GRAPHICS GROUP

IVS Intelligent Vision Systems New Zealand

Computer Graphics and Image Processing

Part 3: Image Processing Assignment Specification

Martin Urschler, PhD

Assignment: QR Code Detection

- Aim: Detect Bounding Box around QR Codes in an Image

Assignment: QR Code Detection

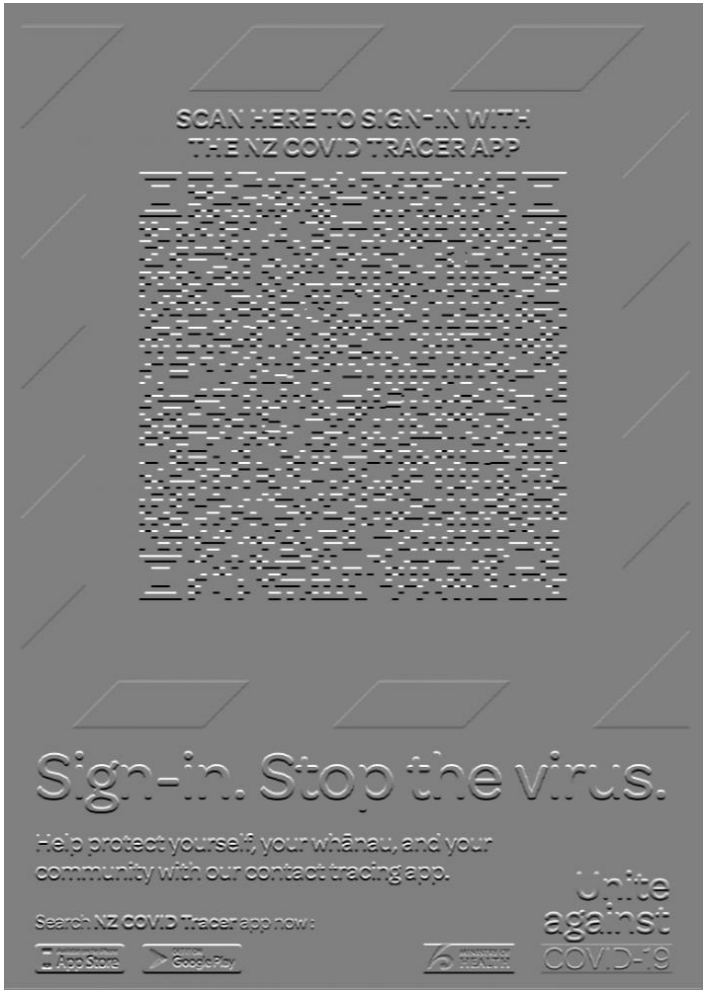
- How?
  - We will use the image processing techniques that we study in the lecture (and implement in Coderunner)
    - Conversion to Greyscale
    - Contrast Stretching
    - Filtering to detect edges
    - Filtering to average
    - Thresholding for Segmentation
    - Morphological operations
    - Connected component analysis

Assignment: QR Code Detection

- Algorithm
  - Step 1: read the input image, convert RGB data to greyscale and stretch the values to lie between 0 and 255
  - See Coderunner Programming Examples in Week 10

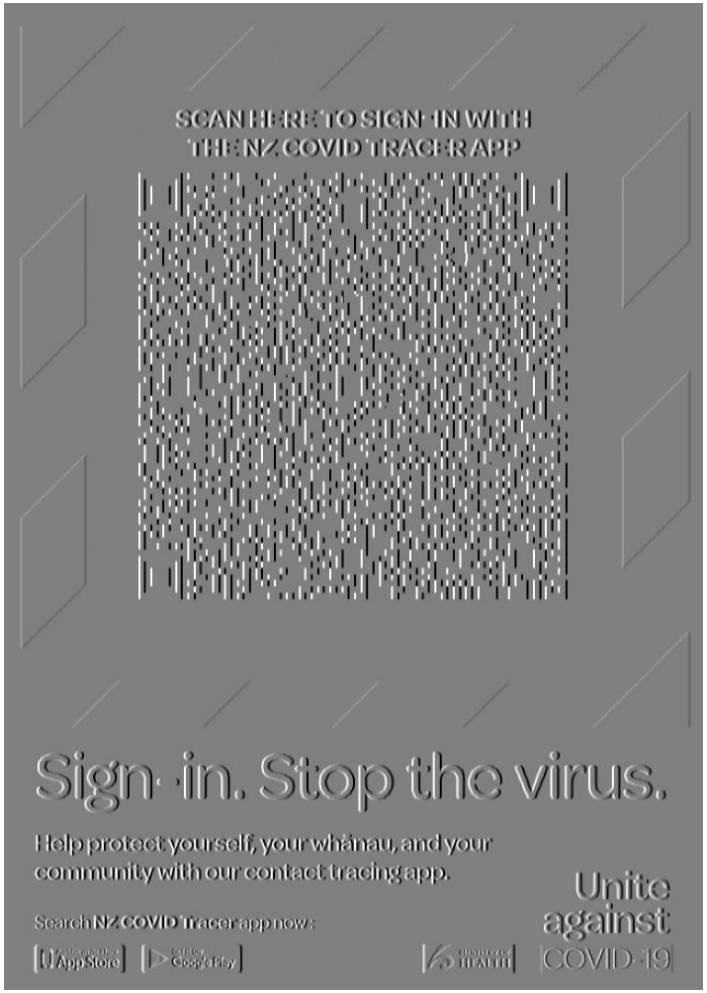
Assignment: QR Code Detection

- Algorithm
  - Step 2: compute horizontal edges, compute vertical edges, compute edge magnitude
  - See Coderunner Programming Examples in Week 11
  - Hint: Use 3x3 Sobel filter masks for edges



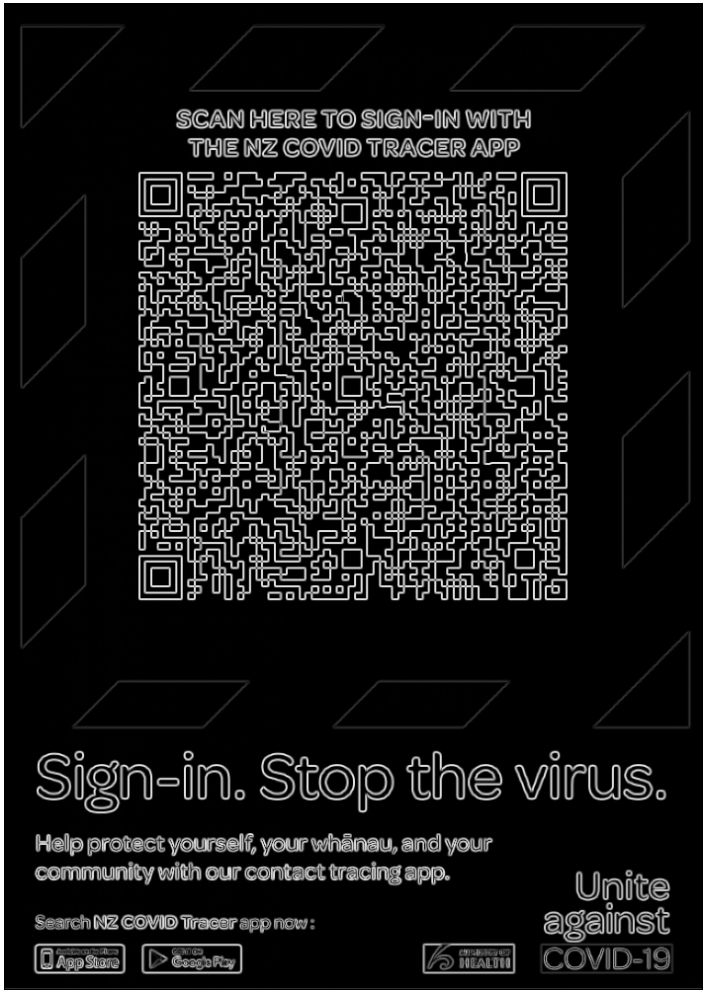
Assignment: QR Code Detection

- Algorithm
  - Step 3: compute horizontal edges, compute vertical edges, compute edge magnitude
  - See Coderunner Programming Examples in Week 11
  - Hint: Use 3x3 Sobel filter masks for edges



Assignment: QR Code Detection

- Algorithm
  - Step 4: compute horizontal edges, compute vertical edges, compute edge (gradient) magnitude
  - See Coderunner Programming Examples in Week 11



Assignment: QR Code Detection

- Algorithm
  - Step 5: smooth over the edge magnitude (mean or Gaussian), and stretch contrast to 0 and 255
  - See Coderunner Programming Examples in Week 11
  - Hint: you can use the 3x3 mean filter and repeat it several times

Assignment: QR Code Detection

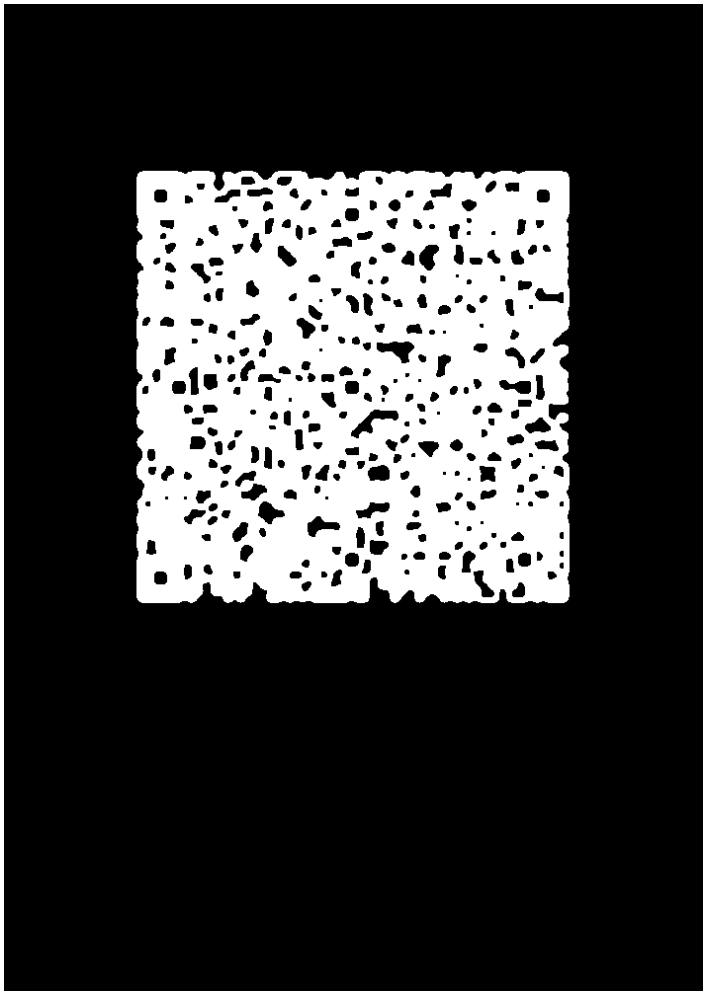
- Algorithm
  - Step 6: perform a thresholding operation to get the edge regions as a binary image
  - See Coderunner Programming Examples in Week 11-12
  - Hint: a good threshold value is 70, if you did the contrast stretching in step 5!

Assignment: QR Code Detection

- Algorithm
  - Step 7: (optional) perform a morphological closing operation to fill holes
  - See Coderunner Programming Examples in Week 12
  - Hint: you could try several dilations followed by several erosions, to get rid of larger holes

Assignment: QR Code Detection

- Algorithm
  - Step 8: perform a connected component analysis to find the largest connected object
  - See Coderunner Programming Examples in Week 12



Assignment: QR Code Detection

- Algorithm
  - Step 9: Extract the bounding box around this region, by looping over the image and looking for the minimum and maximum x and y coordinates of foreground pixels (>0)
  - Your code should show an imshow based plot like this when we run it!

Assignment: QR Code Detection

- Organization
  - Download Python code skeleton from my github (see Assignment description on Canvas for the link)
  - Zip your solution and submit it via Canvas to the Assignment Dropbox
  - Note: Work on assignment at the same time as Coderunner quizzes!
  - 10 marks for solving task for this image
  - Deadline Sunday June 6, 23:59
  - 5 marks for your extension, together with a short reflective report (no report needed for main task!)

Assignment: QR Code Detection

- Important: Use a lab computer to test if your code works on Windows on a different machine (270 students!)

- Examples for extensions (describe in reflective report)
  - Make sure the bounding box works also for rotated images
  - Experiment with the images in folder ‘challenging’. Make sure your code works also on these images. Find more images. Discuss why you think the code fails on some images and what you could do about it.
  - Combine with Python lib ‘pyzbar’ to actually decode the QR code
  - Think about your own ideas for an extension…

Assignment: QR Code Detection