Computer Vision Lab #1
Pointwise Image Processing

Prof. Ricardo Fabbri, Ph.D.*

Polytechnic Institute at the Rio de Janeiro State University
http://wiki.nosdigitais.teia.org.br/CV

November 7, 2013

Abstract

- Familiarize with the Scilab image processing toolbox (SIP).
- Implement pointwise image processing operations such as thresholding, sampling, quantization, and enhancements.

The images you will need for the lab can be downloaded from the course website.

1 The Scilab Image Processing toolbox (SIP)

Initial tasks:

1. Install Scilab and SIP [1, 2]. You will need Linux or OSX.
2. Go through the SIP demos and make sure you understand the `imshow, imread, imwrite`, and the other commands showcased in the demo.

```
exec(SIPDEMO);
```

3. Read our paper about the toolbox and type in the commands [1, 2]. Make things work in the desired way even if they don't, since your version of SIP and Scilab may be different.

2 Thresholding

2.1 Basics

Using the white blood cell image 1 as seen below, threshold it to make any non-cell area black. To do this, you will need to check each pixel in the image, and if

*Based on Image Understanding 2011 lab material from Ben Kimia, Brown University

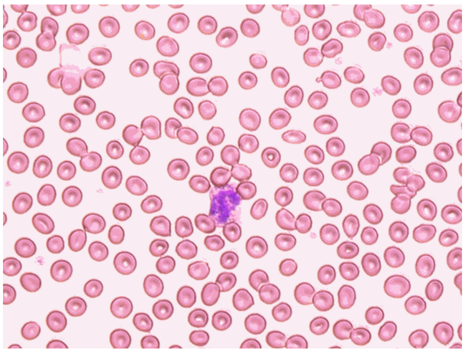

Figure 1: White blood cells

Figure 2: Snowman image to practice thresholding. Notice JPEG compression artifacts as well.

it is over τ_0 (where τ_0 is the threshold you have chosen) set the pixel's intensity to 0, otherwise, set to 1. You will need to play with τ_0 to find its optimal value. You will probably want to pass it into the function as a parameter.

Threshold code snippet

```
// This is sketchy code - devise your own implementation
for i=1:height
  for i=1:width
      // Ceck pixel value of I(i,j) against threshold
      // Act accordingly
  end
end

// Please also provide faster implementations than explicit for loops.
```

2.2 Bilevel Thresholding

Using the snowman image 2, perform bilevel thresholding. i.e. set all pixels with intensity between τ_0 and τ_1 to 255 and set all pixels less than τ_0 to 0 and all pixels greater than τ_1 to 0. The snowman is the region of interest, you are thresholding for (where bob is, should be all white and the rest should be black).

Figure 3: A poor contrast image

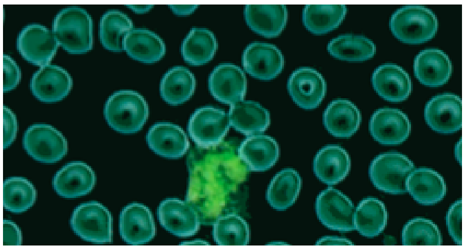

Figure 4: Sample section of one possible color inversion of the image in figure 1.

3 Contrast / Inversion

3.1 Brightness and Clipping

Increase the brightness of the image 3 by 10% or decrease the brightness of the original by 10% by adding/subtracting a value ($I_0 = 25$) from each pixel intensity. Try different values of I_0.

Hint: Make sure you deal with values that fall outside the range of the image (clipping).

3.2 Color Image Inversion

Invert the white blood cell image 1 in RGB. You should get something around the lines of the sample in figure 4

Hint: Make sure pixel intensities are not negative

4 Quantization: Reducing the number of colors

The current cell image 1 has 256 possible values of pixel intensity (but normalized from 0 to 1 `double` type in Scilab). Reduce this to 8 levels (again normalized from 0 to 1). Make some observations about the quantization levels and image quality. Do this by grouping intensity regions, *i.e.*, set the intensity of all pixels with current intensity between 0-31 to 16, between 32-63 to 48, 64-95 to 80,...etc. How would you do this for 16 levels?

5 Sub-Sampling

Reduce the sampling of the white blood cells in image 1 by skipping every other element.

Sub-Sampling code snippet

```
// This is sketchy code - devise your own implementation
for i from 1 to width in steps of 2
   for j from 1 to height in steps of 2
     outputimage (i, j)= inputimage(i, j)

// Please also provide faster implementations than explicit for loops.
```

6 Playing with a Camera

This lab will also give you a chance to become familiar with a digital video camera. Use your own digital camera, smartphone, (or come by the visual computing and computer graphics lab (room 110 at IPRJ/UERJ) to set up a time to use the lab's digital video cameras such as the Sony PS3 Eye or an advanced surveillance camera) to take a few pictures or video clips, and upload them to the computer. You will use this imagery for the next set of problems.

6.1 Sensor noise

1. Take at least 20 images (or a video with at least 20 frames) with your digital camera of the *same* scene under the *same* illumination condition. Make sure *nothing* changes in between taking these images, so using a tripod a camera hooked up to your laptop is best.

2. Show the mean image and the standard deviation image. Scilab functions such as `mean()` and `std()` are available

3. Show the maximum difference from this mean image. How big is this? Does this depend on the mean?

4. Pick one pixel across the 20 images, and plot the histogram, `imhist()`. Discuss what the distribution of pixel intensity looks like. *Why does it look like that?*

5. Repeat under a different lighting condition (light vs dark).

References

[1] INRIA, “The Scilab numerical programming environment,” www.scilab.org.

[2] R. Fabbri, O. M. Bruno, and L. F. Costa, “Scilab and SIP for image processing,” *Arxiv preprint arXiv:1203.4009*, 2012.