


CS1315: Introduction to Media Computation

Movies

Movies, animations, and video

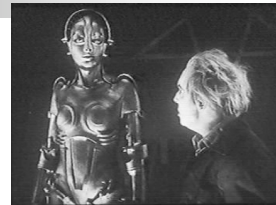
- We're going to refer generically to captured (recorded) motion as "movies."
 - **This includes motion entirely generated by graphical drawings, which are normally called animations.**
 - **This also includes motion generated by some kind of photographic process, normally called video.**

Psychophysics of Movies: Persistence of Vision



- What makes movies work is yet another limitation of our visual system:
Persistence of vision
- We do not see every change that happens in the world around us.
- Instead, our eye *retains* an image (i.e., tells the brain "This is the latest! Yup, this is still the latest!") for a brief period of time.
 - **If this were not the case, you would be aware of every time that your eye blinks because the world would "go away" for a moment.**

16 frames and it's motion



- If you see 16 separate pictures in one second, and these pictures are logically sequenced,
 - **That is, #2 could logically follow from the scene in #1.**
 - **16 pictures of completely different things doesn't work,**
- You will perceive the pictures as being in motion.
- 16 frames per second (fps), 16 pictures in a second, is the lower bound for the sensation of motion.

Beyond 16 fps

- Early silent pictures were 16 fps.
- Motion picture standards shifted to 24 fps to make sound smoother.
- Videocameras (digital video) captures 30 fps
- How high can we go?
 - **Air force experiments suggest that pilots can recognize a blurb of light in 1/200th of a second!**
 - **Video game players say that they can discern a difference between 30 fps and 60 fps.**
- Bottomlines:
 - **Generate at least 16 fps and you provide a sense of motion.**
 - **If you want to process video, you're going to have 30 fps to process (unless it's been modified elsewhere for you.)**

Processing movies

- Our frames are going to be JPEG pictures.
 - **One JPEG file per frame.**
- So, if we're going to be processing movies, we're going to generating or processing sequences of JPEG files.

Using MovieMaker

- To generate a series of frame pictures in a folder from an MPEG file.
- To make a folder of frame pictures into a QuickTime Movie (<http://www.apple.com/quicktime>)
- (Can also use MediaTools application)



MPEG? QuickTime? AVI? JMV?

- MPEG, QuickTime, and AVI are *compressed* movie formats.
 - **They don't record every frame.**
 - **Rather, they record some key frames, and then store data about what parts of the screen change on intervening frames.**
 - **MPEG is an international standard, from the same people who invented JPEG.**
 - **AVI is a Microsoft standard.**
 - **QuickTime is an Apple standard.**
- JMV is a file consisting of JPEG frames in an array.
 - **All frames represented**

Generating other kinds of movies

- Other tools can generate other kinds of movie formats.
- QuickTime Pro (<http://www.apple.com/quicktime>) can read a sequence of JPEG images and produce MPEG, AVI, or QuickTime movies.
- ImageMagick (open source toolkit) can also read a sequence of JPEG images and produce MPEG movies.

Why do we compress movies?

- Do the math:
 - **One second of 640x480 pixels at 30 fps**
 - **30 (frames) * 640 * 480 (pixels) = 9,216,000 pixels**
 - **With 3 bytes of color per pixel, that's 27,648,000 bytes or 27 megabytes of information per second.**
 - **For a 90 minute feature movie (short), that's 90 * 60 * 27,648,000 = 149,299,200,000 bytes (149 gigabytes)**
- A DVD stores 6.47 gigabytes of data.
 - **So even on a DVD, the movie is compressed.**

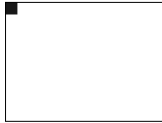
MPEG movie = MPEG frames plus MP3 soundtrack

- An MPEG movie is actually a series of MPEG frames composed with an MP3 soundtrack.
 - **It's literally two files stuck together in one.**
- We're not going to deal with sound movies for now.
 - **The real challenge in doing movie processing is generating and manipulating frames.**

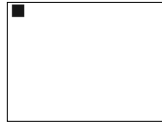
Get the frames in order

- Many tools can process frames in order *if the order is specified.*
- We specify the order by encoding the number of the frame into the name.
 - **If you put in leading zeroes so that everything is the same length, the order is alphabetical as well as numerical.**

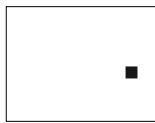
movingRectangle: A Few Frames



frame01.jpg



frame02.jpg



frame50.jpg

Important cool fact: You can draw past the end of the picture!

- addText, addRect, and the rest of the drawing tools will work even if you go beyond the edge of the drawing.
 - Drawings will *clip* what can't be seen in them, so you don't get an array out of bounds error.
 - This is a big deal, because it means that you don't have to do complicated math to see when you're past the end of the drawing.
 - But only for the drawing functions.
 - If you set pixels, you're still on your own to stay in range.

Making a tickertape

```
def tickertape(dir, string):
    for frame in range(0,100): #99 frames
        canvas = makePicture(getMediaPath("640x480.jpg"))
        #Start at right, and move left
        addText(canvas,600-(frame*10),100,string)
        # Now, write out the frame
        # Have to deal with single digit vs. double digit frame
        # numbers differently
        framenum=str(frame)
        if frame < 10:
            writePictureTo(canvas, dir+"//frame0"+framenum+".jpg")
        else:
            writePictureTo(canvas, dir+"//frame"+framenum+".jpg")
```

Can we move more than one thing at once? Sure!

```
def movingRectangle2(directory):
    for frame in range(1,100): #99 frames
        canvas = makePicture(getMediaPath("640x480.jpg"))
        if frame < 50: #Less than 50, move down
            # Generate new positions each frame number
            addRectFilled(canvas,frame*10,frame*5, 50,50,red)
        if frame >= 50: #Greater than 50, move up
            addRectFilled(canvas,(50-(frame-50))*10,(50-(frame-50))*5,
            50,50,red)
        # Let's have one just moving around
        addRectFilled(canvas,100+ int(10 *
        sin(frame)),4*frame+int(10* cos(frame)),50,50,blue)
        # Now, write out the frame
        # Have to deal with single digit vs. double digit frame
        # numbers differently
        framenum=str(frame)
        if frame < 10:
            writePictureTo(canvas,directory+"//frame0"+framenum+".jpg")
        if frame >= 10:
            writePictureTo(canvas,directory+"//frame"+framenum+".jpg")
```