Lab Submission

Lego's Life of George is a game where an image is displayed on the screen for a number of seconds before being blanked. The player of the game then has to construct the shape from Lego blocks from memory. The player photographs the finished blocks and their computer automatically compares the brick pattern to the original image. Similar problems are found in many image processing applications, for example the colour segmentation of real world objects; colour data matrix reading. A colour data matrix has the advantage over conventional blank and white versions in that more information can be stored for a given pattern. An example is shown in figure 1.

For your project you will write a function that automatically reads a colour pattern image from hard-disc and returns an array representing the colour pattern. The function should be of the form: `result=colourMatrix(`*filename*`)`. An example image is shown in figure 1. Note the location of the black circles. These are for finding the edge of the image should you need to use them. **There is no correct orientation - so your results may be flipped.**

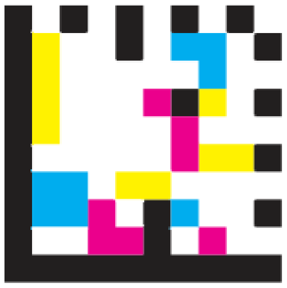

Figure 1: Colour Data matrix

The colours used are red, green ,yellow, blue and white. Your function should return an array of the form:[5]

$$\texttt{result} = \begin{bmatrix} g & y & y & w \\ b & w & g & g \\ w & g & y & r \\ r & g & w & r \end{bmatrix}$$

The images we become progressively more difficult and you may not be able to solve them all. Ideally one function should submitted, but you may submit multiple functions if you need to. Ideally the process should be fully automatic (for full marks) but this is not easy so some manual intervention is allowed. Some images maybe impossible, if you can not solve them please explain why.

[5]You could also use a numeric code to represent the colours. This will be easier since returning the letters code can be fiddly if you do not know about cell arrays in Matlab.

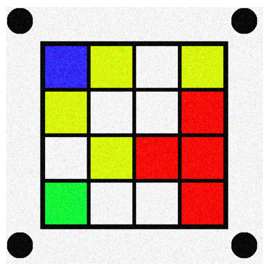

Figure 2: Colour Target

The Image Files

There are three sets of images on Canvas

- `images1.zip` are a set of simulated images
- `images2.zip` are a set of simulated images but have a black boarder which makes them harder to solve.
- `Photos.zip` are a set of real photographs. Do not attempt to solve these, but comment on each one in your report.
- You do not need to solve images1 and images2. Pick one of them.
- You can get a 100% mark for using either set of data.
- The result mark is weighted so getting 70% of the `image2.zip` will result in full marks for the results section. The real images do not carry marks but please discuss the problems and if you algorithm does not work, explain why.

Report

- The length of the report should not exceed 3000 words (about 10 pages) plus an appendix containing the results tables and a full code with listings of your code (containing comments).
- This is to be submitted on-line. See your timetable for the exact date and time.
- Candidates are also asked to hand in a single **ZIP** file containing their Matlab code and their report [6].

[6]I will also accept gzipped tarballs and Python code

- The report format should be a **PDF** file only. Please **do not submit Word files**. I'm marking this on a Linux machine with no wWrd installed so submitting word documents might get corrupted and will certainly make me grumpy.

- The report must describe the approach you took to writing the functions, explain how they work, and discuss any design decisions. The report should include an assessment of the performance of the functions, and suggest improvements that could be made. The main performance criterion is the accuracy of the matches, but you could also consider the time taken to find them.

- You can report on any computational experiments you did, including showing Matlab scripts you used but which are not part of your final system. You should include images in the report where they help show what you have done.

- You may also discuss any techniques you have found out about, but which you have not had time to implement.

- You should not submit copies of Matlab's own functions or toolbox functions, though your code can call such functions freely.

- Programs that find accurate matches are, of course, likely to do well. However, the quality of the code and report will be important factors regardless of performance, and originality and ingenuity will be taken into account.

You can use any approach you wish — though if you want to try something that seems very different to the techniques described in the course it would be sensible to talk to me first. You are free to use any Matlab function or to download any addition Matlab library as long as it is fully referenced.

Marks

The marks break down is given in the table 1.

Report	50%	Quality of report, details of techniques used and why
Colour square discovery method	10 %	Does it make sense? Assumptions made.
Automatic realignment of the squares	20%	Is the process fully automated, is only one function called or are different functions used for each image?
Results	15%	Are the result accurate. Have all the test images been run.
Code quality	5%	Is the code well written and commented.

Table 1: Marks breakdown