Position refinement for a navigating robot using motion information based on honey bee strategies

Erik Wolfart, Dept. of Engineering Science, University of Oxford
Robert B. Fisher, Dept. of Artificial Intelligence, University of Edinburgh
Ashley Walker, Dept. of Artificial Intelligence, University of Edinburgh

Abstract

This paper summarises an algorithm for the position refinement of a robot using visual information. The algorithm was inspired by techniques used by navigating honey bees and is based on separating close and distant landmarks in a scene using their different optical flow. We also use the optical flow to construct a map of the close landmarks representing their 2D layout in the scene. Upon the agent's return to the vicinity of a previously remembered position, the algorithm compares the information immediately perceivable with the remembered data. Using the information about the distant landmarks to correct the orientation under which the scene is observed, it is then possible to utilise spatial cues offered by close landmarks to calculate a vector pointing to the target position. A few iterations of the movement to the estimated target position allows repositioning within a few centimetres of the original position. The algorithm was implemented and tested using a mobile sensor.

1 Introduction

It is known that the honey bee, like many other insects, combines two methods for its navigation: dead reckoning and global localisation. Dead reckoning is used as a coarse-grained navigation method which allows the bee to return to the vicinity of a previously visited and remembered location. During the global localisation process the bee refines its position and thus eliminates the error introduced by dead reckoning. To accomplish this task the bee compares the visual information extracted from the current nearby landmarks with the remembered information of the landmarks as seen from the target position. From this comparison, the bee can calculate motion vectors, which steer it precisely to the target position. By combining these two methods, dead reckoning and global localisation, and applying them recursively, the bee is able to forage over distances up to 35 km and find the way back to its hive.

The limited computational power of insects does not allow them to perform complicated object recognition or 3D reasoning. Therefore they have to use an efficient method to navigate through their environment, applying only a rather low level of processing to the visual data. However, it is still an open question what kind of information is extracted and how it is used for the position refinement.

The most cited work on this subject was performed by Cartwright and Collet[2]. They undertook various experiments with honey bees to investigate how nearby landmarks are used to guide the way to a known food source. From analysing the search behaviour of the bees Cartwright and Collet created a model of how the bee uses a remembered image to localise itself precisely. In the model, landmarks are extracted by segmenting the image into dark and light areas. Each landmark in the current image is matched to the closest landmark in the remembered image. The model bee uses the apparent size of the landmarks to estimate the radial error to the target position and the bearing of the landmarks to calculate the tangential error. These calculations yield, for each landmark, a correction value which is then averaged into a resulting vector. The model assumes a mechanism which allows the bee to observe the scene always under the same orientation as it did when memorising the scene and uses only static snapshots of the scene. Using this model Cartwright and Collet simulated the bees behaviour on a computer and for many constellations of landmarks and food sources, the results proved to be similar to the observed behaviour of the real bees.

However, an implementation of a global localisation algorithm on a mobile robot based on the above model showed that the information gained from static images is not enough to perform the position refinement in a real environment [4]. Since a static snapshot shows only a tiny temporal fraction of the world it is difficult to cope with physical problems, *e.g.*shadows and occlusion, or to gain any information about the 3D structure of the world.

Other recent papers suggest that insects use motion information in a more sophisticated way than it was previously assumed. Franceschini [3], for example, successfully implemented a robot navigation system inspired by the visual behaviour of a housefly. His results confirm that insects may use the visual processing of motion to construct a rough layout of the 3D environment. Experiments undertaken by Srinivasan [8] show that bees use optical flow to enable them to land on contrasting edges, to distinguish an object from background or objects at different distances. He also suggests that the bee uses optical flow for its navigation [7]. Based on these findings Santos-Victor et al. successfully developed a robotic bee employing two divergent cameras. It uses the difference in the optical flow of the left and the right camera to find its way along a corridor and navigate around obstacles [5, 6].

Hence, the idea emerged to develop a more dynamic algorithm, using the optical flow on the observer's retina induced by its own motion. In doing so, we hoped to gain information about the distance of landmarks to the agent and that the algorithm would be more resistant to the problems faced in the real world.

The project described here devised and implemented an algorithm for the position refinement of a robot using visual information of the surrounding scene. Although the algorithm was inspired by techniques believed to be applied by a honey bee for its fine-scale navigation, the technical details of the algorithm were often determined by the engineering constraints and the desire to develop a localisation algorithm for a robot. It is hoped that the project will yield some new insight into ways in which honey bees may employ dynamic navigatory strategies, but developing a techniques for robot navigation is the primary objective.

2 Summary of the devised algorithm

We devised a position refinement algorithm which exploits motion information and tested its performance using a moving sensor.

The algorithm is based on separating the distant and close landmarks of the scene based on the optical flow pattern they induce on the observer's retina. We compare the resulting images of the distant landmarks obtained from the agent's immediate position and a remembered target position to correct the orientation under which the scene is observed. Assuming a constant orientation, we can correct the positional error using the optical flow and the bearing of the close landmarks.

In the setup for the initial experiments, the camera was placed at one end of the lab with an object (*i.e.*nearby landmark) placed in front of it. Moving the camera vertically across the scene, a series of images was taken. An example image can be seen in Figure 1.

Figure 1: This image is one out of the series taken as input for the experiments. It shows the lab, with the wall in the background at 5.7 m away from the camera. The stool in the middle is about half way and the bar in the foreground, serving as the nearby landmark, is 37 cm from the camera. In this scene, motion is vertical, although honeybee motion is normally in a horizontal plane.

2.1 Extracting the distant landmarks

In the first step of the processing, we have to decide how to extract and represent the landmarks in the image. As opposed to Cartwright and Collet, who segmented the image into dark and light areas, we detect the edges in each image and assume that each edge point originates from a landmark. The advantages of edges are that they are easy to extract from the image, they are local (as compared to regions, which require a global aggregation) and that they are well suited for the further optical flow processing.

Since the camera is only moved along one axis of the image plane, it is possible to detect only the edges perpendicular to the movement. This is accomplished by convolving the image with a [-1,0,0,0,1] edge detection mask, followed by non-maximum suppression and a thresholding [1]. To simplify the edge tracking we also introduced a minimum distance between two edges.

The edge detection is performed for each image. Between two successive images, each of the edge pixels moves in the opposite direction to the camera movement by an amount depending on the distance of the corresponding landmark to the camera. The number of pixels Δi moved between two images is given by

$$\Delta i(z) = \frac{\Delta y}{(\frac{z}{f} - 1)\,\alpha} \simeq \frac{f\,\Delta y}{\alpha\, z} \tag{1}$$

where α and f are the pixel width and the focal length of the camera, Δy is the distance by which the camera is moved between two images and z is the distance to the scene edge from the camera. The parameters were chosen so that $\Delta i < 1$ for all but nearby edges.

In the next step, we apply an IIR (Infinite Impulse Response) low-pass filter to the time signal of the image edge strength at each pixel. The filter output value of each pixel in an image k is given by

$$y_{ij}(k) = A\,\omega_z\, x_{ij}(k) + (1 - \omega_z)\, y_{ij}(k-1) \tag{2}$$

with A and ω_z being the DC-amplification and the cut-off frequency of the filter. $x_{ij}(k)$ is the value of this particular pixel in the kth edge image and is, since the edge image is thresholded, either 0 or 1.

The maximum value which an edge can produce in the filtered image $y_{ij}(k)$ depends on the speed with which the corresponding edge pixel moves over the image. Since distant edges move slower over the image, the corresponding pixels will have a higher filter output than an edge pixel corresponding to a close landmark. An example of filtering 20 successive images can be seen in Figure 2. It shows that the edges corresponding to features at the distant wall appear thin with a high intensity whereas the edge corresponding to the object placed in front of the camera is rather blurred and low in intensity.

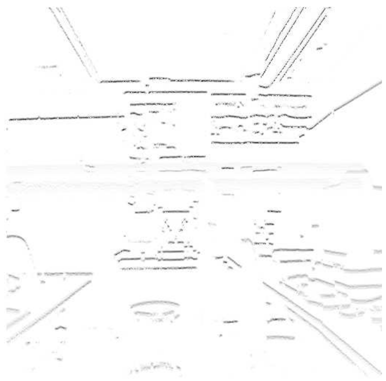

Figure 2: Output of the IIR filtering the edge detections of 20 sequential images, each taken from $\Delta y = 0.9mm$ apart from each other. Note the blurred region in the middle of the image arising from nearby edges.

Figure 2 suggests that we can apply a distance related threshold to the filter output thus receiving an image containing only the edges of the distant landmarks. The relation between the maximum output and the distance to the landmark can be estimated with

$$f_{max}(z) = y(\frac{1}{\Delta i}) = A\,(e^{-\omega_z \frac{1}{\Delta i}} - 1) = A\,(e^{-\omega_z \frac{\alpha\, z}{f\,\Delta y}} - 1) \tag{3}$$

The threshold is simply given by Equation (3), where the threshold τ_0 corresponding to a distance z_0 is $f_{max}(z_0)$. Figure 3 shows an example where a threshold corresponding to a distance $z_0 = 0.9m$ was applied to the above filter output.

2.2 Obtaining the close edges

So far we managed to extract the distant objects from the current scene (and we will use them to correct for the orientation error introduced by dead reckoning). To do the position correction, we will need to extract

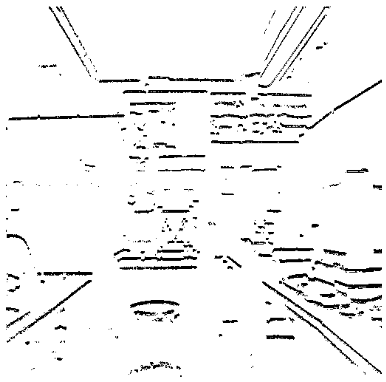

Figure 3: Output of IIR filter (see Figure 2) thresholded with $\tau = 20$, which selects features further than $z_0 = 0.9\,m$.

the edges corresponding to the close objects.

The idea is simple: at each camera position, the algorithm has extracted two images:

1. one image with all edges of the current scene.
2. another image with the edges of the distant objects only (Figure 3).

If we now take the first one and suppress all edge points which also appear in the second, we get, as a result, an image where only the edges corresponding to the close objects remain.

Due to noise, not all edge pixels in the image of close landmarks to actually correspond to close edges. In order to remove those pixels we had to take several measures:

- The detected distant edge image is dilated in the horizontal and vertical direction before it is used to suppress the distant objects in the current edge image. This corrects for small errors in the edge positions and also for the delay of the filter output.
- Single isolated pixels in the resulting image containing the close edges are suppressed, since they are not likely to correspond to any landmarks in the scene.
- The edge detecting process used to obtain all edges from the current image position is made less sensitive than the one used to detect the edges for the filter input. This means that there are more distant objects in the image used to suppress than in the one from which edges are removed.

As can be seen in Figure 4, all remaining pixels in the image of the close edges do actually originate from the landmark placed in front of the camera.

Figure 4: The IIR filter output in Figure 2 was used for suppression. It was previously thresholded with $\tau = 15$ (corresponds to $z_0 = 72\,cm$) and extended in 3 pixels in the vertical and horizontal directions.

2.3 Calculating correction vectors

The deviation from the remembered site introduced by dead reckoning consists of two parts: an error in orientation and an error in position. Since we divided the scene in nearby and distant landmarks we now may correct for these two errors separately. This has the following advantages:

- The orientation can easily be corrected using the distant landmarks.
- The distant landmarks don't provide any information about the local positional error. Removing them from the image simplifies the information from which we have to extract the positional correction vector.
- Correcting the position error is further simplified since we can assume a constant orientation.

2.3.1 Orientation error

To estimate the sensor's current orientation error, the distant landmarks of the current scene are compared with the distant landmarks as seen at the target position. Since we can assume that the observer is in the vicinity of the target position, any difference in the two images of the distant landmarks can only originate from a difference in the observer orientation.

To obtain the error, we first project the 2D edge images into one dimension perpendicular to the direction of image motion. This yields a 1D function (for each the target and the current position) in which local maxima correspond to distant landmarks in the scene. The two functions are then correlated and the position giving the maximum value of the resulting correlation function corresponds to the orientation error between the current and the target position. By projecting the image into one dimension we lose some information and also increase the risk for a mismatch. However, since the introduced error changes the orientation only within a plane the 1D correlation function provides enough information. Test have also shown that for this task the results obtained from the 1D correlation are accurate and reliable enough. The advantage of the 1D correlation over a 2D is the faster processing time.

The equation used to obtain the correlation function from the 2D images is given by

$$C(k) = \frac{1}{N} \sum_{\nu=0}^{N} \left(\frac{1}{M} \sum_{m=0}^{M} d(m,\nu)\, s(\nu - k) \right) \tag{4}$$

where $d(m,\nu)$ is the distant edge image, ν is the direction of motion, m is the orthogonal direction and $s(\nu)$ is the stored image formed by projecting the distant edge image in the m direction, as is done in (4). A conversion factor (determined by the camera parameters) transforms the maximum of $C(k)$ into the actual angle.

2.3.2 Position error

The orientation error induces the same shift into the image of the close edges as it does into the image of the distant edges. Since we now know this shift, we can correct the image of the close edges by shifting it by the appropriate amount. Once we correct for the orientation, we can assume that a difference in the bearing of the nearby landmarks between the current and the remembered scene results from a positional error.

We estimate the optical flow of the close edges by tracking them as the sensor moves. The corresponding problem is quite restricted, since we know the direction of movement and we introduced a minimum distance between the edges. We also assume a minimum distance of the landmark to the camera (i.e. landmarks which are too close won't be detected). This leaves us with a well-defined area in which to search for the matching edge pixel. To obtain a reliable value for the optical flow, we calculate the edge position in sub-pixel accuracy and use the difference of the positions in two successive images as input into a low pass filter. Thus we obtain, at the output of the filter, a smoothed value of the optical flow.

Using the optical flow of the edges, we are able to estimate the distance of the corresponding landmarks and together with their bearing we can construct a map of the two-dimensional layout of the nearby scene. Correlating the maps calculated at the current and the remembered position yields a 2D correlation function. Its maximum value indicates the best overlay of the two maps and hence the position of the maximum value corresponds to the shift between the current and the remembered position. By multiplying the shift with a conversion factor, we obtain a correction vector pointing from the current to the target position.

This pixel by pixel method is opposed to Cartwright and Collet's model, which extracts the landmarks from the image and tries to match each landmark from the current scene to a landmark in the remembered scene. However, even from a biological perspective, we feel justified in taking this method because the correlation approach could more obviously be implemented in neural structures than could a symbolic algorithm such as that used by Cartwright and Collet.

As we now calculate an error for the position and orientation we can correct them by physically moving the sensor. Since the algorithm gets more accurate the closer the current position is to the remembered position the whole algorithm is applied iteratively. Thus we start off with a rough estimate for the initial error and tune our position with further iterations.

3 Results

For the final experiments the algorithm was implemented on a stationary robot holding a camera in its gripper. Features in the background of the lab were used as distant landmarks. Rods were placed in front of the camera to be used as nearby landmarks. In order to memorise the target position, the camera takes a series of 20 images whilst being moved parallel to the ground and perpendicular to its optical axis. At this point, the error was introduced by moving and rotating the camera in the plane parallel to the ground. The performance was tested under various configurations of nearby landmarks, for a positional error up to 25 cm and an angular error up to $10°$.

3.1 Separating distant and close landmarks

As can be seen in Figure 3 and Figure 4 the algorithm manages quite well to separate between close and distant landmarks. Similar results were obtained from the implementation on the robot. Problems might occur when the number of close landmarks is increased. In such a case, distant landmarks might be occluded in many images in a series, thus prohibiting the necessary filter output to be classified as distant. On the other hand, if the edge of a close landmark projects to the same image position as a distant landmark, then it will be suppressed from the image of the close landmarks. However, since we use a filter to track the close edges along the image series, we can cope with an edge missing in some of the images.

3.2 Orientation correction

The algorithm estimates the correct orientation well using the distant landmarks of the scene. No mismatches occurred in the case of one close landmark and, at the end of the iterations, the orientational error was reduced to an average value of $0.1°$. If an error of $\pm 10°$ is introduced it usually takes 2 to 3 iterations to correct this error.

Due to the problems described in the previous section, a mismatch might occur if several close landmarks were in the scene. However, provided the error after the mismatch was not larger than the maximum error we can cope with, the algorithm managed to correct the orientational error in the next iteration.

The maximum error which we can correct for mainly depends on the field of view of the camera. Reliable results were obtained, if half of the scene, as seen from the target position, could still be seen from the current position, so that the maximum error is approximately half of the field of view of the camera.

3.3 Position correction

Provided all landmarks could be seen from the error position, the algorithm always managed to find the vicinity of the target position. The remaining error to the target position differed in radial and tangential direction and depended on the distance of the target position to the landmarks. The average remaining error was in the radial direction less than 2 cm and in tangential direction less than 1 cm. The reason for the higher accuracy in the tangential direction is that we obtain a more accurate measure for the bearing of the landmarks than their distance as estimated from the optical flow.

The accuracy decreases as the target position moves further away from the landmarks. The reason being that then the same amount of positional error results in a smaller difference in the bearing and optical flow between the target and the error position. On the other hand, if we move too close to the landmarks, we have the problem that the apparent size of the landmarks increases and hence it occludes more of the background. Good results were obtained for the distance between target position and landmarks ranging from 30 cm and 90 cm.

The algorithm was unreliable when one or more of the landmarks moved out of the field of view. For example, if two landmarks could be seen from the target position, but only one from the current position, it was ambiguous to which of the two initial landmarks the remaining one should be matched and hence a wrong correction vector might result.

This problem is also the limiting factor for the maximum positional error we can cope with. The maximum error in the tangential direction is set by the requirement that all of the close landmarks remain in the field of view. Hence, it is determined by the field of view of the camera, the distance to the landmark and also on the orientational error. In the radial direction the range is limited by the distance which we use to classify close landmarks. In the final experiments all landmarks closer than 1.5 m were classified as nearby landmarks, hence if the error position is further away than this distance, the algorithm will lose the landmark and won't find its way back to the target position.

The average number of iterations was less than 4 when only a positional error was introduced and about 2 more if there was also an error in the orientation.

The average results for some test runs using 1 landmark at different distances with and without angular error can be seen in Table 1 and Table 2. They show that the angular error is very well corrected after the last iteration and also that the refinement performs better in tangential then in radial direction. The results are best when the landmark is 60 cm from the remembered position and decreases when the distance gets smaller or larger.

Distance to landmark	Average remaining error						Average number of iterations
	after 1. iteration			after last iteration			
	radial	tangential	Angle	radial	tangential	Angle	
45 cm	3.3 cm	0.8 cm	0.6°	2.0 cm	0.3 cm	0.0°	3.7
85 cm	7.0 cm	1.5 cm	0.8°	2.3 cm	1.6 cm	0.1°	3.8

Table 1: This table shows the average error of eight runs for both distances, 45 cm and 85 cm. The initial error position was about 20 cm to 30 cm away from the remembered position, no angular error was introduced.

Distance to landmark	Average remaining error						Average number of iterations
	after 1. iteration			after last iteration			
	radial	tangential	Angle	radial	tangential	Angle	
60 cm	4.2 cm	4.2 cm	3.3°	0.5 cm	0.3 cm	0.1°	5.1

Table 2: Average errors of 6 runs with the landmark being 60 cm from the target position. The orientation error before the first iteration was $\pm 10°$, the positional error about 20 cm to 30 cm.

A graph showing a typical search way of the algorithm can be seen in Figure 5. It shows a test run using 1 close landmark at a distance of 45 cm from the remembered position and no orientational error. In the first iteration, the error is reduced to 45% of the initial value and after the last iteration the remaining error is 0.3 cm in the tangential and 0.2 cm in the radial direction.

4 Conclusions

The project presented in this article devised and implemented an algorithm which a navigating agent can use to refine a previously remembered position using visual data. The algorithm employs the optical flow induced by the sensor's motion to segment the scene into close and distant landmarks and to gain information about the layout of the close landmarks in the 3D scene. The information of the distant landmarks obtained from the remembered and the current position are used to correct the orientational error, whereas the close landmarks are used to correct the positional error.

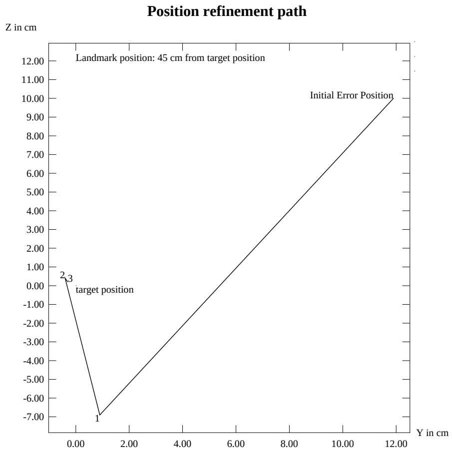

Figure 5: The recorded path of the robot refining its position in 3 iterations.

Our techniques are similar to the one used by Santos et al. [5, 6], although employed for a different purpose. Santos et al. also use spatial and temporal derivatives to compute the optical flow in a series of images, then employing the average optical flow as a qualitative measure for the distance from the camera to an object. By comparing the results obtained from two divergent cameras, their algorithm enables a robot to navigate along a narrow corridor and avoiding obstacles in its way. This shows that the devised strategies might be employed by the honeybee for several purposes, for example during dead reckoning, as shown by Santos et al, and during the global localisation, as in our project. The performance of our algorithm would even improve when using two divergent cameras similar to Santos et al, since the output of the two cameras could be combined into one image yielding a wider field of view.

Although the project was inspired by the bee literature, the technical details of the algorithm were often determined by the engineering constraints. However, using optical flow allowed us to gain important information about the 3D character of the scene. We were able to use a simple technique to keep the observer's orientation constant. Cartwright and Collet also required an orientation correction technique, however, in their model, they assumed it as being given [2]. We could significantly improve the performance compared to an algorithm using only static images [4].

Hence our results confirm the hypothesis that optical flow can provide a navigating agent with a rich supply of information about its world. And although the honey bee's navigation is more precise than our implementation it might well use an algorithm similar to ours as a part of its global localisation.

References

[1] J.F. Canny. A computational approach to edge detection. *IEEE transactions on pattern analysis and machine inteligence*, Nov 1986.

[2] T.S. Collet; B.A. Cartwright. Landmark learning in bees. *Journal of Comparative Physiology.A*, January 1983.

[3] N. Franceschini; J.M. Pichon; C. Blanes. From insect vision to robot vision. *Philosopical Transactions of the Royal Society of London*, 1992.

[4] A. Kick. Autonomous mobile navigation in a complex and real environment using bee-like localisation. *MSc thesis, Dept. of Artificial Intelligence, University of Edinburgh*, 1993.

[5] J. Santos-Victor; G. Sandini; F. Curotto; S. Garibaldi. Divergent stereo for robot navigation: learning from bees. *Proceedings Computer Vision Pattern Recognition, New York*, 1993.

[6] J. Santos-Victor; G. Sandini; F. Curotto; S. Garibaldi. Divergent stereo in autonomous navigation: from bees to robots. *International Journal of Computer Vision, 14, 159-177*, 1995.

[7] M. v. Srinivasan; M. Lehrer; W.H. Kirchner; S.W. Zhang. Range perception through apparent image speed in freely flying honeybees. *Visual Neuroscience, 6:519-535*, 1991.

[8] M. v. Srinivasan. How Bees Exploit Optic Flow: Behavioural Experiments and Neural Models. *Philosopical Transactions of the Royal Society of London*, 1992.

[9] E. Wolfart. Position refinement for a navigating robot using motion information based on honeybee localisation strategies. *MSc thesis, Dept. of Artificial Intelligence, University of Edinburgh*, 1994.