A SIMPLE VISUAL COMPASS WITH LEARNED PIXEL WEIGHTS

Andrew Vardy

Computer Science / Engineering & Applied Science
Memorial University of Newfoundland
St. John's, Canada

ABSTRACT

A mobile robot can determine its rotation with respect to a goal position by comparing the image taken at the goal with the current image. Assuming that both images are omnidirectional, the current image can be shifted horizontally to determine the minimum difference with the goal image, and hence the rotation. The size of the region around the goal wherein this strategy succeeds is reduced by the presence of nearby objects and ambiguous image features. We propose a simple pixel weighting scheme that alleviates these problems. The approach is validated on images captured by a mobile robot in a typical office environment.

***Index Terms*—** Visual compass; visual homing; biologically-inspired robotics.

1. INTRODUCTION

The ability to return to a previously visited places is fundamental to navigation. This return may be guided by a comparison between the image taken at the goal place and the current place. We call this *visual homing*. Visual homing is a demonstrated ability of insects and appears to play a crucial role in their navigational strategies [1, 2]. A number of algorithms for visual homing in robots have been proposed (see reviews: [3, 4]). Many of these algorithms split the homing task into two parts where the rotation and translation are solved for separately. Here we focus on estimating the rotation between the view captured at the goal location and the current view. This estimation process is made much easier by the use of omnidirectional images for which rotations about the vertical axis are equivalent to horizontal shifts of the image.

Methods for estimating the motion of a moving agent (robot or animal) from visual imagery have come from two distinct communities. The computer vision community have proposed relatively unconstrained methods that ideally work in a wide variety of settings [5]. Such methods require estimating the parameters of the motion from noisy data, which tends to be a computationally expensive process. On the other hand, the biologically-inspired robotics community have proposed solutions based on reasonable constraints on a navigating robot (e.g. travel on a planar or near-planar surface). These solutions are generally much simpler than those proposed by the computer vision community. Some researchers have specifically focused on algorithms which might plausibly be implemented in the limited neural hardware of an insect [6, 3].

One such simple homing algorithm proposed by Zeil et al. is based on gradient descent [7]. They found that the raw distance between images, measured by summing the squared pixel differences, increases relatively smoothly with spatial distance. Thus, a robot could travel home by moving so as to minimize this image distance metric. This approach has been referred to as *descent in image distance* (DID) [4]. Further, they found that image distance increased smoothly with angular distance—allowing the the use of a *visual compass* based on DID. An robot at some arbitrary pose (x, y, θ) with respect to a goal position could search in the space of image rotations for the rotated image with minimum distance to the goal image. Ideally, the angle of minimum distance would be θ. Zeil et al. did not explicitly test this idea, but it has been successfully implemented by others [8, 9]. (The smooth relationship between image and spatial distance exists only within a particular region in the neighbourhood of the reference image. Approaches based on DID are only guaranteed to work within this region.)

Stürzl and Möller proposed an improvement to this style of visual compass [9]. They observed that the approach of Zeil et al. works well when all objects are relatively distant from the robot, but poorly when many objects are nearby. The presence of nearby objects has the effect of shrinking the region where the smooth relationship between image and spatial distance exists. Thus, they proposed a very coarse distance filter that could be used to assign differing weights to pixels in the image comparison step based on their estimated distance from the robot. The distance filter they employed involved comparing images taken from very nearby positions with very little change in orientation. A pixel-by-pixel comparison would ideally reveal large changes for nearby objects, but little or no change for distant objects.

However, the presence of nearby objects is not the only problem for a DID-based visual compass. Image regions from different parts of the image with similar appearance can be confused with each other. This can lead to the detection of

false minima in image distance space. Also, the weights used by Stürzl and Möller were obtained by considering both the current and goal images, along with images taken from adjacent positions. However, a suitable distance must be chosen from which to obtain the adjacent images. Further, this chosen distance will depend upon the distribution of objects in the environment and their relative positions.

In this paper we apply a similar strategy to Stürzl and Möller, in that we employ a weighted form of DID to estimate rotation. However in this case the weights are based upon the difference between rotation estimates obtained from individual pixels and the measured rotation obtained from odometry. In general, we do not expect accurate information to be available from odometry when visually homing. However, we assume that at some point in the past the robot did visit the goal position and captured an image. It then moved away from the goal, maintaining an accurate position estimate during the initial part of its outward route. In our visual compass, a pixel will be given a high instantaneous weight during training if its individual rotation estimate agrees with the estimate from odometry. Further, we combine instantaneous weights obtained along the training route into a single weight image.

In the next section the format of our images will be presented. In section 3 the weighting scheme will be formally described. Section 4 then presents experimental results on a database of images collected in a typical office environment. Discussion and brief concluding remarks then follow in sections 5 and 6.

2. RECTANGULAR PANORAMIC IMAGES

The value of a pixel at column i and row j of image I is indicated by $I(i,j)$. The width and height of the image are denoted w and h, respectively. All images considered here were captured from a camera mounted upwards on a robot, viewing a hyperbolic mirror. A low-pass filter is applied to avoid aliasing in the subsequent resampling process. Next, the images are reprojected onto a sphere. This sphere is divided up into a grid of constant angular resolution. The final image is obtained by sampling the low-pass filtered image for all positions of the grid. We will refer to these images as *rectangular panoramic*. See figure 1(d) for four examples of rectangular panoramic images.

One useful property of rectangular panoramic images is that two images taken from the same position but different orientations differ only by a horizontal shift. More formally, let A_θ denote the image captured at position $\vec{a}$ with orientation θ. Then $A_\theta(i,j) = A_{\theta+\Delta\theta}(i+k,j)$. The rotation angle $\Delta\theta$ corresponds to a shift of k pixels. The relationship between $\Delta\theta$ and k is as follows,

$$\Delta\theta = -k\frac{2\pi}{w}$$

We measure angles counter-clockwise from the robot's forward heading, yet image indices increase from left to right (i.e. clockwise). Hence, the negative sign above.

3. A WEIGHTED DID COMPASS

Assume that the snapshot image S was captured from the goal position, which we will also refer to as the snapshot position. The current image C is captured from the robot's current position. We define the squared difference image for a rotation by k pixels,

$$SD(i,j,k) = [C(i+k,j) - S(i,j)]^2$$

The image distance function for shift k is taken here to be the sum of all values in SD,

$$ssd(k) = \sum_i \sum_j SD(i,j,k)$$

The visual compass described by Zeil et al. amounts to finding the horizontal shift k' that minimizes ssd.

$$k' = \underset{k\in[0,w-1]}{\arg\min}\ ssd(k)$$

This method searches through all w possible shifts for the one with the lowest ssd measured across all pixels. If we narrow our focus to consider only one pixel, we would have the following,

$$minshift(i,j) = \underset{k\in[0,w-1]}{\arg\min}\ SD(i,j,k)$$

This amounts to a search for the pixel in row j of C which is most similar to $S(i,j)$. This is exactly the type of search carried out by optic flow methods such as block matching [10]. Here, however, the size of the "block" is 1×1. It may be advantageous to increase the region that must be matched in order to reduce susceptibility to false matches. This can be achieved quite efficiently by applying a low-pass filter to SD. We will indicate the application of such a filter by $filt(SD)$. We redefine $minshift$ to utilize this filter,

$$minshift(i,j) = \underset{k\in[0,w-1]}{\arg\min}\ filt(SD(i,j,k))$$

In general, $minshift(i,j)$ might be quite different from the true shift. However, it may be accurate, meaning that pixel (i,j) is in a patch of the image distinct enough to be matched correctly. The rotation estimate from pixel (i,j) is as follows,

$$\Delta\theta_{i,j} = -minshift(i,j)\frac{2\pi}{w}$$

We wish to assess the accuracy of $\Delta\theta_{i,j}$ and thus the quality of pixel (i,j) for rotation estimation. We take the value from odometry, $\Delta\theta_{odo}$ as an estimate of the true rotation. Assume that $\Delta\theta_{i,j}$ is corrupted from the true $\Delta\theta$ by the noise term ϵ.

$$\Delta\theta_{i,j} = \Delta\theta + \epsilon$$

ϵ is assumed to have a normal distribution with zero mean and standard deviation σ.

We estimate the probability that the true rotation is within some tolerance τ of $\Delta\theta_{i,j}$

$$p_{i,j} = P(\Delta\theta_{i,j} - \tau < \Delta\theta < \Delta\theta_{i,j} + \tau)$$

The probability above can be computed as follows,

$$p_{i,j} = \text{cdf}(\Delta\theta_{i,j} - \Delta\theta_{odo} + \tau, \sigma^2) - \text{cdf}(\Delta\theta_{i,j} - \Delta\theta_{odo} - \tau, \sigma^2)$$

where $\text{cdf}(a, b)$ is the cumulative distribution function of the normal distribution for value a and variance b. Note that if τ is much smaller than σ then $p_{i,j}$ can be approximated as $2 \cdot \tau$ $\text{pdf}(\Delta\theta_{i,j} - \Delta\theta_{odo}, \sigma^2)$, where pdf is the probability density function of the normal distribution.

For the purposes of providing an instantaneous weight for each (i, j) at time t, we utilize the following,

$$W_t(i, j) = 2 \cdot \tau \ \text{pdf}(\Delta\theta_{i,j} - \Delta\theta_{odo}, \sigma^2).$$

It is not sufficient for our final weight to be based on W_t alone. Superior performance is obtained if we estimate instantaneous weight images while travelling away from the snapshot position and then combine them into the final weight image W. On the assumption that the errors are independent, the probability that the rotation estimated for pixel (i, j) is within τ of the true $\Delta\theta$ is given by the product of the instantaneous weights,

$$W(i, j) = \prod_t W_t(i, j)$$

The weights are obtained during a learning phase when odometry provides an estimate of the true $\Delta\theta$ that can be used to evaluate the performance of each pixel in the snapshot image. Pixels will only be given a high weight if they are useful throughout the entirety of the learning phase.

The weighted image distance function for shift k is defined as follows,

$$wssd(k) = \sum_i \sum_j W(i, j) \cdot filt(SD(i, j, k))$$

Notice that if the weight image W contained all ones then the application of $filt$ would have no effect. However, if W is heterogeneous then the filtering will have an impact.

Finally, we estimate the rotation in pixels as,

$$k' = \underset{k \in [0, w-1]}{\arg\min} \ wssd(k)$$

As we care only about the relative values of $wssd(k)$ the constant $2 \cdot \tau$ does not need to be incorporated into the instantaneous weight images W_t. This is advantageous because it means that the only parameters that require selection are σ, those of the low-pass filter $filt$, and the parameters of the training route.

4. EXPERIMENTAL RESULTS

Results were obtained on a database of rectangular panoramic images collected in a typical office environment [3][1]. Images were collected on a 10×17 capture grid with 30 cm spacing between adjacent capture positions. Throughout image capture the robot was oriented in the same direction, with corrections for small changes in heading made afterwards. Thus, ground truth data is readily available: when comparing any two images the rotation should be 0.

For the first experiment position $(5, 0)$ was selected as the snapshot position in the 10×17 image collection known as *original*. The training route was set to run directly upwards from position $(5, 1)$ to position $(5, 16)$.

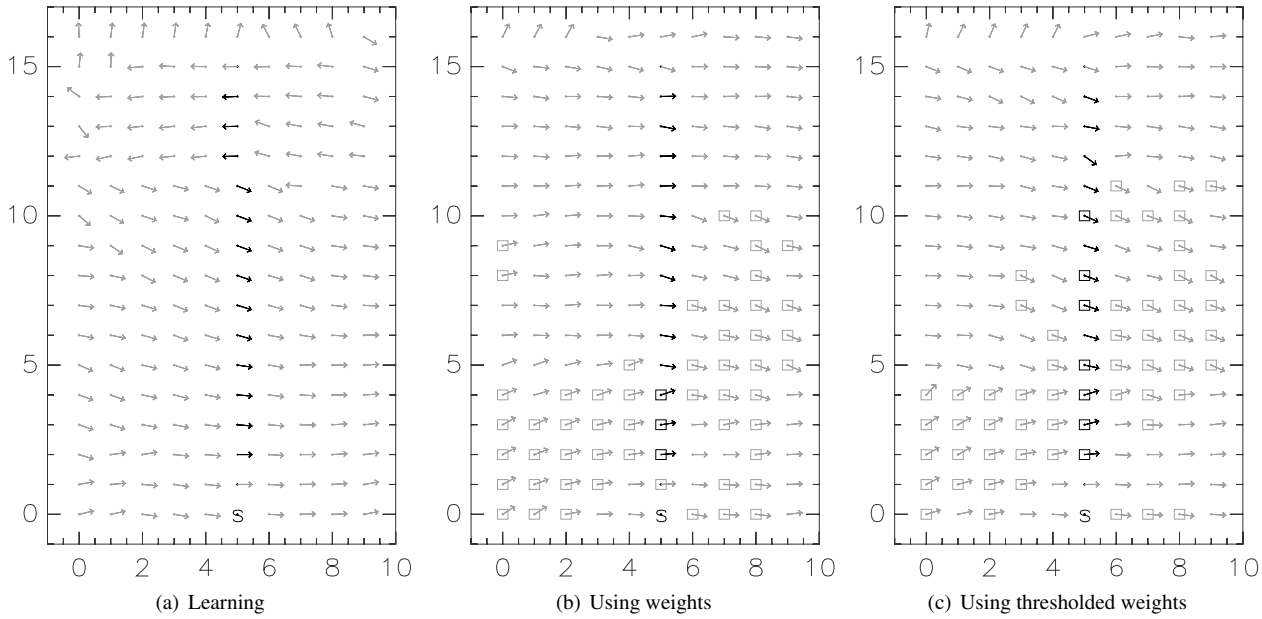
Various subsections of this route were tested for training, and it was found that the weights learned for the route from positions $(5, 2)$ to $(5, 14)$ worked best. We also tested variations on the parameter σ which controls how close a rotation estimate must be to the estimate from odometry for the corresponding pixel to be given a high weight. Performance of the compass appears to be relatively insensitive to this parameter. We chose the value $\sigma = \frac{\pi}{4}$. The only remaining parameters to be chosen are those of the low-pass filter $filt$. We employed a Gaussian filter and tested various values for the radius of the filter's convolution mask (1, 2, 5, 10, 20). We also included the option of no filter at all. A mask radius of 5 was found to yield the best result, although again performance seemed to be relatively robust to changes in this parameter.

Figure 1(a) shows the orientations computed by the standard unweighted DID compass during the learning phase. Figure 1(b) shows the orientations computed when the learned weights are applied. The dark arrows indicate positions along the trained route. Clearly we expect improvement at these positions. However, it is highly advantageous if improvement is seen for other positions. The grey arrows in figure 1(b) show the performance of the weighted DID compass over non-trained positions.

The rotation angles computed by the unweighted compass are qualitatively correct within some region around the snapshot position. It is clear that the size of this region is increased for the weighted compass. The mean and median angular error for the unweighted compass are 0.92 and 0.31 radians, respectively. For the weighted compass they have dropped to 0.18 and 0.13. However, the improvement is not uniform. 55 of 169 positions exhibit less accurate rotation estimates for the weighted compass (indicated by the squares in figure and 1(b)).

Figure 2 compares the performance of the two compasses for snapshot position $(5, 16)$. In this case the improvement is even clearer. The unweighted compass is qualitatively correct only in the very near vicinity of the snapshot. However, the weighted compass is qualitatively correct for almost the entire

[1]Images from this database are available at `http://www.ti.uni-bielefeld.de/html/research/avardy`.



(a) Learning (b) Using weights (c) Using thresholded weights

(d)

Fig. 1. Angle of rotation computed for snapshot position $(5, 0)$ by DID (a), weighted DID (b), and weighted DID with thresholded weights (c). **(d):** Images from positions (5,0) to (5,14) with final weight image W and thresholded weight image W_{thresh}.

capture area. Mean / median angular error drops from 1.29 / 1.70 to 0.30 / 0.24.

The final weight image for the upwards route is shown in figure 1(d) beneath the images used to produce it. It is interesting to note which areas of the image are the most useful for estimating orientation. Areas that maintain a similar appearance throughout the training route have a higher weight. In some cases the similarity is false. For example, in the snapshot image from $(5, 0)$ there is a prominent chair and the upper part of this chair is given a high weight. However, in the training route these pixels are more likely matching the black curtain. This is a false match, but the error is small enough that the estimated rotation is roughly correct (the chair and curtain are relatively close). Note that not all areas which maintain a similar appearance between the snapshot image and the training route are retained. Most notably, the lower part of the image maintains a near constant value of grey—the colour of the floor. However, matches between floor pixels would not tend to correspond to the true rotation. Thus, the majority of floor pixels are assigned a low weight.

Given that the weight image W shown in 1(d) has only a small number of pixels with high weights, we wondered whether it would be possible to reduce the weight values to binary. This was tested by thresholding the final weight image using the average intensity of the original weight image as a threshold. The resulting weight image W_{thresh} has 289 pixels of value 1 out of a total of 11644 ($\approx 2.5\%$) and is shown in figure 1(d). There was a slight reduction in performance found in utilizing this weight image, although the computed rotations were still much more accurate than for the unweighted case (mean/median angular error 0.265 / 0.221). The resulting angles are shown in figure 1(c). For the downwards route the thresholded weight image had 261 pixels with a value of 1. However, in this case there was an improvement in performance (0.298 / 0.243). See figure 2(c).

Finally, in order to discount the possibility that the position of weighted pixels was not critical we tested the use of randomly distributed binary weights: 289 randomly selected pixels were set to 1 for the upwards route and 261 for the downwards route. For 10 different random selections of weights the performance was always worse than with the learned weights on both routes.

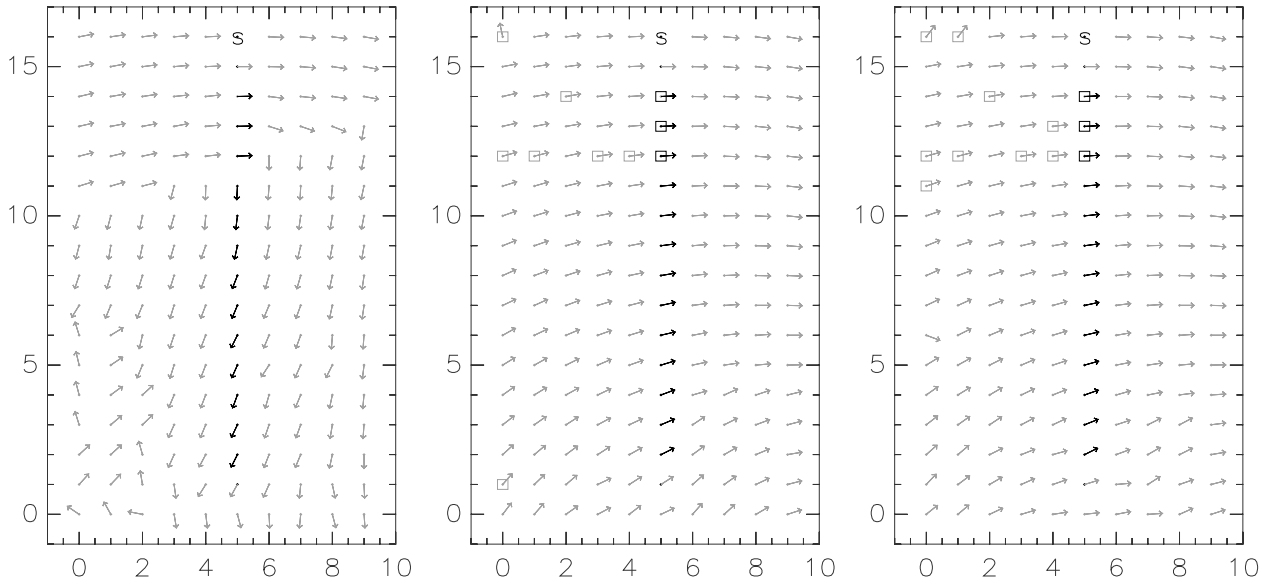
5. DISCUSSION

We have demonstrated that our weighted visual compass provides superior performance to the unweighted scheme for the majority of tested positions. We presume that the unweighted compass performs better in certain positions because it implicitly combines a higher number of matches. While many of these matches may yield incorrect rotations, they will tend to be incorrect in different directions (some positive, some negative) and therefore will cancel each other out. This is an instance of the so-called *democracy effect* noted in [3]. We plan on investigating a means of balancing the advantages of the unweighted and weighted approaches.

It was also shown that the final weight images could be thresholded to binary and still perform better than the unweighted case. This is a positive feature because a binary weight image will clearly be more efficient both to store and apply than a floating-point image.

6. CONCLUSIONS

We have presented a new scheme for assigning weights to pixels for the DID-based visual compass. This scheme has some advantages over the method proposed by Stürzl and Möller in that it will assign low weights both to nearby objects and to image regions that do not provide useful rotation estimates (e.g. the floor). We plan on performing an experimental comparison between these two weighting schemes in the near future. Also, we are investigating the use of the formal framework presented here for developing weights on image features used to estimate translation.



(a) Learning (b) Using weights (c) Using thresholded weights

Fig. 2. Angle of rotation computed for snapshot position $(5, 16)$ by DID (a), weighted DID (b), and weighted DID with thresholded weights (c)

7. REFERENCES

[1] B.A. Cartwright and T.S. Collett, "Landmark learning in bees," *Journal of Comparative Physiology A*, vol. 151, pp. 521–543, 1983.

[2] T.S. Collett and M. Collett, "Memory use in insect visual navigation," *Nature Reviews Neuroscience*, vol. 3, pp. 542–552, 2002.

[3] A. Vardy and R. Möller, "Biologically plausible visual homing methods based on optical flow techniques," *Connection Science*, vol. 17, no. 1/2, pp. 47–90, 2005.

[4] R. Möller and A. Vardy, "Local visual homing by matched-filter descent in image distances," *Biological Cybernetics*, vol. 95, pp. 413–430, 2006.

[5] T. Goedemé, M. Nuttin, T. Tuytelaars, and L. Van Gool, "Omnidirectional vision based topological navigation," *International Journal of Computer Vision*, vol. 74, no. 3, pp. 219–236, 2007.

[6] R. Möller, M. Maris, and D. Lambrinos, "A neural model of landmark navigation in insects," *Neurocomputing*, vol. 26-27, pp. 801–808, 1999.

[7] J. Zeil, M. Hofmann, and J. Chahl, "Catchment areas of panoramic snapshots in outdoor scenes," *Journal of the Optical Society of America A*, vol. 20, no. 3, pp. 450–469, 2003.

[8] Labrosse, "Visual compass," in *Proceedings of Towards Autonomous Robotic Systems, University of Essex, Colchester, UK*, 2004.

[9] W. Stürzl and R. Möller, "An insect-inspired active vision approach for orientation estimation with panoramic images," in *Bio-inspired Modeling of Cognitive Tasks*, Lecture Notes in Computer Science, pp. 61–70. Springer, 2007.

[10] J.R. Jain and A.K. Jain, "Displacement measurement and its application in interframe image coding," *IEEE Transactions on Communications*, vol. 29, no. 12, pp. 1799–1808, 1981.