Navigation using self-initialising active contours.

Derek Molloy[1] and Paul F. Whelan[2]

Vision Systems Laboratory, School of Electronic Engineering,
Dublin City University, Glasnevin, Dublin 9, Ireland.

ABSTRACT

This paper examines a novel approach for extracting motion information to allow the autonomous navigation of an intelligent mobile robot using computer vision in a moving camera, moving object environment. The approach begins by extracting low-level scene feature information using algorithms such as the SUSAN corner and edge detector. A routine is described for converting the information obtained from these stable features to initialisation information for creating active contour models or 'snakes'. Multiple open and closed active contours are identified in an initialisation frame from this primary feature extraction. These contours are allowed to converge more closely to the features to which they are attached. These contours are then allowed to converge to the features within each frame through image sequences, with criteria for the re-initialisation of new contours when motion information in the sequence or a region becomes sparse. The information received from these contour models is then used to determine the motion information in the scene. Reasons for this approach are outlined and justified. This theoretical approach is then applied to the practical cases of a mobile robot navigating indoor scenes. Large sections of this approach have been implemented in the Khoros environment, with new routines written for this approach. Promising results are already available and this approach is being examined to allow the extraction of depth information in the scene for assisting navigation using a form of '3-D snakes'.

Keywords: corners, SUSAN, active contours, snakes, edge-linking.

2. THE PROBLEM AND MOTIVATION

If we have a scene that is being viewed by a camera, movement of the objects in the scene or of the camera itself, will in general result in changes of the intensity values in the image being viewed. Based on these changes we try to deduce the motion of the objects in the scene, or of the camera that caused these changes to occur. In this paper, we will examine the situation of a moving camera and moving objects in the same scene. In such a situation, the moving background plays a huge role, where objects cannot be easily extracted, or differentiated from the moving background. Moving objects also introduce problems with occlusion, not just of the background but of other moving objects in the scene.

This vision system is being used in the development of an autonomous mobile buggy for the navigation of mainly indoor scenes. The scene that has been chosen to be used for this paper is an indoor corridor. Some of the application possibilities of such a robot include robot guide dogs, autonomous indoor messengers and factory transport units [8].

Many different approaches have been examined for this particular application. Optical flow and feature matching techniques were examined in detail and many problems were found with these approaches. Optical flow while giving good results was found overly computationally intensive for this particular implementation. General feature matching techniques, while less computationally intensive can sometimes give more accurate results. However, it was found very important to choose the correct type of features for a particular scene. The initial approach that was adopted was to use corners as features for matching between frames [9].

3. CORNERS AS FEATURES

The term 'corner' is quite vague, and can be defined no more accurately than: *a position in the 2-D array of brightness pixels which humans would associate with the word 'corner'* [1][2]. Corners are therefore points where the gradient direction changes rapidly, and correspond to physical corners of objects.

For motion analysis using feature matching, the detection of features must follow these requirements:

[1] Derek Molloy, Email: molloyd@eeng.dcu.ie, WWW: http://www.eeng.dcu.ie/~whelanp/vsg/vsgper.html
[2] Paul F. Whelan, Email: whelanp@eeng.dcu.ie, WWW: http://www.eeng.dcu.ie/~whelanp/home.html

- Consistency: features that are to be used must be detected consistently through frames if they are to be used as the basis of subsequent processing.
- Accuracy: The features must be located precisely, especially in structure from motion, where an error will be magnified under the projection into 3-D space.
- Complexity: Computational speed is a very important issue, as in real-time applications, this primary stage must be as fast as possible, performed at every iteration of the algorithm. The possibility of parallel implementation would also be beneficial.

3.1 The SUSAN Corner Detector

The 'Smallest Univalue Segment Assimilating Nucleus' (SUSAN) by Smith [1] is a form of corner detector, and is quite unique. The principle behind SUSAN is very clear. Taking a mask, in this case a small circular mask and sequentially moving its center across every pixel in the image set and examining the local information at each step, the following theory results. In *Figure 1*, we see a sample case for the SUSAN corner detector. The algorithm examines the intensity of the pixel at the center of the nucleus of the mask. All pixels surrounding this point with similar intensity are then grouped together into a 'univalue segment assimilating nucleus' (USAN). Mask A, is correctly detected as a corner since it has an USAN area less than one half the area of the entire mask. In the other four cases, the USAN area is too big for a corner to be detected. One of the advantages of this algorithm is that since there are no brightness spatial derivatives, the 'snow' in the image is not amplified. Other corner detection methods have attempted to reduce the noise in the image by smoothing the image before performing the algorithm, but this leads to local distortion and a loss of local information. The default mask size that SUSAN uses is a 5x5 mask, with three pixels added onto each edge, giving a reasonable approximation to a circle.

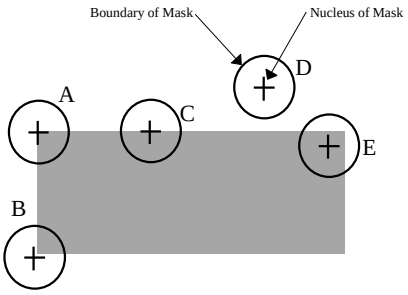


Figure 1. SUSAN sample data case [1].

The use of edges as matching features was dismissed due to the need to apply lines to represent the edges in the images. Lines have the advantage of being more stable than feature points, as they are altered less by lighting conditions. The need for curve fitting and local based aperture problems also suggested corners as a more suitable token for the feature-matching problem. Many commonly used corner detectors, such as the Laplacian fail to detect junctions, so a second stage and detector is needed to compensate this problem. The SUSAN algorithm has been found to be an excellent corner finder that is very suitable for such tasks as motion segmentation and 3-D structure from motion. It is fast and stable, producing well-localised corners. In addition, since there are no derivatives, it is very good at dealing with noise.

4. APPLICATION OF SUSAN TO MOTION ESTIMATION

The SUSAN corner detector method of Smith [1] was applied to the motion estimation problem. This was performed in a similar way to Smith [3] and Brady & Wang [4] where optical flow calculation was performed at the corner feature points. Testing was performed on a sequence of images captured from a corridor in D.C.U. and the SUSAN algorithm was implemented within the Khoros[3] environment. *Figure 2*, shows two of the frames of the sequence after the SUSAN corner detection algorithm had been applied. The 'corners' in the image are displayed by a single black pixel surrounded by 8 white pixels to form a 3x3 square. In the case of Smith [3] and Brady & Wang[4] optical flow techniques were used to determine the motion occurring at the particular 'corner' point in the image.

[3] Khoros™, a product of Khoral Research Inc.(*http://www.khoral.com/*)

An algorithm was developed based on the 3x3 area surrounding the detected 'corner' point, which cannot be described as optical flow calculation, but a less computationally intensive statistical approach. The SUSAN algorithm was modified to output the local grayscale intensities of the surrounding pixels to this corner-matching algorithm. Although the image is

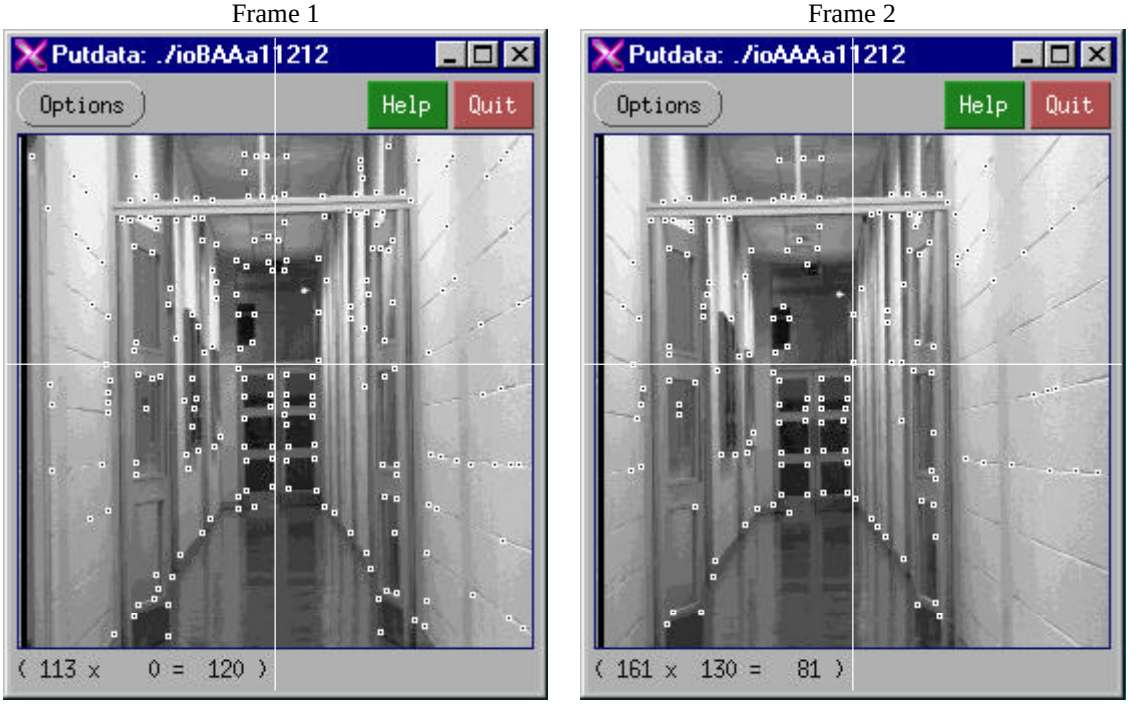
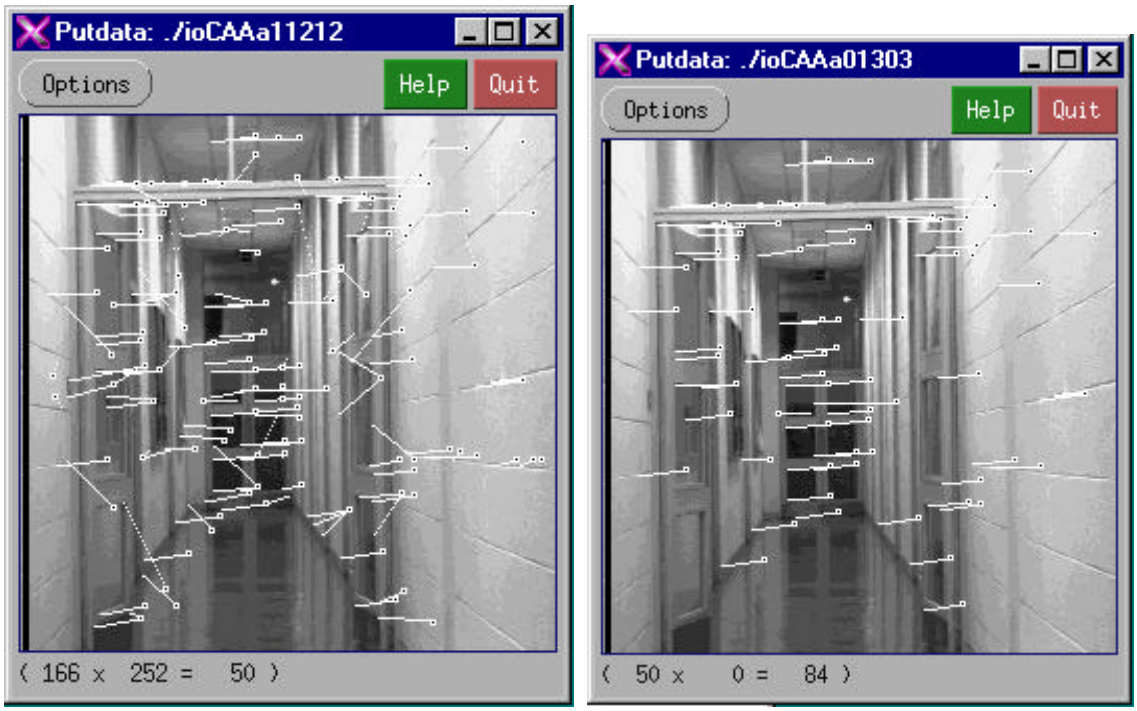


Figure 2. Real world implementation of SUSAN

changing in grayscale intensity, the overall intensities at a particular corner point in the scene will be relatively consistent. *Figure 2*, shows the corners that are detected in two frames of the corridor sequence. A white grid is superimposed over the images to help demonstrate the distances moved by similar corners between the two frames. The developed corner-matching algorithm is then applied to these sets of corners. This is given for a 3x3 local area, as this size of local area was found to be relatively invariant to rotations in the image, allowing scene rotation of up to 45° before results become unstable to any degree. This formula thus calculates the magnitude of the differences in the local image intensities between the detected pair of corners (M_T). This value will be quite low, as the 3x3 spaces are quite similar for a correctly matched pair of scene corners. The algorithm also calculates the magnitude of the distance between the pair of corners (D_T), creating a D_{Total} value by summing the two totals ($D_{Total} = aM_T + bD_T$ *wherea and b are factors of importance*). Whatever pair of corners has the lowest value of D_{Total} is the most likely match and a vector is created between them. They are then removed from the matching list and the matching algorithm continues to the next corner.

This system was implemented within Khoros and initial results were encouraging as shown in *Figure 3(a).* The algorithm matched the corners based entirely on this D_{total} measurement. This method was subsequently improved by the addition of the following features:

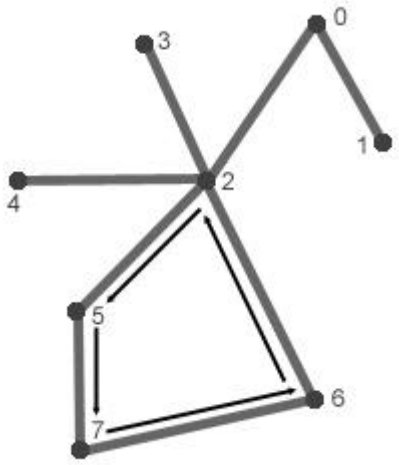
- A distance threshold that defines the maximum distance that a corner may move between two frames, to stop the case where a corner is visible in one frame, but subsequently in the next frame becomes occluded, or is failed to be detected. This basic measure can be calculated dynamically, based on the general movement of the corners in the sequence. When feedback is used, it allows the use of a constraint such as velocity smoothness.
- The addition of a factor of importance of distance also improved results in certain cases, where the distance between a detected corner in one frame and in the next was rated more important than the differing grayscale information by a factor multiple.
- The addition of a pre-median filter was also found to improve results. A median filter before the SUSAN algorithm removes some problem texture information, without smoothing over the corners in the image. This was found to remove weakly detected corners, and improve the overall results of the matching block.



(a) Output results - no filtering (b) Output results - after filtering

Figure 3. The output after post-median filtering

- The addition of a post-median global filter dramatically improved results, but this was found especially applicable to the motion sequence used. It is hoped to implement a local approach in the near future to perform the same filter, without the error propagation of the global approach. The vector field is sparser than the previous results, but using a 'coarse-to-fine' strategy would improve the overall results and allow greater accuracy in the other results. *See Figure 3(b).*



Corner point	Connected to points:				
⓪	①				
①	⓪				
	⓪				

(a) Edge linking algorithm. *(b) Table of connected corners.*

Figure 4. Edge linking algorithm example and extracted table.

5. EDGE LINKING

It was decided to form some relational information between the detected corners in the scene. A few options were examined for this:

- A relational database based on relative position in the scene. However, due to the presence of moving objects it was not clear how such a database would handle occlusion and the presence of the objects.
- A model approach based on relative position was also considered but due to occlusion and the non-rigid behavior of the scene, also caused this to be difficult.

Edge linking was examined to determine this relationship. The edge detection algorithm that was used was also based on the SUSAN algorithm of Smith. This algorithm was found to be a particularly accurate and consistent edge detector. Filters were also applied to the image after the edge detection to attempt to close any broken links in line segments.

An edge-linking algorithm was developed to attempt to extract the relative position of the connected corners within the image. This algorithm had the following form (*examining Figure 4(a)*): For each corner point detected (identified in any order but placed in a sorted list) – follow the edge until each possible direction is found and the corner point that it is linked to is identified. A list is created of the form of *Figure 4(b)*, and this list is then passed onto the next algorithm.

From this table, a routine is then called to follow all the possible paths that may exist. Examining corner ① there is only one path to ⓪, which in turn breaks into two directions towards ① and , but since we came from ① we ignore it and go to corner . At corner , we then examine all the directions except ⓪, so there are four more paths that are possible , , and . Each one is examined in turn in the same way. This routine attempts to find closed loops within the path list with greater than two steps (as this would be a trivial line). It also attempts to find the longest path if a closed loop does not exist. The rational for this is so that these contours can be converted into the initialisation of closed and open active contour models.

6. ACTIVE CONTOURS OR SNAKES

The approach described showed good results and a likely approach for the development of a motion description algorithm. It was decided to couple the previous method to that of active contours and develop an even more stable algorithm describing the relationship between the corner points.

Active Contour Models (Snakes) are a popular method for tracking 'regions' through image sequences. Developed largely by Kass *et al* [5], it involves the tracking of simple geometric shapes, in a similar way that humans observe the scene of a moving vehicle. Humans focus their eyes on a particular feature point of the vehicle, and then rapidly refocus (optokinetic nystagamus) to follow a different feature point on the same vehicle. The concept behind snakes is that they are a mechanism of optimising the location of these particular structures, allowing them to be tracked between frames.

The structure of a snake can be as simple as a set of control points (snaxels) connected by straight lines. Every control point has a 2-D co-ordinate position, with alterations to the snake possible by moving the position of the control points. The data structure is also straightforward, involving a circular list, where the last control point references the first control point for a closed contour. Snakes can have a wide range of properties, through a function called the *energy*, by anthology with physical systems. Open contours are also common but there are some problems in determining the desired energy at the ends of the contours [10].

The energy function for a snake consists of two parts, internal and external energies: $E_{Snake} = E_{Internal} + E_{External}$

Where $E_{Internal}$ depends on properties of the snakes such as length or curvature, whereas $E_{External}$ depends of factors such as image structure and constraints that the user has imposed. Forces, either internal or external are then applied to the snake to reduce its total energy.

Internal Forces: One method to shrink the snake and thus its energy is the 'elastic band' energy function. The snake will be better behaved if the control points are equally spaced around the structure to be enclosed (smoothness), and the closer they get together the tighter the structure is being enclosed. This function thus causes the energy to increase with distance and

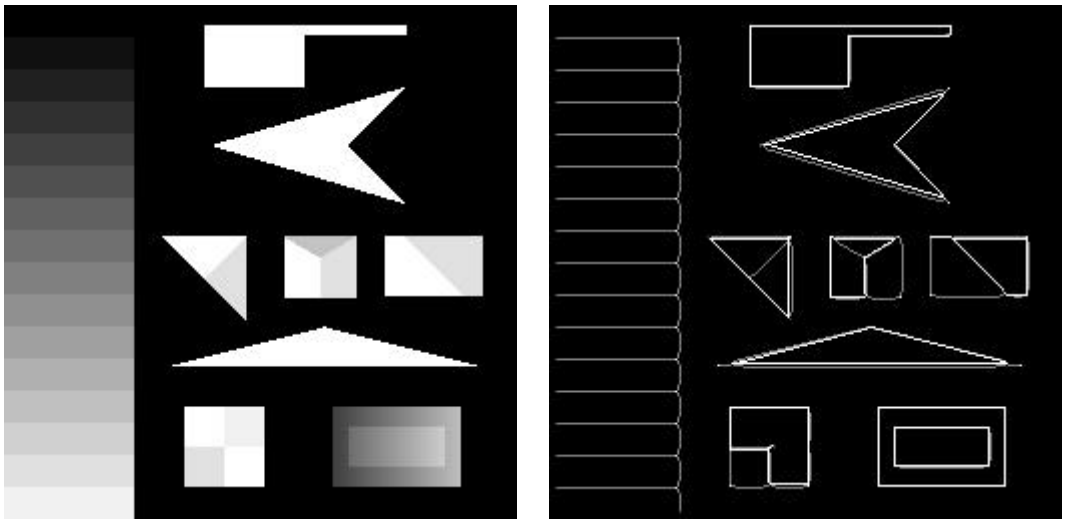
more importantly, increase even faster if the control points are spaced further apart than the usually spaced control points (form of Hookes' Law).

External Forces: External forces determine the relationship of the snake to the rest of the image. This might be a force to simply attach the snake to the brighter areas of the image. This would mean the if the pixel in the direction of increasing *x* was brighter than the pixel in the direction of decreasing *x*, then the control point is pulled in the direction of increasing *x*, as required. It would of course be important in this case to avoid becoming trapped on local pixel locations.

6.1 Initialisation of Active Contours

It is quite common for snakes to be initialised by hand, but for computer vision applications it is necessary to have some automatic way of starting the snakes. If enough computing power is available, it is possible to initialise thousands of snakes on an image, only keeping those snakes that attach themselves to suitable features. This is not an optimal solution.

Other techniques could be used, particularly those techniques that give a region of interest in the image. This might involve differencing techniques or region descriptions built up from clustering [6]. In some cases, the classical and generalised Hough transforms are popular methods that combine modeling and extraction for rigid contours. These methods are relatively insensitive to small vote fluctuation caused by noise and occlusion, however they do not scale very well to recognition in complex scenes where considerable outliers (wild points) are present [7]. The rigid templates used in the Hough transforms cannot account for deformations that arise from changes in the degree of shape of deformable objects in the scene. For very good results, many methods available require a human operator to carefully place the initial snaxels near the boundaries of the features to be tracked.



(a) Standard test template for SUSAN *(b) Output of results after detection*

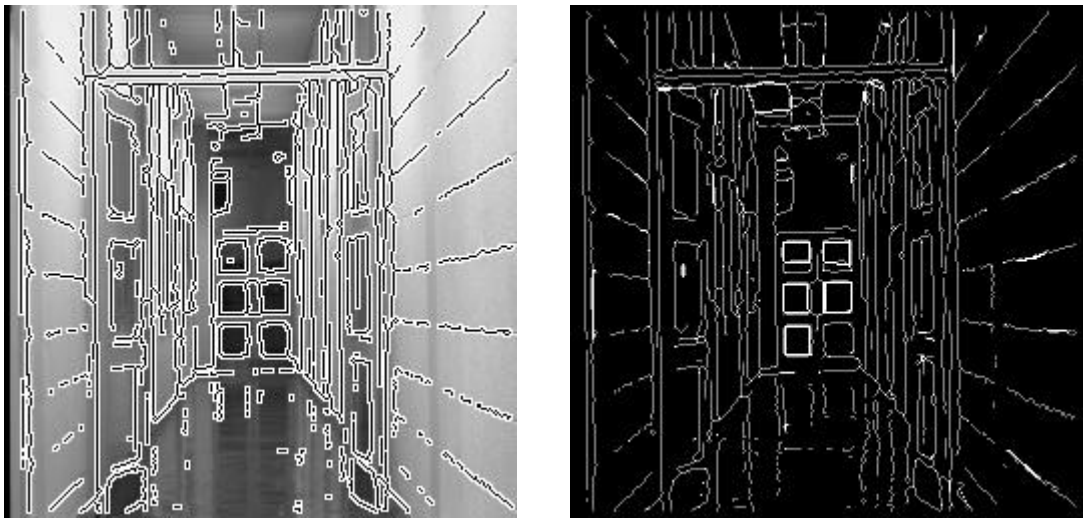
Figure 5. Detection of closed loops for a test template (loops in brightest intensity)

In our case, we wish to wrap a snake around the edges of an object, by minimising the external energy when the control points are on pixels within a high image gradient.

In *Figure 5(a),* we see the sample image used by Smith to demonstrate the SUSAN algorithms. The edge detection and following algorithm that was developed is then used to generate the corner list and connectivity as previously described. In *Figure 5(b),* we see the output of results from the algorithm where the closed loops are detected within the image. These closed loops are indicated in the figure by being double the intensity of the edge-linked detection. On examining the figure, it is clear that more than one loop exists in some of the shapes in the image, however the algorithm only identifies one loop for

a given set of corner points. It could easily be altered to determine multiple loops or the loop with the largest number of corner points if required, but was thought unnecessary in this particular case.

Figure 6(a) shows the algorithm being applied to a real-world scene, the corridor scene. The edges that are detected are outlined with a white pixel on either side of the point where the edge exists. *Figure 6(b)* shows the closed contours displayed over the edge-linked corner image. Clearly, the windows in the center of the figure show the clearest closed loops. In other areas of the image closed loops are detected, but are not clearly visible. This is because the number of corner points in the



(a)Real-world image with edges superimposed *(b) Closed contours displayed brightest on edge image*

Figure 6. Application of this algorithm to detect closed contours in a real-world image

loop is quite low. A threshold on the minimum number of corners can be easily set. The open contours in the image are not displayed here, but are obvious in the image and are easily detected in comparison to the closed contours.

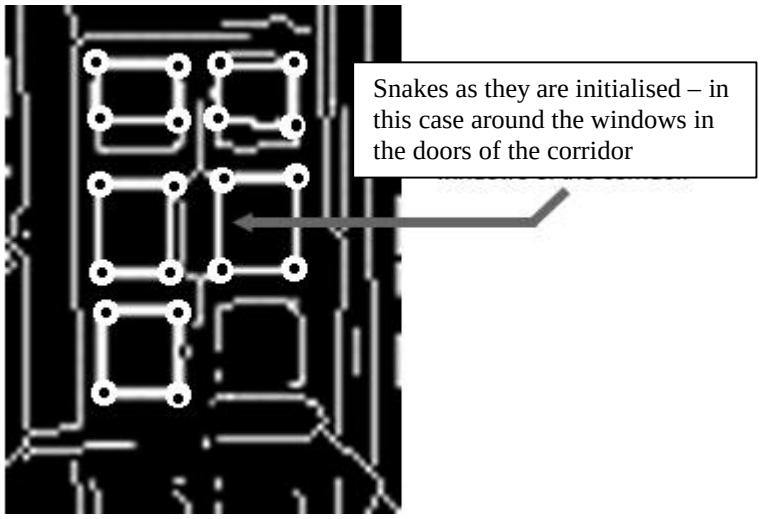


Figure 7. Snakes as they are initialised on the real-world image.

Figure 7, displays an enlarged simulated view of the initialisation of the closed active contour models from *Figure 6*.

The primary snaxels are placed on the detected corner points of the image frame. When the distance between the corner-initialised points is greater than a certain threshold, more snaxels are placed equally on the edge-followed path between the points. The snake is allowed to form itself around the edges of the object, by minimising the external energy when the control points are on the pixels within a high image gradient.

Once the initialisation is performed, the tracking of the features in the scene is assigned to the multiple initialised snakes. There is a momentum assumed in the image so that the snakes can successfully track the features between image frames. This momentum is also useful in dealing with the problems associated with noise and occlusion that may occur in the sequence. When the number of open and closed active contours in the sequence falls below a certain threshold the algorithm halts, re-initialises a new set of contours for the current frame and then continues. This may occur when multiple contours are lost due to severe occlusion or features that leave the view of the camera. After new contours are initialised, an attempt is made to match the contours extracted in the new initialisation with the contours that still existed at the end of the previous image frame. Large sections of this approach have been implemented in the Khoros environment, with numerous new routines written for this approach.

7. CONCLUSIONS

This approach has many advantages over traditional approaches. It is quite a computationally fast method, with the corner detection, edge following and snake initialisation being calculated only at the start of a sequence. The snakes are then allowed to attach to the features through the image sequence. Only after the number of snakes in the scene becomes sparse (falls below a fixed threshold) does the initialisation stage have to be performed again. The snakes also have the advantage of representing actual objects or shapes within the image sequence, allowing for a smooth transition to the motion description section of the problem.

The features that are used for the initialisation of the snakes are stable and can be tracked robustly between the image frames. When occlusion occurs within the image, the assigned momentum property of the snakes causes them to continue to move in the same direction from frame to frame. This means that brief occlusion by objects or of objects does not cause the tracking algorithm to fail. Depending on the amount of occlusion that is expected to occur in the sequence the properties of this momentum can be altered. The algorithm developed also contains the advantages of feature matching, using strong features, while also benefiting from the dynamic properties of active contour models.

At present, the algorithm uses the image gradient to attract the control points of the snakes. A system is currently being developed to attract certain control points to the corners detected by the SUSAN algorithm rather than only the edges of the objects being tracked. The use of region information is being researched in the initialisation of the active contours. This region information could also be used in the dynamic localisation of the control points around the features of interest. Preliminary attempts at this approach have shown some promising results, especially when large objects enter the scene.

This approach is being examined to allow the extraction of depth information in the scene for assisting navigation using a form of '3-D snakes'. A stereo camera approach is to be used where information is extracted by initialising contours on each of the stereo frames and coupling the snakes that have been initialised between each of the two stereo frames. From the difference in the position and shape of two matched active contours, depth information may be extracted. It is expected that this depth information will greatly aid in solving the navigation problem.

ACKNOWLEDGEMENTS

We would like to thank Prof.C.McCorkell and the School of Electronic Engineering for supporting this research to date.

REFERENCES

[1] Noble J. A., *Descriptions of Image Surfaces*, (Ph.D. Thesis), Dept. of Engineering Science, Oxford University, 1989.

[2] Smith S. M., "A New Class of Corner Finder", *in British Machine Vision Conference 1992*, Leeds, Sep, Springer-Verlag, 1992, pp 139-148.

[3] Smith S. M., "ASSET-2: Real-time Motion Segmentation and Shape Tracking", *Proceedings of IEEE Fifth International Conference on Computer Vision*, MIT Cambridge, 1995, pp. 237-244.

[4] Brady M, Wang H, "Vision for mobile robots", *Phil. Trans. R. Soc. Lond*, 337, 1992, pp 341-350.

[5] Kass M, Witkin A, Terzopulos D, "Snakes: Active contour models.", *International Journal of Computer Vision* 1987; 1(4): 321-31.

[6] Etoh, M, Shirai,Y, Asada, M., "Active Contour Extraction Based on Region Descriptions Obtained from Clustering", *Systems and Computers in Japan*, 1993, Vol. 24, PT11, pp55-65.

[7] Fung Lai, K. *Deformable Contours: Modeling, Extraction, Detection and Classification*. Ph.D. Thesis submission at the University of Wisconsin-Madison, 1994.

[8] Molloy D, McGowan T, Clarke K, McCorkell C, and Whelan P, "Application of machine vision technology to the development of aids for the visually impaired" *in Machine Vision Applications, Architectures, and Systems Integration III*, Oct, Boston USA, 1994. SPIE Proceedings 6, pp. 59-69.

[9] Molloy D, *Motion Discrimination in Applied Vision Systems*, (Ph.D. Transfer Report), School of Electronic Engineering, Dublin City University, 1996.

[10] Young, D. "Active Contour Models (SNAKES)", *Sussex Computer Vision, Teach Vision 7*, http://www.cogs.susx.ac.uk/users/davidy/teachvision/vision7.html, March 1995.

Derek Molloy and P.F. Whelan (1997), "Navigation using Self-Initialising Active Contours", in Intelligent Robots and Computer Vision XVI: Algorithms, Techniques, Active Vision, Materials Handling, Proc. SPIE 3208, 14 - 17 October, Pittsburgh, Pennsylvania USA, pp 56-64.