Manipulating Pixmaps

- Operations of interest:
  - Copying pixmaps
    - glReadPixels: frame buffer to off-screen memory
    - glCopyPixels: frame buffer to frame buffer
    - glDrawPixels: pixmap to frame buffer
    - memCopy: off-screen to off-screen
  - Comparing pixmaps
  - Representing and coloring regions in pixmap

Manipulating Pixmaps

- Data types for pixmaps
  - Bitmap: 1 bit, on or off
  - Gray scale: one byte, values 0-255
  - RGB: 3 bytes (red, green, blue)
  - RGBA: 4 byte (red, green, blue, alpha)
- Declaration of RGB triple:
  ```cpp
  class RGB{
      public: unsigned char r, g, b;
  };
  ```
RGBPixmap Class

- OpenGL convention: pixmap (bottom to top, left to right)
- Add draw, read and copy methods (which use openGL)

Class RGB{
    public: unsigned char r, g, b;
    RGBPixmap(); // constructor
    void setPixel(int x, int y, RGB color);
    RGB getPixel(int x, y);
    void draw() { glDrawPixels(nCols, nRows, GL_RGB,
                          GL_UNSIGNED_BYTE, pixel); }
    void read() { glReadPixels(x, y, nCols, nRows, GL_RGB,
                          GL_UNSIGNED_BYTE, pixel); }
    void copy() { glCopyPixels(... Parameters...); }
    int readBMPFile(char *fname);
    void writeBMPFile(char *fname);
};

Note: refer to Hill fig. 10.3 for full RGBPixmap declaration

Scaling and Rotating Images

- Scaling: want a pixmap that has s times more pixels in x, y
  - s > 1: enlargement
  - s < 1: reduction, information is lost!

- OpenGL scaling:
  - glPixelZoom(float sx, float sy)
    - Sets scale factors for subsequent glDrawPixels command
    - Scaling is about current raster position, pt.
    - Pixel row r and column c of pixmap
    - Drawn as rectangle with bottom left current screen coordinates
    - Draws (pt.x + sx*r, pt.y + sy*c)
  - 90, 180 and 270 degree rotations:
    - Copy one pixmap to another doing matrix transposes
  - General rotations:
    - Affine transform of pixmap points to get new pixmap
Combining Pixmaps

- Two pixmaps A and B combined pixelwise to form third pixel C.
- i.e. \( C[i][j] = A[i][j] \odot B[i][j] \)
- Averaging:
  - \( C[i][j] = \frac{1}{2} \cdot (A[i][j] + B[i][j]) \)
- Subtraction:
- Generalized weighting:

Example:

\( A = (14, 246, 97), \ B = (82, 12, 190), f = 0.2 \)

\( C = (27, 199, 115) = 0.8 \ A + 0.2 \ B \)

Combining Pixmaps

- Generalized weighting:
- Example:
  - \( A = (14, 246, 97), \ B = (82, 12, 190), f = 0.2 \)
  - \( C = (27, 199, 115) = 0.8 \ A + 0.2 \ B \)
- Question: How to dissolve image A into B?

Alpha Channel and Image Blending

- Even more generalized weighting = blending/compositing
- Blending:
  - draw partially transparent image over another
  - Add 4th component, alpha value (A) to RGB
  - Interpretation: alpha specifies how opaque each pixel is
  - Transparent (A = 0), Total opacity (A = 255)
  - Alpha most frequently used in scaling colors

```cpp
class RGB{
    public: unsigned char r, g, b, a;
};
```

Alpha Channel and Image Blending

- Alpha channel: series of alpha values in a pixmap
- openGL alpha blending: `glBlendFunc( )`
- Other alpha blending applications:
  - Simulating Chromakey: forcing certain colors to be transparent
  - Applying paint with a paintbrush: apply percentage of new color with each mouse stroke
References

- Hill, chapter 10