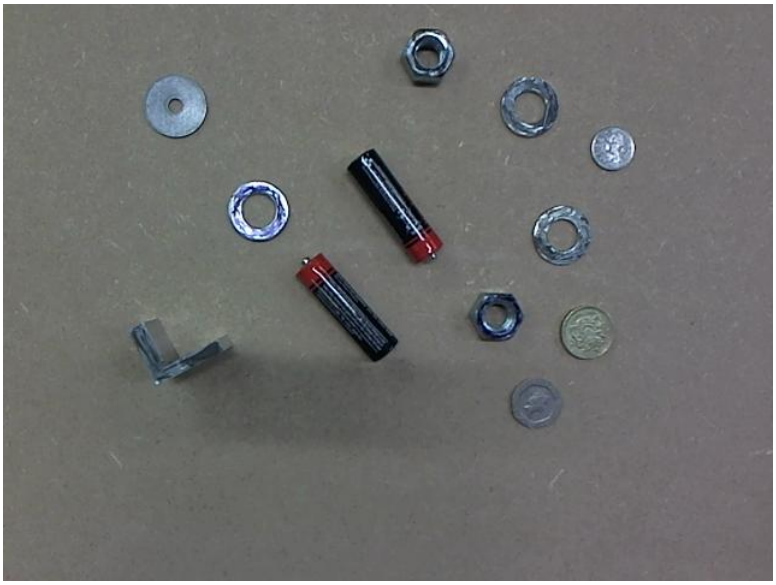
Introduction to Vision and Robotics
Assessed Practical 1: Coin Counter

October 24, 2016

1 Introduction

In this practical you will develop a Matlab program that can function as a coin counter by detecting coins and various other small objects from image. It will detect objects present in the image, segment these objects from the background and classify the coins and distractor objects from a variety of test images.

You will be provided with 14 test images, all captured from a downward facing camera viewing a scene with the objects and a static background.



You will work in pairs and submit a joint report. **This report must be accompanied by a short explanation of how the work was shared and how you think the (joint) mark should be distributed. 50:50 is OK if you shared the assignment fairly.**

You will find the group assignments here:
http://www.inf.ed.ac.uk/teaching/courses/ivr/assignment1/2016-17/ivr-groups.pdf

You should be able to access the IVR lab anytime with your swipe card (avoiding very busy labs for other classes). Webcams are generally in the lab, but you do not need to collect data for this assignment.

2 The Task

The overall goal is to develop an algorithm that reads a set of images and classifies the objects present and counts the value of the coins. There is a fixed set of 10 object classes:

- one and two pound pieces
- 50, 20 and 5 pence pieces
- washer with a small hole (75p)
- washer with a large hole (25p)
- angle bracket (2p)
- AAA battery (no value)
- nut (no value)

Some new coins have been invented and their value is listed above. You should identify, but ignore, the objects listed as having no value.

2.1 Approach

Detection: In lecture, there was an example of background subtraction, but how can one get a background image? If one had the opportunity to take a picture before any objects were in the scene, then could be a solution. However, the data used here has foreground objects in all the images. If the camera and illumination is largely stationary, as it is here, then you can synthesise a background image by median filtering the values observed over time at each pixel. That is, if $\{v_1, v_2, \ldots v_n\}$ are the values observed of the red colour channel at pixel (r, c) at different times, then the value $median(\{v_1, v_2, \ldots v_n\})$ is an estimate of the background value of that colour channel at that pixel. Why? Because we assume that a foreground object will be observed in a few frames at the given pixel, but that most of the observations will be of the background. Repeat this at every pixel and colour channel to build up a background image.

How can you cope with the difference in illumination in different places in the image? Use normalised RGB instead of raw RGB values. Image values will be reasonably independent of the illumination. It also means that white, grey, and black pixels end up with about the same value.

When you threshold the difference images, there might be many small regions detected due to image noise. How can they be removed? Look at the `imerode()` command.

Show examples of correct detections, missed detections and invalid detections. Report the number of correct, missed and incorrect detections in your report. Explain the cause(s) of missed/invalid detections.

Classification: To identify the objects based on the features: The lectures presented a Bayes classifier for simple shapes. You might use a combination of colour and invariant moments to classify each shape. Or you could use a support vector machine if you have learned about these. Be sure to have enough training data to train your classifiers. Then you could use the individual classifications to vote for an object.

You should split your data into 2 sets, half for training and the other half for testing (where the test half was not used when training the classifier).

You should produce algorithms that are insensitive to small changes in illumination and camera position. You should test these aspects of your program.

Challenging Cases: By the nature of this task, certain classes are more difficult to distinguish (e.g. 20p piece from a nut). You should explore how well your approach works using the test data. An approach which cautiously refuses to make a classification for difficult instances would be an interesting approach while an algorithm that eagerly makes mistakes is of less value. Remember that 75% of the marks are for the report - so experimentation is highly valued and should be reported.

What to do if an object is found on the edge of an image or a number of objects lie on top of one another? Again, refusing to make a decision in such a situation is a demonstration of a more advanced method than incorrectly making a classification. But remember to explain how you made such a decision.

You should produce algorithms that are insensitive to changes in illumination and background.

3 Files You Need

The image files needed for development can be downloaded from the IVR webpage:

http://www.inf.ed.ac.uk/teaching/courses/ivr/assignment1/2016-17/ivr-vision-student-data.tar

The file contains 10 images.

Update 19 October 2016: Originally we provided a set of harder images (with objects overlaying one another. We will not test the harder set. You can ignore these.

4 Image Capture Details

If you wish to capture additional test images, feel free to do so. With a typical webcam the following commands will work with a DICE machine:

Use `mplayer tv:// -tv driver=v4l2:width=640:height=480:device=/dev/video0 -frames 5 -vo jpeg` to capture 5 frames of a new sequence. (Note: that the 'l' in v4l2 is video-for-linux - not the number 1.)

Note: The first few frames captured are usually have different exposure than the others.

5 Writing the report

The report should be a concise description of what you did, why, and what happened. The entire report should be no more than 4000 words (excluding appendices). It should contain the following sections:

1. Introduction: an overview of the main ideas used in your approach.
2. Methods: Explain the vision techniques that you used. Then give a functional outline of how these ideas were implemented and the structure of your code. Explain how each part of it is meant to work. Where suitable, justify your decisions, e.g. why you used one method rather than another, what you tried that didnt work as expected, etc.
3. Results: You should provide some actual data of how well your algorithm performs, as described previously. Show an example of your results for each stage of the detection. Well documented failure will get more marks than unsupported claims of success (well-documented success would be even better!).
4. Discussion: Assess the success of your program with regard to the reported results, and explain any limitations, problems or improvements you would make.
5. Code: the new Matlab code that you developed for this assignment should be added as an appendix. Do not include code that you downloaded from the course web pages. Any other code that you downloaded should be recorded in the report, but does not need to be included in the appendix.

The report should not be written as a set of chronological steps (we did this, then that then this), but instead you should explain why you chose a certain approach over another. For example is would be interesting to present two approaches for the same subtask where one fails and the other is successful and explain why.

Your final mark will be based on how well you explain your approach to the task and evaluate the capability of your Matlab program as well as its performance.

6 Live Demonstration

On Friday October 28, between 09:00-17:00, you will have to demonstrate your program working on new images that are similar to the dataset supplied here. Changes to the background and lighting should be expected, but not different object classes.

Your group has been allocated to a demonstration time. The timetable of allocated slots is available in the same document mentioned above.

7 Submission

Your submission should be a single PDF file and should be submitted electronically by 4pm Thursday October 27. The command to use for on-line submission is:

```
submit ivr 1 FILENAME
```

where FILENAME is the name of your PDF file.

This assignment is estimated to take 10-15 hours work. You must do this assignment in pairs and assign credit in your final report. The assignment will be marked as follows:

Issue	Percentage
Program Design, including comments	25%
Report Clarity	25%
Experimental Results (in Report)	25%
Live Demonstration Results	25%

The live demonstration results will be marked based on a combination of successful detection and segmentation (5%), correct classification and coin counting (10%), appropriately dealing with the more challenging images (5%) and processing speed (5%).

For the demonstration, consider what you will show to the marker. Make sure you can enable plotting of the intermediate stages of each algorithm and can pause the processing at each stage. Dont just run your code and present the final result. We want to understand the internals of what it is doing.

8 Good Scholarly Practice

This assignment is expected to be in your own words and code. Short quotations (with proper, explicit attribution) are allowed, but the bulk of the submission should be your own work. Use proper citation style for all citations, whether traditional paper resources or web-based materials.

Please remember the University requirement as regards all assessed work for credit. Details about this can be found at:

http://www.ed.ac.uk/academic-services/students/undergraduate/discipline/academic-misconduct

http://www.ed.ac.uk/academic-services/students/postgraduate-taught/discipline/academic-misconduct

http://web.inf.ed.ac.uk/infweb/admin/policies/academic-misconduct

Furthermore, you are required to take reasonable measures to protect your assessed work from unauthorised access. For example, if you put any such work on a public repository then you must set access permissions appropriately (by permitting access only to yourself, or your group in the case of group practicals).