Nifty Assignments 2005
Photomosaics

Rich Pattis
Computer Science Department
Carnegie Mellon University
Pittsburgh, PA 15213
email: pattis@cs.cmu.edu
Overview

- What are Photomosaics ® + Examples
- Programming Concepts
- Programming Options
- Contests and Databases
- Patent Implications

® Photomosaic is a registered trademark of Runaway Technology
What are Photomosaics

- A *mosaic* is a image rendered by a large number of small objects (historically glass and stone)
- In a *photomosaic*, the image is a large picture, which is rendered from smaller pictures
  - Viewed up close, the smaller pictures should be identifiable; from afar they integrate into the image
- Often there is a relationship between the two:
  - An artist rendered from his/her art works
  - A student rendered from his/her classmates
  - A campus landmark rendered from campus photos
  - ...your students will come up with “interesting” combinations, guaranteed to upset someone: its art!
Database Pictures

Each database picture is rectangular, with the same aspect ratio (here photos are square).

Excerpt from a database of CD covers, containing about 1,200 pictures.
A Large Image to Render

Seymour Cray
The Image Rendered by 980 CD Covers

Limiting 10 reuses of any database picture with a minimum distance of 3 between reused pictures

(it helps to squint)
The Assignment

- Write a program that produces photomosaics

  - The minimal program must...
    - Read a database of pictures, displaying it
      - Displaying a database ~ constructing a photomosaic, spatially
    - Read an image to render
    - Render/display image (regions->database pictures)

- Optional (simple vs. complicated assignment)
  - Square vs. Rectangular picture database
  - Sampling size (increase/decrease rendered image size)
  - Built-in vs. plug-in similarity metric(s)
  - Special constraints on rendering: reuse/distance
  - Input/Calculate/Output vs. GUI (via MVC pattern)
Programming Concepts

- **Arrays**
  - 1-d array of Picture objects (in a picture database)
  - 2-d array of pixels ("implicit" in each picture)

- **Loops**
  - Reading all the Picture files in a folder
  - Examining all the pixels in pictures to summarize them
  - Scanning regions in the Image to render
  - Scanning for the "closest" match in the database

- **Using many classes to build an application**

- **Reading Javadoc**
  - The FileSelector and Picture classes
  - Metric interface
Major Provided Components

- FileSelector class
  - Easy interface to file-selection GUI

- Picture class (about a dozen methods total)
  - constructor
  - makeEmptyPicture
  - Overlay (a region by another picture)
  - display
  - get/set Color (at coordinate), other accessors
  - ...

- Metric interface
  - copy
  - makeSummary (overloaded for whole picture/region)
  - distanceTo (another Metric object)
More On (plug-in) Metrics

- I provide one: Intensity
  - Assume a gray scale
  - Computes/Stores 1 average in [0,255]

- Students write at least one: RGB
  - Computes 3 averages (for R, G, B separately) in [0,255]

- I provide a Quad Metric-Decorator
  - Implements Metric, constructor has Metric parameter
  - Is a Metric that duplicates its Metric parameter for each of 4 quadrants (storing more fine-grained data)
  - Summary delegate; distance delegate (sum of distances)
  - new Quad(new Quad(some metric)) provides 16 regions!
Programming Options

- **Constraints**
  - Maximum reuse for database pictures
  - Minimum distance from a picture to any other use of it
  - Spiral rendering from center (vs. top-down/left-right)

- **Color shifting to increase database size/usefulness**

- **Filtering of pictures (e.g., color -> gray scale)**

- **Control Flow**
  - Input/Calculate/Output
  - Menu (but still text) Driven (write as “Model”)
  - Full-blown GUI

  - I have students write the Model as an earlier assignment (providing them with a textual View/Controller), then have them write a full MVC GUI at the end of the semester
Contests: Something for Everyone

- **Best Metric** (on instructor database and image)
- **Best Art**: (on student database image)
  - Use the Pritchard Scale
  - *If the photomosaic's score for perfection is plotted along the horizontal of a graph, and its importance is plotted on the vertical, then calculating the total area of the curve yields the measure of its greatness.*
- **Quick Rendering**: Severe Time Limitations
- **Best Databases** found or created
Database Sources

- **Google the Web**
  - I’ve found cd covers, beer labels, Christmas cards, etc. (tools are available to batch convert pictures to the appropriate sizes)

- **Ask students**
  - They do even better googling
  - If they have digital cameras, create your own

- **Tools: Converting Movies -> Frames**

- **Cheating: shift pictures**
  - redder/greener/bluer
  - Students can easily write filters for the Picture class
Patent Issues

- This topic is a 50 minute talk itself!
- I am not a lawyer; here are some “facts”
  - Creating photomosaics is patented
    - Unlike copyright, there is no fair use
    - Can teach how to make them, but not make them
  - Owner wouldn’t license patent for class use
  - Damages are based on lost income
  - David Evans (UVa) has made photomosaics; published in campus newspaper and on web
  - Photomosaic program on the web
- Will your school support you in a lawsuit?
  Talk to your school’s legal folks
Using Blobs

- Students can use bw/color blobs to test their programs without ever producing a “photomosaic”

Database of 1000 random colors