AINT308 - Machine Vision and Behavioural Computing
Coursework 1 Report

Student No. 10613591
School of Engineering,
Computing and Mathematics
University of Plymouth
Plymouth, Devon

***Abstract*—Machine Vision is field of study who's applications are becoming rapidly more prevalent amongst contemporary technology, with advancements having large implications in a wide variety of fields. This report details the use of a popular open-source machine vision library, OpenCV, in three different applications: Identifying the most common colour in an image, tracking a moving object within a video, and matching sub-components of an image to a larger image.**

Keywords:

Machine Vision, OpenCV, Object Tracking

I. Task 1 - Colour Sorter

A. Introduction

The first task involves creating a system able to label a dataset of car colours reg, green, or blue.

B. Solution

As part of its suite of features, **OpenCV** includes the ability to retrieve individual pixel's values. The specific values available will vary based on the colour space used. For example, working in the *RGB* colour space enables a pixel's *Red,Green* and *Blue* values to be retrieved. Using this information, it is possible to calculate the mode colour of an image.

```
if ((b>1.5*g) && (b>1.5*r) && (b>125)){
    bluecount++;
    if ((x>cols*(lowlimit)) && (x<cols*(highlimit))){
        bluecount = bluecount + 3;
    }
}
else if ((g>1.5*b) && (g>1.5*r) && (g>125)){
    greencount++;
    if ((x>cols*(lowlimit)) && (x<cols*(highlimit))){
        greencount = greencount +3;
    }
}
else if ((r>1.5*b) && (r>1.5*g) && (r>125)){
    redcount++;
    if ((x>cols*(lowlimit)) && (x<cols*(highlimit))){
        redcount = redcount +3;
    }
```

Fig. 1: Pixel testing code

The code in figure 1 performs the checks needed to assertain a pixel's primary colour. Initial attempts simply checked which of three component values were highest. In practice, this method was inaccurate as it had no way between differentiating between a near-white cell (such as a background cloud) which would have near equal *RGB* values, and one that belonged to the subject (in this case, the car).

To allow the algorithm to differentiate between background elements and the subject, two methods were employed. Firstly, a pixel was only counted if that colour's value was a significant margin higher than the other values. In figure 1 this margin is 50%. Additionally, the pixel's value must be greater than half the colour space's maximum value, in this case 125. This prevents background objects from being counted.

Additionally, extra weighting is applied to pixels within a central rectangle on the screen. This area can be changed by editing the values $highlimit$ and $lowlimit$. By adding this additional weighting to central areas, background areas are further discounted, helping to prevent false positives by valuing true positives more.

C. Limitations

D. Further Improvements

E. Conclusion

II. Task 2 - Object Tracking

A. Introduction

B. Solution

C. Limitations

D. Further Improvements

E. Conclusion

III. Task 3 - Image Mapping

A. Introduction

B. Solution

C. Limitations

D. Further Improvements

E. Conclusion

APPENDIX