3-D Motion and Structure from 2-D Motion Causally Integrated over Time: Implementation[⋆]

Alessandro Chiuso[†‡], Paolo Favaro[†], Hailin Jin[†], and Stefano Soatto[†]

[†] Washington University, One Brookings Dr. 1127, Saint Louis – MO 63130
[‡] Università di Padova, Via Gradenigo 6/a 35131 Padova – Italy
email {chiuso,fava,hljin,soatto}@essrl.wustl.edu

Abstract. The causal estimation of three-dimensional motion from a sequence of two-dimensional images can be posed as a nonlinear filtering problem. We describe the implementation of an algorithm whose uniform observability, minimal realization and stability have been proven analytically in [5]. We discuss a scheme for handling occlusions, drift in the scale factor and tuning of the filter. We also present an extension to partially calibrated camera models and prove its observability. We report the performance of our implementation on a few long sequences of real images. More importantly, however, we have made our real-time implementation – which runs on a personal computer – available to the public for first-hand testing.

1 Introduction

Inferring the three-dimensional (3-D) shape of a moving scene from its two-dimensional images is one of the classical problems of computer vision, known by the name of "shape from motion" (SFM). Among all possible ways in which this can be done, we distinguish between *causal schemes* and *non-causal ones.* More than the fact that causal schemes use – at any given point in time – only information from the past, the main difference between these two approaches lies in their goals and in the way in which data are collected. When the estimates of motion are to be used in real time, for instance to accomplish a control task, a causal scheme must be employed since "future" data are not available for processing and the control action must be taken "now". In that case, the sequence of images is often collected sequentially in time, while motion changes smoothly under the auspices of inertia, gravity and other physical constraints. When, on the other hand, we collect a number of "snapshots" of a scene from disparate viewpoints and we are interested in reconstructing it, there is no natural ordering or smoothness involved; using a causal scheme in this case would be, in the end, highly unwise.

No matter how the data are collected, however, SFM is subject to fundamental tradeoffs, which we articulate in section 1.2. This paper aims at addressing

[⋆] Supported by NSF IIS-9876145 and ARO DAAD19-99-1-0139. We wish to thank Xiaolin Feng, Carlo Tomasi, Pietro Perona, Ruggero Frezza, John Oliensis and Philip McLauchlan for discussions.

such tradeoffs: it is possible to integrate visual information over time, hence achieving a global estimate of 3-D motion, while maintaining the correspondence problem local. Among the obstacles we encounter is the fact that individual points tend to become occluded during motion, while novel points become visible. In [5] we have introduced a wide-sense approximation to the optimal filter and proved that it is observable, minimal and stable. In this paper we describe a complete, real-time *implementation* of the algorithm, which includes an approach to handle *occlusions* causally.

1.1 A first formalization of the problem

Consider an N-tuple of points in the three-dimensional Euclidean space, represented as a matrix

$$\mathbf{X} \doteq \left[\mathbf{X}^1 \; \mathbf{X}^2 \ldots \mathbf{X}^N\right] \in \mathbb{R}^{3\times N} \tag{1}$$

and let them move under the action of a rigid motion represented by a translation vector T and a rotation matrix R. Rotation matrices are orthogonal with unit determinant $\{R \mid R^T R = RR^T = I\}$. Rigid motions transform the coordinates of each point via $R(t)\mathbf{X}^i + T(t)$. Associated to each motion $\{T, R\}$ there is a velocity, represented by a vector of linear velocity V and a skew-symmetric matrix $\widehat{\omega}$ of rotational velocity. Skew-symmetric 3×3 matrices are represented using the "hat" notation

$$\widehat{\mathbf{a}} = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix}. \tag{2}$$

Under such velocity, motion evolves according to

$$\begin{cases} T(t+1) = e^{\widehat{\omega(t)}} T(t) + V(t) \\ R(t+1) = e^{\widehat{\omega(t)}} R(t). \end{cases} \tag{3}$$

The exponential of a skew-symmetric matrix can be computed conveniently using Rodrigues' formula:

$$e^{\widehat{\omega}} = I + \frac{\widehat{\omega}}{\|\omega\|} \sin\left(\|\omega\|\right) + \frac{\widehat{\omega}^2}{\|\omega\|^2}\left(1 - \cos\left(\|\omega\|\right)\right). \tag{4}$$

We assume that - to an extent discussed in later sections - the *correspondence problem* is solved, that is we know which point corresponds to which in different projections (views). Equivalently, we assume that we can measure the (noisy) projection

$$\mathbf{y}^i(t) = \pi\left(R(t)\mathbf{X}^i + T(t)\right) + \mathbf{n}^i(t) \in \mathbb{R}^2 \quad \forall\, i = 1 \ldots N \tag{5}$$

where we know the correspondence $\mathbf{y}^i \leftrightarrow \mathbf{X}^i$. We take as projection model an ideal pinhole, so that $\mathbf{y} = \pi(\mathbf{X}) = \left[\frac{X_1}{X_3} \; \frac{X_2}{X_3}\right]^T$. This choice is not crucial and the discussion can be easily extended to other projection models (e.g. spherical, orthographic, para-perspective, etc.). We do not distinguish between $\mathbf{y}$ and its projective coordinate (with a 1 appended), so that we can write $\mathbf{X} = \mathbf{y}X_3$.

Finally, by organizing the time-evolution of the configuration of points and their motion, we end up with a discrete-time, non-linear dynamical system:

$$\begin{cases} \mathbf{X}(t+1) = \mathbf{X}(t) & \mathbf{X}(0) = \mathbf{X}_0 \\ T(t+1) = e^{\widehat{\omega(t)}} T(t) + V(t) & T(0) = T_0 \\ R(t+1) = e^{\widehat{\omega(t)}} R(t) & R(0) = R_0 \\ V(t+1) = V(t) + \alpha_V(t) & V(0) = V_0 \\ \omega(t+1) = \omega(t) + \alpha_\omega(t) & \omega(0) = \omega_0 \\ \mathbf{y}^i(t) = \pi\left(R(t)\mathbf{X}^i(t) + T(t)\right) + \mathbf{n}^i(t) & \mathbf{n}^i(t) \sim \mathcal{N}(0, \Sigma_n) \end{cases} \tag{6}$$

where $\mathbf{v} \sim \mathcal{N}(M, S)$ indicates that a vector $\mathbf{v}$ is distributed normally with mean M and covariance S. In the above system, α is the relative acceleration between the viewer and the scene. If some prior modeling information is available (for instance when the camera is mounted on a vehicle or on a robot arm), this is the place to use it. Otherwise a statistical model can be employed. In particular, we can formalize our ignorance on acceleration by modeling α as a Brownian motion process[1]. In principle one would like - at least for this simplified formalization of SFM - to find the optimal solution. Unfortunately, as we explain in [5], there exists no finite-dimensional optimal filter for this model. Therefore, at least for this elementary instantiation of SFM, we would like to derive approximations that are provably stable and efficient.

1.2 Tradeoffs in structure from motion

The first tradeoff involves the magnitude of the baseline and the correspondence problem, and has been discussed extensively in [5]. When images are taken from disparate viewpoints, estimating relative orientation is simple, given the correspondence. However, solving the correspondence problem is difficult, for it amounts to a global matching problem – all too often solved by hand – which spoils the possibility of use in real-time control systems. When images are collected closely in time, on the other hand, correspondence becomes an easy-to-solve local variational problem. However, estimating 3-D motion becomes rather difficult since – on small motions – the noise in the image overwhelms the feeble information contained in the 2-D motion of the features.

No matter how one chooses to increase the baseline in order to bypass the tradeoff with correspondence, one inevitably runs into deeper problems, namely the fact that individual feature points can *appear and disappear due to occlusions*, or to changes in their appearance due to specularities, light distribution etc. To increase the baseline, it is necessary to associate the scale factor to an invariant of the scene. Therefore, in order to process that information, the scale factor must be included in the model. This tradeoff is fundamental and there is no easy way around it: information on shape can only be integrated as long as the shape is visible.

[1] We wish to emphasize that this choice is not crucial towards the conclusions reached in this paper. Any other model would do, as long as the overall system is observable.

1.3 Relation to previous work and organization of the paper

We are interested in estimating motion so that we can use the estimates to accomplish spatial control tasks such as moving, tracking, manipulation etc. In order to do so, the estimates must be provided *in real time and causally*, while we can rely on the fact that images are taken at adjacent instants in time and the relative motion between the scene and the viewer is somewhat smooth (rather than having isolated "snapshots"). Therefore, we do not compare our algorithms with batch multi-frame approaches to SFM. This includes iterative minimization techniques such as "bundle adjustment". If one can afford the time for processing sequences of images off-line, of course a batch approach that optimizes simultaneously on all frames will perform better![2]

Our work falls within the category of causal motion and structure estimation that has a long and rich history [10, 7, 18, 4, 23, 19, 32, 9, 30, 24, 8, 12, 26, 11, 31, 34, 33, 13, 25, 16, 1, 2, 22, 35, 15]. The first attempts to prove stability of the schemes proposed are recent [21]. The first attempts to handle occlusions in a causal scheme[3] came only a few years ago [19, 29]. Our approach is similar in spirit to the work of Azarbayejani and Pentland [2], extended to handle occlusions and to give correct weighting to the measurements.

The first part of this study [5] contains a proof of uniform observability and stability of the algorithm that we describe here. In passing, we show how the conditions we impose on our models are tight: imposing either more or less results in either a biased or an unstable filter. The second part, reported in this paper, is concerned with the *implementation* of a system working in real time on real scenes, which we have made available to the public [14].

2 Realization

In order to design a finite-dimensional approximation to the optimal filter, we need an observable realization of the original model[4]. In [5] we have proven the following claim.

Corollary 1 *The model*

$$\begin{cases} \mathbf{y}_0^i(t+1) = \mathbf{y}_0^i(t) & i = 4 \dots N & \mathbf{y}_0^i(0) = \mathbf{y}_0^i \\ \rho^i(t+1) = \rho^i(t) & i = 2 \dots N & \rho^i(0) = \rho_0^i \\ T(t+1) = \exp(\widehat{\omega}(t))T(t) + V(t) & & T(0) = T_0 \\ \Omega(t+1) = Log_{SO(3)}(\exp(\widehat{\omega}(t))\exp(\widehat{\Omega}(t))) & & \Omega(0) = \Omega_0 \\ V(t+1) = V(t) + \alpha_V(t) & V(0) = V_0 & \\ \omega(t+1) = \omega(t) + \alpha_\omega(t) & \omega(0) = \omega_0 & \\ \mathbf{y}^i(t) = \pi\left(\exp(\widehat{\Omega}(t))\mathbf{y}_0^i(t)\rho^i(t) + T(t)\right) + n^i(t) & & i = 1 \dots N. \end{cases} \tag{7}$$

[2] One may argue that batch approaches are now fast enough that they can be used for real-time processing. Our take on this issue is exposed in [5], where we argue that speed is not the problem; robustness and delays are.

[3] There are several ways of handling missing data in a batch approach: since they do not extend to causal processing, we do not review them here.

[4] Observability in SFM has been addressed first in 1994 [6, 27] (see also [28] for a more complete account of these results). Observability is closely related to "gauge invariance" [20].

is a minimal realization of (6). The notation $Log_{SO(3)}(R)$ stands for Ω such that $R = e^{\widehat{\Omega}}$ and is computed by inverting Rodrigues' formula[5]*. Ω is called the "canonical representation" of R.*

Remark 1 *Notice that in the above claim the index for $\mathbf{y}_0^i$ starts at 4, while the index for ρ^i starts at 2. This corresponds to choosing the first three points as reference for the similarity group and is necessary (and sufficient) for guaranteeing that the representation is minimal. As explained in [5] this can be done without loss of generality, i.e. modulo a reordering of the states.*

2.1 Partial autocalibration

As we have anticipated, the models proposed can be extended to account for changes in calibration. For instance, if we consider an imaging model with focal length f[6]

$$\pi_f(\mathbf{X}) = \frac{f \begin{bmatrix} X_1 \\ X_2 \end{bmatrix}}{X_3} \tag{8}$$

where the focal length can change in time, but no prior knowledge on how it does so is available, one can model its evolution as a random walk

$$f(t+1) = f(t) + \alpha_f(t) \qquad \alpha_f(t) \sim \mathcal{N}(0, \sigma_f^2) \tag{9}$$

and insert it into the state of the model (6). As long as the overall system is observable, the conclusions reached in [5] will hold. The following claim shows that this is the case for the model (9) above. Another imaging model proposed in the literature is [2]: $\pi_\beta(\mathbf{X}) = \frac{\begin{bmatrix} X_1 & X_2 \end{bmatrix}^T}{1+\beta X_3}$ for which similar conclusions can be drawn. The reader can refer to [5] for details on definitions and characterizations of observability.

Proposition 1 *Let $g = \{T, R\}$ and $v = \{V, \omega\}$. The model*

$$\begin{cases} \mathbf{X}(t+1) = \mathbf{X}(t) & \mathbf{X}(0) = \mathbf{X}_0 \\ g(t+1) = e^{\widehat{v}} g(t) & g(0) = g_0 \\ v(t+1) = v(t) & v(0) = v_0 \\ f(t+1) = f(t) & f(0) = f_0 \\ \mathbf{y}(t) = \pi_f(g(t)\mathbf{X}(t)) \end{cases} \tag{10}$$

is observable up to the action of the group represented by $\tilde{T}, \tilde{R}, \alpha$ acting on the initial conditions.

Proof: *Consider the diagonal matrix $F(t) = \text{diag}\{f(t),\ f(t),\ 1\}$ and the matrix of scalings $A(t)$ as in the proof of proposition 1 in [5]. Consider then two initial conditions*

[5] A Matlab implementation of $Log_{SO(3)}$ is included in the software distribution.

[6] This f is not to be confused with the generic state equation of the filter in section 3.3.

$\{\mathbf{X}_1, g_1, v_1, f_1\}$ *and* $\{\mathbf{X}_2, g_2, v_2, f_2\}$. *For them to be indistinguishable there must exist matrices of scalings* $A(k)$ *and of focus* $F(k)$ *such that*

$$\begin{cases} g_1\mathbf{X}_1 = F(1)(g_2\mathbf{X}_2) \cdot A(1) \\ e^{\widehat{v_1}} e^{(k-1)\widehat{v_1}} g_1\mathbf{X}_1 = F(k+1)\left(e^{\widehat{v_2}} e^{(k-1)\widehat{v_2}} g_2\mathbf{X}_2\right) \cdot A(k+1) \qquad k \geq 1. \end{cases} \tag{11}$$

Making the representation explicit we obtain

$$\begin{cases} R_1\mathbf{X}_1 + \bar{T}_1 = F(1)(R_2\mathbf{X}_2 + \bar{T}_2)A(1) \\ U_1F(k)\tilde{\mathbf{X}}_kA(k) + \bar{V}_1 = F(k+1)(U_2\tilde{\mathbf{X}}_k + \bar{V}_2)A(k+1) \end{cases} \tag{12}$$

which can be re-written as

$$\tilde{\mathbf{X}}_kA(k)A^{-1}(k+1) - F^{-1}(k)U_1^TF(k+1)U_2\tilde{\mathbf{X}}_k = F(k)^{-1}U_1^T(F(k+1)\bar{V}_2A(k+1) - \bar{V}_1)A^{-1}(k+1). \tag{13}$$

The two sides of the equation have equal rank only if it is equal to zero, which draws us to conclude that $A(k)A^{-1}(k+1) = I$, *and hence* A *is constant. From* $F^{-1}(k)U_1^TF(k+1)U_2 = I$ *we get that* $F(k+1)U_2 = U_1F(k)$ *and, since* $U_1, U_2 \in SO(3)$, *we have that taking the norm of both sides* $2f^2(k+1) + 1 = 2f^2(k) + 1$, *where* f *must be positive, and therefore constant:* $FU_2 = U_1F$. *From the right hand side we have that* $F\bar{V}_2A = \bar{V}_1$, *from which we conclude that* $A = \alpha I$, *so that in vector form we have* $V_1 = \alpha FV_2$. *Therefore, from the second equation we have that, for any* f *and any* α, *we can have* $V_1 = \alpha FV_2$, $U_1 = FU_2F^{-1}$ *However, from the first equation we have that* $R_1\mathbf{X}_1 + T_1 = \alpha FR_2\mathbf{X}_2 + \alpha FT_2$, *whence - from the general position conditions - we conclude that* $R_1 = \alpha FR_2$ *and therefore* $F = I$. *From that we have that* $T_1 = \alpha FT_2 = \alpha T_2$ *which concludes the proof.*

Remark 2 *The previous claim essentially implies that the realization remains minimal if we add into the model the focal parameter. Note that observability depends upon the structural properties of the model,* not *on the noise, which is therefore assumed to be zero for the purpose of the proof.*

2.2 Saturation

Instead of eliminating states to render the model observable, it is possible to design a nonlinear filter directly on the (unobservable) model (6) by *saturating* the filter along the unobservable component of the state space as we show in this section. In other words, it is possible to design the initial variance of the state of the estimator as well as its model error in such a way that it will never move along the unobservable component of the state space.

As proposition 2 in [5] suggests, one can saturate the states corresponding to $\mathbf{y}_0^1, \mathbf{y}_0^2, \mathbf{y}_0^3$ and ρ^1. We have to guarantee that the filter initialized at $\widehat{\mathbf{y}}_0, \widehat{\rho}_0, \widehat{g}_0, \widehat{v}_0$ evolves in such a way that $\widehat{\mathbf{y}}_0^1(t) = \widehat{\mathbf{y}}_0^1, \widehat{\mathbf{y}}_0^2(t) = \widehat{\mathbf{y}}_0^2, \widehat{\mathbf{y}}_0^3(t) = \widehat{\mathbf{y}}_0^3, \widehat{\rho}^1(t) = \widehat{\rho}_0^1$. It is simple, albeit tedious, to prove the following proposition.

Proposition 2 *Let* $P_{\mathbf{y}^i}(0), P_{\rho^i}(0)$ *denote the variance of the initial condition corresponding to the state* $\mathbf{y}_0^i$ *and* ρ^i *respectively, and* $\Sigma_{\mathbf{y}^i}, \Sigma_{\rho^i}$ *the variance of the model error corresponding to the same state, then* $P_{\mathbf{y}^i}(0) = 0$, $\Sigma_{y^i} = 0 \quad i = 1 \ldots 3 \quad \Sigma_{\rho^1} = 0$ *implies that* $\widehat{\mathbf{y}}_0^i(t|t) = \widehat{\mathbf{y}}_0^i(0), \quad i = 1 \ldots 3,$ *and* $\widehat{\rho}^1(t|t) = \widehat{\rho}^1(0)$.

2.3 Pseudo-measurements

Yet another alternative to render the model observable is to add pseudo-measurement equations with zero error variance.

Proposition 3 *The model*

$$\begin{cases} \mathbf{y}_0^i(t+1) = \mathbf{y}_0^i(t) & i = 1 \dots N \qquad \mathbf{y}_0^i(0) = \mathbf{y}_0^i \\ \rho^i(t+1) = \rho^i(t) & i = 1 \dots N \qquad \rho^i(0) = \rho_0^i \\ T(t+1) = \exp(\widehat{\omega}(t))T(t) + V(t) & T(0) = 0 \\ \Omega(t+1) = Log_{SO(3)}(\exp(\widehat{\omega}(t))\exp(\widehat{\Omega}(t))) & \Omega(0) = 0 \\ V(t+1) = V(t) + \alpha_V(t) & V(0) = V_0 \\ \omega(t+1) = \omega(t) + \alpha_\omega(t) & \omega(0) = \omega_0 \\ \mathbf{y}^i(t) = \pi\left(\exp(\widehat{\Omega}(t))\mathbf{y}_0^i(t)\rho^i(t) + T(t)\right) + n^i(t) & i = 1 \dots N \\ \rho^1 = \psi_1 \\ \mathbf{y}_0^i(t) = \phi^i \quad i = 1 \dots 3, \end{cases} \tag{14}$$

where ψ_1 is an arbitrary (positive) constant and ϕ^i are three non-collinear points on the plane, is observable.

3 Implementation: occlusions and drift in SFM

The implementation of an extended Kalman filter based upon the model (7) is straightforward. However, for the sake of completeness we report it in section 3.3. The only issue that needs to be dealt with is the disappearing and appearing of feature points, a common trait of sequences of images of natural scenes. Visible feature-points may become occluded (and therefore their measurements become unavailable), or occluded points may become visible (and therefore provide further measurements). New states must be properly initialized. One way of doing so is described in the next section 3.1. Occlusion of point features do not cause major problems, unless the feature that disappears happens to be associated with the scale factor. This is unavoidable and results in a drift whose nature is explained in section 3.2.

3.1 Occlusions

When a feature point, say $\mathbf{X}^i$, becomes occluded, the corresponding measurement $\mathbf{y}^i(t)$ becomes unavailable. It is possible to model this phenomenon by setting the corresponding variance to infinity or, in practice $\Sigma_{n^i} = MI_2$ for a suitably large scalar $M > 0$. By doing so, we guarantee that the corresponding states $\hat{\mathbf{y}}_0^i(t)$ and $\hat{\rho}^i(t)$ are not updated:

Proposition 4 *If $\Sigma_{n^i} = \infty$, then $\hat{\mathbf{y}}_0^i(t+1) = \hat{\mathbf{y}}_0^i(t)$ and $\hat{\rho}^i(t+1) = \hat{\rho}^i(t)$.*

An alternative, which is actually preferable in order to avoid useless computation and ill-conditioned inverses, is to eliminate the states $\hat{\mathbf{y}}_0^i$ and $\hat{\rho}^i$ altogether, thereby reducing the dimension of the state-space. This is simple due to the diagonal structure of the model (7): the states ρ^i, $\mathbf{y}_0^i$ are decoupled, and therefore it is sufficient to remove them, and delete the corresponding rows from the gain matrix $K(t)$ and the variance $\Sigma_w(t)$ for all t past the disappearance of the feature (see section 3.3).

When a new feature-point appears, on the other hand, it is not possible to simply insert it into the state of the model, since the initial condition is unknown. Any initialization error will disturb the current estimate of the remaining states, since it is fed back into the update equation for the filter, and generates a spurious transient. We address this problem by running a separate filter in parallel for each point using the current estimates of motion from the main filter in order to reconstruct the initial condition. Such a "subfilter" is based upon the following model, where we assume that N_τ features appear at time τ:

$$\begin{cases} \mathbf{y}_\tau^i(t+1) = \mathbf{y}_\tau^i(t) + \eta_{y^i(t)} & i = 1 \dots N_\tau \qquad \mathbf{y}_\tau^i(0) \sim \mathcal{N}(\mathbf{y}^i(\tau), \Sigma_{n^i}) \qquad t > \tau \\ \rho_\tau^i(t+1) = \rho_\tau^i(t) + \eta_{\rho^i(t)} & i = 1 \dots N_\tau \qquad \rho^i(0) \sim \mathcal{N}(1, P_\rho(0)) \\ \mathbf{y}^i(t) = \pi\left(\exp(\widehat{\Omega}(t|t))\left[\exp(\widehat{\Omega}(\tau|\tau))\right]^{-1}\left[\mathbf{y}_\tau^i(t)\rho_\tau^i(t) - T(\tau|\tau)\right] + T(t|t)\right) + n^i(t) \end{cases} \tag{15}$$

where $\Omega(t|t)$ and $T(t|t)$ are the current best estimates of Ω and T, $\Omega(\tau|\tau)$ and $T(\tau|\tau)$ are the best estimates of Ω and T at $t = \tau$. In pracice, rather than initializing ρ to 1, one can compute a first approximation by triangulating on two adjacent views, and compute covariance of the initialization error from the covariance of the current estimates of motion. Several heuristics can be employed in order to decide when the estimate of the initial condition is good enough for it to be inserted into the main filter. The most natural criterion is when the variance of the estimation error of ρ_τ^i in the subfilter is comparable with the variance of ρ_0^j for $j \neq i$ in the main filter. The last step in order to insert the feature i into the main filter consists in bringing the coordinates of the new points back to the initial frame. This is done by

$$\mathbf{X}^i = \left[\exp(\widehat{\Omega}(\tau|\tau))\right]^{-1}\left[\mathbf{y}_\tau^i\rho_\tau^i - T(\tau|\tau)\right]. \tag{16}$$

3.2 Drift

The only case when losing a feature constitutes a problem is when it is used to fix the observable component of the state-space (in our notation, $i = 1, 2, 3$) as explained in [5] [7]. The most obvious choice consists in associating the reference to any other visible point. This can be done by saturating the corresponding state and assigning as reference value the current best estimate. In particular, if feature i is lost at time τ, and we want to switch the reference index to feature

[7] When the scale factor is not directly associated to one feature, but is associated to a function of a number of features (for instance the depth of the centroid, or the average inverse depth), then losing any of these features causes a drift. See [5] for more details.

j, we eliminate $\mathbf{y}_0^i$, ρ^i from the state, and set the diagonal block of Σ_w and $P(\tau)$ with indices $3j-3$ to $3j$ to zero. Therefore, by proposition 2, we have that

$$\hat{\mathbf{y}}_0^j(\tau+t) = \hat{\mathbf{y}}_0^j(\tau) \quad \forall\, t > 0. \tag{17}$$

If $\hat{\mathbf{y}}_0^j(\tau)$ was equal to $\mathbf{y}_0^j$, switching the reference feature would have no effect on the other states, and the filter would evolve on the same observable component of the state-space defined by the reference feature i.

However, in general the difference $\tilde{\mathbf{y}}_0^j(\tau) \doteq \mathbf{y}_0^j(\tau) - \hat{\mathbf{y}}_0^j$ is a random variable with variance $\Sigma_\tau = P_{3j-3:3j-1,3j-3:3j-1}$. Therefore, switching the reference to feature j causes the observable component of the state-space to move by an amount proportional to $\tilde{\mathbf{y}}_0^j(\tau)$. When a number of switches have occurred, we can expect - on average - the state-space to move by an amount proportional to $\|\Sigma_\tau\|$#switches. As we discussed in section 1.2, this is unavoidable. What we can do is at most try to keep the bias to a minimum by switching the reference to the state that has the lowest variance[8].

Of course, should the original reference feature i become available, one can immediately switch the reference to it, and therefore recover the original base and annihilate the bias.

3.3 Complete algorithm

The implementation of an approximate wide-sense nonlinear filter for the model (7) proceeds as follows:

Initialization Choose the initial conditions $\mathbf{y}_0^i = \mathbf{y}^i(0), \quad \rho_0^i = 1, \quad rT_0 = 0, \quad \Omega_0 = 0, \quad V_0 = 0, \quad \omega_0 = 0, \quad \forall\ i = 1 \dots N$. For the initial variance P_0, choose it to be block diagonal with blocks $\Sigma_{n^i}(0)$ corresponding to $\mathbf{y}_0^i$, a large positive number M (typically 100-1000 units of focal length) corresponding to ρ^i, zeros corresponding to T_0 and Ω_0 (fixing the inertial frame to coincide with the initial reference frame). We also choose a large positive number W for the blocks corresponding to V_0 and ω_0.

The variance $\Sigma_n(t)$ is usually available from the analysis of the feature tracking algorithm. We assume that the tracking error is independent in each point, and therefore Σ_n is block diagonal. We choose each block to be the covariance of the measurement $\mathbf{y}^i(t)$ (in the current implementation they are diagonal and equal to 1 pixel std.). The variance $\Sigma_w(t)$ is a design parameter that is available for tuning. We describe the procedure in section 3.4. Finally, set

$$\begin{cases} \hat{\xi}(0|0) \doteq [{\mathbf{y}^4}_0^T, \dots {\mathbf{y}^N}_0^T,\ \rho_0^2, \dots, \rho_0^N,\ T_0^T,\ \Omega_0^T,\ V_0^T,\ \omega_0^T]^T \\ P(0|0) \doteq P_0. \end{cases} \tag{18}$$

[8] Just to give the reader an intuitive feeling of the numbers involved, we find that in practice the average lifetime of a feature is around 10-30 frames depending on illumination and reflectance properties of the scene and motion of the camera. The variance of the estimation error for $\mathbf{y}_0^i$ is in the order of 10^{-6} units of focal length, while the variance of ρ^i is in the order of 10^{-4} units for noise levels commonly encountered with commercial cameras.

Transient During the first transient of the filter, we do not allow for new features to be acquired. Whenever a feature is lost, its state is removed from the model and its best current estimate is placed in a storage vector. If the feature was associated with the scale factor, we proceed as in section 3.2. The transient can be tested as either a threshold on the innovation, a threshold on the variance of the estimates, or a fixed time interval. We choose a combination with the time set to 30 frames, corresponding to one second of video.

The recursion to update the state ξ and the variance P proceed as follows: Let f and h denote the state and measurement model, so that equation (7) can be written in concise form as

$$\begin{cases} \xi(t+1) = f(\xi(t)) + w(t) \quad & w(t) \sim \mathcal{N}(0, \Sigma_w) \\ y(t) = h(\xi(t)) + n(t) \quad & n(t) \sim \mathcal{N}(0, \Sigma_n) \end{cases} \tag{19}$$

We then have

Prediction:

$$\begin{cases} \hat{\xi}(t+1|t) = f(\hat{\xi}(t|t)) \\ P(t+1|t) = F(t)P(t|t)F^T(t) + \Sigma_w \end{cases} \tag{20}$$

Update:

$$\begin{cases} \hat{\xi}(t+1|t+1) = \hat{\xi}(t+1|t) + L(t+1)\left(y(t+1) - h(\hat{\xi}(t+1|t))\right) \\ P(t+1|t+1) = \Gamma(t+1)P(t+1|t)\Gamma^T(t+1) + L(t+1)\Sigma_n(t+1)L^T(t+1). \end{cases} \tag{21}$$

Gain:

$$\begin{cases} \Gamma(t+1) \doteq I - L(t+1)H(t+1) \\ L(t+1) \doteq P(t+1|t)H^T(t+1)\Lambda^{-1}(t+1) \\ \Lambda(t+1) \doteq H(t+1)P(t+1|t)H^T(t+1) + \Sigma_n(t+1) \end{cases} \tag{22}$$

Linearization:

$$\begin{cases} F(t) \doteq \frac{\partial f}{\partial \xi}(\hat{\xi}(t|t)) \\ H(t+1) \doteq \frac{\partial h}{\partial \xi}(\hat{\xi}(t+1|t)) \end{cases} \tag{23}$$

Let $\mathbf{e}_i$ be the i-th canonical vector in $\mathbb{R}^3$ and define $Y^i(t) \doteq e^{\widehat{\Omega}(t)}\, \mathbf{y}_0^i(t)\, \rho^i(t) + T(t)$, $Z^i(t) \doteq \mathbf{e}_3^T Y^i(t)$. The i-th block-row ($i = 1, \dots, N$) $H_i(t)$ of the matrix $H(t)$ can be written as $H_i = \frac{\partial y^i}{\partial Y^i}\frac{\partial Y^i}{\partial \xi} \doteq \Pi_i \frac{\partial Y^i}{\partial \xi}$ where the time argument t has been omitted for simplicity of notation. It is easy to check that $\Pi_i = \frac{1}{Z^i}\left[I_2 \quad -\pi(Y^i) \right]$ and

$$\frac{\partial Y^i}{\partial \xi} = \left[\underbrace{0 \; \dots \; \tfrac{\partial Y^i}{\partial \mathbf{y}_0^i} \; \dots \; 0}_{2N-6} \quad \underbrace{0 \; \dots \; \tfrac{\partial Y^i}{\partial \rho^i} \; \dots \; 0}_{N-1} \quad \underbrace{\frac{\partial Y^i}{\partial T}}_{3} \quad \underbrace{\frac{\partial Y^i}{\partial \Omega}}_{3} \quad \underbrace{0}_{3} \quad \underbrace{0}_{3} \right].$$

The partial derivatives in the previous expression are given by

$$\begin{cases} \frac{\partial Y^i}{\partial \mathbf{y}_0^i} = e^{\widehat{\Omega}} \begin{bmatrix} I_2 \\ 0 \end{bmatrix} \rho^i \\ \frac{\partial Y^i}{\partial \rho^i} = e^{\widehat{\Omega}} \jmath\, \mathbf{y}_0^i \\ \frac{\partial Y^i}{\partial T} = I \\ \frac{\partial Y^i}{\partial \Omega} = \left[\frac{\partial e^{\widehat{\Omega}}}{\partial \Omega_1} \mathbf{y}_0^i \rho^i \quad \frac{\partial e^{\widehat{\Omega}}}{\partial \Omega_2} \mathbf{y}_0^i \rho^i \quad \frac{\partial e^{\widehat{\Omega}}}{\partial \Omega_3} \mathbf{y}_0^i \rho^i \right] \end{cases}$$

The linearization of the state equation involves derivatives of the logarithm function in SO(3) which is available as a Matlab function in the software distribution [14] and will not be reported here. We shall use the following notation:

$$\frac{\partial Log_{SO(3)}(R)}{\partial R} \doteq \left[\frac{\partial Log_{SO(3)}(R)}{\partial r_{11}} \quad \frac{\partial Log_{SO(3)}(R)}{\partial r_{21}} \quad \dots \quad \frac{\partial Log_{SO(3)}(R)}{\partial r_{33}} \right]$$

where r_{ij} is the element in position (i,j) of R. Let us denote $R \doteq e^{\hat{\omega}} e^{\hat{\Omega}}$; the linearization of the state equation can be written in the following form:

$$F \doteq \begin{bmatrix} I_{2N-6} & 0 & 0 & 0 & 0 & 0 \\ 0 & I_{N-1} & 0 & 0 & 0 & 0 \\ 0 & 0 & e^{\hat{\omega}} & 0 & I & \left[\frac{\partial e^{\hat{\omega}}}{\partial \omega_1} T \quad \frac{\partial e^{\hat{\omega}}}{\partial \omega_2} T \quad \frac{\partial e^{\hat{\omega}}}{\partial \omega_3} T \right] \\ 0 & 0 & 0 & \frac{\partial Log_{SO(3)}(R)}{\partial R} \frac{\partial R}{\partial \Omega} & 0 & \frac{\partial Log_{SO(3)}(R)}{\partial R} \frac{\partial R}{\partial \omega} \\ 0 & 0 & 0 & 0 & I & 0 \\ 0 & 0 & 0 & 0 & 0 & I \end{bmatrix}$$

where

$$\frac{\partial R}{\partial \Omega} \doteq \left[\left(e^{\hat{\omega}} \frac{\partial e^{\hat{\Omega}}}{\partial \Omega_1} \right)^{\vee} \quad \left(e^{\hat{\omega}} \frac{\partial e^{\hat{\Omega}}}{\partial \Omega_2} \right)^{\vee} \quad \left(e^{\hat{\omega}} \frac{\partial e^{\hat{\Omega}}}{\partial \Omega_3} \right)^{\vee} \right]$$

and

$$\frac{\partial R}{\partial \omega} \doteq \left[\left(\frac{\partial e^{\hat{\omega}}}{\partial \omega_1} e^{\hat{\Omega}} \right)^{\vee} \quad \left(\frac{\partial e^{\hat{\omega}}}{\partial \omega_2} e^{\hat{\Omega}} \right)^{\vee} \quad \left(\frac{\partial e^{\hat{\omega}}}{\partial \omega_3} e^{\hat{\Omega}} \right)^{\vee} \right]$$

and the bracket $(\cdot)^{\vee}$ indicates that the content has been organized into a column vector.

Regime Whenever a feature disappears, we simply remove it from the state as during the transient. However, after the transient a feature selection module works in parallel with the filter to select new features so as to maintain roughly a constant number (equal to the maximum that the hardware can handle in real time), and to maintain a distribution as uniform as possible across the image plane. We implement this by randomly sampling points on the plane, searching then around that point for a feature with enough brightness gradient (we use an SSD-type test [17]).

Once a new point-feature is found (one with enough contrast along two independent directions), a new filter (which we call a "subfilter") is initialized based on the model (15). Its evolution is given by

Initialization:

$$\begin{cases} \hat{\mathbf{y}}^i_\tau(\tau|\tau) = \mathbf{y}^i_\tau(\tau) \\ \hat{\rho}^i_\tau(\tau|\tau) = 1 \\ P_\tau(\tau|\tau) = \begin{bmatrix} \ddots & & \\ & \Sigma_{n^i}(\tau) & \\ & & \ddots \\ & & & M \end{bmatrix} \end{cases} \tag{24}$$

Prediction:

$$\begin{cases} \hat{\mathbf{y}}^i_\tau(t+1|t) = \hat{\mathbf{y}}^i_\tau(t|t) \\ \hat{\rho}^i_\tau(t+1|t) = \hat{\rho}^i_\tau(t|t) \\ P_\tau(t+1|t) = P_\tau(t+1|t) + \Sigma_w(t) \end{cases} \qquad t > \tau \tag{25}$$

Update:

$$\begin{bmatrix} \hat{\mathbf{y}}^i_\tau(t+1|t+1) \\ \hat{\rho}_\tau(t+1|t+1) \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{y}}^i_\tau(t+1|t) \\ \hat{\rho}_\tau(t+1|t) \end{bmatrix} + L_\tau(t+1) \left(\mathbf{y}^i(t) - \pi(\exp(\widehat{\Omega}(t)) \left[\exp(\widehat{\Omega}(\tau))\right]^{-1} \left[\mathbf{y}^i(t)\rho^i(t) - T(\tau)\right] + T(t)) \right) \tag{26}$$

and P_τ is updated according to a Riccati equation in all similar to (21).

After a probation period, whose length is chosen according to the same criterion adopted for the main filter, the feature is inserted into the state using the transformation (16). The initial variance is chosen to be the variance of the estimation error of the subfilter.

3.4 Tuning

The variance $\Sigma_w(t)$ is a design parameter. We choose it to be block diagonal, with the blocks corresponding to $T(t)$ and $\Omega(t)$ equal to zero (a deterministic integrator). We choose the remaining parameters using standard statistical tests, such as the Cumulative Periodogram of Bartlett [3]. The idea is that the parameters in Σ_w are changed until the innovation process $\epsilon(t) \doteq y(t) - h(\hat{\xi}(t))$ is as close as possible to being white. The periodogram is one of many ways to test the "whiteness" of a stochastic process. In practice, we choose the blocks corresponding to $\mathbf{y}^i_0$ equal to the variance of the measurements, and the elements corresponding to ρ^i all equal to σ_ρ. We then choose the blocks corresponding to V and ω to be diagonal with element σ_v, and then we change σ_v relative to σ_ρ depending on whether we want to allow for more or less regular motions. We then change both, relative to the variance of the measurement noise, depending on the level of desired smoothness in the estimates.

Tuning nonlinear filters is an art, and this is not the proper venue to discuss this issue. Suffices to say that we have only performed the procedure once and for all. We then keep the same tuning parameters no matter what the motion, structure and noise in the measurements.

4 Experiments

The complexity of SFM makes it difficult to demonstrate the performance of an algorithm by means of a few plots. This is what motivated us to (a) obtain analytical results, which are presented in [5], and (b) make our real-time implementation available to the public, so that the performance of the filter can be tested first-hand [14].

In this section, for the sake of exemplification, we present a small sample of the performance of the filter as characterized with a few experiments on our real-time platform.

4.1 Structure error

One of the byproducts of our algorithms is an estimate of the position of a number of point-features in the camera reference frame at the initial time. We use such estimates for a known object in order to characterize the performance of the filter. In particular, the distance between adjacent point on a checkerboard patter (see figure 1) is known to be 2cm. We have run the filter on a sequence of 200 frames and identified adjacent features, and plotted their distance (minus 2cm) in figure 1. It can be seen that the distance, despite an arbitrary initialization, remains well below 1mm.

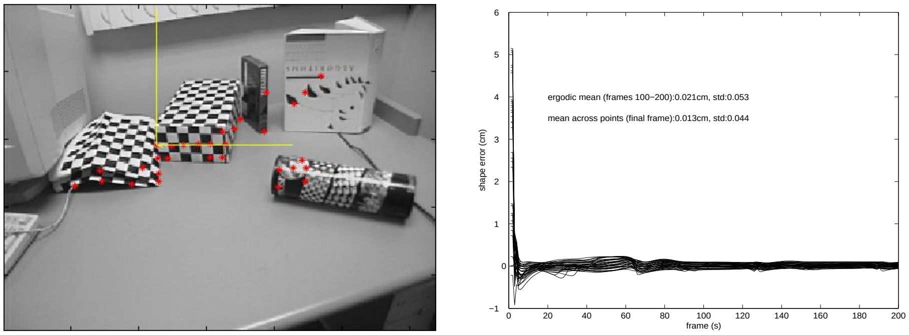

Fig. 1. *(Left)* **A display of the real-time system.** *Selected features are highlighted by asterisks, and a virtual object (a reference frame) is placed in the scene. As the camera moves, the image of the virtual object is modified in real time, according to the estimated motion and structure of the scene, so as to make it appear stationary within the scene. Other displays visualize the motion of the camera relative to an inertial reference frame, and a bird's eye view of the reconstructed position of the points tracked. (Right)* **Structure error:** *the error in mutual distance between a set of 20 points for which the relative position is known (the squares in the checkerboard box on the left) are plotted for a sequence of 200 frames. Mean and standard deviation, both computed across the set of points at the last frame and across the last 100 frames, are below one millimeter. The experiment is performed off-line, and only unoccluded features are considered.*

4.2 Motion error

Errors in motion are difficult to characterize on real sequences of images, for external means of estimating motion (e.g. inertial, magnetic sensors, encoders) are likely to be less accurate than vision. We have therefore placed a checkerboard box on a turntable and moved it for a few seconds, going back to its original position, marked with a accuracy greater than 0.5mm. In figure 2 we show the distance between the estimated position of the camera and the initial position. Again, the error is below 1mm.

Notice that in these experiments we have fixed the scale factor using the fact that the side of a square in the checkerboard is 2cm and we have processed the data off-line, so that only the unoccluded points are used.

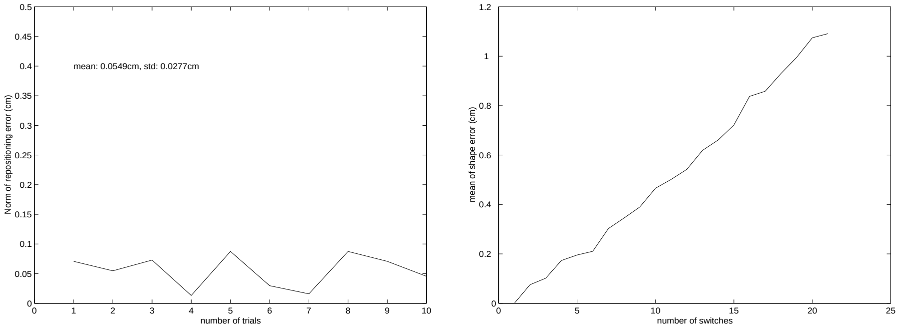

Fig. 2. *(Left)* **Motion error:** *a checkerboard box is rotated on a turntable and then brought back to the initial position 10 times. We plot the distance of the estimated position from the initial time for the 10 trials. The ergodic mean and std are below one millimeter. (Right)* **Scale drift:** *during a sequence of 200 frames, the reference feature was switched 20 times. The mean of the shape error increases drifts away, but at a slow pace, reaching about one centimeter by the end of the sequence*

4.3 Scale drift

In order to quantify the drift that occurs when the reference feature becomes occluded, we have generated a sequence of 200 frames and artificially switched the reference feature every 10 frames. The mean of the structure error is shown in figure 2. Despite being unavoidable, the drift is quite modest, around 1cm after 20 switches.

4.4 Use of the motion estimates for rendering

The estimates of motion obtained using the algorithm we have described can be used in order to obtain estimates of shape. As a simple example, we have taken an uncalibrated sequence of images, shown in figure 3, and estimated its motion and focal length with the model described in section 2.1, while fixing the optical center at the center of the image. We have then used the estimates of motion to perform a dense correlation-based triangulation. The position of some 120,000 points, rendered with shading, is shown in figure 3, along with two views obtained from novel viewpoints.

Although there is no ground truth available, the qualitative shape of the scene seems to have been captured. Sure there are several artifacts. However, we would like to stress that these results have been obtained entirely automatically.

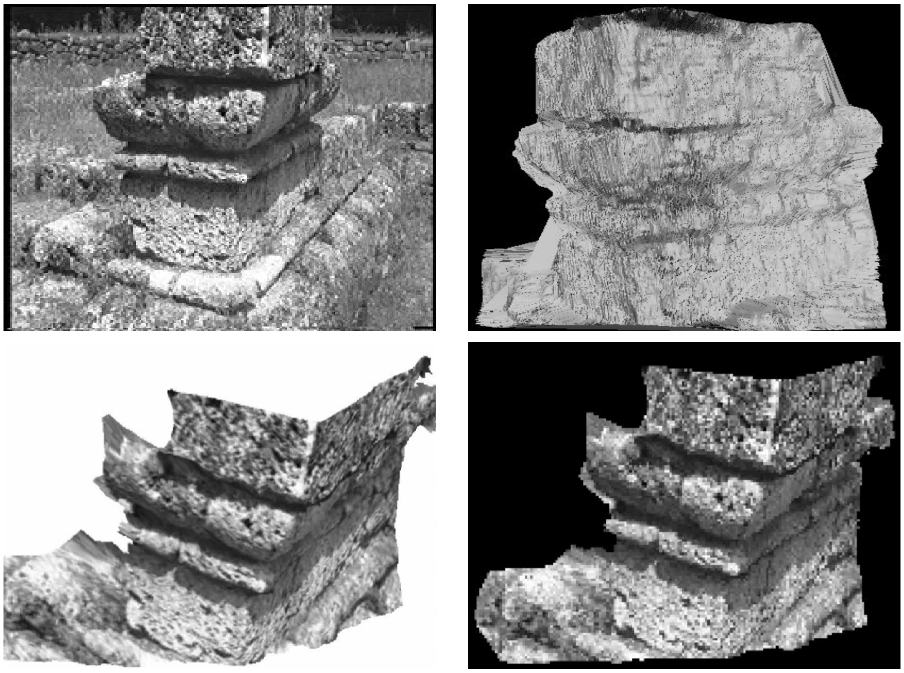

Fig. 3. *The "temple sequence" (courtesy of AIACE): one image out of a sequence of 46 views of an Etruscan temple (top-left): no calibration data is available. The motion estimated using the algorithm presented in this paper can be used to triangulate each pixel, thus obtaining a "dense" representation of the scene. This can be rendered with shading (top-right) or texture-mapped and rendered from an arbitrary viewpoint (bottom left and right). Although no ground truth is available and there are significant artifacts, the qualitative shape can be appreciated from the rendered views.*

5 Conclusions

The causal estimation of three-dimensional structure and motion can be posed as a nonlinear filtering problem. In this paper we have described the implementation of an algorithm whose global observability, uniform observability, minimal realization and stability have been proven in [5].

The filter has been implemented on a personal computer, and the implementation has been made available to the public. The filter exhibits honest performance when the scene contains at least 20-40 points with high contrast, when the relative motion is "slow" (compared to the sampling frequency of the frame grabber), when the scene occupies a significant portion of the image and the lens aperture is "large enough" (typically more than 30^o of visual field).

While it is relatively simple to design an experiment where the implementation fails to provide reliable estimates (changing illumination, specularities etc.), we believe that the algorithm we propose is close to the performance limits for causal, real-time algorithms to recover point-wise structure and motion[9]. In order to improve the performance of motion estimates, we believe that a more "global" representation of the environment is needed. Using feature-points alone, we think this is as good as it gets.

The next logical steps are in two directions. On one hand to explore more meaningful representations of the environment as a collection of surfaces with certain shape emitting a certain energy distribution. On the other hand, a theoretically sound treatment of nonlinear filtering for these problem involves estimation on Riemannian manifolds and homogeneous spaces. Both are open and challenging problems in need of meaningful solutions.

References

1. G. Adiv. Determining three-dimensional motion and structure from optical flow generated by several moving objects. *IEEE Trans. Pattern Anal. Mach. Intell.*, 7(4):348–401, 1985.
2. A. Azarbayejani and A. Pentland. Recursive estimation of motion, structure and focal length. *IEEE Trans. Pattern Anal. Mach. Intell.*, 17(6):562–575, 1995.
3. M.S. Bartlett. *An Introduction to Stochastic Processes.* Cambridge University Press, 1956.
4. T. Broida and R. Chellappa. Estimation of object motion parameters from noisy images. *IEEE Trans. Pattern Anal. Mach. Intell.*, Jan. 1986.
5. A. Chiuso and S. Soatto. 3-d motion and structure causally integrated over time: Theory. In *Tutorial lecture notes of IEEE Intl. Conf. on Robotics and Automation*, April 2000.
6. W. Dayawansa, B. Ghosh, C. Martin, and X. Wang. A necessary and sufficient condition for the perspective observability problem. *Systems and Control Letters*, 25(3):159–166, 1994.
7. E. D. Dickmanns and V. Graefe. Applications of dynamic monocular machine vision. *Machine Vision and Applications*, 1:241–261, 1988.
8. O. Faugeras. *Three dimensional vision, a geometric viewpoint.* MIT Press, 1993.
9. C. Fermüller and Y. Aloimonos. Tracking facilitates 3-d motion estimation. *Biological Cybernetics (67), 259-268*, 1992.
10. D.B. Gennery. Tracking known 3-dimensional object. In *Proc. AAAI 2nd Natl. Conf. Artif. Intell.*, pages 13–17, Pittsburg, PA, 1982.
11. J. Heel. Dynamic motion vision. *Robotics and Autonomous Systems*, 6(1), 1990.
12. X. Hu and N. Ahuja. Motion and structure estimation using long sequence motion models. *Image and Vision Computing*, 11(9):549–569, 1993.
13. A. Jepson and D. Heeger. Subspace methods for recovering rigid motion ii: theory. RBCV TR-90-35, University of Toronto – CS dept., November 1990. Revised July 1991.
14. H. Jin, P. Favaro, and S. Soatto. Real-time 3-d motion and structure from point features: a front-end system for vision-based control and interaction. In *Computer Vision and Pattern Recognition; code available from* `http://ee.wustl.edu/~soatto/research/`, June 2000.

[9] Of course we have no proof of this claim, and even comparisons with theoretical lower bounds are meaningless in this context, since the conditional density of the state is unknown and cannot be computed with a finite-dimensional algorithm, as explained in [5].

15. J. J. Koenderink and A. J. Van Doorn. Affine structure from motion. *J. Optic. Soc. Am.*, 8(2):377–385, 1991.
16. R. Kumar, P. Anandan, and K. Hanna. Shape recovery from multiple views: a parallax based approach. *Proc. of the Image Understanding Workshop*, 1994.
17. B.D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. *Proc. 7th Int. Joinnt Conf. on Art. Intell.*, 1981.
18. L. Matthies, R. Szelisky, and T. Kanade. Kalman filter-based algorithms for estimating depth from image sequences. *Int. J. of computer vision*, pages 2989–2994, 1989.
19. P. McLauchlan, I. Reid, and D. Murray. Recursive affine structure and motion from image sequences. *Proc. of the 3rd Eur. Conf. Comp. Vision*, Stockholm, May 1994.
20. P. F. McLauchlan. Gauge invariance in projective 3d reconstruction. In *IEEE Workshop on Multi-View Modeling and Analysis of Visual Scenes, Fort Collins, CO, June 1999*, 1999.
21. J. Oliensis. Provably correct algorithms for multi-frame structure from motion. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1996.
22. J. Oliensis and J. Inigo-Thomas. Recursive multi-frame structure from motion incorporating motion error. *Proc. DARPA Image Understanding Workshop*, 1992.
23. J. Philip. Estimation of three dimensional motion of rigid objects from noisy observations. *IEEE Trans. Pattern Anal. Mach. Intell.*, 13(1):61–66, 1991.
24. C. Poelman and T. Kanade. A paraperspective factorization method for shape and motion recovery. *Proc. of the 3 ECCV, LNCS Vol 810, Springer Verlag*, 1994.
25. H. S. Sawhney. Simplifying motion and structure analysis using planar parallax and image warping. *Proc. of the Int. Conf. on Pattern Recognition*, Seattle, June 1994.
26. L. Shapiro, A. Zisserman, and M. Brady. Motion from point matches using affine epipolar geometry. *Proc. of the ECCV94, Vol. 800 of LNCS, Springer Verlag*, 1994.
27. S. Soatto. Observability/identifiability of rigid motion under perspective projection. In *Proc. of the 33rd IEEE Conf. on Decision and Control*, pages 3235–3240, Dec. 1994.
28. S. Soatto. 3-d structure from visual motion: modeling, representation and observability. *Automatica*, 33:1287–1312, 1997.
29. S. Soatto and P. Perona. Reducing "structure from motion": a general framework for dynamic vision. part 1: modeling. *IEEE Trans. Pattern Anal. Mach. Intell.*, 20(9):993–942, September 1998.
30. M. Spetsakis and J. Aloimonos. A multi-frame approach to visual motion perception. *Int. J. Computer Vision 6 (3)*, 1991.
31. R. Szeliski. Recovering 3d shape and motion from image streams using nonlinear least squares. *J. visual communication and image representation*, 1994.
32. M. A. Taalebinezhaad. Direct recovery of motion and shape in the general case by fixation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 14(8):847–853, 1992.
33. C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: a factorization method. *Int. J. of Computer Vision*, 9(2):137–154, 1992.
34. J. Weng, N. Ahuja, and T. Huang. Optimal motion and structure estimation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 15:864–884, 1993.
35. Z. Zhang and O. D. Faugeras. Three dimensional motion computation and object segmentation in a long sequence of stereo frames. *Int. J. of Computer Vision*, 7(3):211–241, 1992.