RAFFLES INSTITUTION

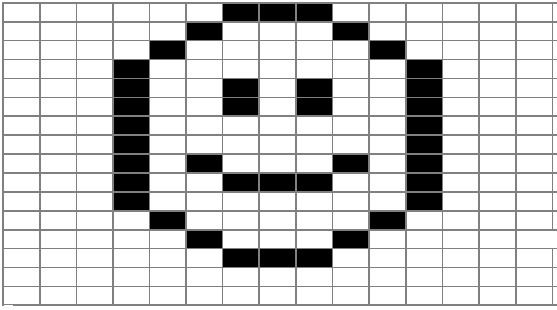
RAFFLES PROGRAMME 2016

Year Two Computer Elective Programme Problem Set 3

Founded 1823Digital Image Processing – Pycture Perfect

Question 1

1. Drawing Smiley Face Take a look at this simple picture.



Write a program, smileyFace()that will draw and save this smiley face on a graphic file, named “smiley.gif”.

1. Rotating Smiley Face

Write a function, rotateImage90CW, such that the image can be rotated clockwise.

Question 2: Pixel-by-Pixel Image Processing

Correcting Photo Exposure using simple methods

With reference to negative and grayscale processing in the notes, devise two new rgbFunctions, incExposure() and decExposure()

incExposure() - increases the exposure of the image by multiplying each of the RGB channel by a exposure factor.

decExposure() - decreases the exposure of the image by dividing each of the RGB channel by a exposure factor.

For instance, for an underexposed photo, after applying an appropriate amount of exposure, the photo will look much better.



Apply these new rgb Functions to images that are either underexposed or overexposed.

Submit your completed functions, applied to 2 chosen image files of your choice.

Remember to submit your chosen gif files.

Choose any 1 challenge from this list of challenges.

Have fun experimenting the cImage library!

Challenge 1:

Take a picture of yourself against a white background. Use the fact that you can“filter” out all the white pixels to place your picture in an interesting scene. This same process is used all the time by weather forecasters on television. The only difference is that they stand in front of a solid blue or solid green background called a chromakey.

Challenge 2:

Another way to put yourself in an interesting picture is to take a picture of yourself against a relatively plain background, then take another picture of exactly the same background (use a tripod here with autofocus off) without you in it. Now you can compare the two images and remove the pixels that are exactly the same, or close to the same. Once you have removed those pixels, you can superimpose yourself on any background.

Challenge 3:

Using getMouse to get the coordinates of a pixel in an image, devise a way to remove the red-eye effect from the area of the image you click on.

Challenge 4:

Using getMouse, write a program that will allow you to “cut” a rectangular region out of an image and place it somewhere in a new image.

Challenge 5:

Research convolution kernels and implement some interesting kernels.