2012 IEEE International Conference on
Multisensor Fusion and Integration for Intelligent Systems (MFI)
September 13-15, 2012. Hamburg, Germany

Visual Homing for a Mobile Robot using Direction Votes from Flow Vectors*

Robert L. Stewart[1], Michael Mills[2] and Hong Zhang[3]

Abstract— **This paper investigates the problem of robot visual homing – the navigation to a goal location by a mobile robot using visual sensory input. The visual homing approach taken is to consider the flow vectors between a robot's current view and a desired milestone view. The flow vectors can be used to determine an angular velocity command that attempts to align the two views under a constant forward speed. Experiments with a mobile robot have been conducted following the teach-replay approach. By using a sequence of milestone images taken successively along a path, preliminary results show that a robot can successfully repeat the path and navigate to its goal autonomously. The method should be useful for route following and other applications involving visual navigation.**

I. INTRODUCTION

In robotics, visual homing describes the process where a robot navigates to a goal using only visual stimuli. With the capacity to perform visual homing, a robot can navigate autonomously between nodes in an appearance-based topological map [1], or retrace a path in route following [2]. Visual homing has become an important area of research due to these applications and the obvious appeal of a vision-only solution to autonomous navigation.

Humans and many other animals nominally rely on vision to function in their environments. These natural precedents provide examples which we can draw on in the design of artificial-intelligent robotic systems. The use of vision by honeybees provides one such natural example that has been significantly studied. Cartwright and Collett, in their insightful and highly relevant work [3], detail a "snapshot model" of how bees might fly to a goal by using a previously captured retinal snapshot of the surrounding landscape taken at the goal. Their model bee "... continuously compares its snapshot with its current retinal image and moves so as to reduce the discrepancy between the two" [3]. This fundamental idea is inherent in many of the engineered visual homing approaches since developed.

In robotics, there have been various approaches to visual homing, and these approaches can be grouped into quantitative and qualitative methods[4]. The quantitative methods are exact methods that use geometric and motion constraints to establish desired motion trajectories. For example, approaches based on epipolar geometry [5], [6], homographies [7], trifocal tensors [8], [9] and mutual information [10] have been reported. Many of these approaches are similar to those used in the related visual-servoing problem of moving a robot arm to a desired goal pose, using images acquired from a camera attached to the arm (see [11] for an overview of visual-servoing).

Qualitative methods, on the other hand, tend to be more rule-based. For example, Chen and Birchfield [12], [13] use rules based on feature coordinates to control a robot, with features determining a funnel lane that directs a robot towards its goal location. Other qualitative techniques include an image cross-correlation method for correcting robot orientation [2], image warping [4], and methods based on gradient decent using images [14]. Additionally, animal navigation models (see [15], [16] for reviews) are informing many of the biologically inspired visual homing strategies that are being implemented (e.g. [17], [18], [19], [20], [16]).

Whilst numerous methods have been developed, due to various limitations, no prevailing method has yet emerged. For example, many of the approaches still rely on odometry to some extent, require the use of panoramic cameras, contain equations that break down under certain conditions, or are difficult to implement. In this paper we detail a simple and minimalist qualitative method for visual homing that does not require the use of odometry and uses a single monocular camera with a relatively narrow field of view.

Of the existing qualitative approaches, our approach is most similar to that of Chen and Birchfield [12], [13]. However, rather than considering the movement of features in the image plane, which requires feature detection and matching, we take a different approach and consider the attributes of flow vectors for points on a uniform grid. Removing the use of features is beneficial since they are not always reliably found or matched under poor lighting conditions and their spatial location may be unevenly distributed. Additionally, our method differs from [13] in that it does not use odometry or probabilistic models for milestone image transitioning.

The method developed, and detailed in this current paper, considers the presence of flow vectors between a robot's current view and a selected target milestone image. The flow vectors indicate the direction local regions in the current image must move in order to align with the target milestone image. For each flow vector, a direction vote can be made by simply determining the sign of the horizontal component. Taken *en masse*, the votes determine a robot

*This work was partially funded by NSERC.

[1]R. L. Stewart is with the Department of Computing Science, University of Alberta, Edmonton, Alberta T6G 2E8, Canada RLStewart@ieee.org

[2]M. Mills is with the Department of Computing Science, University of Alberta, Edmonton, Alberta T6G 2E8, Canada mmills1@cs.ualberta.ca

[3]H. Zhang is with the Department of Computing Science, University of Alberta, Edmonton, Alberta T6G 2E8, Canada hzhang@ualberta.ca

[4]A useful classification scheme for visual navigation methods is presented in [4].

978-1-4673-2511-0/12/$31.00 ©2012 Crown

steering direction that acts to move the current and target images towards alignment. With a technique for transitioning between milestone images also detailed, the method provides a simple visual homing approach that preliminary results suggest may have general utility.

The remaining sections of this paper are as follows. In Section II a general overview of the visual homing algorithm is presented. Section III then details our particular implementation. Experimental results from a preliminary trial using a mobile robot are presented in Section IV. Finally conclusions and directions for future work are given in Section V.

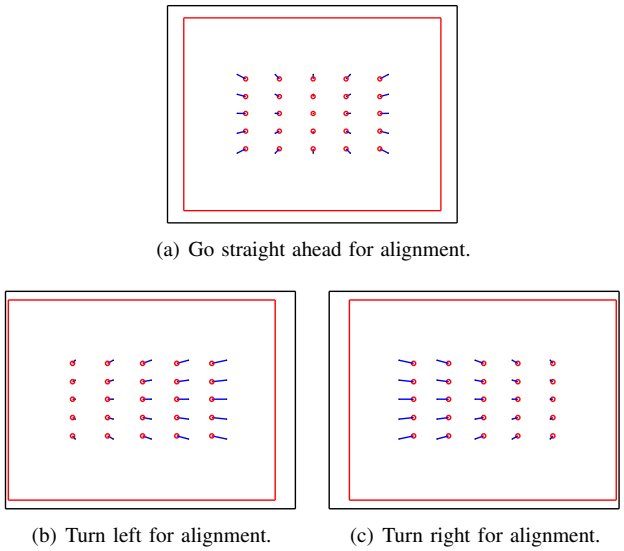
II. A VISUAL HOMING ALGORITHM

For the purposes of developing the visual homing algorithm we have chosen to implement route following, although the results should be more generally applicable to other navigation applications. For route-following, the teach-replay approach is followed where a robot acquires images as it is manually driven along a path (teaching phase), and then placed back at the start location and tasked to repeat the path autonomously (replay phase). During the replay phase, the robot uses its current view of the world and a reference image (automatically selected from the previously recorded images in the teaching phase) to determine its angular velocity. Typically, and in this work, a constant forward speed is used. Visual homing operates during the replay phase.

The first component of a visual homing algorithm is the selection of appropriate reference images during replay as the robot retraces the previously taught route. It is assumed that the robot is placed at the same starting location as in the teaching phase. Hence, the first image stored during the teaching phase can be used as the first reference milestone image. As the robot moves, the scene will change and another reference image will become more appropriate for the robot to align with. Various techniques have been reported in the literature for deciding when to transition between reference images (e.g. see [2], [13]). Since we are interested in developing an all-vision solution to visual homing, we decided not to use odometry based techniques (e.g. as in [13]), but to use an image similarity based technique somewhat similar to that described in [21]. Whilst the technique is not perfect, it is sufficient to test the primary focus of our algorithm described next.

The second component of a visual homing algorithm is to use the current and reference views to select an angular velocity. For our visual homing algorithm, we select an angular velocity that attempts to align the current view with the reference view. To do this, consider the concept of flow vectors computed using the current and reference images. These flow vectors indicate the directions and distances points in the current image are displaced from the corresponding points in the reference image. Figure 1 shows the flow vectors between a grid of points in two synthetic views for a few different alignment scenarios, where a target view is in front of the current view. Here we assume a flow vector is directed from a point in a test image to a corresponding point in a reference image.

Assuming a uniform sampling of points for which flow vectors are computed, when the two images are aligned (i.e. the vanishing points are aligned) the proportion of flow vectors pointing to the left will be the same as the proportion of vectors pointing to the right as shown in Figure 1(a). However, when there is a misalignment, there will be asymmetry such that the proportion of vectors pointing left will differ from the proportion pointing right as shown in Figures 1(b) and 1(c). This suggests a simple technique for determining orientation changes: *move so as to balance the proportions of vectors pointing left and right*.



(a) Go straight ahead for alignment.

(b) Turn left for alignment.

(c) Turn right for alignment.

Fig. 1. Example scenarios each showing a test view (black) and a target reference view (red) and the flow vectors between the two views for a grid of 5×5 points. The target view is offset to the left in (b) and to the right in (c). In (a), the flow vectors pointing left and right are balanced indicating that no adjustment in orientation need be made. Note, we assume vectors are directed from a point in the test view to a point in the reference view.

For example, in Figure 1(b), where the desired image is offset to the left in the current view, the proportion of left pointing flow vectors is greater than the proportion of right pointing vectors. In this situation the robot should be given an angular velocity such that it turns left and tries to align the images. Similarly, in Figure 1(c), where the desired image is offset to the right, and the proportion of right pointing flow vectors dominate, the robot should turn right to try and align the images. When the proportions are balanced such that the vanishing points are aligned (refer to Figure 1(a)), the robot can drive straight ahead.

As mentioned, this technique requires a uniform sampling of points in the image so that when the images are aligned the proportions of left and right flow vectors are in balance. It also requires that the reference and test images to be sufficiently similar such that flow vectors can be computed. Determining the proportions amounts to counting direction votes from the flow vectors where only the sign of the x-axis component of the flow vectors need be considered. That is, it is not necessary to compute the full magnitude and direction of the vectors, only the sign of the horizontal component is

needed. When the desired direction (left, right, or straight) has been determined, an appropriate angular velocity can be selected. As will be seen, a simple bang-bang like control strategy is sufficient for selecting the angular velocity from a desired direction.

The implementation of the visual homing algorithm on a mobile robot will be detailed in the following sections.

III. IMPLEMENTATION

A. System Overview

In this section we provide an overview of the system that was developed and used in the experiment reported.

The mobile robot used was a Pioneer P3-AT equipped with a firewire camera (Point Grey Research Dragonfly IEEE-1394), mounted in a forwards-facing direction on the chassis, offset from the centre[5] as shown in Figure 2.

The robot had an in-built Linux machine and was running a Player (v1.6.5) server[6] which provided control over the robot's motors. Existing code, developed previously within the robotics group, that allowed greyscale images to be acquired from the camera was adapted for our usage. Images of resolution 320 pixels $\times$ 240 pixels were used for the algorithm. The high-level visual homing algorithm was programmed in Matlab (v2010a) which ran under Ubuntu (v11.04) on a laptop. The laptop was connected to the robot's in-built Linux machine via ethernet. Custom made C/C++ TCP/IP server-client programs provided an interface between the Matlab code and the Player server and camera routines. External video footage was captured with a web-cam.


;

Fig. 2. A photograph showing the Pioneer robot used in experiments operating autonomously under laptop control. The forward facing monocular camera can be seen mounted on the front right-hand side of the robot.

B. Teaching Phase

For the teaching phase, a Matlab script was written that allowed the robot to be manually driven under keyboard control along a path. As the robot moved, images were acquired sequentially at an average rate of $\sim$1.35Hz.

When in motion, a constant forward speed of $V_f = 0.3$ m/s was set for the robot, with the manual control allowing the robot's direction to be adjusted through modification of the angular speed, V_a. The robot could thus go straight ($V_a = 0$ rad/s), veer left ($V_a = 0.2$ rad/s) or veer right ($V_a = -0.2$ rad/s).

C. Replay Phase

For the robot to autonomously retrace the route in the replay phase, the visual homing algorithm detailed in Section II was implemented in Matlab. As mentioned earlier, there are two components: selection of an appropriate reference image, and the selection of the angular velocity. We detail our implementation of each of these components in the following two subsections.

1) Reference Image Selection: To determine the reference image that the robot should try and align with, the similarities between the current test image, I_{test}, and the milestone images $\{I_i, I_{i+1}, I_{i+2}\}$ (where I_i is the current reference image) recorded during teaching were computed in each iteration using a mutual information similarity metric. The milestone image that was the most similar to the current test image was taken as the new reference image, provided it was sufficiently better ($> \epsilon = 0.1$) than the others.

Various feature-based techniques have been established for computing image similarity. However, feature extraction can be computationally expensive and requires features to be present in the image. Mutual information "describes the amount of information that one random variable, ... , gives about a second random variable" [22]. We have chosen to compute the mutual information using available routines [23][7] between pairs of images as a measure of similarity (also see [24]). In our work, we are computing the mutual information between two images, and this provides a relative measure of how similar two images are[8]. Alternatively, correlation methods could potentially be used (see [24]). Since we are determining which milestone image is most similar to the current image, a relative measure of similarity (as provided by the computation of the mutual information) is sufficient.

2) Angular Velocity Control: During the replay phase, the robot tries to align its current image with the selected reference image by determining the sign of the horizontal component of flow vectors as described in Section II. In our implementation, the sign is computed for 25 points (in a 5×5 grid)[9]. To do this, for each point, a patch of 51×51 pixels centred on the point are extracted from the reference image (see Figure 3). Then, a moving window region of the same size (51×51 pixels) is taken from the current image, in a location horizontally offset (by δ) from the reference point, and the mutual information between the two regions is computed[10]. The window is then moved one pixel to the

[5]An existing mounting arrangement was used, although mounting the camera centrally may have been a preferable arrangement.

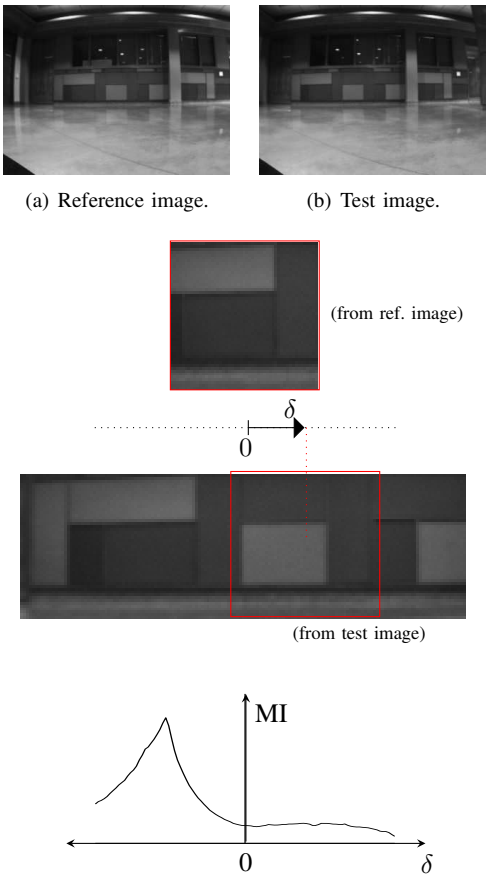
[6]http://playerstage.sourceforge.net/

[7]The Mutual Information computation package by Hanchuan Peng is available from the Matlab Central file exchange: http://www.mathworks.com/matlabcentral/fileexchange/14888

[8]Here we have used greyscale images with 256 levels, however, it is common to reduce the number of levels before computing the mutual information. Later, where we use mutual information for another purpose, the number of levels is reduced.

[9]It should be noted that it is not necessary to compute the x-y components of each flow vector, only the sign of the x-axis component is computed.

[10]Note, first the number of greyscale levels for each image is first reduced from 256 to 11 before the mutual information computations are performed.



(a) Reference image. (b) Test image.

(c) Mutual information (MI).

Fig. 3. This figure illustrates, through an example, how the sign of the horizontal component of a flow vector for a point is computed. A reference patch, shown in (c), centred on a point of interest is first taken from the reference image (a). A test patch is then extracted (using a moving window) from the test image (b) that is offset from the reference patch by δ, and the mutual information between the two patches is computed. This process is repeated for a range of δ. A plot of the mutual information as a function of δ is shown in the example. A peak is evident for the offset where the reference and test regions are considered aligned, and in this case the sign of the horizontal vector component is negative.

right and a new region in the current image extracted and the mutual information computed. This process is carried out for offsets in the range $-51 \leq \delta \leq 51$.

When the two regions are aligned, the mutual information would be expected to peak for an offset of $\delta = 0$. If the mutual information peaks for a positive offset, then the flow vector is interpreted as pointing to the right (positive sign). Whereas, when the mutual information peaks for a negative offset, the flow vector is taken as pointing to the left (negative sign). For the example in Figure 3 the test region is misaligned with the reference region. Here, the test region needs to move right (i.e. the robot needs to veer left) in order for it to align with the reference region, so the horizontal component of the flow vector is negative and the mutual information peaks for a negative offset as might be expected. To allow sufficient room for regions of varying offset to be extracted for each point, the points in the 5×5 grid were positioned at the row indices $\{77, 99, 121, 142, 164\}$ and column indices $\{77, 119, 161, 202, 244\}$.

When the sign of the horizontal component of the flow vector has been computed for each point, it contributes a vote for the direction to be taken (namely, left, right or straight ahead). The votes are then tallied. If the proportion of votes for left or right exceeds a threshold, set as 0.6, then that direction is taken, otherwise the robot moves straight[11].

A simple bang-bang like control scheme proved sufficient for our preliminary trials. Using this control scheme, three angular velocities were switched between in accordance with the overall direction decision. The velocities chosen were half those used during the teaching phase. Namely, a constant forward speed of $V_f = 0.15$ m/s was used and the angular speed was determined by the chosen direction with (i) $V_a = 0$ rad/s for straight, (ii) $V_a = 0.1$ rad/s for veering left, and (iii) $V_a = -0.1$ rad/s for veering right. The reason for scaling the speeds to lower values was to prevent the robot moving too far between updates which are slowed in the replay phase due to computation increases. Additionally, more updates over a given distance should result in better route following. The average update rate (including image capture) during replay was $\sim$0.97Hz.

IV. RESULTS

In this section we detail the results of an experiment in which the robot autonomously completed an approximate[12] loop, inside a building, after being taught the path under manual control. For the teaching phase, a total of 226 images were acquired along the route and during the replay 578 images were captured. The route traversal time from start to finish was $\sim$597s during replay compared to $\sim$167s during teaching (i.e. approximately 3.6 times slower).

In the absence of an accurate ground truth, odometry was recorded for the purposes of visualisation, but note that it was not used by the visual homing algorithm. Figure 4 shows a plot of the odometry readings during the teaching and replay phases. A selection of images acquired during replay are shown along with the reference image used in each case.

During replay, the robot was programmed to stop upon transitioning to the last milestone image. The visual homing algorithm successfully allowed the robot to complete the loop and automatically stop within close proximity to the expected position, as is evidenced by the final pair of images in Figure 4 which can be seen are very similar.

A clearer example is given in Figure 5, where the robot can be seen just after the start of the trial as it begins to navigate the first corner of the route. The reference milestone image that was selected and also the current view, with the direction votes for the 25 sampling points indicated with arrows, are shown in Figures 5(a) and 5(b) respectively. In this particular example, the majority of votes are for the right, so a velocity

[11] A lower value (above 0.5) could likely improve performance – see Section IV.

[12] The robot's final position during teaching was close to its start location in teaching but was not exactly the same because of the difficulty in performing precise steering control manually.

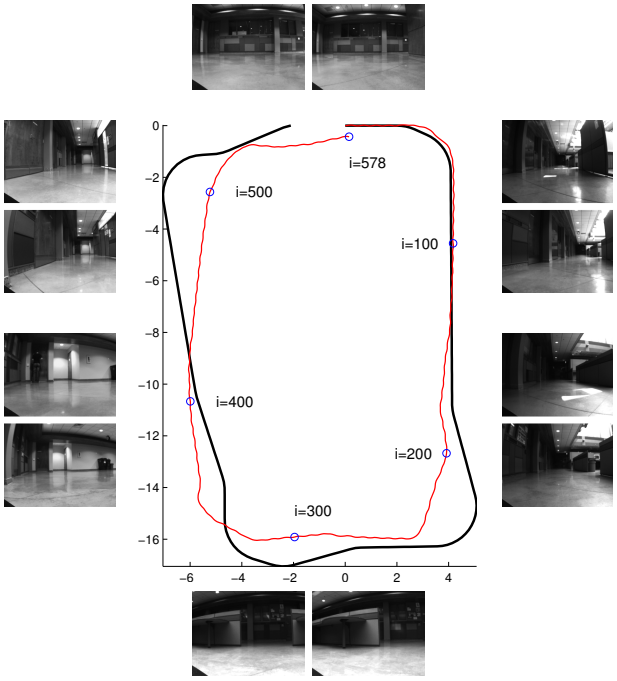


Fig. 4. Trajectories recorded from odometry during the teaching (black line) and replay (red line) phases used for visualisation purposes only (Note, odometry errors have not been corrected for). A selection of images ($\{100, 200, 300, 400, 500, 578\}$) from the replay phase are shown along with the target milestone image shown above or to the left. Units are in metres.

command to steer the robot right was chosen. The mutual information curves for each of the 25 patches are plotted in Figure 5(d). It can be seen that the majority of curves have a peak offset to the right – not centrally placed, as would be the case when the images are aligned.

With the current parameter settings and setup, the robot was observed to be over correcting its heading in both left and right directions, rather than driving mostly straight with just small heading corrections. In some instances, the robot turned too far in one direction so that the patches only minimally overlapped. In these cases, the peaks in the mutual information curves (like in Figure 5(d)) moved outside the domain of offsets considered. Anecdotally, during its journey the robot seemed to move away from the desired path on occasion (coming close to a side wall in one instance), but managed to correct itself sufficiently to allow it to reach its goal. Other trials suggest that if the alignment is displaced too far, so that patches no longer overlap, the robot can 'fall-off track'.

It is worth noting that the algorithm did display some robustness to discrepancies in the scene between the teaching and replay phases, such as the appearance/disappearance of people and objects. However, additional tests are needed to further investigate the robustness of the algorithm in more dynamically changing environments. Improvements could include the addition of obstacle avoidance and a search strategy for finding the path when lost.

To further increase the resilience of the algorithm a number of improvements could be made: (i) the camera could be centrally mounted (to remove any steering bias), (ii) the number and size of patches used could be optimised, (iii) the threshold (currently set at 0.6) for the proportion of votes needed to veer left or right could be reduced (so that a decision to turn is made sooner), (iv) the velocity update rate could be increased, and (v) the robot velocities (relative to those used during the teaching phase) reduced.

V. CONCLUSION AND FUTURE WORK

The visual homing algorithm presented in this paper is relatively simple to implement, requires only a single monocular camera with a relatively small field of view and does not rely on odometry. The results are only preliminary and further testing is needed, but it is clear from the experimental trial reported that the algorithm has the capacity to allow a robot to autonomously follow a previously recorded route using only vision.

The work raises a number of interesting directions for future work. The focus of the paper has been on route following, however, the techniques developed may also have some relevance for robot navigation in appearance-based SLAM (e.g. see [25], [26]). Currently the reference images are acquired densely along a route. The capacity for the algorithm to operate on relatively fewer images needs investigation. Such a scenario may require the size of the regions used for direction determination to be increased so as to ensure some overlap with the target view. Ideally the algorithm would operate with just a small set of representative views of the environment as in the vacation snap shot problem [27].

A further avenue of research would be to conduct extensive trials on uneven terrain to see if any modifications to the algorithm are needed. It is also unclear how the algorithm will perform when the mutual information between the patches is very low, as might be seen in very structured or empty environments with untextured surfaces. The effect of close range objects also needs further investigation and additional scenarios (to those depicted in Figure 1) may need to be considered.

It would also be useful to undertake a convergence/stability analysis for the system, along the lines of the work done in [2], to determine the conditions for which path-following is guaranteed. Additionally, or in conjunction with a convergence analysis, the development of some metrics for assessing the visual homing algorithm would be of great utility. Depending on the emphasis of the application, metrics could be developed to assess (i) the *total variation* incurred along a path between two locations, (ii) maximum displacement from the desired path, (iii) the displacement from the goal upon nominal arrival (given a displacement from the starting location), (iv) difference between desired and actual headings along the route and at the goal, (v) traversal time between locations, and (vi) success rates for reaching a goal or milestone locations. Some of these metrics would require a ground truth to be established (e.g. using a technique such as the ceiling tracker in [28]) so that the

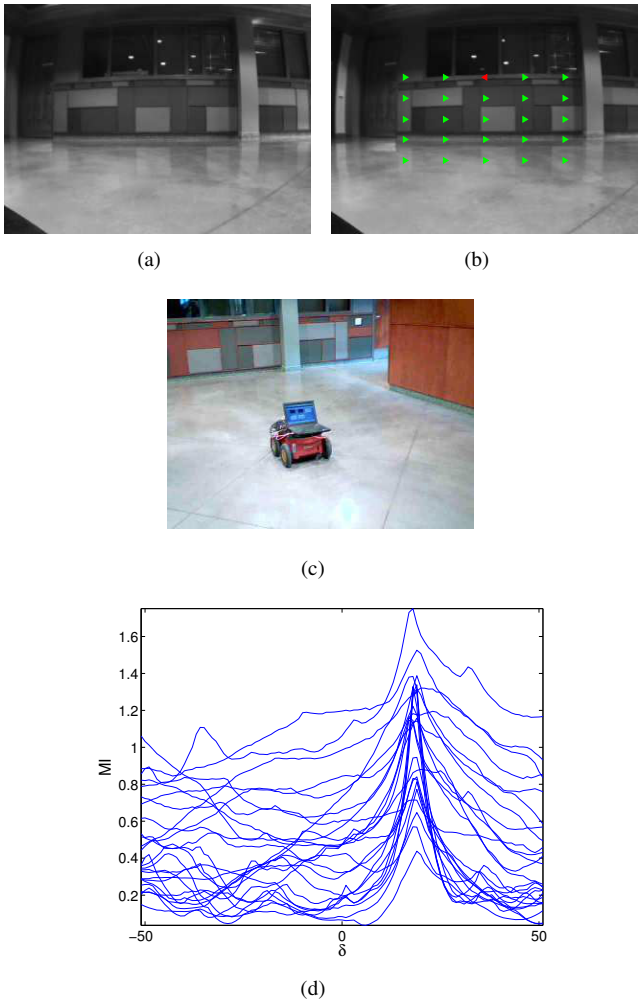


Fig. 5. A step along the replay route showing: (a) the reference image, (b) the current image with direction votes indicated for a grid of sampling points, (c) an external view of the robot soon to turn a corner, and (d) the mutual information for the sampling points.

trajectories and poses during teaching and replay can be quantitatively compared.

ACKNOWLEDGMENT

The authors would like to gratefully acknowledge partial funding of this work by NSERC. The assistance provided by support staff at the University of Alberta and useful discussions regarding the work with family and colleagues has also been most appreciated.

REFERENCES

[1] O. Booij, B. Terwijn, Z. Zivkovic, and B. Kröse, "Navigation using an appearance based topological map," in *Proceedings of 2007 IEEE International Conference on Robotics and Automation (ICRA'07)*, 2007.

[2] A. M. Zhang and L. Kleeman, "Robust appearance based visual route following for navigation in large-scale outdoor environments," *The International Journal of Robotics Research*, vol. 28, pp. 331–356, 2009.

[3] B. A. Cartwright and T. S. Collett, "Landmark maps for honeybees," *Biological Cybernetics*, vol. 57, pp. 85–93, 1987.

[4] R. Möller, M. Kryzkawski, and L. Gerstmayr, "Three 2D-warping schemes for visual robot navigation," *Autonomous Robots*, vol. 29, pp. 253–291, 2010.

[5] R. Basri, E. Rivlin, and I. Shimshoni, "Visual homing: Surfing on the epipoles," *International Journal of Computer Vision*, vol. 33, pp. 117–137, 1999.

[6] G. López-Nicolás, C. Sagüés, J. Guerrero, D. Kragic, and P. Jensfelt, "Switching visual control based on epipoles for mobile robots," *Robotics and Autonomous Systems*, vol. 56, pp. 592–603, 2008.

[7] G. López-Nicolás, J. Guerrero, and C. Sagüés, "Multiple homographies with omnidirectional vision for robot homing," *Robotics and Autonomous Systems*, vol. 58, pp. 773–783, 2010.

[8] H. Becerra, G. López-Nicolás, and C. Sagüés, "Omnidirectional visual control of mobile robots based on the 1D trifocal tensor," *Robotics and Autonomous Systems*, vol. 58, pp. 796–808, 2010.

[9] G. López-Nicolás, J. Guerrero, and C. Sagüés, "Visual control through the trifocal tensor for nonholonomic robots," *Robotics and Autonomous Systems*, vol. 58, pp. 216–226, 2010.

[10] A. Dame and E. Marchand, "A new information theoretic approach for appearance-based navigation of non-holonomic vehicle," in *Proceedings of 2011 IEEE International Conference on Robotics and Automation (ICRA'11)*, 2011.

[11] F. Chaumette and S. Hutchinson, "Visual servo control part I: Basic approaches," *IEEE Robotics and Automation Magazine*, vol. 13, pp. 82–90, 2006.

[12] Z. Chen and S. T. Birchfield, "Qualitative vision-based mobile robot navigation," in *Proceedings of 2006 IEEE International Conference on Robotics and Automation (ICRA'06)*, 2006.

[13] ——, "Qualitative vision-based path following," *IEEE Transactions on Robotics*, vol. 25, pp. 749–754, 2009.

[14] R. Möller, A. Vardy, and S. Kreft, "Visual homing in environments with anisotropic landmark distribution," *Autonomous Robots*, vol. 23, pp. 231–245, 2007.

[15] R. Biegler, "Functional considerations in animal navigation: How do you use what you know?" in *Animal Spatial Cognition: Comparative, Neural and Computational Approaches*, M. F. Brown and R. G. Cook, Eds., 2006.

[16] S. C. Diamantas, "Biological and metric maps applied to robot homing," Ph.D. dissertation, School of Electronics and Computer Science, University of Southampton, United Kingdom, 2010.

[17] R. Möller, D. Lambrinos, R. Pfeifer, and R. Wehner, "Insect strategies of visual homing in mobile robots," in *Proceedings of the Computer Vision and Mobile Robotics Workshop (CVMR'98)*, 1998.

[18] A. Rizzi, D. Duina, S. Inelli, and R. Cassinis, "A novel visual landmark matching for a biologically inspired homing," *Pattern Recognition Letters*, vol. 22, pp. 1371–1378, 2001.

[19] L. Smith, A. Philippides, P. Graham, B. Baddeley, and P. Husbands, "Linked local navigation for visual route guidance," *Adaptive Behavior*, vol. 15, pp. 257–271, 2007.

[20] B. Baddeley, P. Graham, A. Philippides, and P. Husbands, "Holistic visual encoding of ant-like routes: Navigation without waypoints," *Adaptive Behavior*, vol. 19, pp. 3–15, 2011.

[21] Y. Matsumoto, K. Sakai, M. Inaba, and H. Inoue, "View-based approach to robot navigation," in *Proceedings of 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'00)*.

[22] R. E. Blahut, *Principles and Practice of Information Theory*. Reading, MA: Addison-Wesley Publishing Company, 1987.

[23] H. Peng, F. Long, and C. Ding, "Feature selection based on mutual information: criteria of max-dependency, max-relevance, and min-redundancy," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, pp. 1226–1238, 2005.

[24] V. S. Roshni and K. Revathy, "Using mutual information and cross correlation as metrics for registration of images," *Journal of Theoretical and Applied Information Theory*, vol. 4, pp. 474–481, 2008.

[25] A. Angeli, S. Doncieux, J.-A. Meyer, and D. Filliat, "Incremental vision-based topological SLAM," in *Proceedings of 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'08)*.

[26] M. Cummins and P. Newman, "Highly scalable appearance-only SLAM – FAB-MAP 2.0," in *Proceedings of 2009 Robotics: Science and Systems Conference*, 2009.

[27] E. Bourque and G. Dudek, "On the automated construction of image-based maps," *Autonomous Robots*, vol. 8, pp. 173–190, 2000.

[28] J. Klippenstein and H. Zhang, "Performance evaluation of visual SLAM using several feature extractors," in *Proceedings of 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'09)*.