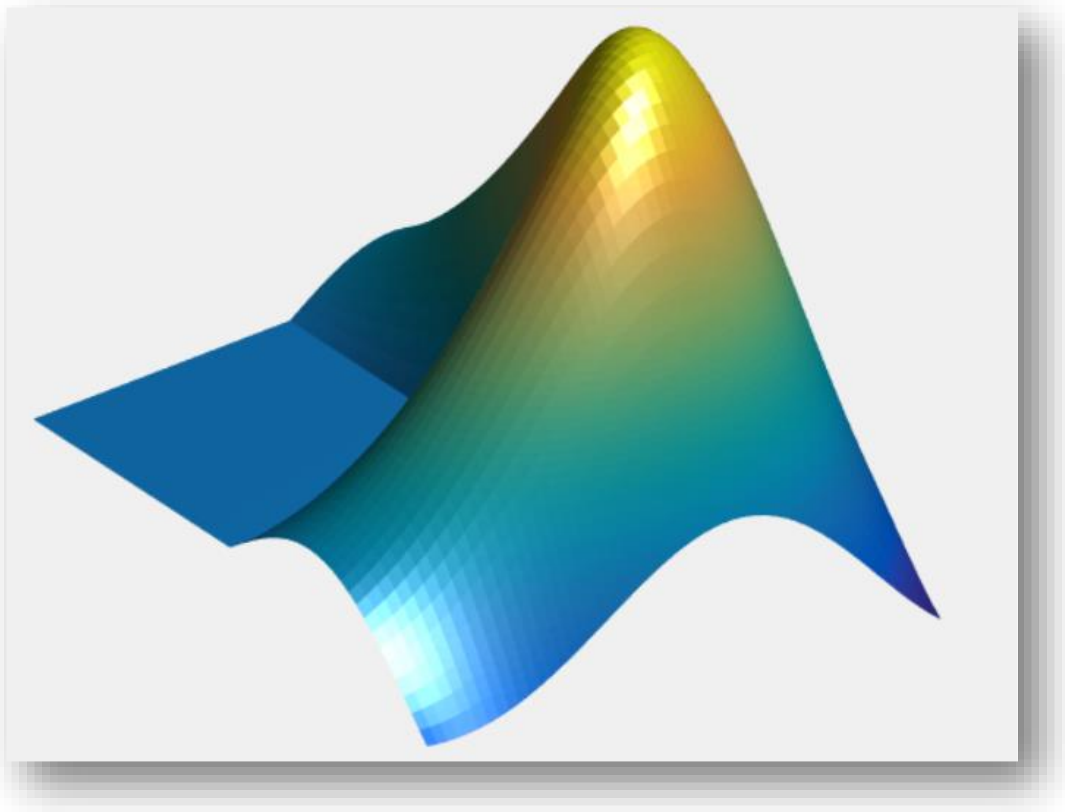
PsychToolbox: Part II



NSC 546 Computing for Neuroscience

Showing and Hiding mouse

```
Nscreen = 0;  [wPtr, rect] = Screen('OpenWindow', Nscreen);

white = WhiteIndex(wPtr);   Screen('FillRect', wPtr, white);
black = BlackIndex(wPtr);   Screen('FillRect', wPtr, black, [100 100 500 500]);
Screen(wPtr, 'Flip');

% hide the mouse cursor
HideCursor(wPtr);
pause(1);

% show the mouse cursor again
ShowCursor(wPtr);
pause(1);

Screen('Close', wPtr);
```

PutImage to the Screen

```
Nscreen = 0; [wPtr, rect] = Screen('OpenWindow', Nscreen);

black = BlackIndex(wPtr);            white = WhiteIndex(wPtr);
gray = (white+black)/2;              inc = white-gray;
Screen('FillRect', wPtr, gray);      Screen('Flip', wPtr);

% create a gabor patch
[x,y] = meshgrid(-200:200, -200:200);
m = exp(-((x/100).^2)-((y/100).^2)) .* sin(0.03*2*pi*x);

I = gray+inc*m;
Screen('PutImage', wPtr, I);

Screen('Flip', wPtr);
Screen('Close', wPtr);
```

Better using MakeTexture, DrawTexture

```
Nscreen = 0; [wPtr, rect] = Screen('OpenWindow', Nscreen);

black = BlackIndex(wPtr);          white = WhiteIndex(wPtr);
gray = (white+black)/2;            inc = white-gray;
Screen('FillRect', wPtr, gray);    Screen('Flip', wPtr);

[x,y] = meshgrid(-200:200, -200:200);
m = exp(-((x/100).^2)-((y/100).^2)) .* sin(0.03*2*pi*x);

w = Screen('MakeTexture', wPtr, gray+inc*m);
Screen('DrawTexture', wPtr, w);

Screen('Flip', wPtr);
Screen('Close', wPtr);
```

Textures are OpenGL objects. Creating them ahead of time means that transferring to the framebuffer is very quick; suitable for creating animations (movies), where speed is especially important.

Better using MakeTexture, DrawTexture

```
I = imread('Grayimg.jpg;');

Nscreen=0;                  [wPtr,rect]=Screen('OpenWindow',Nscreen);
HideCursor;
black=BlackIndex(wPtr);     Screen('FillRect',wPtr,black);
Screen(wPtr, 'Flip');

textureIndex = Screen('MakeTexture', wPtr, I);
Screen('DrawTexture', wPtr, textureIndex);
Screen(wPtr, 'Flip');

ShowCursor;
Screen('Close', wPtr);
```

Drawing Shapes to the Screen

```
Nscreen=0;  [wPtr,rect]=Screen('OpenWindow',Nscreen);
Screen('DrawLine', wPtr, [255 0 0],  100, 100, 200, 200, 10);
Screen('DrawArc',  wPtr, [0 255 0], [200 100 300 200], 45, 45);
Screen('FrameArc', wPtr, [0 0 255], [300 100 400 200], 45, 45, 10);
Screen('FillArc',  wPtr, [255 255 0], [400 100 500 200], 45, 45);
Screen('FillRect', wPtr, [255 0 255], [500 100 600 200]);
Screen('FrameRect', wPtr, [0 255 255], [600 100 700 200], 10);
Screen('FillOval',  wPtr, [100 200 50], [700 125 800 175], 25);
Screen('FrameOval', wPtr, [50 200 100], [800 125 900 175], 10);

pointlist = [200 300; 300 400; 400 300; 300 800];
Screen('FramePoly', wPtr, [200 50 100], pointlist, 10);

pointlist = [500 300; 600 400; 650 450; 500 700];
Screen('FillPoly',  wPtr, [50 100 200], pointlist, 1);

Screen(wPtr, 'Flip'); Screen('Close', wPtr);
```

Text to the Screen

```
Nscreen=0;  [wPtr,rect]=Screen('OpenWindow',Nscreen);

% set text size in pixels
Screen('TextSize', wPtr, 200);

% draw text to screen at a particular location and color
Screen('DrawText', wPtr, 'Test at 200', 200, 200, [255 0 0]);

Screen('TextSize', wPtr, 50);
Screen('DrawText', wPtr, 'Test at 50', 800, 800, [0 0 255]);

Screen(wPtr, 'Flip');

Screen('Close', wPtr);
```

Offscreen Window

```
Nscreen=0;   [wPtr,rect]=Screen('OpenWindow',Nscreen);

[wOff1,rect1]=Screen('OpenOffscreenWindow',wPtr);

Screen('DrawText', wOff1, 'Test Text', 200, 800, [0 0 255]);
Screen('DrawLine', wOff1, [0 255 0], 100, 100, 900, 200, 10);

Screen('CopyWindow', wOff1, wPtr);
Screen(wPtr, 'Flip');
pause(2);

Screen('Close', wPtr);
```

Offscreen Window

```
Nscreen=0;   [wPtr,rect]=Screen('OpenWindow',Nscreen);

[wOff1,rect1]=Screen('OpenOffscreenWindow',wPtr);
```

create an offscreen window

```
Screen('DrawText', wOff1, 'Test Text', 200, 800, [0 0 255]);
Screen('DrawLine', wOff1, [0 255 0], 100, 100, 900, 200, 10);

Screen('CopyWindow', wOff1, wPtr);
Screen(wPtr, 'Flip');
pause(2);

Screen('Close', wPtr);
```

Offscreen Window

```
Nscreen=0;   [wPtr,rect]=Screen('OpenWindow',Nscreen);

[wOff1,rect1]=Screen('OpenOffscreenWindow',wPtr);
```

draw things to the offscreen window

```
Screen('DrawText', wOff1, 'Test Text', 200, 800, [0 0 255]);
Screen('DrawLine', wOff1, [0 255 0], 100, 100, 900, 200, 10);

Screen('CopyWindow', wOff1, wPtr);
Screen(wPtr, 'Flip');
pause(2);

Screen('Close', wPtr);
```

Offscreen Window

```
Nscreen=0;   [wPtr,rect]=Screen('OpenWindow',Nscreen);

[wOff1,rect1]=Screen('OpenOffscreenWindow',wPtr);

Screen('DrawText', wOff1, 'Test Text', 200, 800, [0 0 255]);
Screen('DrawLine', wOff1, [0 255 0], 100, 100, 900, 200, 10);

Screen('CopyWindow', wOff1, wPtr);
Screen(wPtr, 'Flip');
pause(2);

Screen('Close', wPtr);
```

copy from the offscreen window to screen window

Offscreen Window

```
Nscreen=0;   [wPtr,rect]=Screen('OpenWindow',Nscreen);

[wOff1,rect1]=Screen('OpenOffscreenWindow',wPtr);

Screen('DrawText', wOff1, 'Test Text', 200, 800, [0 0 255]);
Screen('DrawLine', wOff1, [0 255 0], 100, 100, 900, 200, 10);

Screen('CopyWindow', wOff1, wPtr);
Screen(wPtr, 'Flip');
pause(2);

Screen('Close', wPtr);
```

Flip

Timing

Many experiments we do demand precise timing over stimulus presentation.

Some are fairly coarse:

```
ITI of 500ms
study a stimulus for 2000ms
ITI of 500ms
study a stimulus for 2000ms
```

-
-
-

But even with coarse timings like this, imagine we are running an fMRI or EEG experiment and need to sync brain measurements with PsychToolbox measurements. You need to make sure timings line up with each other.

Timing

Your computer has 1 (perhaps 2, sometimes 4) CPU cores on it. A CPU core is needed for nearly every process your computer carries out (except for some graphics operations, which are done by the GPU*). And each core can do only one thing at a time.

* PsychToolbox takes advantage of this to improve timing.

Timing

To make sure timing is not disrupted by your computer:

- shut down all other applications
- remove background applications loaded at startup
- disable virus scanners
- turn off power management options
- reboot computer from time to time
- have sufficient computer RAM (virtual memory is slow)
- use a real graphics card (ATI or NVIDIA) with sufficient video memory (VRAM)

Imagine this Scenario

You start your Psychtoolbox Experiment ...

It's running through a priming experiment ...

- there's a variable prime-target interval (100-2000ms)
- so a prime is shown, then a blank interval, then a target
- subject makes a response to the target

During one of the 2000ms intervals, your O/S sees that the CPU is free and allows dropbox to start syncing the 5GB folder that someone in your lab just added a few minutes ago. While dropbox has lower **priority**, there could still be a lag in timing.

Imagine this Scenario

You start your Psychtoolbox Experiment ...

It's running through a priming experiment ...

- there's a variable prime-target interval (100-2000ms)
- so a prime is shown, then a blank interval, then a target
- subject makes a response to the target

Fortunately, PsychToolbox has machinery to help prevent mistiming of stimuli and responses, if they are used properly (because even a "clean" computer still has an O/S that can "decide" to run things in the background whenever it likes).

GetSecs

```
t1 = GetSecs;
```

PsychToolbox routine that uses the highest precision realtime clock on the operating system (often microsecond, sometimes millisecond resolution, depending on O/S).

```
t1 = GetSecs;
[wPtr, rect] = Screen('OpenWindow', 0);
black = BlackIndex(wPtr); white = WhiteIndex(wPtr);

for i=1:10
   Screen('FillRect', wPtr, black, [100 100 500 500]);
   Screen(wPtr, 'Flip');
   pause(0.25);

   Screen('FillRect', wPtr, white, [100 100 500 500]);
   Screen(wPtr, 'Flip');
   pause(0.25);
end

Screen('Close', wPtr);

t2 = GetSecs;
fprintf('Total time = %f\n', t2-t1);
```

```
t1 = GetSecs;
[wPtr, rect] = Screen('OpenWindow', 0);
black = BlackIndex(wPtr); white = WhiteIndex(wPtr);

for i=1:10
    Screen('FillRect', wPtr, black, [100 100 500 500]);
    Screen(wPtr, 'Flip');
    pause(0.25);

    Screen('FillRect', wPtr, white, [100 100 500 500]);
    Screen(wPtr, 'Flip');
    pause(0.25);
end

Screen('Close', wPtr);

t2 = GetSecs;
fprintf('Total time = %f\n', t2-t1);
```

Do not use pause *in a Psychtoolbox experiment.*

```
t1 = GetSecs;
[wPtr, rect] = Screen('OpenWindow', 0);
black = BlackIndex(wPtr); white = WhiteIndex(wPtr);

for i=1:10
   Screen('FillRect', wPtr, black, [100 100 500 500]);
   Screen(wPtr, 'Flip');
   pause(0.25);

   Screen('FillRect', wPtr, white, [100 100 500 500]);
   Screen(wPtr, 'Flip');
   pause(0.25);
end

Screen('Close', wPtr);

t2 = GetSecs;
fprintf('Total time = %f\n', t2-t1);
```

"should be" 10x.5 = ***5*** *sec*

is (on my computer) about ***9-14*** *sec*

```
[wPtr, rect] = Screen('OpenWindow', 0);
black = BlackIndex(wPtr); white = WhiteIndex(wPtr);

t1 = GetSecs;
for i=1:10
    Screen('FillRect', wPtr, black, [100 100 500 500]);
    Screen(wPtr, 'Flip');
    pause(0.25);

    Screen('FillRect', wPtr, white, [100 100 500 500]);
    Screen(wPtr, 'Flip');
    pause(0.25);
end
t2 = GetSecs;

Screen('Close', wPtr);

fprintf('Total time = %f\n', t2-t1);
```

"should be" 10x.5 = ***5*** *sec*

is (on my computer) about ***5.33*** *sec*

```
[wPtr, rect] = Screen('OpenWindow', Nscreen);
black = BlackIndex(wPtr);    white = WhiteIndex(wPtr);

t1 = GetSecs;
when = 0;
for i=1:10
    Screen('FillRect', wPtr, black, [100 100 500 500]);
    [VBLTimestamp StimulusOnsetTime] = Screen(wPtr, 'Flip', when);

    p = 0.25;  when = VBLTimestamp + p;
    Screen('FillRect', wPtr, white, [100 100 500 500]);
    [VBLTimestamp StimulusOnsetTime] = Screen(wPtr, 'Flip', when);

    p = 0.25;  when = VBLTimestamp + p;
end

[VBLTimestamp StimulusOnsetTime] = Screen(wPtr, 'Flip', when);
t2 = GetSecs;
Screen('Close', wPtr);
fprintf('Total time = %f\n', t2-t1);
```

"should be" 10x.5 = ***5*** *sec*

is (on my computer) about ***5.33*** *sec*

```
[VBLTimestamp StimulusOnsetTime] = Screen(wPtr, 'Flip', when);
```

At what time to do the `'Flip'` in seconds.*
If `when=0`, on next screen refresh.

PsychToolbox "sleeps" (by default) until "`when`" occurs and then returns back to your Matlab script.

* Note that this is time (in seconds) like what you would get with GetSecs, not some relative time, like "in 2 seconds".

```
[VBLTimestamp StimulusOnsetTime] = Screen(wPtr, 'Flip', when);
```

Returns the time when the hardware refresh started. Best time stamp to make sure the total timing of all events in a sequence stays consistent.

```
[wPtr, rect] = Screen('OpenWindow', Nscreen);
black = BlackIndex(wPtr);    white = WhiteIndex(wPtr);

t1 = GetSecs;
when = 0;
for i=1:10
    Screen('FillRect', wPtr, black, [100 100 500 500]);
    [VBLTimestamp StimulusOnsetTime] = Screen(wPtr, 'Flip', when);

    p = 0.25;  when = VBLTimestamp + p;
    Screen('FillRect', wPtr, white, [100 100 500 500]);
    [VBLTimestamp StimulusOnsetTime] = Screen(wPtr, 'Flip', when);

    p = 0.25;  when = VBLTimestamp + p;
end

[VBLTimestamp StimulusOnsetTime] = Screen(wPtr, 'Flip', when);
t2 = GetSecs;
Screen('Close', wPtr);
fprintf('Total time = %f\n', t2-t1);
```

"should be" 10x.5 = ***5*** *sec*

is (on my computer) about ***5.33*** *sec*

```
[wPtr, rect] = Screen('OpenWindow', Nscreen);
black = BlackIndex(wPtr);    white = WhiteIndex(wPtr);
FlipInterval = Screen('GetFlipInterval',wPtr)/2;
slack = FlipInterval/2;
[VBLTimestamp StimulusOnsetTime] = Screen(wPtr, 'Flip', 0);
p = 0.25;  when = VBLTimestamp + p;
t1 = GetSecs;
for i=1:10
    Screen('FillRect', wPtr, black, [100 100 500 500]);
    [VBLT StimulusOT] = Screen(wPtr, 'Flip', when-slack);
    p = 0.25;  when = VBLT + p;

    Screen('FillRect', wPtr, white, [100 100 500 500]);
    [VBLT StimulusOT] = Screen(wPtr, 'Flip', when-slack);
    p = 0.25;  when = VBLTimestamp + p;
end
t2 = GetSecs;
Screen('Close', wPtr);

fprintf('Total time = %f\n', t2-t1);
```

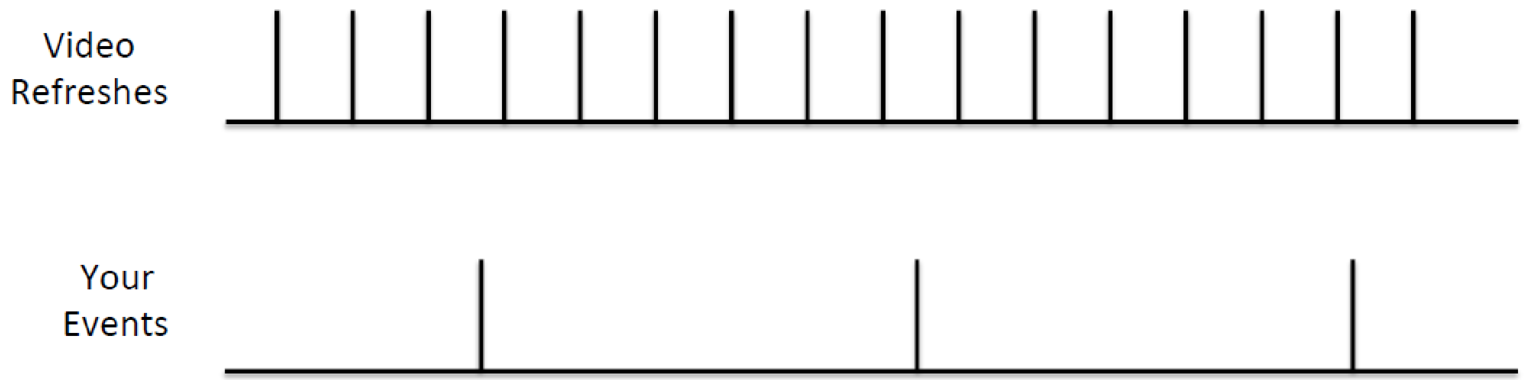
```
FlipInterval=Screen('GetFlipInterval',wPtr);
```

`1/FlipInterval` gives you the monitor refresh rate

```
FlipInterval=Screen('GetFlipInterval',wPtr);
slack = FlipInterval/2;
```

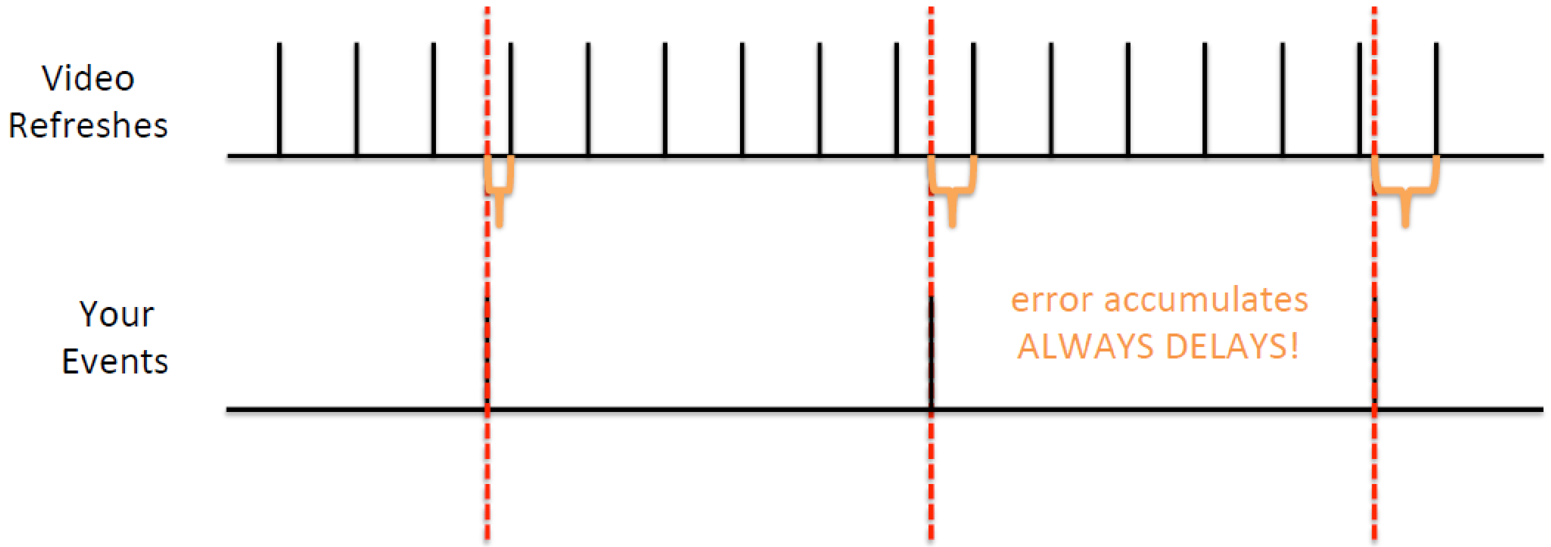
```
FlipInterval=Screen('GetFlipInterval',wPtr);
slack = FlipInterval/2;
when = 200;
[VBLT StimulusOT] = Screen(wPtr, 'Flip', when);
[VBLT StimulusOT] = Screen(wPtr, 'Flip', when);
[VBLT StimulusOT] = Screen(wPtr, 'Flip', when);
```

Without Slack

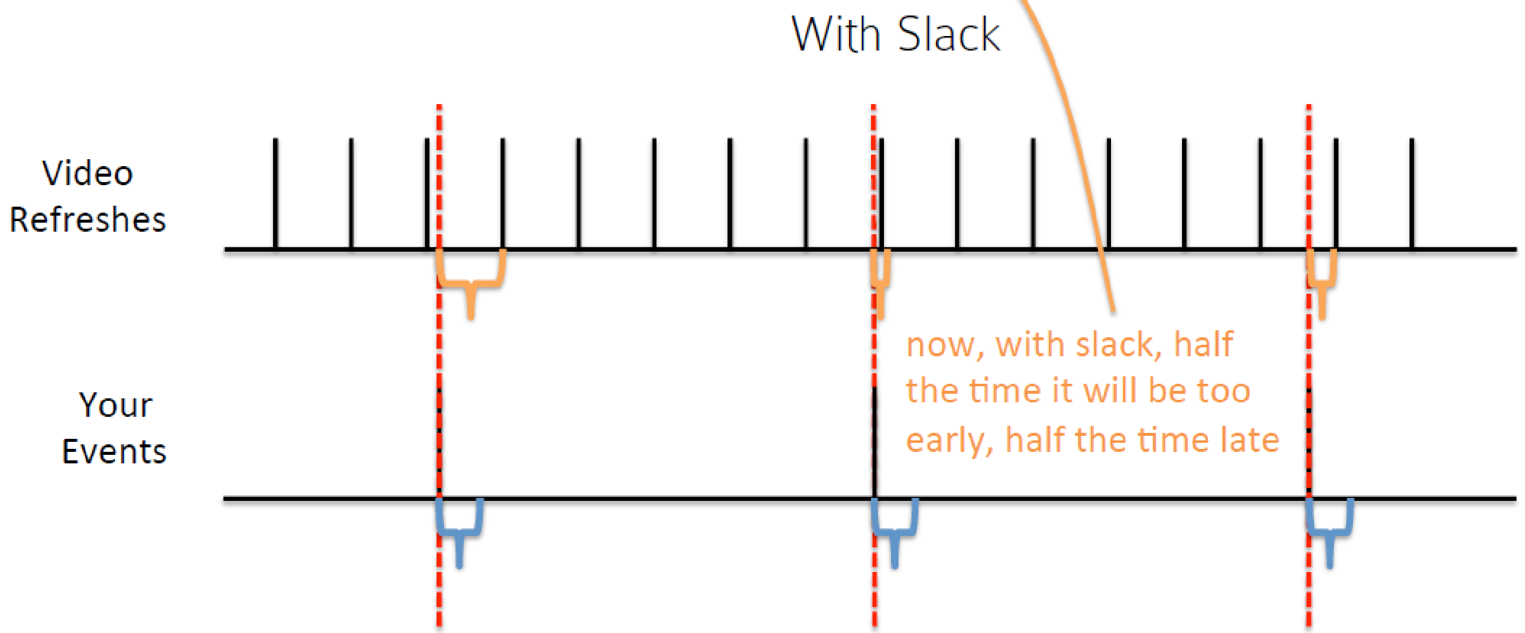


```
FlipInterval=Screen('GetFlipInterval',wPtr);
slack = FlipInterval/2;
when = 200;
[VBLT StimulusOT] = Screen(wPtr, 'Flip', when);
[VBLT StimulusOT] = Screen(wPtr, 'Flip', when);
[VBLT StimulusOT] = Screen(wPtr, 'Flip', when);
```

Without Slack



```
FlipInterval=Screen('GetFlipInterval',wPtr);
slack = FlipInterval/2;
when = 200;
[VBLT StimulusOT] = Screen(wPtr, 'Flip', when-slack);
[VBLT StimulusOT] = Screen(wPtr, 'Flip', when-slack);
[VBLT StimulusOT] = Screen(wPtr, 'Flip', when-slack);
```


With Slack
Video Refreshes
Your Events
now, with slack, half the time it will be too early, half the time late

```
[wPtr, rect] = Screen('OpenWindow', Nscreen);
black = BlackIndex(wPtr);    white = WhiteIndex(wPtr);
FlipInterval = Screen('GetFlipInterval',wPtr);
slack = FlipInterval/2;
[VBLTimestamp StimulusOnsetTime] = Screen(wPtr, 'Flip', 0);
p = 0.25;  when = VBLTimestamp + p;
t1 = GetSecs;
for i=1:10
    Screen('FillRect', wPtr, black, [100 100 500 500]);
    [VBLT StimulusOT] = Screen(wPtr, 'Flip', when-slack);
    p = 0.25;  when = VBLT + p;

    Screen('FillRect', wPtr, white, [100 100 500 500]);
    [VBLT StimulusOT] = Screen(wPtr, 'Flip', when-slack);
    p = 0.25;  when = VBLTimestamp + p;
end
t2 = GetSecs;
Screen('Close', wPtr);

fprintf('Total time = %f\n', t2-t1);
```

"should be" 10x.5 = ***5*** *sec*

is about ***4.99-5.01*** *sec*

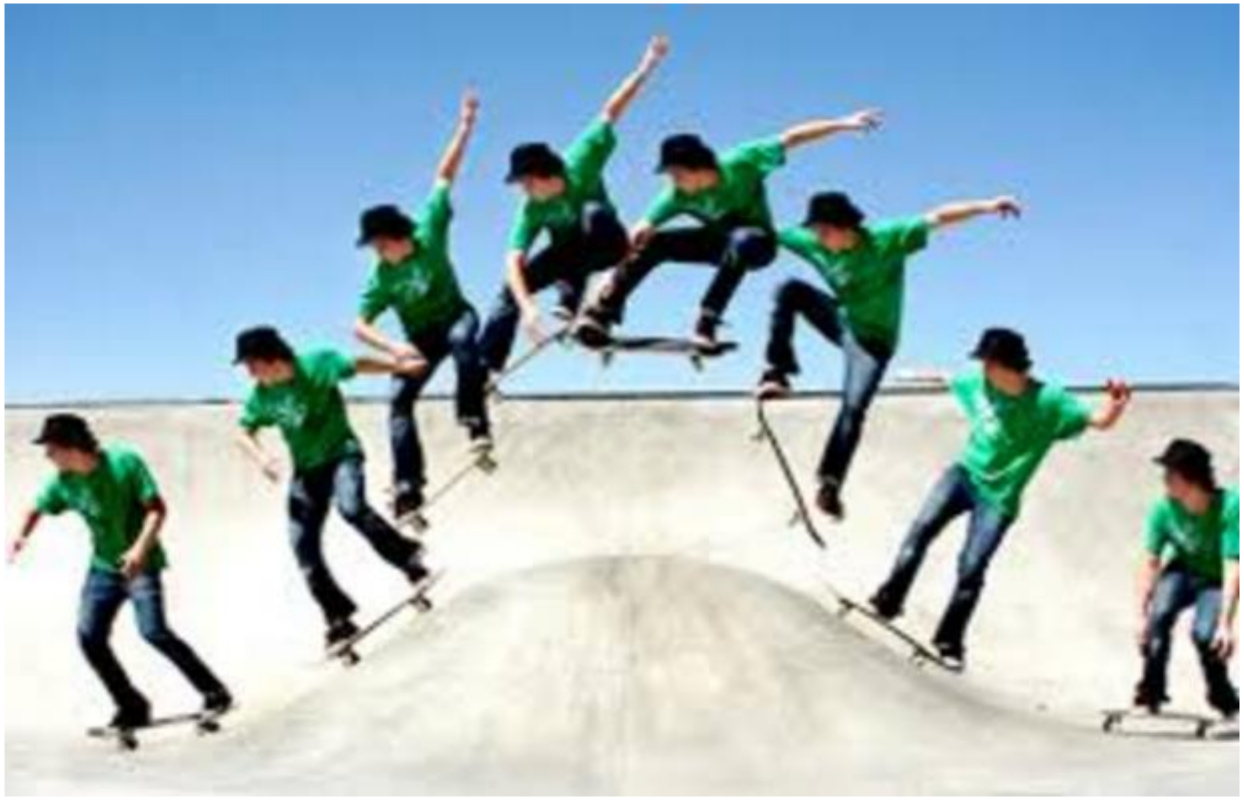
Check the timing always

Calculate how much time your experiment SHOULD take.

Compare with with how long is DOES take.

And do the comparison over multiple trials. Small differences can accumulate over the course of many trials and start to loom large. If they differ by a few millisecond, you're okay. If they differ by hundreds of milliseconds or more, something is seriously wrong.

Motion Frames



Importance of Preloading Images/Textures

Bad: load images, create textures, within the rapid stream

```
for i=1:length(fnames)
    I = rgb2gray(imread(fnames{i}));
    txtImg = Screen('MakeTexture', wPtr, I);
    Screen('DrawTexture', wPtr, txtImg);
    [VBLT SOTime] = Screen(wPtr, 'Flip', …
                          VBLT+(flipSpd*FlipInt)-slack);

    Screen('FillRect',wPtr,black);
    [VBLT SOTime] = Screen(wPtr, 'Flip', …
                          SOTime+(flipSpd*FlipInt)-slack);
end
```

Importance of Preloading Images/Textures

Better: preload images

```
for i=1:length(fnames)
    I{i} = rgb2gray(imread(fnames{i}));
end

for i=1:length(fnames)
    txtImg = Screen('MakeTexture', wPtr, I{i});
    Screen('DrawTexture', wPtr, txtImg);
    [VBLT SOTime] = Screen(wPtr, 'Flip', …
                              VBLT+(flipSpd*FlipInt)-slack);

    Screen('FillRect',wPtr,black);
    [VBLT SOTime] = Screen(wPtr, 'Flip', …
                              SOTime+(flipSpd*FlipInt)-slack);
end
```

Importance of Preloading Images/Textures

Best: preload images and pre-create textures

```
for i=1:length(fnames)
    I{i} = rgb2gray(imread(fnames{i}));
    txtImg{i} = Screen('MakeTexture', wPtr, I{i});
end

for i=1:length(fnames)
    Screen('DrawTexture', wPtr, txtImg{i});
    [VBLT SOTime] = Screen(wPtr, 'Flip', …
                          VBLT+(flipSpd*FlipInt)-slack);

    Screen('FillRect',wPtr,black);
    [VBLT SOTime] = Screen(wPtr, 'Flip', …
                          SOTime+(flipSpd*FlipInt)-slack);
end
```

VBLSyncTest

/Applications/Psychtoolbox/PsychTests/VBLSyncTest.m

will test that the reported refresh rate matches what PsychToolbox calculates from its internal queries to the hardware

it will also test how well your hardware to display what PsychToolbox commands your computer to display*

* Note: This DOES NOT test the physical monitor (see next section).