# Computer Graphics

**Prof. Feng Liu**

**Fall 2018**

http://www.cs.pdx.edu/~fliu/courses/cs447/

**09/25/2018**

# Today

- Course overview and information
- Digital images
- Homework 1 – due Oct. 4 in class
  - No late homework will be accepted

# Pre-Requisites

- C/C++ programming
- Linear algebra
  - A free book by Prof. Jim Hefferon http://joshua.smcvt.edu/linearalgebra/

# Acknowledgement

- This course is based on CS 559 at the University of Wisconsin, Madison taught by Dr. *Stephen Chenney*
- The course materials are adapted and used here with Dr. Chenney's permission


Big Buck
BUNNY
UHD
60fps

Source: https://www.youtube.com/watch?v=GbpqwUUfMAQ


Source: https://www.youtube.com/watch?v=24HJCuWUS0M

# What is Computer Graphics?

- Practically, it's about movies, games, design, training, art, advertising, communication, …
- Technically, it's about the production, manipulation and display of images using computers

# Graphics Building Blocks

- Images and computers
  - Sampling, color, filters, …
- Drawing in 2D
  - Drawing lines and triangles, clipping, transformations
- Drawing in 3D
  - Viewing, transformations, lighting, real-time graphics
- Modeling in 3D
  - Describing volumes and surfaces, drawing them effectively
- Miscellaneous topics
  - Raytracing, animation, …

# People

- Lecturer: Prof. Feng Liu
  - Room FAB 120-09
  - Office hours: TR 3:30-4:30pm
  - fliu@cs.pdx.edu
- TA: Qiqi Hou
  - Fishbowl
  - Office hours:  TR 2:00-3:00pm
  - qiqi2@pdx.edu

# Web and Computer Account

- Course website
  - http://www.cs.pdx.edu/~fliu/courses/cs447/
  - Homework, projects, readings
- Everyone needs a Computer Science department computer account
  - Get account at CAT at http://cat.pdx.edu

# Textbooks & Readings

- Fundamentals of Computer Graphics
  - By Shirley et al.
  - 4rd edition, A.K. Peters
- OpenGL Programming Guide
  - By Shreiner et al.
  - 8th edition (does not matter which edition)
  - Early version available online
    - http://www.glprogramming.com/red/

# Grading

- 20% Midterm
- 25% Final
- 10% Project 1
- 20% Project 2
  - Have the option to work in group
- 25% Homework

# Homework

- Roughly one homework every two weeks
  - 5 homework totally
- Primary to explore topics further and prepare you for the exams
- Some topics will be presented only in homework
  - Review of linear algebra in Homework 1

# Projects

- Project 1: Image editing
- Project 2: Building a virtual theme park
- Visual C++ & FLTK & OpenGL

# Project demo

# C++

- Required for this class
  - You presumably have taken CS 202 or its equivalence for C++
- We’ll provide tutorials for you to use C++ within Visual Studio 2008/2010
  - Help you get familiar with VS ‘08, NOT C++
  - Visual Studio 2012 is similar

# Software Infrastructure

- FLTK will be the user interface toolkit
  - Provides windows, buttons, menus, etc
  - C++ class library, completely portable
  - Available for free: www.fltk.org
- OpenGL will be the 3D rendering toolkit
  - Provides an API for drawing objects specified in 3D
  - Included as part of Windows and in most Unix distributions
    - getting hardware acceleration may take some effort
- Visual Studio 2008/2010/2012 will be the programming environment for grading
- **To be graded, your projects must compile under Visual C++ on the machines in Windows Lab**

# Lab Facilities

- EB 325: Windows Lab
  - Visual Studio 2012 installed
- You need a Computer Science department account to use any CS labs
  - Request one from CAT

# Visual Computing at PSU

- Undergraduate/graduate courses
  - Fall: Computer Graphics
  - Winter: Introduction to Computer Vision
- Graduate course
  - Spring: Computational Photography
  - Senior undergraduate students are welcome too

# Admin Questions?

# Today

- Course overview and information
- Digital Images
- Homework 1 - due Oct. 04 in class

# Images

- An image is intended to describe the light that arrives at your eyes when you view it
  - You can be even more abstract: image describes what you should think when you see it
- Different display devices convey the image content in different ways
  - e.g. printer and computer monitors use two different approaches
  - The same image may look different on different monitors
    - Who cares?

# Image Formats

- We are familiar with many forms of image:
  - Photographs
  - Paintings
  - Sketches
  - Television (NTSC, PAL-SECAM)
  - Digital formats (JPEG, PNG, GIF, BMP, TGA, etc.)
- Each form has its own way of obtaining and storing the information content

# Digital Images

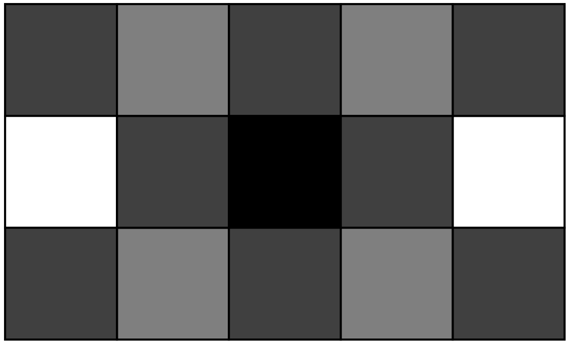
- Many formats exist for storing images on a computer
  - JPEG, PNG, GIF, BMP, TGA, etc.
- There are some conflicting goals:
  - The storage cost should be minimized

# Digital Images

- Many formats exist for storing images on a computer
  - JPEG, PNG, GIF, BMP, TGA, etc.
- There are some conflicting goals:
  - The storage cost should be minimized
  - The amount of information stored should be maximized
    - The size of something and the amount of information is contained are not the same thing
  - Original information versus perceptual equivalence
  - Tracking ownership may be important
- Most formats you are familiar with are *raster* images

# Raster Images

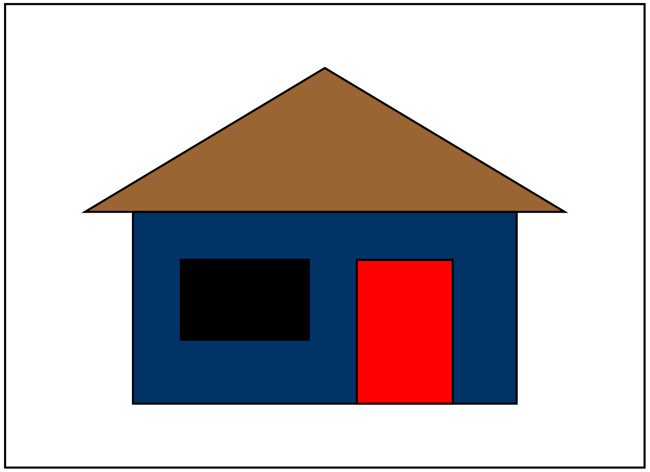
- A raster is a regular grid of *pixels* (picture elements)
- Raster image formats store the color at each pixel, and maybe some other information
  - Easiest is to use a simple array of pixel values
  - Some formats store the pixel information in *very* different ways
  - e.g. a 5x3, floating point, grayscale image



| | | | | |
|---|---|---|---|---|
| 0.25 | 0.5 | 0.25 | 0.5 | 0.25 |
| 1 | 0.25 | 0 | 0.25 | 1 |
| 0.25 | 0.5 | 0.25 | 0.5 | 0.25 |

# Vector Images

- Vector formats offer an alternative way to store images
- The most common use of vector formats are in fonts – images of characters (Postscript, TrueType)
- Store images as collections of geometric primitives
  - E.g. Lines, polygons, circles, curves, …



- It is possible to go from a vector image to a raster image
  - We'll learn how
- It is very hard to go the other way

# Trade-Offs

- Which format, raster or vector, is easier to:
  - Display?
  - Resize (scale bigger or smaller)?
  - Rotate?
  - Crop (cut bits off at the edges)?

# Obtaining Digital Images

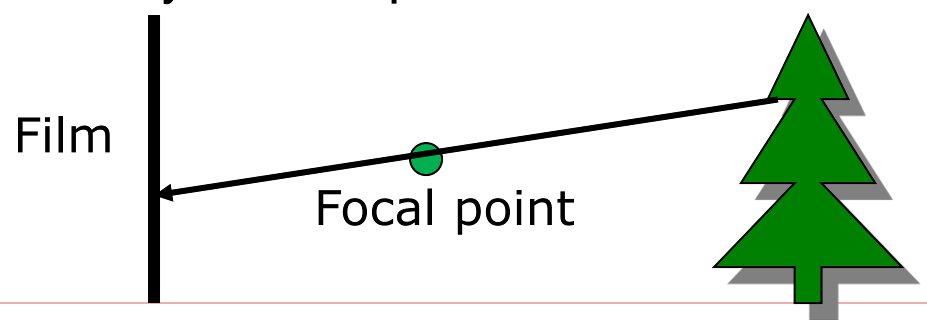
- What are some methods for obtaining a digital image?

# Obtaining Digital Images

- What are some methods for obtaining a digital image?
  - Digital camera
  - Scanning another image
  - Other forms of scanning (e.g. medical)
  - Editing existing digital images
  - Paint or drawing programs
  - Created from abstract data (e.g. math function plot)
  - Rendered from a scene description
  - …

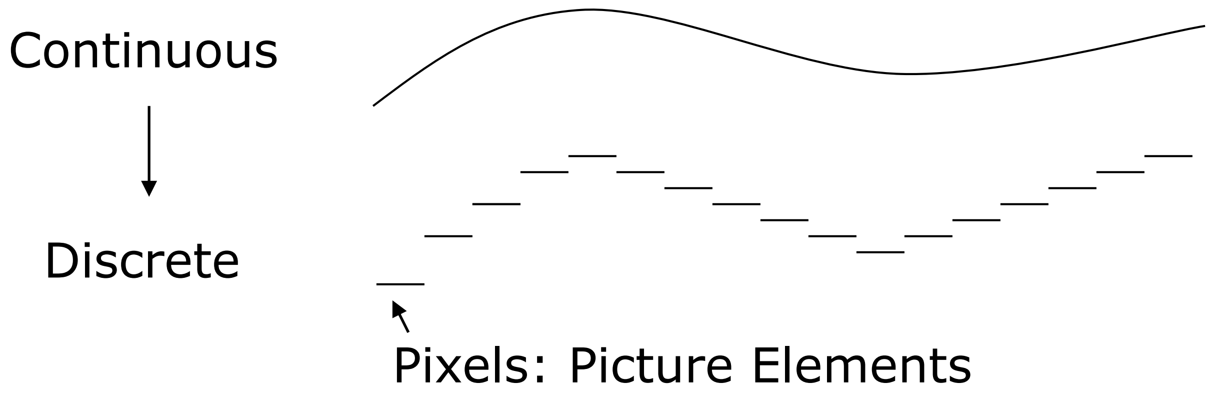
# Ideal Images

- The information stored in images is often continuous in nature
- For example, consider the ideal photograph:
  - It captures the intensity of light at a particular set of points coming from a particular set of directions (it's called *irradiance*)
  - The intensity of light arriving at the camera can be any positive real number, and it *mostly* varies smoothly over space
  - Where do you see spatial *discontinuities* in a photograph?



# Digital Images

- Computers work with discrete pieces of information
- How do we digitize a continuous image?
  - Break the continuous space into small areas, *pixels*
  - Use a single value for each pixel - the *pixel value* (no color, yet)
  - No longer continuous in space or intensity
- This process is fraught with danger, as we shall see



# Discretization Issues

- Can only store a finite number of pixels
  - Choose your target physical image size, choose your resolution (pixels per inch, or dots per inch, dpi), determine width/height in pixels necessary
  - Storage space goes up with square of resolution
    - 600dpi has 4× more pixels than 300dpi
- Can only store a finite range of intensity values
  - Typically referred to as *depth* - number of bits per pixel
    - Directly related to the number of colors available and typically little choice
    - Most common depth is 8, but also sometimes see 16 for grey
  - Also concerned with the minimum and maximum intensity – dynamic range
- What is enough resolution and enough depth?

# Perceptual Issues

- Spatially, humans can discriminate about ½ a minute of arc
  - At fovea, so only in center of view
  - At 0.5m, about 0.1mm ("Dot pitch" of monitors)
  - Sometimes limits the required number of pixels

129 128 125

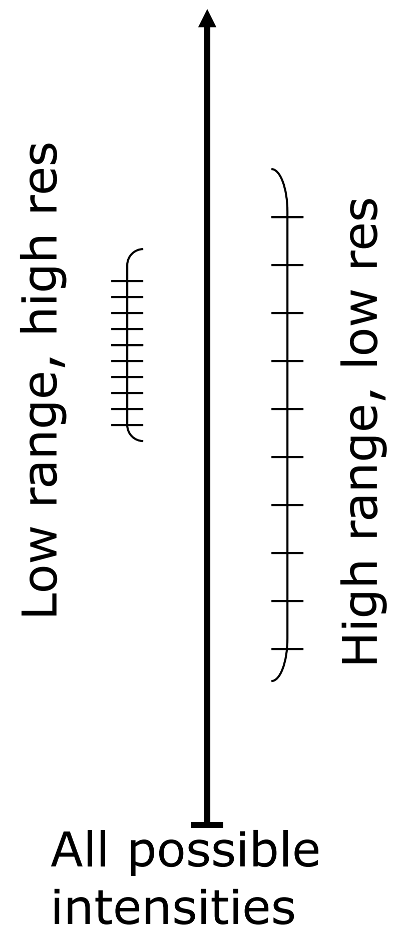
- Humans can discriminate about 8 bits of intensity
  - "Just Noticeable Difference" experiments
  - Limits the required depth for typical dynamic ranges
  - Actually, it's 9-10 bits, but 8 is far more convenient
- BUT, when manipulating images much higher resolution may be required

# Intensity Perception

- Humans are actually tuned to the *ratio* of intensities, not their absolute difference
  - So going from a 50 to 100 Watt light bulb looks the same as going from 100 to 200
- Most computer graphics ignores this, giving poorer perceptible intensity resolution at low light levels, and better resolution at high light levels

# Dynamic Range

- Image depth refers to the number of bits available, but not how those bits map onto intensities
- We can use those bits to represent a large range at low resolution, or a small range at high resolution
- Common display devices can only show a limited dynamic range, so typically we fix the range at that of the display device and choose high resolution



# More Dynamic Range

- Real scenes have very high and very low intensities
- Humans can see contrast at very low and very high light levels
  - Can’t see all levels all the time – use adaptation to adjust
  - Still, high range even at one adaptation level
- Film has low dynamic range around 100:1
- Monitors are even worse
- Many ways to deal with the problem
  - Way beyond the scope of this course

# Display on a Monitor

- When images are created, a *linear* mapping between pixels and intensity is assumed
  - For example, if you double the pixel value, the displayed intensity should double
- Monitors, however, do not work that way
  - For analog monitors, the pixel value is converted to a voltage
  - The voltage is used to control the intensity of the monitor pixels
  - But the voltage to display intensity is *not linear*
  - Similar problem with other monitors, different causes
- The outcome: A linear intensity scale in memory does not look linear on a monitor
- Even worse, **different monitors do different things**

# Gamma Control

- The mapping from voltage to display is usually an exponential function: $I_{display} \propto I_{to-monitor}^{\gamma}$
- To correct the problem, we pass the pixel values through a *gamma function* before converting them to the monitor

$$I_{to-monitor} \propto I_{image}^{1/\gamma}$$

- This process is called *gamma correction*
- The parameter, $\gamma$, is controlled by the user
  - It should be matched to a particular monitor
  - Typical values are between 2.2 and 2.5
- The mapping can be done in hardware or software

# Next Time

- Color