Finding Wally

MSc. Dissertation Report

Cian Booth

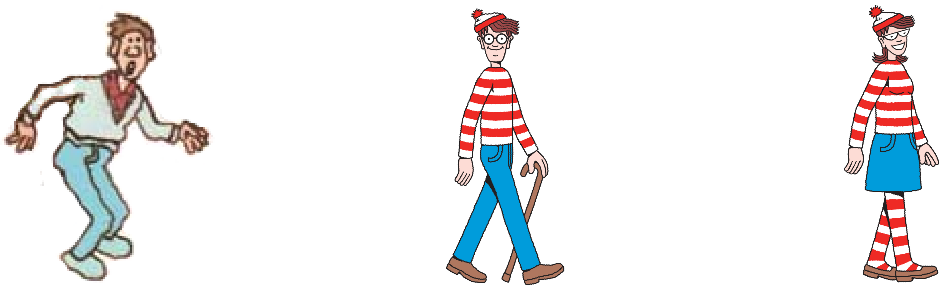
Contents

1 Introduction

Computer vision is a field concerned with using computers to process and analyze images. One of the most important things that human vision is able to do, is object recognition. This means that the brain is able to decipher basic information from the eyes, and deduce the objects that can be seen there. This is a complicated task, and not easily replicated programmatically.

Reliably finding an object within an image is difficult to do at a decent speed. Just as there are several ways to describe an object, there are also several different techniques that can be used to locate an object. By using as many of these techniques as possible, the reliability of the overall object recognition can be increased. Using all available techniques will considerably reduce the speed that objects can be found at. This can be mitigated by computing the techniques in parallel, via a task farm. This means that general tasks (in this case, each method of identifying an object) will be run concurrently. Task farms are especially useful in this case, because it allows the use of libraries that would otherwise be complicated to parallelise.

A good way of testing the task farm method is with Where's Wally puzzles. These are simple cartoons, normally a large image filled with various characters, who wear simply coloured clothing, see Figure 1(a). One of these characters is the eponymous Wally, who is dressed distinctly from the others, see Figure 1(b). Similarly dressed characters exist, Figure 1(c), adding complexity to specifically finding Wally. The cartoon nature of the characters means that shapes are boldly coloured, and often bordered by a black line. As Where's Wally is puzzle, sometimes Wally will be hard to find; he is often obscured, camoflauged or simply small. This provides a non-trivial problem, that is still simple enough to provide solutions for.



(a) A normal person **(b)** Wally **(c)** Wenda

Figure 1: Characters from Where's Wally

1.1 Shape and Colour Analysis

One obvious way of analysing an image with a computer, is to break the image down into shapes and colours. In terms of language, it is easiest to depict an object by describing it's shape and colour, i.e. "the red box" or "the green hand". This is conceptually simple to explain and understand, and thus is more intuitive to program. Most images are saved as raster images (e.g. PNG, BMP, GIF), which is a 2D array of colours that directly map to pixels on a screen. This is opposed to vector images that store the location and colour of geometric primitives (squares, circles, triangles, etc.). Vector images, rather than raster images, are directly easier to perform shape and colour analysis on. However, input devices such as webcams and scanners almost exclusively produce images in raster formats.

Regardless of format, methods of colour analysis are simple to implement programatically. This is because colour is instrinsic to all methods of storing the data of an image. Shape analysis is less simple for raster images. Shape boundaries must be found, which is normally done through edge detection. These boundaries are then analysed to find points of intersection, and the curvature between them. The curves, in turn, must then be examined to find what shapes exist.

These methods allow for a descriptional, heuristic method for locating characters. No previous image of Wally is required, only a description, such as "red and white stripes" or "black glasses". This allows users to extend the solution past Wally, and towards other characters, who do not have to be seen prior. This form of analysis can lead to false positivies, and should be combined with other results.

An example of the usefulness of this, beyond Where's Wally, could be found in augmented reality technology, such as Google Glass. A common problem people experience is losing their keys. Users with access to AR devices could use colour and shape analysis to enhance their searching (i.e. if they are visible, but in a cluttered area). As keys generally have a few well defined shapes and colour schemes, the device would not have to store what the keys look like in advance. Assuming that parallelism is available, this could potentially be done faster than the human eye can search, helping the user significantly.

1.2 Feature Analysis

Some of the most reliable computer vision libraries (such as SIFT [1]), were developed while considering the neuroscience of human vision. Tanaka[2] and Perrett and Oram[3] found that human object recognition identifies objects with features that are invariant to brightness, scale and position. These results have been used as inspiration for feature analysis. This finds 'feature's, regions of an image which are scale, rotation and illumination invariant. These features are most immediately useful when compared with the features of another image. For example, the features from an image of just Wally can be used to locate Wally in a normal puzzle image.

This method generally requires an existing image to find Wally, which leaves However, this method is very reliable, and, as long as Wally is not obscured, will likely locate him. Normally, Wally is obscured, so this method should be combined with other techniques.

Feature analysis benefits from task farming when there is an image needs to be searched for a large number of sub-images. A beyond Wally example would be in detecting employees going into work on a flexitime basis. This would use photos of each employee combined with CCTV to note the time that workers enter and leave their workplace. Users would not need any form of ID other than their own faces, and this can be combined with existing security systems.

2 Literature Review

2.1 Feature Recognition

Detecting features within an image is an important technique that should be used in object recognition. One of the most robust algorithms available for this is SIFT (Scale-Invariant Feature Transform), developed by David Lowe[1]. This algorithm uses various techniques to find scale, rotationally and translationally invariant keys, which are also partially invariant under affine transforms and changes in illumination. The keys contain feature vectors, which describe the area around them. This algorithm is very reliable at finding a given sub-image within a larger image. However, some of the techniques required to produce reliable results require unexpected amounts of memory. To create a scale-space version of the image, four versions of the image must be produced, one of which is scaled to be twice the size of the original. This is not a problem when analysing a single image, or analysing multiple images. If the algorithm is used for analysing multiple images concurrently, memory constraints would limit it's efficiency. This can be seen with Zhang's parallel implementation of SIFT [4], where the scale-space creation is one of the areas the program spends most time on. This paper shows reasonable parallel efficiency, but only has results up to 32 cores. The paper also only solves 5 images at a time, which gives a 7 times speed-up over an optimised version of SIFT. This number of images is not large enough to show the true limits of shared memory parallelism.

SURF (Speeded-Up Robust Features), developed by Bay et. al.[5], offers a similarly robust solution, but with greater efficiency. This algorithm replaces Laplacian of Gaussian filters used in SIFT with box filters, which calculate in constant time, once an integral image has been produced. SURF This algorithm has greater potential for parallelism; calculating versions of the image for the scale space are independent of the previous level and can be done in parallel.

References

[1] D. G. Lowe, “Object recognition from local scale-invariant features,” in *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, vol. 2, pp. 1150–1157, Ieee, 1999.

[2] K. Tanaka, “Mechanisms of visual object recognition: monkey and human studies,” *Current opinion in neurobiology*, vol. 7, no. 4, pp. 523–529, 1997.

[3] D. I. Perrett and M. W. Oram, “Visual recognition based on temporal cortex cells: Viewer-centred processing of pattern configuration,” *Zeitschrift fur Naturforschung C-Journal of Biosciences*, vol. 53, no. 7, pp. 518–541, 1998.

[4] Q. Zhang, Y. Chen, Y. Zhang, and Y. Xu, “Sift implementation and optimization for multi-core systems,”

[5] H. Bay, T. Tuytelaars, and L. Van Gool, “Surf: Speeded up robust features,” in *Computer Vision–ECCV 2006*, pp. 404–417, Springer, 2006.