Introduction to Vision and Robotics: Assignment 1

Chris Swetenham: Student Num, Daniel Mankowitz: S1128165

3/11/2011

1 Introduction

2 Methods

2.1 Linking Algorithm

The linking algorithm is used to link together detections of a single robot in consecutive frames. Since the images provided in the dataset are RGB, the linking algorithm has been implemented to run on each of the three colour channels respectively.

The algorithm is shown in *Figure 1.* Initially, the image on the k^{th} iteration will have been split into three colour channels of red, green and blue respectively. Each colour channel is represented as a separate image. The robots in each colour channel will have been identified and their optimum bounding boxes need to be calculated. These images are fed into the linking algorithm.

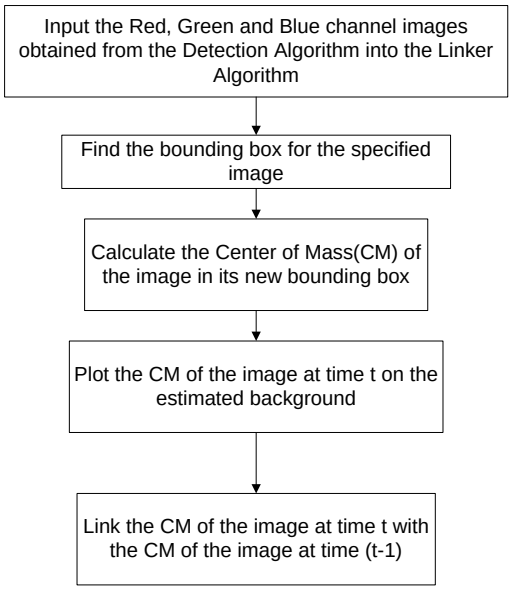

Figure 1: The algorithm developed to link detections of the robot in consecutive frames

Once the images containing the robots have been received, a bounding box for each robot is calculated as well as the box's corresponding centroid. This is achieved in code using the 'calcBoundingBox' function. This function has been developed according to *Figure 2.* Each image is labelled using the 'mybwlabel' function. This will identify all of the objects in the image. The object with the largest area (I.e. the robot) will then be surrounded by its corresponding bounding box. The centroid of this bounding box is then calculated. This is performed on each of the three images corresponding to their repective colour channels.

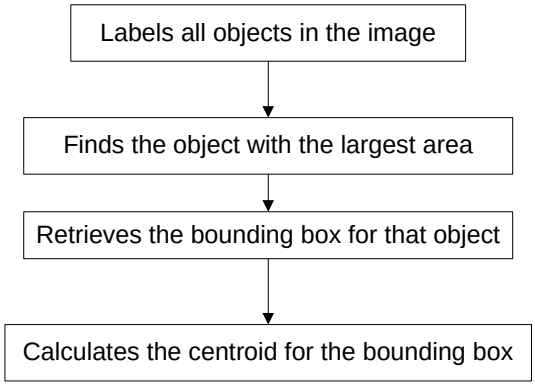

Figure 2: The bounding box calculation for each robot

Once a bounding box for the robot has been determined, the center of mass of the robot needs to be calculated. This is achieved by utilising the 'calcBoundingBoxCM' function.

This algorithm makes use of two important equations shown in *Equation 1* and *Equation 2* respectively.

$$A = \Sigma_r \Sigma_c P_{rc} \tag{1}$$

$$(\widehat{r}, \widehat{c}) = (\frac{1}{A}\Sigma_r \Sigma_c r P_{rc}, \frac{1}{A}\Sigma_r \Sigma_c c P_{rc}) \tag{2}$$

2.2 Background Estimation

Estimation of the background image required the implementation of a variety of different image processing techniques. This is because the datasets provided presented a number of challenges. One such challenge was that of removing robots that were stationary for a significant portion of a frame sequence. This prevented the simple implementation of a median filter to calculate the background image. This is due to the fact that objects that are stationary for a large subset of the frames will be incorporated into the median and subsequently into the background image.

Therefore, in order to solve this problem, an algorithm has been developed to calculate the background image, regardless of stationary robots being in large subsets of the frame sequence.

The background estimation algorithm consists of three main functional blocks. These blocks are *Channel Processing*, *Region Erasing and Filling* and *Median Filtering* respectively. The *Channel Processing* block receives an image frame and processes the image. This includes blurring, normalising, separating the image into its three channels, subtracting the blue channel from the green channel and renormalising the image. This creates an image that is ready for *Region Erasing and Filling*. In order to implement this functional block, the image channels for the current frame are input into a function called *eraseRegion*. A code snippet of the function is shown below.

```
1  function [Out] = eraseRegion(Img, C, Size)
2      Out = Img;
3      %Find the pixels on the vertices of the bounding box
4      %surrounding the robot. top-t, bottom-b, left-l, right-r
5      t = max(1, C(1) - Size/2);
6      b = min(size(Img, 1), C(1) + Size/2);
7      l = max(1, C(2) - Size/2);
8      r = min(size(Img, 2), C(2) + Size/2);
```

```
    %Find the average of the four pixels at the vertices of the
    %bounding box
    Avg = Img(t, l, :) + Img(t, r, :) + Img(b, l, :) + Img(b, r, :);
    Avg = Avg / 4;
    %Replace the image segment (I.e. the robot) with the average
    %of the four pixels
    for y = t:b
        for x = l:r
            Out(y, x, :) = Avg;
        end
    end
```

This function calculates the average value of the pixels found at the corner vertices of the bounding boxes surrounding each robot as shown in *Figure 3*. The corner pixels are averaged across all three colour channels and are stored in the variable *Avg*. All of the pixels in the region containing the robot are then filled with this new average value to produce the image shown in *Figure 3*. This is applied to each of the robots in the image.

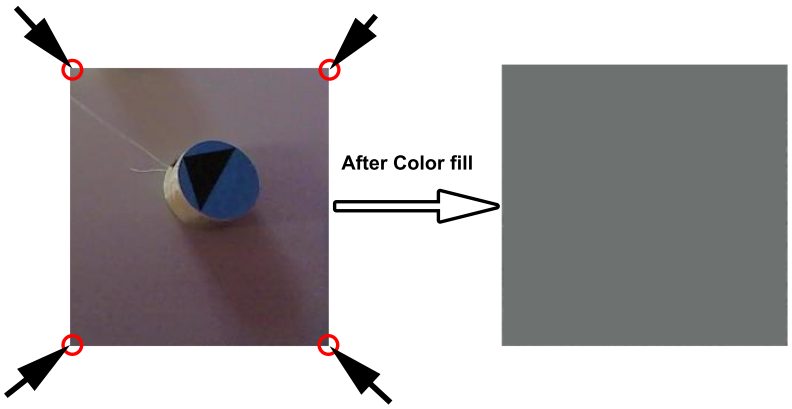

Figure 3: The corner pixel intensities are averaged and are then used to fill the region of the image containing the robot

The *Region Erasing and Filling* functional block is implemented on the three image frames that are used for background estimation. The reason three image frames have been chosen is because this is the minimum number of frames required to effectively use a median filter which is utilised in the *Median Filtering* functional block. In addition to this, using a small number of frames to compute the median minimises the algorithm's processing time. The median of the three images is subsequently computed to produce an estimation of the background image.

3 Results

3.1 Robot Detection

An example of an incorrect detection of a robot due to the robot leaving the image frame.

An example of an incorrect detection of a robot due to a region of highly saturated pixels that have a similar colour to that of the robot.

3.2 Robot Orientation

An example of incorrect and missed orientation in an image frame.

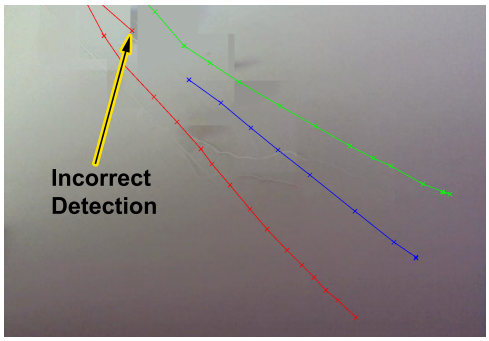

Figure 4: An incorrect detection of a robot

Figure 5: The incorrect detection of the red robot as shown in the red channel thresholded image

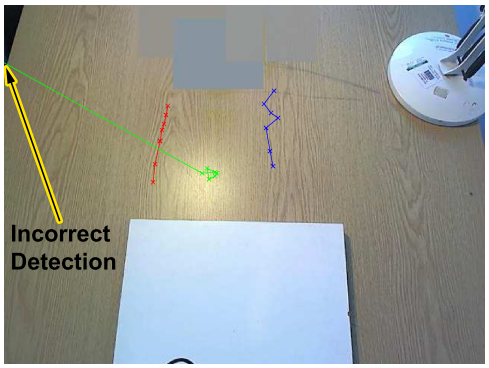

Figure 6: An incorrect detection of the green robot

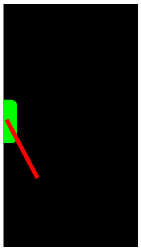

Figure 7: The incorrect detection of the green robot due to a small region of highly saturated pixels

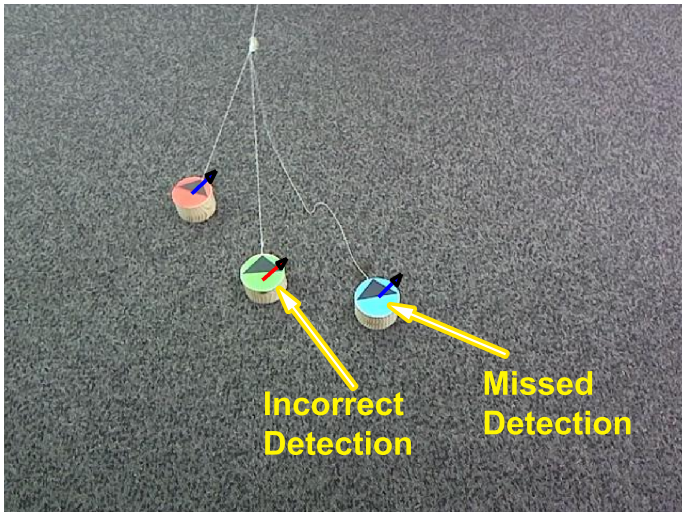

Figure 8: Examples of incorrect and missed detections of invalid arrow orientations

3.3 Background Images

An example of background images produced using the Background Estimation Algorithm detailed in *Section 3.3*.

Figure 9: The estimated background image of the first dataset provided for image processing

Figure 10: The estimated background image of the second dataset provided for image processing

Figure 11: The estimated background image of a new test dataset

Figure 12: The estimated background image of a new test dataset

4 Discussion

5 Code

6 Conclusion

APPENDIX A

A.1 Control Program