

Introduction to Vision and Robotics

Second Assessed Practical: Recognising Visual Input

November 4, 2009

1 Introduction

This practical develops a Matlab program that works in both the real and the Webot worlds to take a camera images and:

1. Identifies coloured location markers.
2. Navigates a robot to a series of the markers using visual servoing.
3. Accurately parks the robot on top of the final target circle.

You will work in pairs and must submit a joint report *but* 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. **You can choose the partner yourself, but you must choose a different partner from the person that you worked with in the first practical assignment.**

You should be able to normally access the IVR lab (AT 3.01) anytime with your swipe card. You need to get and return the Kheperas from the ITO between 9-5, Monday to Friday. The cameras will generally be available in the lab - do not take them elsewhere!

You can use Matlab and Webots at any time on other DICE machines, but there may be license number limits on availability so *never* stay connected to the programs when you are not using them.

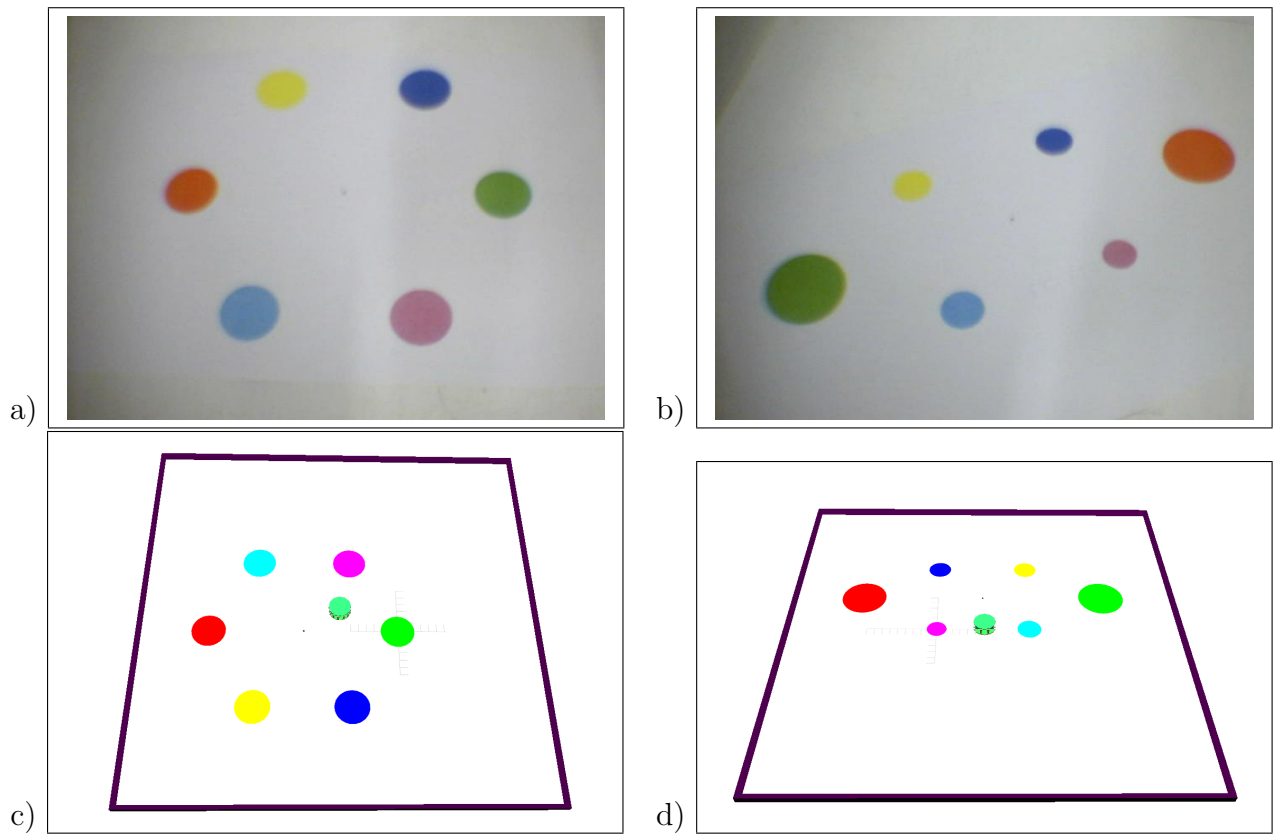


Figure 1: (a) Real scene 1, (b) Real scene 2, (c) Webot scene 1 with khepera, (d) Webot scene 2 with khepera

2 The Task

The practical task involves target recognition and visual servoing and will require use of both the Webots simulator and real camera images with the Khepera robot.

The overall goal is to develop an algorithm that takes images of a scene and navigates a robot to a sequence of locations. The task will involve both webot scenes and real scenes.

Complicating the matter will be the fact that the images may have much perspective distortion and you will not know the exact camera position from where the images were taken.

Figures 1a and 1b show 2 typical real images of the scene layout. You can see 6 coloured circles in a hexagonal arrangement. Figures 1c and 1d show 2 typical webot images of the scene layout. In the real and webot scene 1, the robot must visit the coloured dots in the order: **red-yellow-purple-green**. In the real and webot scene 2, the robot must visit the coloured dots in the order: **blue-red-green-purple**. Notice the small black dot in the centre of the hexagon of coloured circles. This is the starting location for your webot/Khepera robot. A valid visit has the robot completely cover one of the target circles. The robot must end up perfectly centred over the final target.

What you have to do is:

1. Given a random (but not unreasonable) placement of the webcam, capture an image of the scene.
2. Identify the robot (using background comparison) and the coloured circles (thresholding and properties). Use a Bayesian classifier to determine the colour of the circle.
3. Using visual servoing, navigate the robot to visit the given sequence of circles.
4. Servo the robot so that the final position is directly over the last circle.

You must analyse these 2 real and 2 webot scenes for the practical, but you can also make your own images. In webots, move the camera to a new position and take a new image. In the lab, move the paper sheets around and take new webcam images. Paper sheets with the scenes above will be in the lab.

Given an image, use colour and lightness to detect all of the circles and robot. Before you capture a real image, you will probably want to also capture an image of the background white box without the scene sheet or obstacles. You can mask away a small amount of the background at the edges of the image if there are highlights or shadows that give your algorithm problems.

3 Files You Need

You will need these files:

<code>world10910.wbt</code>	scene 1
<code>realscene10910.jpg</code>	scene 1 floor image
<code>world20910.wbt</code>	scene 2
<code>realscene20910.jpg</code>	scene 2 floor image
<code>WEBOT_software_2010.tar.gz</code>	extended webot controller software for IVR

Download these files from the course webpage. The `WEBOT_software_2010.tar.gz` file needs to be moved to `<YOUR WEBOT FOLDER>` and then `gunzip WEBOT_software_2010.tar.gz` and then `tar -xvf WEBOT_software_2010.tar`. Put the first 4 files into your `<YOUR WEBOT FOLDER>/local.webots/worlds/` folder.

4 Image Capture Details

Robot ‘arenas’ will be set up in the IVR lab containing high white walls and the two different patterns printed on paper. Place your webcam so that it can see into the arena.

To grab an image in Webots, pull down the **File** -> **Make Screenshot...** option. This will create a PNG image that can be loaded into Matlab. Use the left/middle/right mouse buttons to move the camera into different views of the scene. You’ll want multiple views for training your classifiers before using the robot in the scene. You should move the real or webot camera viewpoint so you get a variety of training data for the scenes.

In a day or 2, we will have an alternative method where you can also capture a view of the scene from the matlab to webots interface. At the moment, you have to capture the image manually using the pull-down **File** -> **Take Screenshot**.

On some machines, the captured screenshot has a black box where the pulldown menu was. If this is the case: 1) drag the webot window corner to make the panel larger, 2) use the right mouse button to drag the scene so it is not covered by the pulldown panel, 3) write your program to not use the leftmost portion of the webot images.

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 2000 words (excluding appendices). It

should contain the following sections:

1. Introduction: an overview of the main ideas used in your approach.
2. Methods: Give a functional outline of your code. Include your full, commented, code in an Appendix. Do not print any code supplied from the IVR web pages. 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 didn't work as expected, etc.
3. Results: You should provide some actual data, from repeated trials (with the camera or sheets in different positions) on how well your algorithm can park the robot. Show at least your results for the 2 supplied real and 2 webot scenes. 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, and explain any limitations, problems or improvements you would make.

Your final mark will be based on how well you explain your approach to the task and evaluate the capability of your Matlab program. If your navigation algorithm isn't reliable, you won't fail if you can provide a sensible explanation of what you attempted and the limitations and problems in your report.

6 Live Demonstration

On Friday Nov 20, 2-6pm, you will have to demonstrate your program working on one Webot and one real scene. Previously unseen floor patterns using the same symbols but in slightly different placements will be used.

Submission

Your submission should be a single PDF file and should be submitted electronically by *10am Friday Nov 20*. The command to use for on-line submission is:

```
submit ai3 ivr cw2 FILENAME
```

where FILENAME is the name of your file.

This assignment is estimated to take 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	20%
Report Clarity	20%
Experimental Results (in Report)	20%
Demonstration Results (Webots)	20%
Demonstration Results (Real)	20%

You will have to demonstrate your program working on a new webot scene and a new real scene. For each of these, 10% of the mark comes from correctly navigating the given colour sequence. The other 10% comes from accurately servoing the robot over the final target location, with a 1% reduction for each 2mm away from the correct target location.

7 Plagiarism

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. Before submitting please acknowledge any additional sources of code that you use and ensure that your submission follows the school policy on plagiarism: <http://www.inf.ed.ac.uk/teaching/plagiarism.html>.