دانشکده روانشناسی و علوم تربیتی

پژوهشگاه دانش‌های بنیادی

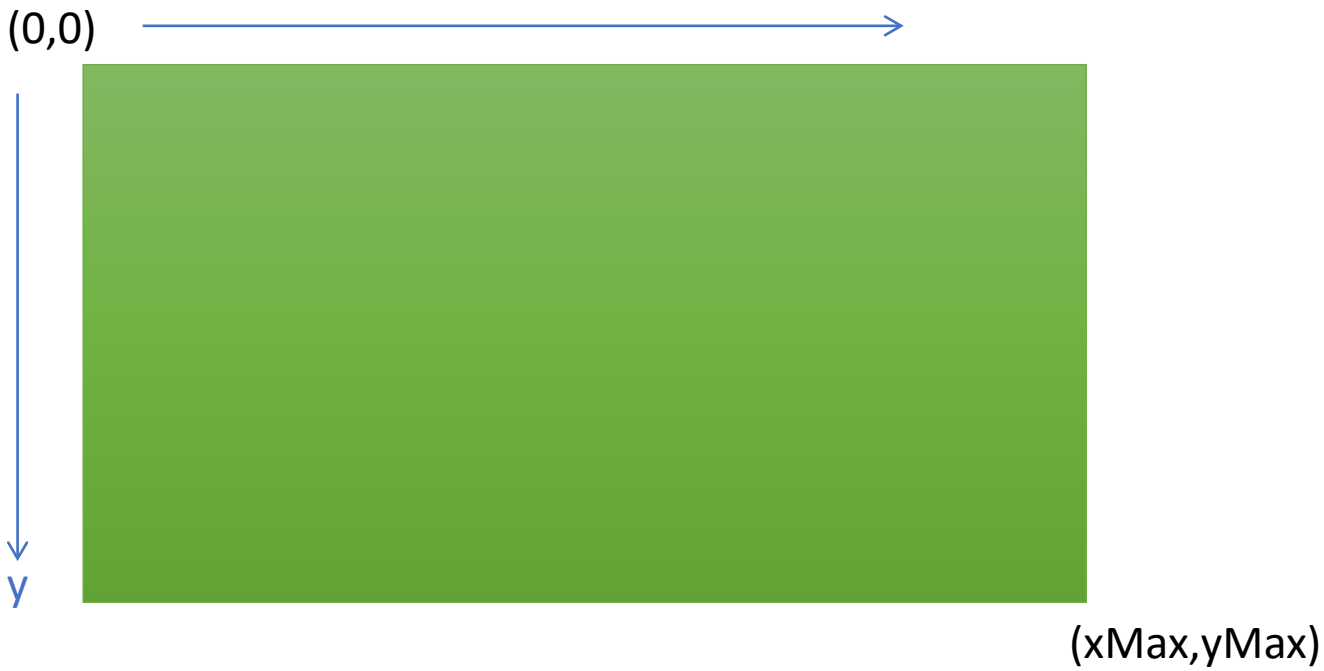
MATLAB for Brain and Cognitive Psychology (Presentation stimuli)

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 (today)
- Stimulus presentation
- Visual Display
- Response collection

Screen coordinate system



Drawing simple shapes

```
>> Screen FillRect?

Usage:

Screen('FillRect', windowPtr [,color] [,rect] )

Fill "rect". "color" is the clut index (scalar or [r g b] triplet or [r g b a]
quadruple) that you want to poke into each pixel;  default produces white with
the standard CLUT for this window's pixelSize. Default "rect" is entire window,
so you can use this function to clear the window. Please note that clearing the
entire window will set the background color of the window to the clear color,
ie., future Screen('Flip') commands will clear to the new background clear color
specified in Screen('FillRect').
Instead of filling one rectangle, you can also specify a list of multiple
rectangles to be filled - this is much faster when you need to draw many
rectangles per frame. To fill n rectangles, provide "rect" as a 4 rows by n
columns matrix, each column specifying one rectangle, e.g., rect(1,5)=left
border of 5th rectange, rect(2,5)=top border of 5th rectangle, rect(3,5)=right
border of 5th rectangle, rect(4,5)=bottom border of 5th rectangle. If the
rectangles should have different colors, then provide "color" as a 3 or 4 row by
n column matrix, the i'th column specifiying the color of the i'th rectangle.

See also: FrameRect
```

Drawing simple shapes

```
Screen('FillRect', wPtr, color, rect);
```

Define the rectangle(s) to fill in.
If you leave this blank, the whole screen will be filled, and the background color will be set to that color (when you Flip the screen, the buffer will clear to this color)

OpenWindow expanded

```
[windowPtr, rect] = Screen('OpenWindow',whichWindow,bgColor,rect)
```

Set the screen's background color (default is white)

Define a rect for the screen to appear in (default is to take up the whole physical screen)
Note: timing may suffer if you are not using full screen

```
function drawSomething()

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

Screen('FillRect', wPtr, [255 0 0],[100 100 500 500]);      %draw a rectangle on the back buffer
Screen('Flip',wPtr);                                        %flip the buffers, showing the rectangle
WaitSecs(10);

Screen('Flip',wPtr);                                        %flip the buffers, clearing the screen
WaitSecs(10);

Screen('FillRect', wPtr, [255 0 0],[100 100 500 500]);      %draw a rectangle on the back buffer
Screen('Flip',wPtr);                                        %flip the buffers, showing the rectangle again
WaitSecs(10);

clear Screen;

end
```

Drawing multiple rects

```
>> rectOne = [100 100 250 400];
>> rectTwo = [250 400 300 450];
>> bothRects = [rectOne', rectTwo']

bothRects =

   100   250
   100   400
   250   300
   400   450

>> Screen('FillRect',w,[0 255 0],bothRects);
>> Screen('Flip',w)
```

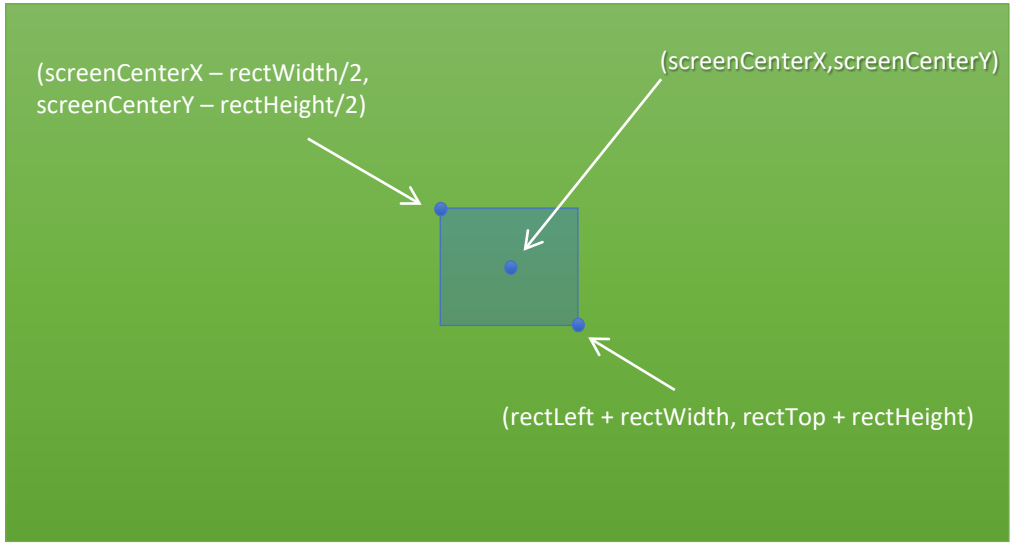
Centering

- Problem: Draw a 100 by 100 square exactly centered in the screen
  - First step: find the center of the screen

Centering

```
>> [wPtr,rect] = Screen('OpenWindow',1)
wPtr =

    11


rect =

           0           0        1680        1050

>> screenWidth = rect(3);
>> screenHeight = rect(4);
>> screenCenterX = screenWidth/2;
>> screenCenterY = screenHeight/2;
>>
>> myRectWidth = 100;
>> myRectHeight = 100;
>>
>> myRectLeft = screenCenterX - myRectWidth/2;
>> myRectTop = screenCenterY - myRectHeight/2;
>> myRectRight = myRectLeft + myRectWidth;
>> myRectBottom = myRectTop + myRectHeight;
>> myRect = [myRectLeft, myRectTop, myRectRight, myRectLeft];
>>
>> Screen('FillRect',wPtr,[0 0 255],myRect);
>> Screen('Flip',wPtr);
```

Centering


(screenCenterX – rectWidth/2,
screenCenterY – rectHeight/2)
(screenCenterX,screenCenterY)
(rectLeft + rectWidth, rectTop + rectHeight)

Drawing functions

```
Screen('DrawLine', windowPtr [,color], fromH, fromV, toH, toV [,penWidth]);
Screen('DrawArc',windowPtr,[color],[rect],startAngle,arcAngle)
Screen('FrameArc',windowPtr,[color],[rect],startAngle,arcAngle[,penWidth] [,penHeight] [,penMode])
Screen('FillArc',windowPtr,[color],[rect],startAngle,arcAngle)
Screen('FillRect', windowPtr [,color] [,rect] );
Screen('FrameRect', windowPtr [,color] [,rect] [,penWidth]);
Screen('FillOval', windowPtr [,color] [,rect] [,perfectUpToMaxDiameter]);
Screen('FrameOval', windowPtr [,color] [,rect] [,penWidth] [,penHeight] [,penMode]);
Screen('FramePoly', windowPtr [,color], pointList [,penWidth]);
Screen('FillPoly', windowPtr [,color], pointList [, isConvex]);
Screen('DrawDots', windowPtr, xy [,size] [,color] [,center] [,dot_type]);
Screen('DrawLines', windowPtr, xy [,width] [,colors] [,center] [,smooth]);
```

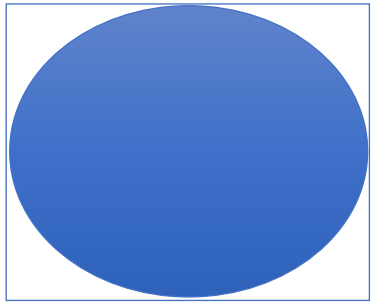
Drawing

```
Screen('FrameRect', wPtr, color, rect, penWidth);
```

Just like FillRect, but the rectangle is not filled in. Change penWidth to change the thickness of the lines

Drawing circles

```
Screen('FillOval', wPtr, color, rect);
```



Drawing circles

```
>> Screen('FillRect',wPtr,[0 0 255],myRect)
>> Screen('FillOval',wPtr,[255 0 0 ],myRect)
>> Screen('Flip',wPtr);
```

Note drawing order! As we add to the screen, new shapes are added on top of old ones.

Alpha blending

```
>> Screen BlendFunction?
>> Screen('BlendFunction',wPtr,GL_SRC_ALPHA,GL_ONE_MINUS_SRC_ALPHA);
>> Screen('BlendFunction',wPtr,GL_ONE,GL_ZERO);
```

ENABLE BLENDING

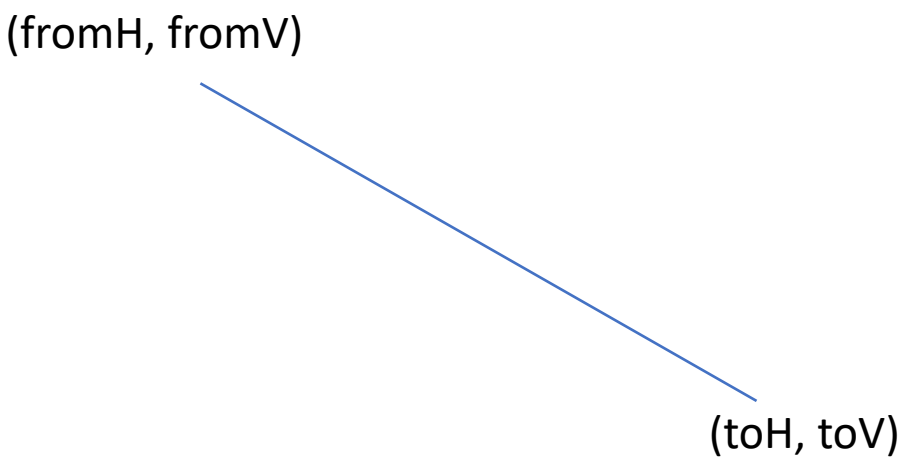
DISABLE BLENDING

Alpha blending

```
function blendShapes()

%Open the screen
[wPtr, rect] = Screen('OpenWindow',1);

%Turn on alpha blending
Screen('BlendFunction',wPtr,GL_SRC_ALPHA,GL_ONE_MINUS_SRC_ALPHA);

%Define colors
color1 = [0 255 0 255];
color2 = [0 0 255 100];

%Draw Shapes
Screen('FillRect',wPtr,color1,[300 300 400 400]);
Screen('FillRect',wPtr,color2,[350 350 450 450]);

%Show them
Screen('Flip',wPtr);

KbWait();
clear Screen;

end
```

Drawing lines

```
Screen('DrawLine', wPtr, color, fromH, fromV, toH, toV, penWidth);
```



Using DrawDots

- Alternate method for drawing several shapes at once. Even though it is called DrawDots, it will draw squares as well as circles.

Screen('DrawDots', windowPtr, xy [,size] [,color] [,center] [,dot_type]);

Matrix containing coordinates of dot centers. Two rows, x and y. Each column is another dot.

Can either be a single value, or a vector of values, one per dot

Single color vector, or matrix with three rows (r,g,b)

Define center coordinates

0 (default) = squares
1 = circles
2 = circles with high quality anti-aliasing

```
>> [wPtr, rect] = Screen('OpenWindow',1);
>> xCenter = rect(3)/2;
>> yCenter = rect(4)/2;
>>
>> colors = [255 0 0; 0 255 0; 0 0 255]
colors =

   255     0     0
     0   255     0
     0     0   255
>> locations = [-100 0 100; 0 0 0]
locations =

  -100     0   100
     0     0     0
>> sizes = [30 40 50];
>> Screen('DrawDots',wPtr,locations, sizes, colors, [xCenter,yCenter], 1);
>> Screen('Flip',wPtr);
```

Using DrawLines

Screen('DrawLines', windowPtr, xy [,width] [,colors] [,center] [,smooth]);

Matrix containing coordinates. Two rows (x and y). Each pair of columns (start and end) specifies a line.

Either a single color, or one color per point in xy. Three rows (r,g,b).

0 (default) = no smoothing
1 = smoothing
2 = high quality smoothing
NOTE: smoothing requires blending to be on

```
>> lines = [-300 300 -300 300; -50 -50 50 50]
lines =

  -300   300  -300   300
   -50   -50    50    50

>> colors = [255 0 0 255; 0 0 0 0 ; 0 255 255 0]
colors =

   255     0     0   255
     0     0     0     0
     0   255   255     0

>> Screen('DrawLines',wPtr,lines,10,colors,[xCenter,yCenter],0)
>> Screen('Flip',wPtr)
```

Drawing a fixation cross

```
function drawCross()

%Define cross characteristics
crossLength = 10;
crossColor = 0;
crossWidth = 3;

%Set start and end points of lines
crossLines = [-crossLength, 0; crossLength, 0; 0, -crossLength; 0, crossLength];
crossLines = crossLines';

%Open the screen
[wPtr, rect] = Screen('OpenWindow',1);

%Define center of Screen
xCenter = rect(3)/2;
yCenter = rect(4)/2;

%Draw the lines
Screen('DrawLines',wPtr,crossLines,crossWidth,crossColor,[xCenter,yCenter]);
Screen('Flip',wPtr);

%Wait for keypress and clear screen
KbWait;
clear Screen;

end
```

Drawing a fixation cross

```
function drawFixationCross(wPtr,rect,crossLength,crossColor,crossWidth)
%
% Draws a fixation cross at the center of the screen.
%
% drawFixationCross(wPtr,rect,crossLength,crossColor,crossWidth)
%
%

%Set start and end points of lines
crossLines = [-crossLength, 0; crossLength, 0; 0, -crossLength; 0, crossLength];
crossLines = crossLines';

%Define center of Screen
xCenter = rect(3)/2;
yCenter = rect(4)/2;

%Draw the lines
Screen('DrawLines',wPtr,crossLines,crossWidth,crossColor,[xCenter,yCenter]);


end
```

Drawing a fixation cross

```
>> crossLength = 10;
>> crossWidth = 10;
>> crossColor = 0;

>> [wPtr, rect] = Screen('OpenWindow',1);
>> drawFixationCross(wPtr,rect,crossLength,crossColor,crossWidth);
>> fixationTime = Screen('Flip',wPtr);
```

Animation

- Create a loop where something changes each time through the loop

```
function colorLoop()

%set some starting color values
red = 0;
green = 0;
blue = 0;

%set the dimensions of our square
square = [100 100 400 400];

%open the screen
[w,rect] = Screen('OpenWindow',1,0);

%record the time we are starting
startTime = GetSecs();

%Continue this loop until 30 seconds have passed
while GetSecs() < startTime + 30

    %draw the square
    Screen('FillRect',w,[red blue green],square);
    Screen('Flip',w);

    %increment red
    red = red + 1;

    %if red is too high, reset it to 0
    if red > 255
        red = 0;
    end

end

clear Screen;

end
```

Drawing text

- Two steps to drawing text:
  - 1. Set up all the properties of the text we want to draw (font, size, style) using separate commands
  - 2. Draw the text using DrawText

Drawing Text

```
Screen('TextSize',wPtr,size);
Screen('TextFont',wPtr,fontString);
Screen('TextStyle',wPtr,style);
```

0 = normal
1 = bold
2 = italic
4 = underline
8 = outline
32 = condense
64 = extend

Drawing Text

```
Screen('DrawText',wPtr,text,x,y,color)
```

Drawing Text

```
>> [wPtr, rect] = Screen('OpenWindow',1);
>> Screen('TextFont',wPtr,'Helvetica');
>> Screen('TextSize',wPtr,48);
>> Screen('DrawText','Hello there',100,100,[200 0 0]);
```

Drawing Text

Sometimes, to position text, we need to know its size in pixels:

```
rect = Screen('TextBounds',wPtr,textString);
```

Drawing Formatted Text

```
DrawFormattedText(wPtr,textString,sx,sy,color,wrapat,flipH
orizontal,flipVertical, vSpacing, rightoleft, winRect)
```

Advantages over DrawText:

Helpful for splitting text into multiple lines. Can include newline characters in the text string (\n).

Can do automatic centering if you set sx to "center" or right justify if you set to "right"

Drawing Text

```
>> [wPtr, rect] = Screen('OpenWindow',1);
>> myText = 'The experiment\nIs about to begin';
>> DrawFormattedText(wPtr,myText,'center',rect(4)/2,0);
```

Assignment Session# 7

Write a function called yourInitials_session7()

The function should take two inputs:

- An integer called **speed**
- An integer called **radius**

The function should:

- Draw a circle in the center of the screen with radius equal to **radius**. Wait for the user to press a key. The circle will then start moving diagonally towards the lower right hand corner of the screen. The speed at which it moves will be determined by **speed**: **speed** is actually the number of pixels that the circle will move each frame. In other words, increment both the x and y position of the circle by speed every frame.
- If the circle should "bounce" off the edge of the screen. That means that once it hits the bottom of the screen, it will start to move up instead of down, and when it hits the right side of the screen it will start to move left instead of right, etc.
- The color of the circle should randomly change whenever it hits a wall
- The circle will continue to bounce around the screen indefinitely