Robust and Efficient Feature Tracking for Indoor Navigation

Ranga Rodrigo, Mehrnaz Zouqi, Zhenhe Chen, and Jagath Samarabandu, *Member, IEEE*

***Abstract*—Robust feature tracking is a requirement for many computer vision tasks such as indoor robot navigation. However, indoor scenes are characterized by poorly localizable features. As a result, indoor feature tracking without artificial markers is challenging and remains an attractive problem. We propose to solve this problem by constraining the locations of a large number of nondistinctive features by several planar homographies which are strategically computed using distinctive features. We experimentally show the need for multiple homographies and propose an illumination-invariant local-optimization scheme for motion refinement. The use of a large number of nondistinctive features within the constraints imposed by planar homographies allows us to gain robustness. Also, the lesser computation cost in estimating these nondistinctive features helps to maintain the efficiency of the proposed method. Our local-optimization scheme produces subpixel accurate feature motion. As a result, we are able to achieve robust and accurate feature tracking.**

***Index Terms*—Distinctive features, feature tracking, motion refinement, multihomographies, nondistinctive features.**

I. Introduction

ROBUST feature tracking is a basic requisite for indoor robot navigation [1], [2]. In this context, we define robustness as the ability to accurately track features over all the image frames in which a feature appears, without being deceived by false matches. Monocular-vision-based robot navigation particularly requires a good feature tracking for robot localization. In the navigational context, the task of the feature tracker is to identify the features that are suitable for tracking and to locate them in subsequent images (Fig. 1). Simple feature trackers such as the Kanade–Lucas–Tomasi (KLT) tracker [3] are not robust enough as they fail to match features accurately in subsequent images [1], [4]. This failure is due to two reasons: First, the features used for tracking are nondistinctive features such as corners using normalized cross correlation (NCC), which are based on constancy over a window. Second, motion models used for tracking are not rich enough to describe the feature motion with respect to the camera. We propose a robust tracker suitable for vision-based robot navigation using a combination of distinctive and nondistinctive features and a multihomographic motion model. Our tracker has applications in fast and reliable object tracking [5], [6], image retrieval [7], mosaicking [8], pose detection [9], multiview 3-D reconstruction [10], and robot navigation [11]–[13].

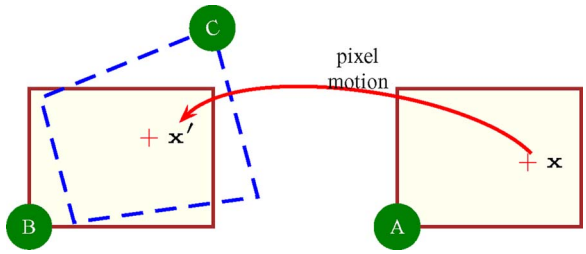


Fig. 1. Motion of a pixel $\mathbf{x}$ between two images. A is the first image, and B is the second image. C is the first image transformed using the homography onto the first image.

The tracker that we propose in this paper constrains the locations of a large number of nondistinctive features by several planar homographies which are strategically computed using distinctive features. To this end, our system performs the following steps:

1) extract distinctive and nondistinctive features for a key frame pair;
2) project nondistinctive features for any other frame pair;
3) compute multiple motion models (homographies) from distinctive features with random sample consensus (RANSAC);
4) refine the tracked feature positions (and, thus, the motion model) for all the frame pairs, given the homographies, using an extended version of the KLT feature tracker;
5) recompute the homographies and update the Kalman filters which track the homographies by exploiting the motion continuity.

A key frame pair is a pair of frames for which we perform the scale-invariant feature transform (SIFT) feature detection and matching in full. Among features, SIFT features are our distinctive features. Distinctive features can be tracked reasonably well without using a motion model. A distinctive feature, while corresponding to a physical entity such as a corner, is a vector of values characterizing the physical feature, its location, scale, and approximate orientation. As a result, they can be matched across wide baselines more accurately than the nondistinctive features. However, the computation of distinctive features and matching are expensive, whereas computing nondistinctive

Manuscript received June 29, 2007; revised November 21, 2007, April 7, 2008, and August 22, 2008. This work was supported in part by the Natural Sciences and Engineering Research Council of Canada, by Precarn Inc., and by the University of Western Ontario. This paper was recommended by Associate Editor Q. Ji.

R. Rodrigo is with the Department of Electronic and Telecommunication Engineering, University of Moratuwa, 10400 Moratuwa, Sri Lanka (e-mail: ranga@ent.mrt.ac.lk).

M. Zouqi and J. Samarabandu are with the Department of Electrical and Computer Engineering, The University of Western Ontario, London, ON N6A 5B9, Canada (e-mail: mzouqi@uwo.ca; jagath@uwo.ca).

Z. Chen is with TVWorks, London, ON N6A 5N6, Canada (e-mail: zchen56@uwo.ca).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSMCB.2008.2008196

features is less expensive. We use these two types of features at two different stages of the algorithm: distinctive features for the homography estimation and nondistinctive features at the motion refinement stage. The motion refinement stage is where we use an extension of the KLT method to refine the motion given by the homographies. Accurate feature matching can be used to calculate the motion model for planar scenes. In our case, this motion model is multihomographic. A homography maps the image of a plane in one image onto the same plane on another image (see Fig. 1). There can be multiple homographies corresponding to many planar structures that are present in the scene. Therefore, our motion model consists of multiple homographies which are strategically computed using distinctive features. These homographies are computed using RANSAC [14] which reduces the effects of outliers. Distinctive features combined with multiple homographies yield almost fully accurate tracking. Assuming motion continuity that exists in typical navigational-type videos, we maintain the multihomographies as the states of a set of Kalman filters and thereby avoid the need of full feature detection and matching for every frame pair. This gives us a large computational advantage. A major contribution of this paper is the use of a combination of distinctive and nondistinctive features in a multihomographic global motion model. We also propose a scheme of illumination-invariant motion refinement and experimentally show that it is advantageous. As a result, we provide a solution to the indoor feature tracking problem without using artificial landmarks. We compare our work with existing approaches in the following section.

A. Related Work

Feature tracking refers to locating a given point in one image with the corresponding point in another image and has applications in autonomous mobile robot navigation systems, 3-D modeling from images, cinema postproduction, etc. There are quite a number of approaches that use computer vision techniques for feature tracking. Some approaches use a video sequence acquired using an onboard camera and perform 3-D reconstruction. For example, Royer *et al.* [15] track Harris corners in recorded video. They use these Harris feature matches across the key frames and perform an offline hierarchical bundle adjustment in turn to come up with accurate robot localization and 3-D reconstruction. Mouragnon *et al.* [2] extend the work of Royer *et al.* by using a fast local bundle adjustment along with the idea of key frames. The first frame is the first key frame. Subsequent key frames are selected as far from each other as possible such that a minimum number of Harris matches is maintained. This helps them in achieving the real-time requirements of localization. In other words, they perform full structure from motion only after several frames. We also make use of this idea to improve the speed of our algorithm.

Nistér *et al.* [13] report a system using only visual input. Their system is able to perform a motion estimation of a stereo or monocular head. They detect Harris corners in each frame and match them between frame pairs. For the sake of speed, they compute the Harris corners using less-expensive techniques. For feature matching (what we call motion refinement in our system), they use NCC within a disparity window. The use of such a disparity window encodes loose geometric constraints and thus requires the motion to be slow. Since motion models (camera poses) are already computed, they can influence the feature association with trajectories. Their system, therefore, belongs to the category of 2-D–3-D trackers, as it tracks the feature positions in 3-D space based on the corresponding 2-D image features. Since we choose to track in a 2-D–2-D fashion, we avoid such camera pose computations. Therefore, we conjecture that our system is able to handle more complicated motion trajectories.

One of the most widely used trackers is the KLT tracker developed by Kanade, Lucas, Tomasi, and Shi [16]–[18]. This tacker is the basis for many other trackers as well. KLT tracks features by minimizing the dissimilarity between the corresponding patches [sum of squared differences (SSD)] using an iterative scheme. Many improvements to this tracker have been proposed. Jin *et al.* [19] propose modifications to handle affine deformations in illumination using a hypothesis verification framework. Fusiello *et al.* [20] propose an outlier rejection scheme. We use a basic local-optimization approach similar to that of the KLT in an affine illumination-invariant manner. Ours does not need an iterative scheme in practice, which is computationally advantageous. Since we rely on RANSAC to reject outliers in our multihomography computation stage, our system is tolerant of outliers. Another disadvantage in KLT-like trackers is the absence of a global motion model. This can lead such algorithms to local minima created by incorrect tracks (see Fig. 15).

None of the aforementioned attempts have exploited the ability of a multihomographic global motion model to constrain the tracks of a large number of features. Since we employ such a model, we are able to eliminate false tracks and perform robust tracking. These motion models are computed using distinctive features and are used to track both distinctive and nondistinctive features. As a result of this combination, we can achieve robust and efficient tracking.

We start the discussion with an outline on distinctive and nondistinctive features in Section II. Then, we justify our selection of Harris, difference of Guassians (DOG), and SIFT features for our work. We then present our multihomography computation scheme in Section III. Section IV describes our local-optimization-based motion refinement. In Section V, we present results to show the performance of our tracker, and we conclude this paper in Section VI.

II. Features and Detection

We indicated in Section I that our feature tracker robustly detects and tracks features. One of the reasons for robustness is the combination of distinctive features with nondistinctive features. Here, we define robustness as the ability to accurately track features across all image frames without being deceived by false matches. Features need to be detected and described using a local neighborhood descriptor. Our descriptors are SIFT descriptors, and we match them across frames to compute the homographies. As a result, we can match a host of nondistinctive features based on these homographies. We use distinctive and nondistinctive features at two different stages

of our algorithm: distinctive features in the multihomographic motion model computation (Section III) and nondistinctive features in the motion refinement stage (Section IV). This section introduces the notion of distinctiveness and shows how the features are computed.

A. Distinctive Versus Nondistinctive Features

Distinctiveness is the main criterion for selecting the type among the intensity-based interest points. For example, Harris corners [21] are nondistinctive features, and SIFT features [22] are distinctive. The size of the neighborhood captured by the feature descriptor and the amount of distillation done with the information within this neighborhood determine the distinctiveness. For example, the Harris detector [21] declares points with intensity varying significantly in two orthogonal directions as interest points. We will describe this in Section II-C. Although a large window is selected, the high distillation of information does not produce a distinctive feature. However, SIFT and its variants store the information of intensities (gradients in particular) in a large vector (128-D vector in SIFT), giving rise to highly distinctive features.

We have categorized features using the notion of distinctiveness. However, this categorization may be seen in another angle. In this point of view, what we identified as nondistinctive features are mere interest points which have the potential of being repeatedly detected. An example is a character in a textbook. This gives us a good feature to track but does not give a method of finding the same feature. The method of finding the features in subsequent images is by collecting all the features that match the criterion and comparing against a descriptor already evaluated at the corresponding feature in the first image. This descriptor–feature pair is what we called a distinctive feature in our earlier classification. In this light, our attempt of combining distinctive and nondistinctive features amounts to using the matches of the descriptors of a few interest points to locate the matches for many interest points. This is sensible since the computation of a descriptor is expensive.

B. SIFT Features

We use SIFT features [22], [23] as our distinctive features. Automatic scale selection by scale-space extrema detection (see [24] for details) is what drives the interest point identification in SIFT. These "good" features [18] are called key points. Once the key points are identified, the accurate location and scale are determined. The next step is the orientation assignment. Finally, a local image-gradient-based descriptor, a 128-D vector, is calculated.

Given any continuous signal $f : \mathbb{R}^D \to \mathbb{R}$, its linear scale-space representation [24], [25] $L : \mathbb{R}^D \times \mathbb{R}_+ \to \mathbb{R}$ is equivalent to the convolution of the function (image) $f(\mathbf{x})$ with Gaussian kernels $g(\mathbf{x}; \sigma)$ of varying width σ

$$L(\mathbf{x}; \sigma) = g(\mathbf{x}; \sigma) * f(\mathbf{x}) \tag{1}$$

where $g : \mathbb{R}^D \to \mathbb{R}$ is given by

$$g(\mathbf{x}; \sigma) = \frac{1}{(2\pi\sigma^2)^{D/2}} \exp\left(-\frac{x_1^2 + \cdots + x_D^2}{2\sigma^2}\right) \tag{2}$$

and $\mathbf{x} = [x_1, \ldots, x_D]^{\mathbf{T}}$. In SIFT, local extrema detection in the stacked DOG images yield interest (key) points. Once the interest points are found, local gradients are distilled into histograms in order to compute the descriptors. This gives rise to the previously mentioned 128-D descriptor. This vector is normalized to unit length to suppress the effects of illumination changes.

C. Nondistinctive Features

We use DOG extrema and Harris and Stephens [21] corners as our nondistinctive features. We mentioned DOG features in Section II-B. Harris corners can be computed quickly [2], [13] using the second moment matrix. The multiscale version of the second moment matrix is given by [26]

$$\mu(\mathbf{x}, \sigma_I, \sigma_D) = \sigma_D^2 g(\mathbf{x}; \sigma_I) * \begin{bmatrix} L_x^2(\mathbf{x}, \sigma_D) & L_x L_y(\mathbf{x}, \sigma_D) \\ L_x L_y(\mathbf{x}, \sigma_D) & L_y^2(\mathbf{x}, \sigma_D) \end{bmatrix}. \tag{3}$$

Harris measure is

$$\text{cornerness} = \det \mu(\mathbf{x}, \sigma_I, \sigma_D) - \alpha \ \text{trace}^2 \mu(\mathbf{x}, \sigma_I, \sigma_D) \tag{4}$$

where $\alpha \in [0.04, 0.15]$, $\sigma_I = \gamma \times \sigma_D$, and $\gamma \in [\sqrt{2}, 2]$, typically. A point is declared an interest point if cornerness passes the Harris threshold and if it is a local maximum.

In summary, we use SIFT features (i.e., a part of DOG extrema combined with the SIFT descriptors) as our distinctive features and DOG extrema and Harris corners as our nondistinctive features. As a result, we have a selected number of distinctive features and a large number of nondistinctive features.

III. Multihomographic Feature Matching

In Section II, we outlined the existing features and detectors. For any of these features, matching is trivial if the image patches are not distorted. However, the appearances of image patches change depending on the camera position and parameters. Therefore, image patches used for feature matching are warped before they can be matched. Warping is done depending on the motion model. The motion model defines how a pixel in one image is related to another. Commonly used motion models are translational and affine. These models can be used at two different levels: local or global. For example, Shi and Tomasi [18] use a local affine model for matching and an implicit global translational model for tracking. In the translational model, pixels are assumed to undergo a uniform 2-D translation. The more useful affine model assumes rotation about the normal to the image plane and anisotopic scaling in addition to translation. In our system, we assume the more general projective motion model, where line parallelism is not preserved [27]. This is an important property in indoor images because of the presence of many planes (walls, ceiling, floor, furniture, etc.), some orthogonal to each other. As a result, the projection cannot be approximated by a global affine transformation. Therefore, we need to use projective transformations, one per plane (Fig. 2). We calculate each of these projective transformations as a planar homography, which is described in Section III-A. Although a homography transforms the features in the first image to the very vicinity of the features in the

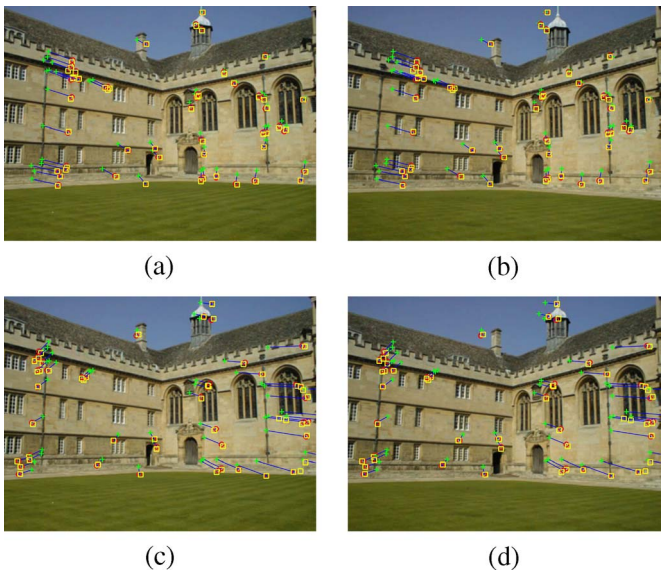


Fig. 2. Homographies for a scene with two major planes. Scene contains two major planes (ignoring the sky and grass). Feature motion using the first (H_1) and second (H_2) homographies shows that one homography is appropriate for only one plane. (a) shows the motion in the first image using H_1. (b) shows the same motion in the second image. (c) and (d) are for H_2. Notice the motion of features on the down sprouts of the gutter. [Cross (+)] Feature in the first image. [Circle (○)] Homography-predicted feature in the second image. [Square (□)] NCC match in the second image. (a) First image H_1. (b) Second image H_1. (c) First image H_2. (d) Second image H_2.

second image, an exact mapping is not possible. This is due to the fact that features encountered in indoor scenes may not always lie in perfect planes. Therefore, a planar homography cannot capture the exact motion. However, the homography mapping is close enough, and it is not unrealistic to carry out an NCC-like local-optimization matching at this stage. We describe this motion refinement process in Section IV. After refinement, the resulting matches are accurate.

A. Homographies

Homography calculation is an important step in our system. In relation to images, a homography is a projective transformation that projects each point $\mathbf{x}_i$ on one image I to $\mathbf{x}'_i$ on another I'. In other words, if we consider a set of point correspondences $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$, the problem is to compute the 3×3 matrix $\mathtt{H}$ such that $\mathtt{H}\mathbf{x}_i = \mathbf{x}'_i$ for each i [27]. This transformation has 8 degrees of freedom. Since a point correspondence provides two constraints, four such correspondences are sufficient to compute $\mathtt{H}$.

The aforementioned homography calculation process has to be done within RANSAC to eliminate the outliers. In order to use RANSAC, we need to pick a set of points at random called the random sample. We choose minimal sampling (four matches). We do sampling inspired by the guided maximum likelihood estimation by sampling consensus [28] based on the scores available in the SIFT matching. This score is based on the Euclidean distance between the SIFT vectors (Section II-B) of the two matching features in question. We rank the feature matches in the increasing order of this distance and pick matches with low scores first. A different method of computing motion models for piecewise planar scenes is by propagating the local affine transformations [29], [30] estimated in the affine covariant feature computation [26]. Algorithm 1 is the homography within RANSAC algorithm. The objective is to compute a homography based on a sample that has a good support (consensus) c. We approximate the inlier fraction by $\alpha = c/n$ where n is the number of matching pairs. Then, we can update the parameter k representing the number of iterations.

Algorithm 1 Homography within RANSAC
Require n number of matching pairs.
1: Set max_trials.
2: Initialize k with a large value (e.g., 10 000).
3: Initialize variables $d \leftarrow 0$, $trial \leftarrow 0$.
4: Initialize probability of seeing only bad samples z e.g., 0.9.
5: Initialize maximum consensus $c_m \leftarrow 0$.
6: **while** $k > d$ and $trial > max\_trials$ **do**
7: Initialize consensus $c \leftarrow 0$.
8: $trial \leftarrow trial + 1$.
9: Pick four point pairs using guided sampling.
10: Calculate $\mathtt{H}$.
11: Find c (transfer error $< \lambda$, e.g., 1.1).
12: **if** $c > c_m$ **then**
13: $c_m \leftarrow c$.
14: Store $\mathtt{H}$ (best $\mathtt{H}$ found until now).
15: $\alpha \leftarrow c/n$.
16: $k \leftarrow \log z/(\log(1 - \alpha^4))$.
17: $d \leftarrow d + 1$.
18: **end if**
19: **end while**

Algorithm 1 enables us to calculate the homography for a plane in two images. Transfer error is the criterion for declaring whether a point pair belongs to the consensus or not. Fig. 3 shows the accuracy of our homography calculation where image patches are transformed using these computed homographies.

There are two more points that we need to mention: First, our approach of computing a single homography is using RANSAC to find the initial model and consensus and then using a least squares computation. Torr and Murray [31] show that RANSAC-like random sampling robust estimators followed by expectation maximization (EM) estimators give the best results for computing motion models. However, we do not use an EM-like optimization, but stop after the least squares solution, favoring speed. Second, clustered outliers are a known weakness of RANSAC. Some outliers may group in a set of points which can be described by a homography or a degenerate version of it. This is a limitation of our motion model computation that needs to be investigated.

B. Multiple Homographies as Motion Segmentation

A single homography algorithm fails when there are several significantly different planes in the image, making a single homography extremely inappropriate as the motion model. For example, Fig. 2 shows two images of a scene with two major planes with scattered features (ignoring sky and grass). The feature transformation done using the first and second

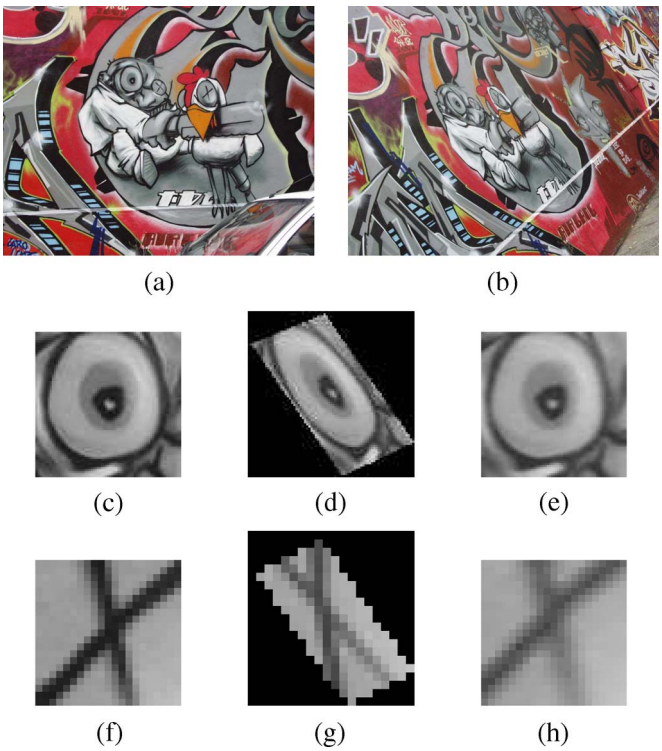


Fig. 3. Accuracy of homography calculation with first and fourth images in the Graffiti sequence. Row 1 is the original pair of images. Row 2 shows the square patch of the eye area of the first image, the homography-predicted region in fourth image, and the back-projected (BP) patch. Row 3 shows the same for the chicken's eye. Patches in image 1 and the BP patch are almost identical. This shows that our homography calculation is correct. (a) Image 1. (b) Image 4. (c) In image 1. (d) In image 4. (e) BP. (f) In image 1. (g) In image 4. (h) BP.

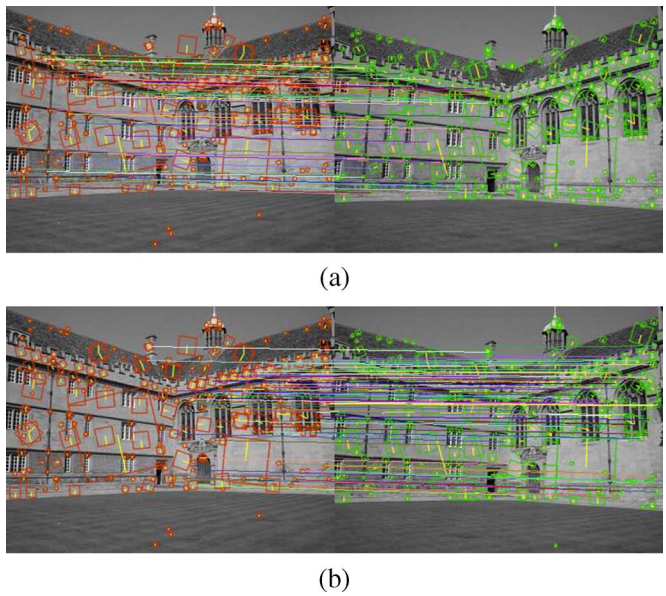


Fig. 4. Consensuses of two homographies. Rows 1 and 2 show the SIFT feature matches that give rise to consensuses (in RANSAC) corresponding to two homographies. Shown in row 1 is the consensus of homography corresponding to the left wall, and row 2 shows the same for the right wall (best viewed in color). (a) Consensus: Left wall. (b) Consensus: Right wall.

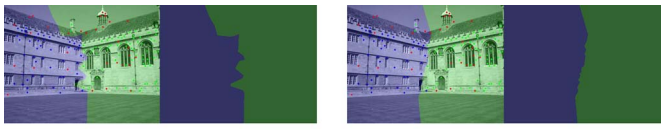
homographies is shown: The first corresponds to the features on the wall in the right half, and the second corresponds to the features on the wall in the left half. This shows that a homography could model only one plane. The solution to this problem is to estimate more than one homography representing the major motion planes. Here, motion segmentation literature helps us.

The problem that we have is clustering the points that conform to a single motion model (a homography) among many and accurately computing these motion models, without knowing the exact number of motion models (number of planes in our case). In other words, we have to simultaneously estimate the motion models and cluster the feature points. We can explain this process in terms of Torr's geometric motion segmentation and model selection concept [32]. In this work, Torr gives a general automatic motion segmentation and grouping algorithm which determines the number of motions in the scene, optimally selects the motion models for these different motions, and optimally computes the parameters of the motion models. The objective is maximizing the probability of the interpretation given the data. The log-likelihood of the mixture model of parameters is maximized using the EM algorithm. Torr selects the motion models based on the geometric robust information criteria, which are a combination of the error, dimension, and number of parameters in the model. In his method, the dimensions of the motion models can differ from each other. In our method, however, since we tailor our system for piecewise planar scenes (indoors), homographies are the only type of motion models. Therefore, we are not burdened with having to select motion models. However, determining the number of such motion models and robustly computing the parameters of each motion model are still unresolved.

When features supposedly conform to multiple motion models, the usual approach is to iterate between a model computation and subtraction of features. This enables the inclusion of a robust motion model computation like RANSAC. Torr [32] uses this in his geometric motion segmentation and model selection. Fitzgibbon and Zisserman also use this in their multibody structure and motion computation [33].[1] This is exactly the iteration between the dominant motion computation and data subtraction used by Rothganger *et al.* [35]. The number of motion models is determined automatically due to the termination criteria. When the number of matches remaining is too few, the iteration between dominant motion model computation and data subtraction terminates. Based on these, it is possible to calculate the first homography, subtract the consensus (mask the consensus), and calculate a second homography. When the consensus is masked, it cannot participate in calculating a new homography. Consequently, we can compute multiple homographies by masking the previous consensuses and reusing Algorithm 1 repetitively. Fig. 4 shows the consensuses corresponding to the left and right walls of the Wadham sequence we considered in Fig. 2. We also use the termination criteria of too few remaining matches, along with a predefined maximum number of homographies. Algorithm 2 summarizes this.

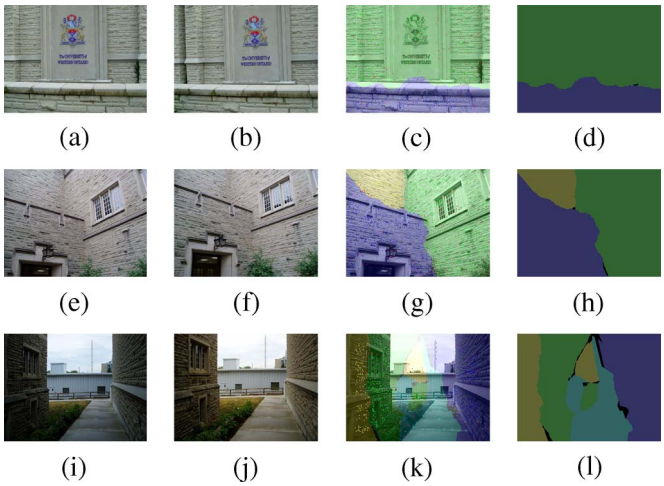
When we have the homographies, it is possible to determine the motion of a given pixel depending on the homography attached to its closest neighbors in consensuses. Let us call this the *closest homography*. We do this using the k-nearest

[1] Reference [34] is a different residual-distribution-based approach.



(a) (b) (c) (d)

Fig. 5. Closest homography using KNN. First and third columns show, using color tints, the pixels closest to neighbors in consensuses attached to a homography, using one- and seven-neighbor methods, respectively. Notice that the images in the Wadham sequence have two major planes and, therefore, two homographies. Second and fourth columns show the tints without the images for clarity. Images were obtained without guided sampling. (a) One-neighbor method. (b) Tint only. (c) Seven-neighbor method. (d) Tint only.



(a) (b) (c) (d)
(e) (f) (g) (h)
(i) (j) (k) (l)

Fig. 6. Closest homography using KNN (seven-neighbor method) showing several homographies. The University of Western Ontario (UWO) images at the top row have two planes. Wall and corridor images at the bottom two rows have three and four planes, respectively. See Fig. 5 for details. Images were obtained without guided sampling. (a) UWO 1. (b) UWO 2. (c) Seven-neighbor method. (d) Tint. (e) Wall 1. (f) Wall 2. (g) Seven-neighbor method. (h) Tint. (i) Corridor 1. (j) Corridor 2. (k) Seven-neighbor method. (l) Tint.

neighbors (KNN) algorithm [36]. In Figs. 5 and 6, we show the closest homography to each pixel using color coding.

Algorithm 2 Multihomographies
Require: n number of matching pairs.
1: Set maximum number of homographies $h_{\max}$.
2: Set consensus fraction threshold c_{thresh} (e.g., 0.8).
3: Initialize the sum of consensuses s to 0.
4: Initialize $mask$ of size n to $unmasked$.
5: Initialize number of homographies h to 0.
6: **while** $c_{thresh} > c$ and $h_{\max} > h$ **do**
7: Select point pairs that are not masked.
8: Use Algorithm 1 to compute a homography.
9: Find the consensus and increase s.
10: Mask out the consensus by setting the mask of appropriate point pairs.
11: Set $c \leftarrow$ ratio between s and n.
12: Increment h.
13: **end while**

IV. Motion Refinement

In Section III, we showed how to compute homographies. The purpose of this is to find the location in the second image of a given point in the first image. When homographies are available, they can be used to match the less distinctive features by transforming the corresponding patches. This means that every point in the image, whether an interest point or not, can be correctly matched if the corresponding homography is known. In other words, if our argument is correct, we will have a dense correspondence or optical flow. However, this argument is not correct.

The flaw in the aforementioned argument is due to the inability to model a real-world scene using a handful of homographies. Each planar homography is able to give us only the *approximate location* of features in the second image. A homography matches points on a plane with the transformed plane in the second image. There are two reasons for this transformed location error. First, the planes in the indoor scene are not perfect planes. For example, consider a door and its knob. Although both may belong to a single homography that models the whole wall containing the door, the door corners and the doorknob are not strictly coplanar. Therefore, if the homography corresponds to the door corners, the transformed location of the doorknob will not be exact. The second reason is linked to the first, although more implementation related than physical. In the homography calculation, we accept every point that falls within a certain error limit as conforming to the consensus. Therefore, homography calculation itself may contribute to the transformed location error.

The solution to the transformed location error is refining the homography-transformed location depending on the local intensity information. We call this process *motion refinement*. We characterize motion refinement in Section IV-A. It is important to note that such a motion refinement attempt only works for an interest point. In other words, textureless areas or areas without bidirectional gradient changes cannot be matched. This is the reason why interest points are detected in the first image as the first step.

A. Motion Models and Refinement

As mentioned in Section I, there are different motion models that describe how features move between two images. Assume that the first frame is I and the second is I', and a point in the first image is $\mathbf{x}$ and the corresponding point in the second image is $\mathbf{x}'$. With the brightness constancy assumption, we can write

$$I(\mathbf{x}) = I'\left(h(\mathbf{x})\right) \tag{5}$$

where $h(\cdot)$ captures the motion and $\mathbf{x}' = h(\mathbf{x})$. In this paper, motion estimations are only for a set of specific points which are indeed the interest points. As indicated previously, what we compute using our homographic motion model $h(\mathbf{x}) - \mathbf{x}$ itself is a good approximation of the motion. Due to the reasons described previously, we need a refined motion estimation. Let us denote the refined motion estimation by $\mathbf{v}$, assuming that it *is* the true motion for notational convenience. If we assume that the motion can indeed be refined to give the actual value, the refined motion $\mathbf{v} \neq h(\mathbf{x}) - \mathbf{x}$ in general. Therefore, the motion refinement problem is, given $\mathbf{x}$ in image I and homographic mapping $h(\cdot)$, to estimate the residual motion $\mathbf{v}^h$. Thus

$$\mathbf{v} = h(\mathbf{x}) - \mathbf{x} + \mathbf{v}^h. \tag{6}$$

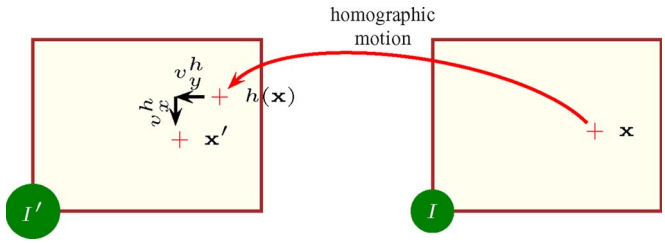


Fig. 7. Motion disparity between the homography transformed point and the actual feature. Point $\mathbf{x}$ is a feature in image I, and the corresponding feature in image I' is $\mathbf{x}'$. Homographic transformation maps $\mathbf{x}$ to $h(\mathbf{x})$ which does not fall exactly at $\mathbf{x}'$. $\mathbf{v}^h = [v_x^h, v_y^h]^{\mathrm{T}}$ is the residual motion to be estimated.

These quantities are shown in Fig. 7. We call this process "motion refinement," and one approach of estimating $\mathbf{v}^h$ is simple NCC. Since this is expensive, we choose a local-optimization approach.

B. Local-Optimization Approach

Here, we propose an approach which closely follows the KLT tracking equation which uses a local-optimization method. The essence of the method [17], [18] is to choose the motion parameters that minimize the dissimilarity between two image patches. Our method differs significantly due to the availability of the corresponding homography. The homography transforms the image patch of interest to the very vicinity of the matching patch in the second image. Our task is to find the smaller (compared to what KLT would give) motion change that would further minimize the dissimilarity. In this section, we first derive the KLT equations as applied to our system and then present our method, indicating the difference between the two.

1) KLT Tracker: The KLT method defines the dissimilarity between two feature windows (image patches) as the SSD over the windows. Feature tracking is essentially finding the displacement or motion. This motion is defined as the one that minimizes the SSD. Let us consider two image patches on two successive images denoted by $f^0(\mathbf{x}) = f(\mathbf{x}, t_0)$ and $f^1(\mathbf{x}) = f(\mathbf{x}, t_1)$. $\mathbf{x} = [x, y]^{\mathrm{T}}$ is the image coordinates vector, and t_0 and t_1, $t_1 > t_0$, are the time instances. For the correct motion vector $\mathbf{v}^{\mathrm{klt}} = [v_x, v_y]^{\mathrm{T}}$

$$f(\mathbf{x}, t_0) = f(\mathbf{x} + \mathbf{v}^{\mathrm{klt}}, t_1), \qquad \mathbf{x} \in \mathcal{W} \tag{7}$$

where $\mathcal{W}$ is a small feature window. The SSD error over the window $\mathcal{W}$ is given by

$$E(\mathbf{v}^{\mathrm{klt}}) = \int_{\mathcal{W}} \left[f(\mathbf{x} + \mathbf{v}^{\mathrm{klt}}, t_1) - f(\mathbf{x}, t_0)\right]^2 d\mathbf{x}. \tag{8}$$

KLT attempts to find the correct motion vector by minimizing $E(\mathbf{v}^{\mathrm{klt}})$.

It is possible to emphasize the motion of the central pixels by using a window function, such as a Gaussian window. For a Gaussian kernel of width $2w + 1$ using the notation from convolution with the 2-D integration variable $\boldsymbol{\tau}$

$$E(\mathbf{v}^{\mathrm{klt}}) = \int_{\mathcal{W}} \left[f(\mathbf{x} + \mathbf{v}^{\mathrm{klt}} - \tau, t_1) - f(\mathbf{x}, t_0)\right]^2 g(\tau) d\tau \tag{9}$$

where $g(\boldsymbol{\tau}) = g(\boldsymbol{\tau} - \mathbf{w}, \sigma_I)$ for a square Gaussian kernel with standard deviation σ_I and $\mathbf{w} = [w, w]^{\mathrm{T}}$. Let us use the notation in the following throughout the discussion for clarity of equations. Therefore, we mean (9) when we write

$$E(\mathbf{v}^{\mathrm{klt}}) = \int_{\mathcal{W}} \left[f(\mathbf{x} + \mathbf{v}^{\mathrm{klt}}, t_1) - f(\mathbf{x}, t_0)\right]^2 g d\mathbf{x}. \tag{10}$$

By using the Taylor expansion for $f(\mathbf{x} + \mathbf{v}^{\mathrm{klt}}, t_1)$, neglecting the second and higher order terms, and dropping the time parameter for clarity

$$E(\mathbf{v}^{\mathrm{klt}}) \approx \int_{\mathcal{W}} \left[f^1(\mathbf{x}) + \nabla f^1(\mathbf{x})^{\mathrm{T}} \mathbf{v}^{\mathrm{klt}} - f^0(\mathbf{x})\right]^2 g d\mathbf{x} \tag{11}$$

where

$$\nabla f^1(\mathbf{x}) = \left[\frac{\partial f^1}{\partial x} \quad \frac{\partial f^1}{\partial y}\right]^{\mathrm{T}} \tag{12}$$

is the image-gradient vector. Differentiating and equating to zero to minimize, we obtain

$$G\mathbf{v}^{\mathrm{klt}} = \mathbf{e} \tag{13}$$

where

$$G = \int_{\mathcal{W}} \nabla f^1(\mathbf{x}) \nabla f^1(\mathbf{x})^{\mathrm{T}} g d\mathbf{x}. \tag{14}$$

G is the 2×2 symmetric coefficient matrix. The 2-D vector $\mathbf{e}$ is given by

$$\mathbf{e} = -\int_{\mathcal{W}} \left[f^1(\mathbf{x}) - f^0(\mathbf{x})\right] \nabla f^1(\mathbf{x}) g d\mathbf{x}. \tag{15}$$

$\mathbf{v}^{\mathrm{klt}}$ can be computed if G is well conditioned. This occurs at an interest point where there are two dominant gradient directions resulting in two large eigenvalues. One important point to notice in (14) is that the gradients need to be calculated in only one image patch (the second patch $f^1(\mathbf{x})$ in the aforementioned formulation).

2) Homographic Tracker: In the implementation of our tracker, we use a similar formulation as in Section IV-B1. We have the homographies available, and our interest points are defined not only by a location but also by a scale parameter. The scale parameter ties an interest point to a level in the scale space. Therefore, using the definition in (1), we represent a scale-space image as

$$f(\mathbf{x}, \sigma, t) = L(\mathbf{x}; \sigma). \tag{16}$$

Now, we can denote two successive patches at the same scale as $f^1(\mathbf{x}, \sigma) = f(\mathbf{x}, \sigma, t_1)$ and $f^0(\mathbf{x}, \sigma) = f(\mathbf{x}, \sigma, t_0)$. Now, assume that a homographic motion model gives

$$f^{0h}(\mathbf{x}, \sigma) = f^0(h_3(\mathbf{x}), \sigma). \tag{17}$$

In fact, the homographic transformation accounts for the scale change of an interest point window as well. Therefore, it is sufficient to consider levels of the same scale. Thus, we use the

same scale level σ for both the patches. If motion refinement $\mathbf{v}^h$ is correct

$$f^{0h}(\mathbf{x},\sigma) = f^1(\mathbf{x}+\mathbf{v}^h,\sigma), \qquad \mathbf{x}\in\mathcal{W}. \tag{18}$$

Now, we can define the dissimilarity as

$$E(\mathbf{v}^h) = \int_{\mathcal{W}} \left[f^1(\mathbf{x}+\mathbf{v}^h) - f^{0h}(\mathbf{x})\right]^2 g d\mathbf{x}. \tag{19}$$

Using a similar computation as in (13)–(15), we can obtain the motion $\mathbf{v}^h$. We note that the v_x^h and v_y^h in vector $\mathbf{v}^h$ are smaller in magnitude than v_x^{klt} and v_y^{klt}. The relation is

$$\mathbf{v}^{\mathrm{klt}} \approx h(\mathbf{x}) - \mathbf{x} + \mathbf{v}^h. \tag{20}$$

Equation (20) relates the result of the KLT algorithm to our motion refinement. This assumes that KLT produces the correct result, i.e., $\mathbf{v}\approx\mathbf{v}^{\mathrm{klt}}$, where $\mathbf{v}$ is the true motion. One more advantage of this formulation is that, since under the assumption of correct homographic transformation, $|\mathbf{v}^h| \ll |\mathbf{v}|$, the omission of higher order terms in the Taylor series can be justified. This is an important aspect of our formulation. For example, the optical flow machinery of Papenberg *et al.* [37] postpones the linearization to the very end to achieve a highly accurate optical flow. Although their goal—dense optical flow computation—is not our goal, the performance of their algorithm clearly shows that, for an optimization approach that handles large motion to work, linearization of image functions are strictly to be avoided. In contrast, we can justify our linearized approach since we know that our motion $|\mathbf{v}^h|$ is much smaller that the actual feature motion.

There is an important computational advantage in the aforementioned asymmetric formulation of motion refinement. It is asymmetric since the motion refinement $\mathbf{v}^h$ is only in the argument of $f^1(\cdot)$ in (19). As a result, the computation only needs gradient $\nabla f^1(\mathbf{x})$ of $f^1(\mathbf{x})$, the untransformed image patch. If the scale-space gradients are available, which is the case in our system, gradients need not be recomputed to obtain $\nabla f^1(\mathbf{x})$.

3) Illumination Invariance: The aforementioned SSD-based formulation is not invariant with respect to affine changes in illumination such as $\alpha f(\mathbf{x})+\beta$. However, NCC is invariant. We verify this with an example in Fig. 8 and Table I. Fig. 8 shows two images taken with two different levels of exposure (they need not be so for this example). We acquire two corresponding patches f^1 and f^{0h} as we have shown in Fig. 3. Then, we apply an affine illumination transformation to f^{0h} and carry out SSD and NCC. Table I shows the values, and it is clear that NCC is invariant to affine illumination changes.

It is clear from the previous discussion that, in order to make our local-optimization approach illumination invariant, we need to consider normalized image patches. However, we want to avoid recomputing the gradients. Here, we derive the illumination-invariant formulation of local optimization. Let us denote the normalized patch of $f^1(\mathbf{x})$ by

$$\tilde{f}^1(\mathbf{x}) = \frac{f^1(\mathbf{x}) - \bar{f}^1}{f_n^1}, \qquad \mathbf{x}\in\mathcal{W} \tag{21}$$

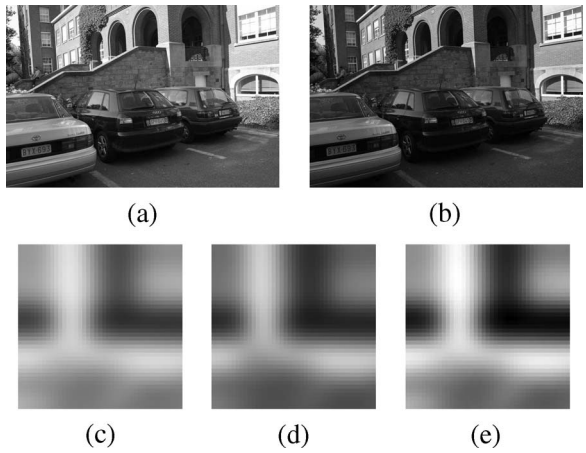


Fig. 8. Patches for comparing SSD and NCC. Top row shows the two images considered. Bottom row shows the patches considered. (e) shows the affine-transformed (in range) version of f^{0h}. See Table I for SSD and NCC values. (a) First image. (b) Second image. (c) Patch on first f^1. (d) Patch f^{0h}. (e) $0.8f^{0h}+0.2$.

TABLE I
NCC VERSUS SSD

Patches considered	SSD	NCC
f^1 and f^{0h}	24.53	0.9866
f^1 and $0.8f^{0h}+0.2$ (affine transformed f^{0h})	65.58	0.9866

Column 1 indicates the patches considered in Fig. 8. Columns 2 and 3 show the sum of squared differece (SSD) values and normalized cross correlation (NCC) values. $f^1(\mathbf{x}), f^{0h}(\mathbf{x}) \in [0,1]$. NCC is invariant to affine change in illumination, but SSD is not.

where

$$f_n^1 = \sqrt{\sum_{\mathcal{W}} (f^1(\mathbf{x}) - \bar{f}^1)^2} \tag{22}$$

is the zero-mean norm of $f^1(\mathbf{x})$. Let us denote the normalized patch of $f^{0h}(\mathbf{x})$ by

$$\tilde{f}^{0h}(\mathbf{x}) = \frac{f^{0h}(\mathbf{x}) - \bar{f}^{0h}}{f_n^{0h}}, \qquad \mathbf{x}\in\mathcal{W} \tag{23}$$

where

$$f_n^{0h} = \sqrt{\sum_{\mathcal{W}} \left(f^{0h}(\mathbf{x}) - \bar{f}^{0h}\right)^2} \tag{24}$$

the zero mean norm of $f^{0h}(\mathbf{x})$. In (21)–(24), $\bar{f}^1$ and $\bar{f}^{0h}$ denote the means of $f^1(\mathbf{x})$ and $f^{0h}(\mathbf{x})$ over $\mathcal{W}$, respectively. The objective function, similar to (19), is

$$E(\mathbf{v}^h) = \int_{\mathcal{W}} \left[\tilde{f}^1(\mathbf{x}+\mathbf{v}^{hn}) - \tilde{f}^{0h}(\mathbf{x})\right]^2 g d\mathbf{x}. \tag{25}$$

Finally, after simplifying, we can get the illumination-invariant homographic tracking equation

$$G\mathbf{v}^{hn} = \mathbf{e}^{hn} \tag{26}$$

where G is the same as in (14). The 2-D vector $\mathbf{e}^{hn}$ is given by

$$\mathbf{e}^{hn} = -\int_{\mathcal{W}} \left[\tilde{f}^1(\mathbf{x}) - \tilde{f}^{0h}(\mathbf{x})\right] \nabla f^1(\mathbf{x}) g d\mathbf{x}. \tag{27}$$

We can solve (26) for $\mathbf{e}^{hn}$ as long as G is well conditioned. If G is ill conditioned, we do not attempt to solve the equation but assume that $\mathbf{v}^{hn} = \mathbf{0}$ for speed concerns. To decide whether G is well conditioned, we simply apply a threshold to the determinant.[2] There is a significant computational advantage in using this formulation when we recall that the gradients of the scale-space images are already computed. G in (26) is computed using these gradients. We can compute $\mathbf{e}^{hn}$ in (27) by simply normalizing the patches extracted from the corresponding scale-space image. This justifies our formulation, in addition to its illumination invariance.

Algorithm 3 Multihomographic Feature Tracking

This high-level algorithm shows the major procedures that take place when a frame arrives. Processing is different for a key frame. Matching obviously needs two frames, and the system waits until the next frame comes after a key frame. However, we choose to ignore this in the algorithm for simplicity.

1: key frame ← true.
2: **while** there is a new frame **do**
  3: **if** key frame is true **then**
    4: Detect features (e.g., DOG, Harris) and compute descriptors (SIFT) for the selected.
    5: Find matches.
    6: Use Algorithm 2 to compute multihomographies and record the consensuses.
    7: Find the closest homography to each feature using KNN.
    8: Refine motion (Section IV-B3); compute NCC score.
    9: **if** NCC score $<$ NCC threshold **then**
      10: Exit.
    11: **end if**
    12: Recompute multihomographies using the same features used in Step 7.
    13: Run least squares homography on each consensus.
    14: Initialize Kalman filter (KF)s (only once if desired or for every key frame).
    15: Correct homographies using KFs.
    16: key frame ← false.
  17: **else**
    18: Propagate features using associated homographies (or detect features if desired).
    19: Refine motion (Section IV-B3); compute NCC score.
    20: **if** NCC score $<$ NCC threshold or frame is outside a multiple view window **then**
      21: key frame ← true.
    22: **end if**
    23: Recompute multihomographies using the consensus of each homography.
    24: Run least squares homography on each consensus.
    25: Correct homographies using KFs.
    26: **end if**
    27: Initialize or augment tracks.
  28: **end while**

[2]We choose to avoid eigenvalue ratio computation to estimate the condition numbers [38].

TABLE II
SYSTEM PARAMETERS

Name	Algorithm	Value/s	Typical value
max_trials	1	200	few trials < 10
transfer error λ	1	[0.1, 4.1]	0.1
maximum homographies h_{max}	2	[3,10]	3 for most cases if known
consensus threshold c_{thresh}	2	[0.7, 0.9]	0.9
NCC threshold	3	0.7	
multiple view window	3	[5,20]	9

First column indicates the parameters used in the algorithms (column two). Column three indicates the values that we used, and column four lists the typical values or the values typically picked by the system.

TABLE III
NCC COMPARISON FOR ILLUMINATION INVARIANCE

	Unrefined		Un-normalized		Normalized	
	Mean	Std.	Mean	Std.	Mean	Std.
G1 and G2, 172 patches	.9487	.1708	.9682	.1710	**.9686**	.1685
G1 and G3, 172 patches	.7724	.3454	.8386	.2983	**.8408**	.3006
L1 and L2, 59 patches	.9858	.0119	.9569	.0246	**.9891**	.0102
L1 and L6, 18 patches	.8893	.0821	.5156	.2571	**.9050**	.1084

This table compares normalized cross correlation (NCC) and standard deviation between patches with different motion estimation techniques. Unrefined: homographic transformation only with no motion refinement. Un-normalized: homographic with motion refinement with un-normalized patches. Normalized: homographic with motion refinement with normalized patches. In the normalized case, the motion estimation is illumination invariant. Graffiti images (G) have no noticeable illumination change, but Leuven images (L) do. Homographic motion itself is quite accurate for these images. Illumination invariant motion refinement (normalized) is advantageous when illumination differences are present. Performance is comparable (if not better) when there is no illumination difference.

We maintain the multihomographies within KFs, using the standard KF. For each homography, there is a KF. This way, we are able to reduce the number of times the full feature detection, matching, and homography computation (full computation) are carried out. This is possible only if the motion continuity assumption is valid. When this assumption is violated, the NCC score makes the system do a full computation. For a key frame pair, we carry out a full computation. For other pairs, we make use of the KF maintained homographies. In our work, both the state and the measurement are the entries of a homography. Therefore, the state transition matrix is the identity matrix. In the update stage, we need a measurement. The homography computed using the point matches after motion refinement acts as the measurement. Therefore, the measurement matrix is the identity matrix as well. Our use of the KFs is different from having all the points as the state of the filter; we only maintain a few homographies. The two major steps which consume computational power are the feature detection and matching steps. The computational complexity of our feature detection is linear in terms of pixels, and feature matching is quadratic in

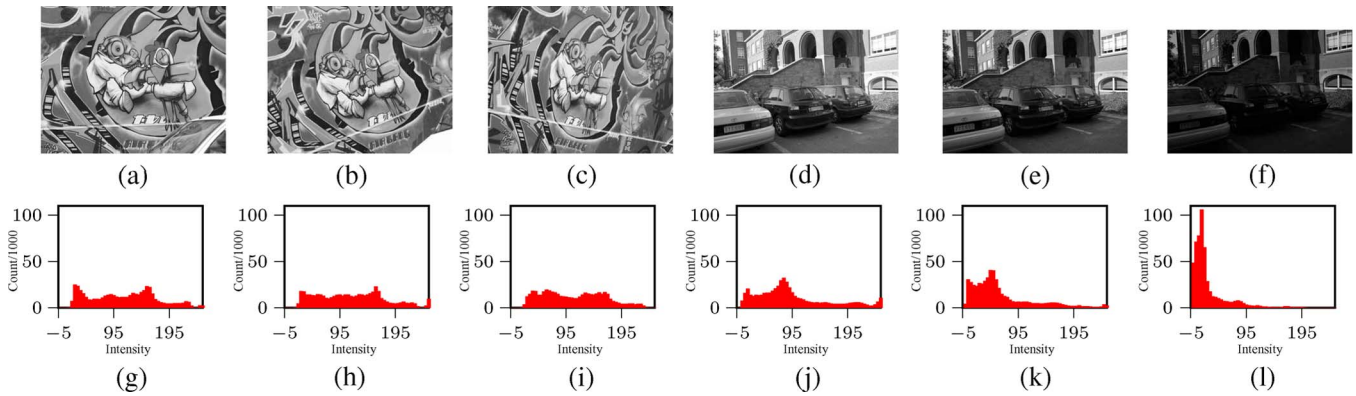


Fig. 9. Graffiti (G) and Leuven (L) images, and histograms showing illumination levels. Note the significant illumination difference between L images. (a) Image G1. (b) Image G2. (c) Image G3. (d) Image L1. (e) Image L2. (f) Image L6.

terms of the number of features. Therefore, the aforementioned mechanism helps us speed up the system.

In summary, we make use of a precomputed homography to transform the window corresponding to an interest point in the first onto the second image. Experimental evidence shows the need for illumination-invariant motion estimation that would refine the homographic motion. We minimize the dissimilarity of the intensities in an illumination-invariant manner. This gives a more accurate location, leading to better motion parameters. The noise performance of this motion refinement shows an improvement of approximately 10%–20% over the unrefined (homographic) motion (see Section V). We summarize the overall procedure in Algorithm 3.

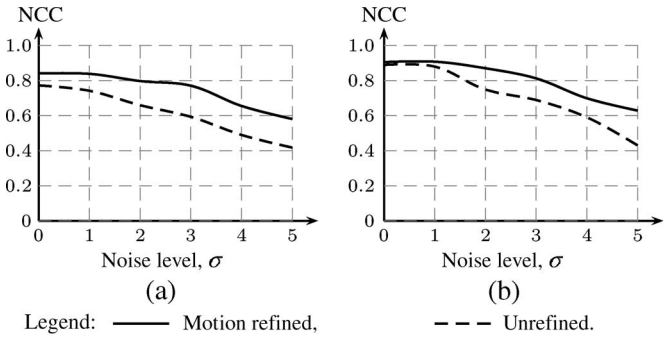


Fig. 10. Noise performance of our illumination-invariant motion refinement. (Solid line) Our method is approximately 10%–20% better than (dashed line) unrefined estimation. (a) Graffiti 1 and 3. (b) Leuven 1 and 6.

V. Results

In this section, we will show that our tracker effectively and robustly tracks. We will also compare our tracker with recent trackers. We will first show the results that establish the illumination invariance. Then, we will show the performance of our system with several indoor sequences and an outdoor sequence. This is followed by a comparison of our system against SIFT-only tracking and KLT in terms of frame rate, NCC score, and robustness. We then compare our system with several recent feature-tracking methods to show that our tracker achieves robust tracking without fully sacrificing the speed. Supplementary material and image sequences are available at http://iris.uwo.ca.

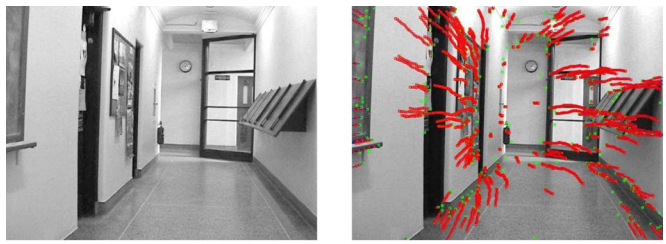


Fig. 11. Continuous sequence tracking images. Figure shows the first image of the sequence and the tracks. The camera is moving forward along the corridor while slightly turning. Tracking a continuous sequence shows good results. (a) First image. (b) Tracks drawn on first image.

A. System Parameters

We use the values listed in Table II in all our experiments, unless noted otherwise.

B. Illumination Invariance

We test the illumination invariance by computing the refined motion with and without illumination invariance. Table III compares NCC values between patches with different motion estimation techniques along with corresponding standard deviation values. Unrefined columns show values just after homographic transformation without motion refinement. Unnormalized columns are after motion refinement with unnormalized patches. Normalized columns show values after motion refinement with normalized patches. In the normalized case, the motion estimation is illumination invariant. The two image pairs are from the Graffiti and Leuven sequences [39] (see Fig. 9 for the images and their histograms). Graffiti images

TABLE IV
Tracking Results

Sequence	Tracks	Frames	NCC	FR
Somerville House (SH) 640 × 480	110	120	.972	2.7
Kinesiology Build. (KB) 640 × 480	155	111	.974	2.6
Car Video (CV) 360 × 240	125	225	.955	6.8
Medical Center (MC) 640 × 480	327	124	.952	2.1
University College (UC) 640 × 480	203	64	.956	2.3
Marble Block (MB) 368 × 366	766	15	.994	1.7

Table shows the name of the sequence, number of tracks (only for refined features) number of frames, the average NCC score, and the frame rate (FR). The size of the multiple view window was 16. See Fig. 14 for images.

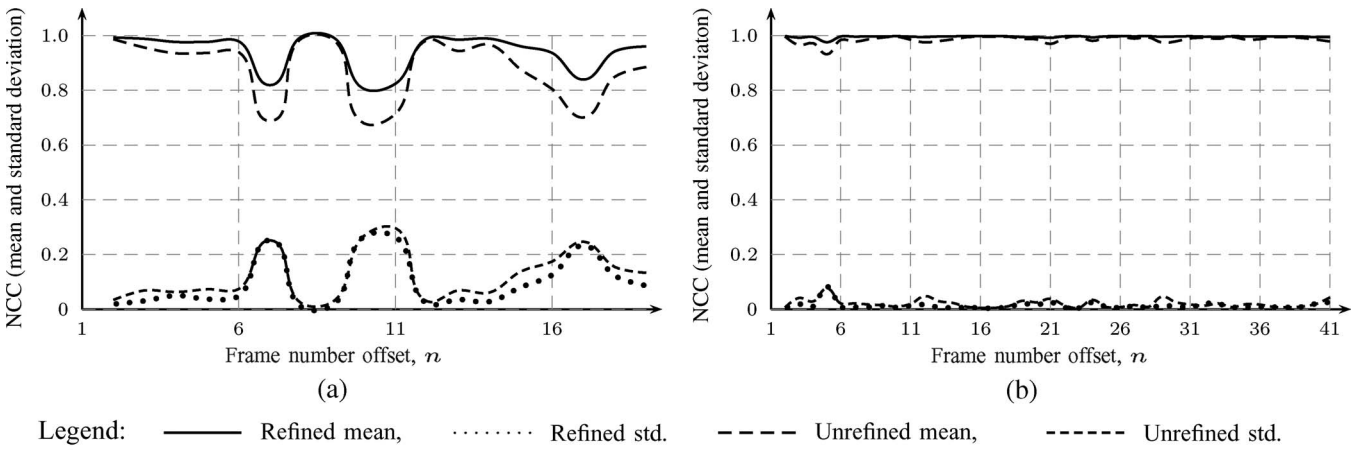


Fig. 12. Tracking with and without motion continuity. Graphs show in terms of NCC values that continuous tracking for a sequence with bad (left—MC sequence) and good (right—UC sequence) motion continuity. Here, we used a window of nine frames in key framing and an NCC threshold of 0.7. Good NCC values show that both the cases are handled, with better results for continuous motion case. We also notice that, in spite of the motion discontinuity at frame 7 of the MC sequence, tracking continues successfully. (a) Discontinuous motion. (b) Continuous motion.

have no noticeable illumination change, but Leuven images do. In general, results show that homographic motion itself is quite accurate. Moreover, illumination-invariant motion refinement (normalized) is advantageous when illumination differences are present. Performance is comparable (if not better) when there is no illumination difference. This verifies the utility of our illumination-invariant motion refinement.

Fig. 10 shows illumination-invariant homographic motion refinement with unrefined motion in terms of the noise performance. We perturb the location of the second patch f^1 by a random (2-D) vector of $\mathbf{0}$ mean and standard deviation $[\sigma, \sigma]^{\mathbf{T}}$. The figure shows graphs with illumination-invariant motion refinement and with no motion refinement. We conclude that our motion refinement is correct and can robustly handle noisy matches.

C. Continuous Tracking

Fig. 12 shows that, with the introduction of key framing, our multihomographic tracker can continuously track in spite of drastic motion discontinuities in terms of the NCC score and standard deviation. Fig. 12(a) is for the Medical Center (MC) sequence which has a drastic motion discontinuity, and Fig. 12(b) is for the University College (UC) sequence (Fig. 11). See Table IV for details about these sequences. There is a drastic motion discontinuity in MC showing effects from frame 6. This result shows that, using the NCC score as a measure of motion discontinuity, we are able to carry out continuous feature tracking in spite of drastic changes in motion. We tracked features for the UC sequence with good motion continuity, and the results show good performance. Fig. 11 shows the first frame and the feature tracks drawn on the first frame. Fig. 12(b) shows the good performance in terms of NCC. On average, there were approximately 200 features.

So long as the scene can be modeled using multihomographies, our tracker can perform reliably. Therefore, even for a sequence with moving objects, the tracker should work. We

Fig. 13. Tracking with moving objects. Figure shows the first image of the sequence and the tracks. Tracking with a moving background and an object shows good results. (In this portion of the sequence, both the car and background occupy significant areas.) (a) First image. (b) Last image. (c) Tracks.

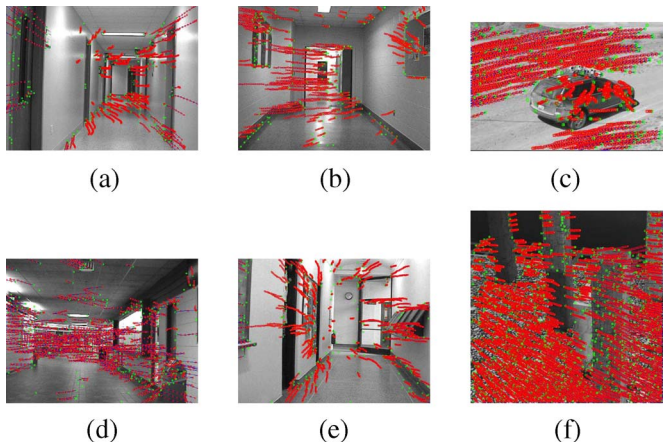


Fig. 14. Representative frames corresponding to the results in Table IV. (a) Somerville House (SH). (b) Kinesiology Building. (c) Car Video. (d) MC. (e) UC. (f) Marble Block.

tested our tracker with a car sequence[3] in which a rotating camera tracks a moving car. Fig. 13 shows the first image we used in the car sequence and the tracks. We tracked, on average, approximately 100 features over 40 frames. The multiple view window was 9, and the average NCC score was 0.98. This shows that our tracker works as expected for scenes with moving background and objects as long as they are rigid. We summarize more tracking sequences in Table IV (Fig. 14).

[3]This was downloaded from http://robotics.stanford.edu/dstavens/cs223b.

TABLE V
COMPARISONS WITH OTHER WORKS

Author/s	Space	Method	Motion Range	Baseline	Size	Features	No.	Robust?	FR
Zhu *et al.* [41] 2001	2D-2D	SVM regression in place of RANSAC, edge template matching			320 × 240	Harris-like	32	Yes (affine)	2Hz
Saeedi *et al.* [42] 2003	2D–3D	Search window 70 × 70, NCC, KF for features	180cm		320 × 240	Harris-like		No	2.8Hz
Nistér [13] 2004	2D-3D	Calibrated, search window 3%–30%, 11 × 11 NCC	600m	28cm	720 × 240	Fast Harris		No	13Hz
Royer *et al.* [15] 2005	2D-3D	Calibrated, off-line SFM, NCC marching, hierarchial bundle adjustment, No KF	500m		320 × 240	Harris	1500	No	Offline
Mouragnon *et al.* [2] 2006	2D-3D	Calibrated SFM, NCC, local bundle adjustment, No KF	70m		512 × 384	Harris	400	No	7.5Hz
Mejías *et al.* [43] 2006	2D–3D	KLT, KF used	Several meters	Small to medium		Corners	4	No	20Hz
Chen and Birchfield [44] 2006	2D-2D	KLT, No KF	100m	Small	320 × 240	KLT-like	50	No	5Hz
Kim and Kweon [40] 2007	2D-3D	1-D KLT, minimize SSD, No KF	50m	Medium	320 × 240	KLT, SIFT	400	No	8.5Hz
Our method	2D–2D	Multi-homographies, KLT, KF for homographies	10–100m	Small to medium	640 × 480	DoG, Harris	240	Yes	2.39Hz

Table compares our method with several recent feature trackers. No. represent the number of feature tracked. FR is the frame rate. We used a Core 2, 2.4 GHz PC, and our software is written in C++ in a single thread. Although inferior to Nistér [13], and Kim and Kweon [40] which use stereo heads for reported results, our frame rates are comparable with recent approaches. Robustness is important: If the features do not get falsely matched we term the tracker robust. Our tracker is the only robust tracker which can handle multiple planes.

TABLE VI
COMPARISON WITH SIFT-ONLY TRACKING AND KLT

Method	Seq.	MVW	Features	NCC	STD	FR
SIFT	UC		261	.996	.013	0.77
Ours	UC	15	240	.993	.018	**2.39**
KLT	UC		249	.980	.033	3.30
SIFT	KB		210	.997	.005	0.91
Ours	KB	15	195	.993	.015	**2.26**
KLT	KB		201	.967	.055	3.45
SIFT	MB		824	.998	.007	0.37
Ours	MB	15	766	.994	.022	**1.67**
KLT	MB		763	.993	.027	2.61

Table shows the average NCC score, its standard deviation, and the frame rate (FR) for the UC, KB, and MB sequences. Approximately 250 features were tracked across approximately 40 frames. Computational time is reasonable compared to KLT. This is low for SIFT only tracking.

D. Comparisons

We compare several recent tracking methods in Table V. When we consider the frame rate, our tracker performs better than others in the category of robust trackers. The trackers that show higher frame rates are all nonrobust. They either use a stereo head or track only a few feature points.

We compare our tracker with KLT and SIFT-only tracking in Table VI.[4] Our frame rates are quite superior than SIFT-only tracking and reasonable (70% for some cases) compared to the (affine corrected version) of KLT. KLT can easily get carried away by local minima. However, our method is not usually affected this way since it is guided by the global motion models. We show this in Fig. 15 using the marble block sequence.[5]

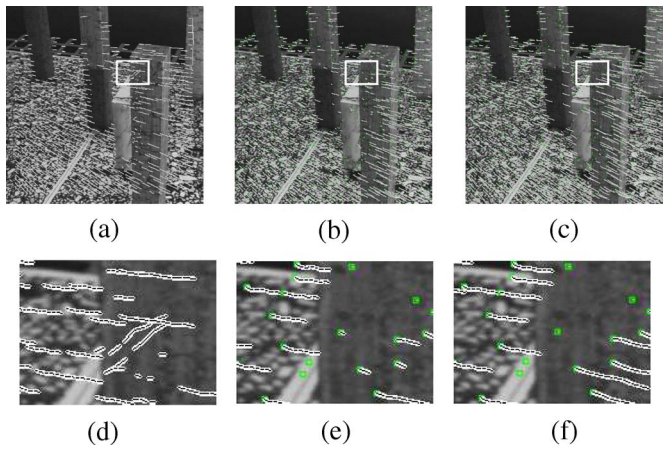


Fig. 15. Visual comparison of tracks shows that the proposed method is robust. The general motion of pixels is approximately from left to right. The tracks and a zoomed-in portion are shown. SIFT and our method produce robust tracking. However, for some features, KLT fails, evident from incorrect tracks (moving upward) in zoomed-in regions. (a) KLT. (b) SIFT. (c) Ours. (d) KLT—zoomed. (e) SIFT—zoomed. (f) Ours—zoomed.

In summary, our tracker is able to robustly track features in piecewise planar scenes, with or without moving objects. The timing performance is reasonable.

VI. CONCLUSION

We proposed a tracker which uses global motion models strategically computed using distinctive features that guide the motion of a large number of features. These motion parameters are refined using a local-optimization method to achieve accurate motion. Our major achievement in this work is establishing that multihomographic feature tracking is feasible. While reaching this goal, we developed our robust feature tracker that we described previously. Proposing a multihomographic global motion model for feature motion, deriving and

[4]In our system, we do the motion refinement based on the image patches from the current and previous frames, and compute the NCC based on these. However, KLT carries out the local optimization based on the first and current frames. Therefore, if we report the NCC scores based on the NCC computed using the previous and current frames, KLT scores low values because that is not where the optimization is done. Thus, we choose to report NCC scores for KLT based on the first and current frames.

[5]This was obtained from http://i21www.ira.uka.de/image_sequences.

validating an illumination-invariant local-optimization method for motion computation and refinement, and using a combination of distinctive and nondistinctive features are the major contributions.

Our system performs accurate feature tracking for distinctive and nondistinctive features, as evidenced by good values. This is due to our multihomographic motion model and illumination-invariant motion refinement. We showed that our method is robust and performs tracking at reasonable frame rates.

ACKNOWLEDGMENT

The authors would like to thank Y. Boykov and K. Ranaweera for the insightful discussions.

REFERENCES

[1] Z. Chen, J. Samarabandu, and R. Rodrigo, "Recent advances in simultaneous localization and map-building using computer vision," *Adv. Robot.*, vol. 21, no. 3/4, pp. 233–265, Mar./Apr. 2007.

[2] E. Mouragnon, M. Lhuillier, M. Dhome, F. Dekeyser, and P. Sayd, "Real time localization and 3D reconstruction," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recog.*, New York, Jun. 2006, vol. 1, pp. 363–370.

[3] S. Baker and I. Matthews, "Lucas-Kanade 20 years on: A unifying framework," *Int. J. Comput. Vis.*, vol. 56, no. 3, pp. 221–255, Feb./Mar. 2004.

[4] R. Rodrigo and J. Samarabandu, "Monocular vision for robot navigation," in *Proc. IEEE Int. Conf. Mechatronics Autom.*, Niagara Falls, ON, Canada, Jul. 2005, pp. 707–712.

[5] K. Nickels and S. Hutchinson, "Model-based tracking of complex articulated objects," *IEEE Trans. Robot. Autom.*, vol. 17, no. 1, pp. 28–36, Feb. 2001.

[6] D. J. Jacques, R. Rodrigo, K. A. McIsaac, and J. Samarabandu, "An object tracking and visual servoing system for the visually impaired," in *Proc. IEEE Int. Conf. Robot. Autom.*, Barcelona, Spain, Apr. 2005, pp. 3510–3515.

[7] J. Sivic, B. C. Russel, A. A. Efros, A. Zisserman, and W. T. Freeman, "Discovering objects and their location in images," in *Proc. 11th IEEE Int. Conf. Comput. Vis.*, Beijing, China, Oct. 2005, pp. 370–377.

[8] M. Brown and D. Lowe, "Recognising panoramas," in *Proc. 9th Int. Conf. Comput. Vis.*, Nice, France, Oct. 2003, pp. 1218–1225.

[9] P. David, D. DeMenthon, R. Duraiswami, and H. Samet, "Softposit: Simultaneous pose and correspondence determination," *Int. J. Comput. Vis.*, vol. 59, no. 3, pp. 259–284, Sep. 2004.

[10] Y. Lu, J. Z. Zhang, Q. M. J. Wu, and Z.-N. Li, "A survey of motion-parallax-based 3-D reconstruction algorithms," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 34, no. 4, pp. 532–548, Nov. 2004.

[11] G. N. DeSouza and A. C. Kak, "Vision for mobile robot navigation: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 2, pp. 237–267, Feb. 2002.

[12] S. Se, D. Lowe, and J. Little, "Vision-based global localization and mapping for mobile robots," *IEEE Trans. Robot.*, vol. 21, no. 3, pp. 364–375, Jun. 2005.

[13] D. Nistér, O. Naroditsky, and J. Bergen, "Visual odometry," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recog.*, Washington, DC, Jun./Jul. 2004, vol. 1, pp. 652–659.

[14] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, Jun. 1981.

[15] E. Royer, M. Lhuillier, M. Dhome, and T. Chateau, "Localization in urban environments: Monocular vision compared to a differential GPS sensor," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recog.*, San Diego, CA, Jun. 2005, vol. 1, pp. 114–121.

[16] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Proc. 7th Int. Joint Conf. Artif. Intell.*, Vancouver, BC, Canada, Aug. 1981, pp. 674–679.

[17] C. Tomasi and T. Kanade, "Detection and tracking of point features," Carnegie Mellon Univ., Pittsburgh, PA, Tech. Rep. TR 92-1270 and CMU–CS–91–132, Apr. 1991.

[18] J. Shi and C. Tomasi, "Good features to track," in *Proc. Conf. Comput. Vis. Pattern Recog.*, Jun. 1994, pp. 593–600.

[19] H. Jin, P. Favaro, and S. Soatto, "Real-time feature tracking and outlier rejection with changes in illumination," in *Proc. 8th IEEE Int. Conf. Comput. Vis.*, Los Alamitos, CA, Jul. 2001, pp. 684–689.

[20] A. Fusiello, E. Trucco, T. Tommasini, and V. Roberto, "Improving feature tracking with robust statistics," *Pattern Anal. Appl.*, vol. 2, no. 4, pp. 312–320, Nov. 1999.

[21] C. Harris and M. J. Stephens, "A combined corner and edge detector," in *Proc. 4th Alvey Vis. Conf.*, Manchester, U.K., Aug. 1988, pp. 147–152.

[22] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, Nov. 2004.

[23] D. G. Lowe, "Object recognition from local scale-invariant features," in *Proc. 7th IEEE Int. Conf. Comput. Vis.*, Kerkyra, Greece, Sep. 1999, vol. 2, pp. 1150–1157.

[24] T. Linderberg, "Principles for automatic scale selection," Dept. Numerical Anal. Comput. Sci. KTH (Roy. Inst. Technol.), Stockholm, Sweden, Tech. Rep. ISRN KTH/NA/P–98/14–SE, 1998.

[25] T. Linderberg, *Scale-Space Theory in Computer Vision*. Norwell, MA: Kluwer, 1994.

[26] K. Mikolajczyk and C. Schmid, "Scale and affine invariant interest point detectors," *Int. J. Comput. Vis.*, vol. 60, no. 1, pp. 63–86, Oct. 2004.

[27] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge, U.K.: Cambridge Univ. Press, 2003.

[28] B. J. Tordoff and D. W. Murray, "Guided-MLESAC: Faster image transform estimation by using matching priors," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 10, pp. 1523–1535, Oct. 2005.

[29] O. Choi, H. Kim, and I. S. Kweon, "Simultaneous plane extraction and 2D homography estimation using local feature transformations," in *Proc. Asian Conf. Comput. Vis.*, Tokyo, Japan, Jun. 2007, pp. 269–278.

[30] J. Kannala and S. S. Brandt, "Quasi-dense wide baseline matching using match propagation," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recog.*, Minneapolis, MN, Jun. 2007, pp. 1–8.

[31] P. Torr and D. Murray, "The development and comparison of robust methods for estimating the fundamental matrix," *Int. J. Comput. Vis.*, vol. 24, no. 3, pp. 271–300, Sep. 1997.

[32] P. H. S. Torr, "Geometric motion segmentation and model selection," *Philos. Trans. Roy. Soc. London A, Math. Phys. Sci.*, vol. 356, no. 1740, pp. 1321–1340, 1998.

[33] A. W. Fitzgibbon and A. Zisserman, "Multibody structure and motion: 3-D reconstruction of independently moving objects," in *Proc. Eur. Conf. Comput. Vis.*, Dublin, Ireland, Jun./Jul. 2000, vol. 1842, pp. 891–906.

[34] W. Zhang and J. Kosecká, "Nonparametric estimation of multiple structures with outliers," in *Proc. Workshop Dynamical Vis., Eur. Conf. Comput. Vis.*, Graz, Austria, May 2006, vol. 4358, pp. 60–74.

[35] F. Rothganger, S. Lazebnik, C. Schmid, and J. Ponce, "Segmenting, modeling, and matching video clips containing multiple moving objects," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 3, pp. 477–491, Mar. 2007.

[36] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*, 2nd ed. New York: Wiley-Interscience, 2001.

[37] N. Papenberg, A. Bruhn, T. Brox, S. Didas, and J. Weickert, "Highly accurate optic flow computation with theoretically justified warping," *Int. J. Comput. Vis.*, vol. 67, no. 2, pp. 141–158, Feb. 2006.

[38] G. H. Golub and C. F. Van-Loan, *Matrix Computations*. Baltimore, MD: Johns Hopkins Univ. Press, 1983.

[39] K. Mikolajczyk and C. Schmid, "A performance evaluation of local descriptors," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 10, pp. 1615–1630, Oct. 2005.

[40] J. Kim and I.-S. Kweon, "Robust feature matching for loop closing and localization," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, San Diego, CA, Oct. 2007, pp. 3905–3910.

[41] W. Zhu, S. Wang, R.-S. Lin, and S. Levinson, "Tracking of object with SVM regression," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recog.*, Kauai, HI, Dec. 2001, vol. 2, pp. 240–245.

[42] P. Saeedi, D. G. Lowe, and P. D. Lawrence, "3D localization and tracking in unknown environments," in *Proc. IEEE Int. Conf. Robot. Autom.*, Taipei, Taiwan, Sep. 2003, vol. 1, pp. 1297–1303.

[43] L. Mejías, P. Campoy, S. Saripalli, and G. S. Sukhatme, "A visual servoing approach for tracking features in urban areas using an autonomous helicopter," in *Proc. IEEE Int. Conf. Robot. Autom.*, Orlando, FL, May 2006, vol. 1, pp. 2503–2508.

[44] Z. Chen and S. T. Birchfield, "Qualitative vision-based mobile robot navigation," in *Proc. IEEE Int. Conf. Robot. Autom.*, Orlando, FL, May 2006, pp. 2686–2693.

Ranga Rodrigo received the B.Sc.Eng. degree (first class honors) from the University of Moratuwa, Moratuwa, Sri Lanka, in 2000 and the M.E.Sc. and Ph.D. degrees from The University of Western Ontario, London, ON, Canada, in 2004 and 2008, respectively.

He has been with the Department of Electronic and Telecommunication Engineering, University of Moratuwa since January 2008, where he is a Senior Lecturer. His research interests are in the general area of computer vision. In particular, he works in feature tracking, reconstruction, and optical flows.

Mehrnaz Zouqi received the B.Sc. degree in electrical engineering from Amirkabir University of Technology, Tehran, Iran, in 2001. She is currently working toward the M.E.Sc. degree in electrical engineering at The University of Western Ontario, London, ON, Canada.

From 2001 to 2005, she held positions in the field of electrical engineering and software development for telecommunication and automation systems with the Iran Telecommunication Research Center and ParsArc Company Ltd., Tehran, Iran. Her current research interests are in the fields of image processing and computer vision with focus on image segmentation in biomedical image processing.

Zhenhe Chen received the B. E. degree in electrical engineering from South China University of Technology, Guangzhou, China, in 1996, the M.A.Sc. degree in electrical and computer engineering from Ryerson University, Toronto, ON, Canada, in 2003, and the Ph.D. degree in electrical and computer engineering from the University of Western Ontario, London, ON, in 2007.

From 1996 to 2000, he was with the State China Administration of Taxation as a Project Coordinator of the Golden Tax of the Chinese government. He has been with TVWorks, London, since 2007. He has been a reviewer for journals and conferences in his area of research. His research interests are robot navigation within probabilistic frameworks, and computer vision and video segmentation by independent component analysis.

Dr. Chen is a registered Professional Engineer in Ontario. He was a recipient of the Precarn Scholarship in 2006.

Jagath Samarabandu (M'92) received the B.Sc.Eng. degree (first class honors) in electronic and telecommunication from the University of Moratuwa, Moratuwa, Sri Lanka, in 1982 and the M.S and Ph.D. degrees in electrical engineering from the State University of New York, Buffalo, in 1990 and 1994, respectively.

He has been with The University of Western Ontario, London, ON, Canada, since 2000, where he is currently an Associate Professor with the Department of Electrical and Computer Engineering. His research interests are image processing, biomedical imaging and visualization, computer vision, artificial intelligence, and image understanding.

Dr. Samarabandu was the recipient of a Fulbright scholarship in 1987 for postgraduate study.