Images, Sound, and Multimedia

Reminder: Next Quiz on 4/15
No Surprises

• Images, Sound, Music, and Movies
  – It’s all numbers
  – Binary Numbers

• Today we’ll discuss how multimedia is converted to and from numbers, and processed on a computer

Sampling in the Space Domain
Colors

- In a computer’s memory colors are represented as a combination of three-additive primary colors (red, green, blue)
- Nearly every color that you perceive can be generated by combining these colors
- Under normal conditions, humans can discern about 100 shades of a color (Just noticeable differences)
Visual Sensitivity

A light sensitive organ; the highest bandwidth channel into the brain.
Trichromatic Sensitivity

- Two kinds of cells, rods and cones
- Three types of cones, red, green, and blue
Computer Colors

- Combining what we know
  - Only 3 colors are needed to stimulate the perception of nearly all others
  - Roughly 100 shades of each are perceivable
- Three one-byte channels (r, g, b) each with 256 levels
- Alternatives: 15-bit color, 5-bits per r, g, b (banding is sometimes noticeable)
Some Preliminaries

• Download the Python Imaging Library from http://www.pythonware.com/products/pil
Creating an Image

Personal firewall software may warn about the connection IDLE makes to its subprocess using this computer's internal loopback interface. This connection is not visible on any external interface and no data is sent to or received from the Internet.

```
IDLE 1.2.2
>>> import Image
>>> im = Image.new("RGB", (200,200), (255,0,0))
>>> print im.size, im.mode
(200, 200) RGB
>>> im.show()
```
Loading an Image from a file

```python
IDLE 1.2.2
>>> import Image
>>> im = Image.new("RGB", (200,200), (255,0,0))
>>> print im.size, im.mode
(200, 200) RGB
>>> im.show()

>>> im = Image.open("C:\Pictures\mandrill.png")
>>> print im.size, im.mode
(512, 512) RGB
>>> im.show()
```
Changing an Image

IDLE 1.2.2
>>> import Image
>>> im = Image.new("RGB", (200,200), (255,0,0))
>>> print im.size, im.mode
(200, 200) RGB
>>> im.show()

>>> im = Image.open("C:\\Pictures\\mandrill.png")
>>> print im.size, im.mode
(512, 512) RGB
>>> im.show()

>>> for y in range(im.size[1]):
    for x in range(im.size[0]):
        r, g, b = im.getpixel((x, y))
        im.putpixel((x, y), (b, g, r))

>>> im.show()
Drawing a Circle

• And resizing an image

```python
>>> im.show()
>>> colors.show()
>>> circle = Image.new('RGB', (256, 256))
>>> for y in range(256):
...     for x in range(256):
...         if (x-128)**2 + (y-128)**2 < 100**2:
...             circle.putpixel((x,y), (2*abs(x-128), 0, 2*abs(y-128)))

>>> circle.show()
```

```python
>>> im = Image.open("C:\Pictures\mandrill.png")
>>> im.thumbnail((64, 64))
>>> im.show()
```
Image Compression

- There are 2 common types of image compression
  - Lossy - (.jpg)
    - Higher compression rate (fewer bits/pixel)
    - Good for “Natural images”
  - Lossless - (.bmp, .png)
    - Lower compression rate (more bits/pixel)
    - Best for “Graphics Arts Images”
    - Preserves Sharp lines
Image Compression Example

```python
>>> import Image
>>> im = Image.new("RGB", (256,256))
>>> for y in range(256):
    for x in range(256):
        if (x % 10 < 3) or (y % 10 < 3):
            im.putpixel((x,y), (100,192,255))

>>> im.show()
>>> im.save("grid.png", "PNG")
>>> im.save("grid.jpg", "JPEG")
```

PNG version

![PNG version](grid.png)

JPG version

![JPG version](grid.jpg)
Sound by Numbers

- Sound is a result of pressure waves propagating through the air
- You perceive sound as the intensity and frequency of these pressure waves
- How are these “waves” represented in a computer
Sound Perception

- The human ear can nominally hear sounds in the range 20 Hz to 20,000 Hz (20 kHz).
- This upper limit tends to decrease with age.
- Most adults are unable to hear above 16 kHz.
- The ear itself does not respond to frequencies below 20 Hz.
- Low frequencies are perceived via the body's sense of touch.
Some Quick and Dirty Sound

• Python provides a range of libraries for playing and processing sound

```python
IDLE 1.2.2
>>> import winsound
>>> soundfile = "C:\Windows\Media\tada.wav"
>>> winsound.PlaySound(soundfile, winsound.SND_FILENAME|winsound.SND_ASYNC)
```

• Now, let's look at some details about this file

```python
>>> import wave
>>> sound = wave.open(soundfile, 'r')
>>> sound.getnchannels()
2
>>> sound.getframerate()
22050
>>> sound.getnframes()
42752
>>> secs = float(sound.getnframes())/sound.getframerate()
>>> print secs
1.93886621315
```
Audio Sampling

- The sampling rate, defines the number of samples per second taken from a continuous signal to make a discrete (digital) signal.
- It is measured in hertz (Hz)
- The inverse of the sampling frequency is the sampling period or sampling interval, which is the time between samples.
## Common Sampling Rates

<table>
<thead>
<tr>
<th>Rate</th>
<th>Use</th>
</tr>
</thead>
<tbody>
<tr>
<td>8,000 Hz</td>
<td>Telephone, digital voice</td>
</tr>
<tr>
<td>22,050 Hz</td>
<td>Half CD rate, for medium-quality digital audio files</td>
</tr>
<tr>
<td>32,000 Hz</td>
<td>Consumer MiniDV, Digital camcorders, FM digitizers</td>
</tr>
<tr>
<td>44,100 Hz</td>
<td>Audio CD, and MPEG-1 audio</td>
</tr>
<tr>
<td>48,000 Hz</td>
<td>High-quality MiniDV, Digital Audio recorder</td>
</tr>
</tbody>
</table>

- The table above gives the common sampling rates used in many audio applications.
- The highest reproducible audible frequency by any format is 1/2 of its sampling rate.
Making a Sound

• A 1000 Hz tone

```python
>>> import math, array, wave, winsound
>>> new = wave.open("test.wav", 'w')
>>> new.setnchannels(1)
>>> new.setframerate(22050)
>>> new.setsampwidth(2)
>>> data = [int(32000.0*math.sin(2*math.pi*i/22.050)) for i in range(44100)]
>>> new.writeframes(array.array('h',data).tostring())
>>> new.close()
>>> data[0:100]
[0, 8995, 17265, 24143, 29073, 31659, 31691, 29168, 24292, 17457, 9214, 227, -8776, -17073, -23992, -28977, -31625, -31722, -29261, -24440, -17647, -9432, -455, 8557, 16880, 23841, 28880, 31589, 31751, 29352, 24586, 17837, 9649, 683, -8337, -16685, -23688, -28781, -31552, -31779, -29442, -24731, -18026, -9866, -911, 8116, 16490, 23535, 28680, 31513, 31805, 29531, 24875, 18214, 10083, 1139, -7896, -16295, -23379, -28579, -31473, -31829, -29618, -25018, -18401, -10299, -1367, 7675, 16098, 23223, 25475, 31431, 31852, 29703, 25160, 18587, 10515, 1595, -7453, -15901, -23066, -28371, -31387, -31873, -29787, -25300, -18772, -10730, -1822, 7231, 15702, 22907, 28265, 31342, 31892, 29870, 25439, 18956, 10944, 2050]
>>> winsound.PlaySound("test.wav", winsound.SND_FILENAME|winsound.SND_ASYNC)
>>>]
```
A Square Wave Tone

```python
>>> for i in range(len(data)):
    if data[i] < 0:
        data[i] = -10000
    else:
        data[i] = 10000

>>> sqr = wave.open("rasp.wav", 'w')
>>> sqr.setnchannels(1)
>>> sqr.setframerate(22050)
>>> sqr.setsampwidth(2)
>>> sqr.writeframes(array.array('h',data).tostring())
>>> sqr.close()
>>> winsound.PlaySound("rasp.wav", winsound.SND_FILENAME|winsound.SND_ASYNC)
>>>```
>>> import random
>>> snd = wave.open("noise.wav", 'w')
>>> snd.setnchannels(1)
>>> snd.setframerate(22050)
>>> snd.setsampwidth(2)
>>> data = [random.randint(-10000,10000) for i in range(441000)]
>>> snd.writeframes(array.array('h',data).tostring())
>>> snd.close()
>>> winsound.PlaySound("noise.wav", winsound.SND_FILENAME|winsound.SND_ASYNC)
>>>
More to Come…

• More on Images
• Sound and other Media