Pose and Motion Recovery from Feature Correspondences and a Digital Terrain Map

Ronen Lerner, Ehud Rivlin, *Member*, *IEEE*, and Héctor P. Rotstein, *Member*, *IEEE*

Abstract—A novel algorithm for pose and motion estimation using corresponding features and a Digital Terrain Map is proposed. Using a Digital Terrain (or Digital Elevation) Map (DTM/DEM) as a global reference enables the elimination of the ambiguity present in vision-based algorithms for motion recovery. As a consequence, the absolute position and orientation of a camera can be recovered with respect to the external reference frame. In order to do this, the DTM is used to formulate a constraint between corresponding features in two consecutive frames. Explicit reconstruction of the 3D world is not required. When considering a number of feature points, the resulting constraints can be solved using nonlinear optimization in terms of position, orientation, and motion. Such a procedure requires an initial guess of these parameters, which can be obtained from dead-reckoning or any other source. The feasibility of the algorithm is established through extensive experimentation. Performance is compared with a state-of-the-art alternative algorithm, which intermediately reconstructs the 3D structure and then registers it to the DTM. A clear advantage for the novel algorithm is demonstrated in variety of scenarios.

Index Terms—Pose estimation, vision-based navigation, DTM, structure from motion.

1 Introduction

This paper deals with the problem of estimating the *pose* (i.e., location and orientation) and motion of a calibrated camera from multiple views of a scene. As opposed to most of the computer-vision literature dealing with the subject, the starting point in this work is that, in addition to having pairs of corresponding features from two consecutive frames, the location, orientation, and motion of the camera are *approximately* known and a model of the scene is available in the form of a Digital Terrain (or Elevation) Map DTM/DEM. The main objective of the work is to show that the a priori solution can be improved by using the extra information provided by the correspondence pairs and the DTM.

The current research is motivated by the intrinsic difficulties involved in maintaining a good estimate of pose and motion. Indeed, in the seminal work [24], it was shown that, for calibrated camera motion, reconstruction was possible only up to a similarity transformation, which meant not only that the absolute position/orientation could not be estimated, but also that the scene structure and camera motion could only be recovered up to a certain scale. After two decades of intense research, a number of theoretical results, algorithms, and experimental results are now available for solving the Structure From Motion (SFM) problem under various scenarios. For example, [14] contains a thorough discussion of theoretical issues associated with multiple view geometry, [2], [4], [8], [10], [11], [15], [19], [28], [30], [32], [33], [34], [37], [38], [40], [41], [43], [44], [45] elaborate several alternative SFM algorithms, and [29] and [42] discuss the advantages and disadvantages of such implementations. It is important to stress that the present paper is not "yet another work" on SFM but rather an attempt to incorporate additional information to overcome the intrinsic weaknesses of today's solutions while preserving the multiview flavor.

In order to remove the ambiguity in the reconstruction, absolute information about the motion or the scene is required and several alternative methods for acquiring it can be found in the literature. For example, one could complement the system with a set of known landmarks distributed in the scene. Once a set of features from a frame is matched to the supplied landmarks, the pose of the camera can be determined to an accuracy that will usually depend on the relative size of the set of landmarks and their spatial distribution. Additional processing should also allow the computation and elimination of motion drift. Variations of landmarks-based pose estimation algorithms have been presented in [7], [12], [13], [16], [18], [20], [23], [27], [39]. In spite of these apparent merits, the landmarks approach suffers from two important drawbacks that rule it out as a solution to the problem at hand under the "minimalistic" conditions assumed in this research. First, this approach requires having an available and accessible database of known and calibrated landmarks. Second, the approach assumes that the system has enough resources to match a set of image features to the stored landmarks. Put together, these two requirements demand substantial preliminary work—selecting and storing the landmarks—and, also, sufficient processing power for the online implementation.

The landmark approach can be reformulated by replacing the landmarks with a database describing the structure of the scene. The DTM is a natural candidate to meet this requirement, given that it has already been used successfully as a positioning aid [3], [5]. In the case considered in the aforesaid references, relative altitude (clearance) information is used to compute the location of a flying platform. The procedure consists of storing a data record of

- *R. Lerner and E. Rivlin are with the Department of Computer Science, Technion—Israel Institute of Technology, Haifa 32000, Israel. E-mail: {ronenl, ehudr}@cs.technion.ac.il.*
- *H.P. Rotstein is with Rafael, Israel and the Department of Electrical Engineering, Technion—Israel Institute of Technology, Haifa 32000, Israel. E-mail: hector@ee.technion.ac.il.*

Manuscript received 11 July 2004; revised 7 Dec. 2005; accepted 28 Dec. 2005; published online 13 July 2006.
Recommended for acceptance by K. Daniilidis.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number TPAMI-0345-0704.

terrain altitudes under the platform and then matching this data to the DTM database. Good results for this class of algorithms have been reported in the literature for systems capable of independently measuring motion to a good accuracy; note that motion knowledge is required in order to transform the clearance measurements into a 3D curve that can later be registered into the 3D terrain for the pose computation. One might think that 2D visual projections should compare favorably with the 1D altitude measurement, but the scaling ambiguity involved in the reconstruction makes this an involved procedure. In a straightforward translation of clearance-based to vision-based approach, an SFM algorithm could be used to recover motion and reconstruct an unscaled patch of the scene. Thereafter, a correlation algorithm could be used for matching the patch to the DTM. Variations of this very basic idea have been used in [31] and [35]. One of the difficulties of this approach is that it requires the intermediate step of reconstructing the 3D geometry of the scene, which is prone to errors in the face of inaccurate information about the motion. As will be shown in the present work, the intermediate step of structure reconstruction can be skipped by addressing simultaneously the problem of pose **and** motion computation, which leads to more accurate estimates for the navigation parameters. The DTM will be locally approximated around each feature point by a plane and a constraint will be written given the a priori location of the feature and its corresponding projections on the two consecutive camera frames.

2 The Two-View Constraint

This section contains a derivation of the two-view constraint, relating feature correspondences with the absolute position, orientation, and motion of a calibrated camera. The section begins by introducing some notation and a formal presentation of the problem of interest. After that, a solution is built in three steps. The solution is then compared with a more classical SFM result. Some properties of the solution are also discussed.

2.1 Notation and Problem Definition

The problem definition requires two coordinate systems:

1. $C(t)$: Camera-fixed coordinate system at time t, such that:
   - The origin of $C(t)$ is at the center of projection of the camera.
   - The Z axis coincides with the optical axis.
   - The X and Y axes lie parallel to the horizontal and vertical axes of the image plane.
2. W: Global (i.e., "World") coordinate system.

For discrete time instances t_i, the somewhat simpler notation $C_i = C(t_i)$ will often be used below. Moreover, the global coordinate system is chosen to facilitate a "flat earth" assumption.[1] The location and orientation of $C(t)$ with respect to W will be denoted $p(t)$ and $R(t)$, respectively, with $p(t) \in \mathbb{R}^3$ and $R(t)$ a rotation matrix. Throughout this paper, and whenever required for clarity, a left superscript will be added to vectors to denote the coordinate frame in which they are expressed (for example: ${}^F v$ denotes the vector v in the F frame). Again for simplicity, the superscript will be dropped for vectors in the World coordinate system—see the definition of $p(t)$ above. Also, $R(t)$ will always denote the rotation from camera to world coordinates, so that, for instance,

$$v = {}^W v = R(t)\,{}^C v + p(t).$$

Consider now two consecutive time instances t_1 and t_2: The corresponding two frames of the camera will be referred to as C_1 and C_2 and, likewise, $p_i = p(t_i)$, $R_i = R(t_i)$ for $i = 1, 2$. The ego-motion transformation connecting the two frames is given by the translation vector p_{12} (which is the position of the *origin* of the camera at t_1 under the C_2 frame) and the rotation matrix $R_{12} = R_2^T R_1$, such that

$${}^{C_2} v = R_{12}\,{}^{C_1} v + p_{12}.$$

The next ingredient in the formulation is the feature correspondence pairs. Given $Q_i \in \mathbb{R}^3$, a feature point, $\{u_{i_k}\}$ $(i = 1 \ldots n, k = 1, 2)$, denotes the perspective projections of the point on the image planes. More specifically, $u_{i_1} \in \mathbb{R}^2$ and $u_{i_2} \in \mathbb{R}^2$ represent the location in the image during the first and second frames, respectively. It is implicit in this notation that the *correspondence* problem has been solved between the projections in the two images. Assuming the focal distance has been conveniently normalized to unity, let ${}^{C_1} q_{i_1}$ and ${}^{C_2} q_{i_2}$ be ${}^{C_k} q_{i_k} = ({u_{i_k}}^T, 1)^T \in \mathbb{R}^3$. Although the q_{i_k} vectors are expressed in the corresponding camera frame, their left superscripts will be omitted throughout this paper for the sake of notation simplicity.

Unlike the classical SFM problem, in which feature correspondence (or optical flow) is the only available data, the present work makes use of additional information provided by a DTM. In practice, a DTM is an ASCII or binary file that contains only spatial elevation data in a regular grid pattern in raster format for the whole or part of the earth. DTMs are usually classified in levels according to their grid size. For instance, Level 0 denotes a 30 arcsec (or approximately 1 km) grid map, while Level 5 denotes a fine, almost 1 m, grid map. Not all of these maps are available to the general public, but 30 m are or will be available for most of the earth, while 10 m ones can be obtained for some regions (in particular, for the United States). For instance, the Shuttle Radar Topography Mission generated an elevation map with a 30 m grid and 16 m absolute vertical accuracy.

For the discussion below, it is convenient to assume that the DTM is a function $h : \mathbb{R}^2 \to \mathbb{R}$ giving the altitude of the terrain, say, over the mean local sea level, for each geographical location $(U, V) \in \mathbb{R}^2$. Since (U, V) are taken in W, this assumption involves extensive, database-dependent, albeit straightforward, manipulations with coordinate systems over the earth. When reporting numerical results in later chapters, the discrete and noisy nature of the data will be taken into account.

The constraint to be derived next assumes that the DTM can be linearized around a point. This in turn requires one to assume that a sufficiently good estimate of the pose of the camera at t_1 and its ego-motion between the two time instances are available. These estimates will be denoted by the subscript "E," i.e., p_{1_E}, R_{1_E}, p_{12_E} and R_{12_E}, to stress that these are a priori estimated quantities. Estimates can be obtained, for instance, from a dead-reckoning algorithm that uses inertial-system measurements.

With the notations introduced above, the *Pose and Motion from Correspondence and DTM* problem can be formulated as follows:

1. For instance, W may be the *local-level* frame with the same origin as $C(t_0)$ for some reference time t_0.

Given the following data,

1. a priori estimates for the camera pose and ego-motion p_{1_E}, R_{1_E}, p_{12_E}, and R_{12_E},
2. correspondence pairs $\{u_{i_k}\}$, and
3. a DTM function $h : \mathbb{R}^2 \rightarrow \mathbb{R}$,

find the true pose and motion p_1, R_1, p_{12}, and R_{12} of the camera.

In practice, the presence of noise will not allow the computation of true pose and motion and, hence, one should settle for a posteriori estimates of these quantities.

2.2 Single-Frame Geometry

To begin the discussion, a single feature point on the terrain, Q_T, will be considered in this and the next sections. Assuming a pinhole model for the calibrated camera, this feature is perspectively projected onto a point q_1 on the image-plane of the first camera frame $C1$. The present section concentrates on the single-view geometry that will eventually lead to the two-view geometry discussed in the next section.

Using an initial guess of the camera pose at t_1, the line passing through p_{1_E} along the direction of q_1 can be intersected with the DTM. A ray-tracing algorithm can be used for this purpose. The intersection point can be computed as

$$Q_E = p_{1_E} + \lambda_E R_{1_E} q_1, \tag{1}$$

for some λ_E computed, e.g., by using ray-tracing. The subscript letter "E" again highlights the fact that this point is an estimated location. The true feature location Q_T can similarly be expressed by

$$Q_T = p_1 + \lambda_T R_1 q_1, \tag{2}$$

and, in general, $Q_E \neq Q_T$. There are two main error sources that explain the difference between Q_T and Q_E: the error in the a priori estimates for the pose and the errors in the determination of Q_E caused by DTM discretization, as well as intrinsic errors. However, it is assumed that, for reasonable a priori estimates and DTM-related errors, the two points are sufficiently close so that Q_T can be approximated as belonging to a plane tangent to the DTM at the point Q_E; see Fig. 1. Specifically, if N denotes the normal to the DTM at Q_E, then

$$N^T(Q_T - Q_E) = 0. \tag{3}$$

The parameter λ_T is the *depth* of the feature point and encodes the information about the structure of the scene. In order to avoid the structure reconstruction, the linearization assumption can be used to eliminate λ_T from the expressions above. Indeed, from (2),

$$Q_T - Q_E = p_1 + \lambda_T R_1 q_1 - Q_E, \tag{4}$$

and, hence, using (3) and after some reordering,

$$N^T(p_1 - Q_E) + \lambda_T N^T R_1 q_1 = 0,$$

implying

$$\lambda_T = -\frac{N^T(p_1 - Q_E)}{N^T R_1 q_1}. \tag{5}$$

The above expresses the depth of the scene point as a function of the camera pose and the (known) linearization plane parameters. It is this expression which enables us to avoid handling the depths of the scene points as unknowns. Instead, this expression can replace unknowns in our equations, which will eventually result in a system of 12 unknowns (of the pose and motion parameters) instead of $n + 12$ unknowns for n tracked features.

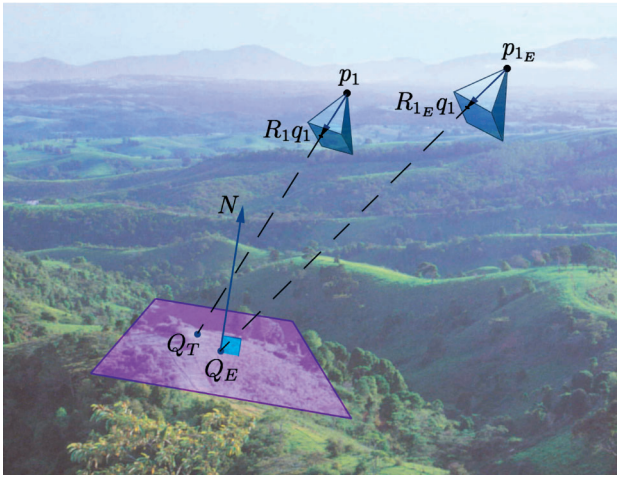


Fig. 1. The terrain feature, Q_T, is perspectively projected to the image plane point q_1 under the true first camera frame (where p_1 represents its position and R_1 its orientation). Using this projected point and the estimated pose of the camera (p_{1_E} and R_{1_E}), the ray from p_{1_E} in the direction of $R_{1_E}q_1$ can be intersected with the DTM at Q_E. The DTM is linearized around this point and Q_T is assumed to lie on that tangent plane.

Substituting (5) in (4) and grouping the different terms, one gets

$$Q_T - Q_E = \left(I - \frac{R_1 q_1 N^T}{N^T R_1 q_1}\right)(p_1 - Q_E). \tag{6}$$

In order to further simplify the expression and facilitate its geometrical interpretation, the following projection operator is introduced:

$$\mathcal{P}(u, n) \doteq \left(\mathrm{I} - \frac{u n^T}{n^T u}\right). \tag{7}$$

This operator projects vectors onto the plane orthogonal to n. Notice that the projection is not orthogonal but, rather, along the direction of u. By using the above definition, it is straightforward to verify that $n^T \cdot \mathcal{P}(u, n) \equiv 0$ and $\mathcal{P}(u, n)u \equiv 0$. See Fig. 2a for a geometrical interpretation of $\mathcal{P}$.

Using the above operator, one can rewrite (6) as

$$Q_T - Q_E = \mathcal{P}(R_1 q_1, N)(p_1 - Q_E), \tag{8}$$

with the operator

$$\mathcal{P}(R_1 q_1, N) = \left(\mathrm{I} - \frac{R_1 q_1 N^T}{N^T R_1 q_1}\right) \tag{9}$$

projecting vectors onto the tangent plane to the DTM at Q_E along the direction of $R_1 q_1$.

Equation (8) has a nice geometric interpretation as shown in Fig. 2b. The unknown vector $Q_T - Q_E$ is the vector from Q_E to Q_T in the frame W. It can be obtained by taking the vector from Q_E to p_1 and using the $\mathcal{P}$ operator in order to project it onto the linearization plane orthogonal to N along

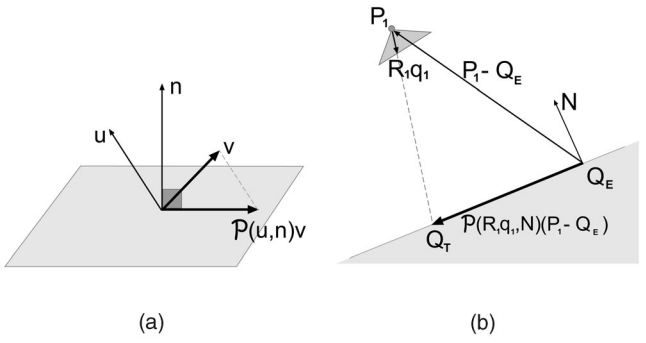


Fig. 2. (a) The vector v is being projected by $\mathcal{P}(u,n)$ onto the plane orthogonal to n along the direction of u. (b) In order to obtain the unknown vector $Q_T - Q_E$, the vector $p_1 - Q_E$ is being projected onto the linearization plane and along the $R_1 q_1$ direction using the $\mathcal{P}$ projection operator.

the q_1 direction. ($R_1 q_1$ will be used since the world's frame representation of q_1 is required.)

2.3 Two-Frame Geometry

Suppose next that a second frame is now available. Then, the location of the feature point in the C_2 frame can be expressed as

$$^{C_2}Q_T = p_{12} + R_{12}\,^{C_1}Q_T. \tag{10}$$

Since

$$Q_T = p_1 + R_1\,^{C_1}Q_T,$$

(10) can also be expressed as

$$\begin{aligned} ^{C_2}Q_T &= p_{12} + R_{12}\left[R_1^T(Q_T - p_1)\right] \\ &= p_{12} + R_2^T(Q_T - p_1). \end{aligned} \tag{11}$$

Using a standard reprojection argument, one can claim that $^{C_2}Q_T$ can also be written using its projection onto the image plane:

$$^{C_2}Q_T = \lambda_{T2} q_2, \tag{12}$$

where λ_{T2} is the depth of the feature in C_2. To eliminate the dependence on the depth, use the equality

$$\left(I - \frac{q_2 q_2^T}{q_2^T q_2}\right) q_2 = \mathcal{P}(q_2, q_2) q_2 = 0. \tag{13}$$

For ease of notation, call

$$\mathcal{P}(q_2) \doteq \mathcal{P}(q_2, q_2).$$

Using this in (11), one gets the constraint

$$\mathcal{P}(q_2)\left[p_{12} + R_2^T(Q_T - p_1)\right] = 0. \tag{14}$$

The last step in getting a useful constraint and avoiding structure reconstruction is to substitute

$$Q_T - p_1 = (Q_T - Q_E) + (Q_E - p_1)$$

in the equation above and use the one-view geometry constraint (8) to get

$$\mathcal{P}(q_2)\left[p_{12} + R_2^T(I - \mathcal{P}(R_1 q_1, N))(Q_E - p_1)\right] = 0. \tag{15}$$

Using the definition of the projection operator (7),

$$\mathcal{P}(q_2)\left[p_{12} + R_2^T \frac{R_1 q_1 N^T}{N^T R_1 q_1}(Q_E - p_1)\right] = 0, \tag{16}$$

or

$$\mathcal{P}(q_2)\left[p_{12} + \frac{R_{12} q_1 N^T}{N^T R_1 q_1}(Q_E - p_1)\right] = 0. \tag{17}$$

This basic constraint involves all pose and ego-motion parameters defining the two frames of the camera and involves the measurements in the image plane and the estimated location for the feature point Q_E. The pose and ego-motion parameters are, therefore, constrained to verify this equation.

Remark. Notice that (15) and its variants are trivially verified by multiplying by q_2^T on the left, so that this equation is equivalent to two—and not three—linearly independent equations.

2.4 Multiple Features

Suppose next that n feature points are tracked in two frames, so that the estimated locations Q_{Ei} and projections onto the image plane q_{1i} and q_{2i} are estimated and measured, respectively, for $i = 1, \cdots, n$. Associated with each Q_{Ei} is the normal vector to the DTM at this point, namely, N_i. Taking this into account, one can rewrite (17) in matrix form as

$$\begin{bmatrix} -\mathcal{P}(q_{2i}) & \mathcal{P}(q_{2i})\frac{R_{12} q_{1i} N_i^T}{N_i^T R_1 q_{1i}} \end{bmatrix} \begin{bmatrix} p_{12} \\ p_1 \end{bmatrix} = \mathcal{P}(q_{2i}) \frac{R_{12} q_{1i} N_i^T}{N_i^T R_1 q_{1i}} Q_{Ei}. \tag{18}$$

Repeating this for each feature point:

$$\begin{bmatrix} -\mathcal{P}(q_{21}) & \mathcal{P}(q_{21})\frac{R_{12} q_{11} N_1^T}{N_1^T R_1 q_{11}} \\ -\mathcal{P}(q_{22}) & \mathcal{P}(q_{22})\frac{R_{12} q_{12} N_2^T}{N_2^T R_1 q_{12}} \\ \vdots & \vdots \\ -\mathcal{P}(q_{2n}) & \mathcal{P}(q_{2n})\frac{R_{12} q_{1n} N_n^T}{N_n^T R_1 q_{1n}} \end{bmatrix} \begin{bmatrix} p_{12} \\ p_1 \end{bmatrix} = \begin{bmatrix} \mathcal{P}(q_{21})\frac{R_{12} q_{11} N_1^T}{N_1^T R_1 q_{11}} Q_{E1} \\ \mathcal{P}(q_{22})\frac{R_{12} q_{12} N_2^T}{N_2^T R_1 q_{12}} Q_{E2} \\ \vdots \\ \mathcal{P}(q_{2n})\frac{R_{12} q_{1n} N_n^T}{N_n^T R_1 q_{1n}} Q_{En} \end{bmatrix}. \tag{19}$$

In compact notation:

$$\mathcal{A}_n \begin{bmatrix} p_{12} \\ p_1 \end{bmatrix} = \mathcal{B}_n. \tag{20}$$

Note that $\mathcal{A}_n$ and $\mathcal{B}_n$ depend on known quantities: the estimated features, the normals of the DTM tangent planes, and the images of the features at the two time instances, together with the unknown orientation R_1 and the relative rotation R_{12}. At this point in our discussion, several remarks are in order.

Remark 1. The constraint (19) involves 12 "unknowns," namely, the pose and ego-motion of the camera. From the remark at the end of the previous section, the equation involves at most $2n$ linearly independent constraints, so that at least six features at different locations Q_{Ti} are required to have a determinate system of equations. Usually, more vectors will be used in order to define an overdetermined system and, hence, reduce the effect of noise. Clearly, there are degenerate scenarios in which the obtained system is singular, no matter what the number of available features is. Examples for such scenarios include flying above completely planar or spherical terrain (see Section 2.5). However, in the general case where the terrain has "interesting" structure the system is nonsingular and the 12 parameters can be obtained.

Remark 2. The constraint (19) is nonlinear and, therefore, no analytic solution to it is readily available. Thus, an iterative scheme will be used in order to solve this system. A robust algorithm using Newton iterations and an M-estimator will be described in following sections.

Remark 3. Given Remark 2, one observes that the location and translation appear linearly in the constraint. Using the pseudoinverse, these two vectors can be solved explicitly to give

$$\begin{bmatrix} p_{12} \\ p_1 \end{bmatrix} = \mathcal{A}_n^{\dagger}\mathcal{B}_n, \tag{21}$$

so that, after resubstituting in (20),

$$\left(I - \mathcal{A}_n\mathcal{A}_n^{\dagger}\right)\mathcal{B}_n = 0. \tag{22}$$

This remark leads to two conclusions:

1. If the rotation is known to good accuracy and measurement noise is relatively low, then the position and translation can be determined by solving a linear equation. This fact may be relevant when "fusing" the procedure described here with other measurement, e.g., with inertial navigation.
2. Equation (22) shows that the estimation of rotation (both absolute and relative) can be separated from that of location/translation. This fact is also found when estimating pose from a set of visible landmarks as shown in [25]. In that work, similarly to the present, the estimate is obtained by minimizing an objective function that measures the errors in the *object-space* rather than on the image plane (as in most other works). This property enables the decoupling of the estimation problem. Note, however, that [25] addresses only the pose rotation and translation decoupling while, here, the six parameters of absolute and relative rotations are separated from the six parameters of the camera location and translation.

2.5 Degenerate Scenarios

The proposed algorithm utilizes the information derived from the 2D movement of the tracked features on the image plane. It relies on the assumption that these movements dictate the ego-motion of the camera and the structure of the 3D features up to similarity. Next, the additional information supplied by the DTM is assumed to dictate the unknown similarity transformation by restricting the 3D features to lay on the terrain. However, in any case in which one of these assumptions does not hold, a degenerate scenario arises and, thus, a singular system of constraints will be obtained.

Pure rotational ego-motion is a classic scenario where the first assumption does not hold. It is well established that, under such motion, the depth of the 3D feature has no influence on the projected features' displacement. Collinear features are another example where the ego-motion cannot be determined.

Intuitive examples for scenarios where the second assumption does not hold include a planar or spherical terrain. Once the 3D structure of the features constellation was derived (from the image displacements), a whole manifold of solutions embedded in the similarities' configuration space is adequate. In order to study the conditions under which the terrain surface yields a degenerate scenario, we follow the *Constraint Analysis* proposed by [36] and extend it from Euclidean transformations to similarities.

Assuming one is supplied with the true similarity (which registers the 3D features into the terrain) as an initial guess for the algorithm, a degenerate scenario could be differentially characterized by the existence of infinitesimal perturbation of the similarity parameters such that the quality of the registration will not deteriorate. Let $Q = {}^{C_2}Q_T = {}^{C_2}Q_E$ be a 3D feature lying on its corresponding tangent plane. By applying an infinitesimal translation of $\delta t \in \mathbb{R}^3$, scale of $1 + \delta s \in \mathbb{R}$, and rotation of $\|\delta\omega\|$ around the $\delta\omega \in \mathbb{R}^3$ axis, these features are transformed to

$$Q' = (1 + \delta s)(Q + \delta\omega \times Q) + \delta t. \tag{23}$$

This equation is obtained from the first order approximation of the Rodrigues formula. Therefore, a degenerate scenario arises when there are non-all-zero $\delta t, \delta\omega, \delta s$ such that

$${}^{C_2}N_i^T\left[Q_i' - Q_i\right] = {}^{C_2}N_i^T[\delta s \cdot Q_i + (1 + \delta s)(\delta\omega \times Q_i) + \delta t] = 0 \tag{24}$$

for all tracked features ($i = 1...n$). The above constraint verifies that the vector of the 3D feature displacement induced by the similarity perturbation is parallel to the corresponding tangent plane and, thus, has no effect on the registration quality. One should notice that, in case such perturbation is found, the scaled perturbation $\lambda \cdot \delta t, \lambda \cdot \delta\omega, \lambda \cdot \delta s$ (for any $\lambda \in \mathbb{R}$) should also verify the constraint in order to create a whole subspace of adequate solutions in the similarities configuration space:

$${}^{C_2}N_i^T[\Delta Q_i(\lambda)] = 0, \tag{25}$$

where

$$\Delta Q_i(\lambda) = \lambda\delta s \cdot Q_i + \lambda\delta\omega \times Q_i + \lambda^2\delta s(\delta\omega \times Q_i) + \lambda\delta t. \tag{26}$$

Dividing (25) by λ, subtracting the result from (24), and once again dividing by $1 - \lambda$ yields

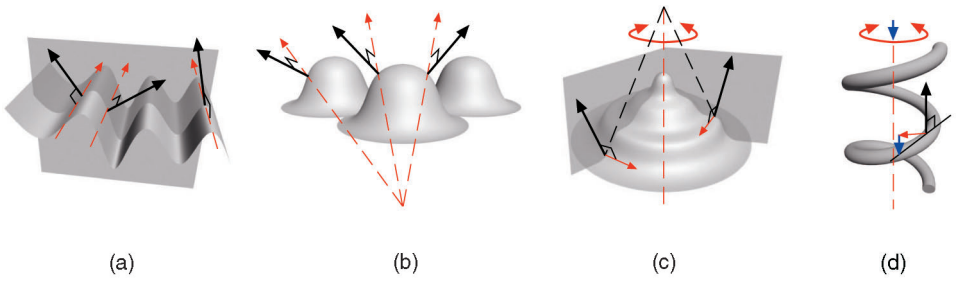


Fig. 3. Examples of constellations which lead to singularities of the algorithm: (a) features from a surface which can be swept out by moving a planar curve along constant direction, (b) features laying on a silhouette of arbitrary surface, (c) surface of revolution, (d) and spiral.

$$\delta s(\delta\omega \times Q_i) = 0. \tag{27}$$

Since the 3D features are not collinear, not all of them are parallel to $\delta\omega$. Therefore, either $\delta s = 0$ or $\delta\omega = 0$.

In case $\delta\omega = 0$, (24) reduces to

$$^{C_2}N_i^T[\delta s \cdot Q_i + \delta t] = 0. \tag{28}$$

If $\delta s = 0$, then we remain with $^{C_2}N_i^T\delta t = 0$. This means that δt is orthogonal to all the normal vectors N_i, which, therefore, must be coplanar. Surfaces with this characteristic are those which can be swept out by moving a planar curve along the δt direction (see Fig. 3a). If $\delta s \neq 0$, then it can be assumed that $\delta s = 1$ (since the scale of the perturbation is arbitrary). This leads to the constraint $^{C_2}N_i^T(Q_i + \delta t) = 0$, which means that, after moving the camera by δt, all 3D features belong to the surface's silhouette (see Fig. 3b).

In the presence of rotational perturbation ($\delta\omega \neq 0$), there is no scale change. Hence, (24) reduces to

$$^{C_2}N_i^T[\delta\omega \times Q_i + \delta t] = 0. \tag{29}$$

In the special case where $\delta\omega \perp \delta t$, δt can be expressed as a cross-product of $\delta\omega$ and some $d \in \mathbb{R}$, which leads to the following representation of (29): $^{C_2}N_i^T[\delta\omega \times (Q_i + d)] = 0$. This equation shows that, after translating the 3D features by d, the rotation-axis $\delta\omega$, the translated feature $(Q_i + d)$, and the surface normal are coplanar. Such behavior is obtained from surfaces of revolution such as a sphere or a Gaussian hill (see Fig. 3c). In the general case of arbitrary translation, δt can be decomposed into two components: $\delta t^{\perp}$ orthogonal to $\delta\omega$ and $\delta t^{\parallel}$ parallel to it. Therefore, one obtains

$$^{C_2}N_i^T\left[\delta\omega \times (Q_i + d) + \delta t^{\parallel}\right] = 0.$$

Surfaces which are consistent with that constraint include cylinder, spiral, and others (see Fig. 3d).

Remark 4. Since only infinitesimal perturbations are considered, only a small surface environment of each feature is significant for singularity conditions. Therefore, it is enough that the surface will satisfy the above conditions, piecewise.

2.6 The Epipolar Constraint Connection

Before proceeding any further, it is interesting to look at (17) in the light of previous work in SFM and, in particular, epipolar geometry. In order to do this, it is worth deriving the basic constraint in the present framework and notation. Write

$$^{C_2}Q_T = \lambda_2 q_2 = p_{12} + \lambda_1 R_{12} q_1 \tag{30}$$

for some scalars λ_1 and λ_2 (see Fig. 4). It follows that

$$p_{12} \times \lambda_2 q_2 = p_{12} \times \lambda_1 R_{12} q_1, \tag{31}$$

and, hence,

$$q_2^T(p_{12} \times R_{12} q_1) = 0. \tag{32}$$

For a vector $x \in \mathbb{R}^3$, let $x^{\wedge}$ denote the skew-symmetric matrix:

$$x^{\wedge} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}^{\wedge} = \begin{bmatrix} 0 & -x_3 & x_2 \\ x_3 & 0 & -x_1 \\ -x_2 & x_1 & 0 \end{bmatrix}.$$

Then, it is well known that the vector product between two vectors x and y can be expressed as

$$x \times y = x^{\wedge} y.$$

Using this notation, the epipolar constraint (32) can be written as

$$q_2^T(R_{12}q_1)^{\wedge}p_{12} = 0 \tag{33}$$

and symmetrically as

$$q_1^T R_{12}^T q_2^{\wedge} p_{12} = 0. \tag{34}$$

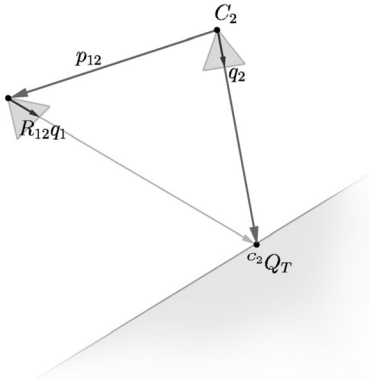


Fig. 4. The examined scenario from the second camera frame's (C_2) point of view. q_2 is the perspective projection of the terrain feature $^{C_2}Q_T$ and, thus, the two should coincide. Additionally, since q_1 is also a projection of the same feature in the C_1 frame, the epipolar constraint requires that the two rays (one in the direction of q_2 and the other from p_{12} in the direction of $R_{12}q_1$) will intersect.

The important observation here is that, if the vector p_{12} verifies the above constraint, then the vector $\kappa \cdot p_{12}$ also verifies the constraint, for any number κ. This is an expression of the ambiguity built into the SFM problem. On the other hand, the constraint (17) is nonhomogeneous and, hence, does not suffer from the same ambiguity. In terms of the translation alone (and for only one feature point!), if p_{12} verifies (17) for given an R_1 and R_{12}, then $p_{12} + \kappa q_2$ will also verify the constraint and, hence, the ego-motion translation is defined up to a one-dimensional vector. However, one has the following trivially:

$$q_1^T R_{12}^T q_2{}^\wedge q_2 = 0, \tag{35}$$

and, hence, the epipolar constraint does not provide an additional equation that would allow us to solve for the translation in a unique manner. Moreover, observe that (17) can be written using a vector product instead of the projection operator as

$$q_2{}^\wedge \left[p_{12} + \frac{R_{12} q_1 N^T}{N^T R_1 q_1} (Q_E - p_1) \right] = 0. \tag{36}$$

Taking into account the identity

$$(R_{12} q_1)^T q_2{}^\wedge R_{12} q_1 \equiv 0, \tag{37}$$

it is possible to conclude that (36) $\longrightarrow$ (34) and, hence, the new constraint "contains" the classical epipolar geometry. Indeed, one could think of the constraint derived in (17) as strengthening the epipolar constraint by requiring not only that the two rays (in the directions of q_1 and q_2) should intersect, but, in addition, that this intersection point should lie on the DTM's linearization plane. Observe, moreover, that taking more than one feature point would allow us to completely compute the translation (at least for the given rotation matrices).

3 Algorithm Implementation

In this section, we elaborate a possible implementation for the proposed algorithm. As mentioned above, the constraint is nonlinear and, hence, needs to be solved using a numerical procedure. In particular, a least-squares solution using the Newton-iterations scheme can be used.

3.1 Internal versus External Iterations

In a practical implementation, the approximation of the DTM by a plane is true only locally and, hence, the Newton iterations presented above can only partially correct the errors in the initial a priori estimate. Notice that, once the ray-tracing algorithm has located the estimated terrain features—the Q_Es, these points, together with the tangent planes they determine, are kept fixed during the iterations. Since it is assumed that the Q_T points lie somewhere on these planes, the Newton iterations will not converge to the true pose and motion but, rather, to the best pose and motion for which the 3D features are on the required planes.

The limitation described above can be easily ameliorated by reactivating the ray-tracing algorithm between consecutive iterations. Namely, after each Newton iteration, the updated pose of the first camera frame could be used for the ray-tracing, leading to more accurate estimates of the Q_Es and a refinement of the tangent-plane approximation. In the theoretical scenario of perfect DTM and image features location (infinite resolution and error-free), and when the initial guess of the camera pose is not too far from the true pose, such a scheme would converge to the true camera pose and ego-motion parameters, as will be empirically shown in Section 4. The resulting algorithm exacts, nevertheless, a high price in terms of computational cost. Indeed, in spite of having been a topic of continuous research in the computer-graphics community, ray-tracing algorithms are still considered to be involved and time consuming. Consequently, and taking into account real-time considerations, it is desirable to reduce the number of ray-tracing steps as much as possible. This observation leads to an alternative scheme based on *internal* and *external iterations*. The internal iterations are the Newton iterations discussed above. During these iterations, the Q_Es and tangent planes are kept constant and the algorithm proceeds until a convergence criterion has been met. When this occurs, an external iteration is performed using the best available pose and motion data. During this iteration, ray-tracing is used to compute a new set of estimated locations and of tangent planes. The overall algorithm continues until the estimated locations converge.

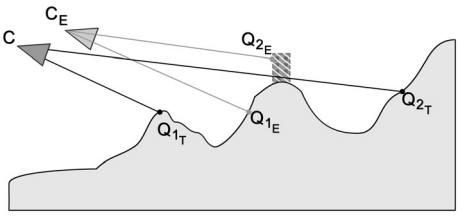


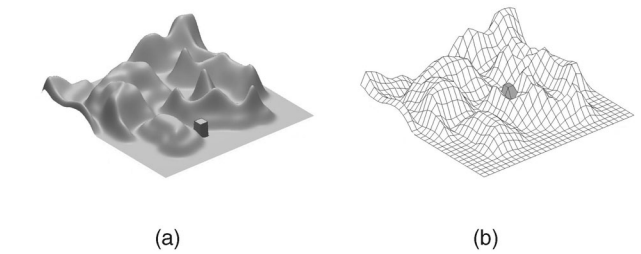
Fig. 5. Outliers caused by terrain shape and DTM mismatch. C_T and C_E are true and estimated camera frames, respectively. Q_{1_E} and Q_{2_E} are outliers caused by terrain shape and by terrain/DTM mismatch, respectively.

3.2 Dealing with Outliers

In order to handle real data, a procedure for dealing with outliers must be included in the implementation. Three kinds of outliers should be considered:

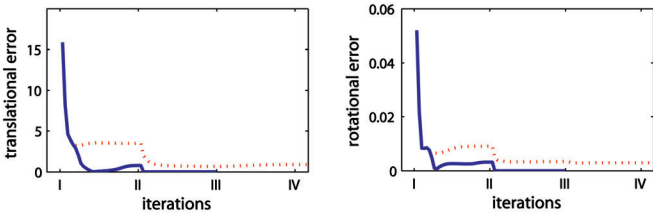
1. outliers present in the correspondence solution (i.e., "wrong matches"),
2. outliers caused by the terrain shape, and
3. outliers caused by relatively large errors between the DTM and the observed terrain.

The latter two kinds of outliers are illustrated in Fig. 5. The outliers caused by the terrain shape appear for terrain features located close to large depth variations. For example, consider two hills, one closer to the camera, the other farther away, and a terrain feature Q located on the closer hill. The ray-tracing algorithm using the erroneous pose may "miss" the proximal hill and erroneously place the feature on the distal one. Needless to say, the error between the true and estimated locations is not covered by the linearization. To visualize the errors introduced by a relatively large DTM-actual terrain mismatch, suppose a building was present on the terrain when the DTM was acquired, but is no longer there when the experiment takes place. The ray-tracing algorithm will locate the feature on the building, although the true terrain-feature belongs to a background that is now visible.



(a) (b)

Fig. 6. (a) The virtual terrain and (b) the DTM constructed from this terrain (grid-spacing is coarser than in the experiment, for visualization purposes). Note that the building on the virtual terrain (the box at the bottom of (a)) has been moved to the gray bump at the center of the DTM.



(a) (b)

Fig. 7. (a) Translational and (b) rotational errors of the calculated pose as a function of the number of iterations. The symbols I, II, III, and IV denote external iterations. Each iteration contains 30 internal iterations. The blue solid line is an error-free scenario while the red dotted line is a scenario with Gaussian error of 0.5 pixel S.D. Units are meters and radians.

As discussed above, the multifeature constraint is solved in a least-squares sense for the pose and motion variables. Given the sensitivity of least-squares to incorrect data, the inclusion of one or more outliers may result in the convergence to a wrong solution. A possible way to circumvent this difficulty is by using an M-estimator, in which the original solution is replaced by a weighted version. In this version, a small weight is automatically assigned to the constraints involving outliers, thereby minimizing their effect on the solution. See [17] for further details about M-estimation techniques.

4 Experimental Results

Experiments were performed to verify the applicability, accuracy, and robustness of the algorithm. Two types of experiments were conducted: the first using synthetic data and the second using an experimental setup where data was obtained by a real camera focusing on a terrain model.

4.1 Simulation Results

In this experiment, a virtual-terrain of 300×300 meters was synthesized. The terrain contains patches of varying slopes representing hills of various heights, the tallest one being 60 meters high. The terrain also contains a $15 \times 15 \times 25$ meter "box" representing a man-made building. The terrain was then discretized to produce a model, i.e., the DTM, with a one-meter spatial grid. After computing the DTM, the synthetic terrain was modified by changing the location of the building so as to introduce a substantial terrain-DTM mismatch (see Fig. 6). Images from the terrain were obtained by using a virtual camera from various positions and orientations with respect to the terrain. A collection of 100 different correspondence pairs was analytically derived. The a priori estimate of the position and orientation of the camera was obtained by adding an error of approximately 17 m and 3 degrees.

Fig. 7 shows a typical example of the convergence of the algorithm. One can see that convergence is achieved after four external iterations. When the synthesized correspondence measurements were error-free, the estimation process was able to completely remove the error from the pose initial guess. The outliers caused by the mislocated building did not deteriorate the estimate accuracy due to the utilization of the M-estimator. When i.i.d. Gaussian noise of $\sigma = 0.001$ (roughly equivalent to 0.5 pixels for a 500×500 camera) was added to the correspondence measurements, a less accurate estimate was obtained for the camera pose. However, as shown in Fig. 7, convergence speed was not significantly affected. During the tests, 30 internal iterations were performed for each external one, although it is clear that fewer iterations would have produced essentially the same result. In the face of measurements error, the obtained accuracy depends on different parameters of the confronted scenario: the number of corresponding features, the image and DTM resolutions, the structure of the visible terrain, and the length of the ego-motion baseline. Lerner et al. [22] present an extensive simulation and detailed discussion regarding the effects of these parameters on the algorithm performance.

4.2 Robustness of the Algorithm Against Large Errors in the Initial Guess

The optimization scheme used by the algorithm can only search for a local minimum. Since the problem is non-convex in the general case, there may be scenarios in which the camera will converge to the wrong pose. These scenarios are characterized by a large bias between the visible patch from the scene and the apparently visible patch taken from the DTM. The severity of this bias cannot be determined absolutely, but rather with respect to the "frequency" of the observed terrain: Rough mountainous terrain (that contains high frequencies) will be more sensitive to small biases compared to a terrain with soft and smooth hills (that only contains low frequencies).

Three factors should be considered for the characterization of such problematic scenarios: the magnitude of the initial guess error, the distance between the camera and the observed features, and the roughness of the terrain. The second factor is very important when considering angular errors in the camera pose. In such cases, the bias magnitude of the scene features will be larger for distant features.

In order to check the robustness of the algorithm, a series of simulations was conducted. A DTM of real terrain with 10 m grid spacing was used [1]. In light of the above observations, a rough terrain from Montana's Rocky Mountains was chosen to examine the algorithm in relatively difficult scenarios (see Fig. 8a).

Five virtual cameras were placed in different locations and orientations and about 225 feature correspondences were analytically derived for short baseline ego-motion of

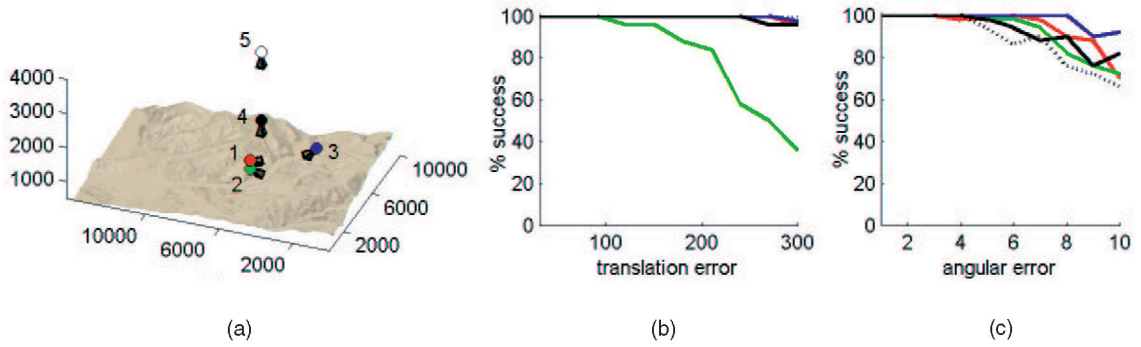


Fig. 8. The robustness simulations were conducted for five virtual cameras located in different poses. (a) A DTM of Montana's Rocky mountains was chosen to examine the algorithm on relatively rough terrain. (b, c) The percentage of success in converging to the true camera pose for different magnitudes of (b) translational and (c) angular errors in the algorithm's initial guess. The red lines correspond to camera 1, green to camera 2, blue to camera 3, black to camera 4, and the dotted line to camera 5.

20 m along the camera Z direction. In each test of each virtual camera, a variety of positional errors (30 m to 300 m) and angular errors (1 degree to 10 degrees) in the pose initial guess were randomly generated. The percentages of success under the tested error magnitudes are shown in Figs. 8b and 8c.

As can be seen, convergence to the true pose was obtained for any initial guess error smaller than 100 m of the camera position and 4 degrees of its orientation. As expected, camera 5 is the most sensitive to angular error due to its large distance to the terrain. Camera 2, on the other hand, is very sensitive to translational error. This may result from its low altitude, which leads to a relatively small observable patch, which is not very informative.

Performance was also tested using larger baselines: 20 m and 100 m along the camera Z direction and along the camera X direction. However, similar results were obtained for all types and magnitudes of baselines. This is in line with the former argument regarding the factors that should influence the algorithm robustness. The baseline does not influence the above-mentioned bias and, thus, should not influence the algorithm robustness.

Errors with magnitude of 100 m and 4 degrees, as mentioned above, are considered huge for airborne vehicles. As was mentioned before, the accuracy of the pose computed by the proposed algorithm depends on different parameters: the number of corresponding features, the image and the DTM resolution, the structure of the terrain and the ego-motion baseline (see [22]). In most realistic scenarios, the average error is expected to be approximately 10 m and 0.6 degree (compare, for example, to the errors in Fig. 7 and the results in Section 5). Therefore, in case it is desired to keep the errors under 15 m and 1 degree, for example, the vision-based algorithm should be activated in time intervals that prohibit the inertial navigation system to drift the pose by more than 5 m and 0.4 degrees. In what follows, the minimal rate for keeping the navigation error within the above-mentioned margins is computed. An example for available consumer off-the-shelf navigation systems is the MIDG-II series IMU/INS system of Microbotics (see http://www.microbotics.com). Using this system for pure inertial navigation, the orientation solution diverges in $0.05^\circ/\mathrm{s}/\sqrt{\mathrm{Hz}}$. This leads to accumulated angular error of $0.05 \cdot \sqrt{\Delta t}$ after Δt seconds. As a result, an interval of no more than $(0.4/0.05)^2 = 64$ seconds should be kept for the desired orientation accuracy. As for the positional error, the inertial drift can be expressed by $\Delta p = \Delta a \cdot \Delta t^2/2$, where Δa is the acceleration error, which is approximately $0.003\ \mathrm{m/s^2}$ in the MIDG-II system. Thus, an interval of $\sqrt{2 \cdot 5/0.003} \simeq 57$ seconds is required. To conclude, by activating the proposed algorithm at a minimal rate of $1/57\,\mathrm{Hz}$, the expected navigation errors will be kept far lower than its robustness breaking point. Therefore, the algorithm can be practically used for realistic navigation systems even when confronting rough terrains such as the Rocky Mountains and even when flying far away from the observed features.

4.3 Lab Experiment Results

Lab experimentation was performed using a real 3D model of a terrain and real images obtained by a camera. The dimensions of the model were 50×77 cm with elevation variations as high as 24 cm (see Fig. 9a). A laser-based 3D scanner was used to capture the terrain and build a DTM with a 1 cm spatial grid (see Fig. 9b).

In each experiment, the camera moved along a trajectory while attached to a robot manipulator. This configuration allowed moving of the camera in a controlled manner while also providing *true* measurements for the pose of the camera at all time instances. Fig. 10 shows examples for two of the trajectories evaluated. The first trajectory (*a* in the figure) contains mostly translational camera motion with

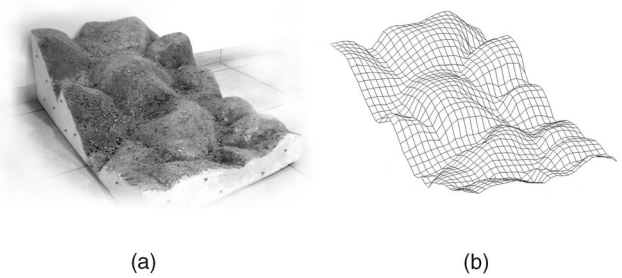


Fig. 9. (a) A 3D terrain model of horizontal dimension 50×77 cm. (b) The DTM was constructed by using a laser-based 3D-scanner. The spatial grid was 1 cm (the one in the figure has a coarser grid for visualization purposes).

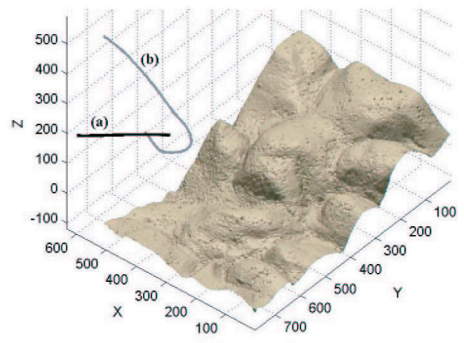


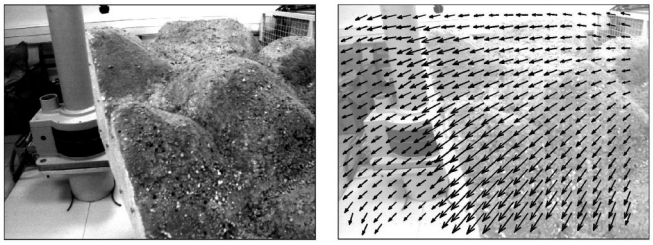
Fig. 10. Two of the tested trajectories. Trajectory *a* is mostly a translation while trajectory *b* has significant changes in orientation.

the orientation held essentially constant. For the second trajectory (*b* in the figure), the position and the orientation of the camera were changed in a significant manner. Although highly accurate "ground-truth" data for the trajectory of the camera was obtained from the robotic manipulator, this trajectory was corrupted using a simulated error model so that the "true" and the a priori trajectories drifted away with time. The error model was quite extreme: 7.4 mm/s and 5 degrees/s, respectively. In order to compensate for this drift, the pose/motion estimation algorithm was called at 3/2 Hz rate. The two images used for the processing were the latest one available and a one-second-old frame. The a priori information was derived from the available drifted pose at these two frames. When used for a real navigation system, it might be preferable to use an adaptive time gap for the two frames, which takes into account the estimated velocities and the already-reconstructed trajectory. As was mentioned in the previous section, the magnitude of translation between the two frames is important to the accuracy and stability of the algorithm.

During the experiments, gray-scale images of 1024×768 were obtained using a Dragonfly video camera at a rate of 15 frames per second. Correspondence between about 400 features was derived using the Lucas-Kanade tracking method ([26], [6]). Features were not selected using an image-dependent algorithm but, rather, by using a regular grid spanned over the image plane. A typical frame and its features correspondence are shown in Fig. 11.

As shown in Figs. 12a and 12b, the algorithm converged to reasonable estimates for the navigation parameters along the two trajectories described above. The figures show the ground truth together with two trajectories computed using the error model: The first contains no updates while the second was updated periodically by using the pose/motion algorithm, at a 3/2 Hz rate. The figures clearly show that the corrected path remains close to the true path along the whole trajectory.

Fig. 13a shows the position errors of the drifted and corrected paths for experiment *b*. It can be seen that the errors of the corrected path are kept small while the errors in the uncompensated path increase gradually. Fig. 13b shows the orientation errors for the two computed paths. The sawtooth-shaped graph of the corrected path is characteristic: The orientation errors accumulate between updates but are strongly reduced each time the algorithm is used.



(a) (b)

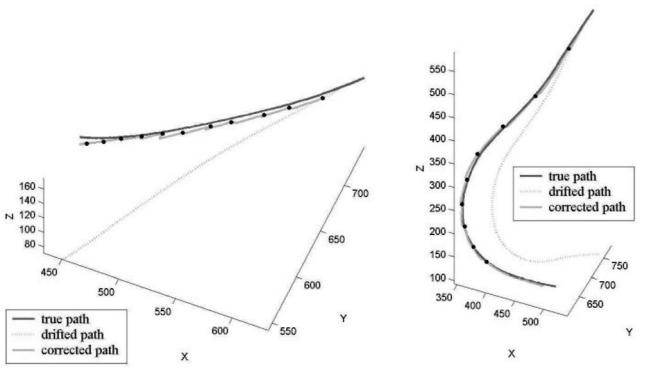
Fig. 11. (a) A frame taken from one of the camera's trajectories. (b) The estimated correspondence of 400 features taken from a 20×20 regular grid over the image plane of this frame.

5 A Comparison with an SFM and a Registration Algorithm

As mentioned in the introduction, the algorithm introduced in this paper is not the only possible approach to the problem at hand. An alternative is to divide the problem into subproblems and use existing algorithms as building blocks for a solution. For instance, one can formulate a two-step approach by first estimating the motion and structure using correspondences pairs and an SFM algorithm and then finding the pose by matching the reconstructed structure to the DTM. The purpose of this section is to present the implementation details for an algorithm as such and then compare its performance with the new one-step formulation. As the experiments confirm, the fact that the novel algorithm uses the DTM to constrain simultaneously pose *and* motion computation is advantageous over the two-step alternative.

5.1 The "SFM+ICP" Algorithm

In this section, the implementation details of the two-step algorithm are presented. Starting from correspondence pairs in two frames, numerous algorithms have been developed and studied for estimating ego-motion and reconstructing the scene. The algorithm presented in [30] was selected for the first step. In this work, the camera ego-motion was first derived and the structure of the scene was later reconstructed using the corresponding pairs and the



(a) (b)

Fig. 12. Experimental results for trajectories *a* and *b* (see Fig. 10). The diverging trajectories use the error model and no updates. The updated paths use the pose/motion algorithm to bound divergence.

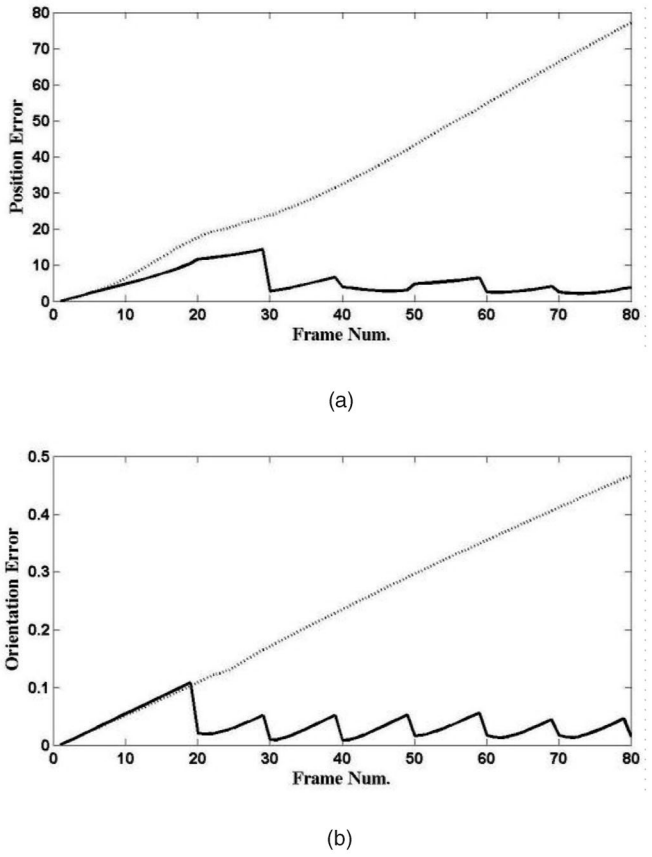


Fig. 13. (a) Position errors and (b) orientation errors of the drifted path (dotted line) and of the corrected path (solid line) of the second trajectory.

estimated motion. Being visual-based, this algorithm suffers from the velocity versus structure-scale ambiguity discussed in the introduction. Additionally, the algorithm makes no use of the DTM information and, hence, can only estimate camera motion.

Once the structure has been recovered, the "Iterative Closest Point" algorithm (ICP) can be used to estimate pose. By using the ICP algorithm presented by Chen and Medioni [9], the Euclidean transformation that best matches a set of points to a given surface can be estimated. In the present context, the points of the reconstructed structure given in the coordinates frame of the camera can be fed into the ICP algorithm to find the transformation, giving the best matching with the actual terrain surface as encoded by the DTM. Given that the SFM algorithm yields the scene structure only up to an unknown scale-factor, a slightly modified version of ICP is required, in which a *similarity* transformation is optimized instead of the more usual Euclidean one. The camera pose and the scale factor can be extracted easily from the estimated similarity transformation, and the scale factor ambiguity can be removed from the translational component of the ego-motion.

5.2 Performance Comparison

The performance of the algorithm presented in this paper was compared to the two-step approach discussed above by performing a large number of numerical experiments. In order to have a completely controlled environment, a 3×3 kilometer synthetic terrain was created, similar to the one used in the previous section (see Fig. 14b). Several different views were obtained using a virtual camera constrained to 600 meters above the terrain. A pure translation was selected as the virtual ego-motion, with a relatively large baseline of $\|p_{12}\| = 150m$. Observe that the length of the baseline should have a similar effect on both approaches to the problem.

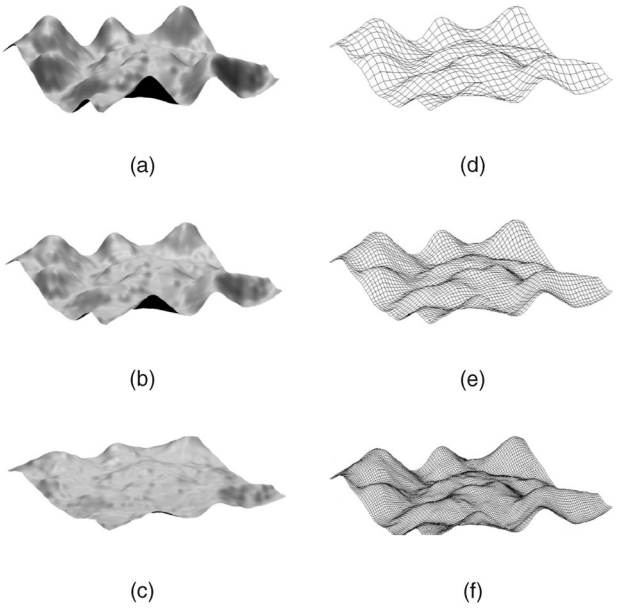


Fig. 14. The synthetic terrain was scaled to obtain a variety of elevation variations: (a) 800 m, (b) 600 m, (c) 300 m. Different DTMs were obtained for terrain (b) by sampling the terrain under different spatial grids (resolutions): (d) 100 m, (e) 50 m, (f) 30 m.

Performance was studied under different scenarios. Each scenario was characterized by the following parameters: The grid spacing of the DTM (also referred to as the DTM resolution), the altitude variations on the observed terrain, the resolution of images obtained by the virtual camera, and the number of corresponding pairs being used by the algorithm.

At each simulation, all parameters except for the one being tested were kept at predefined values. For example, in the *default scenario*, the terrain was scaled to contain 600 m elevation differences (Fig. 14b) and a DTM with a 50 m spatial grid was used as a model of the terrain (Fig. 14e). The camera is assumed to consist of 500×500 pixels and a maximum of 400 corresponding pairs were analytically derived prior to the calculations.

Each of the simulations described below studies the influence of a different parameter. A variety of values were examined and 150 random tests were performed for each tested value. For each test, the camera position and orientation were randomly selected (except for height over the terrain). Additionally, the direction of the ego-motion translation was chosen at random.

Fig. 15 shows the results of the first simulation where the resolution of the DTM was varied and its influence on the accuracy was studied. All parameters were set to their default values except for the DTM resolution, which was varied from 10 m up to a worst case of 100 m between adjacent grid points (see Figs. 14d, 14e, and 14f). Figs. 15a and 15b show that better estimates for the camera position and orientation were obtained by using the single-step algorithm, for all tested resolutions. Better estimates were

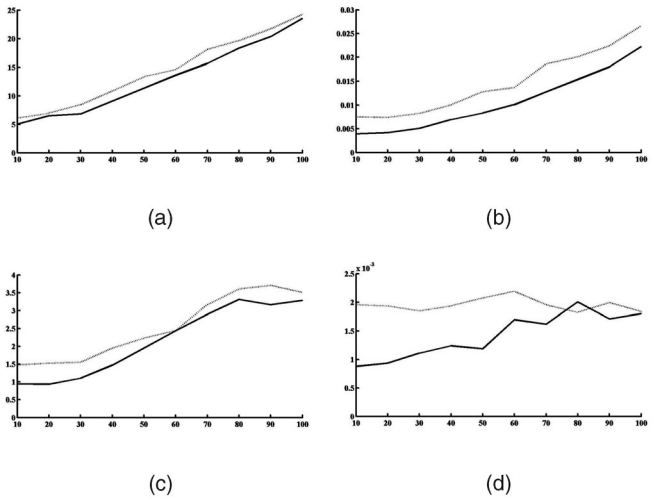


Fig. 15. Pose and ego-motion estimation accuracy using the single-step algorithm (solid line) and the SFM+ICP algorithm (dotted line) for different DTM resolutions. Resolution varies from 10 m to 100 m. (a) Position errors, in meters. (b) Orientation errors, in radians. (c) Motion translation errors, in meters. (d) Motion rotation errors, in radians.

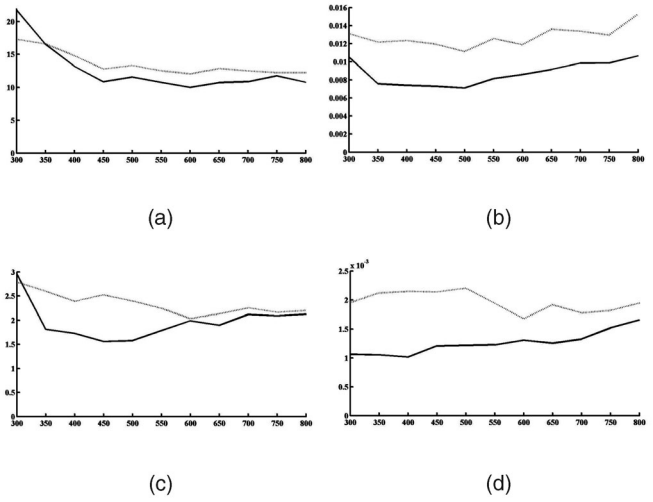


Fig. 16. Pose and ego-motion accuracies obtained by the single-step algorithm (solid line) and the SFM+ICP algorithm (dotted line) for terrains with varying elevation differences (from 300 m to 800 m). (a) Position error, in meters. (b) Orientation error, in radians. (c) Motion translation error, in meters. (d) Motion rotation error, in radians.

obtained for most ego-motion parameters as well, although this advantage becomes marginal as the DTM grid spacing increases (see Figs. 15c and 15d). This behavior was expected since the advantage of the single-step algorithm stems from the utilization of the DTM data for the ego-motion computation. Notice that the new algorithm strongly outperforms the two-step procedure when the grid-spacing is 40 m and better—a level compatible with modern DTM databases.

The next simulation demonstrates the relative importance of different terrain structures on the achievable accuracy. According to the discussion in Section 2.5, in the extreme scenario of flying above a planar terrain, the observed ground features do not contain the required information for the camera pose derivation and the system of equations becomes singular. As the slope and the variability of the terrain increases, the features become more informative and better estimates can be derived. For this simulation, the virtual terrain elevation differences were scaled to vary from 300 m to 800 m (Figs. 14a and 14c). As can be seen in Fig. 16, better estimates for the camera pose and motion were obtained by using the single-step algorithm, when elevation differences were greater than 350 m. However, as the terrain flattens, the advantage of the single-step algorithm can easily change to a disadvantage. Motion estimation is not directly influenced by the structure of the terrain when using the SFM algorithm. The single-step algorithm, on the other hand, estimates the pose and motion simultaneously. Hence, in a noninformative scenario of relatively flat terrain, pose and motion are drifting simultaneously, leading to an overall larger drift. As a demonstration of the above property, one can see how the gap between the two algorithms becomes small and even favors the SFM+ICP for low elevation differences.

As could be expected, performance is improved for both algorithms as image resolution increases. In the third set of simulations, the image resolution was varied from a low resolution of 200×200 to a high resolution of $1,000 \times 1,000$. Fig. 17 shows that the single-step algorithm achieves better pose accuracies for all resolutions. However, the gap between the two algorithms becomes small for very high resolutions and a small difference actually favors the SFM+ICP algorithm in ego-motion accuracies, as can be seen in Figs. 17c and 17d. This characteristic could be expected since the "fusion" of noisy DTM information in the motion computation can improve or damage the obtained accuracy compared with the SFM, which ignores this information. In the theoretical scenario of infinite image resolution, it is clear that a perfect motion estimate can be obtained using SFM (excluding translation scale), while the single-step algorithm will still diverge due to errors encoded in the DTM.

The final simulation compares the two algorithms for different numbers of corresponding features. The features

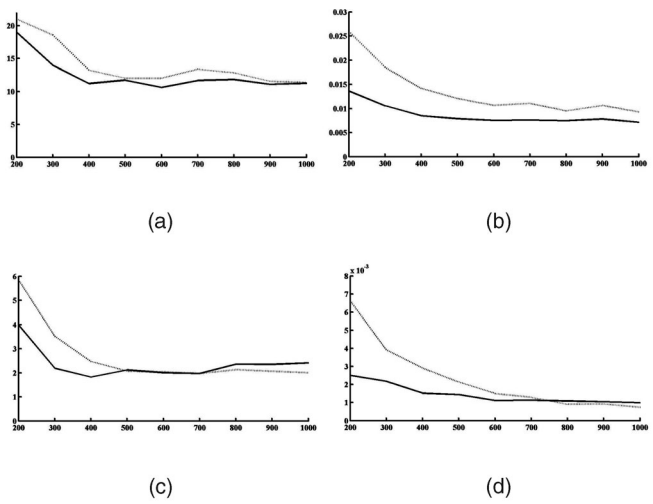


Fig. 17. Pose and ego-motion accuracy obtained by the single-step algorithm (solid line) and the SFM+ICP algorithm (dotted line) for different image resolutions (from 200×200 to $1,000 \times 1,000$). (a) Position error, in meters. (b) Orientation error, in radians. (c) Motion translation error, in meters. (d) Motion rotation error, in radians.

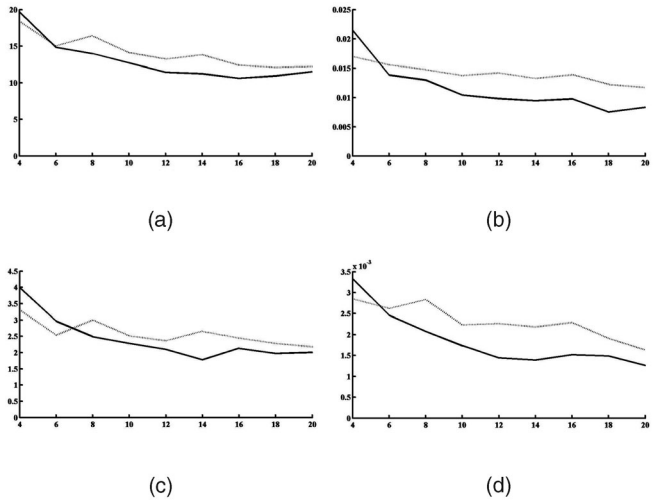


Fig. 18. Pose and ego-motion accuracy obtained by the single-step algorithm (solid line) and the SFM+ICP algorithm (dotted line) for different numbers of corresponding features pairs. The features were selected from a regular grid that was spanned over the image plane, where the grids resolutions varied from 4×4 (16 features) up to 20×20 (400 features). (a) Position error, in meters. (b) Orientation error, in radians. (c) Motion translation error, in meters. (d) Motion rotation error, in radians.

were not selected using an image-dependent selection algorithm but, rather, from a regular grid that was spanned over the image plane, where the resolution of the grid varies from 4×4 (16 features) up to 20×20 (400 features); see Fig. 9b for an illustration of this grid. Fig. 18 shows that the single-step algorithm achieves better estimates for the pose and motion parameters when at least 64 corresponding features were available. However, the gap between the two algorithms converges for large numbers of features. This result is due to the Gaussian error assumption on the image measurements that leads to improving the estimate of the navigation parameters as the number of features increases.

6 Conclusions and Further Work

This paper has introduced an algorithm for computing the pose (position and orientation) and motion (translation and rotation) of a calibrated camera with respect to an external reference system. The approach uses correspondence between feature points in two images and the information provided by a DTM to build a simultaneous constraint on the pose and motion variables. The constraint requires a priori information about the pose of the camera in the first frame and assumes that the DTM can be linearized in the sense discussed in Section 2. The final constraint is nonlinear and, hence, in general, needs to be solved by using numerical techniques. The constraint characterizing the two views plus DTM geometry presents several interesting features. First, at least six correspondence points are required to solve the different variables. Second, the formulation includes epipolar geometry, showing that the DTM effectively encodes additional valuable information about the 3D scene. Third, the constraint does not suffer from the ambiguity that haunts the SFM problem. A study of the degenerate scenarios was also presented. In addition to the theoretical results, the paper contains implementations details for the algorithm, including an M-estimator scheme for outliers elimination. Next, rather thorough numerical studies on synthetic and model-data are presented. The paper closes by comparing the performance of the novel algorithm to an alternative two-step algorithm constructed from "state of the art" building-blocks. A clear advantage has been shown for the novel algorithm in most reasonable scenarios.

Research is under way on several different aspects of the work presented here. First, an error analysis is being performed (see the forthcoming publication [21]) to understand the accuracy limitations of the algorithm. In particular, the balance between position and orientation errors requires further understanding. Second, the counterpart of the present algorithm using optical flow is of interest and can be derived along the lines of Section 2. Third, the fact that position and translation appear linearly in the constraints strongly suggests the possibility of using the current scheme in a filtering scheme. It seems that fusing the approach with inertial navigation may provide an effective and accurate inertial/optical navigation algorithm.

Acknowledgments

Part of this research was performed while H.P. Rotstein was a visiting professor at the Dept. of Aerospace Engineering and Mechanics, University of Minnesota, Minneapolis.

References

[1] "USGS Geographic Data Download Page," http://edc. usgs.gov/geodata/, 2005.
[2] P. Anandan, K. Hanna, and R. Kumar, "Shape Recovery from Multiple Views: A Parallax Based Approach," *Proc. IEEE Int'l Conf. Pattern Recognition,* vol. A, pp. 685-688, 1994.
[3] W. Baker and R. Clem, "Terrain Contour Matching [TERCOM] Primer," Technical Report ASP-TR-77-61, Aeronautical Systems Division, Wright-Patterson AFB, Aug. 1977.
[4] J.L. Barron and R. Eagleson, "Recursive Estimation of Time-Varying Motion and Structure Parameters," *Pattern Recognition,* vol. 29, no. 5, pp. 797-818, 1996.
[5] D. Boozer and J. Fellerhoff, "Terrain-Aided Navigation Test Results in the AFTI/F-16 Aircraft," *J. Inst. Navigation,* vol. 35, no. 2, pp. 161-175, 1988.
[6] J.Y. Bouguet,, "Pyramidal Implementation of the Lucas Kanade Feature Tracker, Description of the Algorithm," technical report, Intel Research Lab, 1999.
[7] D. Burschka and G. Hager, "V-GPS (SLAM): Vision-Based Inertial System for Mobile Robots," *Proc. IEEE Int'l Conf. Robotics and Automation,* vol. I, pp. 409-415, 2004.
[8] R. Chellappa, G. Qian, and S. Srinivasan, "Structure from Motion: Sparse versus Dense Correspondence Methods," *Proc. IEEE Int'l Conf. Image Processing,* pp. 492-499, 1999.
[9] Y. Chen and G. Medioni, "Object Modelling by Registration of Multiple Range Images," *Image and Vision Computing,* vol. 10, no. 3, pp. 145-155, 1992.
[10] A. Chiuso, R. Brockett, and S. Soatto, "Optimal Structure from Motion: Local Ambiguities and Global Estimates," *Int'l J. Comp. Vision,* vol. 39, no. 3, pp. 195-228, 2000.
[11] A. Chiuso, P. Favaro, H. Jin, and S. Soatto, "MFM: 3-D Motion from 2-D Motion Causally Integrated over Time," *Proc. European Conf. Comp. Vision,* 2000.
[12] D. DeMenthon and L. Davis, "Model-Based Object Pose in 25 Lines of Code," *Int'l J. of Computer Vision,* vol. 15, no. 1-2, pp. 123-141, 1995.
[13] R. Haralick, H. Joo, C. Lee, X. Zhuang, V. Vaidy, and M. Kim, "Pose Estimation from Corresponding Point Data," *IEEE Trans. Systems, Man, and Cybernetics,* vol. 19, no. 6, pp. 1426-1446, 1989.
[14] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision.* Cambridge Univ. Press, 2000.
[15] D. Heeger and A. Jepson, "Subspace Methods for Recovering Rigid Motion I: Algorithm and Implementation," *Int'l J. Computer Vision,* vol. 7, pp. 95-117, 1992.

[16] Y. Hel-Or and M. Werman, "Absolute Orientation from Uncertain Point Data: A Unified Approach," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 77-82, 1992.
[17] *Understanding Robust and Exploratory Data Analysis,* D. Hoaglin, F. Mosteller, and J. Tukey, eds. John Wiley & Sons, 1983.
[18] S.. Hsu, S. Samarasekera, R. Kumar, and H. Sawhney, "Pose Estimation, Model Refinement, and Enhanced Visualization Using Video," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* vol. 1, pp. 448-495, 2000.
[19] M. Irani, B. Rousso, and S. Peleg, "Robust Recovery of Ego-Motion," *Proc. Conf. Computer Analysis Images and Patterns,* pp. 371-378, 1993.
[20] D. Jacobs and R. Basri, "3-D to 2-D Pose Determination with Regions," *Int'l J. Computer Vision,* vol. 34, no. 2-3, pp. 123-145, 1999.
[21] R. Lerner, H. Rotstein, and E. Rivlin, "Error Analysis of an Algorithm for Pose and Motion Recovery from Correspondence and a Digital Terrain Map," in preparation.
[22] R. Lerner, P. Rotstein, and E. Rivlin, "Error Analysis for a Navigation Algorithm Based on Optical-Flow and a Digital Terrain Map," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* vol. 1, pp. 604-610, 2004.
[23] Y. Liu, T. Huang, and O. Faugeras, "Determination of Camera Location from 2-D to 3-D Line and Point Correspondences," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 12, no. 1, pp. 28-37, Jan. 1990.
[24] H.C. Longuet-Higgins, "A Computer Algorithm for Reconstructing a Scene from Two Projections," *Readings in Computer Vision: Issues, Problems, Principles, and Paradigms,* M.A. Fischler and O. Firschein, eds., pp. 61-62, Los Altos, Calif.: Kaufmann, 1987.
[25] C. Lu, G. Hager, and E. Mjolsness, "Fast and Globally Convergent Pose Estimation from Video Images," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 22, no. 6, pp. 610-622, June 2000.
[26] B.D. Lucas and T. Kanade, "An Iterative Image Registration Technique with an Application to Stereo Vision," *Proc. Seventh Int'l Joint Conf. Artificial Intelligence,* pp. 674-679, 1981.
[27] D. Nister, "A Minimal Solution to the Generalised 3-Point Pose Problem," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* vol. 1, pp. 560-567, 2004.
[28] J. Oliensis, "A Multi-Frame Structure from Motion Algorithm under Perspective," *Int'l J. Computer Vision,* vol. 34, no. 2, pp. 163-192, 1999.
[29] J. Oliensis, "A Critique of Structure-from-Motion Algorithms," *Computer Vision and Image Understanding,* vol. 80, pp. 172-214, 2000.
[30] J. Oliensis, "Exact Two-Image Structure from Motion," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 24, pp. 1618-1633, 2002.
[31] J. Rodriguez and J.K. Aggarwal, "Matching Aerial Images to 3-D Terrain Maps," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 12, no. 12, pp. 1138-1149, Dec. 1990.
[32] S. Seitz and C. Dyer, "Complete Scene Structure from Four Point Correspondences," *Proc. IEEE Int. Conf. Computer Vision,* pp. 330-337, 1995.
[33] I. Shimshoni, R. Basri, and E. Rivlin, "A Geometric Interpretation of Weak-Perspective Motion," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 21, no. 3, pp. 252-257, Mar. 1999.
[34] H. Shum, Q. Ke, and Z. Zhang, "Efficient Bundle Adjustment with Key Frames: A Hierarchical Approach to Multi-Frame Structure from Motion," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* vol. 2, pp. 538-543, 1999.
[35] D.G. Sim, R.H. Park, R.C. Kim, S.U. Lee, and I.C. Kim, "Integrated Position Estimation Using Aerial Image Sequences," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 24, no. 1, pp. 1-18, Jan. 2002.
[36] D. Simon, "Fast and Accurate Shape-Based Registration," Technical Report CMU-RI-TR-96-45, Robotics Inst., Carnegie Mellon Univ., Dec. 1996.
[37] S. Srinivasan, "Extracting Structure from Optical Flow Using the Fast Error Search Technique," *Int'l J. Computer Vision,* vol. 37, no. 3, pp. 203-230, 2000.
[38] P. Sturm and B. Triggs, "A Factorization Based Algorithm for Multi-Image Projective Structure and Motion," *Proc. European Conf. Computer Vision,* pp. 709-720, 1996.
[39] H. Takeda, C. Facchinetti, and J. Latombe, "Planning the Motion of a Mobile Robot in a Sensory Uncertainty Field," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 16, no. 10, pp. 1002-1017, Oct. 1994.
[40] T. Tan, K. Baker, and G. Sullivan, "3D Structure and Motion Estimation from 2D Image Sequences," *Image and Vision Computing,* vol. 11, pp. 203-210, 1993.
[41] C. Taylor and D. Kriegman, "Structure and Motion from Line Segments in Multiple Images," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 17, pp. 1021-1032, 1995.
[42] B. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon, "Bundle Adjustment—A Modern Synthesis," W. Triggs, A. Zisserman, and R. Szeliski, eds., *Vision Algorithms: Theory and Practice,* pp. 298-375, Springer Verlag, 2000.
[43] J. Weng, T. Huang, and N. Ahuja, "Motion and Structure from Two Perspective Views: Algorithms, Error Analysis, and Error Estimation," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 11, no. 5, pp. 451-476, May 1989.
[44] T. Wu, R. Chellappa, and Q. Zheng, "Experiments on Estimating Egomotion and Structure Parameters Using Long Monocular Image Sequences," *Int'l J. Computer Vision,* vol. 15, no. 1-2, pp. 77-103, 1995.
[45] Z. Zhang, "Motion and Structure from Two Perspective Views: From Essential Parameters to Euclidean Motion through the Fundamental Matrix," *J. Optical Soc. of Am.,* vol. 14, no. 11, pp. 2938-2950, 1997.

Ronen Lerner received the BSc degree in computer science and mathematics from the University of Haifa and the MSc degree from the Technion, Israel Institute of Technology. He is currently studying toward the PhD degree in the Computer Science Department at the Technion—Israel Institute of Technology. His current research interest is in vision-based navigation.

Ehud Rivlin received the BSc and MSc degrees in computer science and the MBA degree from the Hebrew University in Jerusalem and the PhD from the University of Maryland. Currently, he is an associate professor in the Computer Science Department at the Technion—Israel Institute of Technology. His current research interests are in machine vision and robot navigation. He is a member of the IEEE.

Héctor P. Rotstein received the ingeniero electricista degree from the Universidad Nacional del Sur, Argentina, and MSc and PhD degrees from the California Institute of Technology. Since 1997, he has been with the Missile Division of Rafael, where he is now a chief research engineer in control and navigation systems. His research interests include robust and vision-based control and the design of integrated navigation systems. Dr. Rotstein is also a cofounder of M&H Engineering Consultant and has been recently involved in the design of the Scolio-Scan system for detecting infant scoliosis. He is a member of the IEEE.