KINECT for Windows

Creative Coding Kinect v2 Workshop
September 25, 2014

Microsoft

Computer Set-up

- Please start setting up
  - Plug in the HDD to your Mac
  - Hold down Option key at boot and select the HDD (Windows)
  - Once into Windows:
    - Go thru new computer setup (skip license entry -> we will email your key)
    - Install Visual Studio 2013 expressfor Desktop
    - Install Kinect for Windows v2 SDK
    - Install Unity - activate 30 day trial
    - Unzip Cinder and openFrameworks files
    - Reboot and connect sensor

Agenda for the Day

	Topic	Notes
09:00 – 09:30	Workshop Start and System Setup Agenda & Logistics	That’s right now ☺
09:30 – 11:00	Intro to Kinect v2	Entire platform fly thru, lots of demos
11:00 – 12:00	Using Kinect with Unity	
12:00 – 12:45	Lunch break	45 minutes
12:45 – 13:15	Kinect Common Bridge	
13:15 – 14:45	Using Kinect with Cinder	
14:45 – 15:00	Break	15 min break
15:00 – 16:30	Using Kinect with oF	
16:30 – 17:00	Hacking time	Play, finish up exercises
17:00	Finish Up	

Welcome!

- **Wifi** -> ssid: INST-INT | password: n/a
- **Twitter** -> @KinectWindows
- **Restrooms** -> Near the elevator (more on floors below)
- **You Should Have**
  - Kinect for Windows v2 Sensor & carrying bag
  - External HDD, Windows 8.1 Pro License
  - T-shirt, sticker, puzzle cube

Ben Lower | @benlower

- Sr. Program Manager and Developer

  Community Manager, Kinect for Windows
  - Responsible for global K4W dev community
  - Helps developers get help & resources they need to build apps & experiences
  - Speaks at developer events about Kinect for Windows
  - Formerly worked on Windows Phone developer marketing team

Carmine Sirignano

- Developer Support Engineer, Microsoft
  - Currently supports Kinect developers in forums, at in-person events, and by authoring developer content and sample applications
  - Prior to Kinect, supported developers working with Windows Media SDKs, Silverlight, WPF, and DirectX

Kinect Overview and Introduction

- Kinect Overview
- Kinect Sensor Overview
  - Features & Components
- SDK Download & Installation
- Kinect Data Sources
- High-level Architecture
- Quick Overview of Other Platform Tech
- Helpful Resources

Kinect Overview

Microsoft

Welcome to Kinect

- A new kind of Camera, that can see in 3D
- A traditional video camera
- A 4-element microphone array
- A powerful SDK which can:
  - Recognize that a person is in front of the sensor, and what position their body is in
  - A software controllable directional microphone, and speech recognition technology
  - And lots more

What’s New in K4W v2

- Single set of cross-platform Kinect APIs (Desktop, Store, Xbox)
- Create & publish Kinect apps to the Windows Store
  - Still have robust support for WPF & Windows desktop applications
- Kinect Unity plugin for both desktop & store apps
- Kinect Common Bridge for openFrameworks and Cinder

Usage Scenarios

RETAIL

THERAPY

HEALTHCARE

EDUCATION

MFG/TRAINING

Sensor Overview

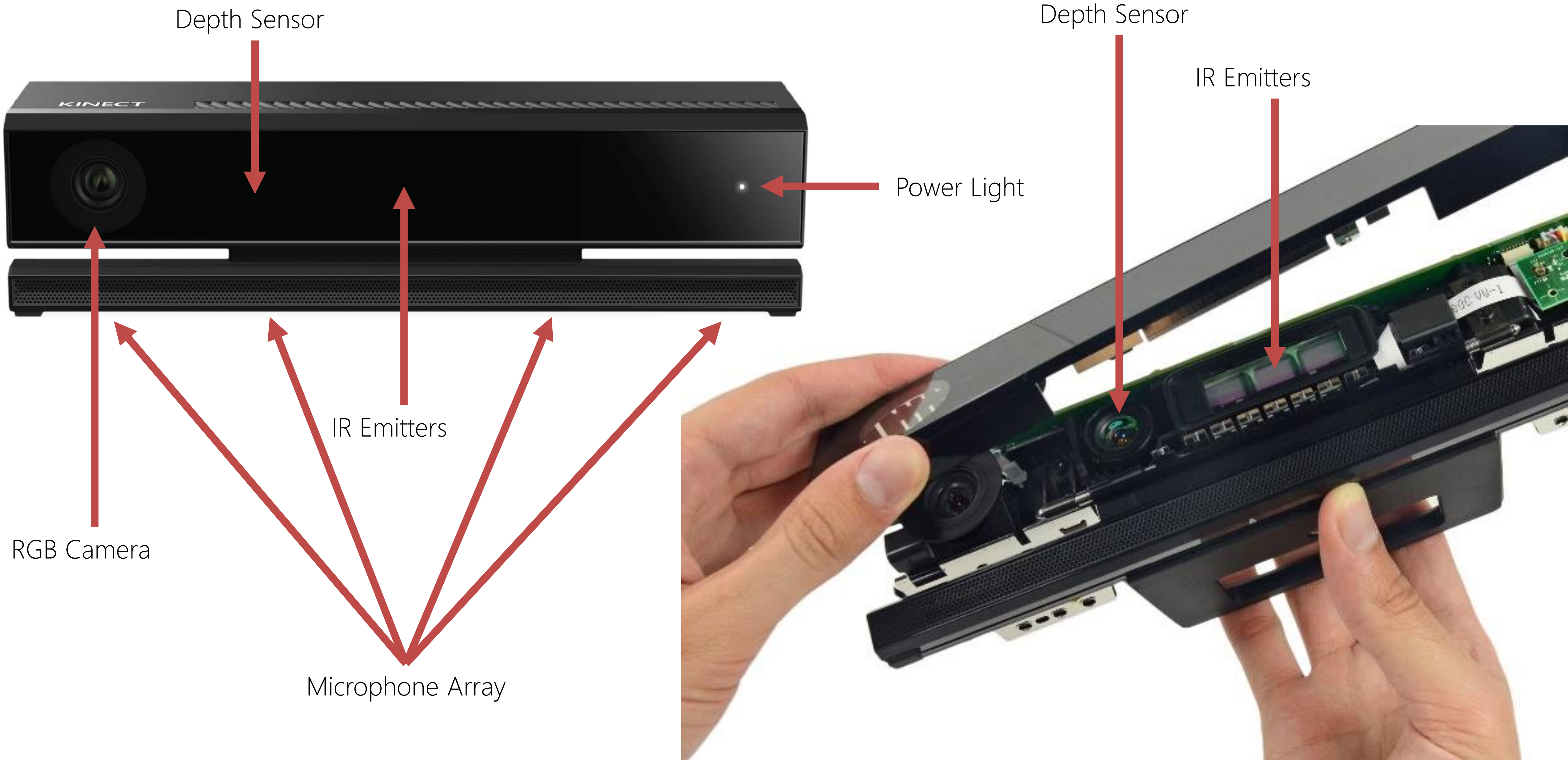
Microsoft

Sensor Features, Sensor Components

Kinect for Windows v2 Sensor Features

- **Wider field of view** (horizontally and vertically) for depth and color
- **3x depth fidelity**
- **1080p, HD color** camera (30fps)
- **Lighting independent IR** (30 fps)
- **Improved microphones** (zero balanced)
- **25 skeletal joints for six people** -> Bio correct skeletal joints (particularly hips/shoulders/spine) plus more points and rotations
- **Thumb** tracking, **end of hand** tracking, Open/Closed
- **Improved range** -> .5M near, 4.5M far (extended depth to 8M)

Sensor Components



Depth Sensor

Power Light

IR Emitters

RGB Camera

Microphone Array

Depth Sensor

IR Emitters

Image: iFixit, http://www.ifixit.com

DEMO

SDK Download & Installation (http://aka.ms/k4wv2previewsdk)
Connecting the Sensor
SDK Browser (Samples & Tools)
Installing & Running a Sample

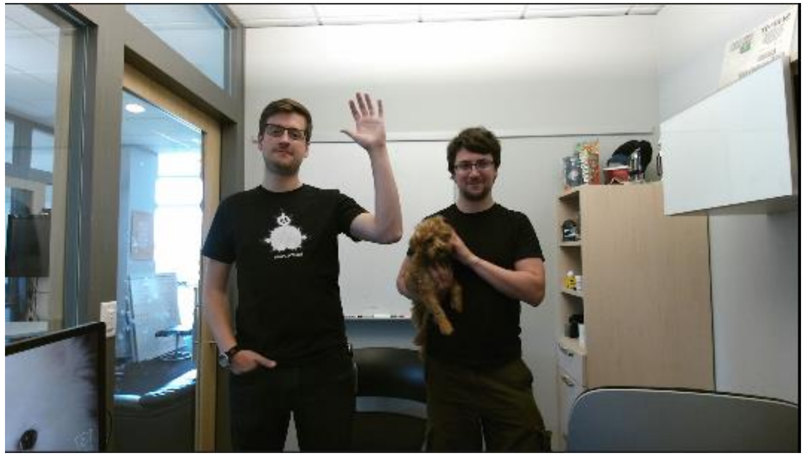
Kinect Data Sources

Microsoft

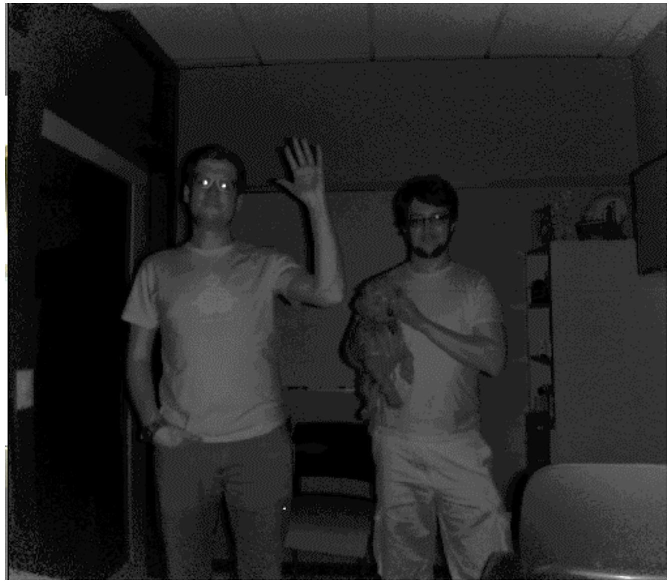
Color, Infrared, Depth, BodyIndex, Body, Audio

Kinect Data Sources

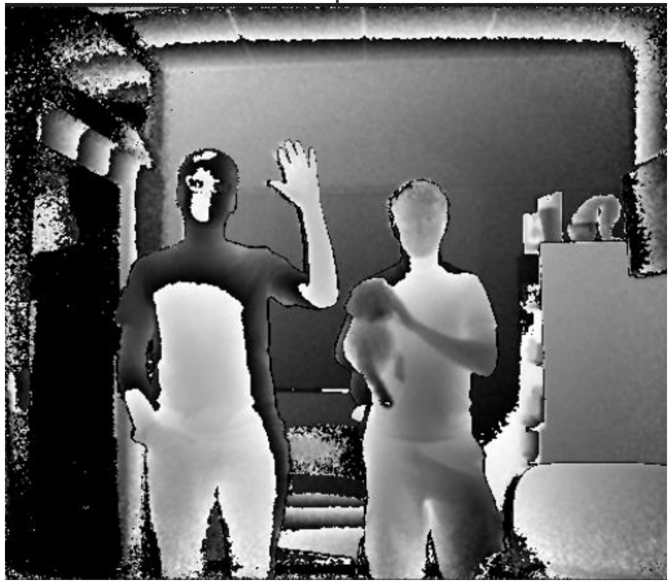
Color



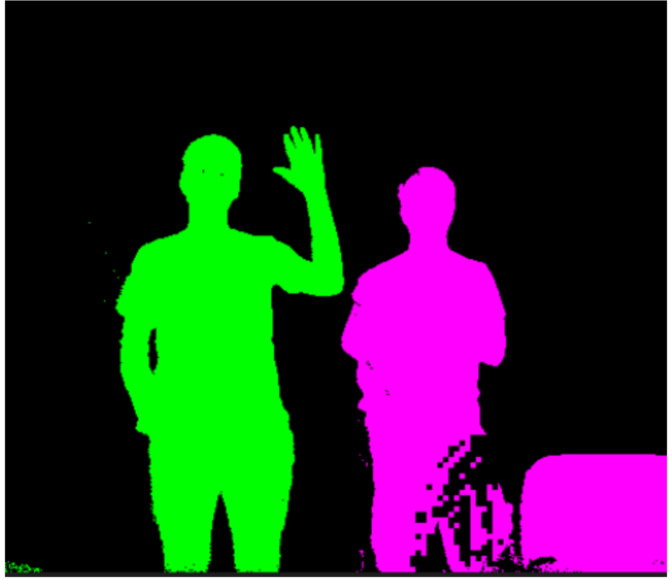
Infrared



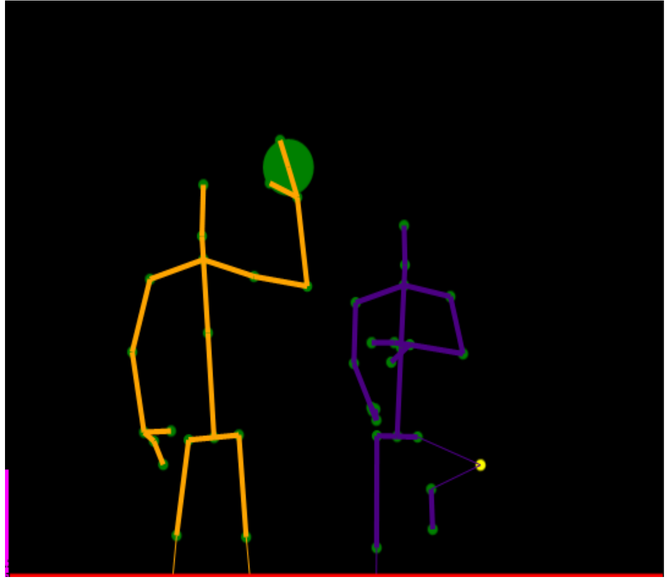
Depth



BodyIndex



Body



Audio

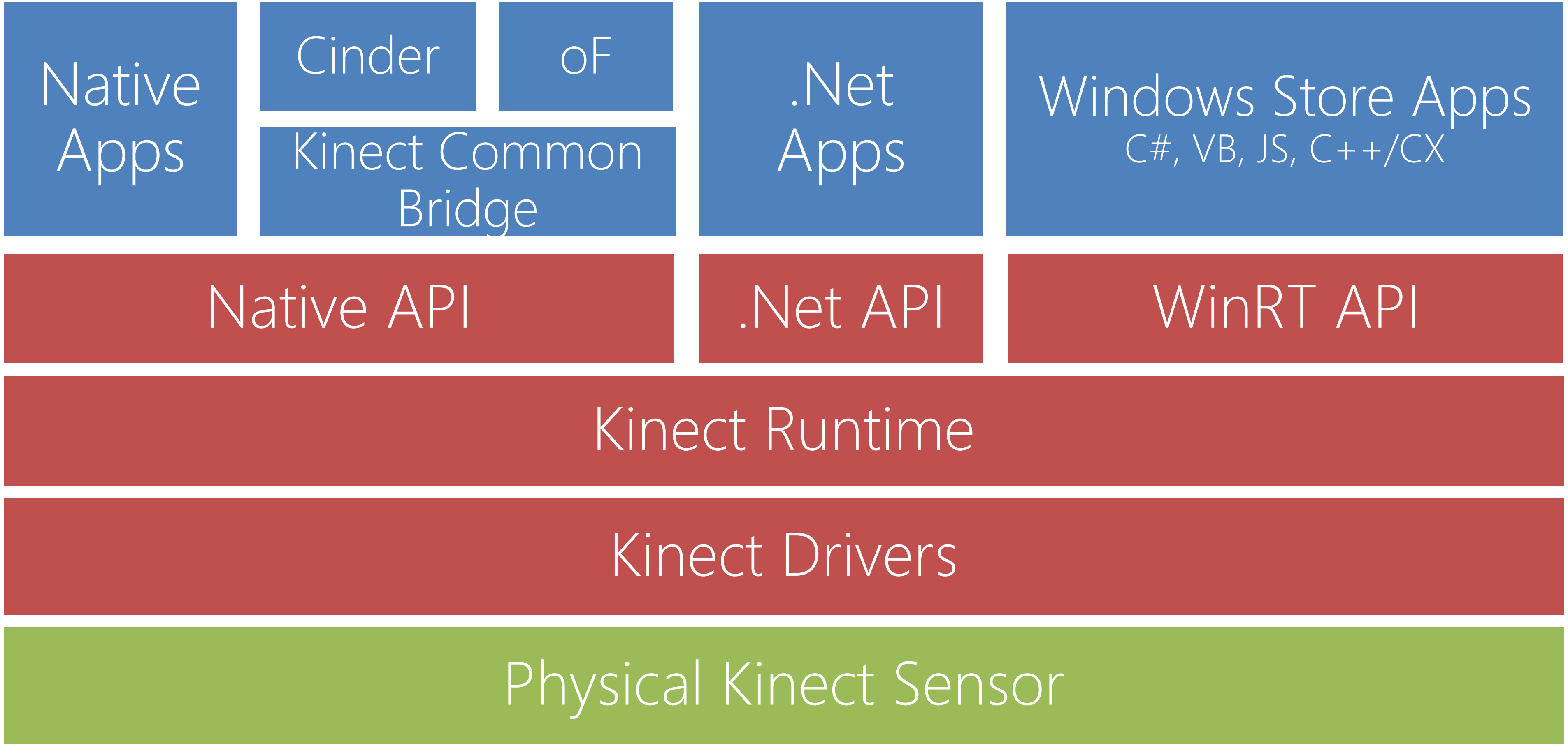
DEMO

Kinect Evolution

Architecture, HW Spec, Design Principles

Microsoft

High-level architecture



Native Apps

Cinder

oF

Kinect Common Bridge

.Net Apps

Windows Store Apps
C#, VB, JS, C++/CX

Native API

.Net API

WinRT API

Kinect Runtime

Kinect Drivers

Physical Kinect Sensor

Multiple Kinect-enabled applications can run simultaneously

Recommended Hardware

- CPU -> i7 (recommended)
- RAM -> 4GB (recommended)
- GPU -> DirectX 11 (required)
- USB 3.0 -> Intel or Renesas chipsets (required)
- OS -> Windows 8.0/8.1 (required)

Design Principles

- Minimize per-frame allocations
- Low latency – high throughput
- Expose low level data *and* high level interpretations

Accessing Kinect Data Sources

Microsoft

The KinectSensor class

- Represents a single physical sensor
- Always valid

C#

```
this.sensor =  KinectSensor.GetDefault();
this.sensor.Open();
// Make the world a better place with Kinect
this.sensor.Close();
```

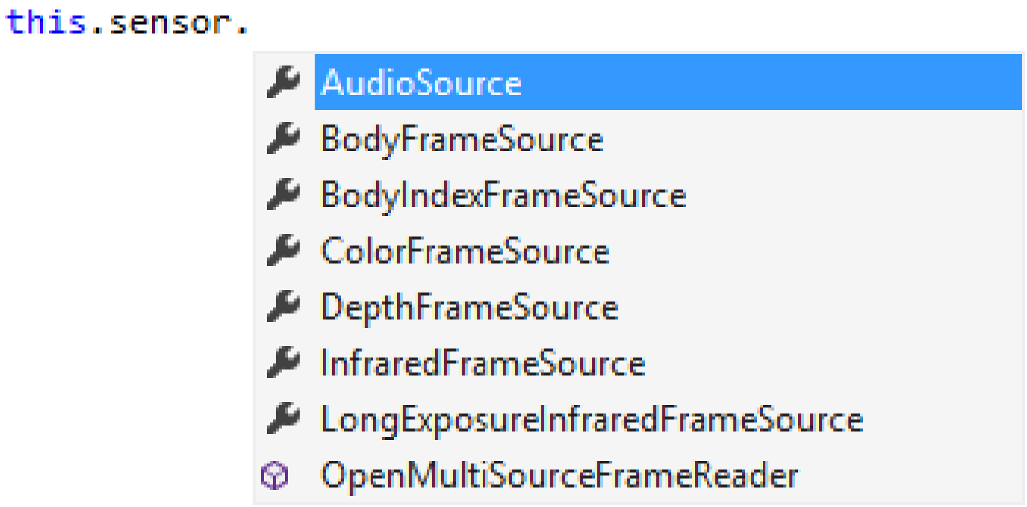
JavaScript

```
sensor = kinect.KinectSensor.getDefault();
sensor.open();
// Make the world a better place with Kinect
sensor.close();
```

Sensor | Source | Reader | Frame Ref | Frame

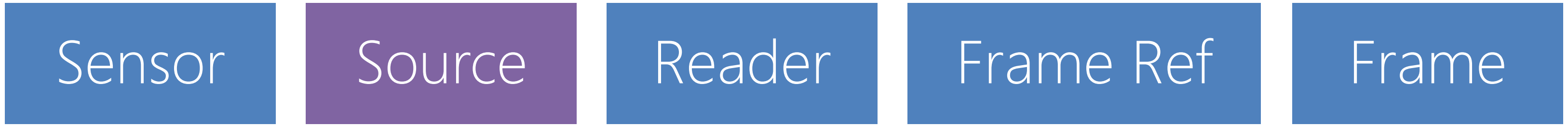
Sources

- Expose metadata about the source and give access to readers
- Sensors expose one source per data type



```
this.sensor.
```

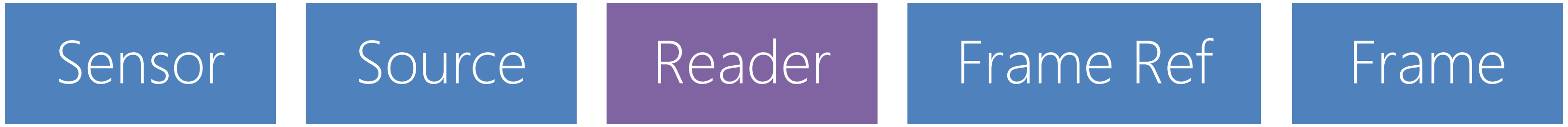
AudioSource
BodyFrameSource
BodyIndexFrameSource
ColorFrameSource
DepthFrameSource
InfraredFrameSource
LongExposureInfraredFrameSource
OpenMultiSourceFrameReader



Sensor
Source
Reader
Frame Ref
Frame

Readers

- Give access to frames
  - Events
  - Polling
- Multiple readers may be created on a single source
- Readers can be paused



Sensor

Source

Reader

Frame Ref

Frame

Readers

C#

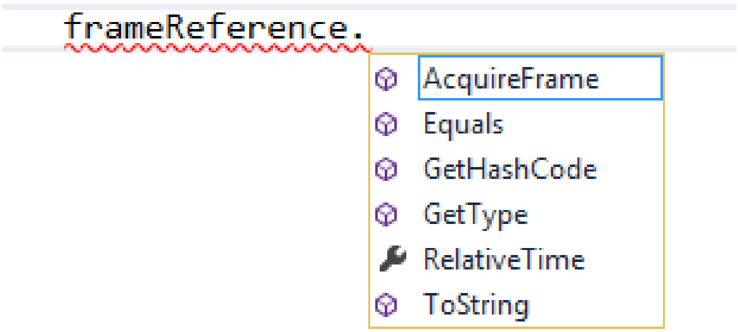
```
InfraredFrameReader reader = sensor.InfraredFrameSource.OpenReader();
reader.FrameArrived += InfraredReaderFrameArrived;
...
// see in the dark
...
```

JS

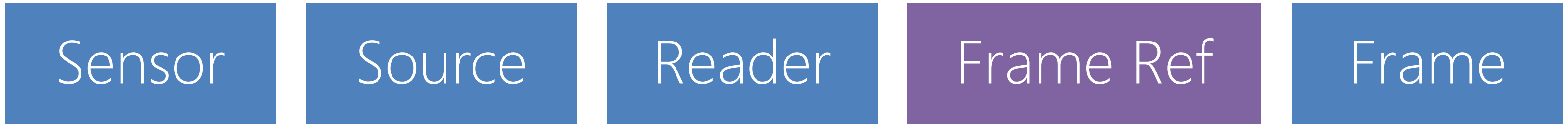
```
reader = sensor.infraredFrameSource.openReader();
reader.onframearrived = function (args) { ... };
...
// see in the dark
...
```

Frame references

- Sent in frame event args
- AcquireFrame gives access to the actual frame
- RelativeTime allow you to temporally correlate frames



```
frameReference.
```



Frame references

C#

```
void irReader_FrameArrived(InfraredFrameReader sender,
                           InfraredFrameArrivedEventArgs args){
    using (InfraredFrame frame = args.FrameReference.AcquireFrame())
    {
       if (frame != null)
       {
           // Get what you need from the frame
       }
    }
}
```

JS

```
var frame = args.frameReference.acquireFrame();
if(frame != null) {
    // Get what you need from the frame
    frame.close();
}
```

Frames

- Gives access to the frame data
  - Make a local copy or access the underlying buffer directly
- Contains metadata for the frame
  - e.g. Color: format, width, height, etc.
- Important: Minimize how long you hold onto the frame
  - Not Disposing frames will cause you to not receive more frames

Sensor

Source

Reader

Frame Ref

Frame

Data Source Details

Physical image sensors, ColorFrameSource, InfraredFrameSource, DepthFrameSource, BodyIndexFrameSource, BodyFrameSource, AudioFrameSource

Two Physical “Image” Sensors

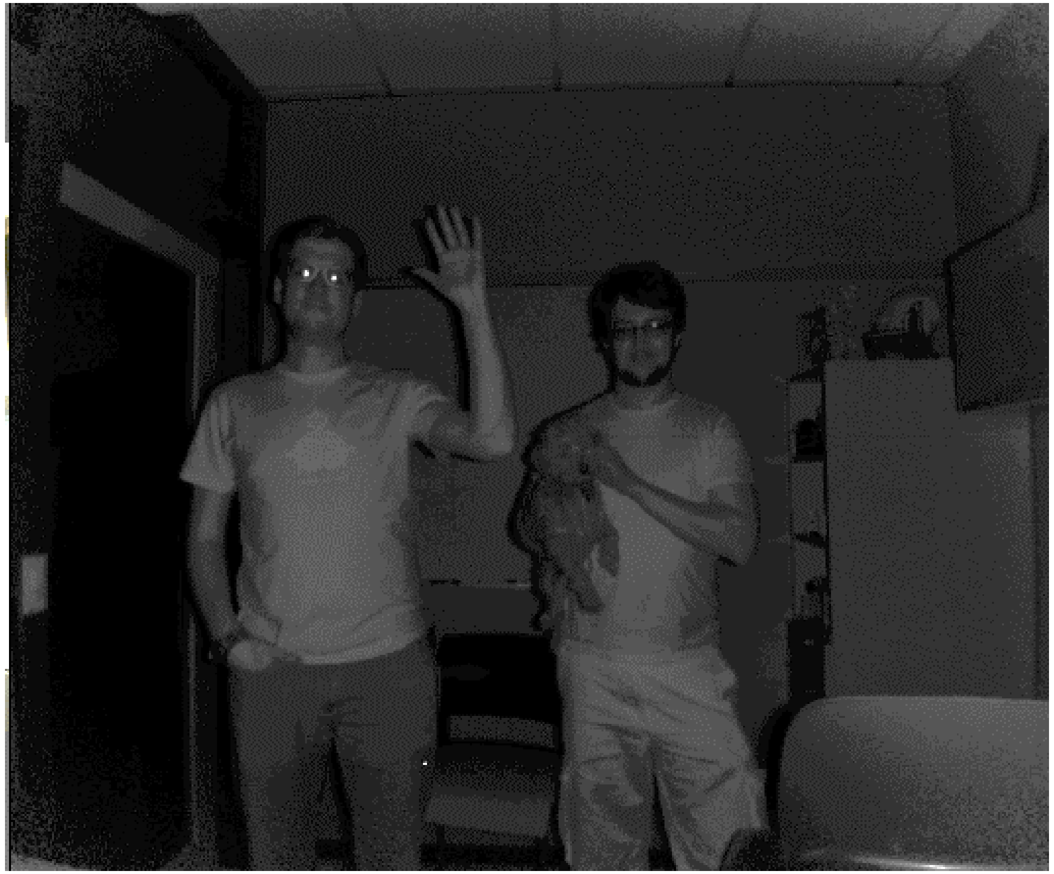
- Color Sensor
  - 1920 x 1080
  - 30 or 15 fps, based on lighting conditions

- IR Sensor
  - 512 x 424
  - 30 fps

ColorFrameSource

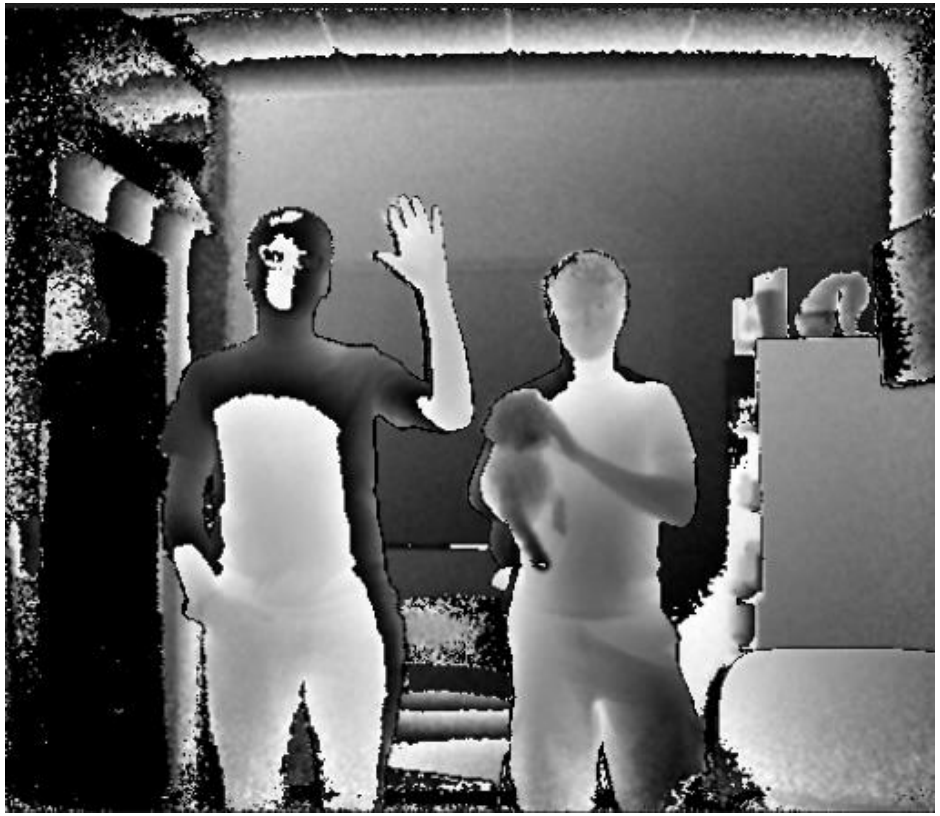
- 1920 x 1080 array of color pixels
  - 30 or 15 fps, based on lighting conditions
- Mirrored image to match other formats

InfraredFrameSource



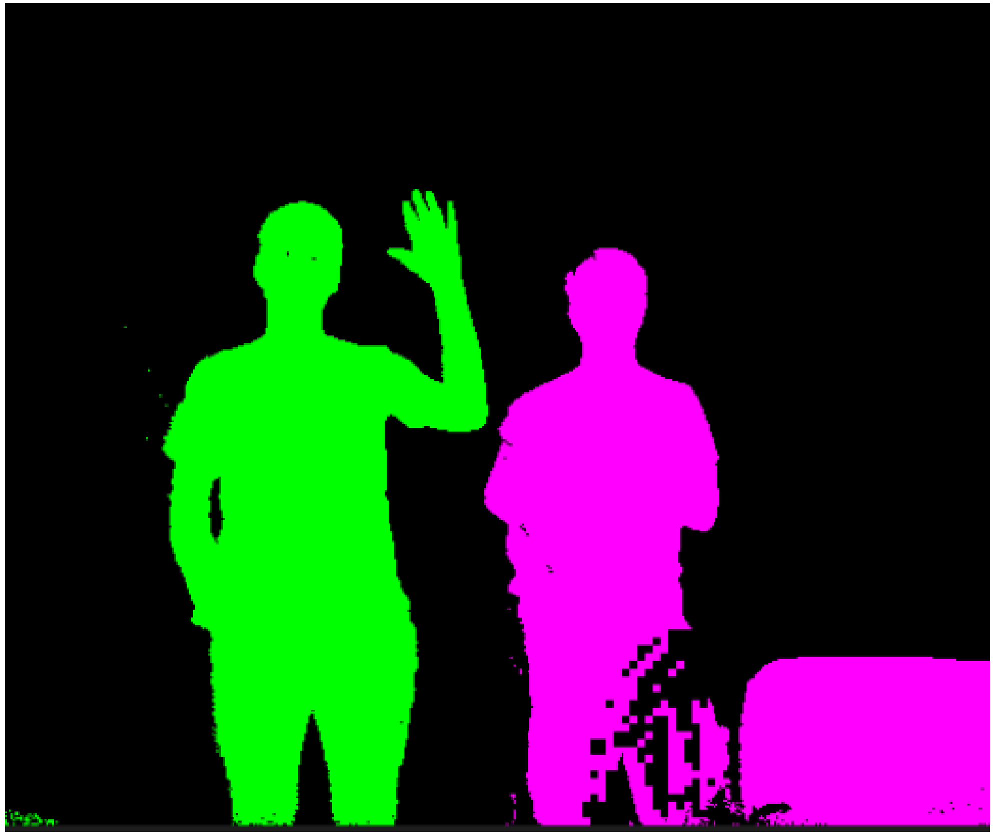
- Pixel Data
  - 16-bit IR intensity value
- Ambient light removed

DepthFrameSource



- Range: 0.5 - 8 meters
- Pixel Data
  - 16-bit distance in millimeters from the sensor’s focal plane

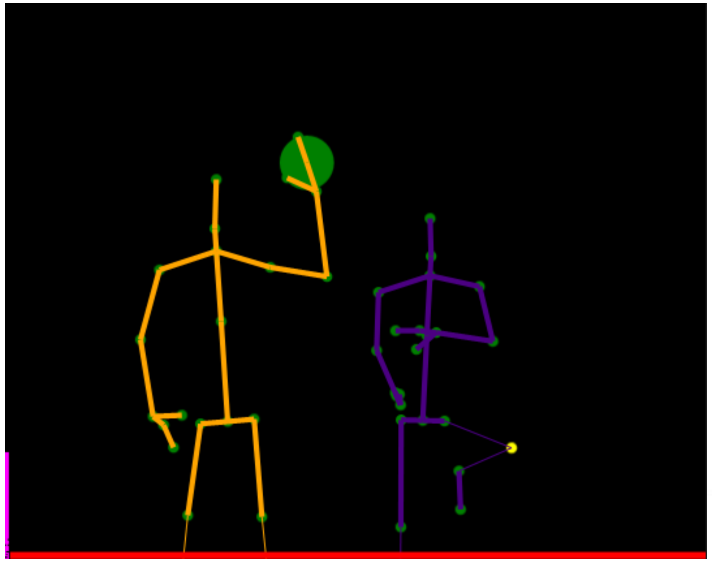
BodyIndexFrameSource

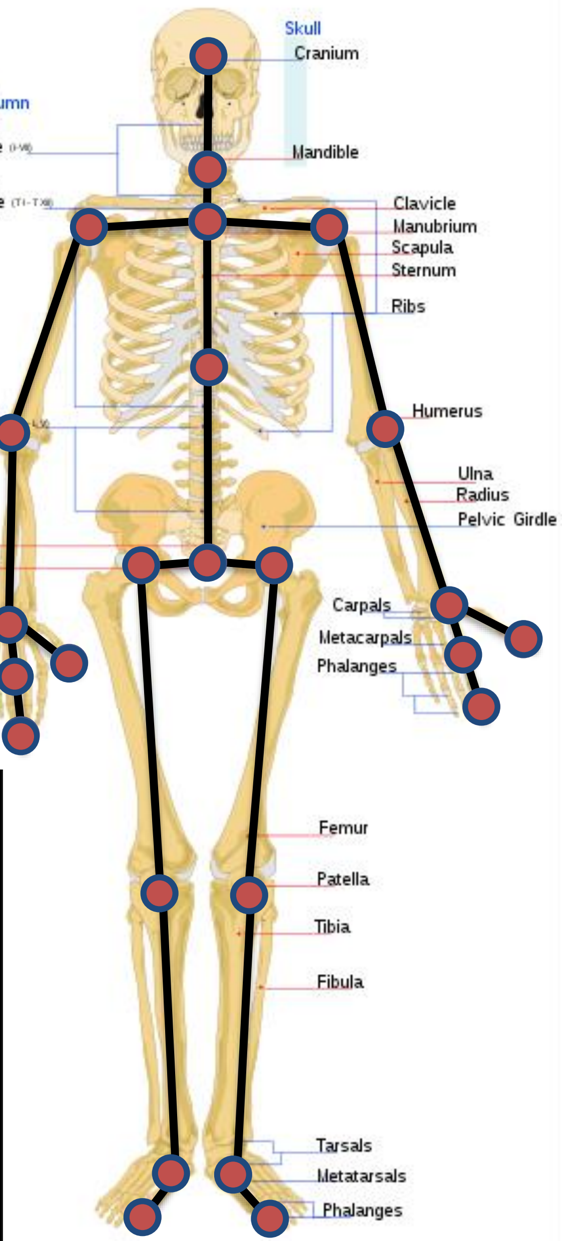


- Pixel Data
  - 0 to 5: Index of the corresponding body, as tracked by the body source
  - > 5: No tracked body at that pixel

BodyFrameSource

- Range is 0.5-4.5 meters
- Frame data is a collection of Body objects each with 25 joints
  - Each joint has position in 3D space and an orientation
- Up to six simultaneous bodies
- 30fps
- Hand State on 2 bodies
- Lean





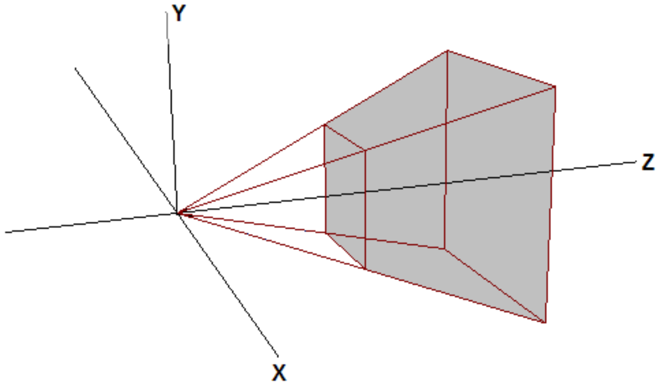
AudioFrameSource

- Data is audio samples captured over a specific interval of time
- Audio data is associated with an “audio beam”
  - A steerable “cone” of focus for audio
  - May be automatically or manually aimed

Other Important APIs

Coordinate mapping, MultiSourceFrameReader

Coordinate mapping



- Three coordinate systems

Name	Applies to	Dimensions	Units	Range	Origin
ColorSpacePoint	Color	2	pixels	1920x1080	Top left corner
DepthSpacePoint	Depth, Infrared, Body index	2	pixels	512x424	Top left corner
CameraSpacePoint	Body	3	meters	–	Infrared/depth camera

- Coordinate mapper provides conversions between each system
- Convert single or multiple points

MultiSourceFrameReader

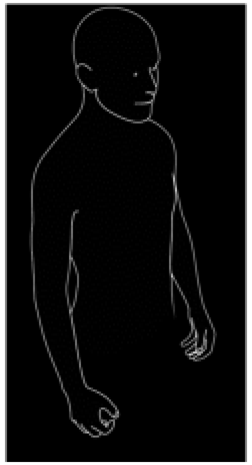
- Allows the app to get a matched set of frames from multiple sources on a single event
- Caveat: Delivers frames at the lowest FPS of the selected sources

Other Features, Resources

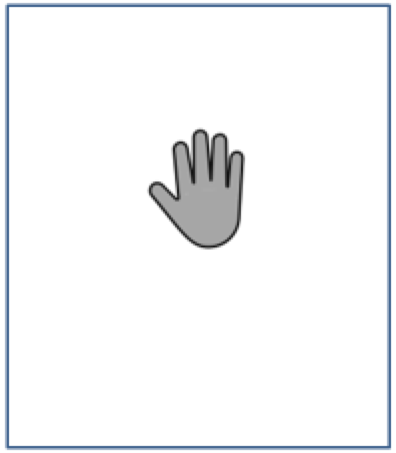
Jumpstart videos: http://aka.ms/k4wv2jumpstart

Hand Pointer Gestures and Speech

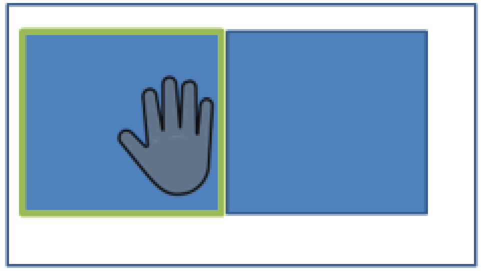
Hand pointer gestures



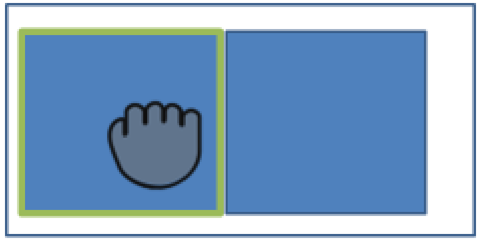
Engagement



Targeting



Press



Panning/Zoom

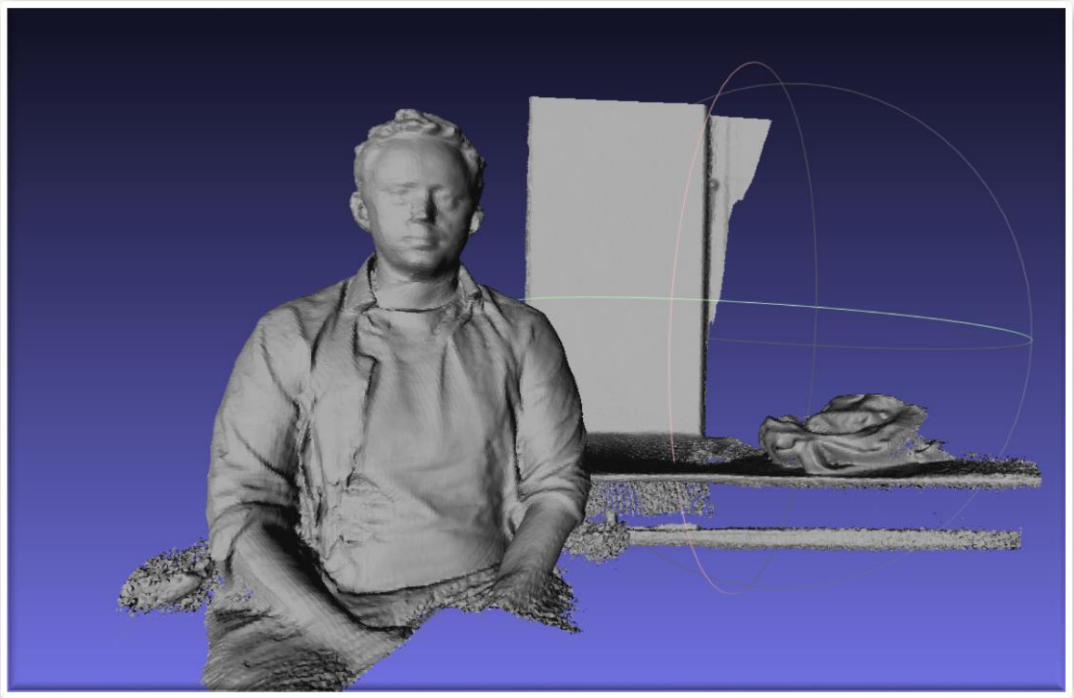
Available to desktop or store apps as an Opt-In input mechanism with Kinect for Windows 2.0

Speech

	Engagement for Global Commands	Engagement for App Commands	"In Experience" Commands
Xbox One w/ Kinect	*"Xbox"*	*"Xbox Select"*	*App-defined grammar*
Windows 8.0 or 8.1 w/ Kinect	*N/A*	*"Kinect" or …*	*App-defined grammar*

Available to desktop apps (ONLY) as an Opt-In input mechanism with Kinect for Windows 2.0

Fusion, Face, HD Face



Face

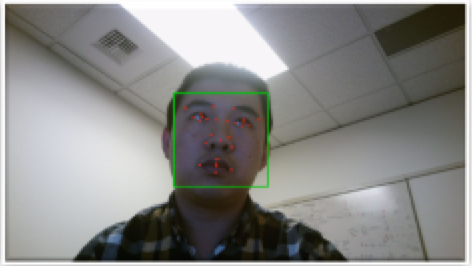
Fusion | Face | HD Face

Detection
- Outputs a bounding box around the face
- Can be visualized in color or IR

Alignment
- Identifies 5 facial landmarks on the face
- Can be visualized in color or IR

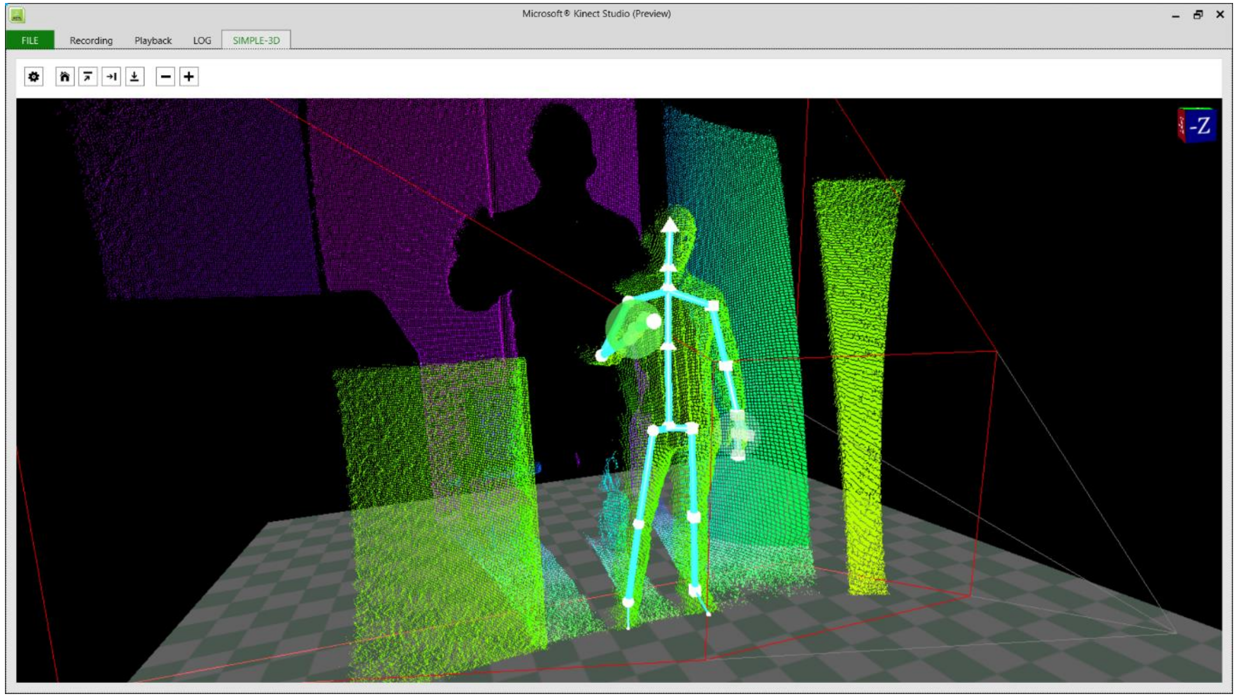
Orientation
- Returns quaternion of the head joint with respect to the sensor
- Quaternion prevents gimbal lock

Expressions
- Provides classifiers for happy, left / right eye open, engagement, mouth open and mouth moving



Our alignment and detection perform the operation in IR, but are converted to color using the coordinate mapper for output

Recording, Playback, and Gesture Recognition

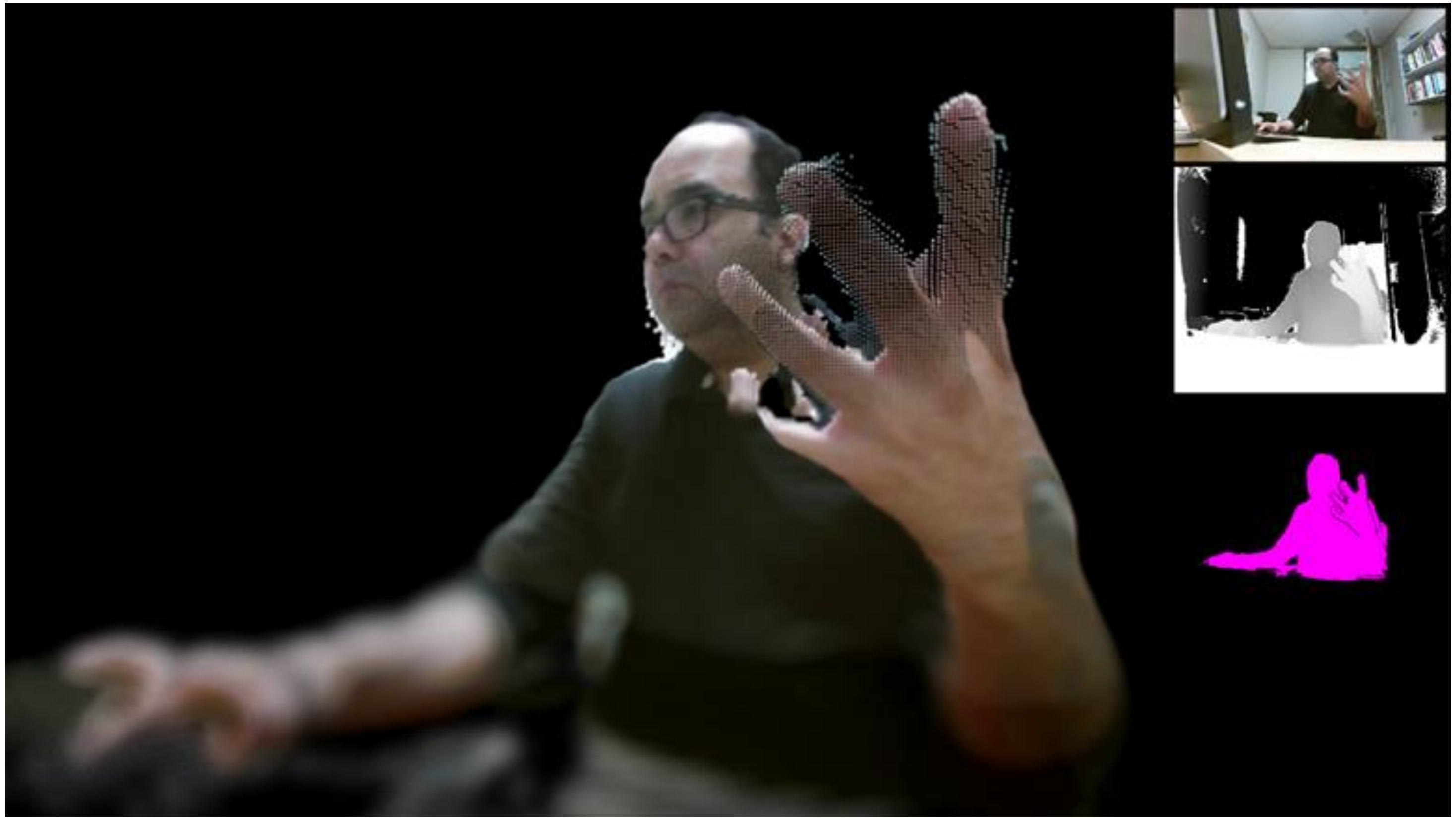


- *Kinect Studio*
  - Recording sensor data
  - Playback of data
  - APIs with Microsoft.Kinect.Tools.dll
- Gesture Recognition
  - Heuristic Recognition
  - Machine Learning Based Recognition [w/ *Gesture Builder*]

DEMO

Kinect Studio
Visual Gesture Builder
Custom Gesture Detection
Interactions
Fusion

Kinect Common Bridge and Unity Plug-in



CINDER

OF

unity

Unity3D

Microsoft

Using Kinect with Unity

- Setting up the Kinect Plug-in
- Unity Basics Demo
- Walk-thru of Managers
- Particle Emitter Demo
- Shaders in Unity
- Playtime

Unity Exercises

- Add other hand with particles

Kinect Common Bridge

Microsoft

Overview

Examples

Hello World!

Kinect Common Bridge

- Easy way to get Kinect data sources into other libraries
- Launched on Github in 2013 (MS Open Tech)
- Tight collaboration with Cinder & oF folks
- Native COM API still available for full control

DEMO

Overview of the KCB library (kcbv2lib.h)
Included examples: Color & IR
Hello World!

Cinder

Microsoft

Cinder-KCB2
Using the plugin: Kinect data sources into a Cinder app
Samples & Exercises

Using Kinect with Cinder

- Setting up the Cinder block (Cinder-KCB2)
- Using the plug-in: Getting Kinect data into a Cinder app
- Exercise: Drawing ribbons in air
- Particle Man
- Playtime

Cinder Exercises

- Change colors & length of the ribbon
- Draw two ribbons (one per hand)
- Have one hand erase the points from the other

openFrameworks

Setting up the plug-in (ofxKinectV2)

Using Kinect with openFrameworks

- Setting up the plug-in (ofxKinectV2)
  - Note: there are two with the same name
- Using the plug-in: Getting Kinect data sources into an oF app
- Fireflies demo
- Exercise: Simple point cloud, simple color, point cloud shader
- Playtime

oF Exercises

- Change size of particles
- Optimize the code (hint: concurrency::parallel_for)

Kinect Resources

SDK & Libraries

- K4W v2 SDK -> http://aka.ms/k4wv2previewsdk
- Kinect Common Bridge -> http://aka.ms/KCBv2
- Cinder -> http://aka.ms/CinderKCBv2
- openFrameworks -> http://aka.ms/oFKCBv2
- Creative Coding Setup -> http://aka.ms/k4wccsetup
- Metrilus Finger Tracking -> http://aka.ms/aiolosk4wv2

Tutorials, Content, Support Forum

- 7 Hours of Deep Content -> http://aka.ms/k4wv2jumpstart
- VGB Deep Dive -> http://aka.ms/k4wv2vgbdeepdive
- Windows Store oF -> http://aka.ms/OFWinrtDemos
- Developer Forums -> http://aka.ms/k4wv2forum

Events, Sensors, General Info

- Vancouver (Nov 8) -> http://aka.ms/k4whackvan
- Purchase Sensor -> http://aka.ms/k4wv2purchase
- General Info & Blog -> http://kinectforwindows.com

@KinectWindows, @benlower

We want your feedback and pull requests!