Real Time Unnatural Object Detection for an In-Flight UAV

Adam Hazeldene [1], Andrew Ross Price [2],

[1] *Aerobotics Group, Dept Electrical and Computer Systems Engineering, Monash University, Melbourne ,Victoria, 3800 Australia*
[2] *Aerobotics Group, Dept Electrical and Computer Systems Engineering, Monash University, Melbourne ,Victoria, 3800 Australia. Andrew.price@eng.monash.edu.au*

Summary: The determination of natural and unnatural objects in real time from a small lightweight Unmanned Aerial Vehicle (UAV) is the basis of many mission scenarios including detection of ships at sea, vehicles in trees, buildings and other man made objects in a natural environment. The problem with many image processing operations is the time and computing power required. In real-time situations, and on an airborne system with a total weight of less than 5kg, both time and computing power are dramatically limited. With these limitations in mind, it was decided to investigate how well natural and unnatural objects could be distinguished using fundamental concepts and features. While measures such as color are of some use, for example bright red is not a common natural color but fairly prevalent in vehicles such as fire trucks (and Ferraris), they are only of limited use when natural colored objects are actually unnatural objects. This investigation explores detecting unnatural objects based on features **other** than color. Only simple techniques which required minimal computing were permitted. The results presented in this investigation were tested on poor quality, grainy, distorted footage with low color separation captured from a UAV in flight. Vehicles, buildings, roads and markers were separated from trees, scrub and natural bushland using Commercial off-the-shelf low resolution capture hardware and a typical PC. This research was completed as part of an undergraduate thesis project at Monash University. Figuree 1 describes the problem space.

Keywords: UAV, Unnatural Object Detection. Onboard Image Processing.

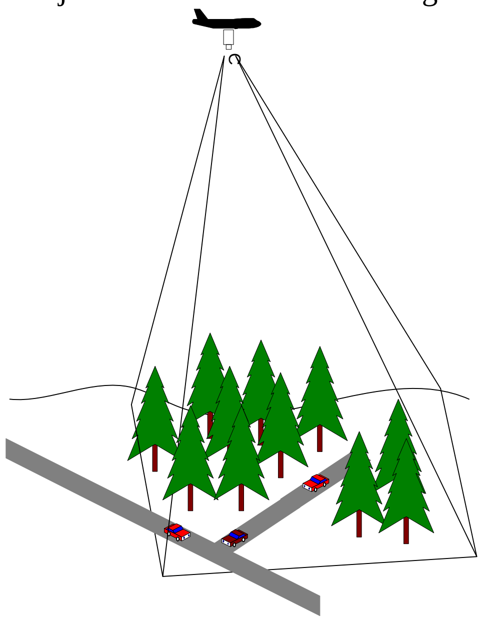


Figure 1. Camera mounted on UAV

Introduction

While many elegant techniques exist for object determination, in a real-time, low power, low cost airborne image processing system, it is desirable to do as many tasks with as few resources as possible. A typical problem involves determining vehicles, roads and buildings from the surrounding bushland. Such information is useful in planning search and rescue missions, bushfire support and surveillance. The UAV's under investigation for this purpose have an all-up flying weight of less than 5kg. The payload is only part of this weight hence computing power, including camera, is limited. This investigation seeks to use fundamental techniques to try and determine the presence of unnatural objects in typical Australian semi-rural bushland. A technique was devised based on the premise that even though natural and unnatural objects may be of the same or similar coloration, unnatural objects are more likely to have a higher prevalence of straight lines, and a higher prevalence of parallel lines. These measures are computationally inexpensive and independent of lighting and color. Experiments were conducted using low resolution video footage taken from a UAV in flight. The system has been successfully tested in real time. Results are included.

System Design

System Components

The intended outcome was to produce a low cost, fairly reliable system capable of being used in a large range of applications.
The components that were used in the design and testing of the system were low cost commercial products that are readily available. A Go Video CCS-C81 CCD Camera along with a LifeView FlyTV 3000 PCI Video Capture Card were used to capture data for analysis via an interfacing PC running Linux with the Video for Linux API. This allowed the capture of PAL standard data, formatted for 640x480 capture at a frame rate of 25 frames per second.
Figure 2 shows the breakdown of system components and the flow of the overall system from one component to the next in both hardware and software. Once the segmentation algorithms outlined in Figure 3 had analysed the capture data, a method of display was chosen, using both the GLUT/OpenGL graphics libraries.
A conjunction of a number of computationally inexpensive, simple pixel point operations as well as convoluting spatial operations were used to minimise the calculations needed for effective edge, object and straight line detection.

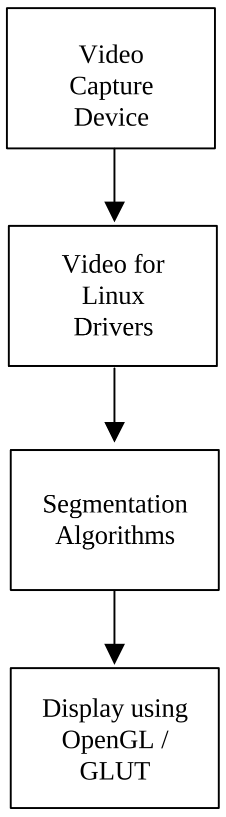


Figure 1. Overall System Design

Figure 3 shows the approach that has been developed. Several algorithms have been employed, including a computationally inexpensive edge detector, a recursive segmentation algorithm that defines possible objects and a trigonometric line detector. Objects that may be of interest due to their grayscale representative color are extracted using low computationally intensive algorithms [2]. A recursive algorithm [1] is then used to segment edge pixels into lines. By means of thresholding the segmented data, the relative density of straight lines and parallel lines is determined.

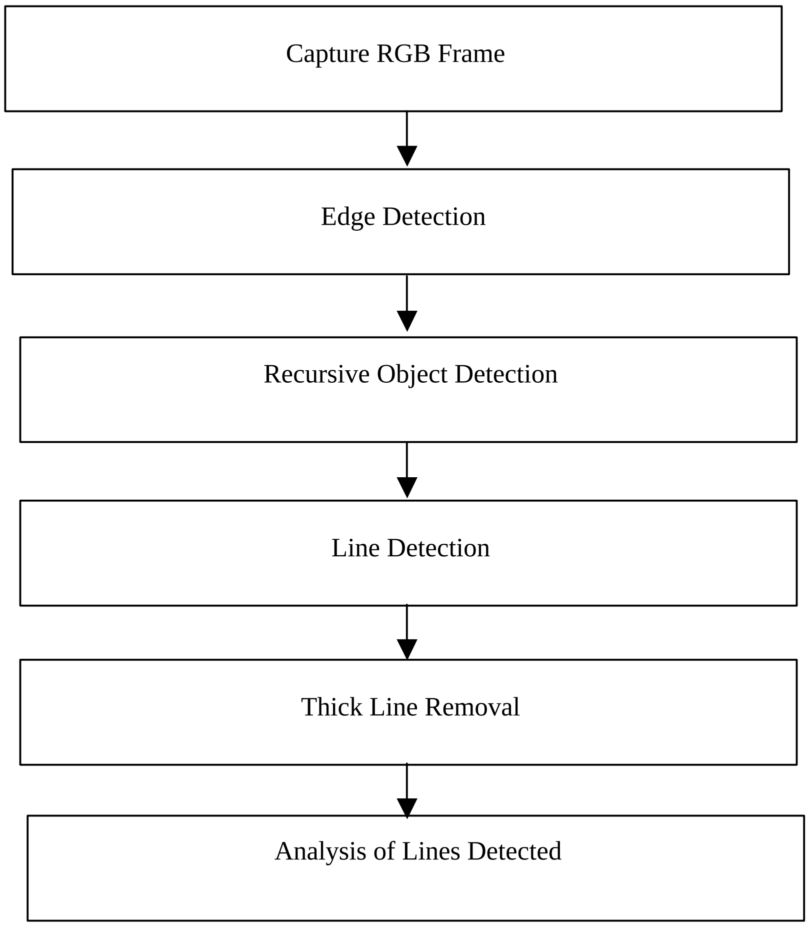
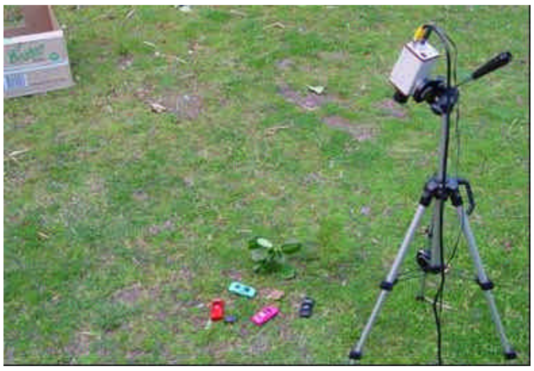


Figure 1. The Segmentation Algorithm Components

Results



The simulated scene was set under ordinary outside lighting conditions, as can be seen from Figure 4. The test setup was decided to have a mixture of simulated cars, rocks, trees as well as ground foliage that may be seen when using the device in a paddock.

Figure 2. The Simulated Scene Setup

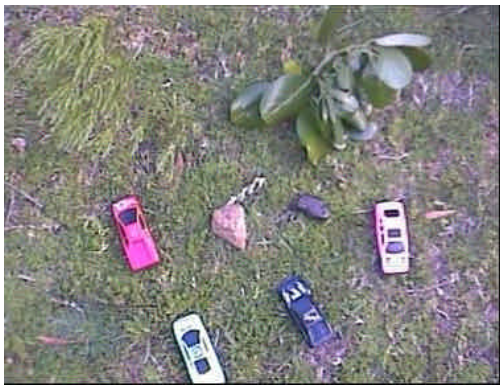


Figure 5 shows the input through the CCD capture device. One important thing to notice is the lack of colour depth when compared to the digital camera photograph in Figure 4.

Figure 3. RGB Capture from CCD capture device

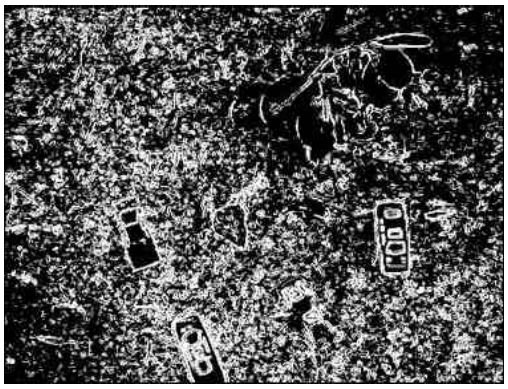
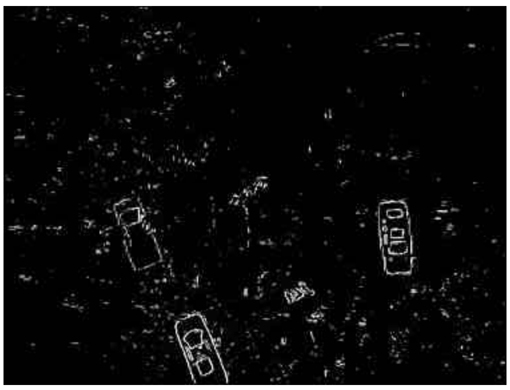


Figure 6 shows the edge detection images show the resultant after they have been passed through the edge detection filter. The low threshold picks up a lot of variable noise due to the low tolerance in differences in the background grayscale image. The right hand tree is clearly seen, while the black rock is missed completely, while the black rock car is very hard to distinguish from noise.

Figure 4. Edge Detection with Low Threshold



The higher threshold edge detection is seen in Figure 7, it shows the edges of the objects a lot more defined. The outlines of the cars are clearly seen, as well as a little noise near where the light coloured rock is situated. Like the low threshold image, the high threshold seems to miss the detection of the black rock completely, but now also the black car.

Figure 5. Edge Detection with High Threshold

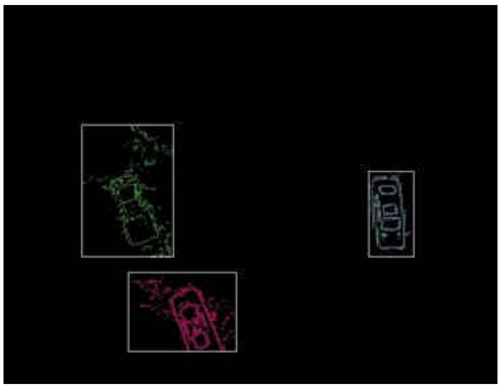


Figure 6. Object extraction run on High Threshold Edge Detection

Figure 8 shows the resultant output of the object detection algorithm. The three cars are clearly defined, with little or no background noise. The white squares are automatically produced by the program output as being the bounds of each object.

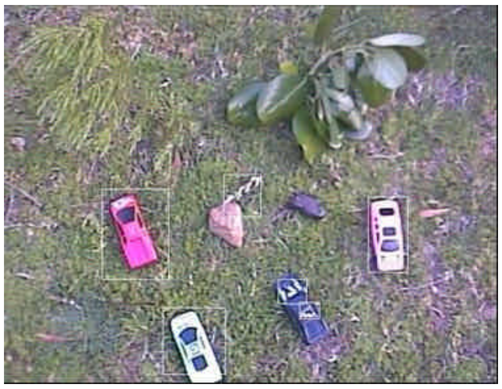


Figure 7. Image with detected

Figure 9 shows the object extraction results on the original RGB frame. The input to the object detector was the low threshold edge detector, with a lot of noise. This shows the resultant of how well the object extraction method can be tweaked to work quite well.

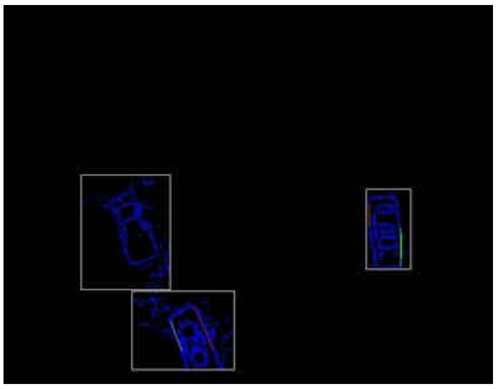


Figure 8. Straight Line Detection run on Object Detected Image

Figure 10 presents the straight line detection, with the background input image shown in blue. Straight lines can be clearly seen on all cars, and ignoring the rest of the scenery such as the foliage, the rock and the trees.

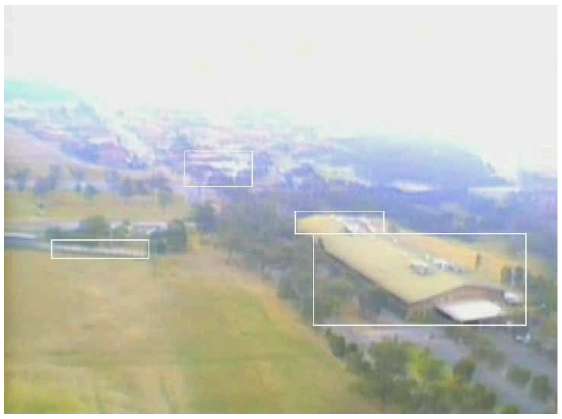


Figure 9. Real-Time Video Footage with Detected Objects

Figure 11 shows the detected objects from realtime footage of a UAV in flight. The detected objects include both the building and the section of road, as expected. There is also a wrongly detected object towards the horizon that may be hard to see, but upon inspection of the edge detected image, it is clear to see why this was mistaken.

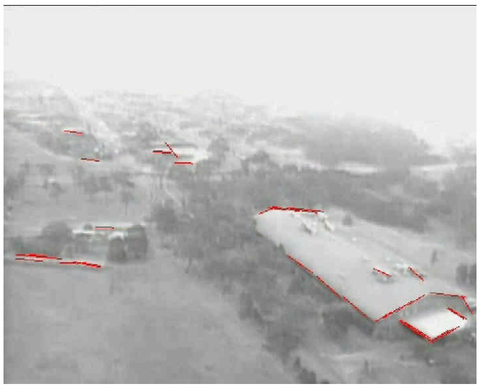


Figure 10. The Detected Straight Lines

The line detection algorithm shows the extracted lines found from each object in Figure 12. It is quite clear to see that they are mapping the edges of the building as well as the section of road. There are small lines extracted also from the objects closer to the horizon that were visible in the object detected image above.

Figure 13 shows another RGB captured image that was used. It displays much the same style of objects, being a paddock, a building or series of buildings and foliage.

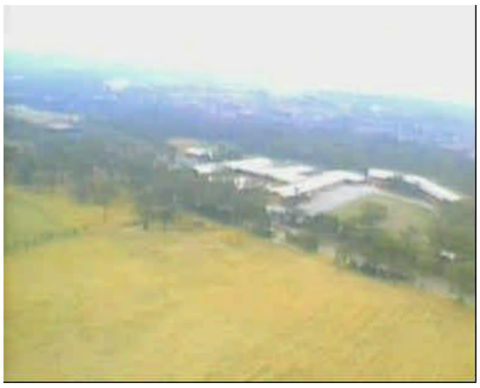


Figure 11. Real Time Video Footage

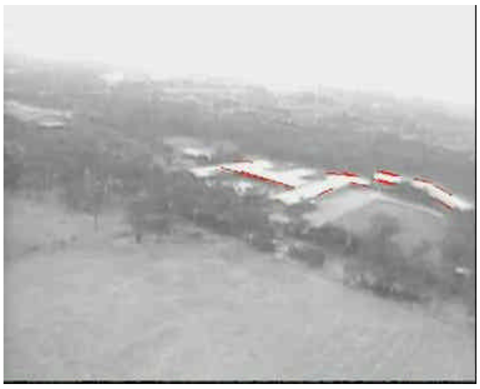


Figure 12. The Detected Straight Lines

Finally, Figure 14 shows the straight lines detected in the image shown above, most building segments were detected as being straight lines and all trees and foliage were ignored.
The results show that the algorithms implemented in this design are satisfactorily extracting objects of interest, as well as extracting some straight line segments from these detected objects.

Improvements / Future Prospects

This paper has demonstrated one way in which unnatural objects can be extracted from real-time video images. This avenue still requires a large amount of research, and is by no means a topic that has been solved particularly elegantly. Due to the effects of ever increasing processing power, these methods can be iterated over several generations of hardware.

.

This type of system was never designed with the thought of implementing a home PC onboard a UAV. It was however thought of being used onboard a UAV with the use of VHDL or an equivalent hardware based language. Computational devices such as FPGA's can be programmed with the algorithms discussed in this thesis for processing, determination and logging all onboard the UAV.

The fundamental reason for building this type of system is because it helps to solve the limitations of both the GPS system and the remote control device. The remote controlled UAV mechanisms in place are limited by the range at which they must be used. In addition, GPS type systems must follow a pre-programmed set flight path, in which the user sets various waypoints and loads the coordinates into the onboard computer. The computer then flies out the course, possibly taking images and video footage of interesting unnatural objects.
The design of the device described in this paper could possibly be seen to run in conjunction with these other two techniques, providing a system capable of flying a set route until it locates an interesting unnatural object, where the autonomous object detection takes place. After the detection has been successfully satisfied, the system could then return to its set course plotted by the GPS system or change its mission specification entirely.

Conclusion

An unnatural object detection device was successfully implemented and was found to produce the expected output in most of the images that were processed. Both a simulated scenario as well as real UAV footage was tested, with the results being quite promising. The object detection device was found to function correctly in approximately 80-90% of situations that it was presented, while the line detection algorithm successfully extracted a significant-enough amount of straight lines from each object that contained them.
This type of system is by no means a fully automated flight control device, merely a system capable of helping to bridge the gaps in terms of current system limitations.
As processing power is increased, more complex algorithms can be implemented, providing a more effective and robust system of operation.

References

1. Price, A.R., "Mobility and Vision for Mobile Robots in Non-Deterministic, Competitive Environments", 1999, Deakin University.
2. Fisher, R., Perkins, S., Walker, A., and Wolfart, E., "Image Processing Learning Resources", http://homepages.inf.ed.ac.uk/rbf/HIPR2/hipr_top.htm
3. Sinopoli, B., Micheli, M., Donato, G., and Koo, T., "Vision Based Navigation for an Unmanned Aerial Vehicle", 2001, University of Berkley http://www.dam.brown.edu/people/mariom/PUBLICATIONS/icra01.pdf