دانشکده روانشناسی و علوم تربیتی

MATLAB for Brain and Cognitive Psychology (Stimulus Presentation)

Presented by:
Ehsan Rezayat, Ph.D.
Faculty of Psychology and Education, University of Tehran.
Institute for Research in Fundamental Sciences (IPM), School of Cognitive Sciences,
emails: rezayat@ut.ac.ir, rezayat@ipm.ir, erezayat.er@gmail.com

Today: Steps in the Psychophysics Lab

- Generating Stimuli
- Visual Display
- Stimulus presentation
- Response collection

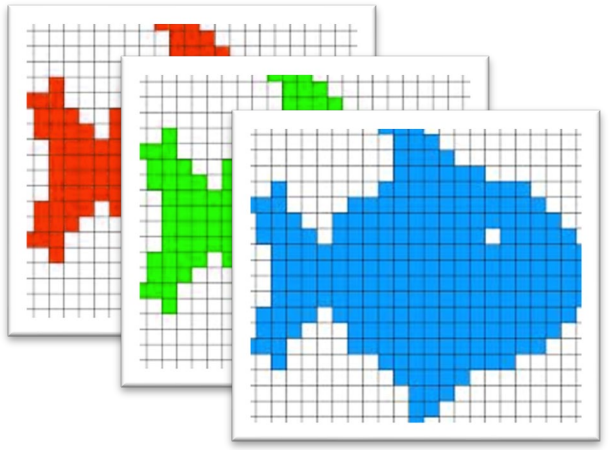
Displaying pictures

- Steps to displaying a picture:
  - 1. Use imread() to read the image into a matrix of numbers
  - 2. Use MakeTexture to create an OpenGL texture using that matrix
  - 3. Use DrawTexture to draw the texture to the screen

Displaying images

```
>> faceData = imread('sadface.jpg');
>> size(faceData);
ans =

   650   506     3

>> faceTexture = Screen('MakeTexture',wPtr,faceData);
>> Screen('DrawTexture',wPtr,faceTexture);
>> Screen('Flip',wPtr);
```

Images



Images

- Step 1: Read in the image data using imread()
- Supported image formats:
  - BMP, GIF, JPEG, PNG, TIFF, etc.

```
A = imread('mypicture.jpg');

[A, map] = imread('mypicture.jpg');

[A, map, alpha] = imread('mypicture.jpg');
```

Displaying images

```
>> faceData = imread('sadface.jpg');
>> size(faceData);
ans =

   650   506     3
```

Images

- Step 1: Read in the image data using imread()
- Step 2: Make a texture

```
myTextureIndex = Screen('MakeTexture',wPtr, imageMatrix)
```

Displaying images

```
>> faceData = imread('sadface.jpg');
>> size(faceData);
ans =

   650   506     3

>> faceTexture = Screen('MakeTexture',wPtr,faceData);
>> Screen('DrawTexture',wPtr,faceTexture);
>> Screen('Flip',wPtr);
```

```
function showPic()

%Open the screen
[wPtr,rect] = Screen('OpenWindow',max(Screen('Screens')));

%Create texture
faceData = imread('sadface.jpg');
faceTexture = Screen('MakeTexture',wPtr,faceData);

%Draw it
Screen('DrawTexture',wPtr,faceTexture);
Screen('Flip',wPtr);

%Wait for keypress and clear
KbWait();
clear Screen;


end
```

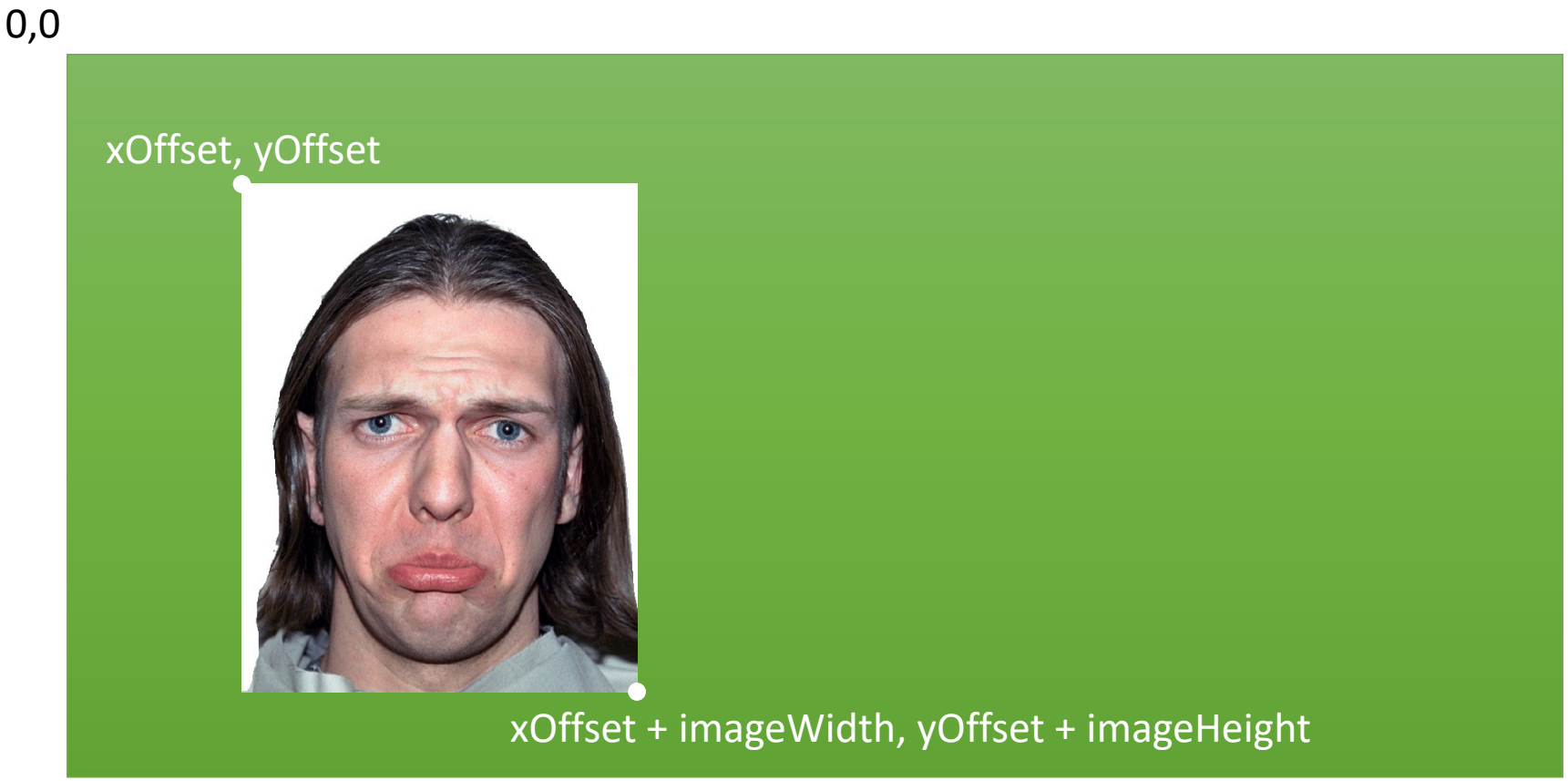
Drawing Images

```
Screen('DrawTexture', windowPointer, texturePointer [,sourceRect]
[,destinationRect] [,rotationAngle] [, filterMode] [, globalAlpha]
[, modulateColor] [, textureShader] [, specialFlags] [, auxParameters]);
```

rect defining subpart of picture to present, default is whole picture

rect defining subpart of screen to present picture in (defaults to center of screen)

Moving images



```
function moveImage

%Open the screen
[wPtr,rect] = Screen('OpenWindow',max(Screen('Screens')));
xCenter = rect(3)/2;
yCenter = rect(4)/2;

%Create texture
faceData = imread('sadface.jpg');
faceTexture = Screen('MakeTexture',wPtr,faceData);

%Get size of image
[imageHeight, imageWidth, colorChannels] = size(faceData);

%Define image rect
imageRect = [0 0 imageWidth imageHeight];

%Draw it
Screen('DrawTexture',wPtr,faceTexture,[],imageRect);
Screen('Flip',wPtr);

%Wait for keypress
KbWait();

%Move the image to 50,100
xOffset = 50;
yOffset = 100;
imageRect = [xOffset, yOffset, xOffset+imageWidth, yOffset+imageHeight];

%Draw new version
Screen('DrawTexture',wPtr,faceTexture,[],imageRect);
Screen('Flip',wPtr);

%Wait for keypress and clear
WaitSecs(2);
KbWait();
clear Screen;

end
```

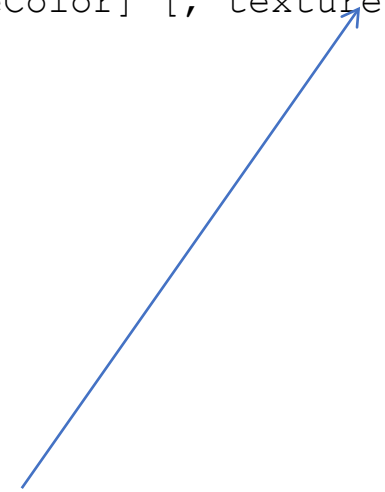
Scaling images

- To scale an image, change the size of the destination rectangle

```
function scaleImage

%Open the screen
[wPtr,rect] = Screen('OpenWindow',max(Screen('Screens')));
xCenter = rect(3)/2;
yCenter = rect(4)/2;

%Create texture
faceData = imread('sadface.jpg');
faceTexture = Screen('MakeTexture',wPtr,faceData);

%Get size of image
[imageHeight, imageWidth, colorChannels] = size(faceData)

%Define image rect
imageRect = [0 0 imageWidth imageHeight];

%Center it
destinationRect = CenterRect(imageRect,rect);

%Draw it
Screen('DrawTexture',wPtr,faceTexture,[],destinationRect);
Screen('Flip',wPtr);

%Wait for keypress
KbWait();

%Shrink by a factor of 2
imageRect = imageRect./2;
destinationRect = CenterRect(imageRect,rect);

%Draw shrunken version
Screen('DrawTexture',wPtr,faceTexture,[],destinationRect);
Screen('Flip',wPtr);

%Wait for keypress and clear
WaitSecs(2);
KbWait();
clear Screen;


end
```

Rotating images

```
Screen('DrawTexture', windowPointer, texturePointer [,sourceRect]
[,destinationRect] [,rotationAngle] [, filterMode] [, globalAlpha]
[, modulateColor] [, textureShader] [, specialFlags] [, auxParameters]);
```



set rotation angle.
upright image is 0
degrees

```
function rotateImage

%Open the screen
[wPtr,rect] = Screen('OpenWindow',max(Screen('Screens')));

%Create texture
faceData = imread('sadface.jpg');
faceTexture = Screen('MakeTexture',wPtr,faceData);

%Draw it
Screen('DrawTexture',wPtr,faceTexture);
Screen('Flip',wPtr);

%Wait for keypress and clear
KbWait();

angle = 0;

start = GetSecs();
duration = 30;

while GetSecs < start + duration
    Screen('DrawTexture',wPtr,faceTexture,[],[],angle);
    Screen('Flip',wPtr);
    angle = angle + 1;
end

clear Screen;

end
```

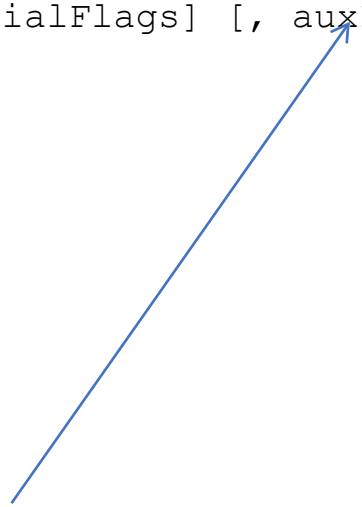
Multiple images

- You can draw multiple image textures to the back buffer, and then flip to show them at the same time

```
function twoPics

%Open the screen
[wPtr,rect] = Screen('OpenWindow',max(Screen('Screens')));
xCenter = rect(3)/2;
yCenter = rect(4)/2;

%Create textures
sadFaceData = imread('sadface.jpg');
sadFaceTexture = Screen('MakeTexture',wPtr,sadFaceData);

angryFaceData = imread('angryface.jpg');
angryFaceTexture = Screen('MakeTexture',wPtr,angryFaceData);

%Get size of image (both images are the same size in this example)
[imageHeight, imageWidth, colorChannels] = size(sadFaceData);

%Define upper left hand corner of image rect
distanceFromCenter = 50;
sadX = xCenter - imageWidth - distanceFromCenter;
sadY = yCenter - imageHeight/2;
angryX = xCenter + distanceFromCenter;
angryY = yCenter - imageHeight/2;

%Define destination rects
sadRect = [sadX, sadY, sadX+imageWidth, sadY+imageHeight]
angryRect = [angryX, angryY, angryX+imageWidth, angryY+imageHeight];

%Draw them
Screen('DrawTexture',wPtr,sadFaceTexture,[],sadRect);
Screen('DrawTexture',wPtr,angryFaceTexture,[],angryRect);
Screen('Flip',wPtr);

%Wait for keypress and clear
KbWait();
clear Screen;

end
```

Transparency

```
Screen('DrawTexture', windowPointer, texturePointer [,sourceRect]
[,destinationRect] [,rotationAngle] [, filterMode] [, globalAlpha]
[, modulateColor] [, textureShader] [, specialFlags] [, auxParameters]);
```



set alpha for image, alpha blending must be on
0 = fully transparent
1 = fully solid

Alpha blending

```
>> Screen BlendFunction?

>> Screen('BlendFunction',wPtr,GL_SRC_ALPHA,GL_ONE_MINUS_SRC_ALPHA);  ENABLE BLENDING

>> Screen('BlendFunction',wPtr,GL_ONE,GL_ZERO);  DISABLE BLENDING
```

Displaying images

- Loading images in and making textures can take time
- Don't wait until you want to present the images to load them in
- Make your textures at the beginning of your script, then present them on time

Movies

http://www.gstreamer.com

Movies

OSX: Must have XCode installed

Make sure to press "customize" and check all the boxes in the installer to install all the parts

Movies

- 1. OpenMovie to open the movie file
- 2. PlayMovie to start playing
- 3. Loop:
  - GetMovieImage to create frame texture
  - Draw texture and flip screen
- 4. PlayMovie to stop playing
- 5. CloseMovie to close movie file

Movies

```
[ moviePtr [duration] [fps] [width] [height] [count [aspectRatio]]=Screen('OpenMovie',
windowPtr, moviefile [, async=0] [, preloadSecs=1] [, specialFlags1=0][, pixelFormat=4]
[, maxNumberThreads=-1]);
```

Movies

```
Screen('PlayMovie', moviePtr, rate, [loop], [soundvolume]);
```

0 = stop playback
1 = play, normal speed
-1 = play, normal speed backwards

0 = mute
1 = max volume

Movies

```
[ texturePtr [timeindex]]=Screen('GetMovieImage', windowPtr,
moviePtr, [waitForImage=1], [fortimeindex], [specialFlags = 0]
[, specialFlags2 = 0]);
```

```
function playMovie()

%Set the movie path and filename
pathToMovie= [pwd,'/rooster.mov'];

%Set clip info
toTime=inf;     % second to stop in movie
soundvolume=1;  % 0 to 1

%Open the screen
[w,rect]=Screen('OpenWindow', max(Screen('screens')), 0);

%Open the movie
[movie,dur,fps,width,height]=Screen('OpenMovie', w, pathToMovie);

%Play the movie
Screen('PlayMovie', movie, 1, 0, soundvolume);

%Mark starting time
t = GetSecs();

%loop through each frame of the movie and present it
while t < toTime

    %get the texture
    tex = Screen('GetMovieImage', w, movie);

    %if there is no texture, we are at the end of the movie
    if tex<=0
        break;
    end

    %draw the texture
    Screen('DrawTexture', w, tex);
    t=Screen('Flip', w);

    %discard this texture
    Screen('Close',tex);
end

%Stop the movie
Screen('PlayMovie', movie, 0);

%Close the movie
Screen('CloseMovie', movie);

%Clear the screen
clear Screen;

end
```

Steps to playing a sound

- InitializePsychSound
- open audio channel with PsychPortAudio('Open')
- fill audio buffer with PsychPortAudio('FillBuffer')
- start playing a sound with PsychPortAudio('Start')
- stop playing a sound with PsychPortAurio('Stop')
- close the audio channel with PsychPortAudio('Close')

Step 1: Intialize

- InitializePsychSound
  - Loads the sound driver. Place this at the beginning of your script.
  - on Windows, things may not work with high precision timing without an ASIO sound card (read help InitializePsychSound if you are on Windows)

Step 2: Open audio channel

```
pahandle = PsychPortAudio('Open' [, deviceid][, mode]
[, reqlatencyclass][, freq][, channels] [, buffersize]
[, suggestedLatency][, selectchannels][, specialFlags=0]);
```

how aggressively to take over the sound device in order to assure latency
default is 1. Higher numbers give better latency but have consequences.

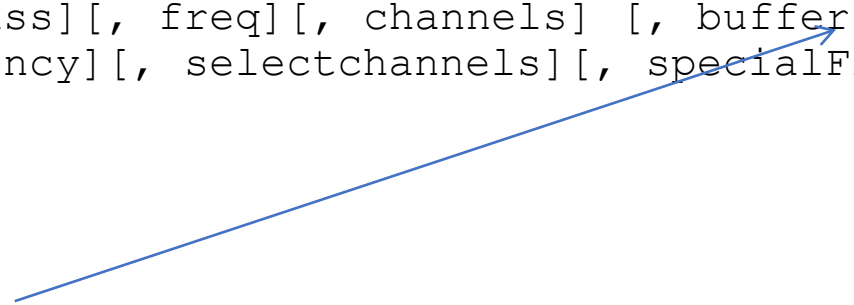
requested playback rate in Hz

playback channels:
1 = mono
2 = stereo
etc.
default is 2

Which audio device to use for playback.
PsychPortAudio('GetDevices') to list all available devices

Step 2: Open audio channel

```
pahandle = PsychPortAudio('Open' [, deviceid][, mode]
[, reqlatencyclass][, freq][, channels] [, buffersize]
[, suggestedLatency][, selectchannels][, specialFlags=0]);
```



1: sound playback only (default)
2: audio capture
3: simultaneous capture and playback (may not work on all hardware)

Step 3: Fill the audio buffer

```
PsychPortAudio('FillBuffer', pahandle, bufferdata);
```

This is analogous to drawing on the back buffer with the Screen command. We fill the buffer now, but it will not be heard until we play it.

Step 4: Start playback

```
startTime = PsychPortAudio('Start', pahandle [, repetitions=1]
[, when=0] [, waitForStart=0] [, stopTime=inf] [, resume=0]);
```

Wait until this time to start playing (default is play now)

0: Ask playback to start and move on
1: wait for playback to actually begin.
A 1 here is necessary if you want to get timing info back

set a time to stop playing

Set to 0 to repeat indefinitely

Remaining steps

- Stop playback if necessary: PsychPortAudio('Stop',pahandle);
- Close the audio driver: PsychPortAudio('Close',pahandle);

```
function playSound()

%Mark script start time
scriptStart = GetSecs();

%Initialize the sound driver
InitializePsychSound;

%Read in the sound data from file
wavfilename = [ PsychtoolboxRoot 'PsychDemos' filesep 'SoundFiles' filesep 'funk.wav'];
[soundData, freq] = wavread(wavfilename);

%Prepare sound data (make it two rows for stereo playback)
soundData = soundData';
soundData = [soundData; soundData];
numChannels = 2;

%Open the audio driver
pahandle = PsychPortAudio('Open', [], [], 0, freq, numChannels);

%Fill the buffer
PsychPortAudio('FillBuffer',pahandle,soundData);

%Play the sound
playTime = PsychPortAudio('Start',pahandle,[],[],1);

fprintf('Sound started playing %.2f seconds after start of script\n',playTime-scriptStart);

%Close the audio driver
PsychPortAudio('Close',pahandle);

end
```

Sound recording

- Also done through PsychPortAudio
- See BasicSoundInputDemo.m

Sound recording steps

- Initialize sound driver: InitializePsychAudio
- Open audio channel for recording with PsychPortAudio('Open') setting mode to 2
- Clear a buffer using PsychPortAudio('GetAudioData')
- Start recording with PsychPortAudio('Start')
- Stop recording with PsychPortAudio('Stop')
- Get audio data using PsychPortAudio('GetAudioData')

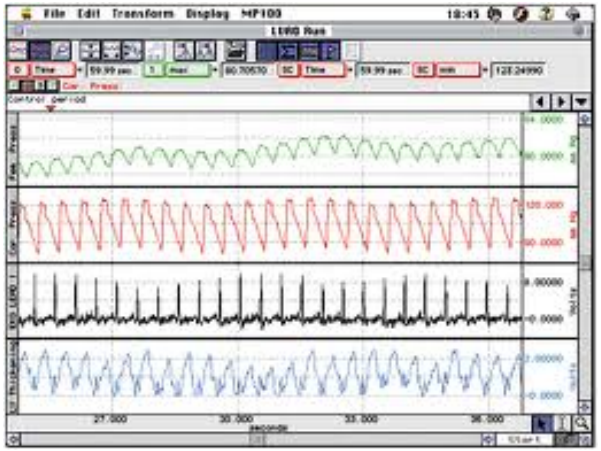
```
function audioData = recordSound

%Initialize sound driver
InitializePsychSound;
duration = 5;

%Open audio channel for recording using mode 2
freq = 44100;
pahandle = PsychPortAudio('Open', [], 2, 0, freq, 2);

%Set up buffer for recording
PsychPortAudio('GetAudioData', pahandle, duration);

%Start recording
PsychPortAudio('Start', pahandle, 0, 0, 1);

%Go until keypress
fprintf('Recording...\n');
WaitSecs(duration);
fprintf('Done recording.\n');

%Stop Recording
PsychPortAudio('Stop', pahandle);

%Get the audio data we recorded
audioData  = PsychPortAudio('GetAudioData',pahandle);

%Close the audio channel
PsychPortAudio('Close',pahandle);

end
```

DAQ toolbox

- DAQ = Data Acquisition device
- For communicating with the USB-1208FS from Measurement Computing
- Allows input and out of digital and analog signals

Using the DAQ to synchronize

external measurement system

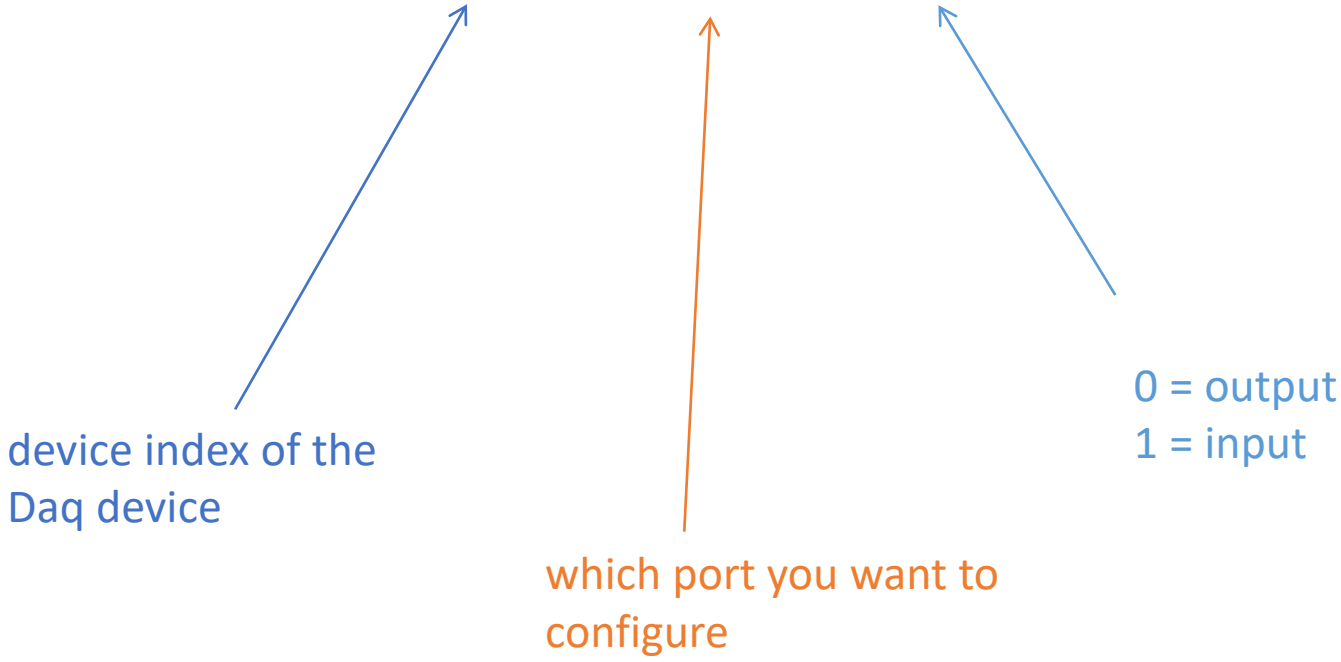


Sending output with the DAQ

- 1. Identify the DAQ device in the PsychHID device list
- 2. Initialize the DAQ device with DaqDConfigPort()
- 3. Send output with DaqDOut()

Initializing a port

```
DaqDConfigPort(DeviceIndex, port, direction)
```



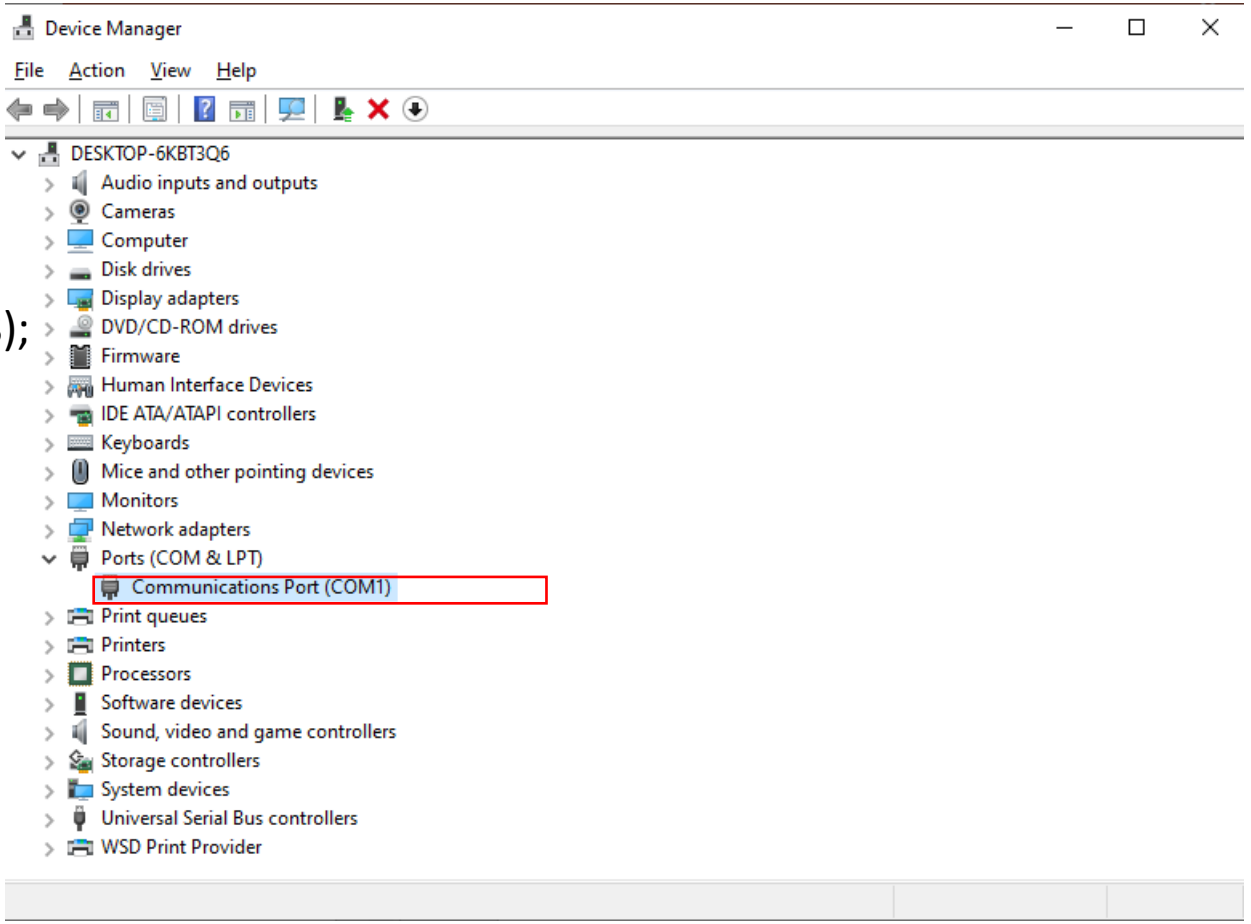
Sending output

```
DaqDOut(DeviceIndex, port, data)
```

value you want to send to
the output channel

Serial USB port

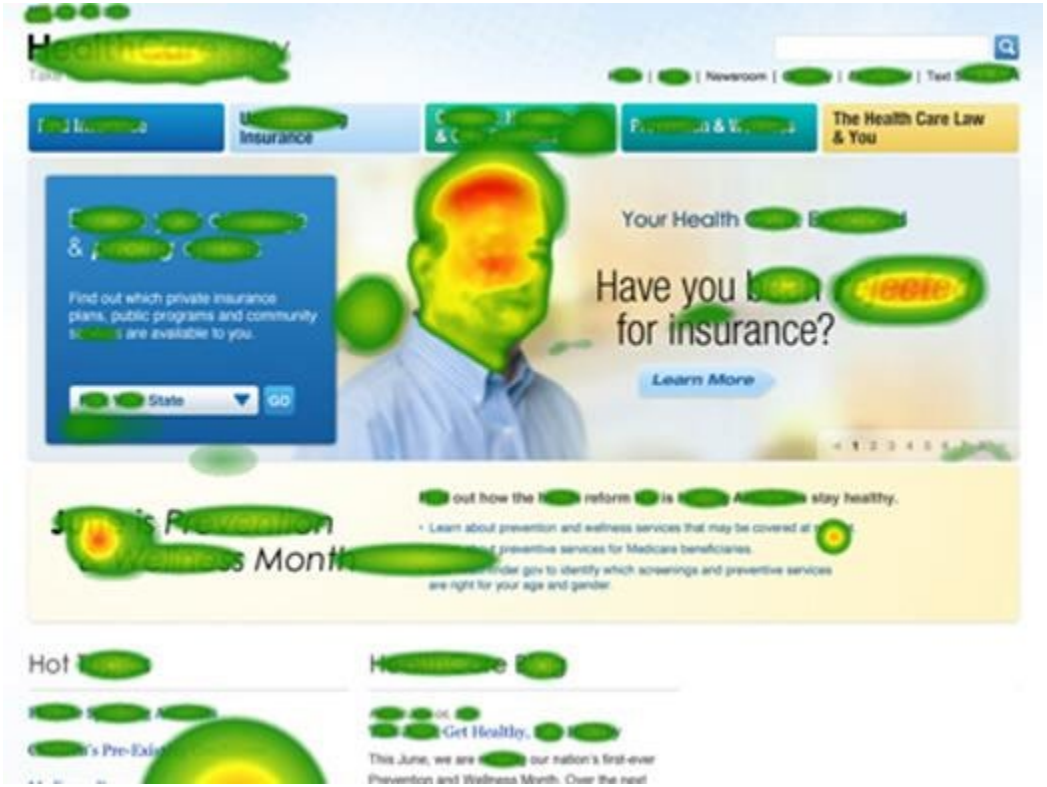
[handle, errmsg] = IOPort('OpenSerialPort', port [, configString]);




```
%% IO Port Initialization
porta = serial('COM1', 'BaudRate', 57600, 'DataBits', 8);

fopen(porta);



% when you  want to send event
 eve = 1;    % a number in [0 :255]
 fwrite(porta,eve);
```

Eye tracking

- Eyelink 1000- 2000
- Tobi 300 Hz
- Zistkankash toos
- Farmed tajhiz



Assignment # 8

Write a function called yourInitials_session8()

The function should:

- Draw a image in the center of the screen. Wait for the user to press a key. Then clear the image from Screen and wait for the user to press a key. Then show the first image with another image (distractor) on screen . Wait for the user to press a key. Repeat this trial for 5 time. the location of image and distractor image should be defined randomly for each trials.