CIS*4720 Assignment 2
Intermediary Image Processing: analyzing colour images

Deadline: Wednesday, March 6, 9am
Weight: 25%

Overview

The series of topics which make up this assignment have to do with aspects of image processing and computer vision. In many cases it involves performing some form of processing on an image and then extracting information.

This assignment gives you a chance to explore one specific topic in some depth. The hands-on approach is the best way to become familiar with image processing - you learn by doing. You have a number of topics to choose from. Some of them require you to design and implement one specific algorithm, using the reference algorithms as jumping-off points. Others focus on comparing existing algorithms. Pick whatever you think would be more fun.

You can work in pairs or individually on this assignment.

Learning outcomes

Upon completing this assignment, students will be able to implement some more complex image processing algorithms to process and analyze images. Students will build skills in problem solving, algorithm design, evaluative processes and developing testing regimes.

Task

Choose one of the topics, use the references provided and any other relevant material, and implement the algorithm(s) as specified in the topic descriptions. Test the algorithms (moderate testing with 7-8 differing images), and present your results in the form of a research report. Your report should include the following:

Key Issues:

- Provide a discussion of the algorithms and their pseudo-code.
- Describe the experiments you carried out, provide a selection of appropriate results and list the important factors indicated by the results.

References:

Each topic contains a list of references, however you are not restricted to using those references. A good source of related papers on each of the topics can be found on CiteSeer (http://citeseerx.ist.psu.edu). Find a paper, then check on the “Citations” tab. They often have PDF access to the papers as well. If not, check the UofG’s Library search engine (see announcement on Moodle for more details on how to use it).

Code and Results:

Algorithms should be implemented in Python, or Julia, or some similar language. For example, the code could also be in the form of an ImageJ plug-in, C, or Matlab if you wish. Please supply the all the code you used to generate your results, along with a README file or

sufficient comments to allow us to test it out. The source code - a full listing of your program containing comments which clearly indicate the function of each component - should be submitted with the assignment. Code should be well-styled and appropriately documented.

Report:
Present your results in the form of a small report (5-6 pages) and submitted as a PDF. Remember to reference appropriate material used in the investigation. To make life easier, you pretend that you are writing a conference paper, and use one of the conference templates from IEEE (Word of LateX). https://www.ieee.org/conferences_events/conferences/publishing/templates.html

Test image results illustrating the results of particular algorithms can be included in a compressed folder. However, make sure you downsize the images, so your overall submission does not exceed 100MB.

Grading Criteria
This assignment is graded using Rubric 2. The grade for this project will be based on the evidence that you have analyzed and implemented the algorithms, compared the algorithms, and provided a quantitative analysis of the results. Your grade will depend on the feasibility of your testing process, but also on a clear presentation of your results and good writing style. It is your responsibility to find a way to explain clearly how you solved the problem. You can also describe test images that failed, and problems encountered along the way.

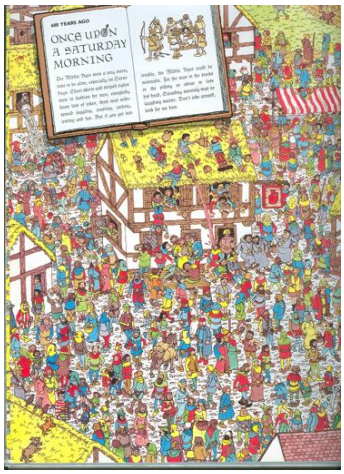
Note, that for a grading of 90%+ for this project you will need to demonstrate that you have exceeded the requirements of the project. To exceed the requirements, you must show that your algorithm contains a clear, well-defined structure, is innovative and original and can be validated using some quantitative measure, or comparison with an existing solution. You may also have used novel metrics, or decided to compare more than two algorithms.

In addition, for a grading of 90%+ you could try applying a technique from a previous course (e.g. AI), if you think you can adapt to to the problems described below.

Assignment submission:
Ideally you should submit all your files in one archive (zip) containing the source code, README, example images, report, and anything else you might like to submit. You can also upload the files separately.

1. Where is Waldo?

This is an exercise based on the “Where’s Waldo” book series by Martin Handford. In each image Waldo is hiding in the midst of a busy crowd scene. Waldo can be identified by a distinct red and white striped sweater and hat. However what he is doing may vary from scene to scene, for example he may be carrying a stack of books.



Task: Write an algorithm to detect Waldo. You will also be given a series of images of Waldo to help you in solving this problem. Unfortunately the scenes also contain several Waldo doppelgängers. Here are some of the fake Waldo’s to avoid:

To help solve this problem, it is necessary to look at the intrinsic characteristics of Waldo. How do humans find Waldo? Test your algorithm on a series of images, both easy and hard.

Potential algorithms you may wish to use:

- Ennesser, F., Medioni, G., “Finding Waldo, or focus of attention using local color information”, IEEE Trans. Pattern Analysis and Machine Intelligence, Vol.17(8), pp.805-809 (1995)
- Yoo, T-W., Oh, I-S., “A fast algorithm for tracking human faces based on chromatic histograms”, Pattern Recognition Letters, Vol.20, pp.967-978 (1999)

Hints:

- Is it possible to find Waldo by means of facial recognition?
- Maybe it’s possible to use some form of template matching (on the stripped shirt)?
- Using colour - his sweater is a distinctive shade of red.

2. Spot the Fire

One of the goals of vision-based fire detection is to identify fire in structures where a traditional fire-detection devices may be challenged. This includes (i) enclosed regions such as rail and road tunnels, (ii) atriums, warehouses and parking garages, (iii) historic buildings, (iv) external spaces such as farms, and lumber mills, and even (v) household kitchens.

Task: Design and implement an algorithm to extract the fire region from a series of images. Extensively test a series of images and derive criteria to measure success. Some ground truth images will be provided.

Potential algorithms you may wish to reference or use:

- Celik, T., “Fast and efficient method for fire detection using image processing”, ETRI Journal, Vol.32(6), pp.881-890 (2010) http://etrij.etri.re.kr/Cyber/Download/PublishedPaper/3206/etrij.dec2010.0881.pdf
- Ebert, J., Shipley, J., “Computer vision based method for fire detection in color videos”, http:// digital.cs.usu.edu/~xqi/Teaching/REU07/Website/Jessica/JessicaFinalPaper.pdf
- Wirth, M.A. Zaremba, R., “Flame region detection based on histogram backprojection”, Canadian Conference on Computer and Robot Vision, IEEE, pp.167-174 (2010)

Hints:

• Use some form of colour space other than RGB. Be careful of false positives.

3. Measuring Canopy Cover

Publicly accessible aerial and satellite images (e.g. Google maps) can be used by community groups and municipalities interested in developing green space to estimate vegetative cover. The challenge is that such images often contain objects that have the appearance of vegetative cover. For example in the example image below, false-positives include the tennis courts, and to a less extent one of the pools.

Task: Design and implement an algorithm for extracting the vegetative cover from aerial/ satellite images (via segmentation). Provide a comparison with other (maybe more generic) segmentation methods. The reference below will give you some insight (and you can use it to compare against). This is essentially a colour segmentation problem, but false-positives will be an issue. Is there a way to differentiate different types of vegetation? Extensively test a series of images. Some images with ground truths will be provided.

Potential algorithms you may wish to reference or use:

- Perez, A., Lopez, F., Benlloch, J., Christensen, S., “Color and shape analysis techniques for weed detection in cereal fields,” Computers and Electronics in Agriculture, 25, pp.197–212 (2000).
- Woebbecke, D., Meyer, G., Bargen, K.V., Mortensen, D., “Color indices for weed identification under various soil, residue and lighting conditions,” Trans. of ASAE, 38, pp. 259–269 (1995).
  HINTS:

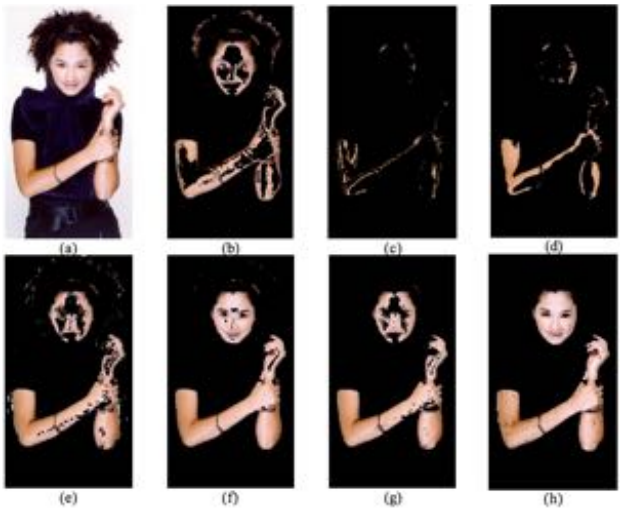
The best way to approach this problem is through some form of colour segmentation.

Expectation maximization or histogram backprojection may be interesting approaches.

Look for literature that relates to weed detection, or try different image segmentation techniques (some of them could be in existing libraries such as OpenCV).

4. Skin Detection Algorithms

Skin area detection has been a key to different recognition algorithms such as facial recognition, and human motion detection. There are many varied algorithms for detecting skin, but how well are they designed to deal with the varied skin tones found in humans? One of the easiest algorithms is histogram backprojection, but this works based on the assumption that a model image is provided. There are many algorithms for skin detection, many of which are quite complex, but it is often the simple ones that are interesting.



Task: Compare at least three skin detection algorithms, by implementing them, and designing an extensive testing regime to test against a good cross-section of skin-tones. Does any one algorithm work better? Some ground truth images will be provided.

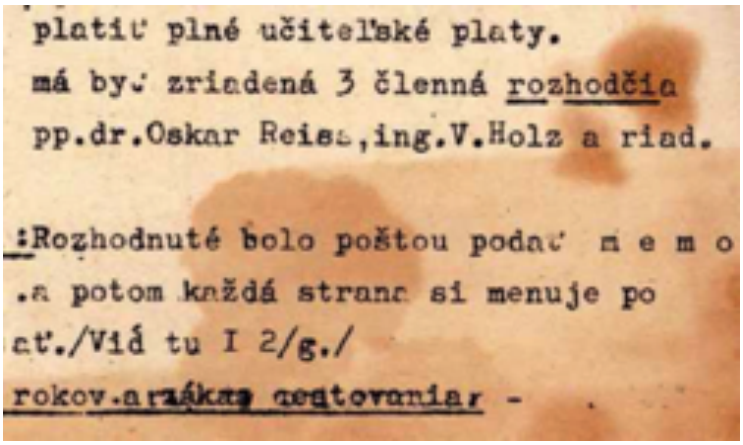
Potential algorithms you may wish to reference or use:

- http://www.cs.rutgers.edu/~elgammal/pub/skin.pdf
- Vezhnevets, V., Sazonoz, V., Andreeva, A., “A survey on pixel-based skin color detection techniques”, in Proc. Graphicon, pp.85-92 (2003)
- Xiang, F-H., Suandi, S.A., “Fusion os multi-color space for human skin region segmentation”, Int. Journal of Information and Electronics Engineering, 3(2), pp.172-174 (2013)

Hints:
There are lots of references in the paper cited above (Vezhnevets et al.). Don’t choose uber complex algorithms.

5. Text Binarization

The first step in OCR (optical character recognition) is binarizing the text through segmentation, usually some form of thresholding. The problem with older texts is that the text often suffers from various forms of degradation: foxing (the rusty coloured blotches), blurriness due to scanning artifacts etc. Thresholding can be performed using a global threshold, or more localized thresholding algorithms.



Task: Compare a series of three different localized thresholding algorithms over the various test text images provided. Compare them against a global thresholding algorithms such as Otsu. Examples of localized algorithms are included in the references below, but for 100% mark, you can find at least one example not on this list and implement it. Test the original images - some will have to be converted to grayscale - and then apply some form of post-processing to the more difficult images, and test them again.

Potential algorithms you may wish to reference or use:

- Niblack, W., An Introduction to Digital Image Processing, p.115-116, Prentice Hall, NJ. (1986)
- Sauvola, J., Pietaksinen, M., “Adaptive document image binarization”, Pattern Recognition, Vol.33 p. 225-236. (2000)
- Zhang, Z., Tan, C. L., “Restoration of images scanned from thick bound documents”, in Proc. of the 2001 International Conference on Image Processing, Vol.1, p.1074-1077. (2001)
- Eikvil, L., Taxt, T., and Moen, K., “A fast adaptive method for binarization of document images”, In: Proc. ICDAR, France, pp. 435-443 (1991)

HINTS:
You may like to try using background subtraction, or other means of enhancing the quality of the image as a precursor to thresholding as a means of improving the thresholding results. Try the algorithms with and without pre-enhancement.

6. Fruit Recognition

Harvesting fruit is a time-consuming and expensive task. Due to variable ripening times, it can also result in a lot of wastage. For example, tomatoes do not ripe simultaneously. One of the main challenges in the design of automated harvesting systems is their ability to recognize and localize of ripe fruit on the plant.

Task: Design and implement an algorithm to localize, and count the number of ripe fruit on a plant. Fruit could be tomatoes, or citrus fruit such as oranges.

Potential algorithms you may wish to reference or use:

- Zhao, J., Tow, J., Katupitiya, J., “On-tree fruit recognition using texture properties and fruit data”, IEEE/RSJ International Conference on Intelligent Robots and Systems, pp.3993-3998 (2005).
- Arefi, A,, Motlagh, A.M., Mollazade, K., Teimourlou, R.F., “Recognition and localization of ripen tomato based on machine vision”, Australian Journal of Crop Science, 5(10), pp. 1144-1149 (2011) http://www.cropj.com/arefi_5_10_2011_1144_1149.pdf
- Jimenez, A.R., Jain, A.K., Ceres, R., Pons, J.L., “Automatic fruit recognition: a survey and new results using range/attenuation images,” Pattern Recognition, 32, pp.1719–1736 (1999).
- Choi, K., Lee, G., Han, Y.J., Bunn, J.M., “Tomato maturity evaluation using color image analysis”, Transactions of the ASAE, (1995) http://ucanr.edu/datastoreFiles/608-1043.pdf
- Mery, D., Pedreschi, F., “Segmentation of colour food images using a robust algorithm”, Journal of Food Engineering, 66, pp.353-360 (2005)

Hints

- Use some form of different colour space.
- There are plenty of images to test on the net.