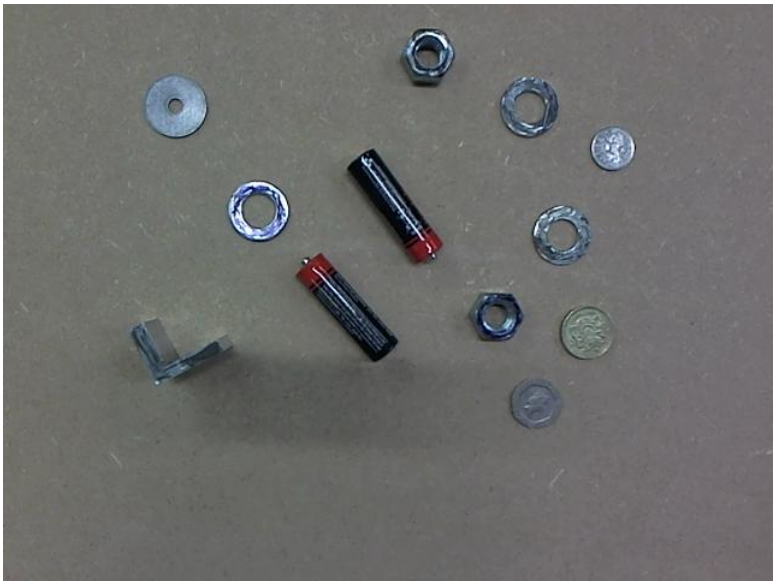
Introduction to Vision and Robotics
Assessed Practical 1: Coin Counter

October 1, 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 any objects present in the image, segment these objects and classifies the coins and other objects from a variety of test images.

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



You will work in pairs and must 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 Some Theory

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, and most of the observations will be of the background. Repeat this at every pixel and colour channel to build up a background image. This can be slow if you use all of the time samples, so it is better to use a subset, e.g. one out of every 5-10 images because we assume that the foreground object is moving.

How can you decide if the robot seen in one frame is the same as that seen in the next frame? There are two useful sources of information: 1) Position - assuming that the frame rate is high and the object velocity is low, then the target will not move much between images. 2) Colour - the average RGB value of the pixels of a target will not change much between consecutive images.

How can you cope with the difference in illumination in different places in the image? Use normalised RGB instead of raw RGB values.

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 `bwmorph(fore,erode,1)` command.

How might you determine the orientation of the robot? Look at properties of the detected and thresholded region. Investigate moments, centre-of-mass, template matching, etc.

How might you find the robot regions? Look at the `bwlabel(foremm,4)` and `regionprops(labeled,['basic'])` commands. What to do if a robot is not detected in a given frame? Use an algorithm that allows the tracking to skip over up to N frames (e.g. $N = 3$).

3 The Task

The overall goal is to develop an algorithm that takes the sequence of images and tracks each of the three robots.

Two sets of images are supplied. There are also sets of the robots and webcams available in the lab so you can capture additional image sets.

What you have to do is:

1. Develop an algorithm that detects the robots in each image. In your report, present the criteria that defines a detection. Show examples of correct detections, missed detections and invalid detections (i.e. detections reported at locations where no robot is present). Report the number of correct, missed and incorrect detections. Explain the cause(s) of missed/invalid detections.

2. Develop an algorithm that correctly links together detections of the robot in consecutive frames. Show the path formed by linking together the centres of the detected robots drawn on top of the estimated background image. In your report, present the criteria that defines when two detections should be linked into a track. Show examples of correct tracks, broken tracks and incorrect tracks (i.e. tracks that join detections of different robots. Report the number of correct, broken and incorrect tracks. Explain the cause(s) of invalid/broken tracks.

3. Develop an algorithm that correctly identifies the direction that the robot is facing. There is an arrowhead on each robot that indicates the forward direc- tion. Identify that direction and draw a line from the centre-of-mass of the robot in that direction. Do this for all 3 robots. Plot this line on top of each image frame, pausing using the pause(1) command to allow one second for the viewer to assess if the direction line is drawn correctly. Show

examples of correct and incorrect directions in your report. Report the number of correct and incorrect directions over each sequence (requires manual assessment). Explain the cause(s) of invalid tracks.

You should produce algorithms that are insensitive to changes in illumination and camera position (but assuming that both remain fixed during the capture of a sequence).

4 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 14 images - split into an easier and harder set.

5 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 other.

6 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, from repeated trials (with the camera or robots in different positions) on 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 cronological steps (we did this, then that then this) but 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 the performance.

7 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 different background or 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 on the IVR course website in the same document mentioned above.

8 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 cw1 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%), appropiately dealing with the more challenging images (5%) and processing speed (5%).

For this 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.

9 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).