Motivation

- **Wiimote controller**
  - provides 3D UI in the home
  - 3DUI in mobile environments
- **Makes games accessible to casual users**
  - great competitive edge over Xbox 360 / PS3
- **Need to understand the device**
  - advantages and disadvantages
  - how to develop 3DUIs
The Wiimote Device

- Wiimote features
  - uses Bluetooth for communication
  - senses acceleration along 3 axes
  - optical sensor for pointing (uses sensor bar)
  - provides audio and rumble feedback
  - standard buttons and trigger
  - uses 2 AA batteries
- Supports two handed interaction
  - can use 2 Wiimotes simultaneously
- Easily expandable

Wiimote Attachments

- Nunchuk
- Steering Wheel
- Zapper
- Wii Helm
- Boxing Gloves
- Sports Pack
- Fishing Reel
The Wiimote – Coordinates

Wiimote Coordinates

Data from optical sensor
- uses sensor bar
  - 10 LED lights (5 of each side)
  - accurate up to 5 meters
- triangulation to determine depth
  - distance between two points on image sensor (variable)
  - distance between LEDs on sensor bar (fixed)
- roll (with respect to ground) angle can be calculated from angle of two image sensor points

Advantages
- provides a pointing tool
- gives approximate depth

Disadvantages
- line of sight, infrared light problems
- only constrained rotation understanding
The Wiimote – Motion Data

- Data from 3-axis accelerometer
  - senses instantaneous acceleration on device (i.e., force) along each axis
  - arbitrary units (+/- 3g)
  - always sensing gravity
    - at rest acceleration is g (upward)
    - freefall acceleration is 0
  - finding position and orientation
    - at rest – roll and pitch can be calculated easily
    - in motion – math gets more complex
    - error accumulation causes problems
    - often not needed – gestures sufficient

- Advantages
  - easily detect course motions
  - mimic many natural actions

- Disadvantages
  - ambiguity issues
  - player cheating
  - not precise (not a 6 DOF tracker)

The Wii Motion Plus

- Current Wiimote device
  - gives user a lot of useful data
  - not perfect
    - ambiguities
    - poor range
    - constrained input

- Wii Motion Plus
  - moving toward better device
  - finer control
  - uses dual axis “tuning fork” angular rate gyroscope
  - true linear motion and orientation
Visualizing Wiimote Data

- Important to see data to understand device

Programming with the Wiimote

- Connect to computer
  - does not work for every bluetooth device
- Obtain Wiimote software
  - many variations and APIs (C, C++, C#, Java, Flash)
    - Brian Peek's API (www.coding4fun.com)
      - low level API
    - Paul Varcholik’s XNA 3DUI Framework (www.bespokesoftware.org)
      - contained within larger framework
      - include gesture recognizer
- Write code and enjoy
  - heuristics
  - gesture analysis
Example Code – Bespoke XNA 3DUI Framework – I

```csharp
public GestureGame() { // constructor
    mWiimoteComponent1 = new WiimoteComponent(this, Bespoke.Common.Wiimote.PlayerIndex.One);
    Services.AddService(typeof(WiimoteComponent), mWiimoteComponent1);
    Components.Add(mWiimoteComponent1);
}

protected override void Update(GameTime gameTime) {
    UpdateWiimoteState();
    base.Update(gameTime);
}
```

Example Code – Bespoke XNA 3DUI Framework – II

```csharp
private void UpdateWiimoteState() {
    mCurrentWiimoteState = mWiimoteComponent1.Wiimote.WiimoteState;
    if (mWiimoteComponent1.WasButonPressedThisFrame(Bespoke.Common.Wiimote.Buttons.B)) {
        // Start collecting a new set of points.
        mCurrentWiimoteSamplePoints = new WiimotePointCollection();
        mHelpLabel = "Release B Button to End Gesture";
    } else if (mWiimoteComponent1.WasButonReleasedThisFrame(Bespoke.Common.Wiimote.Buttons.B) &&
        mCurrentWiimoteSamplePoints.Count > 1) {
        // Create a gesture sample out of the collected points
        mCurrentSample = new Gesture(mCurrentWiimoteSamplePoints);
        mHelpLabel = "Press B Button to Begin Gesture";
        if (mStatisticalClassifier != null) {
            ClassifiedGestureCollection classifiedGestureSet =
                mStatisticalClassifier.ClassifyGesture(mCurrentSample);
            UpdateClassifiedGestureOutput(classifiedGestureSet);
        }
    } else if (mWiimoteComponent1.IsButtonHeldDown(Bespoke.Common.Wiimote.Buttons.B)) {
        DateTime timestamp = DateTime.Now;
        if (mCurrentWiimoteSamplePoints.ContainsKey(timestamp) == false) {
            mCurrentWiimoteSamplePoints.Add(new WiimotePoint(mCurrentWiimoteState, timestamp));
        }
    }
}
```