Computer Vision: Answers to Selected Exercises

Linda Shapiro
The University of Washington
Seattle, Washington
shapiro@cs.washinton.edu
and
George Stockman
Department of Computer Science
Michigan State University
East Lansing, MI
stockman@pixel.cps.msu.edu

This document is created for instructors using the textbook **Computer Vision** *by Shapiro and Stockman. Hopefully, it will help in selection of exercises to be assigned and in assessment of answers. These answers should not in general be made available to students. They will benefit most by trying the exercises themselves without the benefit of answers from someone else. This will be needed in their career when they encounter new problems without their teachers.*

Many but not all of the exercises in the textbook have been answered by the authors when the book was written. Some were done by students studying from the book and some others were done by lower division undergraduate honors students. Some questions, therefore, were answered by several different people. It was important to not only answer the questions, but also to ascertain their clarity and difficulty. The answers compiled here are a collection of the authors' work and that of some students. Answers are brief when the question is considered straightforward; otherwise, elaboration is given.

Some answers are missing due to three reasons—first, some are meant as programming projects; second, we frankly have not gotten to writing up answers to all exercises; third, some are meant to require open-ended essay answers or outside research and believe it is counter productive for us to provide simple answers based on our own experience or opinion.

Chapter 1

Introduction

Exercise 1 Hole counting

Consider the following three images, which are 4x5, 4x4, and 4x5 respectively.

1	1	1	1	1
1	0	1	0	1
1	0	1	0	1
1	1	1	1	1

1	1	1	1
1	1	0	1
1	0	1	1
1	1	1	1

1	1	1	1	1
1	0	1	0	1
1	0	0	0	1
1	1	1	1	1

In scanning for corner patterns, 12, 9, and 12 2x2 neighbors are checked by Algorithm **??** for the three images above. Each 2x2 neighborhood matches one of these patterns 'e', 'i','n', for external corner, internal corner, and neither. (a) For each of the three images, how many of each 2x2 pattern are there? (b) Does the holecounting formula work for all three images?

Answer:

(a)

The image at the left has 12 2×2 neighborhoods. #e = 8; #i = 0; #n = 4.
The center image has 9 neighborhoods; #e = 6; #i = 0; #n = 3.
The image at the right has 12 neighborhoods; #e = 6; #i = 2; #n = 4.

(b)

The formula $\#holes = (\#e - \#i)/4$ works on the left and right images but not on the center image. (The reason is that the regions of 0's are not *4-connected*. This corresponds to the real-world situations where the resolution is too coarse or where there are two holes tangent or overlapping, or where one hole has a neck in it.)

Exercise 2 How many pixels per hole?

Consider the application of counting holes in truck crossbars at a more detailed level. Suppose that the area of the crossbar that is imaged is 50 inches long and 10 inches wide and suppose that this area almost fills up a digital image of 100 rows and 500 columns of pixels. Suppose a particular bolt hole in the crossbar is 1/2 inch in diameter. What would you expect the radius and area of its image to be in terms of pixels?

Answer:

Assuming that 50 inches images onto 500 pixels, each pixel has nominal resolution of 0.1×0.1 inches. This would make the diameter of a 0.5 inch hole image to 5 pixels. Theoretically, we expect the area of the hole image to be $\pi(d/2)^2$, which is about 19.625 pixels. However, quantization effects will cause variability in our observation (measurement). Ideal placement of the hole relative to the sensing array and ideal thresholding so that greater than 0.5 pixel seeing hole maps to 0 yields an area as large as 21 and as small as 16 pixels. The experiment pictured below was done with graph paper and a metal washer of diameter 5 pixels.

Exercise 3 Imaging coins as pixels.

This problem is related to the one above. Obtain some graph paper (0.25 inch squares would be good) and a quarter. Randomly place the quarter on the graph paper and trace its circumference: do this five times. For each of the five placements, estimate the area of the image of the quarter in pixel units (a) by deciding whether a pixel is part of the quarter or not (no fractions) and (b) for each pixel cut by the circumference, estimate to the nearest tenth of a pixel how much of the pixel is part of the image of the quarter. After doing these measurements, compute the mean and standard deviation of the image area separately for methods (a) and (b).

Answer:

An experiment was done by John Ross using graph paper with 8×8 squares per square inch (see Figure below). For part (a) he only counted a pixel in the area when it was entirely contained within the disk, obtaining values 35,36,37,35,34 for a mean of 35.4 pixels and standard deviation of $\sigma = 1.14$. Modifying his data by including a pixel whenever 0.6 or more of it is inside the disk, yielded a mean of 46.8 and $\sigma = 1.94$. For part (b), he estimated the five areas as 46.5, 46.5,46.6, 47.2,47.7 obtaining a mean of 46.9 and $\sigma = 0.534$. So, by estimating the parts of area to the nearest 0.1 pixel, the mean is almost the same as in (a), but σ is smaller since the 5 estimates agree much more closely.

Exercise 4 Other problem areas.

Describe a problem different from those already discussed for which machine vision might provide a solution. If you do not have a special application area in mind, choose one for the moment. What kind of scenes would be sensed? What would an image be like? What output would be produced?

Answer:

There are many possible answers. One student wanted to remove commercials from our TV signal. Equipment can now be bought to do this, but how does it work? Another student wanted to draw a sketch of an electronic circuit, scan that into an image file, and then have a program translate the sketch into a circuit representation that could then be simulated, or perhaps be input to actual VLSI manufacturing processes. (Author Stockman had multiple senior design teams succeed in this.)

Exercise 5 Examining problem context.

Problems can be solved in different ways and a problem solver should not get trapped early in a specific approach. Consider the problem of identifying cars in various situations: (a) entering a parking lot or secured area, (b) passing through a toll gate, (c) exceeding the speed limit. Several groups are developing or have developed machine vision methods to read a car's license plate. Suggest an alternative to machine vision. How do the economic and social costs compare to the machine vision approach?

Answer:

Transponders are now used for every railroad car in the US and many trucks are also carrying them. These are tiny radios implemented via IC technology. When querried, the transponder can respond with an identification and even a description of the cargo. These devices are used on various toll roads so that the users employing them do not have to stop to pay toll; they are billed according to the account number provided by the transponder. Such devices can be used for permission to enter parking lots as well. (Many farm animals and pets have such devices implanted under their skin for identification purposes. Such a device can also allow a particular dairy cow to enter a particular stall where she will be fed a particular formula!) Many Americans would object to this technology because it restricts their privacy. Machines could know their locations at particular times. This information could also be used for wrong purposes by criminals.

Exercise 6

Identify some defects remaining in the right image of Figure 1.8 and describe simple neighborhood operations that will improve the image.

Answer:

Some bacteria have holes and several have ragged edges. We should be able to improve on these faults by changing white pixels that have 5 or more black neighbors. Perhaps more than one iteration of this process would be good. Such operations are considered in more detail in Chapter 3.

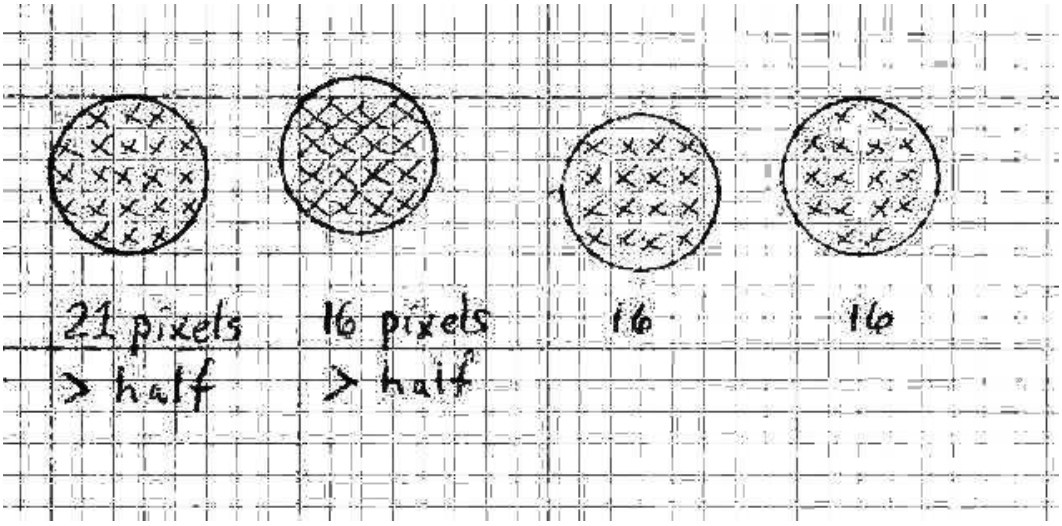


Figure 1.1: Examples of binarizing the image of a disk of diameter 5 pixels.

Exercise 7

What invariant features of the following objects enable you to recognize them in rain or sunshine, alone or alongside other objects, from the front or side: (a) your tennis shoes, (b) your front door, (c) your mother, (d) your favorite make of automobile?

Answer:

(a) Our tennis shoes will appear to have the same color as long as we have a minimal amount of light. Their shape will be seen to change depending on their orientation, however, there are a few distinct shapes, such as top, front, side. In general, most views would allow us to see that they were some type of *container* about big enough to hold our feet.

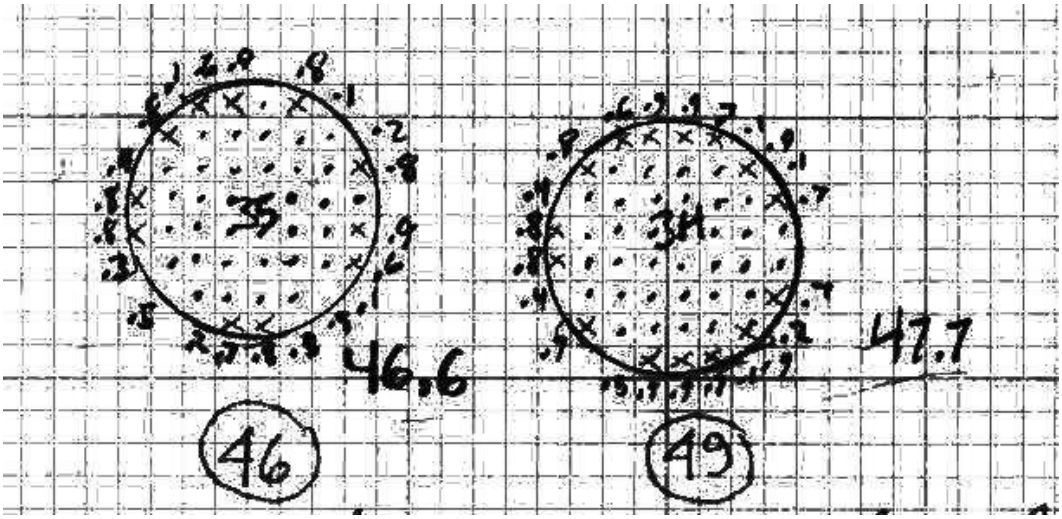


Figure 1.2: Examples of binarizing the image of a quarter.

Chapter 2

Imaging and Imaging Devices

Exercise 1 examination of a CCD camera

If you have access to a CCD camera, obtain permission to explore its construction. Remove the lens and note its construction; does it have a shutter to close off all light, does it have an aperature to change the size of the cone of rays passing through? Is there a means of changing the focal length – e.g. the distance between the lens and CCD? Inspect the CCD array. How large is the active sensing area? Can you see the individual cells – do you need a magnifying glass?

Answer:

Digital cameras are now quite common in the consumer market so one should be able to get hold of a camera used for teleconferencing or for taking digital still photos. Some of these devices will be easy to open and examine. Cameras used for machine vision are typicaly easy to disassemble and examine. Magnification will be needed to see individual sensing cells.

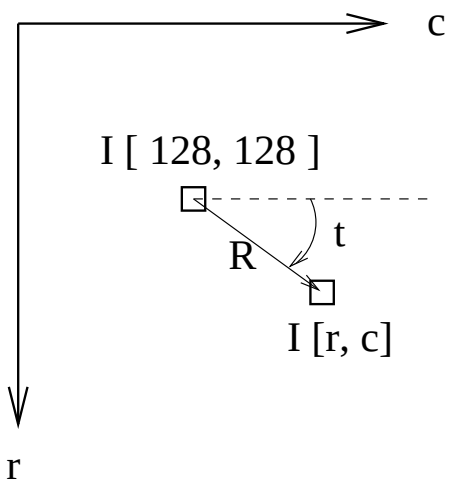


Figure 2.1: Defining a sequence of pixels on the circumference of a circle.

Exercise 2

Suppose that an analog clock is to be read using a CCD camera that stares directly at it. The center of the clock images at the center of a 256 x 256 digital image and the hour hand is twice the width of the minute hand but 0.7 times its length. To locate the images of the hands of the clock we need to scan the pixels of the digital image in a circular fashion. (a) Give a formula for computing $\mathbf{r}(t)$ and $\mathbf{c}(t)$ for pixels $\mathbf{I}[\mathbf{r}, \mathbf{c}]$ on a circle of radius R centered at the image center $\mathbf{I}[\mathbf{256}, \mathbf{256}]$, where t is the angle made between the ray to $\mathbf{I}[\mathbf{r}, \mathbf{c}]$ and the horizontal axis. (b) Is there a problem in controlling t so that a unique sequence of pixels of a *digital circle* is generated? (*c) Do some outside reading in a text on computer graphics and report on a practical method for generating such a digital circle.

Answer:

(a) Refer to the figure below. Row $r(t) = round(128 + R\ sin\ t)$ and column $c(t) = round(128 + R\ cos\ t)$.
(b*) To do a useful circular scan, the pixel sequence (r_j, c_j) should continuously pass through neighbors so that there are no gaps and no repeated pixels. Some graphic texts discuss generation of such circles, which must be done very efficiently if part of real-time rendering. Our task is a little easier, however, since we should be able to generate a circle once and save the sequence. One could use Δt corresponding to a fraction of a pixel (say $R\Delta t \approx 0.3 pixel$) and create a circumference by iteration. Repeated pixels should be culled from the sequence. Also, the sequence should be checked to pass through neighbors. We should replace two consecutive 4-neighbors making a right angle with a single 8-neighbor move.

Exercise 3

Assume that a human eyeball is 1 inch in diameter and that 10^8 rods and cones populate a fraction of $1/\pi$ of its inner surface area. What is the average size of area covered by a single receptor? (Remember, however, that foveal receptors are packed much more densely than this average, while peripheral receptors are more sparse.)

Answer:

The surface area of a sphere of radius 1 inch is $4\pi r^2$, with $r = 1/2$. Receptors cover only $1/\pi$ of this area, so we actually have an area of 1 square inch containing 10^8 receptors. Thus, on average we have 10^{-8} square inches per receptor, which corresponds to a little square with side of 10^{-4} inches.

Exercise 4 On variation of area

Consider a dark rectangle on a white paper that is imaged such that the rectangle measures exactly 5.9 x 8.1 pixels on the real image. A binary image is to be produced such that pixel values are 0 or 1 depending upon whether the pixel sees more object or more background. Allow the rectangle to translate with its sides parallel to the CCD rows and columns. What is the smallest area in pixels in the binary image output? What is the largest area?

Answer:

We assume that any pixel that is covered more than 0.5 by the image of the rectangle is counted as 1 unit of area. The least area is 40 pixels, which results by placement so that the width is 0.45+5+0.45 pixels and the length is x+7+(1.1-x) pixels. Clearly either x or 1.1-x must be greater than 0.5, so let's take x=1 and 1-x=0.1. The greatest area is 54 pixels. This can be obtained by making x=0.55 in the above argument, forcing length to be 9 pixels. The width can easily be observed as 6 pixels (5+0.9). We really should carefully consider the corner pixels. The above placement makes the image of two of the rectangle's corners as 0.55 by 0.9 pixels, which is 0.5445 or just above threshold!

Exercise 5 On loss of thin features

Consider two bright parallel lines of conductor on a printed circuit board. The width of each line is 0.8 pixels on the image plane. Is there a situation where one line and not the other would disappear in a binary image created as in the exercise above? Explain.

Answer:

It is easy to imagine that one line would image across a single row of pixels and thus be observed, since 0.8 of each pixel would be covered. Suppose that the other line parallel to this first one images precisely across two rows so that pixels in each row observe only 0.4 of the object. Thresholding is likely to drop this feature and the second line would be invisible. This emphasizes the *two pixel rule*—to be sure that a feature is detected, engineer the image resolution so that the smallest feature to be detected is at least 2 pixels across.

Exercise 6 sensing paper money denominations

Consider the design of a sensor for a vending machine that takes U.S. paper money of denominations $1, $5, $10, and $20. You only need to create a representation for the recognizer to use; you need not design a recognition algorithm, nor should you be concerned about detecting counterfeit bills. (Be sure to obtain some samples before answering.) Assume that a linear CCD array must be used to digitize a bill as it enters the machine. (a) What kind of lens and what kind of illumination should be used? (b) How many pixels are needed in the linear array? Explain.

Answer:

This is a *design problem.* Students should acquire a sample of each bill. They should then decide the size in pixels of the images of the digits that is needed in order to represent them for recognition. This might be done by scanning each bill, cropping out the digits, and then successively cutting resolution to find one coarse enough to be efficient but expressive enough for recognition. The typical flatbed scanner is a good model for the hardware that could be used in the vending machine.

Exercise 7 Creating a PPM picture

A color image can be coded in PBM format by using the magic value "P3" and three (R,G,B) intensity values for each pixel, similar to the coded monochrome "P2" file shown in Figure 2.12. Using your editor, create a file bullseye.ppm encoding 3 coincentric circular regions of different colors. For each pixel, the three color values follow in immediate succession, rather than encoding three separate monochrome images as is done in some other formats. Display your picture using an image tool or web browser.

Exercise 8

Locate an image viewing toolset on your computer system. (These might be available just by clicking on your image file icon.) Use one image of a face and one of a landscape; both, should originally be of high quality, say 800 x 600 color pixels, from a flatbed scanner or digital camera. Convert the image among different formats – GIF,TIFF,JPEG etc. Record the size of the encoded image files in bytes and note the quality of the image; consider the overall scene plus small details.

Answer:

It's better to start with a TIFF or GIF where no compression was used in scanning. These files will typically be large. Converting to JPEG can considerably reduce the resulting image file size. Students should examine the converted image to see if edges or colors have been blurred.

Exercise 9 * JPEG study

(a) Research the JPEG compression scheme for 8×8 image blocks. (b) Implement and test the DCT scheme in a lossless manner (except for possible roundoff error). (c) Create a lossy compression using some existing image tool. (d) Using the 64 coefficients from lossy compression, regenerate an 8×8 image and compare its pixel values with those of the original 8×8 image.

Exercise 10 observe as an artist

Consciously make observations in two different environments and sketch some of the cues discussed above. For example, try a busy cafeteria, or a city street corner observed from a height of a few floors, or a spot in a woods.

Exercise 11

Think about some of your own dental X-rays: what was bright and what was dark – the dense tooth or the softer cavity? Why?

Answer:

The X-ray machine penetrates material with X-rays and measures the energy penetrating through to sensing elements (film or digital) on the far side. Dense material impedes transmission, whereas non dense material allows transmission of more energy. Processing of the X-ray film inverts the energy relation so that dense materials appear brighter than nondense material. Bones are brighter than soft tissue and good tooth is brighter than a softer decay. Author Stockman has worked with colleagues who studied the human body by X-raying cadavers that had small tungsten carbide spheres implanted in important locations in the skeleton. These appear as bright circles in the processed X-ray film.

Chapter 3

Binary Image Processing

Exercise 1 Efficiency of counting objects

What is the maxiumum number of times that procedure *count_objects* examines each pixel of the image? How can procedures *external_match* and *internal_match* be coded to be as efficient as possible?

Answer:
The procedure clearly raster scans the entire image, visiting each pixel exactly once; so, a simple answer is 1. Looking deeper, at each pixel there is a check for a match to an external corner, which requires that 3 other pixels be examined in addition to the current $[r, c]$ to check against a single external corner pattern. Since there are 4 such patterns, a maximum of 16 pixel examinations could be done. Similarly, 16 pixel examinations could be done for internal patterns, thus yielding that 32 pixels might be examined for each raster position! The second part of the question sheds more light on the issue.

The procedures need not actually compare each of the image neighborhood patterns with the 8 corner patterns, instead they need only count the number of 1-pixels. An external corner has a total of one 1-pixel and an internal corner has a total of 3 - it doesn't matter where the 1's are located, just their number. For completeness, consider how many patterns there are with 0,1,2,3 and 4 1-pixels. There are $2^4 = 16$ distinct 2x2 neighborhoods; these fall into the following classes: (0 1-pixels, 1 pattern), (1 1-pixel, 4 patterns), (2, 6), (3, 4), (4, 1).

Exercise 2 Driving around corners

Obtain some graph paper to represent a pixel array and blacken some region of connected squares (keep it small at first). The blackened squares correspond to the foreground pixels and the empty squares correspond to the background. Imagine that the pixels are all city blocks and you are driving around the blackened region in a clockwise direction. Do your right turns correspond to E corners or I corners? What about left turns? Is there a relationship between the number of left turns and the number of right turns made in driving the complete perimeter? If so, what is it? In driving the entire perimeter, did you ever cross over or touch a previously visited intersection? Is that ever possible? Why or why not? Before answering, consider the case of only two blackened blocks touching diagonally across a single shared intersection. Do your left-right counting rules still hold? Does the object-counting formula still hold?

Exercise 3 Relabeling

Because equivalent labels are merged into one equivalence class, some of the initial labels from Pass 1 are lost in Pass 2, producing a final labeling whose numeric sequence of labels often has many gaps. Write a relabeling procedure that converts the labeling to one that has a contiguous sequence of numbers from 1 to the number of components in the image.

Answer:

1. Form a list $\mathbf{L}[]$, or array, containing the labels used after Pass 2. This list has length K and largest label N with $K \leq N$.
2. Sort the list $\mathbf{L}[]$ into ascending order. After sorting, we will have the needed translation: if label $m = \mathbf{L}[j]$ then recode m with j.
3. Since we do not want to have to search $\mathbf{L}[]$ for every image pixel, make a translation table $\mathbf{T}[]$ of length N, such that $\mathbf{T}[m] = j$ for all j used by list $\mathbf{L}$.
4. Visit every pixel in the image once and translate using table $\mathbf{T}$; if the pixel has label m, then replace it by $\mathbf{T}[m]$.

Exercise 4 Labeling Algorithm Comparison

Suppose a binary image has one foreground region, a rectangle of size 1000 by 1000. How many times does the recursive algorithm look at (read or write) each pixel? How many times does the classical procedure look at each pixel?

Exercise 5 Run-Length Encoding

Design and implement a row-by-row labeling algorithm that uses the run-length encoding of a binary image instead of the image itself and uses the LABEL field of the structure to store the labels of the runs.

Exercise 6 Using elementary operations of binary morphology

A camera takes an image I of a penny, a dime, and a quarter lying on a white background and not touching one another. Thresholding is used successfully to create a binary image B with 1 bits for the coin regions and 0 bits for the background. You are given the known diameters of the coins D_P, D_D, and D_Q. Using the operations of mathematical morphology (dilation, erosion, opening, closing) and the logical operators AND, OR, NOT, and MINUS (set difference), show how to produce three binary output images: P, D, and Q. P should contain just the penny (as 1 bits), D should contain just the dime, and Q should contain just the quarter.

Answer:

We can create three structuring elements that are disks with diameters corresponding to the quarter, penny, and dime. Denote them as DISK(Dq), DISK(Dp), and DISK(Dd) and note that the diameters should be the largest diameter so that the disk will fit just inside of the image of the given coin. The image of a quarter, Q, can be obtained by opening I by DISK(Dq), which will essentially delete pixels of the smaller penny and dime and preserve an approximation of the image of the quarter. Then form $I2 = I\ MINUS\ Q$, which will have no trace of the quarter but preserve the 1 bits of the penny and dime. We repeat the process to isolate the penny in image P as I2 opened by DISK(Dp). Then form $D = I2\ MINUS\ P$.

Exercise 7 Structuring element choices

Sternberg used a ring structuring element to detect the centers of the holes in the gear-tooth inspection task. If your system only supports disk and box structuring elements, what can you do to detect the centers of the holes?

Exercise 8 Morphological processing application

Suppose a satellite image of a region can be thresholded so that the water pixels are 1's. However, bridges across reivers produce thin lines of 0's cutting across the river regions. a) Describe how to restore the bridge pixels to the water region. b) Describe how to detect the thin bridges as separate objects.

Answer:

(a) Assume that the widest bridge is W pixels across. Close the image using DISK($2W$), which will extend the boundary of the water region to change the 0 bridge pixels neighboring the 1 water pixels into 1 pixels.

(b) The bridge regions themselves can be extracted by first subtracting the original thresholded image from the closed result obtained in part (a). This result should then be opened by DISK(D), where D is just small enough to be smaller than the smallest bridge width, yet large enough to erase tiny noise regions that may have resulted from the original thresholding of I.

Exercise 9 Using the area property

The gear-tooth example was designed to use only morphological and logical operations that could be rapidly executed on a specially-designed machine. Given that we are looking for larger-than-normal gaps between the teeth, how could the detection be performed in a way that minimizes the morphological operations for general purpose machines on which they do not run rapidly?

Exercise 10 Region from perimeter

Describe an algorithm to generate a binary image of a region without holes, given only its perimeter.

Answer:

1. Visit the entire perimeter list in order to compute the min and max row and column coordinates for perimeter pixels.
2. Create an image B[r,c] of all zero pixels and large enough to contain all of the perimeter points, or of some larger size related to the application. Also, create the image so that B[0,0] is outside of the bounding box determined in step 1.
3. Revisit the perimeter set: for each (Rj, Cj) in P, set B[Rj, Cj] = 1.
4. Recursively color the background pixels color 2, beginning from B[0,0]. The interior of the region within the perimeter will be 0 pixels.
5. Recolor the entire image using the simple label translation: color 2 becomes 0; color 1 stays 1; color 0 becomes 1.

Exercise 11 Area from perimeter

Design an algorithm to compute the area of a region without holes, given only its perimeter. Is it possible to perform the task without regenerating the binary image?

Answer:

An obvious method of computing area is to lay out the region in an image as done in the previous problem and simply count the number of 1-pixels resulting in the last step. It is possible to compute the area of a region without holes from only its perimeter, and there are instruments that can do this on a map. We can accumulate area of a region by summing the incremental area formed by the triangles formed by the origin (0, 0) and each successive pair of points (r_k, c_k) and (r_{k+1}, c_{k+1}) on the perimeter. See the related figure. Incremental area is negative when the torque is negative and positive when the torgue is positive.

```
// look in course software collection for a similar C++ program
real computeArea ( perimeter set P )
{
    area := 0.0;
    for k from 0 to the size of P
        {
          area := area + P[k].r * P[k+1].c - P[k+1].r * P[k].c ;
        }
    // don't know if perimeter was clockwise or counterclockwise
    if  (area < 0.0)  then  area := -area ;
    return (0.5 * area); // 0.5 factored out of the element formula above
}
```

Exercise 12 Using properties

Suppose you have a collection of two-dimensional shapes. Some of them are triangles, some are rectangles, some are octagons, some are circles, and some are ellipses or ovals. Devise a recognition strategy for these shapes. You may use the operations of mathematical morphology and/or the properties defined so far.

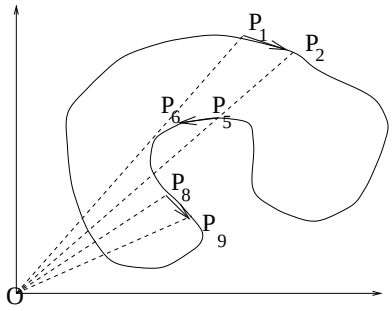


Figure 3.1: Computing the area of a region from only its perimeter set. The triangular element OP_1P_2 has positive area, while the triangular element OP_5P_6. Signs on these area elements depend upon the convention used and can be implemented via vector cross products. Despite the apparent smoothness of a perimeter, a digital perimeter is always a polygon as are all the triangular area elements, so the summation above gives an exact result within precision constraints.

Exercise 13 Program to compute point set features

Write a program module, or C++ class, that manages a *bag* of 2D points and provides the following functionality. A *bag* is different from a *set* in that duplicate points are allowed.

- construct an initially empty bag of 2D points (r,c)
- add point (r,c) to the bag
- compute the centroid of the current bag of points
- compute the row and column moments of the current bag of points
- compute the bounding box
- compute the best and worst axes and the second moments about them

Exercise 14 Program to compute features from images

After creating the feature extraction module of the previous exercise, enhance it to compute the second moments about horizontal, vertical and diagonal axes through the centroid of points. Thus, five different second moments will be available for any bag of points. Create a set of 20x20 binary images of digits from '0' to '9' for test data, or access some existing data. Write a program that scans an image of a digit and computes the five moments. Study whether or not the five moments have potential for recognizing the input digit.

Exercise 15 Compute the extremes of inertia

Differentiate the formula in Equation 3.22 and show how the best (and worst) axes are obtained in Equation 3.23.

Exercise 16 Verify that the best axis passes through the centroid

Verify that the axis of least inertia must pass though the centroid. Consult the references at the chapter's end or other references on statistical regression or mechanics; or, prove it yourself.

Exercise 17 Efficient RAG construction

Design a data structure for keeping track of adjacencies while constructing a region adjacency graph. Give algorithms that construct the graph from an arbitrary labeled image and that attempt to minimize references to the data structure. Discuss how you would store the final RAG in permanent storage (on disk) and how you would handle the case where the RAG is too large to keep in internal memory during its construction.

Answer:
If there are n possible image labels, then the RAG could have up to $n(n-1)/2$ edges. So, with 1000 labels, there could be one half million adjacencies to represent. It is more likely that there will be a limit of 1000c adjacencies, for a fairly small c.

One algorithm for recording the adjacencies is just to write them to a file: write (k,m) whenever region k is found to be adjacent to region m. The file is created by raster scanning the entire labeled image and writing any adjacency discovered in any neighborhood. After the file is written, it can be sorted and all duplicates can be discarded. One variation on this method would be to use a cache of all the pairs (k, m) for an entire row, and sorting and culling duplicates for each row before writing the results to the file.

For most images, it should be possible to handle everything in main memory via *adjacency lists*. n lists are maintained, one for each image label. Linked lists can all share a single pool of cells to keep memory use to a minimum. Linear search of a list to check if a relationship (k, m) is already recorded would be fast so long as the list is short. Yet another alternative is to store all pairs (k, m) in a single set, let's say implemented by a balanced binary search tree. (One million pairs can be accessed via a path no longer than 20.) To do this simply, we can combine both labels to get a single unique key = 1000k + m. (This is easy to obtain using the C++ STL map class.) Of course, it probably would still pay to use a local caching method so that we minimize repeated attempts to record the pair (k, m).

If a huge RAG must reside on disk even when accessed by algorithms, it would be good to store n adjacency lists for random access so that for any k all adjacent labels m could be accessed via a single disk read.

Exercise 18 Automatic threshold determination

Write a program to implement the Otsu automatic threshold finder. Try the program on several different types of scanned images.

Chapter 5

Image Enhancement and Filtering

Several of the answers in this chapter are thanks to Greg Bloy.

Exercise 1

An input image of 200 pixels has the following histogram: $Hin = [0, 0, 20, 30, 5, 5, 40, 40, 30, 20, 10, 0, 0, 0, 0, 0]$. (a) Using the formula for equalizing the histogram (of 15 grey levels), what would be the output image value $f(8)$? (b) Repeat question (a) for $f(11)$. (c) Give the lookup table $T[]$ implementing f for transforming all input image values.

Answer:

There are 200 pixels and 15 gray levels, or 13.3 pixels per level. For concretness, let's assume input gray levels 1,2,3, ,14,15. The ideal cumulative sums are 13.3, 26.6, 40, 53.3, , 186.6, 200, whereas the cumulative sums from the actual input image are 0, 0, 20, 50, 55, 60, 100, 140, 170, 190, 200, 200, 200, 200, 200. To answer (c), the lookup table (of Zi) would be as follows: 1, 1, 2, 4, 5, 5, 8, 11, 13, 15, 15, X, X, X, X. This means that f(1) = f(2) = 1; f(3) = 2; f(8) = 11; f(11) = 15, which answers (a) and (b). The function value f(g) is not defined for $g > 11$ and need not be, since values $g > 11$ do not appear in the input image. A better mapping may be to have f(10)=14 and f(11)=15.

The computed thresholds are 3,4,4,5,7,7,7,8,8,8,9,9,10,10,15

Exercise 2 An algorithm for histogram equalization

Use pseudocode to define an algorithm for histogram equalization. Be sure to define all the data items and data structures being used.

Answer:

We have the following data objects:

1. input image inImage[R x C] with L gray levels {1,2, , L }
2. output image outImage[R x C] with L gray levels
3. histogram of input image H[L]
4. array of thresholds T[L]
5. look up table F[L]

```
procedure histogram_Equalize ( inImage, outImage}, R, C, L)
{
      compute_Lookup_Table ( inImage, R*C/L  L,  F );
      for r:= 0 to R-1;
        for c:= 0 to C-1;
          {
             g := inImage[r,c];
             outImage[r,c]:= F[g];
          }
}
procedure compute_Lookup_Table ( inI, average, L, F )
{
      t := 1;   lowSum := 0;   middle := 0;  j := 0;
      while ( t < L )
        {
           j:=j+1;
           middle:=middle+average;
           while not ( lowSum <= middle < sum+H[t] )
              {
                  sum:=sum+H[t];
                  t:=t+1;
              }
           T[j]:=t;
        }
        t := 1;
        for j:= 1 to L
          {
             while (t<T[j])
                {
                    F[t] := j;
                    T:=t+1;
                }
          }
        return;
}
```

Exercise 3 A histogram equalization program

(a) Using the pseudocode from the previous exercise, implement and test a program for histogram equalization. (b) Describe how well it works on different images.

Answer:

Some advice for testing the program : create a test image with the statistics used in the previous problem. When the program works correctly for that image, try others, such as the Alaskan Pipeline image.

Exercise 4

Give some arguments for and against "randomizing" function f in order to make the output histogram more uniform.

Answer:

Looking back at the histogram in the first equalization problem, we see that output level 3 is not used but 30 pixels have level 4. For some images, it may be better to split the 30 pixels into sets of 15 pixels at level 3 and 15 pixels at level 4. This may improve the appearance of the output by reducing the blockiness or edgeness of the output. On the other hand, it may appear that a region split in this manner is defective or noisy. Making such random decisions adds information to the image, which may not actually be there.

Exercise 5 Modifying Quicksort 1

(a) Find pseudocode for the traditional quicksort algorithm from one of the many data structures and algorithms texts. Modify the algorithm to return only the median as soon as it can be decided. (b) What is the computational effort of finding the median relative to performing the entire sort? (c) Implement your algorithm in some programming language and test it on some example images.

Answer:

Quicksort is a clever algorithm that is difficult to analyze. However, converting the full sorting algorithm to a median algorithm is easy. The key observation is that whenever Quicksort places a pivot element, then that element never changes position in further sorting operations. Therefore, whenever Quicksort places a pivot element in position m = (n-1)/2 of the array, then this element must be the median! (See the text comment about whether there is an even or an odd number of elements.) Moreover, the algorithm does not need to recurse on any interval that does not contain the m-th position. These changes create an algorithm that is of order n for the average case rather than of order n log n. Consult research journals or advanced algorithms texts for a proof that this is so.

Exercise 6 Modifying Quicksort 2

Consider using the quicksort algorithm from above to detect steps in the picture function, such as the steps from black to white squares in the checkerboard images. Suppose the median of a neighborhood about **I**[r,c] has just been found by placing a pivot value in array position **A**[(n-1)/2]. Describe how the rest of the array can be processed to decide whether or not the pixel at [r,c] is or is not on the boundary between regions of different brightness.

Answer:

Let g_m be the gray level placed in **A**[(n-1)/2] by Quicksort. All intensity values in lower array positions are $\leq g_m$ and all intensity values in higher array positions are $\geq g_m$. If there is a step in the image, then there should be a cluster of low values in the low end of the array and a cluster of high values in higher positions of the array. There may be a small cluster of intensities in the middle that are similar to g_m and between the low and high values: this would be the case if there are mixed pixels resulting from sampling along a step edge. Histogramming the low values separately from the high values will show this. We need to check one more condition: for the "median pixel" to be on the boundary, there should be approximately equal numbers of intensities in the clusters on either side of it.

Exercise 7

Implement Algorithm 5.1 in some programming language. Code both the boxcar and median filtering operations and test them on some images such as in Figure 5.9.

Exercise 8

If a true gradient magnitude is needed, why might the Sobel masks provide a faster solution than the Prewitt masks?

Answer:

Using the Sobel, there is a division by 8, not the 6 used in the Prewitt operator, which may be performed very quickly by shifting instead of division.

Exercise 9 *Optimality of Prewitt masks

The problem is to prove that Prewitt masks give the weights for the best fitting plane approximating the intensity surface in a 3×3 neighborhood, assuming all 9 samples have equal weight. Suppose that the 9 intensities $I[r+i, c+j]; i, j = -1, 0, 1$ of a 3×3 image neighborhood are fit by the least squares planar model $I[r, c] = z = pr + qc + z_0$. (Recall that the 9 samples are equally spaced in terms of r and c.) Show that the Prewitt masks compute the estimates of p and q as the partial derivatives of the least squares planar fit of the intensity function.

Answer:

Students unfamiliar with least squares fitting should read forward in Section 10.4 and Section 11.4, or in other outside literature, before tackling this problem. When the formula for z is derived, the student should see that $\partial z/\partial r = p$ and $\partial z/\partial c = q$.

Exercise 10 Properties of the LOG filter

Suppose that the 9 intensities of a 3×3 image neighborhood are perfectly fit by the planar model $I[r, c] = z = pr + qc + z_0$. (Recall that the 9 samples are equally spaced in terms of r and c.) Show that the simple LOG mask

0	-1	0
-1	4	-1
0	-1	0

has zero response on such a neighborhood. This means that the LOG filter has zero response on both constant regions and ramps.

Exercise 11

Give more detailed support to the above arguments that the integrating cells shown in Figure 5.25 (a) respond to contrasting spots or blobs that image within the center of the field and (b) respond to boundaries between two large regions that just barely cross into the center of the field.

Answer:

(a) Suppose the center blob has intensity h and it is surrounded by lower intensities l. h will be multiplied by the positive weights in the overall sum, whereas l will be multipled by negative weights. Recall that the sum of all the positive weights is equal to the negative of the sum of the negative weights. Because l is smaller than h, the positive components will exceed the negative components and there will be a positive response. If there is a dark spot on a brighter background, then the roles of h and l are reversed and the ANN will produce a strong negative response. (b) Refer to the figure below, which shows a straight edge crossing a receptive field. Intensities at the lower right of the edge are higher than those at the upper left. Now, the positive central weights balance with the negative surround weights to produce a 0 response with a uniform intensity field. This nonuniform intensity field has higher intensities over the lower right surround, which will not be balanced by the remaining integrated response. Thus, the overall response will be negative. If the positions of the intensities l and h are exchanged, then the overall response of the receptive field would be positive. Since the receptive field and all of its weights are isotropic, the edge can pass through in any direction with the same result. (If several receptive fields along a straight line are integrated at another cell at a higher level, this cell would detect a straight edge from the response to several edgels along that line.)

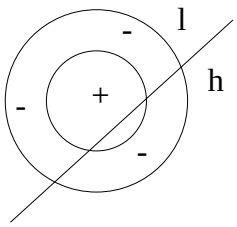


Figure 5.1: Sketch of receptive field as center with positive gains and surround of negative gains.

Exercise 12

Choose any 5 of the 9 listed properties of vector spaces and show that each is true.
Answer:

1 $U \oplus V = (u_1 + v_1, u_2 + v_2, \ldots, u_n + v_n) = (v_1 + u_1, v_2 + u_2, \ldots, v_n + u_n) = V \oplus U$ since the coordinates are all real numbers and hence commutative.

2 $U \oplus (V \oplus W) = (u_1, u_2, \ldots, u_n) \oplus (v_1 + w_1, v_2 + w_2, \ldots, v_n + w_n) = (u_1 + (v_1 + w_1), u_2 + (v_2 + w_2), \ldots, u_n + (v_n + w_n)) = ((u_1 + v_1) + w_1, (u_2 + v_2) + w_2, \ldots, (u_n + v_n) + w_n) = (U \oplus V) \oplus W$. Thus, associativity of vectors with real coordinates follows easily from the associativity of real numbers.

3 The zero vector is the vector $O = (0, 0, \ldots, 0)$ so that for any vector $V = (v_1, v_2, \ldots, v_n)$, $O \oplus V = (0 + v_1, 0 + v_2, \ldots, 0 + v_n) = (v_1, v_2, \ldots, v_n) = V$.

4 The negative vector $(-1)V = (-1v_1, -1v_2, \ldots, -1v_n)$ with the property that $V \oplus (-1)V = (v_1 + (-1v_1), v_2 + (-1v_2), \ldots, v_n + (-1v_n)) = O$.

9 For any vector $V, (-1V) \circ V = (-1v_1, -1V_2, \ldots, -1v_n) \circ (v_1, v_2, \ldots, v_n)$
$= (-1(v_1 v_1), -1(v_2 v_2), \ldots, -1(v_n v_n)) = (-1)(V \circ V) = (-1) \| V \|^2$.

Exercise 13

(a) Following the boxed example above, represent the vector $[10, 14, 15]$ in terms of the basis $\{\frac{1}{\sqrt{2}}[-1, 0, 1],\ \frac{1}{\sqrt{3}}[1, 1, 1],\ \frac{1}{\sqrt{6}}[-1, 2, -1]\}$. (b) Now represent $[10, 19, 10]$: to which basis vector is it most similar? Why?

Answer:

(a) Since the basis is an orthonormal basis, we project the vector $[10, 14, 15]$ onto each basis vector to obtain its components. The magnitudes of the projections onto $\mathbf{W_1}, \mathbf{W_2}, \mathbf{W_3}$ are, respectively, $\frac{5}{\sqrt{2}}$, $\frac{39}{\sqrt{3}}$ and $\frac{3}{\sqrt{6}}$. This gives the expansion $\mathbf{V1} = [10, 14, 15] = \frac{5}{\sqrt{2}}\mathbf{W_1} \oplus \frac{39}{\sqrt{3}}\mathbf{W_2} \oplus \frac{3}{\sqrt{6}}\mathbf{W_3}$. Thus, $\mathbf{V_1}$ is more like a ramp ($\mathbf{W_1}$) after the constant component $\mathbf{W_2}$ is removed.

(b) Doing the three projections, we obtain $\mathbf{V_2} = 0\mathbf{W_1} \oplus \frac{39}{\sqrt{3}}\mathbf{W_2} \oplus \frac{18}{\sqrt{6}}\mathbf{W_3}$. Thus, $\mathbf{V_2}$ is more like a cap or tent ($\mathbf{V_3}$) than a ramp after the constant component is removed.

Exercise 14

Verify that the Roberts basis vectors shown in Figure 5.30 are orthonormal.

Exercise 15

Sketch the following five vectors and compute the normalized dot product, or *cos* of the angle between each pair of the vectors: $[5, 5]$, $[10, 10]$, $[-5, 5]$, $[-5, -5]$, $[-10, 10]$. Which pairs are perpendicular? Which pairs have the same direction? Which have opposite directions? Compare the relative directions to value of the normalized dot product.

Answer:

The vectors are sketched in the figure below. The magnitudes of the vectors with coordinates of 10 are $10\sqrt{2}$, while the magnitudes of the vectors with coordinates of 5 are $5\sqrt{2}$. The normalized dot products are given in the table below. Perpendicular pairs are indicated by the 0 dot products. Pairs with the same direction are indicated by a 1 in the table and pairs with opposite directions are indicated by a -1.

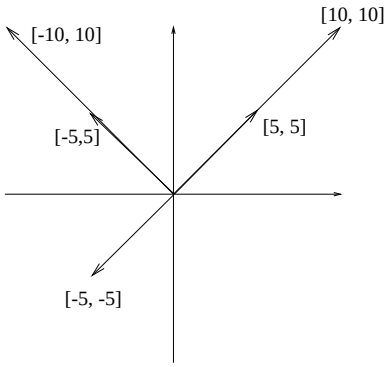
$\circ$	$[5, 5]$	$[10, 10]$	$[-5, 5]$	$[-5, -5]$	$[-10, 10]$
$[5, 5]$	1	1	0	-1	0
$[10, 10]$	1	1	0	-1	0
$[-5, 5]$	0	0	1	0	1
$[-5, -5]$	-1	-1	0	1	0
$[-10, 10]$	0	0	1	0	1

Exercise 16

Consider the vector space of all 2x2 images with real-valued pixels. (a) Determine the values of the a_j so that the image $\begin{array}{|c|c|}\hline 10 & 5 \\ \hline 5 & 0 \\ \hline \end{array}$ is represented as a linear combination of the four Roberts basis images W_j. (b) Explain why we can always find unique a_j for any such 2x2 image.

Exercise 17

Suppose that the 2x2 image $\begin{array}{|c|c|}\hline a & b \\ \hline c & d \\ \hline \end{array}$ has energy e_1, e_2, e_3, e_4 along the respective four Roberts basis vectors W_1, W_2, W_3, W_4 respectively. What are the formulas for computing the four e_1 in terms of a, b, c, d?



Exercise 18

Verify that the set of nine vectors shown in Figure 5.32 is an orthonormal set.

Answer:

It should be instructive to see a C++ program that implements the checks that $W_j \circ W_k = 1$ only when $j = k$. 81 checks need to be made to be thorough; only a few are given in the figures below.

Exercise 19

(a) Represent the intensity neighborhood $\mathbf{N_1} = \begin{array}{|c|c|c|}\hline 0&0&1\\\hline 0&1&0\\\hline 1&0&0\\\hline\end{array}$ in terms of the basis vectors shown in Figure 5.32. Is all the energy distributed along the *line* basis vectors $\mathbf{W_5}$ and $\mathbf{W_6}$?

(b) Repeat the question (a) for the intensity neighborhood $\mathbf{N_2} = \begin{array}{|c|c|c|}\hline 0&1&0\\\hline 0&1&0\\\hline 0&1&0\\\hline\end{array}$.

Answer:

(a) $\mathbf{N_1} = \mathbf{1W_6} \oplus \mathbf{1W_7} \oplus \mathbf{1W_9}$, so the answer is not a clear yes. It is not surprising that $\mathbf{W_9}$ is needed, but $\mathbf{W_7}$ is also needed to represent all of the energy of $\mathbf{N_1}$.

(b) $\mathbf{N_2} = \mathbf{1W_5} \oplus \mathbf{1W_8} \oplus \mathbf{1W_9}$. The interpretation is the same as in (a).

```
/*  C++ code by GCS toward solution of 5.18-5.20 in the S&S CV text  */
/*           Even students who don't know C++ can benefit.           */

#include <iostream.h>
#include <math.h>

/* return the dot product of vectors V and U, each of dimension N */

double dotProduct ( const double U[], const double V[], const int N )
{
  double sum = 0.0;
  for (int k=0; k<N; k++) sum = sum + U[k]*V[k];
  return sum;
}

/* scale vector V by multiplying each coordinate by scaleFactor */

void scaleVector ( double V[], const int N, const double scaleFactor)
{
  for (int k=0; k<N; k++)  V[k] = V[k]*scaleFactor;
}
```

Figure 5.2: Functions to handle the dot product and scaling of 3x3 matrices represented as a 1D vector of 9 coordinates. These functions are used in answering other exercises as well.

```
/* main program to define basis vectors and to test dot products */

int main (void)
{
  double W1[9] = { 1, sqrt(2.0), 1,  0, 0, 0,  -1, -sqrt(2.0), -1 };
  scaleVector(W1,9, 1/(sqrt(8.0)));
  cout << "W1 o W1 = " << dotProduct(W1,W1,9) << endl;
  double W2[9] = { 1, 0, -1,  sqrt(2.0), 0, -sqrt(2.0), 1, 0, -1};
  scaleVector(W2,9, 1/(sqrt(8.0)));
  cout << "W2 o W2 = " << dotProduct(W2,W2,9) << endl;
  cout << "W1 o W2 = " << dotProduct(W1,W2,9) << endl;
  double W9[9] = {1, 1, 1,  1, 1, 1,  1, 1, 1};
  scaleVector(W9,9,1.0/3.0);
  cout << "W1 o W9 = " << dotProduct(W1,W9,9) << endl;
  cout << "W2 o W9 = " << dotProduct(W2,W9,9) << endl;
  cout << "W9 o W9 = " << dotProduct(W9,W9,9) << endl;
  double W6[9] = { -1,0,1,  0,0,0,  1,0,-1 };
  scaleVector(W6,9,0.5);
  cout << "W6 o W6 = " << dotProduct(W6,W6,9) << endl;
  cout << "W6 o W1 = " << dotProduct(W6,W1,9) << endl;
  cout << "There are a total of 9 x 9 = 81 dot products to check!" << endl;
  return 1;
}

<139 ocarina:>g++ dotProduct.cpp
<140 ocarina:>a.out
W1 o W1 = 1
W2 o W2 = 1
W1 o W2 = 0
W1 o W9 = 1.38778e-17
W2 o W9 = 0
W9 o W9 = 1
W6 o W6 = 1
W6 o W1 = 0
There are a total of 9 x 9 = 81 dot products to check!
```

Figure 5.3: Main program and output; shows some dot products of Frei-Chen basis vectors.

Exercise 20

(a) Represent the intensity neighborhood $\begin{array}{|c|c|c|}\hline 10 & 10 & 10 \\ \hline 10 & 20 & 10 \\ \hline 10 & 10 & 10 \\ \hline \end{array}$ in terms of the basis vectors shown in Figure 5.32. Is all the energy distributed along certain basis vectors as you expect? (b) Would the interpretation of the intensity neighborhood $\begin{array}{|c|c|c|}\hline 0 & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & 0 & 0 \\ \hline \end{array}$ be different: why or why not? (c) What kind of image neighborhoods give responses to only W_7 and W_8?

Answer:

(a) By using the following main program with the C++ functions already supplied in the answer to Exercise 5.18, the expansion of this neighborhood in terms of $\mathbf{W_1} \ldots \mathbf{W_9}$ is achieved. Let $\mathbf{M}$ denote the intensity neighborhood; then we have $\mathbf{M} = 6.67\mathbf{W_7} \oplus 6.67\mathbf{W_8} \oplus 33.3\mathbf{W_9}$. Since this is a brighter (one pixel) blob on a lower background, we should expect the energy to be in $\mathbf{W_7}$ and $\mathbf{W_8}$ after the energy along $\mathbf{W_9}$ is taken out.

(b) The interpretation of the intensity neighborhood $\mathbf{M} = \begin{array}{|c|c|c|}\hline 0 & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & 0 & 0 \\ \hline \end{array}$ is much the same. $\mathbf{M} = 0.67\mathbf{W_7} \oplus 0.67\mathbf{W_8} \oplus 0.33\mathbf{W_9}$, which can be obtained by computing the projection of this neighborhood onto each of the 9 Frei-Chen basis vectors. Note that the magnitudes are one tenth of what they were for the other neighbohood, but in the same ratio.

(c) Any 3x3 neighborhood with a nonzero center pixel surrounded by 8 neighbors of zero intensity would result in energy only along basis vectors $\mathbf{W_7}$, $\mathbf{W_8}$, and $\mathbf{W_9}$.

Exercise 21

Write a program to implement the detection of pixels using the Frei-Chen basis as outlined in the algorithm above. Allow the user of the program to input the subspace **S** of interest as a string of 9 bits. The user should also be able to input the noise energy level and the threshold determining the minimum energy required in the selected subspace. Test your program on some real images and also on some test patterns such as those in the exercises above.

Answer:

A good part of the functionality of the program is contained in the functions already provided for the answers to 5.18 to 5.20. Each neighborhood N(r,c) is examined to see if its energy in the given subspace is above threshold.

```
procedure detect_Frei_Chen
{
    input an image I[][], the 9 bits B[] specifying the subspace,
         and the energy threshold t;
    create a separate output image Out[][];
    set the border of Out[][] to all 0's;

    for every pixel I[r][c] interior to image I
       {
           for each basis vector j that is ON in bitmap B
             {
                total_energy = total_energy + ( N(r,c) o Wj )^2;
             }
           if  ( total_energy > t )  Out[r][c] = 1; (or = total_energy)
           else  Out[r][c] = 0;
       }
    write out image Out[][] to show pixels in detected neighborhoods;
}
```

Exercise 22

Suppose that an image **F**[] has all 0 pixels except for a single 1 pixel at the image center. What output image **G**[] results from convolving **F**[] with the 3×3 boxcar shown in Figure 5.33?

Answer:

All output pixels will be 0, except those in the 3x3 neighborhood of the pixel with value 1. For each one of these pixels, exactly one of the mask pixels, of value $\frac{1}{9}$, is overlaid with the 1 of image function **F**. Thus, the output image function **G** is an image of all zeros and a 3x3 neighborhood of pixels with intensity $\frac{1}{9}$ centered where $\mathbf{F}[r, c] = 1$.

Exercise 23

Design a single mask to detect edge elements making an angle of 30^0 with the X-axis. The mask should not respond strongly to edge elements of other directions or to other patterns.

Exercise 24 Corner detection

(a) Design a set of four 5×5 masks to detect the corners of any rectangle aligned with the image axes. The rectangle can be brighter or darker than the background. (b) Are your masks orthogonal? (c) Specify a decision procedure to detect a corner and justify why it would work.

Exercise 25 Rectangle detection

Use the corner detection procedure of the previous exercise to implement a program that detects rectangles in an image. (Rectangle sides are assumed to align with the sides of the image.) The first step should detect candidate rectangle corners. A second step should extract subsets of four candidate corners that form a proper rectangle according to geometric constraints. An optional third step might further check the four corners to make sure that the intensity within the candidate rectangle is uniform and contrasting with the backgorund. What is the expected behavior of your program if it is given a noisy checkerboard image? Test your program on a noisy checkerboard, such as in Figure 5.7 and an image of a building with rectangular windows, such as in Figure 5.16.

Exercise 26

Given $\mathbf{H} =$

1	0	-3
0	4	0
-3	0	1

and $\mathbf{F} =$

5	5	5	5	5	5
5	5	5	5	8	8
5	5	5	8	8	8
5	5	8	8	8	8
8	8	8	8	8	8

,

compute $\mathbf{G} = \mathbf{F} \otimes \mathbf{H}$.

Exercise 27 Point spread

Given kernel $H =$ $\begin{array}{|c|c|c|}\hline 1 & 2 & 1 \\ \hline 2 & 5 & 2 \\ \hline 1 & 2 & 1 \\ \hline \end{array}$, what is the result of convolving it with an image $F[x, y]$ which has $F[x_0, y_0] = 1$ and all other pixels 0?

Exercise 28

Suppose that function $h(x)$ takes value 1 for $-1/2 \leq x \leq 1/2$ and is zero elsewhere, and suppose that function $f(x)$ takes value 1 for $10 \leq x \leq 20$ and zero elsewhere. (a) Sketch the two functions f and h. (b) Compute and sketch the function $g(x) = f(x) \star h(x)$. (c) Compute and sketch the function $g(x) = h(x) \star h(x)$.

Exercise 29

The *pointilist* school of painters created their paintings by dabbing their paintbrush perpendicularly on the canvas and creating one dab/point of color at a time. Each dab is similar to a single pixel in a digital image. The human viewer of the artwork would stand back and a smooth picture would be perceived. Implement a program to do pointilist art. The program should present a palette of colors and some other options, such as choosing paintbrush size or whether '+' or 'XOR' would be used for dabbing, etc. When your program is working, create a painting of a starry night. Your program should work with an external file representing the painting in progress so that a user could save the session for later continuation.

Exercise 30

Implement a program to convolve a mask with an image. The program should read both the image and mask from input files in the same format. Perhaps you can tes t it on the artwork produced by the program from the previous exercise.

Exercise 31

Suppose in the search for extra terrestrial intelligence (SETI) deep space is scanned with the hope that interesting signals would be found. Suppose signal **S** is a concatenation of the first 100 prime numbers in binary form and that **R** is some received signal that is much longer than **S**. Assume that **R** has noise and has real valued coordinates. To find if **S** is embedded in **R**, would cross correlation or normalized cross correlation work? Why?

Answer:

Cross correlation is unlikely to work well because the match value will rise in proportion to the magnitude of the received signal **R**, and this may have little to do with **S**. For example, if the received signal has only 5 very high values that align with 1-bits of **S**, the resulting correlation might be higher than a low power signal with many bits aligned with **S**. Normalization will help as long as the energy of **R** is well above the noise level. But, if **R** is near the noise level, it may match with **S** over time, just due to chance. It would be better if the energy of **R** is first tested, if insufficiently above the noise level, no match will be declared, otherwise, a match will be declared if the normalized cross correlation is significant.

Exercise 32

Consider the set of all continuous functions f defined on the interval $x \in [x_1, x_2]$. Show that this set of functions $f, g, h, \ldots$, together wit h scalars $a, b, c, \ldots$ forms a vector space by arguing that the following properties hold.

$$\begin{aligned} f \oplus g = g \oplus f \qquad & (f \oplus g) \oplus h = f \oplus (g \oplus h) \\ c(f \oplus g) = cf \oplus cg \qquad & (a+b)f = af \oplus bf \\ (ab)f = a(bf) \qquad & 1f = f \\ 0f = \mathbf{0} & \end{aligned}$$

Exercise 33

For the space of continuous functions of $x \in [x_1, x_2]$, as in the exercise a bove, define a dot product and corresponding norm as follows.

$$f \circ g \;=\; \int_a^b f(x)g(x)dx \;\;;\;\; \|f\| = \sqrt{f \circ f} \tag{5.1}$$

Argue why the following four properties of a dot product hold.

$$\begin{aligned} (f \oplus g) \circ h \quad &= (f \circ g) + (g \circ h) \\ f \circ f \quad &\geq 0 \\ f \circ f = 0 \Longleftrightarrow f = \mathbf{0} & \\ f \circ g \quad &= g \circ f \\ (cf) \circ g \quad &= c(f \circ g) \end{aligned}$$

Exercise 34 Odd and even functions

A function is an *even function* iff $f(-x) = f(x)$ and a function is an *odd function* if $f(-x) = -f(x)$. (a) Show that $cos(mx)$ is an even function and that $sin(nx)$ is an odd function, where m, n are nonzero integers. (b) Let f and g be odd and even continuous functions on $[-L, L]$, respectively. Argue that $\int_{-L}^{L} f(x)g(x)dx = 0$.

Answer:

(a) We know that by its definition $cos(-x) = cos(x)$ for all real x, so the *cos* function is even. We also know that for all x, $sin(-x) = -sin(x)$, so the *sin* is an odd function. (b) Consider the integral as the sum of two parts, one for the interval $[0, L]$ and one for the interval $[-L, 0]$. These parts will cancel each other. Consider what happens when $x = a$, some arbitrary positive constant. $f(a)g(a)\Delta x$ is used in one part of a Riemann sum in the definition of the right integral, whereas $f(-a)g(-a)\Delta x$ is used in a part in the left integral. But, $f(-a)g(-a) = -f(a)g(a)$ since f is odd and g is even, proving our point.

Exercise 35

Using the definition of the dot product given in Exercise 33 above, show that the set of sinusoidal functions f_k defined below are orthogonal on the interval $[-\pi, \pi]$.
$f_0(x) = 1$; $f_1(x) = sin(x)$; $f_2(x) = cos(x)$; $f_3(x) = sin(2x)$; $f_4(x) = cos(2x)$; $f_5(x) = sin(3x)$; $f_6(x) = cos(3x); \ldots$

Exercise 36

What is the special meaning of $F(0,0)$, where $F(u,v)$ is the Fourier transform of picture function $f(x,y)$?

Exercise 37 Some basics about complex numbers

Use the definition $e^{j\omega} = cos\ \omega + j\ sin\ \omega$. (a) Show that $(e^{j\omega})^n = cos(n\omega) + j\ sin(n\omega)$. (b) Show that $x = e^{j\frac{2\pi k}{N}}$ is a solution to the equation $x^N - 1 = 0$ for $k = 0, 1, \ldots, N-1$. (c) Let $x_0 = 1 = e^{j\frac{2\pi 0}{N}}, \ldots, x_k = e^{j\frac{2\pi k}{N}}$ be the N roots of the equation $x^N - 1 = 0$. Show that $x_1 + x_2 + x_3 + \ldots + x_{N-1} = 0$.

Exercise 38 Proof of invertability of DFT/IDFT transforms

We want to prove that substituting the $F[u,v]$ from Equation **??** into Equation **??** returns the exact original value $I[x,y]$. Consider the following summation, where x, y, s, t are integer parameters in the range $[0, N-1]$ as used in the transform definitions:

$$G(x,y,s,t) = \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} e^{\frac{j\ 2\pi}{N}((x-s)u\ +\ (y-t)v)}.$$

(a) Show that if $s = x$ and $t = y$, then $G(x,y,s,t) = \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} 1 = N^2$. (b) Show that if $s \neq x$ or $t \neq y$ then $G(x,y,s,t) = 0$.
(c) Now show the major result that the inverse transform applied to the transfor m returns the original image.

Exercise 39

Follow the steps given above for proving the Shift Theorem and Convolution Theor em in the 1D case and extend them to the 2D case.

Chapter 7

Texture

Exercise 1 Texel-Based Descriptions

Find or create a set of 5 images showing textures that have obvious texels that can be detected via a simple procedure such as thresholding based on gray-tone or color ranges. Try to find at least one texture that has more than one kind of texel. Draw the Voronoi tesselation for a small area on this image.

Answer:
This is an exploration problem with no specific answer.

Exercise 2 Edge-Based Texture Measures

Obtain a set of images that have lots of man-made structures with clearly defined edges. Write a program to compute the texture measure F_{magdir} of equation 7.2 for each of these images, and compare them using the L_1 distance of equation 7.3.

Answer:
This is a programming problem with no specific answer.

Exercise 3 LPB

Using the images from the previous exercise, write another program to compute the histogram representing the LBP texture measure of each image. Again compute the L_1 distances between pairs of images using this measure. Compare to your previous results.

Answer:
This is a programming problem with no specific answer.

Exercise 4 Co-occurence Matrices

Construct the gray-tone co-occurrence matrices $C_{(1,2)}$, $C_{(2,2)}$, and $C_{(2,3)}$ for the image of Figure 7.6 (shown below).

Answer:

1	1	0	0
1	1	0	0
0	0	2	2
0	0	2	2

Original Image

i		
		j

0	0	2
2	0	2
0	0	0

$C_{[1,2]}$

i		
		j

0	0	0
0	0	4
0	0	0

$C_{[2,2]}$

i			
			j

0	0	0
0	0	2
0	0	0

$C_{[2,3]}$

Exercise 5 Normalized Co-occurrence

Compute the normalized co-occurrence matrix $N_{(1,1)}$ for the image of Figure 7.7 assuming that the black pixels have gray tone 0, the gray pixels have gray tone 1, and the white pixels have gray tone 2. How does it represent the texture pattern of the image?

Answer:

20	0	0
0	21	0
0	0	76

.17	0	0
0	.18	0
0	0	.65

Exercise 6 Laws Texture Energy Measures

Write a program to compute the Laws texture energy measures that inputs a gray-scale image and outputs a set of nine images, one for each of the texture energy measures. Obtain a set of images of both man-made and natural textures and perform a sequence of tests on them. For each test, make one of the images the *test image* and call the others the *database images*. Write an interactive front end that allows a user to select a pixel of the test image and then looks for those database images that have a texture similar to that of the selected pixel anywhere in that database image using the L_1 distance on the set of nine texture energy measures available for each pixel. The brute force way to do this is to merely compare the nine values for the test image pixel with the nine values for each pixel of each database image and select an image as soon as any of its pixels has similar enough texture energy measure values. How might you do this more efficiently?

Answer:

This is a programming exercise. I do have C code for it, as shown below:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>

int width,height;
unsigned char *image;    // original image
short *image2;           /* preprocessed image - orig image with local average
    subtracted out */

int get_image_value(int i,int j)
{
   int sum;
   if (i<0)
      i = 0;
   if (j<0)
      j = 0;
   if (i>height-1)
      i = height-1;
   if (j>width-1)
      j = width-1;
   sum = image[i*width*3+j*3]+image[i*width*3+j*3+1] + image[i*width*3+j*3+2];
   return(sum/3);
}

// subtract the local average from each point in image
// to account for variations in illumination
void subtract_average()
{
```

```
   int sum,i,j;

   if (image2) free(image2);
   image2= (short*) malloc(sizeof(short) * width * height);
   for (i=0;i<height;i++)
      for (j=0;j<width;j++) {
 sum = 0;
 for (int l=-7;l<=7;l++)
    for (int m=-7;m<=7;m++)
       sum += get_image_value(i+l,j+m);
 sum/=225;
 image2[i*width+j] = get_image_value(i,j) - sum;
      }
}

void read_ppm_file(char *fn)
{
    FILE *f;
    char str[80];
    int i,j,size;

    if (!(f = fopen(fn,"r"))) {
        printf("%s not found\n",fn);
        exit(1);
    }
    fgets(str,80,f);                        /* P6 - identifies it as ppm */
    fgets(str,80,f);                        /* might be width and height */
    while (str[0] == '#') fgets(str,80,f);  /* or just a comment */
    sscanf(str,"%d %d",&width,&height);
    fgets(str,80,f);                        /* max value per pixel */
    size = width * height * 3;
    if (image) free(image);
    image= (unsigned char*) malloc(size);   /* allocate memory for image */
    fread(image,size,1,f);                  /* read image */
    fclose(f);
}

// calculate Laws' texture energy measures at every point in an image
// K. Laws. Rapid texture identification, In SPIE Vol. 238: Image
// Processing for Missile Guidance, pp376-380, 1980
void extract_texture_Laws(short* output[])
{
   int L[4][5] = {{ 1, 4, 6, 4, 1},
  {-1,-2, 0, 2, 1},
  {-1, 0, 2, 0,-1},
  { 1,-4, 6,-4, 1}};

   short* temp[4];
```

```
  int* filtered[4][4];
  int* energy[4][4];
  int i,j,k,l,m;
  int colSum[1024];
  const int WIN_SIZE=15;
  const int HALF_WIN=WIN_SIZE/2;

      /* first pass - convolve horizontally */
  for (m=0;m<4;m++) {
     temp[m] = (short*) calloc(width*height,sizeof(short));
     for (i=0;i<height;i++) {
for (j=2;j<width-2;j++) {
   for (k=0;k<5;k++)
      temp[m][i*width+j] += L[m][k] * image2[i*width+j+k-2];
}
temp[m][i*width+0] = temp[m][i*width+1] = temp[m][i*width+2];
temp[m][(i+1)*width-1] = temp[m][(i+1)*width-2]
   = temp[m][(i+1)*width-3];
     }
  }
      /* second pass - convolve vertically */
  for (l=0;l<4;l++) {
     for (m=0;m<4;m++) {
filtered[l][m] = (int*) calloc(width*height,sizeof(int));
for (j=0;j<width;j++) {
   for (i=2;i<height-2;i++) {
      for (k=0;k<5;k++)
 filtered[l][m][i*width+j] +=
    L[l][k] * temp[m][(i+k-2)*width+j];
   }

   filtered[l][m][0*width+j] = filtered[l][m][1*width+j] =
      filtered[l][m][2*width+j];
   filtered[l][m][(height-1)*width+j] =
      filtered[l][m][(height-2)*width+j] =
      filtered[l][m][(height-3)*width+j];
}
     }
  }

  for (i=0;i<4;i++)
     free(temp[i]);

      /* calculate energy - averaging over window of size WIN_SIZE */
  for (l=0;l<4;l++) {
     for (m=0;m<4;m++) {

for (i=0;i<width;i++)
```

```
    colSum[i] = 0;

 energy[l][m] = (int*) calloc(width*height,sizeof(int));

     /* sum up absolute values in each column */
 for (i=0;i<width;i++) {
    for (j=0;j<WIN_SIZE;j++) {
       colSum[i] += abs(filtered[l][m][j*width+i]);
    }
 }

     /* sum up each row */
 for (i=HALF_WIN;i<height-HALF_WIN;i++) {
/* first sum */
    for (j=0;j<WIN_SIZE;j++)
       energy[l][m][i*width+HALF_WIN] += colSum[j];
/* remainder of the columns */
    for (j=HALF_WIN+1;j<width-HALF_WIN;j++)
       energy[l][m][i*width+j] =
  energy[l][m][i*width+j-1] -
  colSum[j-HALF_WIN-1] + colSum[j+HALF_WIN];

/* update column sums */
    for (j=0;j<width;j++) {
       colSum[j] =  colSum[j] -
  abs(filtered[l][m][(i-HALF_WIN)*width+j]) +
  abs(filtered[l][m][(i+HALF_WIN+1)*width+j]);
    }
 }
      }
   }

   for (i=0;i<4;i++)
      for (j=0;j<4;j++)
 free(filtered[i][j]);

   /* copy values to boundary pixels */
   for (i=HALF_WIN;i<height-HALF_WIN;i++) {
     for (j=0;j<HALF_WIN;j++)
       for (k=0;k<4;k++)
 for (m=0;m<4;m++)
   energy[k][m][i*width+j] = energy[k][m][i*width+HALF_WIN];
     for (j=width-HALF_WIN;j<width;j++)
       for (k=0;k<4;k++)
 for (m=0;m<4;m++)
   energy[k][m][i*width+j] = energy[k][m][(i+1)*width-HALF_WIN-1];
   }
   for (i=0;i<HALF_WIN;i++)
```

```
     for (j=0;j<width;j++)
       for (k=0;k<4;k++)
 for (m=0;m<4;m++)
   energy[k][m][i*width+j] = energy[k][m][HALF_WIN*width+j];
   for (i=height-HALF_WIN;i<height;i++)
     for (j=0;j<width;j++)
       for (k=0;k<4;k++)
 for (m=0;m<4;m++)
   energy[k][m][i*width+j] = energy[k][m][(height-HALF_WIN-1)*width+j];

   for (i=0;i<9;i++)
     output[i] = (short*) malloc(width*height*sizeof(short));

   int c = -1;
   for (m=0;m<4;m++) {
      for (l=m;l<4;l++,c++) {
 if (!(m==0 && l==0))
    for (i=0;i<height;i++) {
       for (j=0;j<width;j++) {
  output[c][i*width+j] =
     (energy[m][l][i*width+j] + energy[l][m][i*width+j])/
     (WIN_SIZE*WIN_SIZE*2);
       }
    }
      }
   }
   for (i=0;i<4;i++)
      for (j=0;j<4;j++)
 free(energy[i][j]);

}
```

Exercise 7 Texture Segmentation

Use your program that computes the Laws texture energy measures for an image to investigate how well they would perform for texture segmentation. Write another interactive front end that allows the user to draw boxes around sections of the image that contains a single category of texture such as flowers or grass or sky. For each box, compute the the average values for the nine texture features. Produce a table that lists each texture category by name and prints the nine averages for that category. Compare the results on several different categories.

Answer:

This is an exploratory exercise that has no specific answer.

Chapter 8

Content-based Image Retrieval

From: shapiro@cs.washington.edu Sent: Wednesday, August 06, 2003 2:20 PM To: stockman@cse.msu.edu Subject: DatabaseAnswers

---

Exercise 1 Keyword Queries

Give an SQL query that will retrieve the image of Figure 8.2(a), but will not retrieve the image of Figure 8.2(b). Use whatever categories and keywords you think are appropriate.

Answer:

```
SELECT * FROM IMAGEDB
WHERE KEYWORD = 'PIGS'
AND CATEGORY = 'ANIMALS'
```

or

```
SELECT * FROM IMAGEDB
WHERE KEYWORD = 'PIGS'
AND KEYWORD = 'PINK'
```

Exercise 2 Color Histogram Distances

Implement a $4 \times 4 \times 4$ color histogram distance measure that can imput two images and compare either the whole images or selected subimages of each. Use this basic measure to implement a gridded-color distance measure that allows the user to specify the dimensions of the grid and combines the distances between each pair of corresponding grid squares into a single distance as shown in equation **??**. Try your gridded histogram distance measure on several pairs of color images and with grid dimensions 1×1, 4×4, and 8×8.

Answer:
This is a programming problem with no specific answer.

Exercise 3 Texture Distance Measures

Pick several different texture measures from Chapter 7 and implement them as image distance measures that compare the texture in a subimage of a query image to that in a subimage of a database image. Then write a program that implements a gridded-texture distance measure and can call on any of these as the individual measures used in each of the grid squares. Compare results on a set of images using each of the individual measures and trying several different grid sizes. Test on a database of images that each have several regions of different textures.

Answer:
This is a programming problem with no specific answer.

Exercise 4 Shape Histograms

Write a program that implements a shape-histogram distance measure using the tangent angle at each pixel on the boundary of the shape. Make it rotationally invariant by rotating the histogram of the query image so that each bin gets a turn as the first bin and the result is the minimum distance returned by each of these rotations. Use your distance measure to compare shapes that you extract from real images either by thresholding or interactively.

Answer:
This is a programming problem with no specific answer.

Exercise 5 Boundary Matching

While there are many known algorithms for boundary shape matching, they have not been heavily used so far in content-based retrieval. Can you explain why?

Answer:
Boundary matching is not very useful, because 2 image regions of the same 3D object can have many different shapes depending on viewpoint and because segmentation is not yet good enough to guarantee that extracted regions really represent objects or even significant parts of objects.

Exercise 6 Sketch Matching

Design and implement a sketch-matching distance measure along the lines of the ART MUSEUM system. Use it to retrieve a set of known images according to the user's sketch.

Answer:
This is a programmimg problem with no specific answer.

Exercise 7 Flesh and Face Finding

Implement a flesh finder to find regions of flesh color. Select regions of a specified size and larger and try to find evidence of facial features: in particular, the eyes, nose, and mouth, in that order of priority. Based on the features you find, assign to each region the probability of being a face.

Answer:
This is a programming problem with no specific answer.

Exercise 8 Retrieval by Objects and Relationships

Obtain or write a program that segments a color image into regions based on color and, if possible, texture. Run the program on a set of training images in which each class of object, such as tiger, sky, jungle, is present in several images. Train a classification algorithm on these known regions according to their color and texture properties. Write a program that uses the segmenter and classifier to produce a set of labeled regions for an input image and then computes the spatial relationships *above*, *below*, *left-of*, *right-of*, and *adjacent-to* between pairs of regions. Then write an interactive front end that allows users to input a graph structure in which the nodes are objects from the training set and the edges are the required relationships. The program should return all database images that satisfy the user's query.

Answer:
This is a large programming project that is suitable for a term project in a computer vision course or an independent study project for a good student.

Exercise 9 Indexing

Suppose a set of images is to be indexed according to Laws' texture energy measures. Explain how you could use R-trees as your indexing mechanism for this system.

Answer:

First form regions that have similar Laws texture energy values, using a clustering algorithm, such as the K-means algorithm given at the beginning of Chapter 10. Now these regions can be indexed using R-trees.

Chapter 10

Image Segmentation

From: shapiro@cs.washington.edu Sent: Wednesday, August 06, 2003 2:21 PM To: stockman@cse.msu.edu Subject: SegmentationAnswers

Exercise 1 Isodata vs. K-means clustering

The isodata algorithm gave better results than the K-means algorithm on the football images in that it correctly grouped the dark green areas at the top of the image with those near the bottom. Why do you think the isodata algorithm was able to perform better than the K-means algorithm?

Answer: The isodate algorithm does not look for a fixed number of clusters K, but instead uses statistical measures to decide when to split and when to merge.

Exercise 2 Histogram mode seeking

Write a program that finds the modes of a multimodal histogram by first breaking it into two parts, as does the Otsu method in Chapter 3, and then recursively trying to break each part into two more parts, if possible. Test it on gray-tone and color images.

Answer:
This is a programming problem with no specific answer.

Exercise 3 Region Growing

Implement the Haralick region-growing operator as a program and use it to segment gray-tone images.

Answer:
This is a programming problem with no specific answer.

Exercise 4 Computing area and perimeter

Consider an image region represented by (a) a labeled image and (b) a chain code representation.

1. Give algorithms for computing the area and the perimeter of the region.
2. Give the running times of your algorithms.

Answer:

a. Labeled Image

1. Area: Scan through the image counting all pixels with the selected label. $O(image\_size)$
2. Perimeter: Find the first pixel via a left-right / top-bottom scan. Then trace the border using a border following procedure such as the one given in Section 10.3.1. $O(image\_size + border\_size) \simeq O(image\_size)$

b. Chain-Code Representation

1. Area: Area is very difficult to compute directly from the chain code. It would be easier to convert the code to a polygon [$O(chain\_length)$], and use polygon area algorithms, which are quite simple and $O(num\_segments)$.
2. Perimeter: This is just the length of the chain. $O(chain\_size)$ if not already computed and constant complexity if it is always stored with the chain.

Exercise 5 Testing for pixels in a region

Consider an image region represented by (a) a labeled image and (b) a polygonal approximation to the boundary.

1. In each case, give an algorithm for testing if an arbitrary pixel [r,c] is in that region.
2. Give the running times of your algorithms in terms of the appropriate parameters, ie. number of pixels in the region or number of segments in the polygonal approximation.

Answer:

a. Labeled Image: Region Label L

1. if I[r,c] = L then pixel [R,C] is in the region.
2. $O(1)$

b. Polygonal Approximation

1. Construct a line segment from pixel [r,c] to [maxr,maxc]. Count how many segments of the polygon it intersects. If the number of segments is odd, the point [r,c] is inside the polygon; if it is even, [r,c] is outside the polygon. This can have numerical stability problems, so it is best to compute several (say three) such line segments to three different points ([maxr,maxc],[maxr,minc],[minr,maxc]) and let them vote.
2. $O(image\_length)$ to create the segments and $O(num\_segments)$ to make the decision.

Exercise 6 Limitations of the border tracking algorithm.

The border tracking algorithm makes certain assumptions about the regions that it is tracking. Under what conditions could it fail to properly identify the border of a region?

Answer:

The border tracking algorithm algorithm fails if a region is only one pixel wide.

Exercise 7

Consider the contour following phase of the Canny edge detection algorithm. When following an image contour by tracing pixels of high gradient magnitude, would it be a good idea to select the next possible pixel only from the two neighbors that are perpendicular to the gradient direction? Why or why not? Show specific cases to support your answer.

Answer:

Stockman needs to answer this one.

Exercise 8 Measuring across Canny edges

Perform the following experiment. Obtain a program for the Canny edge detector or some image tool that contains it. Obtain some flat objects with precise parallel edges, such as razor blades, and some rounded objects, such as the shanks of drill bits. Image several of these objects in several different orientations: use high resolution scanning, if possible. Apply the Canny edge detector and study the quality of edges obtained, including the repeatability of the distance across parallel edges. Is there any difference between measuring the razor blades, which have "sharp edges" and measuring the drill bits, which may have "soft edges" in the image.

Answer:
This is an experiment problem and has no specific answer.

Exercise 9 Determining the type of a pixel

Give the code for the operator *pixeltype*, using a 3×3 neighborhood about a pixel to classify it as one of the types: isolated, start or end, interior, junction, and corner.

Answer:

This is an analysis and coding problem; it is related to the Yokoi connectivity algorithm given in the Haralick and Shapiro text, Vol. 1, pages 272-274. This is the 8-connectivity version. Given the 3 *times* 3 neighborhood about pixel p, first determine if p is an interior point, which is the case if p's value and that of each of its eight neighbors are the same. If p is not an interior point, the algorithm calculates a value called the *connectivity number* of p, which is the number of times a 4-connected neighbor of p has a different value than p and at least one pixel in the corresponding three-pixel corner has the same value as p. The connectivity number then tells us the classification of pixel p: 0) isolated, 1) edge, 2) connecting, 3) branching, 4) crossing. The labels interior and isolated come directly from the Yokoi algorithm. Start and end points are edge points in the Yokoi classification, but all points on the edges of regions are also edge points, so extra tests must be done to determine start and end. Junction points can be branching or crossing. Corners are also specialized edge points.

Exercise 10

This exercise follows the work of R. Kasturi: the problem is to apply the Hough transform to identify lines of text. Apply existing programs or tools and write new ones as needed to perform the following experiment. (a) Type or print a few lines of text in various directions and binarize the image. Add some other objects, such as blobs or curves. (b) Apply connected components processing and output the centroids of all objects whose bounding boxes are appropriate for characters. (c) Input the set of all selected centroids to a Hough line detection procedure and report on how well the text lines can be detected.

Answer:
This is an experimental exercise that does not have a specific answer.

Exercise 11 Burns compared to Hough

Implement both the Hough transform and the Burns operator for line finding and compare the results on real-world images having a good supply of straight lines.

Answer:
This is a programming exercise without a specific answer. We have found that the Burns operator works much better than the original Hough transform.

Exercise 12 Line Detection

Implement the following approach to detecting lines in a gray-tone image I.

for all image pixels I[R,C]
{
compute the gradient G_{mag} and G_{dir}
if $G_{mag} >$ threshold
then output [G_{mag},G_{dir}] to set H
}
detect clusters in the set H;

The significant clusters will correspond to the significant line segments in I.

Answer:
This is a programming problem with no specific answer.

Exercise 13 Fitting a line to 3 points

Using Equation 10.25, compute the parameters c_1 and c_0 of the best line through the points (0, -7), (2, -1) and (4, 5).

Answer:
This is for Stockman to do.

Exercise 14 normal equations

(a) Derive the matrix form for the equations constraining the 4 parameters for fitting a cubic polynomial $c_3x^3 + c_2x^2 + c_1x + c_0$ to observed data $(x_j, y_j), j = 1, n$. (b) From the pattern of the matrix elements, predict the matrix form for fitting a polynomial of degree four.

Answer:
This is for Stockman to do.

Exercise 15 fitting a planar equation to 3D points

(a) Solve for the parameters a, b, c of the model $z = f(x, y) = ax + by + c$ of the least squares plane through the five surface points (20, 10, 130), (25, 20, 130), (30, 15, 145), (25, 10, 140), (30, 20, 140). (b) Repeat part (a) after adding a random variation to each of the three coordinates of each of the five points. Flip a coin to obtain the variation: if the coin shows heads, then add 1, if it shows tails then subtract 1.

Answer:
This is for Stockman to do.

Exercise 16 Prewitt operator is "optimal"

Show that the Prewitt gradient operator from Chapter 5 can be obtained by fitting the least squares plane through the 3x3 neighborhood of the intensity function. To compute the gradient at $I[x, y]$, fit the nine points $(x + \Delta x, y + \Delta y, I[x + \Delta x, y + \Delta y])$ where Δx and Δy range through -1, 0, +1. Having the planar model $z = ax + by + c$ that best fits the intensity surface, show that using the two Prewitt masks actually compute a and b.

Answer:
This is for Stockman to do.

Exercise 17

Figure 10.27shows a dark ring on a light background centered at the image origin. Sketch the parameter space that would result when such an image is transformed using the Hough Transform as shown in Figure 10.26

Answer:
This is for Stockman to do.

Exercise 18 Hough Transform for Ribbons

Write a computer program to study the use of the Hough Transform to detect ribbons. (a) Use the Sobel operator to extract the gradient magnitude and direction at all pixels. Then transform only pixels of high magnitude. (b) Detect any clusters in $[d, \theta]$-space. (c) Detect pairs of clusters, $([d_1, \theta_1], [d_2, \theta_2])$, where θ_1 and θ_2 are π apart. (d) Delete pairs that are not approximately symmetrical across an axis between them.

Answer:
This is a programming problem with no specific answer.

Exercise 19

Describe how to change the ribbon detection algorithm so that (a) it only detects ribbons that are nearly vertical, (b) it detects ribbons that are no wider than W.

Answer:
This is for Stockman to do.

Exercise 20

Given two lines parameterized by $([d_1, \theta_1], [d_2, \theta_2])$, (a) derive a formula for their intersection point $[x, y]$ and (b) derive a formula for their axis of symmetry $[d_a, \theta_a]$.

Answer: This is for Stockman to do.

Exercise 21

Obtain two successive images of a scene with moving objects and compute a spatio-temporal image from them using Equation 10.27. (Two frames from a Motion JPEG video would be good. Or, one could digitize some dark cutouts on a flatbed scanner, moving them slightly for the second image.)

Answer:
This is an experimental exercise with no specific answer.

Exercise 22

Describe how Algorithm 10.10 can be modified to make it simpler and faster by specializing all of its steps for the ASL application.

Answer:

This is for Stockman to do.

Exercise 23

Suppose we have two motion trajectories $\mathbf{P}_j, j = 1, N$ and $\mathbf{Q}_k, k = 1, M$, where $\mathbf{P}_j$ and $\mathbf{Q}_k$ are points in 2D in proper time sequence. Devise an algorithm for matching two such trajectories, such that 1.0 is output when both trajectories are identical and 0.0 is output when they are very different. Note the M and N may not be equal.

Answer:

This is for Stockman to do.

Chapter 11

Matching in 2D

Exercise 1

Describe how to enhance the right image of Figure 11.2 to lessen the quantization or aliasing effect.

Exercise 2 scaling for a non-square pixel camera

Suppose a square CCD chip has side 0.5 inches and contains 480 rows of 640 pixels each on this active area. Give the scaling matrix needed to convert pixel coordinates [r, c] to coordinates [x, y] in inches. The *center of* pixel [0,0] corresponds to [0, 0] in inches. Using your conversion matrix, what are the integer coordinates of the center of the pixel in row 100 column 200?

Exercise 3

(a) Sketch the three points [0,0], [2,2], and [0,2] using an XY coordinate system. (b) Scale these points by 0.5 using Equation 11.2 and plot the results. (c) Using a new plot, plot the result of rotating the three points by 90 degrees about the origin using Equation 11.4. (d) Let the scaling matrix be S and the rotation matrix be R. Let SR be the matrix resulting from multiplying matrix S on the left of matrix R. Is there any difference if we transform the set of three points using SR and RS?

Exercise 4 rotation about a point

Give the 3x3 matrix that represents a $\pi/2$ rotation of the plane about the point [5, 8]. *Hint*: first derive the matrix $\mathbf{D}_{-5,-8}$ that translates the point [5, 8] to the origin of a new coordinate frame. The matrix which we want will be the combination $\mathbf{D}_{5,8}\ \mathbf{R}_{\pi/2}\ \mathbf{D}_{-5,-8}$. Check that your matrix correctly transforms points [5, 8],[6, 8] and [5, 9].

Exercise 5 reflection about a coordinate axis

A *reflection* about the y-axis maps the basis vector [1, 0] onto [-1, 0] and the basis vector [0, 1] onto [0, 1]. (a) Construct the matrix representing this reflection. (b) Verify that the matrix is correct by transforming the three points [1, 1], [1, 0], and [2, 1].

Exercise 6 converting image coordinates to workbench coordinates

Assume an environment as in Figure 11.5. (Perhaps the vision system must inform a pick-and-place robot of the locations of objects.) Give, in matrix form, the transformation that relates image coordinates $[x_i, y_i, 1]$ to workbench coordinates $[x_w, y_w, 1]$. Compute the four parameters using these control points: ${}^{i}\mathbf{P_1} = [100, 60]$, ${}^{w}\mathbf{P_1} = [200, 100]$; ${}^{i}\mathbf{P_2} = [380, 120]$, ${}^{w}\mathbf{P_2} = [300, 200]$.

Exercise 7 are transformations commutative?

Suppose we have matrices for three primitive transformations: $\mathbf{R}_\theta$ for a rotation about the origin, $\mathbf{S}_{s_x,s_y}$ for a scaling, and $\mathbf{D}_{x_0,y_0}$ for a translation. (a) Do scaling and translation commute; that is, does $\mathbf{S}_{s_x,s_y}\ \mathbf{D}_{x_0,y_0} = \mathbf{D}_{x_0,y_0}\ \mathbf{S}_{s_x,s_y}$? (b) Do rotation and scaling commute; that is, does $\mathbf{R}_\theta\ \mathbf{S}_{s_x,s_y} = \mathbf{S}_{s_x,s_y}\ \mathbf{R}_\theta$? (c) Same question for rotation and translation. (d) Same question for scaling and translation. Do both the algebra and intuitive thinking to derive your answers and explanations.

Exercise 8

Construct the matrix for a reflection about the line y=3 by composing a translation with $y_0 = -3$ followed by a reflection about the x-axis. Verify that the matrix is correct by transforming the three points [1, 1], [1, 0], and [2, 1] and plotting the input and output points.

Exercise 9

Verify that the product of the matrices $\mathbf{D}_{x_0,y_0}$ and $\mathbf{D}_{-x_0,-y_0}$ is the 3x3 identity matrix. Explain why this should be the case.

Exercise 10

Solve Equation 11.17 using the following three pairs of matching control points: ([0,0],[0,0]), ([1,0],[0,2]), ([0,1],[0,2]). Do your computations give the same answer as reasoning about the transformation using basis vectors?

Exercise 11

Solve Equation 11.17 using the following three pairs of matching control points: ([0,0],[1,2]), ([1,0],[3,2]), ([0,1],[1,4]). Do your computations give the same answer as reasoning about the transformation using basis vectors?

Exercise 12

Take three pairs of matching control points from Figure 11.10 (for example, ([288, 210, 1], [31, 160, 1])) and verify that the affine tranformation matrix maps the first into the second.

Exercise 13 Consistent Labeling Problem

Show that the labeling f given above is a consistent labeling. Because the relations are symmetric, the following modified constraint must be satisfied:

If $(p_i, p_{i'}) \in R_P$, then $(f(p_i), f(p_{i'})) \in R_L$ or $(f(p_{i'}), f(p_i)) \in R_L$.

Answer:

R_P	R_L	Satisfied
(S1,S2)	(Sj,Sa)	yes
(S1,S5)	(Sj,Si)	yes
(S1,S6)	(Sj,Sk)	yes
(S2,S3)	(Sa,Sb)	yes
(S2,S4)	(Sa,Sn)	yes
(S3,S4)	(Sb,Sn)	yes
(S3,S9)	(Sb,Sd)	yes
(S4,S5)	(Sn,Si)	yes
(S4,S7)	(Sn,Sg)	yes
(S4,S11)	(Sn,Sh)	yes
(S5,S6)	(Si,Sk)	yes
(S5,S7)	(Si,Sg)	yes
(S5,S11)	(Si,Sh)	yes
(S6,S8)	(Sk,Sl)	yes
(S6,S11)	(Sk,Sh)	yes
(S7,S9)	(Sg,Sd)	yes
(S7,S10)	(Sg,Sf)	yes
(S7,S11)	(Sg,Sh)	yes
(S8,S10)	(Sl,Sf)	yes
(S8,S11)	(Sl,Sh)	yes
(S9,S10)	(Sd,Sf)	yes

Exercise 14

Give detailed justification for each of the labels being in or out of each of the label sets after pass 1 as shown in Table 11.5.

Answer:

Stockman answers this one.

Exercise 15 Continous Relaxation

Figure 11.20 shows a model and an image, each composed of line segments. Two line segments are said to be in the relationship *closadj* if their endpoints either coincide or are close to each other. a) Construct the attributed relation R_P over the parts of the model defined by $R_P = \{(p_i, p_j, d) | p_i \ closadj \ p_j\}$ and the attributed relation R_L over the labels of the image defined by $R_L = \{(l_i, l_j) | l_i \ closadj \ l_j\}$. b) Define the compatibility coefficients by $c_{ij} = 1 \ if \ (p_i, p_j) \in R_P \ else \ 0$. Use R_P and R_L together to define R in a manner of your choosing. Let $pr_i(l_j)$ be given as 1 if p_i has the same orientation as l_j, 0 if they are perpendicular, and .5 if one is diagonal and the other is horizontal or vertical. Define pr for the parts of the model and labels of the image. c) Apply several iterations of continuous relaxation to find a probable labeling from the model parts to the image labels.

Answer:

a) Relations R_P and R_L (variable d is not needed)

R_P	R_L
(p1,p2)	(l1,l2)
(p1,p4)	(l1,l4)
(p2,p3)	(l1,l5)
(p3,p4)	(l2,l3)
	(l2,l4)
	(l2,l5)
	(l2,l7)
	(l5,l6)
	(l6,l7)

b) $c_{12} = c_{13} = c_{23} = c_{34} = 1$

i	j	$pr_i(l_j)$	i	j	$pr_i(l_j)$
1	1	1	3	1	1
1	2	0	3	2	0
1	3	1	3	3	1
1	4	.5	3	4	.5
1	5	1	3	5	1
1	6	0	3	6	0
1	7	1	3	7	1
2	1	0	4	1	0
2	2	1	4	2	1
2	3	0	4	3	0
2	4	.5	4	4	.5
2	5	0	4	5	0
2	6	1	4	6	1
2	7	0	4	7	0

$R_{ij}(l, l') = pr_i(l_l) \times pr_j(l'_l) \times close(i, j, l, l')$ where $close(i, j, l, l') = 1$ if $(p_i, p_j) \in R_P \Rightarrow (l_l, l'_l) \in R_L$ and 0 otherwise.

c) Part c requires writing and running a program.

Exercise 16 Relational Distance Treesearch

Modify the algorithm for Interpretation Tree Search to find the relational distance between two structural descriptions and to determine the best mapping in the process of doing so.

Answer:

P is the set of detected image features.
L is the set of stored model features.
R_P is the relationship over the image features.
R_L is the relationship over the model features.
f is the labeling being constructed, initially NIL.
f_error is the error of f so far, initially 0.
$best\_map$ is the global best mapping so far.
$bound\_error$ is the global lowest error so far.

```
procedure Interpretation_Tree_Search(P, L, R_P, R_L, f, f_error);
{
p := first(P);
for each l in L
   {
   f' = f ∪ {(p, l)}; /* add part-label to interpretation */
   f'_error = f_error;
   /* check on relations */
   for each N-tuple (p_1, ..., p_N) in R_P containing component p
      and whose other components are all in domain(f)
      if (f(p_1), ..., f(p_N)) is not in R_L
      then f'_error = f'_error + 1;
   for each N-tuple (l_1, ..., l_N) in R_L containing component l
      and whose other components are all in range(f)
      if (f^-1(l_1), ..., f^-1(l_N)) is not in R_P
      then f'_error = f'_error + 1;
   if f'_error < bound_error then
      {
      P' = P - p;
      L' = L - l;
      if isempty(P') then
         {
         best_map = f';
         bound_error = f'_error;
         }
      else Interpretation_Tree_Search(P', L', R_P, R_L, f', f'_error);
      }
    If you get here, the error is not less than the bound, so do nothing */
   }
}
```

Exercise 17 One-Way Relational Distance

The above definition of relational distance uses a two-way mapping error, which is useful when comparing two objects that stand alone. When matching a model to an image, we often want to use only a one-way mapping error, checking how many relationships of the model are in the image, but not vice versa. Define a modified one-way relational distance that can be used for model-image matching.

Answer:
The only change needed is in equation 11.22; for one-way relational distance, it changes to:

$$E_S^i(f) =\mid R_i \circ f - S_i \mid$$

Exercise 18 NIL mappings in Relational Distance

The above definition of relational distance does not handle NIL labels explicitly. Instead, if part j has a NIL label, then any relationship (i, j) will cause an error, since $(f(i), NIL)$ will not be present. Define a modified relational distance that counts NIL labels as errors only once and does not penalize again for missing relationships caused by NIL labels.

Answer:

Again, the change should be to the error computation. For the relational part of the error, equation 11.22 changes to:

$$E_S^i(f) =\mid \{(a_1, \ldots, a_n) \in R \ni \forall i \ f(a_i) \neq NIL \ and \ (f(a_1), \ldots, f(a_n)) \notin S\} \mid$$
$$+ \mid \{(b_1, \ldots, b_n) \in S \ni \forall i \ f^{-1}(b_i) \neq NIL \ and \ (f^{-1}(b_1), \ldots, f^{-1}(b_n)) \notin R\} \mid$$

Then there would be a separate error that might merely sum the number of parts that have NIL corresponding labels and the number of labels that have NIL corresponding parts.

Exercise 19 Attributed Relational Distance

The above definition also does not handle attributed relations in which each tuple, in addition to a sequence of parts, contains one or more attributes of the relation. For example, a connection relation for line segments might have as an attribute the angle between the connecting segments. Formally, an attributed n-relation R over part set A and attribute set P is a set $R \subseteq A_n \times P_m$ for some nonnegative integer m that specifies the number of attributes in the relation. Define a modified relational distance in terms of attributed relations. (Note: sets A and P have been reversed for the following answer.)

Answer:

¿From Haralick and Shapiro, Vol. I, page 464:

Consider an attributed relation $R \subseteq A_n \times P_m$ over part set A and property value set P. Let $r \in R$ be an $n+m$-tuple having n parts followed by m property values. Let $S \subseteq B^n \times P^m$ be a second attributed relation over part set B and property value set P. We define the composition $r \circ f$ of attributed tuple r with f by

$$r \circ f = \{(b_1, \ldots, b_n, p_1, \ldots, p_m) \in B^n \times P^m$$
$$\mid \textit{ there exists } (a_1, \ldots, a_n, p_1, \ldots, p_m) \in R \textit{ with } f(a_i) = b_i, i = 1, \ldots, n\}$$

Assume that if $(b_1, \ldots, b_n, p_1, \ldots, p_m) \in S$ and $(b_1, \ldots, b_n, q_1, \ldots, q_m) \in S$ then $p_1 = q_1, \ldots p_m = q_m$. That is, each n-tuple of parts has only one m-tuple of properties. The error of a tuple $t = (b_1, \ldots, b_n, p_1, \ldots, p_m)$ with respect to a relation $S \subseteq B^n \times P^m$ is given by

$$e(t, S) = norm\_dis[(p_1, \ldots, p_m), (q_1, \ldots, q_m)]$$

if $(b_1, \ldots, b_n, q_1, \ldots, q_m) \in S$ and 1 otherwise, where $norm\_dis$ returns the Euclidean distance (or any other desired distance) between two vectors, normalized by dividing by some maximum possible distance. Thus $e(t, S)$ is a quantity between 0 and 1. Now we can extend the definition of the structural error of f for the ith pair of corresponding relations to

$$E_s^i(f) = \sum_{r \in R_i} e(r \circ f, S_i) + \sum_{s \in S_i} e(s \circ f^{-1}, R_i)$$

Total error and relational distance follow from the above.

Exercise 20

(a) Determine the transformation that maps a circular region of an image onto a hemisphere and then projects the hemisphere onto an image. The circular region of the original image is defined by a center (x_c, y_c) and radius r_0. (b) Develop a computer program to carry out the mapping.

Exercise 21

Show that radial distortion in Equation 11.25 with $c_4 = 0$ can be modeled exactly by a polynomial mapping of the form shown in Equation 11.26.