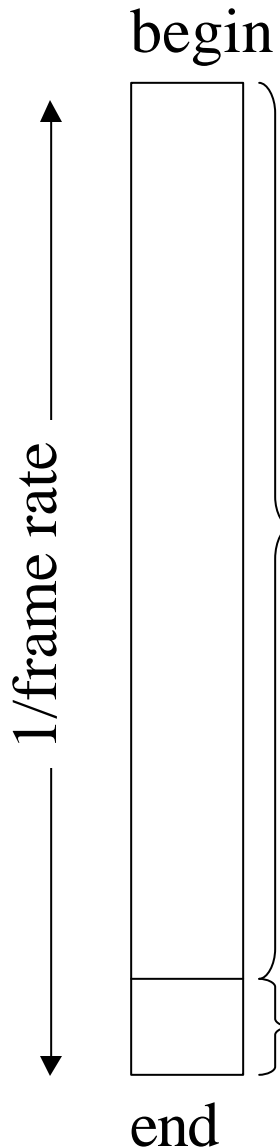


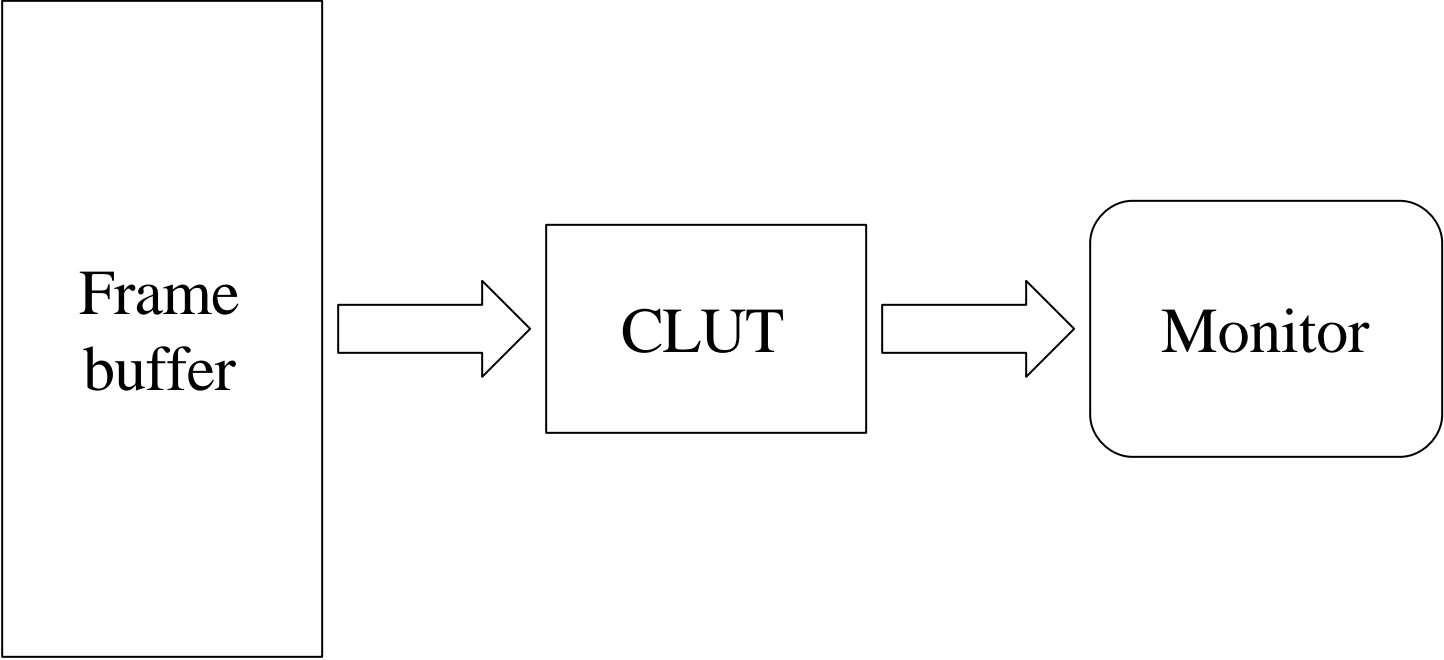
Life in a video frame



- Rastering: transfer pixels from frame buffer (video memory) to CRT, from top left to bottom right, line by line (called a raster line). Changing the content of the frame buffer during this time will cause flickers.
- Vertical retrace: resetting the electron guns to upper left. This normally takes a couple ms during which time nothing is transferred to the screen. Changing the frame buffer during this time will not cause any flicker.
- Refresh (or frame) rate: number of frames per second, the maximum rate at which the video can be updated.

Flicker-free video play back

- Draw quickly into the frame buffer during vertical retrace (the video normally signals the on-set of a vertical retrace).
 - screen: ‘waitblanking’ to wait for the next vertical retrace
 - screen: ‘setclut’ or ‘clutmovie’ to do CLUT animation
 - screen: ‘copywindow’ to draw a new frame quickly (limited by the size of the window)
- Double buffering – draw to a “back” buffer during rastering of the current frame, then swap the front and back buffer during vertical retrace. Screen does not have function to support this. You will have to do this in C via ActiveX or OpenGL.



Generating motion

- Image animation
 - straightforward, one image per frame
 - preparation: draw frames to off-screen windows
 - animation: copy off-screen windows to the on-screen window at frame rate
 - Disadvantage: maximum frame rate limited by image size
- CLUT animation
 - a single fixed image, with pixel values coding pixel positions and groupings
 - CLUT is changed at each frame, thus affecting the final rasterized image
 - Advantage: frame rate not limited by image size.
 - Disadvantage: number of independent pixel group limited by the number CLUT entries (mostly 256)

'Rush' command

- Take a string or an array of strings that specifies a list of matlab command and runs it in high priority. E.g.:

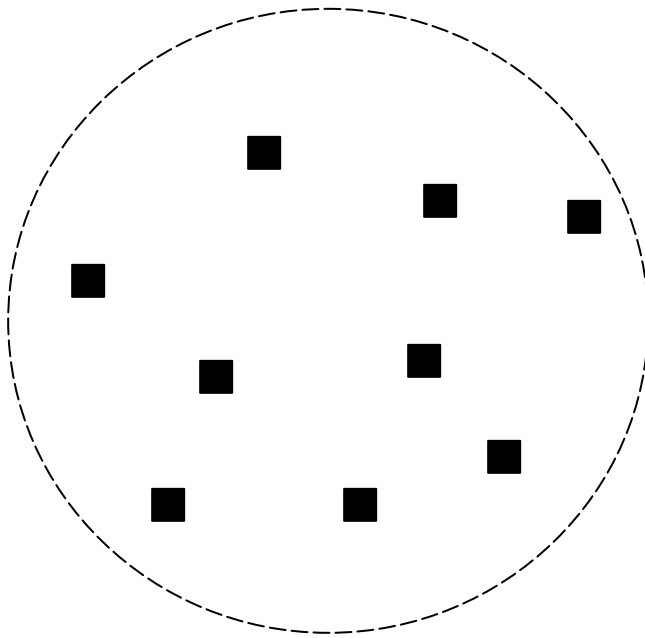
```
SCREEN('Screens');      % Load all SCREEN.mex functions to memory
i=0;                    % Allocate all variables.
loop={
    'for i=1:100;'
        'SCREEN(window, 'WaitBlanking');'
        'SCREEN('CopyWindow',w(i),window);'
    'end;'
};
priorityLevel=MaxPriority(window, 'WaitBlanking');
RUSH(loop,priorityLevel); % run commands in 'loop'
```

Common low-level motion stimuli

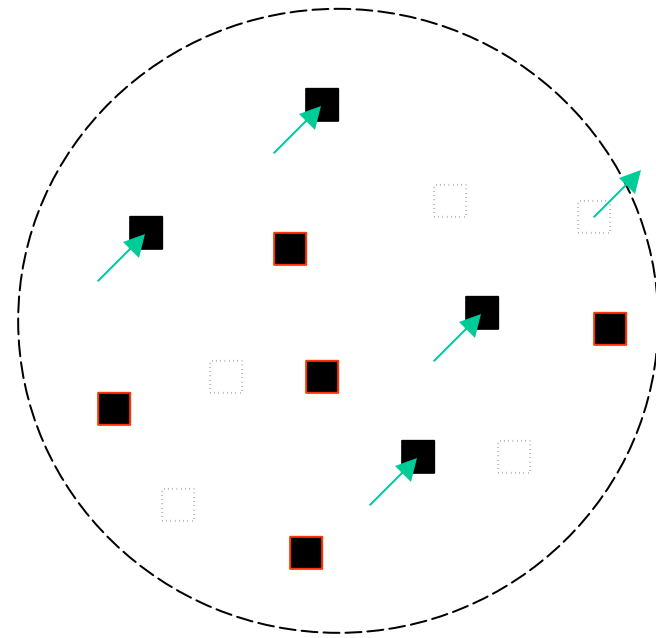
- Random dots
 - unlimited life time
 - limited life time
 - a dot stays on a trajectory for a fixed number of frames before disappearing and replaced by another random dot
- Gratings
 - simple gratings
 - single spatial frequency
 - compound gratings
 - multiple spatial frequencies




Limited life-time random dot display

Frame 1



Frame 2



-  moved dots
-  expired / out of bound dots
-  new, randomly placed dots

Algorithm for generating limited lifetime random dots

- n dots, k frames per dot
- Initialization:
 - generate n randomly positioned dots
 - randomly divide the dots into k equal-size groups, each assigned one of k ages
- Iteration:
 - calculate the new position of each dot
 - add 1 to each dot's age
 - remove expired ($\text{age} > k$) and out of bound dots
 - replace the removed dots by randomly placed new dots (age set to 1)
 - create the new frame

Simple moving grating

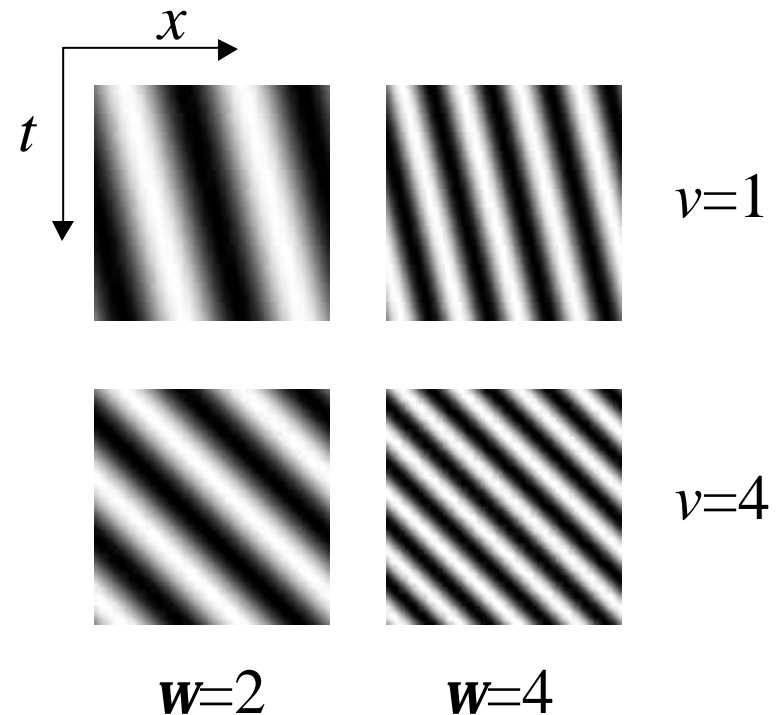
$$c(x, t) = A \sin(2\pi w(x - vt))$$

w : spatial frequency

v : grating velocity

A : grating contrast

Space-time diagram



Spatial-temporal filtering

- Filter the S-T diagram, but watch out the temporal phase (i.e. only the past is available to the filter).