SOLUTIONS MANUAL FOR

A COMPUTATIONAL INTRODUCTION TO DIGITAL IMAGE PROCESSING

SECOND EDITION

by

Alasdair McAndrew
Victoria University
Melbourne, Australia

CRC Press
Taylor & Francis Group

SOLUTIONS MANUAL FOR
A COMPUTATIONAL INTRODUCTION TO DIGITAL IMAGE PROCESSING
SECOND EDITION

by

Alasdair McAndrew
Victoria University
Melbourne, Australia

CRC Press
Taylor & Francis Group
Boca Raton London New York

CRC Press is an imprint of the
Taylor & Francis Group, an **informa** business

CRC Press
Taylor & Francis Group
6000 Broken Sound Parkway NW, Suite 300
Boca Raton, FL 33487-2742

Printed on acid-free paper
Version Date: 20160201

International Standard Book Number-13: 978-1-4822-4738-1 (Ancillary)

Visit the Taylor & Francis Web site at
http://www.taylorandfrancis.com

and the CRC Press Web site at
http://www.crcpress.com

Solutions manual for

A Computational Introduction to Digital Image Processing

Alasdair McAndrew
Alasdair.McAndrew@vu.edu.au

Note that many of the exercises ask you to apply an imaging method to an image, or to several images, and compare the results for different parameters of the method. As such, these questions have no "correct" answer—they are all about your own observation and interpretation, and so no solutions are given. For many other questions the answers can be checked against your computer system, and so many other questions are also not given solutions.

1 Introduction

4. Just to get you started, here are a few applications:
   - Astronomy: enhancement of telescope images, denoising of images, "stitching" of images (to create a panorama), comparing images to see shifts of objects against the background (this is how the planet Pluto was found in 1930)
   - Sport: analysis of plays (speed of a ball, for example), training, trajectories of balls, human biomechanics

2 Images files and file types

2. The commands to use would be

```
>> im = imread(imagefile);
>> [x,map] = rgb2ind(im);
```

MATLAB/Octave

3. Some of the images included with `skimage` are

File name	Type	Size	Description
`brick.png`	Gray scale	512×512	Brick paving
`chelsea.png`	(True) Color	451×300	A cat
`grass.png`	Gray scale	512×512	Grass!
`ihc.png`	Color	512×512	Cells
`moon.png`	Gray scale	512×512	Close-up of the lunar surface
`page.png`	Gray scale	384×191	Image of part of a book page
`text.png`	Gray scale	448×172	An oblique view of some written text

4. There are a lot of images in the `/images/imdata/` directory. For example:

File name	Type	Size	Description
`board.tif`	True color	306×648	A circuit board
`forest.tif`	Indexed color	447×301	An ancient forest
`circles.png`	Binary	256×256	Several circles all the same size
`circuit.tif`	Gray scale	272×280	Micrograph of part of a circuit

5. A PNG image of 60918 bytes (`wombats.png`) becomes a JPG image of 7682 bytes or a PGM image of 242522 bytes.

7. This is a 300×246 grayscale image.

8. This is a color image, of size 16×16.

3 Image Display

1. The date type should be unsigned 8-bit integer.

2. Australia's tallest native bird! (The *heaviest* bird is the Southern Cassowary, which is however shorter than the emu).

3. (a) in this case there is no change: the initial values are all unsigned 8 bit integers. (b) The index matrix has only 64 distinct entries, between 0 and 63; the final image matrix has only 55 distict entries, but spread out between 32 and 239.

4. Since the cameraman image is already of type `uint8` the function will cause no change.

5. (a) the cameraman image becomes unrecogizable at 16×16; (b) the emu at 32×32; (c) the blocks at 16×16; (d) the buffalo at 16×16.

6. All images are quite recognizable at 2 gray scales, except perhaps the blocks image, which is less "natural" than the others. Its best minimum seems to be 4 gray scales.

7. Some newspapers still print in black and white only, using half-toning for the appearance of grayscale. So you may only see black and white when magnified.

9. You can experiment with dither matrices on a 256×256 image `im` as follows:

```
>> d = reshape(16*(randperm(16)-1),4,4)
>> imshow(im > repmat(d,[64,64]))
```

MATLAB/Octave

or

```
In :  d = np.reshape(np.random.permutation(16)*16,[4,4])
In :  io.imshow(b>np.tile(d,[64,64]))
```

Python

What properties of the dither matrix are necessary for a good result?

Note that the matrix D_2, divided by 16, is

$$E = D_2/16 = \begin{pmatrix} 0 & 8 & 2 & 10 \\ 12 & 4 & 14 & 6 \\ 3 & 11 & 1 & 9 \\ 15 & 7 & 13 & 4 \end{pmatrix}$$

Consider the matrix

$$M = D/64 = \begin{pmatrix} 0 & 2 \\ 3 & 1 \end{pmatrix}$$

Then we have three other matrices

$$M + 4 = \begin{pmatrix} 4 & 6 \\ 7 & 5 \end{pmatrix} \qquad M + 8 = \begin{pmatrix} 8 & 10 \\ 11 & 9 \end{pmatrix} \qquad M + 12 = \begin{pmatrix} 12 & 14 \\ 15 & 13 \end{pmatrix}$$

Notice how E is constructed of a sort of "interleaving" of these four matrices $M, M+4, M+8, M+12$.

10. With only a few grayscales caused by quantization, the contouring will start to be apparent in the dithered output. However, dithering may also cover up false contouring. Check the thylacine image at 8 gray levels.

4 Point processing

1. Multiplication by two (for `uint8`):

0	1	2	3	4	5	⋯	125	126	127	128	129	⋯	252	253	254	255
0	2	4	6	8	10	⋯	250	252	254	255	255	⋯	255	255	255	255

Complements:

0	1	2	3	4	5	⋯	125	126	127	128	129	⋯	252	253	254	255
255	254	253	252	251	250	⋯	130	129	128	127	126	⋯	3	2	1	0

2. Since `b` is of type `uint8`, the command `b/64` will produce only integer outputs, and produce only values 0–4 (MATLAB and Octave), or 0–3 (Python). Multiplying by 64 will produce again only 5 or 4 outputs, so the result will be quantized.