Robotica: (2020) volume 38, pp. 787–803. © Cambridge University Press 2019
doi:10.1017/S0263574719001061

Evaluation of Field of View Width in Stereo-vision-Based Visual Homing

D. M. Lyons*, B. Barriage and L. Del Signore

Robotics and Computer Vision Lab, Fordham University, Bronx, NY 10458, USA.
E-mails: bbarriage@fordham.edu, ldelsignore@fordham.edu

(Accepted June 1, 2019. First published online: July 3, 2019)

SUMMARY
Visual homing is a local navigation technique used to direct a robot to a previously seen location by comparing the image of the original location with the current visual image. Prior work has shown that exploiting depth cues such as image scale or stereo-depth in homing leads to improved homing performance. While it is not unusual to use a panoramic field of view (FOV) camera in visual homing, it is unusual to have a panoramic FOV stereo-camera. So, while the availability of stereo-depth information may improve performance, the concomitant-restricted FOV may be a detriment to performance, unless specialized stereo hardware is used. In this paper, we present an investigation of the effect on homing performance of varying the FOV widths in a stereo-vision-based visual homing algorithm using a common stereo-camera. We have collected six stereo-vision homing databases – three indoor and three outdoor. Based on over 350,000 homing trials, we show that while a larger FOV yields performance improvements for larger homing offset angles, the relative improvement falls off with increasing FOVs, and in fact decreases for the widest FOV tested. We conduct additional experiments to identify the cause of this fall-off in performance, which we term the 'blinder' effect, and which we predict should affect other correspondence-based visual homing algorithms.

KEYWORDS: Visual homing; RGB-D; Autonomous robots; Computer vision; Mobile robots.

1. Introduction

Modern applications for robots demand the ability to deal with unstructured and unknown terrain, such as exploration on planetary surfaces, disaster sites, and underground tunnels.[1] Navigation is challenging for an autonomous robot operating in an unstructured environment, and several approaches to this problem have been investigated. These include the generation of metric maps using algorithms, such as SLAM[2] followed by explicit path-planning,[3] as well as the construction of a qualitative, topological map[4] and the associated techniques for navigating locally to and from map nodes. Visual homing is a navigation technique often used in visual navigation with a topological map.[3] It does not require the use or construction of a metric map or a 3D reconstruction but instead uses information calculated from comparing a goal, 'home' visual image with an image taken at the robot's current location to generate motion commands that navigate the robot from its current location to the location at which the home image was taken. Visual homing sidesteps the memory requirement and sensor uncertainty issues associated with maintaining a large area map. The burgeoning area of service robots is another application area for visual homing, where robots need to operate in unstructured terrain.[5]

It has been shown that visual homing performs better when depth-related cues are included,[6] for example, using a stereo-camera.[7] It is typical to assume a panoramic field of view (FOV) for visual homing.[8] While panoramic stereo-camera technology is available,[9,10] it is a specialized technology, and the typical stereo-cameras (and more recently RGB-D cameras) used in robotics have a

* Corresponding author. E-mail: dlyons@fordham.edu

https://doi.org/10.1017/S0263574719001061 Published online by Cambridge University Press

limited FOV. In this paper, we present an evaluation of various FOV widths on the performance of a stereo-vision-based homing algorithm using a common stereo-camera for a variety of visual homing missions in both indoor and outdoor situations totaling over 350,000 visual homing trials. The results indicate that performance improvement is seen for increasing FOVs from the narrowest tested (66°) up to 210°. However, for FOV higher than this, we do not see much performance improvement, and in fact for the widest FOV tested (354°), there is a small but consistent *drop in performance*. We refer to this phenomenon as the 'blinker' effect and we show how it is a consequence of the trade-off in the advantage of wider FOVs for handling larger homing offset angles against the relative quality of feature matching in larger images. We predict that this phenomenon will be evident in any correspondence-based homing approach where the FOV is increased without a reduction in resolution. The homing trials were conducted on six stereo-vision visual homing databases which we have collected and which we make available to the research community. The interested researchers will find the details of the databases, such as how to access the databases and how to use the data for the homing experiments in Section 4, is given in Appendix I.

We briefly review the state of the art in visual homing in Section 2 and explain where our contribution fits. It would be extremely challenging to conduct a sufficient quantity of experiments on a physical robot platform to study the effect of various sized FOVs. However, conducting experiments in a simulated world has the disadvantage that the sensory input to the algorithm, the visual imagery, is not real, and the resulting performance may not be a realistic representation of real robot behavior. One approach to addressing this problem is to store a database of visual imagery (e.g., ref. [11]) and to apply the algorithm to this database; this has the advantage that the input is real video, increasing confidence that the resulting performance is representative of real robot behavior, and this is the approach we take in this paper. Note that the visual homing database is not a part of the homing algorithm evaluated here; just a tool to evaluate its potential performance in realistic conditions. In Section 3, we describe the stereo-vision-based visual homing algorithm we use and explain how it is used with the stereo databases we have collected to conduct homing experiments. Section 4 presents the stereo databases we use for this study. In Section 5, our experimental procedure to investigate the effect of varying the FOV is presented, and the results of the investigation are presented in Section 6. Section 7 discusses these results and presents our conclusions and future directions.

2. Prior Literature

There are two main approaches to matching a home and a current image to determine a direction to move toward home: holistic and feature based. Common holistic approaches include the *warping method*, which warps the current image to look like the home image and calculates the movement vectors that correspond to these changes,[12,13] *parameter methods*, which compare symbolic descriptions of the visual structure of the current and goal images,[14] and *descent in image distance* methods that compare the current and goal images by *image distance*.[15] Unlike holistic methods, feature-based methods identify visual landmarks. Examples of these methods include *Average Landmark Vector* (*ALV*)[16,17] and related methods that calculate a unit motion vector by differencing the ALVs in home and current images and *correspondence methods*[6,13,18] which use features such as Scale-Invariant Feature Transform (SIFT) to identify the correspondence between current and home images to identify a motion vector. By identifying correspondences and calculating the corresponding angular differences, the need for a global compass in ALV-like methods[19] is eliminated. Homing approaches can be applied to short-range navigation (where the home location is in view but distant)[16,20] but also to long-range navigation (where the final home location is not in view) as a series of short-range navigations between places in a topological map.[1,8,21] Although many researchers just address the issue of determining homing motions in flat, uncluttered space, a more real-world homing in rough terrain and in cluttered space has been addressed in refs. [1, 8, 22] and in ref. [23]; we also integrate stereo-vision-based homing with navigation in rough and cluttered terrain.

Much of the existing work in visual homing for robots focuses on bearing-only homing, comparing a home image from an omnidirectional camera with a current image at each time step (e.g., ref. [19]). In such an approach, the robot moves some distance in the direction calculated from the bearing measurements. Some researchers[6,22] have also used the SIFT scale information not only to estimate bearing but also to travel distance. The advantage of using depth information is that it provides an estimate of how far to move in addition to which direction to move. It has been shown[7,24,25]

https://doi.org/10.1017/S0263574719001061 Published online by Cambridge University Press

Table I. Notations used in visual homing formulas.

Symbol	Meaning
$p = (x, y) \in R^2$	Position of the robot, in meters.
$\theta \in \{-\pi, \ldots, \pi\}$	Orientation of the robot, in radians.
$H = (I_{Hv}, I_{Hd})$, $C = (I_{Cv}, I_{Cd})$	Home (Current) image pair of color image and depth image
$I_{Hd}(u, v) = (x, y, z) \in R^3$	Depth image pixels, in meters
$I_{Hv}(u,v) = (r,g,b)$	Visual image color pixel
(Θ_a, Θ_d)	Homing vector pair of homing angle and homing distance
$h_{m,q}$ $c_{m,q}$ for $q \in \{x, y, z\}$	Matched home and current depth image pixel
$sx\ (h_m)$	Sign of the x value for any feature h_m
g_a, g_d	Gain constants: scaling homing angle and homing distance, respectively
A_{tol}, D_{tol}	Tolerance values for homing angle (in radians) and distance (in meters), respectively

that the performance of feature-based approaches can be improved by fusing visual features with additional sensory data from hardware, such as laser rangefinders. Nirmal and Lyons[7] modified the SIFT feature-based *HiSS* homing algorithm of ref. [6] to leverage depth information from a stereo-camera and showed that it improved algorithm performance while maintaining the purely visual nature of the sensing used for homing. Nirmal and Lyons used four performance measurements – angular error, positional error, number of steps, and return ratio – to evaluate the performance of HSV on a series of homing trials using a Pioneer 3AT robot. They favorably compare HSV performance to that of the *HiSS* algorithm,[6] an algorithm that uses SIFT scale information to approximate depth. They also present a convergence proof for the HSV algorithm under the assumptions of error-free SIFT matching and commanded robot motion.

A panoramic FOV camera is commonly assumed in visual homing; Argyros et al.[8] cited advantages for this including: no need for elaborate gaze control, less susceptibility to dynamic scene elements, and a longer life-cycle for tracked features. Technology exists to construct panoramic stereo-vision images, including a system of cameras, a single camera moving along a curve, and specialized mirrors or lens (see Table I of ref. [9]). However, multiple camera solutions require specialized calibrations, and moving camera solutions typically require a static scene. Aggarwal et al.'s solution[9] comes closest to a 'drop in' replacement for a typical panoramic camera as used in visual homing, though it requires a carefully constructed 'coffee-filter' mirror. However, many existing robot systems have common, non-panoramic stereo-cameras, or more recently RGB-D cameras, and it would be valuable to understand how these existing systems can be used effectively in visual homing.

All the stereovision homing experiments in ref. [7] were conducted with the commonly available Bumblebee2 stereo-camera with 66° FOV. Wide FOV stereo images were obtained by concatenating (not stitching) multiple visual and depth images. We note that a limited FOV camera may have the advantage of providing higher resolution for the portion of the scene it views than a panoramic camera. Since the concatenation method effectively allows the FOV to be varied during homing trials by including more or less panned images in the composite image, it begs the question of what value various widths of horizontal FOV have on homing performance.

It is challenging to conduct a large number of visual homing experiments on a physical robot, and prior work in this field has included a simulation approach using databases of visual panoramas collected *a priori*.[11] In addition to these monocular panoramic databases, there are other stereo databases for scene recognition as well as navigation. However, our requirements that we are able to simulate many different FOVs at every location rule these out. Most recently, Ammirato et al.[26] have assembled a large database of Kinect RGB-D data of indoor scenes primarily for object recognition. Any RGB-D sensor, as opposed to a stereo-camera, could be used with the homing algorithm we describe in this paper. However, the Kinect is not, in general, a good sensor for RGB-D visual homing due to its limited depth of field. While the database could be repurposed for homing experiments, the performance data would not then be realistic in that case, eliminating the value of any results.

To allow us to complete the study in this paper, therefore, we have collected six databases for the Bumblebee stereo-camera which allow us to address the FOV question. The databases we have

https://doi.org/10.1017/S0263574719001061 Published online by Cambridge University Press

collected, though small, are unique in collecting 3D point data from the Bumblebee stereo-camera for a wide range of orientations at each point, in outdoor as well as indoor locations, and at two spatial resolutions. The scenes are not staged and occasionally capture random pedestrian traffic and weather-related light changes. We make the databases freely available to other researchers – see Section 4 for details of the databases and how to use them.

3. Background: Visual Homing Using Feature Depth Information

In this section, we present the visual homing algorithm used in our investigation. The algorithm is a correspondence-based visual homing algorithm that uses feature depth information from a stereo-vision system to calculate both a home direction and a home distance. The algorithm is based on Nirmal and Lyons's 'Homing with Stereovision (HSV)'[7] which is, in turn, a modification of Churchill and Vardy's 'Homing in Scale Space (*HiSS*)'[6] to use stereo-depth instead of scale. That algorithm is reviewed here along with the extensions necessary to support variable FOV; the modified algorithm is called *Homing Using Depth* (*HUD*) to emphasize the fact that it can be used with any RGB-D input (not just stereo-vision). Note that the HUD algorithm, like the HSV and *HiSS* algorithms that inspired it, is designed to take live video imagery as an input and is not dependent on the visual homing database for performance – the database is just a tool which will be substituted for live video and then used to evaluate homing performance.

3.1. Homing using depth (HUD)

Let the position of the robot be $p = (x, y) \in R^2$ in meters, and its orientation be $\theta \in \{-\pi, \ldots, \pi\}$ in radians. The robot takes a *home image H* at the (goal) home pose (p_g, θ_g), and this image is stored. A stereo-vision system generates an image that includes both visual image information and stereo disparity information for each pixel in that image. We will assume that the disparity information is processed using the camera calibration to generate two aligned images: $H = (I_{Hv}, I_{Hd})$, where the visual pixel $I_{Hv}(u,v)$ is from a point in the scene with coordinates in a robot-centered reference frame R, $I_{Hd}(u, v) = (x, y, z) \in R^3$ in meters and where $(u,v) \in \{0,\ldots, \mathrm{N}_u - 1\} \times \{0,\ldots, \mathrm{N}_v - 1\}$, in units of pixel row and column, for image size $N_u \times N_v$; we will refer to the combination as the RGB-D image.

A homing mission begins by moving the robot to an arbitrary start position p_s. At this, and each successive position in the homing mission, the robot takes a current RGB-D image C. The current and home images are compared, and from that comparison a *homing vector* (Θ_a, Θ_d) is calculated as follows:

$$(\Theta_a, \Theta_d) = \mathrm{compare}(C, H) \tag{1}$$

This vector describes how much the robot must be rotated Θ_a in radians and then translated Θ_d in meters, relative to its current location, to reach the home pose (p_g, θ_g). The *compare* function extracts SIFT features from the C and H visual images and produces a list of matched image feature locations

$$M = \{(c_m, h_m) | m = 0, \ldots N \text{ and } c_m, h_m \in \{0 \ldots N_u - 1, 0 \ldots N_v - 1\}\} \tag{2}$$

Here, c_m is the image coordinates, in units of pixel row and column, of the SIFT feature in the current visual image, I_{Cv}, and h_m is the image coordinates, in units of pixel row and column, in the home visual image, I_{Hv}, for each matched SIFT feature m. Recall that the depth images and visual images are aligned so that the visual pixel in the home image $I_{Hv}(c_m)$ is from a point in the scene with robot-centered coordinates $I_{Hd}(c_m) = (x, y, z)$, measured in meters, for each matched feature $m = 0, \ldots N$.

We will use $c_{m,q}$ for $q \in \{x, y, z\}$ to refer to the q coordinate of matched feature m from the current depth image, and $h_{m,q}$ is defined similarly for the home image. The rotation component of the homing vector is calculated as the average angle difference in radians between current and home depth images for all feature points in the xy plane

$$\Theta_a = \frac{1}{|M|} \sum_{m \in M} \left(\tan^{-1} \frac{h_{m,y}}{h_{m,x}} - \tan^{-1} \frac{c_{m,y}}{c_{m,x}} \right) \tag{3}$$

https://doi.org/10.1017/S0263574719001061 Published online by Cambridge University Press

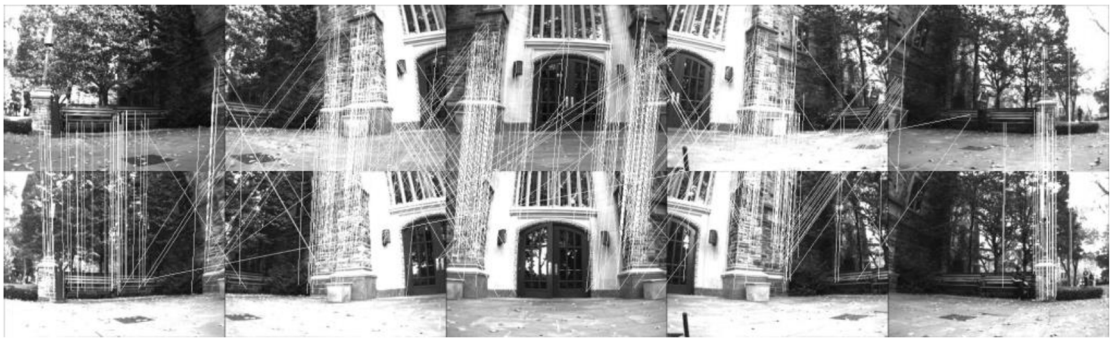


Fig. 1. Wide field of view SIFT matching of home image (top) and current image (bottom) with lines between matched features (G14 database).

Nirmal and Lyons[7] calculated the distance component of the homing vector as the average difference of x coordinates. This produces small motions when features are at the edge of the FOV, so instead we use the difference in distances to the feature in home and current image, and where $sx(.)$ is the sign function on the x axis

$$\Theta_d = \frac{1}{|M|} \sum_{m \in M} sx(h_m)\ (|h_m| - |c_m|)$$

$$sx\ (h_m) = \begin{cases} +1 & \text{if } h_{m,x} \geq 0 \\ -1 & \text{else} \end{cases} \tag{4}$$

The final step in homing is to rotate and translate the robot by the homing vector. To account for potential inaccuracies in the feature matches and stereo-calculations, the robot is rotated by $g_a\Theta_a$ and then translated $g_d\Theta_d$, relative to its current location. The g_a, g_d dimension-less constants in the algorithm are motion gains in the half open interval (0,1]. Homing is considered to have terminated when $|\Theta_a| \leq A_{\text{tol}}$ and $|\Theta_d| \leq D_{\text{tol}}$, where the A_{tol}, D_{tol} constants are the desired angular (in radians) and position (in meters) error tolerance parameters.

3.2. FOV

Nirmal and Lyons proposed that each home and current image was composed not of a *single* visual and *single* depth image from the stereo-camera, but by a *composite* image made by panning the stereo-camera across the scene. Each composite image was a sequence of five concatenated images: one directly in front of the robot, two images clock-wise, and two images counter-clock-wise from the front. The five RGB-D images were simply concatenated and not stitched or otherwise transformed, since according to Nirmal and Lyons this is faster and the image overlap improves the number of matches found. The height of the resultant image is the same as a single image, but the width is five times larger (i.e., image resolution is not reduced by the wide FOV). They calculate the FOV of the resultant composite visual and depth images as 270°.

Figure 1 shows an example visual image comparison. The top strip of five images is the composite home visual image. The bottom strip of five images is the composite current visual image. (They happen to be the same in this case; that is, the robot is at the home location.)

Recall that while visual homing using depth from stereo-vision has better performance than depth from scale,[7] which in turn has shown better performance than not using depth information at all.[6] However, restricting the FOV may arguably lead to a decrease in performance. The composite image approach of Nirmal and Lyons suggests a straightforward method to evaluate this: vary the number of panned images included in the composite image. The robot used in our visual homing research is a Pioneer 3-AT robot with a Bumblebee2 stereo-camera mounted on a Pan-Tilt (PT) base. The Bumblebee2 with 3.8 mm lens has a 66° horizontal FOV for each camera. A single image would therefore have a FOV of 66°; two images would increase the FOV to at most 132° and three would be at most 198°. By conducting the same homing missions but with different numbers of images in the composite image, the effect of the FOV on homing performance can be evaluated. (In fact,

https://doi.org/10.1017/S0263574719001061 Published online by Cambridge University Press

panned images were taken at 36° intervals, allowing for a small overlap between adjacent images, for robustness to small pan errors.)

If the stereo-camera can operate at f frames/section, then requiring a composite image of n frames slows the effective video rate to f/n frames/second. However, this delay is typically negligible compared to the mechanical pan time for the pan-tilt base: if the pan-tilt base operates at a constant v degrees/second, then the time t for a composite image of n frames for a camera that starts with a zero pan angle comprises $36(\frac{n-1}{2})v$ to move to the leftmost position, $36\ (n-1)\ (v+\tau)$ to move to each image position (with a delay of τ to allow the unit to settle), then $36(\frac{n-1}{2})v$ to move back to a zero pan angle, that is, in total the delay time is $t = 36(n-1)(2v+\tau)$. More generally, the time penalty is of order $O(n)$ in the number of images.

4. Experimental Method: Stereo-Vision Databases

It is challenging to conduct a large number of visual homing experiments on a physical robot: issues of battery life, weather, robot maintenance, and the regular needs of human researchers to eat and sleep limit the total number of physical experiments that can be performed. Thus, it is quite advantageous to have a realistic simulation capability on which to conduct extensive visual homing trials to evaluate performance under a wide range of conditions. A good simulation facility for visual homing requires realistic video imagery, and a realistic relationship between robot motion and (resultant) video imagery. In Section 4.2, we describe the stereo-vision database used in our FOV investigation.

An approach to building visual homing databases is described in ref. [11]: a spatial grid is superimposed on the area to be recorded, and panoramic visual imagery is collected at each grid cell. The information is used as follows. A robot begins a simulation run with a home position grid cell and a start position grid cell. The visual information the robot receives is the stored imagery for the grid cell it occupies. A motion vector is calculated by comparing this imagery to the home grid cell imagery. This motion may move the robot to another grid cell, and the imagery comparison and motion calculation continue.

4.1. Visual and depth imagery

The robot used to record the visual homing stereo databases is a Pioneer 3-AT robot with a Bumblebee2 stereo-camera mounted on a Pan-Tilt (PT) base and equipped with two 3.8 mm lens giving a 66° horizontal FOV for each camera. The PT base is leveraged to collect 10 visual and stereo-depth images at intervals of 36° apart, starting at 0° with respect to the robot X axis. The visual image stored is the left image from the stereo pair. The RH coordinate frame has the X axis along the direction in which the robot is facing, the Y pointing left, and is centered on the robot. The final angle requires the robot to be reoriented (in addition to the PT unit motion) due to PT unit pan limits.

Each visual image is a 1024×768 8-bit gray level image I_g, and each depth image I_d is a listing of the robot-centered coordinates for each point in the 1024×768 for which stereo disparity can be calculated. Figure 2 shows the visual and depth imagery from grid cells on two different databases. Figure 2(a) shows all 10 visual images for the first grid cell of the outdoor database G13. The robot would be in the center of the decagon with the X axis pointing straight up and the Y axis to the left. The images in the database are not keystone warped. Figure 2(b) shows the same information for the first cell in the outdoor database G14.

4.2. Database descriptions

Six databases have been collected and used for the performance study in this paper. The databases, described in Table II, are labeled G11 through G16 and include three indoor and three outdoor locations. Grid dimensions, resolutions, and spatial extent vary over the six databases: there are four 4×4 (with 16 cells) grids and two 7×7 (with 49 cells) grids; there are two 1.0-m resolution grids, and four 0.5-m resolution grids; there are two grids with a spatial extent of 4 m^2 grids, two with 9 m^2, and two with 16 m^2. In total, there are $10 \times 16 \times 4 + 10 \times 49 \times 2 = 1620$ visual images and 1620 depth files with a total of over 700 million depth points representing physical ranges greater than 50 m for the outdoor databases. Tables III and IV summarize the parameter settings for the robot, camera, and datasets collected.

https://doi.org/10.1017/S0263574719001061 Published online by Cambridge University Press

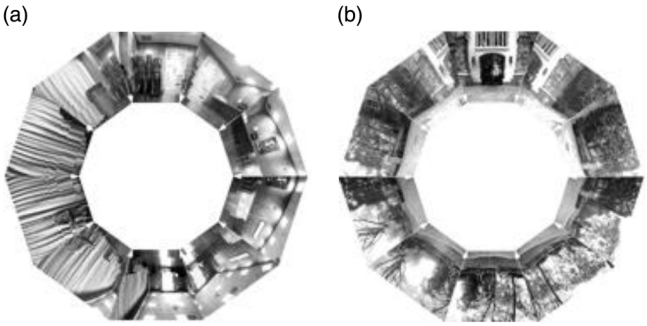


Fig. 2. (a) and (b) Set of visual images for a single square on the G15 and G14 databases.

Table II. Database location descriptions.

 **G11:** Grid 11 is an indoor location in a crowded robotics laboratory. The grid dimension is 4×4, and it spans 4 m^2.	 **G14:** Grid 14 is an outdoor location in front of a large building surrounded by landscaping on three sides. The grid dimension is 4×4 spanning 4 m^2.
 **G12:** Grid 12 is an outdoor location on a stone patio area in front of a large building. The grid dimension is 4×4, and it spans 16 m^2.	 **G15:** Grid 15 is an indoor location in a large, wood-paneled room. The grid dimension is 7×7 spanning 12.25 m^2.
 **G13:** Grid 13 is an indoor location on an upper floor lobby area of a large building. The grid dimension is 4×4, and it spans 16 m^2.	 **G16:** Grid 16 is an outdoor grassy courtyard between three buildings. The grid dimension is 7×7 spanning 12.25 m^2.

The database collection procedure is relatively straightforward and is presented in Appendix I, rather than here, to maintain the flow of the paper. Instead, we show two-point clouds composed from RGB-D information in databases G14 and G16 in Fig. 3 to illustrate the raw data collected. Figure 3(a) shows a point cloud constructed from the 160 individual depth images for the outdoor database G14. Each individual depth image was translated to a point cloud which was then translated and rotated to the location and orientation at which it was acquired. This registration is for display only*;* the algorithm does not need the stereo data to be registered and these point clouds are not used.

Statistical and voxel filtering was applied, but the data are otherwise as generated by the Bumblebee stereo-camera. Clearly, more processing could be applied, and the input data substantially improved. We chose to use the unprocessed stereo data for our experiments here and accept that our results will therefore be a lower bound on performance; improving the data should improve homing performance.

Figure 3(b) shows a point cloud constructed from the 490 individual depth images for the outdoor database G16 in a similar fashion. The grid of darker circles in the foreground of each image (a 4×4 grid in Fig. 3(a) and a 7×7 grid in Fig. 3(b)) shows the locations of the grid vertices, the locations at which the robot took measurements. We make these databases freely available for all researchers: all six databases are described and available at the http://goo.gl/h3pU7Q web address.

https://doi.org/10.1017/S0263574719001061 Published online by Cambridge University Press

Table III. Camera and robot parameter settings.

Robot	Pioneer 3AT
Camera	Bumblebee2®
Stereo baseline	120 mm
Focal length	3.8 mm
CCD	1/3 in.
Image	1024 × 768
Pan-tilt unit	Directed-perception D46
Pan-tilt resolution	0.013°
Tilt setting for datasets	15°
Pan settings	Every 36°
Histogram equalization	OpenCV2 equalizeHist
Outlier removal	Point cloud library Statistical outlier removal $K = 50$, SD = 0.2

Table IV. Dataset parameter settings.

4 m^2 grids	Grid 11, Grid 13
Size	4×4 grid cells
Resolution	0.5 m
Number of images	160 of 772K each
Depth files	160 of ~20M each
Number of points	51 million, 51 million
12.25 m^2 grids	Grid 15, Grid 16
Size	7×7 grid cells
Resolution	0.5 m
Number of images	490 of 772K each
Depth files	490 of ~20M each
Number of points	219 million, 267 million
16 m^2 Grids	Grid 12, Grid 13
Size	4×4 grid cells
Resolution	1 m
Number of images	160 of 772K each
Depth files	160 of ~20M each
Number of points	79 million, 51 million

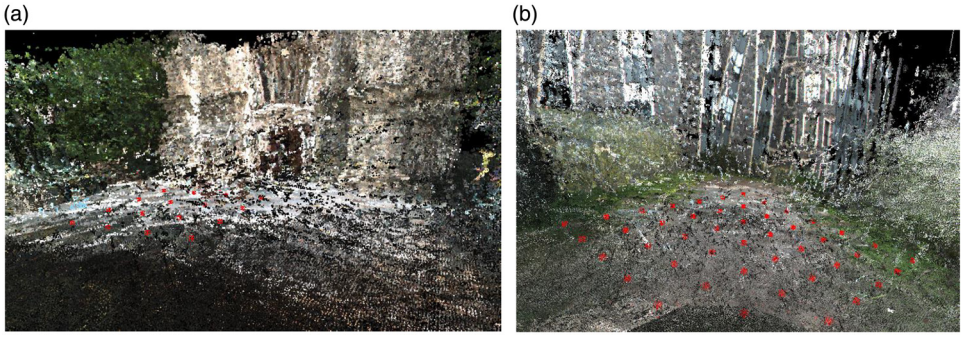


Fig. 3. (a) A composite point cloud created from all 160 raw stereo data imagery of G14 and registered using the spatial relationship of the grid and (b) shows this for the 490 clouds in G16. The grid of darker circles superimposed on the data in the foreground of each shows the grid vertices.

5. Experimental Method: Evaluation of Varying FOV Width

The main contribution of this paper is an experimental comparison of the effect of varying the FOV width on stereo-vision-based visual homing performance. In overview: our experimental method is to run a large number (over 350,000) of homing missions using the six stereo databases previously

https://doi.org/10.1017/S0263574719001061 Published online by Cambridge University Press

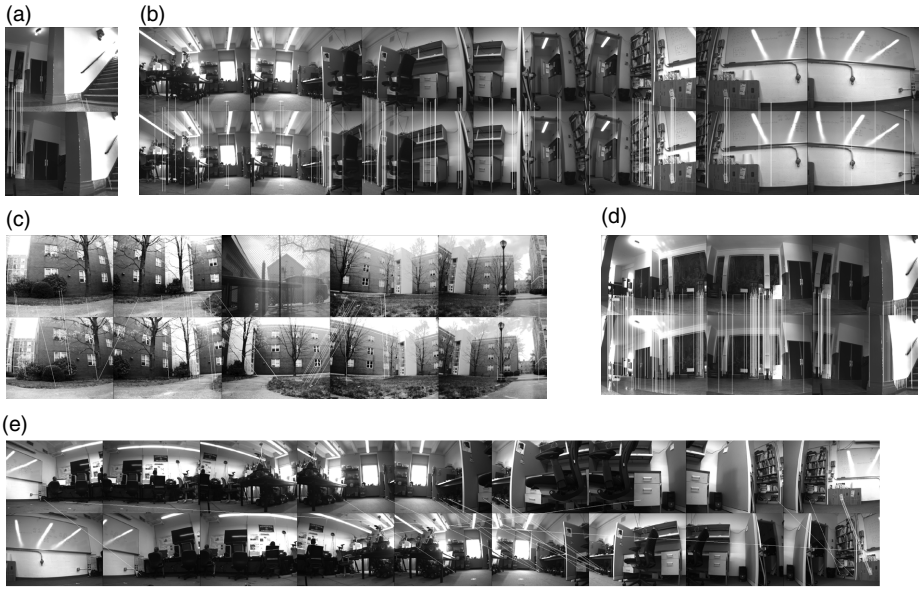


Fig. 4. Composite image SIFT matching for iMax equals (a) 1, (b) 7, (c) 5, (d) 3, and (e) 9. Home image is on top.

described, varying the FOV available to the robot by adjusting the number of images in a composite image, and measuring the subsequent homing performance. This method is now presented in more detail.

An arguably plausible hypothesis is that a wider FOV improves homing performance in those situations where the robot is no longer directly oriented toward the goal scene. With this motivation, homing missions were conducted with a variety of angular offsets, in degrees, between the home image orientation and the start orientation of the homing mission

$$Offsets = \{0, 30, 60, 90, 120, 150, 180\} \tag{5}$$

To evaluate the effect the FOV plays in each of these homing missions, each was run with a different FOV, effected by including a different number of panned images into the composite image. The parameter *iMax* is the number of images in a composite image and it directly constrains the FOV. The construction of a composite image starts with a single image at pan angle 0 ($iMax = 1$ and $FOV = 66°$) and then adds additional images symmetrically to the left and right of this image. For $iMax = 3$, the composite image consists of one central image, one image on to its left at pan angle 36°, and one image to its right at pan angle −36°. For $iMax = 5$, the composite image consists of one central image, two images to the left at 36° and 72°, respectively, and two to the right at −36° and −72°, respectively. Homing experiments were conducted with the *iMax* parameter set to a range of values from 1 image to 9 images, covering a FOV of from 66° through 354°,

$$iMax = \{1, 3, 5, 7, 9\} \tag{6}$$

Since successive *iMax* values add two images on each side rotated 36°, these *iMax* values are equivalent to a FOV, in degrees, of

$$FOV = \{66, 138, 210, 282, 354\} \tag{7}$$

For all the work in this paper, *iMax* is constrained to be odd and $1 \leq iMax \leq 9$. Figure 4 shows SIFT matching of home and current visual images (like Fig. 1) for all five values of *iMax* used in these experiments.

Recent work has suggested that choice of home location strongly affects performance,[27] therefore, performance was evaluated on three distinct kinds of missions:

https://doi.org/10.1017/S0263574719001061 Published online by Cambridge University Press

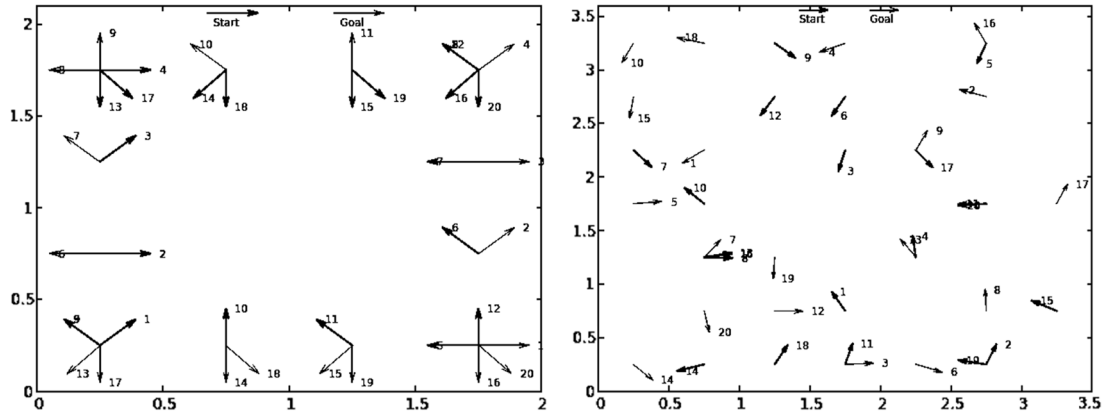


Fig. 5. (a) Example 4×4 RN script, offset = 30 and (b) example 7×7 RC script, offset = 120. Each start/goal arrow is annotated with a mission number 1–20.

1. *Reorient* (**RO**) *missions* requiring the robot just to reorient itself from a start orientation to a home orientation whose coordinates were all within one database grid cell.
2. *Row and column* (**RC**) *missions* requiring the robot to traverse positions encompassing entire rows, columns, or diagonals of the database grid.
3. *Random* (**RN**) *missions* where the start and home locations are selected from a uniform random distribution across the entire range of coordinates.

Note that the robot position is not limited to the grid structure of the database and could be any position in the 2D plane. However, the database imagery is not valid for locations very distant from those at which the images were taken, so, in practice, the position is restricted to the subset of the 2D plane where the images were collected (Appendix I Eq. (A2)). Figures 5(a) and (b) show examples of RN and RC mission scripts, respectively. Each mission script contains 20 missions of one of the three kinds: RO, RC, or RN. For RO and RC scripts, the orientation offset for each script can be set to one of the seven offsets in For a random mission script, offsets are chosen randomly for each of the 20 missions in the script in the range between two adjacent values in Eq. (5). For example, random orientations between 30° and 60°. Thirty trials of each of the 20 missions in each script were run, and the results were averaged. Finally, each of the mission scripts was run for all six databases. This results in 12,000 different homing examples, each of which was run 30 times. In summary, the total number of trials, T, was

$$T = [|Offset| \times |\{RO, RN\}| + (|Offset| - 1)|\{RC\}|] \times |iMax| \times 20 \times 30 \times 6 = 360{,}000 \quad (8)$$

The performance of each homing mission is measured by how close the robot got to the home location and orientation. The performance p of a mission is defined as the weighted sum

$$p = 0.5\frac{1}{e_p} + 0.5\frac{1}{e_a} \quad (9)$$

where e_p is the normalized position error (in meters) between the home location and where the algorithm terminated and e_a is the normalized angular error (in degrees) between the home orientation and the orientation at which the algorithm terminated. The error is normalized by dividing each raw error by the largest raw error for the trial. A larger value of p indicates a better homing performance.

The algorithm was implemented in C++ with MPI,[28] leveraging both the OpenCV library and Lowe's SIFT library,[29] with the database interface described in Appendix II. All the experiments were run on 60 cores of a high-performance cluster. Trial results were averaged to generate average errors per mission script. Performance values were then calculated for every mission script, value of *iMax* and value of *Offset*, and for every database; generating 12,000 performance values. These values are summarized in Section 6.

https://doi.org/10.1017/S0263574719001061 Published online by Cambridge University Press

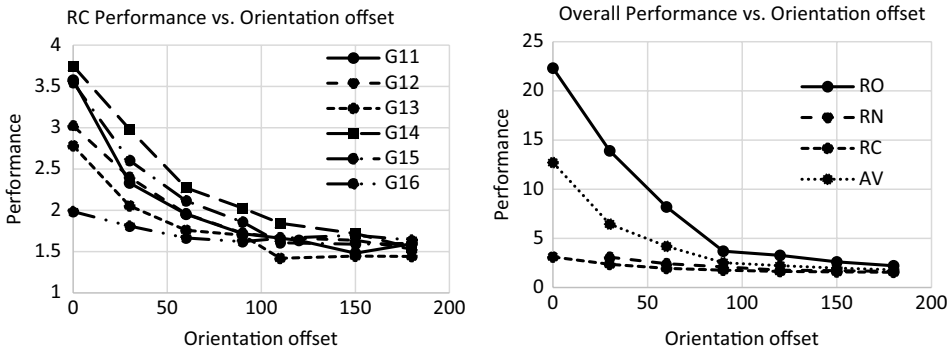


Fig. 6. (a) Graph of average performance versus orientation offset for RC type missions for all iMax for each database and (b) graph of average performance versus orientation for each mission type over all databases and iMax.

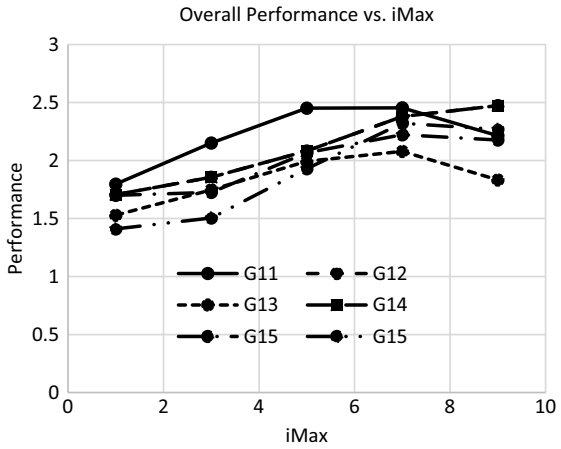


Fig. 7. (a) Graph of average performance versus iMax for all type missions for all offsets for each database.

6. Results

Figure 6(a) graphs homing performance averaged over all FOV (*iMax*) values versus offset angle between the initial orientation and the home orientation for the RC (row, column, and diagonal traverses) type mission scripts, for each database G11 through G16. Homing performance varies between databases, with the algorithm producing the best result on G14 and producing its poorest performance on G16. However, all performance curves have a very similar shape: homing performs better for smaller offset angles with performance leveling off for offsets beyond 90°. Note that odd-numbered databases are indoors and even-numbered are outdoors. There does not appear to be any pattern to performance for indoors versus outdoors.

Figure 6(b) shows the homing performance averaged over all FOVs (*iMax*) values versus offset angle for all three types of missions, for each database G11 through G16. This graph shows that the algorithm performed best on RO (reorientation with any translation) missions. These averaged graphs show the same decreasing relationship between performance and orientation offset. (Note that the RN mission has no data-point for offset zero: the data-point for orientation 30° measures the performance of missions with random orientation offsets between 0° and 30°.)

Figure 7 shows the averaged performance of *all* mission types for *all* orientation offsets against the FOV (*iMax*) per database. Figure 8 shows all these data averaged for each mission type rather than per database and plotted against the FOV in degrees, rather than against *iMax* number. The dotted line in Fig. 8 is the average performance of *all* mission types and *all* offsets against the FOV in degrees.

https://doi.org/10.1017/S0263574719001061 Published online by Cambridge University Press

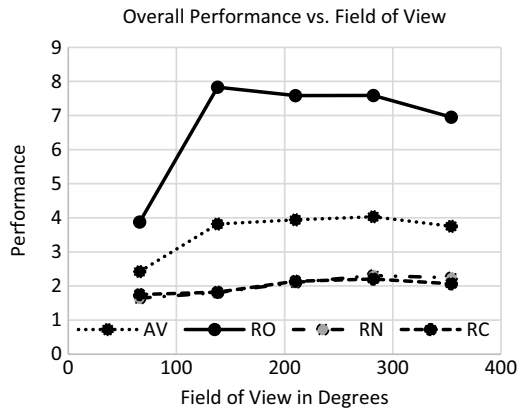


Fig. 8. Graph of average performance versus FOV in degrees for each mission type over all databases and offsets.

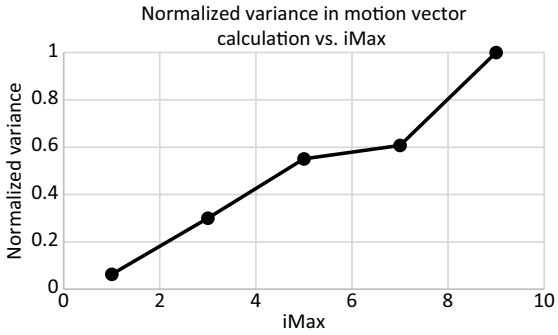


Fig. 9. Graph of homing vector variance versus FOV (iMax).

6.1. Discussion

The graphs in Figs. 7 and 8 initially rise from left to right – indicating improved performance for a wider FOV. Since these results are averaged over all the values of the orientation offset, they include results from cases where the robot initially is oriented away from home. A mission with a small FOV and large orientation offset will in all likelihood fail since no home features are in view. Note that visual homing performance has a stochastic element due to the stochastic nature of image noise and feature matching. For a mission with a large orientation offset, a larger FOV is more likely to contain more features from the home image and improve the performance in that case. This first result coincides with the intuition that a larger FOV is better than a smaller FOV for homing. It can also be seen in these graphs that increasing FOVs yield diminishing performance returns. As long as the FOV allows some portion of the home image to be visible, it is likely that the robot will turn to make more of the home image visible.

Surprisingly, the average in Fig. 8 shows not just a plateau of performance beyond 210°, but a slight decrease in performance for the widest FOVs. The average performance shows a peak at a FOV of 282° and a performance decline of about 7% at a FOV of 354°. The drop-off in performance suggests that there is a trade-off of the advantage bestowed by a wider FOV for handling larger offsets against a growing disadvantage to a wide FOV. Several additional experiments were carried out to evaluate what this growing disadvantage might be.

The motion values from Eqs. (3) and (4) were logged for 21,000 homing runs on the G11 database using the RC script (averaged over multiple start/end locations, offsets, and repetitions, as before) with all values of *Offset* and *iMax* and the variance in these values was also recorded. The two variances were normalized to compensate for their different ranges, averaged to single normalized variance value, and are graphed against the FOV (*iMax*) value in Fig. 9. The graph shows that as the FOV increases so does the variance in the motion vector, evidencing a decrease in quality of the homing vector with a larger FOV.

https://doi.org/10.1017/S0263574719001061 Published online by Cambridge University Press

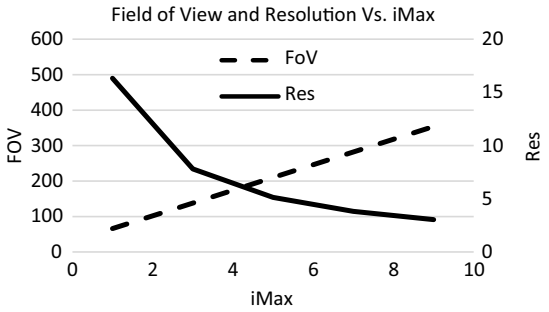


Fig. 10. Graph of the relationship between the FOV (FOV) and the image resolution defined as image width over FOV.

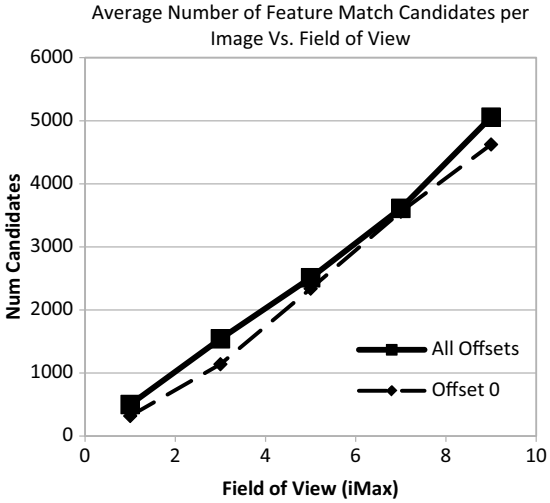


Fig. 11. Graph of the average number of feature matches per image (solid) for all offsets and for offset 0 (dashed) versus FOV (iMax).

Our first hypothesis was that this decreasing quality was related to image resolution: After all, if a 66° FOV lens is replaced with a 138° FOV lens, the resolution of the digital image for the wider FOV will be less than for the narrower FOV. Because the camera image is of the same fixed size, as the FOV increases, the resolution decreases as illustrated in Fig. 10. As the FOV increases, homing is in general more successful, since large offsets between the initial orientation and home orientation can now be navigated successfully. However, the cost of the wider FOV is decreased resolution, resulting in a decrease in the quality of the feature matches.

While this argument would apply to a robot using the same camera with different lens for different FOVs, it does not in fact apply to our experiments. While the image height is fixed in our experiments for all FOVs, the image width is not, and increases linearly with the number of images in the composite image. Thus, decreasing resolution cannot be the cause of the small drop in performance in Fig. 8. Additional experiments were conducted to collect information on SIFT feature matching across all FOVs and offsets. Figure 11 (solid line) shows the average number of candidate feature matches that SIFT generates for each feature of the home image graphed against the increasing FOV. As the FOV increases, the number of possible matches for each feature also increases. This is, perhaps, reasonable if more home image features are coming into view for larger FOVs. However, Fig. 11 (dashed line) shows this relationship for just the offset 0 missions, where the robot faces the home location and a wider FOV will, in fact, not show additional home features; and this graph also shows that the number of candidates increases with the FOV.

As mentioned, there is a stochastic element to SIFT matching due to image noise and image content uncertainty. Figure 11 in conjunction with Fig. 9 strongly indicates that as the FOV expands, more false match candidates are identified for each home feature, due to this stochastic element. The more images added to iMax to increase the FOV, the more likely one of these images will contain

https://doi.org/10.1017/S0263574719001061 Published online by Cambridge University Press

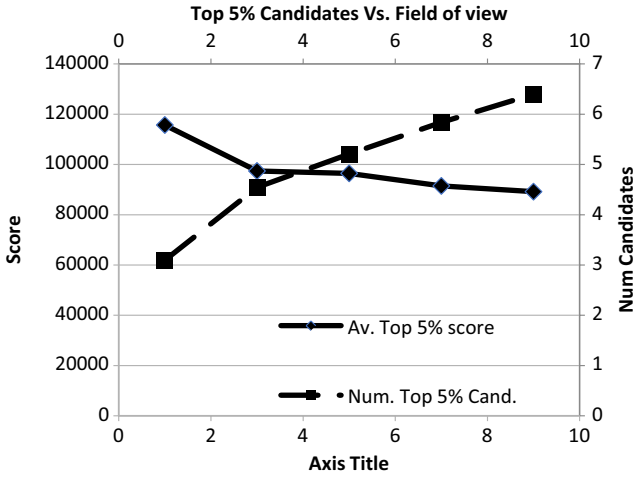


Fig. 12. Graph of the average number of matches in top 5% of score range (dashed) and top 5% score (solid) versus FOV (iMax).

a false match since there are simply more features to match against. Figure 12 shows the average number of strong candidates per feature (defined as top 5% of the score) as a dashed line, and the value of the top 5% score as a solid line, versus the FOV. Taken together, this shows that not only are there more false candidates per feature with increasing FOV, there are more strong, false candidates per feature.

Once the FOV is large enough to encompass most of a home image, in any further increase, the result of the *disadvantage* of more potential false matches is the slight decrease in home performance in Fig. 8. Restricting the FOV to be just wide enough to encompass most of a home image, and not wider, has a performance advantage. We refer to this as the 'blinder effect': horses pulling loads on busy streets are often equipped with blinders that restrict their FOV to just the front so as to avoid distractions. Restricting the FOV during homing can eliminate spurious false candidates by just restricting the number of potential matches. This issue does not arise specifically from our algorithm; any matching-based visual homing approach should suffer from this constraint.

7. Conclusions

This paper has evaluated the effect of varying the size of the horizontal FOV in stereo-vision-based visual homing. The visual homing algorithm introduced here is a modification of Nirmal and Lyons,[7] a correspondence-based homing algorithm that uses stereo-depth information to calculate home distance as well as home direction. Most visual homing has leveraged a panoramic FOV under the intuition that the home location needs to be visible from any starting location. However, common stereo-cameras typically have a smaller FOV, and so we use a composite wide FOV image, constructed by concatenating several visual and depth images together. This begs the question of how the choice of width of the horizontal FOV affects visual homing performance.

To be able to conduct a sufficiently large number of homing examples to answer this question, we presented a stereovision database methodology and six databases we have collected to conduct visual homing with stereovision experiments. A series of over 350,000 visual homing trials was carried out on the six visual homing databases, including three indoor and three outdoor databases. Because of the intuition that a wide FOV is necessary to address the situation where the robot is not facing the home location when homing starts, we conducted trials of different homing experiments with a variety of initial orientation offsets from the home orientation of from 0 to 180° and a variety of FOVs from 66° ($iMax = 1$) to 354° ($iMax = 7$).

Our conclusions can be summarized as follows:

1. $iMax = 1$, $FOV = 66°$ produces best homing performance in all cases when the offset angle is small (typically less than 50 to 100°).

https://doi.org/10.1017/S0263574719001061 Published online by Cambridge University Press

2. Increasing the FOV from 66° does improve homing performance for most FOVs, supporting the value of a wider FOV in handling larger initial orientation offsets.
3. For FOVs above 210°, the advantage of widening the FOV is small at best because, at that width, most of the home image is in view anyway.
4. Increasing the FOV to the widest FOV tested 354° produced slightly worse results, since the advantage of overlapping the initial FOV with the home image is no longer as strong and outweighed by the disadvantage of increasingly likely false matches.

While the current work has focused on stereo-vision, we will also study whether these results hold for non-stereovision-based homing methods. As far as we know, while there is work on the effect of limiting the vertical FOV,[30] to understand the effect of the 'skyline' cue, there does not seem to be similar studies for HFOV.

Funding
Research partially supported by Defense Threat Reduction Agency (DTRA) basic research award #HDTRA1-11-1-0038.

References

1. A. Stelzer, E. Mair and M. Suppa, "Trail-Map: A Scalable Landmark Data Structure for Biologically Inspired Range-Free Navigation," *Proceedings of the IEEE International Conference on Robotics and Biomimetics*, Bali, Indonesia (2014) pp. 2138–2145.
2. S. Thrun, W. Burgard and D. Fox, *Probabilistic Robotics* (MIT Press, Cambridge, MA, 2005).
3. G. Dudek and M. Jenkin, *Computational Principles of Mobile Robotics* (Cambridge University Press, Cambridge, NY, 2000).
4. G. Dudek, M. Jenkin, E. Milios and D. Wilkes, "Robotic exploration as graph construction," *IEEE. Trans. Robot. Autom.* **7**(6), 859–864 (1991).
5. R. Moller, M. Krzykawski, L. Gerstmayr-Hillen, D. Fleer and J. de Jong, "Cleaning robot navigation using panoramic views and particle clouds as landmarks," *Robot. Auton. Syst.* **61**(12), 1415–1439 (2013).
6. D. Churchill and A. Vardy, "Homing in Scale Space," *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*, Nice, France (2008).
7. P. Nirmal and D. Lyons, "Homing with stereovision," *Robotica* **34**(12), 2741–2758 (2015).
8. A. Argyros, K. Bekris, S. Orphanoudakis and L. Kavraki, "Robot homing by exploiting panoramic vision," *Auton. Robots* **19**, 7–25 (2005).
9. R. Aggarwal, A. Vohra and A. Namboodiri, "Panoramic Stereo Videos with a Single Camera," *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition*, Las Vegas, NV (2016).
10. W. Feng, B. Zhang, J. Roning, X. Zong and T. Yi, "Panoramic Stereo Vision," *Proceedings of the SPIE Conference on Intelligent Robotics and Computer Vision XXX: Algorithms and Techniques*, Burlingame, CA (2013).
11. D. Churchill and A. Vardy, "An orientation invariant visual homing algorithm," *J. Intell. Robot. Syst.* **17**(1), 3–29 (2012).
12. M. Franz, B. Scholkopf, M. Mallot and H. Bulthoff, "Where did I take that snapshot? Scene-based homing by image matching," *Biol. Cybern.* **79**, 191–202 (1998).
13. Q. Zhu, C. Liu and C. Cai, "A novel robot visual homing method based on SIFT features," *Sensors.* **15**, 26063–26084 (2015).
14. R. Moller, "Do insects use templates or parameters for landmark navigation?" *J. Theor. Biol.* **210**, 33–45 (2001).
15. R. Moller and A. Vardy, "Local visual homing by matched-filter descent in image databases," *Biol. Cybern.* **95**, 413–430 (2006).
16. D. Lambrinos, R. Moller, T. Labhart, R. Pfeifer and R. Wehner, "Mobile robot employing insect strategies for navigation," *Robot. Auton. Syst.* **30**, 39–64 (2000).
17. A. Ramisa, A. Goldhoom, D. Aldavert, R. Toledo and R. de Mantaras, "Combining invariant features and the ALV homing method for autonomous robot navigation based on panoramas," *J. Intell. Robot. Syst.* **64**, 625–649 (2011).
18. J. Pons, W. Huhner, J. Dahmen and H. Mallot, "Vision-Based Robot Homing in Dynamic Environments," *Proceedings of the 13th IASTED International Conference on Robotics and Applications*, Würzburg, Germany (2007).
19. R. Tron and K. Daniilidis, "An Optimization Approach to Bearing-Only Visual Homing with Applications to a 2-D Unicycle Model," *Proceedings of the IEEE International Conference on Robotics and Automation,* Hong Kong, China (2014).
20. B. Cartweight and T. Collet, "Landmark learning in bees," *J. Comp. Physiol.* **151**, 521–543 (1983).
21. Y. Fu, T. Hsiang and S. Chung, "Multi-waypoint visual homing in piecewise linear trajectory," *Robotica* **31**(3), 479–491 (2013).

https://doi.org/10.1017/S0263574719001061 Published online by Cambridge University Press

22. M. Liu, C. Pradlier, F. Pomerleau and R. Siegwert, "Scale-Only Visual Homing from an Omnidirectional Camera," *Proceedings of the IEEE International Conference Robotics and Automation*, Saint Paul, MN (2012).
23. F. Fang and D. Lyons, "Robust Homing with Stereovision," *Proceedings of the SPIE Unmanned Systems Technology XX*, Orlando, FL (2018).
24. D. Choi, I. Shim, Y. Bok, T. Oh and I. Kweon, "Autonomous Homing Based on Laser-Camera Fusion System," *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Vilamoura, Portugal (2012).
25. Y. Jin and M. Xie, "Vision Guided Homing for Humanoid Service Robot," *Proceedings of the 15th International Conference on Pattern Recognition (ICPR)* vol. 4, Barcelona, Spain (2000).
26. P. Ammirato, P. Poirson, E. Park, J. Kosecka and A. Berg, "A Dataset for Developing and Benchmarking Active Vision," *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Singapore, Singapore (2017).
27. M. M. Müller, O. J. N. Bertrand, D. Differt and M. Egelhaaf, "The problem of home choice in skyline-based homing," *PLoS One.* **13**(3), e0194070 (2018).
28. D. Lyons, *Cluster Computing for Robotics and Computer Vision* (World Scientific, Singapore, 2011).
29. D. Lowe, "Distinctive image features from scale invariant keypoints," *J. Comput. Vis.* **60**(2), 91–110 (2004).
30. K. Basten and H. Mallot, "Simulated visual homing in desert ant environments: Efficiency of skyline cues," *Biol. Cybern.* **102**, 413–425 (2010).
31. A. Vardy, *Biologically Plausible Methods for Robot Visual Homing,* Ph.D. Thesis (School of Computer Science, Carleton University, 2005).
32. R. Moller, A. Vardy, S. Kreft and S. Ruwisch, "Visual homing in environments with anisotropic landmark distribution," *Auton. Robots* **23**, 231–245 (2007).

Appendix I. Stereo-vision Homing Databases

A.1. Collection procedure

The first step in collecting a visual homing database for a physical location is to lay out a regular, square grid in the location. Our established procedure is to lay out the grid using a measuring tape, marking each grid cell on the vertex of the grid with tape. The robot is teleoperated to the vicinity of the first grid cell and its position and orientation manually adjusted so that fiducial markings on the body of the robot line up with the tape. The automated recording and storage of the imagery for the current grid cell is then initiated. When recording is finished, the robot is teleoperated to the vicinity of the next grid cell and the procedure repeated.

The dimension n of the (square) grid is the number of vertices along one side. The resolution r of the grid is the distance between grid vertices in *meters*. The grid cells are numbered in row-major order: a grid cell $(i, j) \in \{0, \ldots, n-1\}^2$ corresponds to database square $k = (j + (n-1)i) \in \{0, \ldots (n-1)^2\}$, and the database folder $SQUARE_k$ contains the 10 visual and 10 depth images for that grid cell. The spatial location of the grid cell (i, j) is given by $p = (x, y) = (ir, jr) \in \{0, \ldots (n-1)r\}^2$. Any position $p = (x, y)$ can be translated to its grid coordinates $(i, j) = (x \text{ div } r, y \text{ div } r)$ for $0 = x, y = (n-1)r$.

Each set of visual images for a cell is 7.5 MB in size and each set of depth images is 200 MB. The Triclops^{TM} library, which provides software support for the Bumblebee2 stereo-camera[1], was used to carry out the stereo calculations. Each image can yield over 780,000 depth points – one per image pixel. However, image texture does not allow for stereo-depth calculation at every pixel and in general there are about 400,000 points per image. Minimal enhancement and noise elimination is carried out on image and depth data before it is stored; the stored data are very similar therefore to what would come from the real hardware. The visual images are histogram equalized, and the stereo images filtered using the point cloud library statistical outlier remover[2]. Additional filtering and cleaning could no doubt be done at the discretion of a user of our dataset. Our results here show the performance with no additional filtering or cleaning. The visual images and depth files are named for the orientation at which they were collected in $SQUARE_i$.

[1] (https://www.ptgrey.com/triclops)

[2] http://pointclouds.org/documentation/tutorials/statistical_outlier.php (Num. neighbors = 50, Stand. Dev. multiplier = 0.2)

https://doi.org/10.1017/S0263574719001061 Published online by Cambridge University Press

A.2. Retrieving database imagery for homing

The homing algorithm is modified to include the parameter *iMax* that indicates how many images to concatenate together to form the input image for homing

$$C = \text{capture}(iMax, p, \theta) \tag{A1}$$

where $p = (x, y)$ is the current robot location and θ is the current robot orientation. Note that the robot position is not limited to grid locations, but the position of the robot is restricted to lie in the subset of the 2D reals given by (x,y), in meters, where

$$0 \leq x, y \leq (n-1)r \tag{A2}$$

The $SQUARE_k$ folder to retrieve the images for concatenation is calculated as

$$k = (x \text{ div } r) + (y \text{ div } r)(n-1) \tag{A3}$$

where r and n are the database resolution (in meters) and size (in number of grid cells).

The visual and depth images loaded are those corresponding to the orientations (in degrees, where $\alpha = 36°$)

$$((\theta \text{ div } \alpha) - (iMax \text{ div } 2))\alpha \ldots ((\theta \text{ div } \alpha) + (iMax \text{ div } 2))\alpha \tag{A4}$$

Once a homing vector (Θ_a, Θ_d) is calculated, the new position p_{t+1} (in meters) and orientation θ_{t+1} (in radians) of the robot are calculated from the current position p_t and orientation θ_t,

$$\begin{aligned} \theta_{t+1} &= \theta_t + g_a\, \Theta_a \\ x_{t+1} &= x_t + g_d\, \Theta_d \cos(\theta_{t+1}) \\ y_{t+1} &= y_t + g_d\, \Theta_d \sin(\theta_{t+1}) \end{aligned} \tag{A5}$$

A.3. Accessing the databases

All six databases are described and are freely available at the http://goo.gl/h3pU7Q web address.

https://doi.org/10.1017/S0263574719001061 Published online by Cambridge University Press