HRI Lab #7, SP2024 525.786: Human Robotics Interaction

Introduction:

In this lab we will explore interfacing various types of motion tracking tools. These tools, in combination with the bioinstrumentation inputs you've used previously, can be used for natural and intuitive Human Robotic Interaction. The tools that we'll be using include the following devices:

- Inertial measurement Unit (3-axis, 6 Degrees of Freedom) on the Myo
- Kinect (20 Degrees of Freedom, Whole Body)

Setup the MiniVIE environment

```
thisPath = cd;
cd('C:\GitHub\MiniVIE');
MiniVIE.configurePath;
```

```
[MiniVIE.m] Configured MiniVIE path at: C:\GitHub\minivie
```

```
cd(thisPath);
```

Part 1: Inertial Measurement Unit on the Myo

Operating Principle: the Myo armband contains an Angular rate sensor, accelerometer, and magnetometer to resolve orientation. These sensors can be used to track motion of the arm in addition to the EMG signals

Before running the code below, be sure to start MyoUdp.exe in the MiniVIE/+Inputs folder

It might be more straightforward to run this in the Command Window (select, then hit F9)

```
% Setup the Myo Armband in matlab
hMyo = Inputs.MyoUdp.getInstance();
```

```
[MyoUdp] Calling constructor
```

```
hMyo.initialize()
```

```
[UserConfig.m] Calling constructor with config file C:\GitHub\minivie\user_config.xml
[UserConfig.m] myoUdpRate=200
[UserConfig.m] myoUdpPort1=10001
[UserConfig.m] myoUdpPort2=10002

Loaded pnet Version  2.0.5  2003-09-16 Copyright (C) Peter Rydesäter, Sweden, et al. , 1998 - 2003
[PnetClass] Opened pnet socket #0 at local port: 10001; Default destination: port 45001 @ 127.0.0.1
[PnetClass] Opened pnet socket #1 at local port: 10002; Default destination: port 45001 @ 127.0.0.1
[MyoUdp] UDP Data Stream 1-8 NOT Detected
[MyoUdp] UDP Data Stream 9-16 NOT Detected
ans = 0
```

```
pause(1)

% Ensure data is streaming
myoData = hMyo.getData();
```

```
if isempty(hMyo.Orientation)
    error('Setup Failed.  MyoUdp Running?');
end

% Setup figure to show orientation
figure(1);
clf
daspect([1 1 1])
view(-170,15)

% Plot reference triad
PlotUtils.triad(eye(4),0.5)
```

```
ans = 10.0002
```

```
% Plot 'moving' triad
myoTriadHandle = PlotUtils.triad(eye(4),2);
StartStopForm([])
while StartStopForm()
    drawnow
    hMyo.getData();  % update stream receive
    Rxyz = hMyo.getEulerAngles; % Get roll pitch yaw
    disp(Rxyz)
    Rmat = LinAlg.makeRotationMtx(Rxyz); % create rotation matrix
    transMat = [Rmat [0; 0; 0]; 0 0 0 1];
    set(myoTriadHandle,'Matrix', transMat);
end
```

```
Warning: The JavaFrame figure property will be removed in a future release. For more information see UI
Alternatives for MATLAB Apps on mathworks.com.
   -5.8929
   75.7420
   27.1762

   -5.9357
   75.7230
   27.1506

   -5.9529
   75.7237
   27.1286

   -5.9821
   75.7151
   27.1041

   -6.1734
   75.6645
   26.8575

   -6.2194
   75.6566
   26.8114

   -6.2194
   75.6566
```

26.8114

-6.2194
75.6566
26.8114

-6.2627
75.6529
26.7696

-6.2655
75.6487
26.7654

-6.3087
75.6450
26.7236

-6.3375
75.6364
26.6995

-6.3546
75.6371
26.6777

-6.3546
75.6371
26.6777

-6.3834
75.6285
26.6536

-6.3834
75.6285
26.6536

-6.3834
75.6285
26.6536

-6.4265
75.6248
26.6119

-6.4682
75.6223
26.5689

-6.4723
75.6168
26.5661

-6.5181
75.6089
26.5203

-6.5160
75.6189
26.5320

-6.5836
75.6266

```
 26.4897

 -6.6732
 75.6352
 26.4302

 -6.6732
 75.6352
 26.4302

 -6.7398
 75.6625
 26.3746

 -6.7461
 75.6783
 26.3621

 -6.7085
 75.6920
 26.3840

 -7.5919
 75.5544
 25.1261

 -7.9197
 75.4123
 24.4553

 -8.7152
 74.8127
 22.3710

-10.0339
 74.4230
 20.4492

-10.6882
 74.2017
 19.4056

-11.6443
 73.8037
 17.9319

-12.2212
 72.7944
 15.9580

-11.9317
 70.5630
 13.2794

-18.6511
 59.6893
  0.3174

-20.2767
 43.3989
-15.6442

-21.8452
 29.4117
```

```
-23.2543

-19.2249
 23.9366
-25.1445

 -9.6569
 16.9353
-24.2780

 -5.6019
 15.3574
-21.7821

 -3.7587
 14.1628
-15.8936

 -1.0289
 15.6943
 -9.7559

  5.3394
 16.9041
 -4.7580

  5.9294
 17.4500
 -2.8948

  1.6480
 16.5952
 -4.9006

  2.7881
 16.1932
 -5.3087

  7.0680
 16.4266
 -4.0779

  4.8871
 16.0045
 -4.4679

  4.0072
 16.0786
 -4.4679

  5.2186
 15.5209
 -4.5403

  5.2750
 15.3337
 -4.6960

  4.9484
 14.8577
 -5.1106

  5.4063
 14.2127
```

-5.9619

 6.2692
14.5867
-6.3905

 5.9488
15.4881
-6.6596

 6.6869
13.6763
-6.2227

 5.8749
14.5392
-5.8829

 5.9952
15.3489
-5.8239

 5.8580
14.4133
-5.5513

 5.5885
18.1618
-5.6054

 5.4010
21.1641
-5.6897

 5.7894
18.9542
-5.4830

 5.8292
17.2479
-5.3872

 5.5374
17.8662
-5.4618

 5.4172
18.1683
-5.3835

 5.8481
15.8154
-5.0937

 5.7933
15.7112
-5.0562

 5.1452
17.3000
-5.2036

 5.3668
16.1167

-5.1117

 5.2153
16.0444
-5.1002

 5.2846
15.3935
-5.0316

 5.2245
15.3240
-4.9717

 5.0580
15.2824
-4.8542

 4.9813
15.2236
-4.7533

 4.9187
15.1528
-4.6582

 4.8398
15.1186
-4.6202

 4.6360
15.1466
-4.4977

 3.8275
15.4448
-4.1786

 3.3512
15.7009
-3.9818

 2.8138
16.1383
-3.6863

 2.6804
16.1881
-3.6193

 2.7216
16.1472
-3.6353

 2.8562
15.9781
-3.7189

 2.8068
15.8987
-3.7344

 2.7427
15.9025

-3.6867

 2.6862
15.9182
-3.5811

 2.6650
15.8958
-3.5351

 2.6578
15.8442
-3.4945

 2.6512
15.8232
-3.4959

 2.7469
15.7416
-3.4844

 2.7923
15.6932
-3.4720

 2.9308
15.5691
-3.4133

 2.9472
15.5268
-3.4049

 2.9577
15.5048
-3.3826

 2.9690
15.4551
-3.3610

 3.0371
15.3931
-3.3391

 3.0375
15.3793
-3.3394

 3.0842
15.3098
-3.3207

 3.0774
15.2958
-3.3220

 3.1026
15.2602
-3.2982

 3.1256
15.2319

-3.2886

 3.1260
15.2181
-3.2889

 3.1190
15.2110
-3.2900

 3.1730
15.1771
-3.2626

 3.1662
15.1552
-3.2639

 3.1592
15.1481
-3.2650

 3.1743
15.1207
-3.2638

 3.1673
15.1058
-3.2649

 3.1769
15.0848
-3.2501

 3.1771
15.0779
-3.2503

 3.1947
15.0560
-3.2272

 3.1950
15.0422
-3.2275

 3.1812
15.0210
-3.2300

 3.1812
15.0132
-3.2300

 3.1814
15.0062
-3.2301

 3.1746
14.9922
-3.2314

 3.2853
14.9381

-3.2041

 3.2853
14.9381
-3.2041

 3.3861
14.9344
-3.1912

 3.3304
14.9711
-3.0705

 3.3474
14.9778
-3.0469

 3.4220
14.9296
-3.0242

 3.5397
14.8754
-2.9961

 3.5635
14.8546
-2.9796

 3.5567
14.8406
-2.9809

 3.5860
14.8671
-2.9127

 3.5862
14.8601
-2.9129

 3.5929
14.8397
-2.9198

 3.6583
14.7916
-2.9126

 3.6516
14.7707
-2.9142

 3.6517
14.7637
-2.9144

 3.6517
14.7559
-2.9144

 3.6956
14.7152

-2.9101

 3.8375
14.5924
-2.9237

 3.9552
14.4776
-2.9618

 4.0692
14.3197
-2.9808

 4.1250
14.2513
-2.9972

 4.1182
14.2373
-2.9985

 4.1716
14.1952
-2.9721

 4.2585
14.0993
-2.9717

 4.3979
14.0036
-2.9438

 4.4225
13.9825
-2.9205

 4.3687
13.9467
-2.7871

 4.2597
14.0128
-2.5724

 4.1971
14.1097
-2.3793

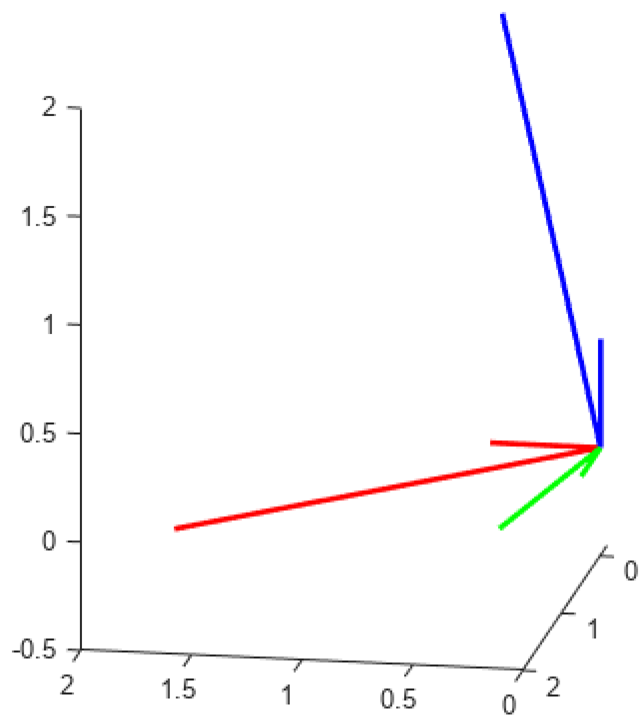
 4.1722
14.1309
-2.2333

 4.2841
14.0295
-2.0821

 4.3534
13.9727
-2.0404

 4.6493

```
   13.8018
   -2.0039
```




```
hMyo.close()
```

```
[MyoUdp] Deleting Udp comms object
[PnetClass.m] Closed pnet socket #0 at local port: 10001
[PnetClass.m] Closed pnet socket #1 at local port: 10002
```

```
cleanup;
```

```
Warning: Objects of 'onCleanup' class exist.  Cannot clear this class or any of its superclasses.
Warning: Objects of 'onCleanup' class exist.  Cannot clear this class or any of its superclasses.
Warning: Objects of 'onCleanup' class exist.  Cannot clear this class or any of its superclasses.
Warning: Objects of 'onCleanup' class exist.  Cannot clear this class or any of its superclasses.
Warning: Objects of 'onCleanup' class exist.  Cannot clear this class or any of its superclasses.
Warning: Objects of 'onCleanup' class exist.  Cannot clear this class or any of its superclasses.
Warning: Objects of 'onCleanup' class exist.  Cannot clear this class or any of its superclasses.
Warning: Objects of 'onCleanup' class exist.  Cannot clear this class or any of its superclasses.
Warning: Objects of 'onCleanup' class exist.  Cannot clear this class or any of its superclasses.
Warning: Objects of 'onCleanup' class exist.  Cannot clear this class or any of its superclasses.
Loaded pnet Version  2.0.5  2003-09-16 Copyright (C) Peter Rydesäter, Sweden, et al. , 1998 - 2003
Warning: instrfindall will be removed in a future release. There is no simple replacement for this.
Warning: Objects of 'onCleanup' class exist.  Cannot clear this class or any of its superclasses.
Warning: Objects of 'onCleanup' class exist.  Cannot clear this class or any of its superclasses.
Warning: Objects of 'onCleanup' class exist.  Cannot clear this class or any of its superclasses.
Warning: Objects of 'onCleanup' class exist.  Cannot clear this class or any of its superclasses.
Warning: Objects of 'onCleanup' class exist.  Cannot clear this class or any of its superclasses.
Warning: Objects of 'onCleanup' class exist.  Cannot clear this class or any of its superclasses.
Warning: Objects of 'onCleanup' class exist.  Cannot clear this class or any of its superclasses.
Warning: Objects of 'onCleanup' class exist.  Cannot clear this class or any of its superclasses.
Warning: Objects of 'onCleanup' class exist.  Cannot clear this class or any of its superclasses.
```

```
cleanup
-------------------------------------
-------------------------------------
-------------------------------------
```

Questions:

1. What is the 'global' position of the myo armband?
2. global Position is 0,0,0
3. What is its local coordinate frame?
4. Local coordinate Frame is the orientation of the MyoBand relative to the global Frame.
5. What processing would you do to use the myo armband as a sensor to control the elbow flexion angle of a robotic device?
6. You could map the rotation of the MyoBand about the Y-Axis onto the elbow angle of the robot

Use the Myo Orientation Sensors to Plot Arm Position

Edit the code below to show (in a streaming fashion) the real-time orientation and position of the forearm.

It might be more straightforward to run this in the Command Window (select, then hit F9)

```
hMyo = Inputs.MyoUdp.getInstance();
```

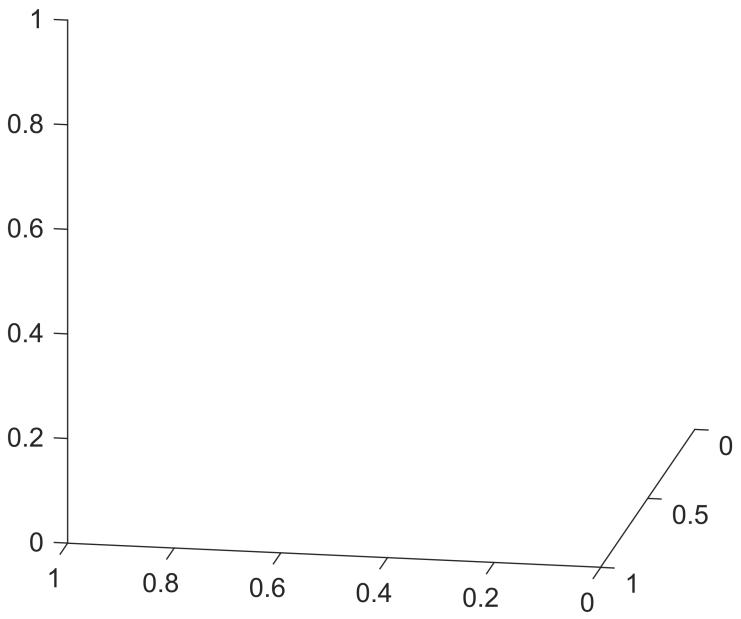
```
[MyoUdp] Calling constructor
```

```
hMyo.initialize()
```

```
[UserConfig.m] Calling constructor with config file C:\GitHub\minivie\user_config.xml
[UserConfig.m] myoUdpRate=200
[UserConfig.m] myoUdpPort1=10001
[UserConfig.m] myoUdpPort2=10002

Loaded pnet Version  2.0.5  2003-09-16 Copyright (C) Peter Rydesäter, Sweden, et al. , 1998 - 2003
[PnetClass] Opened pnet socket #0 at local port: 10001; Default destination: port 45001 @ 127.0.0.1
[PnetClass] Opened pnet socket #1 at local port: 10002; Default destination: port 45001 @ 127.0.0.1
[MyoUdp] UDP Data Stream 1-8 Detected
[MyoUdp] UDP Data Stream 9-16 NOT Detected
ans = 0
```

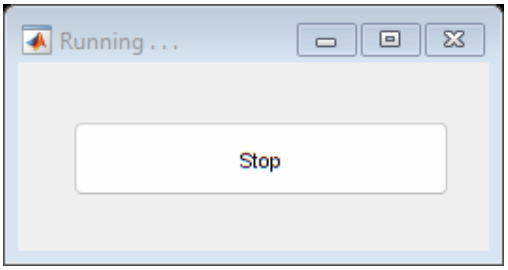
```
pause(1)

% Ensure data is streaming
myoData = hMyo.getData();
if isempty(hMyo.Orientation)
    error('Setup Failed.  MyoUdp Running?');
end

% Can you create a real time plot of your hand's position in space using
% the real-time orientation matrix from the Myo?  assume the distance from
% Myo to hand is 25 cm.
myoOrigin = [0, 0, 0];
handPosStart = [0.25, 0, 0]; % in meters
```

```
figHandle = figure(1);
clf
daspect([1 1 1])
view(-170,15)
```




```
StartStopForm([])
handPos=0;
while StartStopForm()
    % put your code here for grabbing the rotation matrix
    hMyo.getData();  % update stream receive
    Rxyz = hMyo.getEulerAngles; % Get roll pitch yaw
    Rmat = LinAlg.makeRotationMtx(Rxyz); % create rotation matrix

    % put your code here for calculating a new handPos based on the
    % rotation matrix and handPosStart
    handPos=handPosStart*Rmat;
    %
    plot3([0, handPos(1)], [0, handPos(2)], [0, handPos(3)], '.-');
    set(gca, 'XLim', [-0.25 0.25], 'YLim', [-0.25 0.25], 'ZLim', [-0.25 0.25]);
    xlabel('x'); ylabel('y'); zlabel('z');

end
```

```
Warning: The JavaFrame figure property will be removed in a future release. For more information see UI
Alternatives for MATLAB Apps on mathworks.com.
```




```
Unrecognized function or variable 'handPos'.
```

```
hMyo.close()
cleanup;
```

Part 2: Kinect

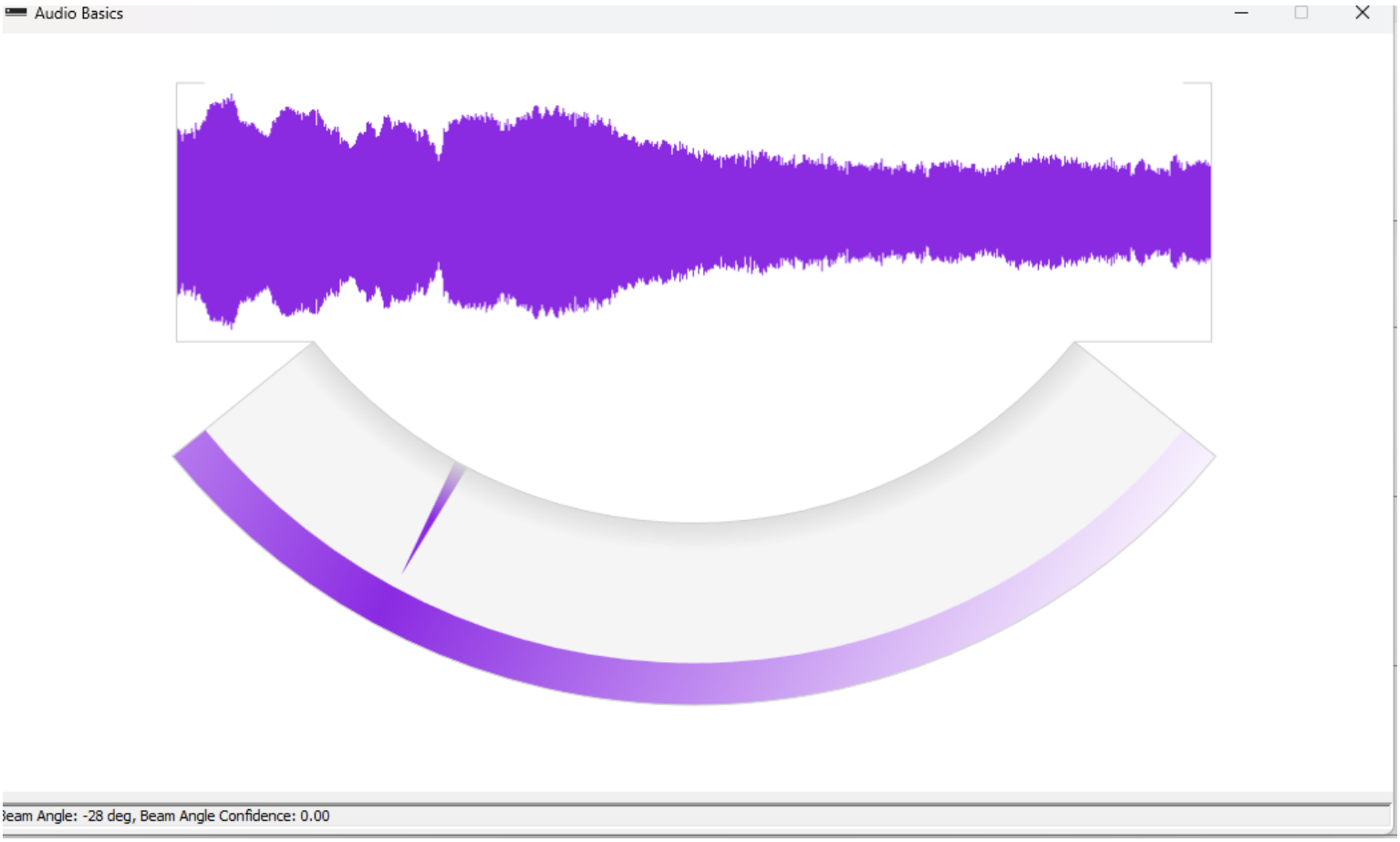
Open "SDK Browser for Kinect" (search in Windows Start Menu)

Run the following examples:

1. Audio Basics
2. Depth Basics
3. Face Tracking Basics
4. Skeletal Basics

Copy and paste window screenshots from each:

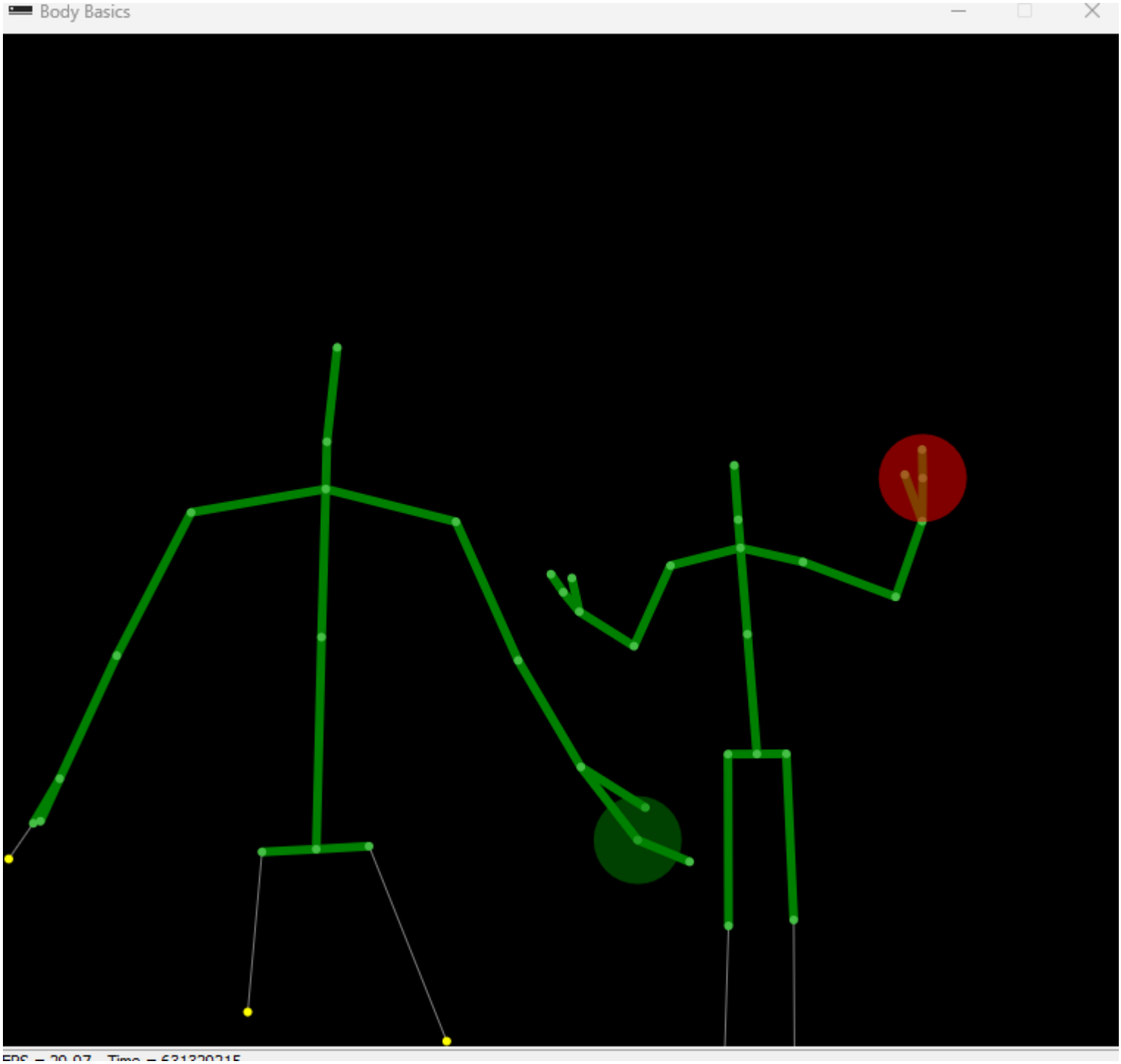
1. **Audio Basic**



2. Depth Basic

3. Face Tracking Basic

4. Skeletal Basic



Questions:

- What advantage do microphone arrays have over traditional microphones / speech input devices? **Arrays allow directional sensing of audio, while traditionals do not.**
- At what range is the face tracking effective? How might this be used as a robot controller? **Face tracking seems to be limited to between 1 and 20 feet. The further away the face, the harder the**

program works to determine mouth or eye data. A robot could use this data to determine mood or anticipate needs based on facial expressions.

- What kind of motions and postures does the skeletal model capture? When does it break down? **The skeletal model captures the major joints and limbs and can recognize a closed fist versus open, and generally tracks the position of the thumb. It breaks down too close to the sensor if there are not enough joints visible.**

Kinect sampling from MATLAB

derived from: https://www.mathworks.com/help/imaq/examples/using-the-kinect-r-for-windows-r-from-image-acquisition-toolbox-tm.html

required MATLAB Support Package for Kinect: https://www.mathworks.com/hardware-support/kinect-windows.html

It might be more straightforward to run this in the Command Window (select, then hit F9)

```
utilpath =
'C:\ProgramData\MATLAB\SupportPackages\R2019b\toolbox\imaq\supportpackages\kinectruntime\kinectforwindowsruntimeexamples';
addpath(utilpath);

% The Kinect for Windows Sensor shows up as two separate devices in IMAQHWINFO.
hwInfo = imaqhwinfo('kinect')
```

```
hwInfo = struct with fields:
       AdaptorDllName: 'C:\ProgramData\MATLAB\SupportPackages\R2023a\toolbox\imaq\supportpackages\kinectruntime\adap
    AdaptorDllVersion: '6.7.1 (R2023a)'
          AdaptorName: 'kinect'
            DeviceIDs: {[1]  [2]}
           DeviceInfo: [1×2 struct]
```

```
hwInfo.DeviceInfo(1)
```

```
ans = struct with fields:
              DefaultFormat: 'BGR_1920x1080'
       DeviceFileSupported: 0
                DeviceName: 'Kinect V2 Color Sensor'
                  DeviceID: 1
     VideoInputConstructor: 'videoinput('kinect', 1)'
    VideoDeviceConstructor: 'imaq.VideoDevice('kinect', 1)'
          SupportedFormats: {'BGR_1920x1080'}
```

```
hwInfo.DeviceInfo(2)
```

```
ans = struct with fields:
              DefaultFormat: 'Depth_512x424'
       DeviceFileSupported: 0
                DeviceName: 'Kinect V2 Depth Sensor'
                  DeviceID: 2
     VideoInputConstructor: 'videoinput('kinect', 2)'
    VideoDeviceConstructor: 'imaq.VideoDevice('kinect', 2)'
          SupportedFormats: {'Depth_512x424'}
```

```

```

```
% Create the VIDEOINPUT objects for the two streams
colorVid = videoinput('kinect',1)
```

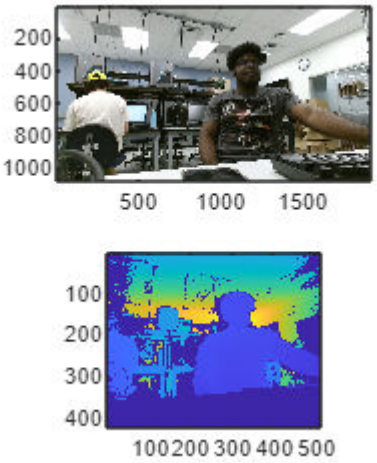
```
Summary of Video Input Object Using 'Kinect V2 Color Sensor'.

   Acquisition Source(s):  Kinect V2 Color Source is available.

  Acquisition Parameters:  'Kinect V2 Color Source' is the current selected source.
                           10 frames per trigger using the selected source.
                           'BGR_1920x1080' video data to be logged upon START.
                           Grabbing first of every 1 frame(s).
                           Log data to 'memory' on trigger.

      Trigger Parameters:  1 'immediate' trigger(s) on START.

                  Status:  Waiting for START.
                           0 frames acquired since starting.
                           0 frames available for GETDATA.
```

```
depthVid = videoinput('kinect',2)
```

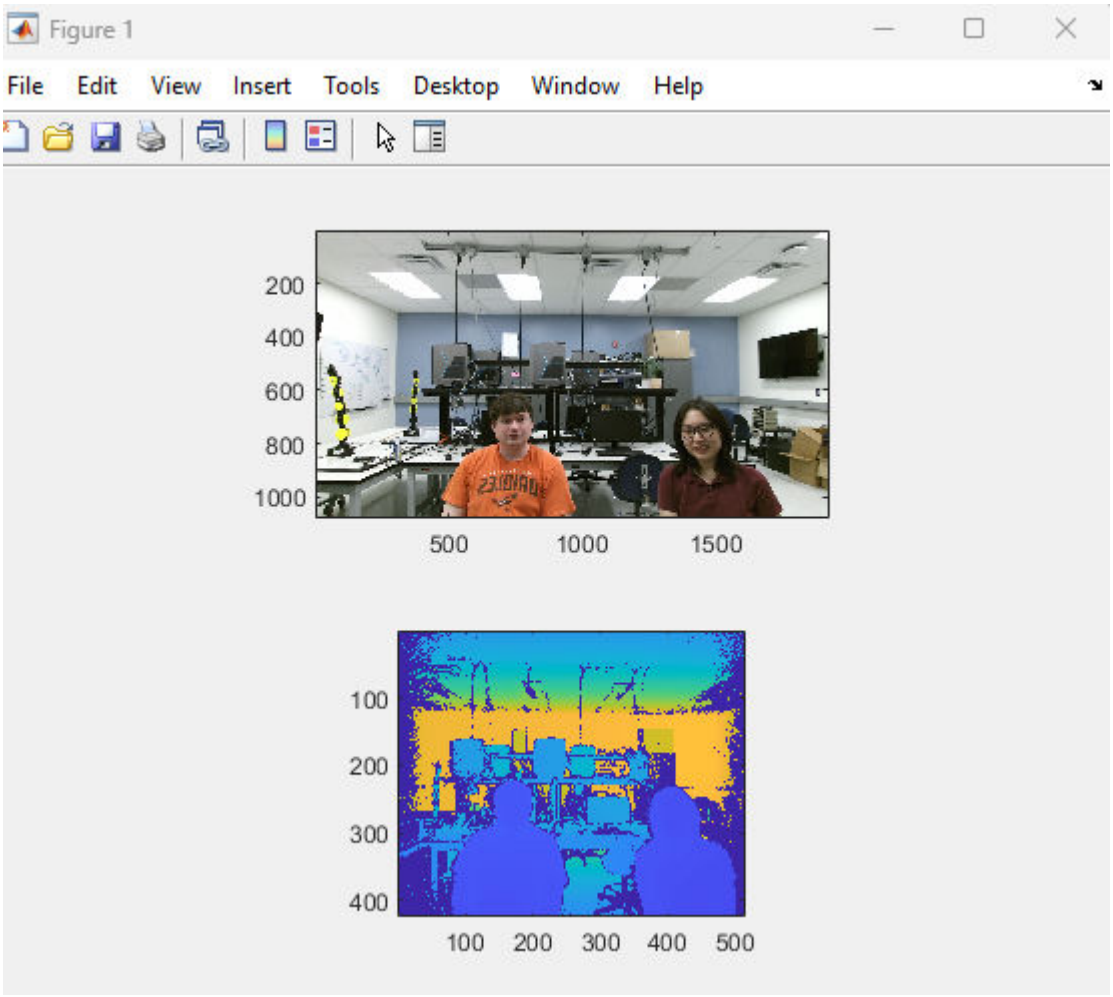
```
Summary of Video Input Object Using 'Kinect V2 Depth Sensor'.

   Acquisition Source(s):  Kinect V2 Depth Source is available.

  Acquisition Parameters:  'Kinect V2 Depth Source' is the current selected source.
                           10 frames per trigger using the selected source.
                           'Depth_512x424' video data to be logged upon START.
                           Grabbing first of every 1 frame(s).
                           Log data to 'memory' on trigger.

      Trigger Parameters:  1 'immediate' trigger(s) on START.

                  Status:  Waiting for START.
                           0 frames acquired since starting.
                           0 frames available for GETDATA.
```

```
% Set the triggering mode to 'manual'
triggerconfig([colorVid depthVid],'manual');
colorVid.FramesPerTrigger = 100;
depthVid.FramesPerTrigger = 100;

% Start the color and depth device. This begins acquisition, but does not
% start logging of acquired data.
start([colorVid depthVid]);

% Trigger the devices to start logging of data.
trigger([colorVid depthVid]);

% Retrieve the acquired data
[colorFrameData,colorTimeData,colorMetaData] = getdata(colorVid);
[depthFrameData,depthTimeData,depthMetaData] = getdata(depthVid);

% Stop the devices
stop([colorVid depthVid]);

% display one of the frames
```

```
figure(1);
subplot(2, 1, 1);
imagesc(colorFrameData(:, :, :, 50));axis equal tight;
subplot(2, 1, 2);
imagesc(depthFrameData(:, :, :, 50)); axis equal tight;
```



Copy and the screenshots of your RGB and depth maps:



Skeletal tracking

Grab a frame including a person's body (i.e., "skeleton" in Kinect terms)

```
depthSrc = getselectedsource(depthVid)
```

```
depthSrc =
   Display Summary for Video Source Object:

      General Settings:
        Parent = [1x1 videoinput]
        Selected = on
        SourceName = Kinect V2 Depth Source
        Tag = [0x0 string]
        Type = videosource

      Device Specific Properties:
        EnableBodyTracking = on
```

```
% Turn on skeletal tracking.
depthSrc.EnableBodyTracking = 'on';

% Acquire 100 frames with tracking turned on.
% Remember to have a person in person in front of the
```

```
% Kinect for Windows to see valid tracking data.
colorVid.FramesPerTrigger = 100;
depthVid.FramesPerTrigger = 100;

start([colorVid depthVid]);
trigger([colorVid depthVid]);

% Retrieve the frames and check if any Skeletons are tracked
[frameDataColor] = getdata(colorVid);
[frameDataDepth, timeDataDepth, metaDataDepth] = getdata(depthVid);

% View skeletal data from depth metadata
metaDataDepth
```

```
metaDataDepth = 100×1 struct
```

...

Fields	AbsTime	BodyIndexFr...	BodyTrackin...	ColorJointI...	DepthJointI...	FrameNumber	HandLeftCon...	HandLeftState
1	[2025,3,...	424×512 dou...	[7.2058e+16...	25×2×6 double	25×2×6 double	1	1×6 logical	[2,1,1,1...
2	[2025,3,...	424×512 dou...	[7.2058e+16...	25×2×6 double	25×2×6 double	2	1×6 logical	[2,1,1,1...
3	[2025,3,...	424×512 dou...	[7.2058e+16...	25×2×6 double	25×2×6 double	3	1×6 logical	[2,1,1,1...
4	[2025,3,...	424×512 dou...	[7.2058e+16...	25×2×6 double	25×2×6 double	4	1×6 logical	[2,1,1,1...
5	[2025,3,...	424×512 dou...	[7.2058e+16...	25×2×6 double	25×2×6 double	5	1×6 logical	[2,1,1,1...
6	[2025,3,...	424×512 dou...	[7.2058e+16...	25×2×6 double	25×2×6 double	6	1×6 logical	[2,1,1,1...
7	[2025,3,...	424×512 dou...	[7.2058e+16...	25×2×6 double	25×2×6 double	7	1×6 logical	[2,1,1,1...
8	[2025,3,...	424×512 dou...	[7.2058e+16...	25×2×6 double	25×2×6 double	8	1×6 logical	[2,1,1,1...
9	[2025,3,...	424×512 dou...	[7.2058e+16...	25×2×6 double	25×2×6 double	9	1×6 logical	[2,1,1,1...
10	[2025,3,...	424×512 dou...	[7.2058e+16...	25×2×6 double	25×2×6 double	10	1×6 logical	[2,1,1,1...
11	[2025,3,...	424×512 dou...	[7.2058e+16...	25×2×6 double	25×2×6 double	11	1×6 logical	[2,1,1,1...
12	[2025,3,...	424×512 dou...	[7.2058e+16...	25×2×6 double	25×2×6 double	12	1×6 logical	[2,1,1,1...
13	[2025,3,...	424×512 dou...	[7.2058e+16...	25×2×6 double	25×2×6 double	13	1×6 logical	[2,1,1,1...
14	[2025,3,...	424×512 dou...	[7.2058e+16...	25×2×6 double	25×2×6 double	14	1×6 logical	[2,1,1,1...

⋮

```
% Check for tracked skeletons from depth metadata
anyBodiesTracked = any(metaDataDepth(95).IsBodyTracked ~= 0)
```

```
anyBodiesTracked = logical
   1
```

```
% See which skeletons were tracked.
trackedBodies = find(metaDataDepth(95).IsBodyTracked)
```

```
trackedBodies = 1
```

```
jointCoordinates = metaDataDepth(95).JointPositions(:, :, trackedBodies)
```

```
jointCoordinates = 25×3
    0.0780   -0.5827    1.3692
    0.0805   -0.2876    1.4040
    0.0816   -0.0014    1.4224
    0.0911    0.1579    1.4083
   -0.1140   -0.0854    1.3964
   -0.3362   -0.0806    1.2848
   -0.4303    0.0710    1.1144
   -0.4509    0.1336    1.0826
    0.2839   -0.0892    1.4164
    0.4828   -0.1047    1.3156
      ⋮
```

```
% Skeleton's joint indices with respect to the color image
jointIndices = metaDataDepth(95).ColorJointIndices(:, :, trackedBodies)
```

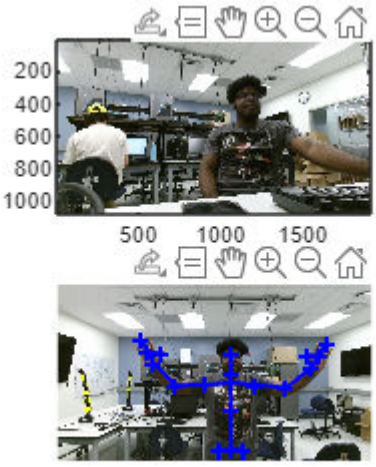
```
jointIndices = 25×2
10^3 ×
    1.0629    1.0112
    1.0610    0.7698
    1.0593    0.5481
    1.0670    0.4251
    0.9091    0.6141
    0.7167    0.6168
    0.5872    0.4803
    0.5552    0.4156
    1.2160    0.6148
    1.4016    0.6321
      ⋮
```

```
% Pull out the 95th color frame
image = frameDataColor(:, :, :, 95);

% Find number of Skeletons tracked
nBodies = length(trackedBodies);
```

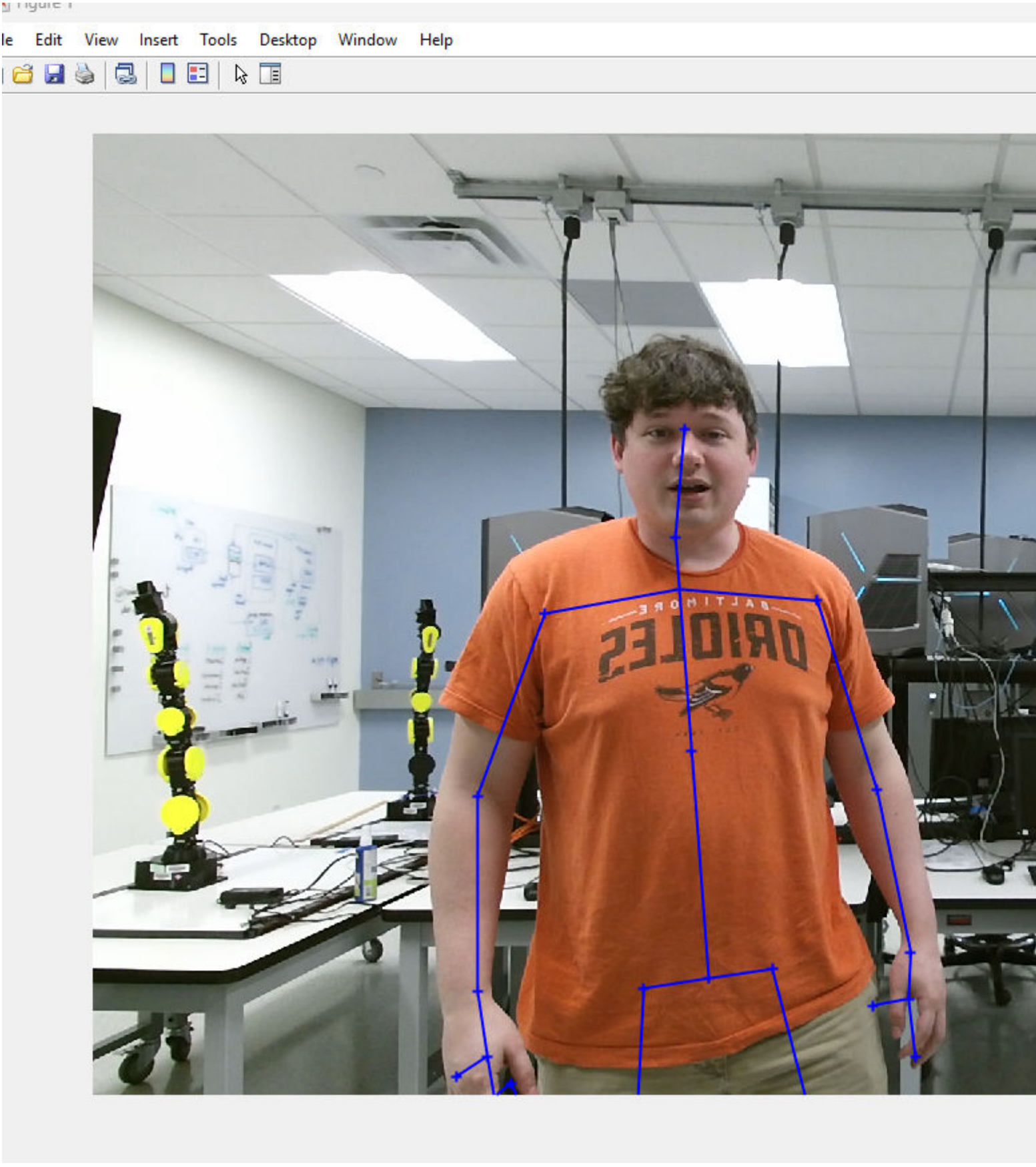
Now plot the skeleton over the RGB image

```
% Create skeleton connection map to link the joints.
SkeletonConnectionMap = [ [4 3];  % Neck
                          [3 21]; % Head
                          [21 2]; % Right Leg
                          [2 1];
                          [21 9];
                          [9 10];  % Hip
                          [10 11];
                          [11 12]; % Left Leg
                          [12 24];
                          [12 25];
```

```
                            [21 5];  % Spine
                            [5 6];
                            [6 7];   % Left Hand
                            [7 8];
                            [8 22];
                            [8 23];
                            [1 17];
                            [17 18];
                            [18 19];  % Right Hand
                            [19 20];
                            [1 13];
                            [13 14];
                            [14 15];
                            [15 16];
                          ];

% Marker colors for up to 6 bodies.
colors = ['b';'r';'g';'c';'y';'m'];

% Display the RGB image.
imshow(image);

% Overlay the skeleton on this RGB frame.
for i = 1:24
     for body = 1:nBodies
         X1 = [jointIndices(SkeletonConnectionMap(i,1),1,body)
jointIndices(SkeletonConnectionMap(i,2),1,body)];
         Y1 = [jointIndices(SkeletonConnectionMap(i,1),2,body)
jointIndices(SkeletonConnectionMap(i,2),2,body)];
         line(X1,Y1, 'LineWidth', 1.5, 'LineStyle', '-', 'Marker', '+', 'Color',
colors(body));
     end

    hold on;
 end
 hold off;
```



Copy and paste a screenshot overlaid with the extracted skeleton:



Tasks/Questions:

- Did the skeleton model accurately capture your body geometry? **Yes, it seems to have properly identified joints.**
- How would you estimate the elbow angle from the skeletal model? **The elbow angle can be estimated by calculating the angle between the shoulder, elbow, and wrist joint as a triangle.**

Submitting This Lab

Export this *.mlx file as Lab7_##_<LastName1>_<LastName2>_<LastName3>_<LastName4>.pdf (## should be your computer number) and email to Lauren.Diaz@jhuapl.edu