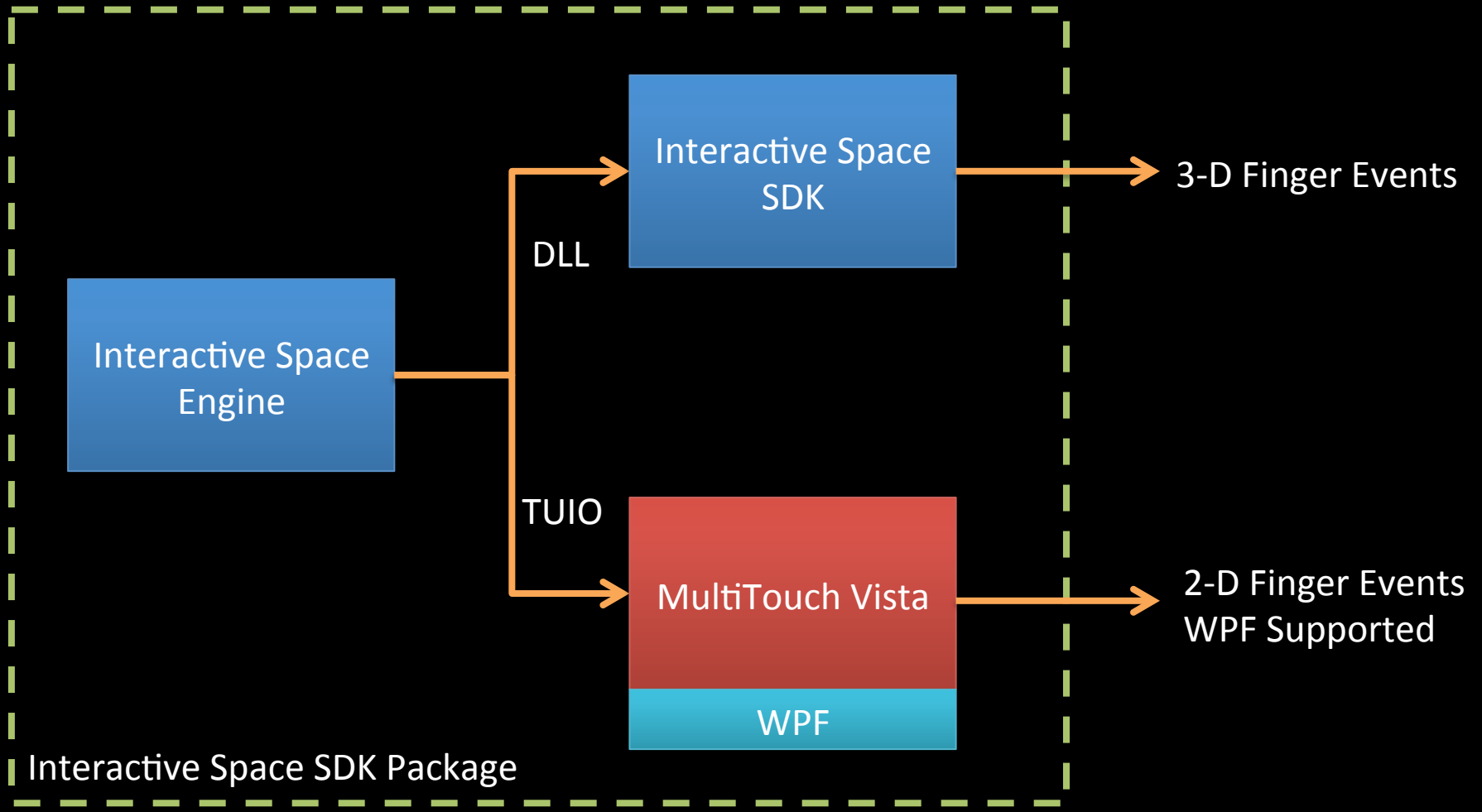
Yang Liu

Features

- Finger Tracking
  - Multi-touch
  - 3-D coordinates
- WPF Integrated
  - via MultiTouch Vista
- Full-screen app on projector
- Hand Tracking (Coming soon)

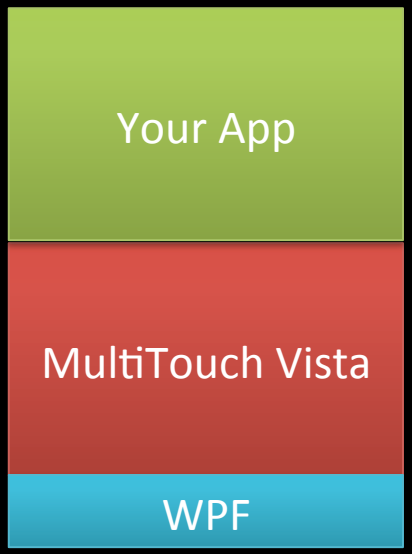
Sample App: Risk Board Game

System Framework



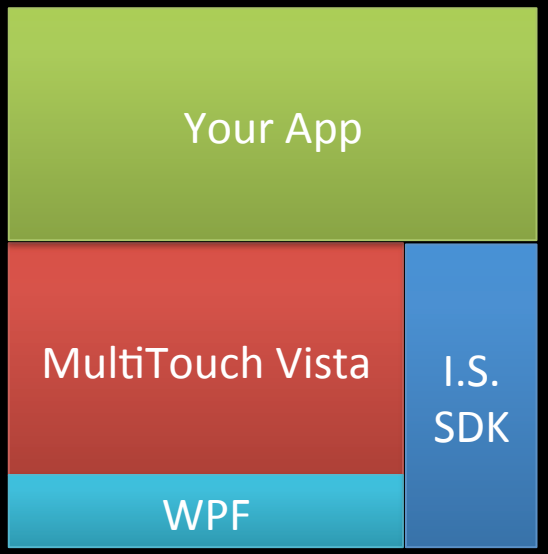
You app could be…

- In C# and WPF.
- Based on MultiTouch Vista.
- Listen to multi-touch events as easy as mouse-click events.
- No 3-D coordinates.



You app could be…

- In C# and WPF.
- Based on MultiTouch Vista and Interactive Space SDK.
- WPF integrated 2-D finger events.
- 3-D finger events.

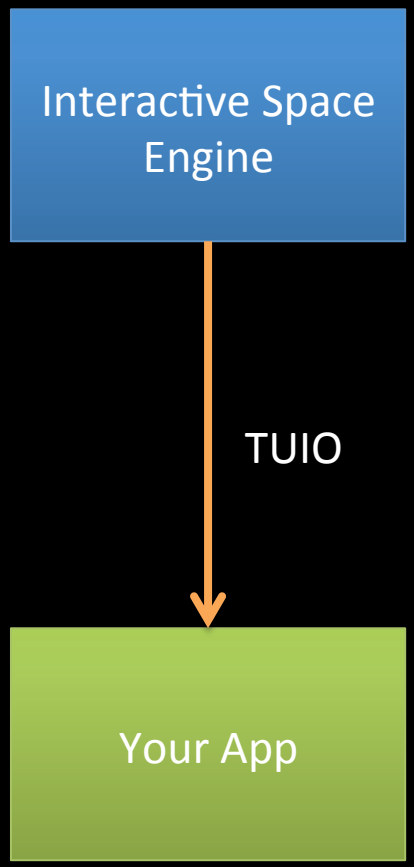


You app could be…

- In C#.
- WPF, Windows Forms, XNA, whatever…
- Based on Interactive Space SDK.
- 3-D finger events.
- You have to implement your own GUI logic. (e.g. Which object is being touched.)
- A little bit crazy.

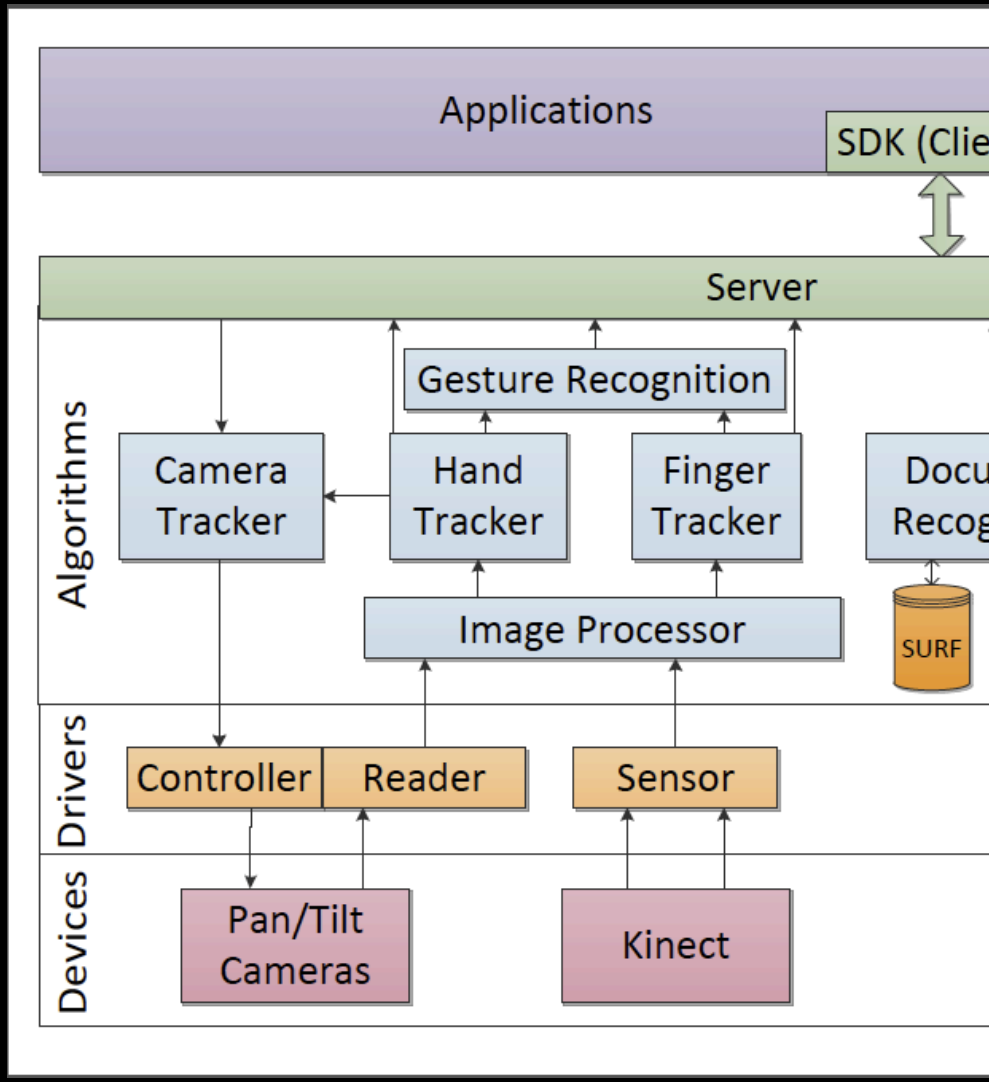
You app could be…

- Any language. Any GUI framework.
- Find TUIO client implementation for the language you choose.
- No 3-D coordinates.
- This is crazy.



“Can I get more crazy?”

- “TUIO with 3-D coordinates?”
- “Your algorithm sucks!”
- “Machine learning, please!”
- Talk to Nadir and Yang.
- We have source code, native (C++) core, numerous issues…

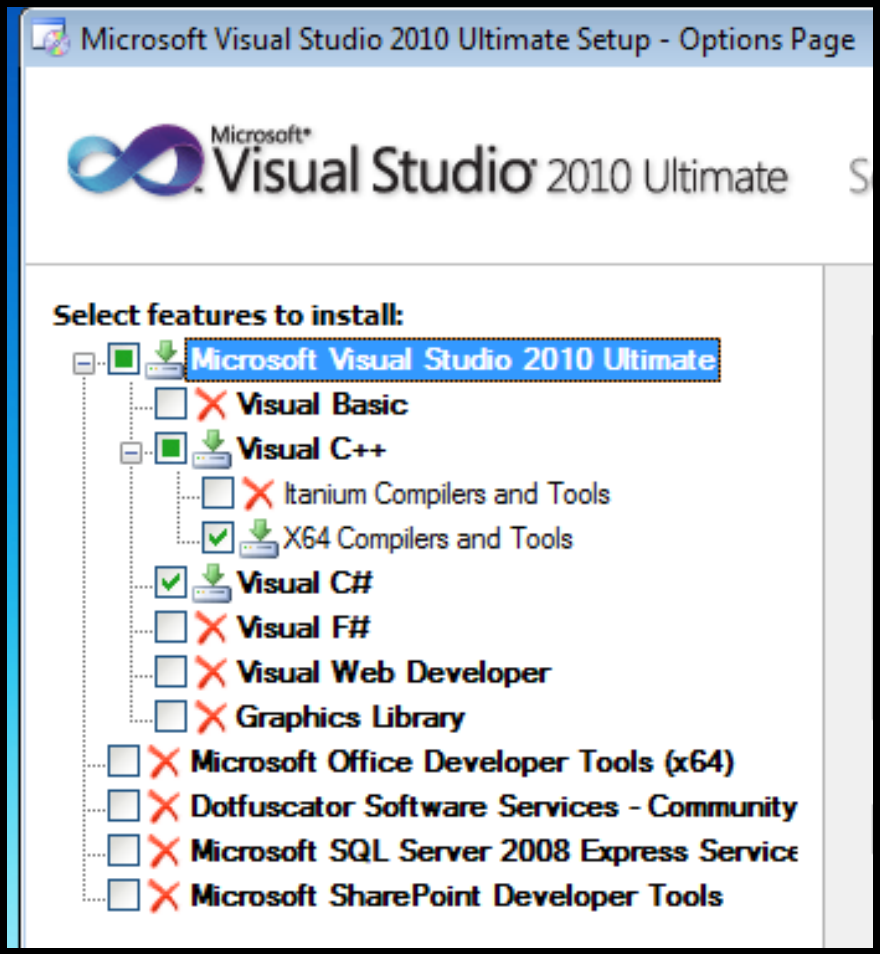


System Requirement

- Windows vista/7 **64bit**
- Visual Studio 2010 Professional

If no Visual Studio…

- Go to Microsoft DreamSpark
  - https://www.dreamspark.com/Student
- Create an account and get verified with your UCSD email address.
- Download Visual Studio 2010 Professional from “software catalog”.



Required Visual Studio Components

Don’t worry about “Ultimate”. Professional works.

Notes About Visual Studio

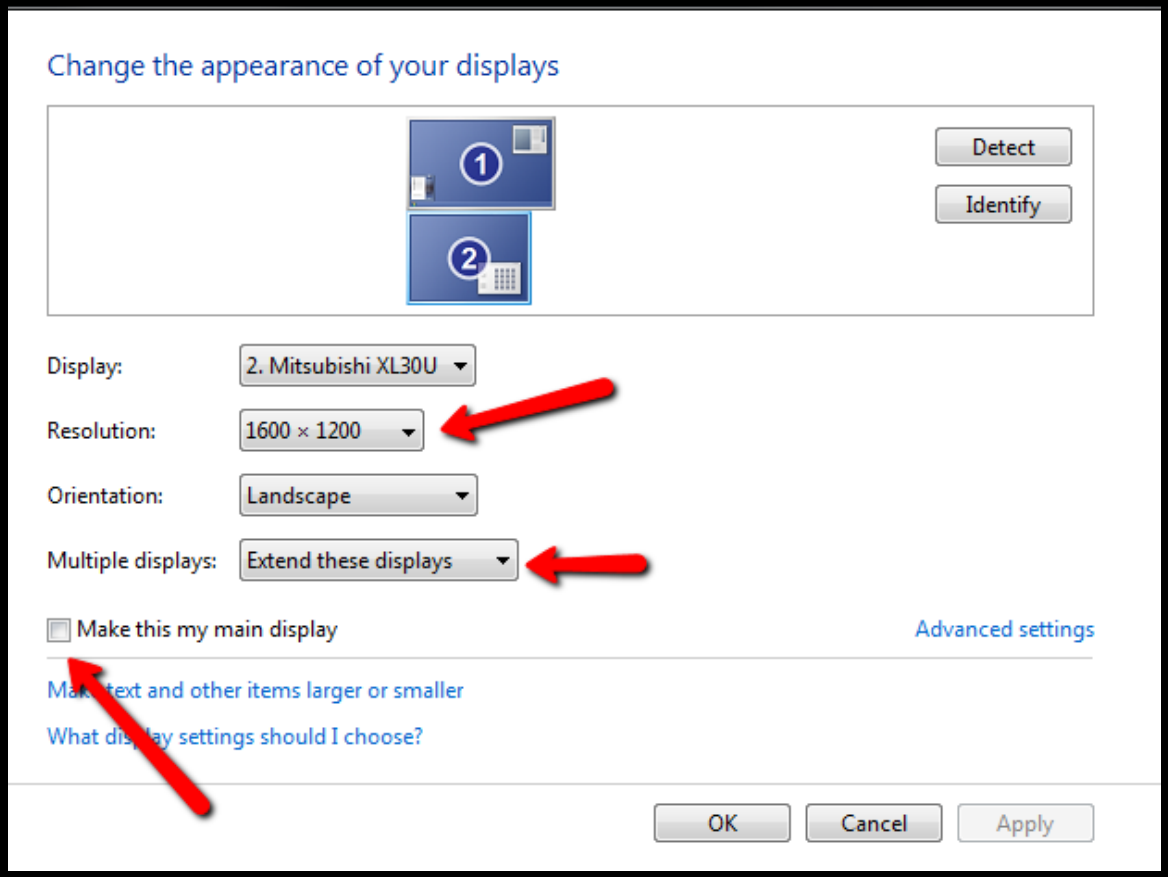
- Probably works on Visual C# Express 2010.
  - But lack of fancy debugging tools.
- Not tested on Visual Studio 2012.
  - You might try it though.

If no Windows 7 x64…

- Solution 1: Get and install it.
  - Free Windows / VMware for CSE students:
    - http://www.cse.ucsd.edu/node/1972
    - (Send request to CSEHelp ASAP!)
  - PC Users: Upgrade or VMware Workstation.
  - Mac Users: Bootcamp or VMware Fusion.
- Solution 2: Build a 32bit SDK for us.
  - Need to build C++ code, OpenCV and Boost library.
  - Talk to Nadir and Yang about this.

Projector Configuration

1. Connect your computer to the projector.
2. In Windows “Screen resolution”, set “Extend these displays”.
3. The projector resolution must be 1600x1200.
4. Do not make the projector as main display.
5. Refer to the live demo in class.



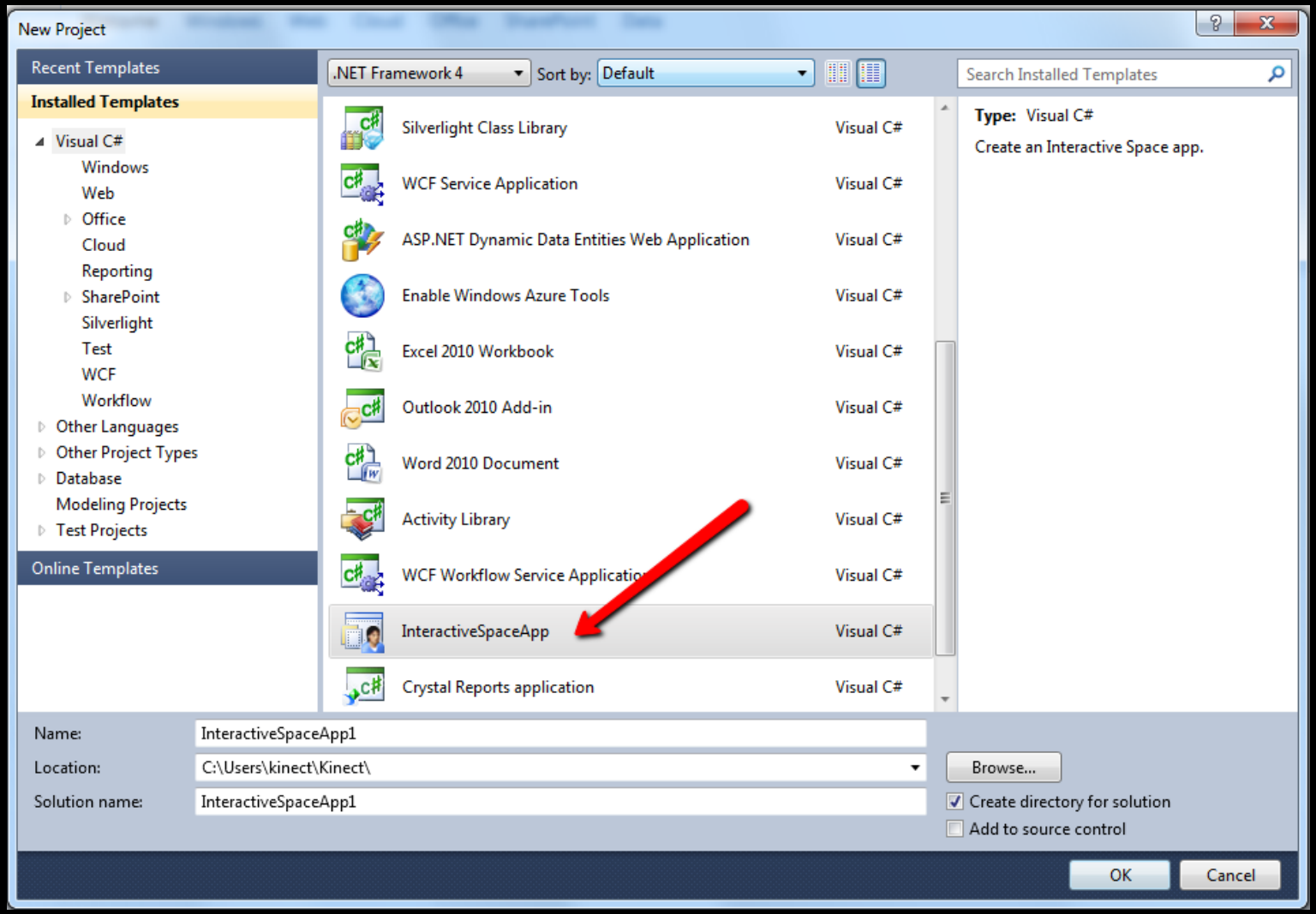
Projector Configuration

SDK Configuration

1. Install Kinect for Windows SDK and Kinect for Windows Developer Toolkit
   - http://www.microsoft.com/en-us/kinectforwindows/develop/developer-downloads.aspx
2. Download latest InteractiveSpaceTemplate
   - https://github.com/DCog-HCI-UCSD/InteractiveSpaceEngine/downloads
3. Move InteractiveSpaceTemplate_(ver).zip to
   - %HOMEPATH%\Documents\Visual Studio 2010\Templates
   - (Do not extract the zip)

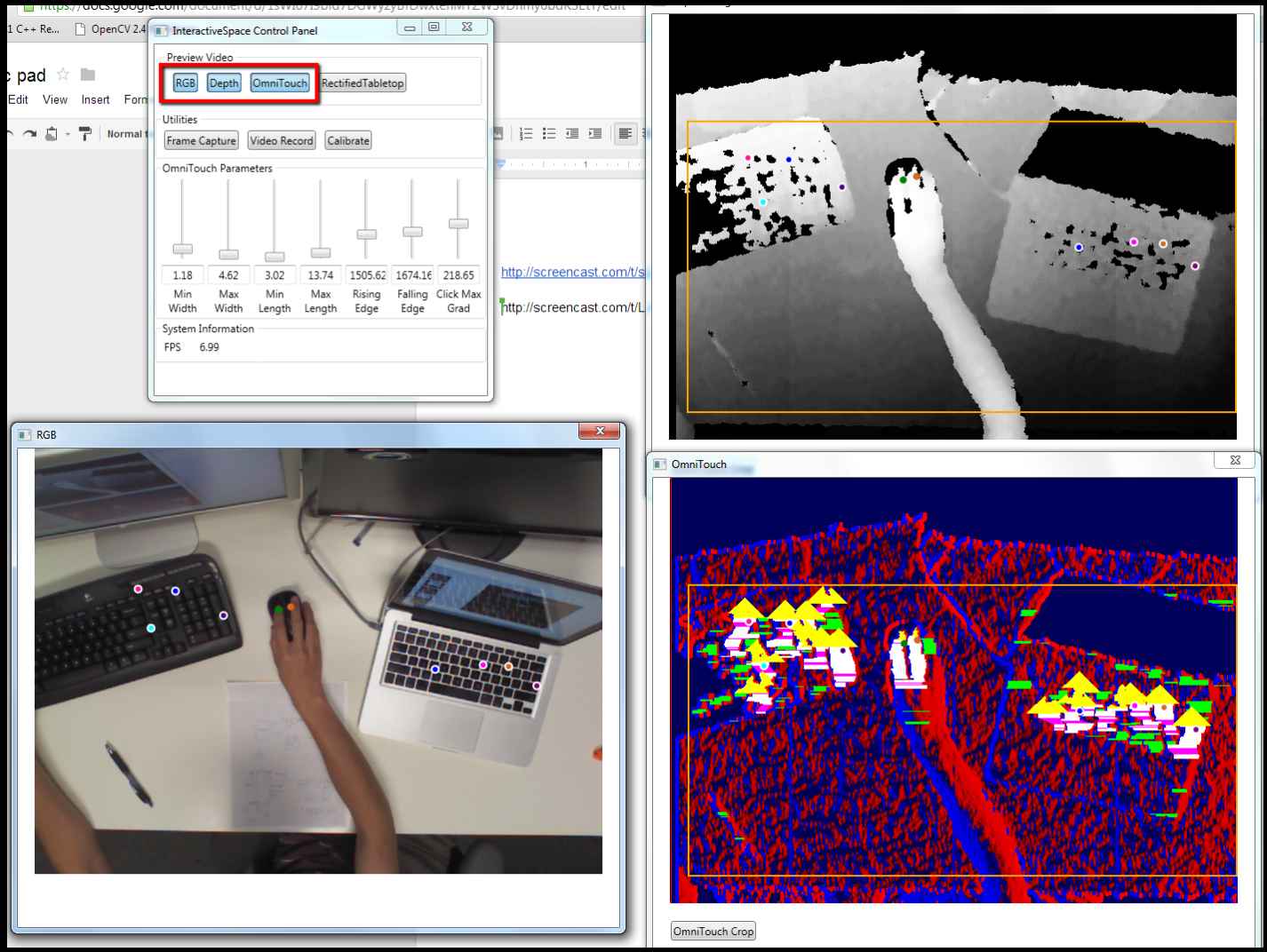
Your First App

- Connect your projector and Kinect.
- Create an “InteractiveSpaceApp” project.
- Build and run it.
- Good luck.



Your First App

Create “InteractiveSpaceApp” project.

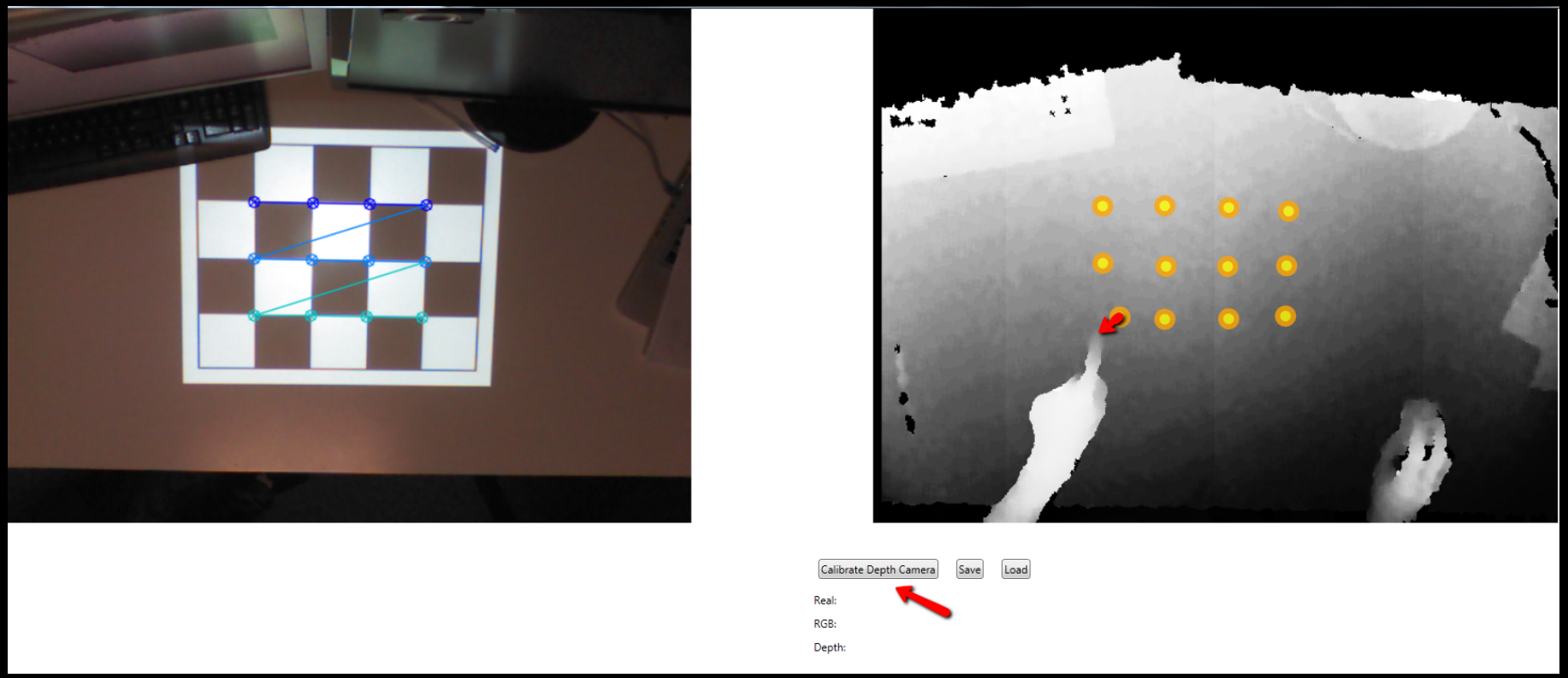


Your First App

Connect your Kinect. Build and run. If you're so lucky…

Calibration

- You need to calibrate the system before playing with your new multi-touch table.
  1. Click “Calibrate”
  2. After the chessboard is detected in the left video, drag yellow marks to the chessboard corners in the right video. Use your finger to indicate the corner.
  3. Click “Calibrate Depth Camera”.
- You don’t need to re-calibrate until you move your projector or Kinect.



Calibration

Calibrate the system before playing with your new multi-touch table.

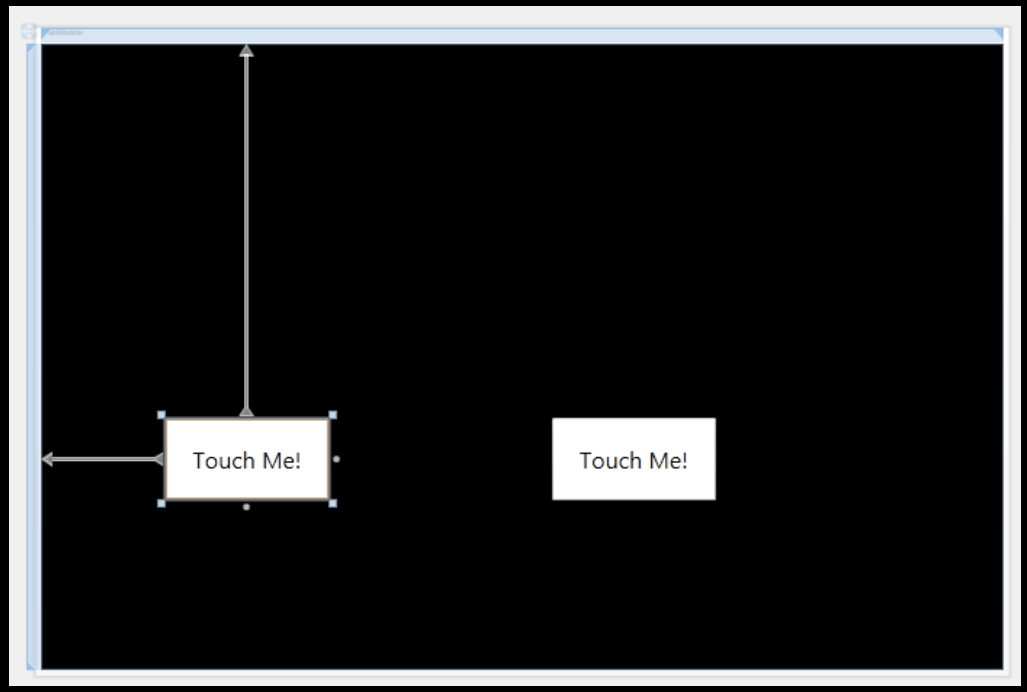
Make sure your Kinect can see the whole projected screen.

Drag yellow marks to the chessboard corners.

Crop noise area

- Our noise reduction algorithm sucks, so you need to crop noise area from finger detection.
  1. Open Depth or OmniTouch video.
  2. Click “OmniTouch Crop”.
  3. Drag your mouse to select noise-free area.

Listen to touch events




```
<mt:Window x:Class="InteractiveSpaceTemplate.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        xmlns:mt="http://schemas.multitouch.com/Multitouch/2008/04"
        xmlns:space="clr-namespace:InteractiveSpace.SDK.GUI;assembly=InteractiveSpace.SDK"
        Title="MainWindow" Height="1120" Width="1678" Closing="Window_Closing">
    <Grid Name="mainGrid">
        <space:TrackingVizLayer x:Name="vizLayer" HorizontalAlignment="Stretch" VerticalAlignment="Stretch" IsFingerVisible="True" Background="Black"/>
        <Button Name="button1" mt:MultitouchScreen.NewContact="button_NewContact" mt:MultitouchScreen.ContactRemoved="button_ContactRemoved" Content="Touch Me!"
        <Button Name="button2" mt:MultitouchScreen.NewContact="button_NewContact" mt:MultitouchScreen.ContactRemoved="button_ContactRemoved" Content="Touch Me!"
    </Grid>
</mt:Window>
```

Initialize Interactive Space

```
public partial class MainWindow : Multitouch.Framework.WPF.Controls.Window
{
    InteractiveSpaceProvider spaceProvider;

    public MainWindow()
    {
        InitializeComponent();

        MultitouchScreen.AllowNonContactEvents = true;

        spaceProvider = new InteractiveSpaceProviderDLL();
        spaceProvider.Connect();

        //Uncomment these lines to draw fingers on the projected screen
        //spaceProvider.CreateFingerTracker();
        //vizLayer.SpaceProvider = spaceProvider;
    }

    private void button_NewContact(object sender, NewContactEventArgs e)
    {
        Button b = (Button)sender;
        b.Background = Brushes.OrangeRed;
        b.Content = "Touching";
    }

    private void button_ContactRemoved(object sender, ContactEventArgs e)
    {
        Button b = (Button)sender;
        b.Background = Brushes.White;
        b.Content = "Touch Me!";
    }

    private void Window_Closing(object sender, System.ComponentModel.CancelEventArgs e)
    {
        spaceProvider.Close();
    }
```

Trouble Shooting

- No InteractiveSpaceApp in New Project
  - In VS, Check Tools -> Options -> Projects and Solutions -> General -> User project templates location. Put the SDK zip file there.
- Cannot detect chessboard when calibrating
  - Make sure your Kinect can see the whole projected screen (check RGB video).
  - Turn off your room light.

Trouble Shooting

- FPS 0.00, no preview video
  - Re-plug your Kinect.
  - Run a demo app from Developer Toolkit and exit.
  - Usually happens in virtual machines. I'm working on fixing this issue.

Trouble Shooting

- Cannot find/load InteractiveSpaceEngine.dll
  - Make sure:
    1. Working on 64-bit Windows.
    2. X64 Compilers and Tools checked when installing VS.
    3. InteractiveSpaceEngine.dll, opencv_*.dll and tbb.dll exist in the lib folder of your app project.
    4. All files in lib copied to bin/debug. (They should be automatically copied.)
  - Let me know if not working.

Helpful Links

- Risk Board Game source code
  - https://github.com/DCog-HCI-UCSD/RiskBoardGameApp
- MultiTouch Vista
  - http://multitouchvista.codeplex.com/
- Interactive Space source code
  - https://github.com/DCog-HCI-UCSD/InteractiveSpaceEngine/tree/cse118

Helpful Links

- C# Programming Guide
  - http://msdn.microsoft.com/en-US/library/vstudio/67ef8sbd
- Getting start with WPF
  - http://msdn.microsoft.com/en-us/library/ms752299.aspx
  - Note: You just need a tiny part of WPF for this class.
- google.com, stackoverflow.com

Contact me

- Interactive Space SDK is faaaaaaaar from mature. I'm working on it and you'll get updated versions.
- Let me know if you get annoyed at any SDK issues, and feel free to ask any questions.
- I'll check and answer questions on Piazza.

Yang Liu
yal060@cs.ucsd.edu