Image Stream Synchronization for Visual Navigation

Helio Perroni Filho (Doctorate Course, 2nd year) †1

Intelligent Robotics Laboratory, OHYA's group

Abstract— Appearance-based navigation methods commonly require pairing up images from two separate sequences – a reference *teach step* record and the record of the current *replay step* – in terms of viewpoint proximity. This task is complicated by environmental variations such as lighting changes, viewer pose differences, moving objects and landmark occlusion. This article presents an image similarity metric, *feature point correlation*, which is shown to behave well under such conditions.

Keywords: Image Processing, Machine Learning, Navigation

1. Introduction

Appearance-based navigation is a robot navigation paradigm where the environment is represented as image sequences collected along a route, which are matched to live sensory input in order to estimate current robot pose [1]. In comparison to Visual SLAM methods [2], it constitutes a less computationally expensive, more intuitive model, closer to how humans navigate our surroundings. The typical appearance-based navigation use case is *teach-replay*, where a robot is led through an environment by a guide (the teach step), and must then autonomously retrace the original path, orienting itself by the data gathered during the guided phase (the replay step) [3]. Figure 1 illustrates the concept.

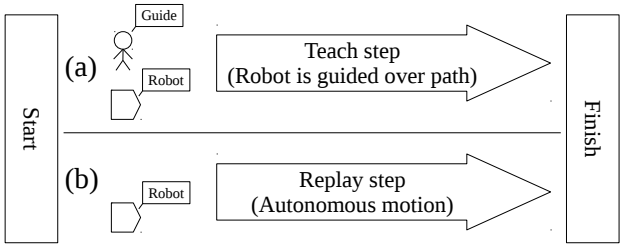


Fig. 1 Teach-replay navigation use case. A mobile robot is first taken along a path by a guide (a), and must later retrace it autonomously (b).

Appearance-based navigation methods commonly require pairing up teach and replay images by viewpoint proximity. Formally, let the image sequence collected over the teach step and the image sequence collected so far over an ongoing replay step be represented respectively by:

$$S_{teach} = [I_1, \ldots, I_i, \ldots, I_m] \tag{1}$$

$$S_{replay} = [I'_1, \ldots, I'_j, \ldots, I'_n] \tag{2}$$

The robot's pose at the time an image I was taken is given by the *viewpoint function*:

$$o(I) = (x, y, \theta) \tag{3}$$

The *pairing function* $g(I'_j)$ is defined over the domain of replay step images such that $g(I'_j) = I_i$ is the teach image whose corresponding viewpoint $o(I_i)$ is the most similar to the viewpoint $o(I'_j)$ of replay image I'_j:

$$g(I'_j) = \arg \min_{I_i \in S_{teach}} \|o(I_i) - o(I'_j)\| \tag{4}$$

Obviously if $o(I)$ were reliably known at any one time there would be no need for image pairing in the first place. Therefore what is needed is an approximation $g'(I'_j)$ that circumvents this requirement. In particular, *appearance-based pairing* methods exploit the insight that changes to robot pose cause predictable changes to visual input, and therefore it must be possible to design an image similarity metric that correlates well to pose proximity [4]. Table 1 relates a number of such methods proposed in the literature.

Table 1 Summary of appearance-based image pairing methods proposed in the literature, characterized in terms of employed sensors, test environments, and variations allowed between teach and replay steps.

Method	Sensors	Tested	Landscape Variations		
			Lighting	Occlusion	Movement
Minimal template matching error [5]	Monocular camera	Indoors	No	No	No
Cost function minimization [6]	Monocular camera	Indoors	No	No	No
Mutual information [7]	Monocular camera	Indoors	No	Yes	Yes
Feature point tracking [8]	Monocular camera	Indoors, Outdoors	Yes	No	Yes
Local best match and sequence recognition [9]	Monocular camera	Outdoors	Yes	Yes	Yes
Stereo feature matching and visual motion estimation [10]	Stereo camera	Indoors	Yes	Yes	Yes
Average Landmark Vector [11]	360^o panoramic camera, polarized light sensor, wheel encoders	Simulation, Outdoors	Yes	No	No
Block matching Optical Flow [12]	360^o panoramic camera, wheel encoders	Indoors	Yes	Yes	No

This article describes the *feature point correlation* appearance-based pairing method, designed to work well under both natural and artificial lighting conditions, and also against changes in lighting, landmark occlusion and moving objects. The article's remaining sections are organized as follows: first, related works are presented in terms of proposed image pairing methods. Next feature point correlation is described, followed by experiments demonstrating its performance. Directions for further research are discussed in the conclusion.

2. Related Work

Image pairing methods based on cost minimization define a mismatch metric between any two images, then pair up replay images to least mismatched teach images. Template matching [5] is a straightforward way to implement this. A more involved alternative, noted for its resource efficiency [1] is to ex-

†1 Graduate School of Systems and Information Engineering, University of Tsukuba

tract vertical lines from teach and replay images, then define mismatch in terms of differences in intensity, position and number of lines in each set [6]. However, these metrics are vulnerable to changes in lighting, landscape appearance and the presence of moving objects, having been successfully tested only in static indoors environments. Mismatch metrics in general are also susceptible to ambiguities in environments with many repeating features, such as long corridors.

Mutual information [7] and Kanade-Lucas-Tomasi (KLT) feature tracking [8] have been proposed as more robust mismatch metrics. Ambiguities over repeating environmental features can be resolved by splitting teach records in segments delimited by milestone frames, and then limiting the search to a single segment at a time. The transition from one segment to the next can be determined by watching the intensity of the mismatch between current input and the next milestone: a reversal from a decreasing to an increasing trend indicates the milestone has been passed [5, 8].

Another way to make mismatch minimization more robust is to look not for a global best match for each visual input, but instead to look for coherent sequences of "good" matches between successive replay inputs and the teach record [9]. Tested on video clips recorded from manually driven cars, this method was highly successful in recognizing revisited locations over a wide range of variations (day, night, rain, moving cars, seasonal changes, etc), but no tests involving actual autonomous navigation were performed, and test landscapes were restricted to open outdoors spaces.

Stereo vision systems can also use the KLT tracker to find correspondences between feature points in three-dimensional space. This method is less vulnerable to ambiguities in appearance, though it must still be complemented by motion estimation for cases where feature matching is not possible (e.g. due to the presence of moving elements or changes to the landscape) [10]. This method was shown to reliably retrace paths as long as $60m$, but no results on lighting changes or outdoor tests have been reported.

Finally, panoramic cameras make possible to use methods that compare current visual input and stored images to compute *direction vectors* pointing towards a target pose [11, 12]. In this way the need for image pairing is sidestepped. However those methods require that a common landmark set remain visible throughout both teach and replay steps, limiting applicable range as well as robustness to landscape changes. Additionally, ensuring the panoramic camera will have a clear field of view all around the robot might be problematic from an engineering perspective, depending on form-factor considerations and application scenarios.

3. Method

Figure 2 provides an schematic overview of the feature point correlation method. The remainder of this section is dedicated to explaining each step of the procedure in detail.

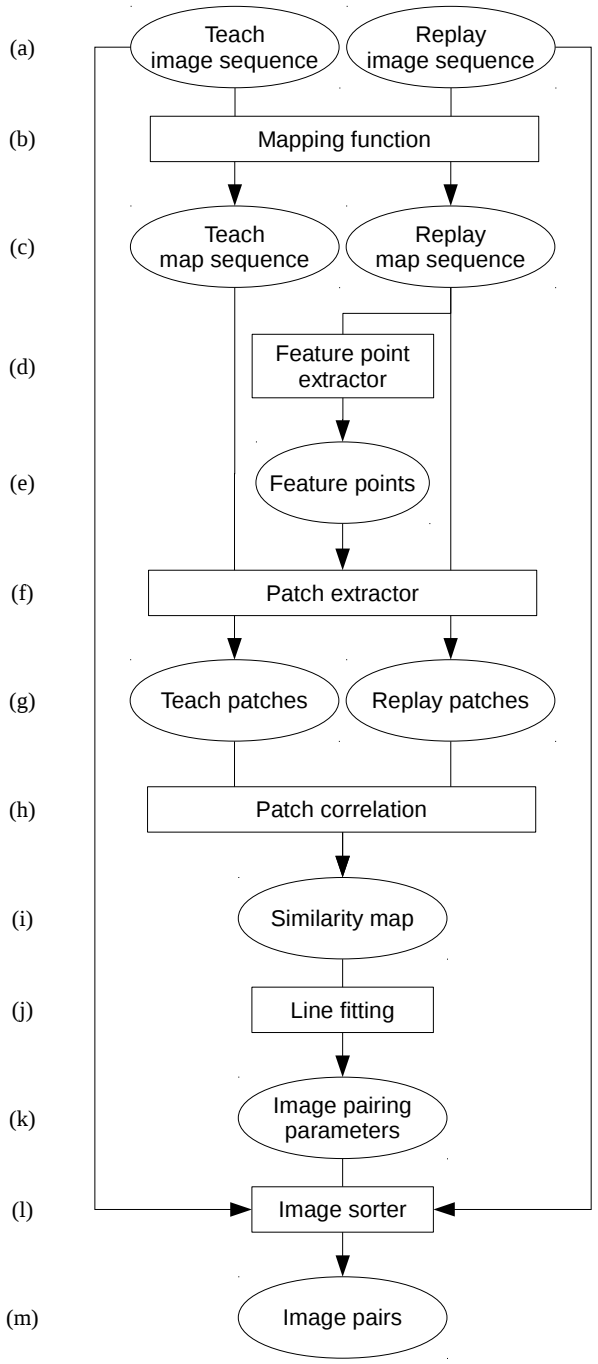


Fig. 2 Image pairing using the feature point correlation method. Image sequences recorded during teach and replay steps (a) are transformed by a mapping function (b) into respective teach and replay map sequences (c). Replay maps are dispatched to a feature extractor (d) which generates lists of feature points (e). The feature points along with teach and replay maps are dispatched to a patch extractor (f) which produces lists of patches from both the teach and replay map sequences (g). Patch correlation (h) is performed over these lists to produce a similarity map (i). A line-fitting algorithm (j) is executed over the similarity map to optimize the image pairing parameters (k), which are passed to the image sorter (l) along with the original teach and replay image sequences to assemble the sequence of image pairs (m).

As stated before, let the image sequence collected over the teach step and the image sequence collected so far over an on-

going replay step be represented respectively by:

$$S_{teach} = [I_1, \ldots, I_i, \ldots, I_m]$$
$$S_{replay} = [I'_1, \ldots, I'_j, \ldots, I'_n]$$

First, a mapping function $\eta(I) = H$ is applied to each image I in the teach and replay sequences, producing a *map* H less susceptible to lighting variations:

$$Z_{teach} = [H_i \mid H_i = \eta(I_i)] \tag{5}$$
$$Z_{replay} = [H'_j \mid H'_j = \eta(I'_j)] \tag{6}$$

From each replay map a list of *feature points* is extracted according to some definition of "feature":

$$P(H'_j) = [p_{j,1}, \ldots, p_{j,k}, \ldots, p_{j,z}] \tag{7}$$

Where $p_{j,k} = (u_{j,k}, v_{j,k})$ is a point in map coordinates. Feature points determine locations over the field of view for comparison between a replay map and maps from the teach sequence.

Individual points are unlikely to provide enough similarity information, therefore whole regions surrounding feature points must be extracted for comparison. Given a feature point $p_{j,k}$, a map H (not necessarily related to $p_{j,k}$ in any way) and a *padding* w, a *patch* $C(p_{j,k}, H, w)$ is defined as a square section of side $2w+1$ taken from map H around $p_{j,k}$:

$$C(p_{j,k}, H, w) = H[p_{j,k} - w : p_{j,k} + w] \tag{8}$$

For each feature point, a patch of padding α is extracted from its originating replay map, and one patch of padding β (such that $\alpha < \beta$) is extracted from each teach map. The *response* generated by a replay patch relative to a teach patch is defined as:

$$r(i, j, k, \alpha, \beta) = \max\left(C(p_{j,k}, H_i, \beta) \star C(p_{j,k}, H'_j, \alpha)\right) \tag{9}$$

Where $\star$ is the normalized cross-correlation operator [13]. The response provides a measure of how much maps H_i and H'_j "look like" each other in the neighborhood of feature point $p_{j,k}$, given that a limited degree of sliding of H'_j over H_i is allowed.

The *local best match* $b(j, k, \alpha, \beta) = i' \in [1, n]$ is the index of the teach map that produces the best response for a given replay map H'_j and feature point $p_{j,k}$ subject to paddings α and β, that is:

$$b(j, k, \alpha, \beta) = \arg\max_{i \in [1,m]} r(i, j, k, \alpha, \beta) \tag{10}$$

The *similarity* $l(i, j, \alpha, \beta)$ between teach map H_i and replay map H'_j can then be defined as the number of feature points extracted from H'_j whose local best matches are found on H_i, that is:

$$l(i, j, \alpha, \beta) = \sum_{k=1}^{z} \begin{cases} 1 & \text{if } b(j, k, \alpha, \beta) = i \\ 0 & \text{otherwise} \end{cases} \tag{11}$$

Because replay patches are allowed to "slide" over the larger teach patches to find a good match, and a relative majority of correct matches can "vote out" mistaken ones, this definition of similarity is robust against limited viewpoint variations and landscape changes. However it is still possible to get poor pairings, especially when there is not a clear consensus among local best matches. Results can be improved by exploiting the monotonic nature of the pairing function – map indexes must always increase, at a rate roughly proportional to the robot's speed at the time the sequence was recorded.

A *similarity map* $\Gamma_{\alpha,\beta}$ is a $m \times n$ matrix such that:

$$\Gamma_{\alpha,\beta}^{m \times n} = [\gamma_{i,j} = l(i, j, \alpha, \beta) \mid i \in [1, m],\ j \in [1, n]] \tag{12}$$

In a similarity map, high similarity values tend to gather in roughly diagonal clusters: this is the direction along which both teach and replay map indexes increase. A diagonal line can be fit over such clusters by solving the following optimization problem:

$$\begin{aligned} \underset{i_1, j_1, i_u, j_v}{\text{maximize}} \quad & L(i_1, j_1, i_u, j_v) = \sum_{j=j_1}^{j_v} \Gamma_{\alpha,\beta}[j \frac{i_u - i_1}{j_v - j_1}, j] \\ \text{subject to} \quad & i_1 = 1 \text{ or } j_1 = 1 \\ & i_u = m \text{ or } j_v = n \end{aligned} \tag{13}$$

An approximation of the pairing function can then be defined as:

$$g'(I'_j) = I_i \mid i = j \frac{i_u - i_1}{j_v - j_1} \tag{14}$$

In the discussion above the mapping function $\eta(I)$, the feature point selector $P(H'_j)$ and the line optimization procedure are left undefined. This is by design, as the feature point correlation method is not bound to any particular definitions of these. For the experiments reported in the next section the following definitions were adopted.

Edge maps provide a simple way to convert images to a representation relatively invariant to lighting changes. A mapping function for color images can therefore be constructed by application of the grayscale transform followed by the bi-dimensional Sobel operator [14]:

$$\eta_E(I) = Sobel_{XY}(Grayscale(I)) \tag{15}$$

For feature point selection the Shi-Tomasi corner detector [15] is used. Given function $ST(H'_j, u, v) = \sigma \in \mathbb{N}^+$ which returns the corner strength of map pixel $H'_j[u, v]$, the feature quality matrix Q_j is defined such that:

$$Q_j^{m \times n} = [q_{u,v} = ST(H'_j, u, v) \mid u \in [1, m],\ v \in [1, n]] \tag{16}$$

From Q_j the ordered set Φ_j can be defined such that:

$$\Phi_j = (\phi_{j,k} = (u_{j,k}, v_{j,k}) \mid Q[\phi_{j,k}] > Q[\phi_{j,k'}]\ \forall\, k' > k) \tag{17}$$

The feature selector function is then defined, for z top-quality features, such that:

$$P_{ST}(H'_j, z) = [\phi_{j,k} \in \Phi_j \mid k \leq z] \tag{18}$$

The line fitting problem can be solved efficiently by following the steps below:

1) Calculate the Hough Transform [16] of the similarity maps with parameters $\rho = 1$ and $\theta = 1^o$, generating the list $L = [f_1, \ldots, f_l, \ldots, f_w]$ such that $f_l(x) = y$ is a line in $\mathbb{Z}^2$;
2) Generate the sublist $L' \subseteq L$ by discarding any non-increasing lines, i.e. lines where $f(x_1) \geq f(x_2) \mid x_1 < x_2$;
3) For each remaining line $f'_l \in L' \mid f'_l(x) = y$, compute the quality function $\psi(f) = \sum_{x=0}^{m} \Gamma[f(x), x]$;
4) Select the line of maximum quality:

$$f_b = \arg\max_{f' \in L'} \psi(f')$$

5) If $f_b(1) \geq 1$ then make $j_1 = 1$ and $i_1 = f_b(1)$, otherwise make $i_1 = 1$ and solve $f_b(j_1) = 1$ to find j_1;
6) If $f_b(n) \leq m$ then make $j_v = n$ and $i_u = f_b(n)$, otherwise make $i_u = m$ and solve $f_b(j_v) = m$ to find j_v.

4. Experiments

Two test environments were selected in the campus of the University of Tsukuba, henceforth referred to as "indoors" and "outdoors": the corridor in front of laboratory room 3D402 (the "indoors" environment) and the paved way by the side of faculty building 3D (the "outdoors" environment). The indoors environment featured a smooth floor; the outdoors environment's floor, while basically flat, was rough enough to add a noticeable amount of vibration to robot movement. Straight trips of length $20m$ were recorded in the indoors environment; in the outdoors environment, in order to avoid reaching a patch of excessively rough terrain, trip lengths were reduced to $15m$.

A Yamabico robot with a front-mounted camera was set up to perform short trips over straight paths across each environment, collecting video records along the way. These videos, recorded at 20 Frames Per Second (FPS), were later sampled at a rate of 1 FPS to produce the teach and replay image sequences used

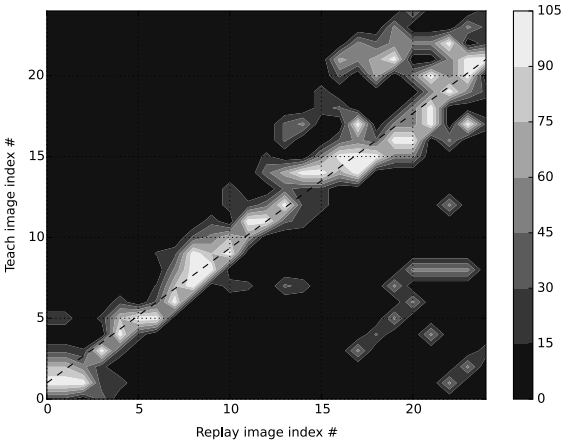
(a) Indoors teach example

(b) Indoors replay example

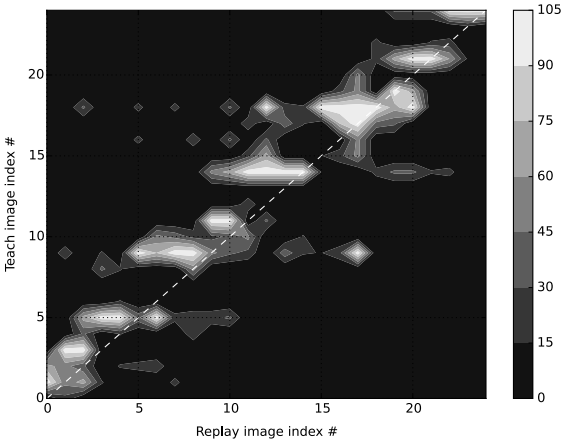
(c) Outdoors teach example

(d) Outdoors replay example

Fig. 3 Image samples from the indoors teach (a) and replay (b) step sequences, as well as outdoors teach (c) and one of the replay (d) step sequences. Between images #30 and #50 in the indoors replay step three people came from behind the robot, staying on the right side of the field of view as they walked away; likewise, in the outdoors teach step a pedestrian came from behind the robot, staying on the left side of the visual field as he walked away from image #20 to #40.



(a) Indoors similarity map



(b) Outdoors similarity map

Fig. 4 Indoors (a) and outdoors (b) similarity maps. Horizontal axis represents replay step images by index number, and the vertical axis, teach step images. Brighter pixels show higher similarity values, and darker pixels, lower similarity. The dashed line indicates the estimated correlation between the image sequences.

(a) Paired replay image

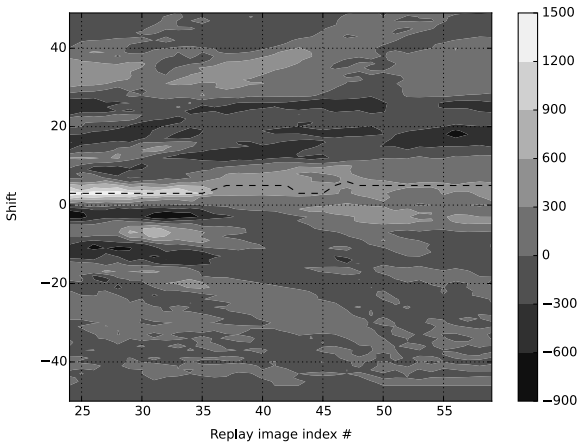
(b) Paired teach image

Fig. 5 Example indoors image pair, formed by an image from the replay sequence (a) and matched teach image (b).

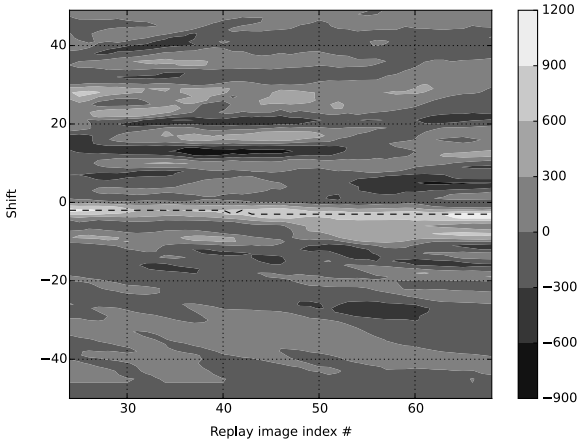
(a) Paired replay image

(b) Paired teach image

Fig. 6 Example outdoors image pair, formed by an image from the replay sequence (a) and matched teach image (b).



(a) Indoors shift map



(b) Outdoors shift map

Fig. 7 Shift maps for indoors (a) and outdoors (b) tests. Horizontal axis is replay step image by index number, and vertical axis is shift between teach and replay images – positive values indicate left shift, and negative values, right. The contour map shows shift likelihood for each replay image across all possible shifts, with brighter pixels representing higher likelihood. A dashed line indicates selected shifts across the replay sequence. The indoors shift map (a) correctly indicates a left shift, consistent with the robot moving leftwards in the replay step relative to the teach step. The outdoors shift map (b) also correctly shows a right shift, consistent with the robot moving rightwards in the replay step relative to the teach step.

in the experiments. The image pairing model described in the previous section was implemented on the Open Source C++ library **Cight** [17], but unfortunately it was not possible to come up with a real-time implementation at this time. Therefore only off-line tests were performed. Method parameters were set to ($\alpha = 30, \beta = 90, z = 10$). The first 25 images of each sequence were used to compute the similarity maps. Initially, a single teach-replay session was recorded at each environment, with linear speed always set to $0.3m/s$.

The indoors environment was deserted when the teach step was recorded, but during the replay step three people came from behind the robot, staying on the right side of the field of view as they walked away; this is reflected in replay images #30 to #50. The robot's initial pose in the replay step was turned 2^o to the left in relation to the teach step's, but otherwise both steps were recorded from the same start location. Figure 3a-b show samples of these image sequences.

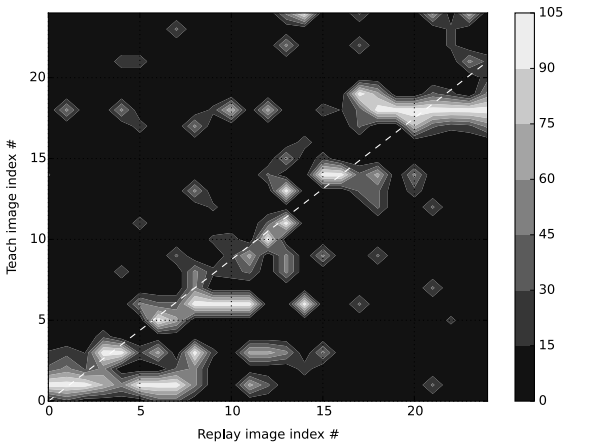
In the outdoors environment a pedestrian came from behind the robot during the teach step, staying on the left side of the visual field as he walked away (this is reflected in teach images #20 to #40), but during the replay step there was no one in sight. The robot's initial pose in the replay step was turned 2^o to the right in relation to the teach step's, but otherwise both steps were recorded from the same start location. Figure 3c-d show samples of these image sequences.

Figure 4a shows the similarity map computed for the teach and replay trips recorded in the indoors environment, while Fig. 4b shows the similarity maps between the outdoors teach and replay trips. The horizontal axis of each similarity map represents replay step images by index number, and the vertical axis, teach step images. Brighter pixels represent higher similarity values, and darker pixels, lower similarity. The dashed line indicates the estimated correlation between the image sequences. Figure 5a-b and Figure 6a-b show a replay image and matched teach image for the indoors and outdoors tests respectively.

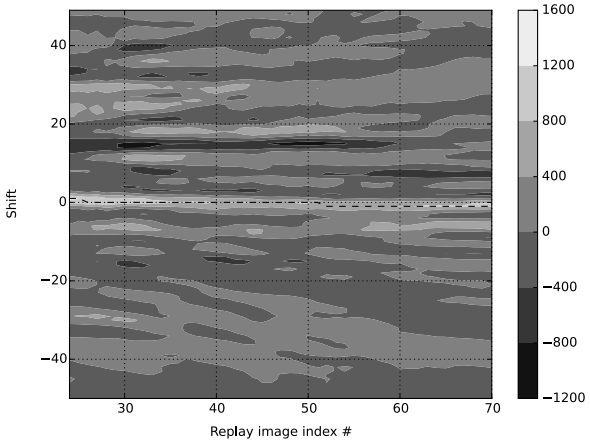
In order to assess the adequacy of the computed image pairings for appearance-based navigation, DiVS shift maps [18] were computed (with parameters $w = 25, d = 50, e = 5$) on the image pairings defined by the above correlations. Figure 7a shows results for the indoors scenario, and Fig. 7b shows results for the outdoors scenario. For each map horizontal axis is replay step image by index number, and vertical axis is shift between teach and replay images – positive values indicate left shift, and negative values, right. The contour map shows shift likelihood for each replay image across all possible shifts, with brighter pixels representing higher likelihood. A dashed line indicates selected shifts across the replay sequence. For the indoors test a left shift is correctly reported, consistent with the robot moving leftwards in the replay step relative to the teach step. The outdoors test also correctly reports a shift consistent with the direction of robot motion in the replay step relative to the teach step, in this case to the right.

Additional tests were later recorded in the outdoors environment to assess the method's performance under varying speeds. Three extra replay steps were recorded, from the exactly same initial pose as the teach step, and linear speeds of $0.3m/s$, $0.45m/s$ and $0.6m/s$.

The results for the first test, with linear speed $0.3m/s$, are shown in Figure 8a-b. Despite a relatively misaligned similarity map, the DiVS shift map correctly reported absent or very small shift, consistent with the replay step following the same path as the teach step. Results for the second test (with linear speed $0.45m/s$) are shown in Figure 9a-b: in comparison with the first test, the slope of the line over the similarity matrix is much more steep, which is a positive result – it correctly reflects the faster speed (and hence rate of change in images) of the re-

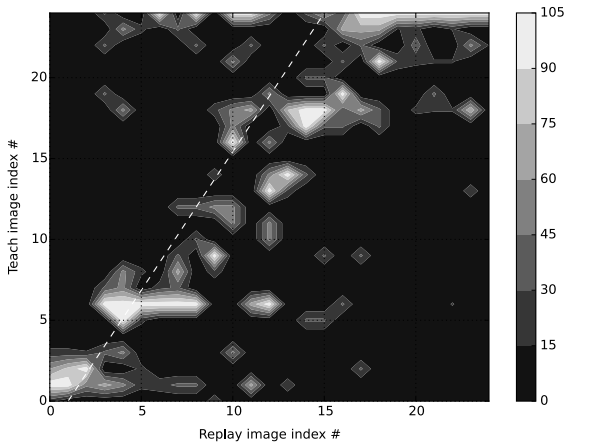


(a) Similarity map

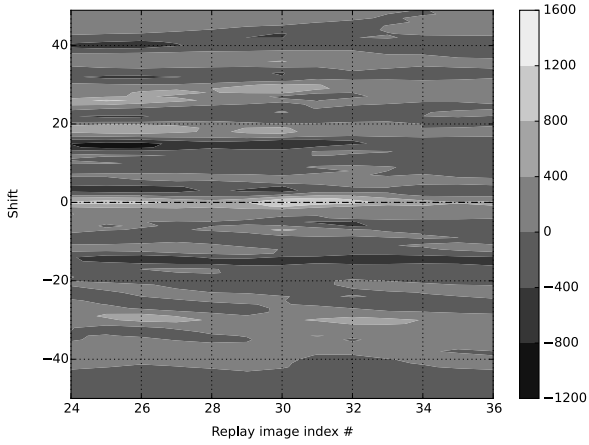


(b) Shift map

Fig. 8 Later outdoors test with identical initial poses for teach and replay steps. Linear speed was set to $0.3m/s$ at both steps. In the similarity map, horizontal axis represents replay step images by index number, and the vertical axis, teach step images. Brighter pixels show higher similarity values, and darker pixels, lower similarity. The dashed line indicates the estimated correlation between the image sequences. In the shift map, horizontal axis is replay step image by index number, and vertical axis is shift between teach and replay images – positive values indicate left shift, and negative values, right. The contour map shows shift likelihood for each replay image across all possible shifts, with brighter pixels representing higher likelihood. A dashed line indicates selected shifts across the replay sequence. Despite a relatively misaligned similarity map, the shift map correctly reports absent or very small shift, consistent with the replay step following the same path as the teach step.



(a) Similarity map



(b) Shift map

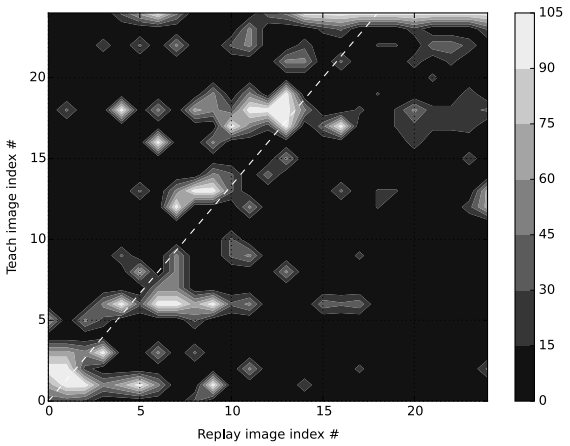
Fig. 9 Outdoors test with identical initial poses for teach and replay steps. Linear speed was set to $0.3m/s$ at the teach step, and $0.45m/s$ at the replay step. In the similarity map, horizontal axis represents replay step images by index number, and the vertical axis, teach step images. Brighter pixels show higher similarity values, and darker pixels, lower similarity. The dashed line indicates the estimated correlation between the image sequences. In the shift map, horizontal axis is replay step image by index number, and vertical axis is shift between teach and replay images – positive values indicate left shift, and negative values, right. The contour map shows shift likelihood for each replay image across all possible shifts, with brighter pixels representing higher likelihood. A dashed line indicates selected shifts across the replay sequence. In comparison with the first test, the slope of the line over the similarity matrix is much more steep, which is a positive result – it correctly reflects the faster speed (and hence rate of change in images) of the replay step. The shift map also correctly reports no drift between teach and replay steps.

play step. The shift map also correctly reports no drift between teach and replay steps.

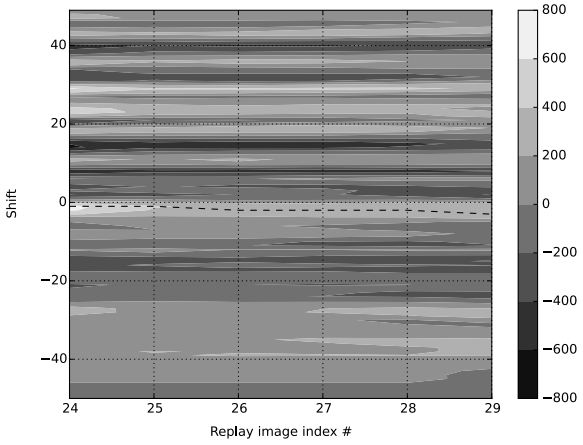
Results for the last test, with linear speed $0.6m/s$, are somewhat problematic, however. As can be seen in Figure 10a-b, the slope of the line over the similarity line is *less* steep than in the previous test – the opposite of what was expected, given the increased speed. Also the DiVS shift map reports a drift to the right, while in fact the robot was again traveling along the same path as the teach step.

5. Conclusion

This article presented *feature point correlation*, a method for pairing images from different sequences collected over successive trips along the same route. Image pairings are shown to correlate well with viewpoint proximity, which is a useful feature for appearance-based navigation systems. Tests performed under different environments demonstrate the method's robust-



(a) Similarity map



(b) Shift map

Fig. 10 Outdoors test with identical initial poses for teach and replay steps. Linear speed was set to $0.3m/s$ at the teach step, and $0.6m/s$ at the replay step. In the similarity map, horizontal axis represents replay step images by index number, and the vertical axis, teach step images. Brighter pixels show higher similarity values, and darker pixels, lower similarity. The dashed line indicates the estimated correlation between the image sequences. In the shift map, horizontal axis is replay step image by index number, and vertical axis is shift between teach and replay images – positive values indicate left shift, and negative values, right. The contour map shows shift likelihood for each replay image across all possible shifts, with brighter pixels representing higher likelihood. A dashed line indicates selected shifts across the replay sequence. The slope of the line over the similarity line is *less* steep than in the previous test – the opposite of what was expected, given the increased speed. Also the DiVS shift map reports a drift to the right, while in fact the robot was again traveling along the same path as the teach step.

ness. Moreover, tests employing DiVS shift computation show the computed pairings to be fit for practical use.

Currently the method's main weakness is the time it takes to compute a similarity matrix, in the order of $30s$ for a matrix as small as 25×25. Several measures can be taken to improve performance though, such as restricting matrix computation to a sparse image set (so only a handful of characteristic images are inspected), optimizing feature selection (so within each image only a handful of high-quality feature points are inspected), and parallel feature inspection. Accordingly, more reliable image mapping, feature selection and line fitting methods could also be researched.

References

[1] F. Bonin-Font, A. Ortiz, and G. Oliver, "Visual navigation for mobile robots: A survey," *Journal of Intelligent and Robotic Systems*, vol. 53, no. 3, pp. 263–296, nov 2008.

[2] J. Aulinas, Y. Petillot, J. Salvi, and X. Lladó, "The slam problem: A survey," in *Proceedings of the 2008 Conference on Artificial Intelligence Research and Development: Proceedings of the 11th International Conference of the Catalan Association for Artificial Intelligence*. IOS Press, 2008, pp. 363–371.

[3] D. Burschka and G. Hager, "Vision-based control of mobile robots," in *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, vol. 2, 2001, pp. 1707–1713.

[4] H. Perroni Filho, "Monocular estimation of spatial displacement," in *Proceedings of the 2013-3 Yamabico Symposium in Robotics*, 2014.

[5] Y. Matsumoto, M. Inaba, and H. Inoue, "Visual navigation using view-sequenced route representation," 1996, pp. 83–88.

[6] T. Ohno, A. Ohya, and S. Yuta, "Autonomous navigation for mobile robots referring pre-recorded image sequence." in *IROS*. IEEE, 1996, pp. 672–679.

[7] R. Stewart, M. Mills, and H. Zhang, "Visual homing for a mobile robot using direction votes from flow vectors," in *Multisensor Fusion and Integration for Intelligent Systems (MFI), 2012 IEEE Conference on*, Sept 2012, pp. 413–418.

[8] Z. Chen and S. T. Birchfield, "Qualitative vision-based mobile robot navigation," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2006.

[9] M. Milford and G. Wyeth, "Seqslam : visual route-based navigation for sunny summer days and stormy winter nights," in *IEEE International Conferece on Robotics and Automation (ICRA 2012)*. IEEE, 2012, pp. 1643–1649.

[10] J. Kim, Y. Bok, and I. Kweon, "Robust vision-based autonomous navigation against environment changes," in *IROS*, 2008, pp. 696–701.

[11] D. Lambrinos, R. Möller, T. Labhart, R. Pfeifer, and R. Wehner, "A mobile robot employing insect strategies for navigation," *Robotics and Autonomous Systems*, vol. 30, no. 1âĂŞ2, pp. 39 – 64, 2000.

[12] A. Vardy and R. Möller, "Biologically plausible visual homing methods based on optical flow techniques," *Connection Science, Special Issue: Navigation*, vol. 17, pp. 47–90, 2005.

[13] H. Perroni Filho and A. F. De Souza, "On multichannel neurons, with an application to template search," *Journal of Network and Innovative Computing*, vol. 2, no. 1, pp. 10–21, 2014.

[14] I. Sobel and G. Feldman, "A 3x3 Isotropic Gradient Operator for Image Processing," 1968, never published but presented at a talk at the Stanford Artificial Project.

[15] J. Shi and C. Tomasi, "Good features to track," in *Computer Vision and Pattern Recognition, 1994. Proceedings CVPR '94., 1994 IEEE Computer Society Conference on*, Jun 1994, pp. 593–600.

[16] R. O. Duda and P. E. Hart, "Use of the hough transformation to detect lines and curves in pictures," *Commun. ACM*, vol. 15, no. 1, pp. 11–15, Jan. 1972.

[17] H. Perroni Filho, "Cight," 2014. [Online]. Available: https://github.com/xperroni/Cight

[18] H. Perroni Filho, "Mobile robot path drift estimation using visual streams," in *Proceedings of the 2014-1 Yamabico Symposium in Robotics*, 2014, pp. 113–120.