Essential Matrix Estimation Using Adaptive Penalty Formulations

Mohammed E. Fathy
mefathy@cs.umd.edu
Michael C. Rotkowitz
mcrotk@umd.edu
University of Maryland, College Park
Maryland, USA

Abstract

Given six or more pairs of corresponding points on two calibrated images, existing schemes for estimating the essential matrix (EsM) use some manifold representation to tackle the non-convex constraints of the problem. To the best of our knowledge, no attempts were made to use the more straightforward approach of integrating the EsM constraint functions directly into the optimization using Adaptive Penalty Formulations (APFs). One possible reason is that the constraints characterizing the EsM are nonlinearly dependent and their number exceeds the number of free parameters in the optimization variable.

This paper presents an iterative optimization scheme based on penalty methods that integrates the EsM constraints into the optimization without the use of manifold-based techniques and differential geometry tools. The scheme can be used with algebraic, geometric, and/or robust cost functions. Experimental validations using synthetic and real data show that the proposed scheme outperforms manifold-based algorithms with either global or local parametrizations.

1 Introduction

The estimation of the essential matrix (EsM) plays a central role in structure-from-motion algorithms for datasets of all sizes starting from small (e.g. two images) [11] to large (e.g. thousands of images) [1] ones. Given a set S of more than five noisy (but outlier-free) correspondences, several algorithms have been proposed to compute the 3×3 EsM that best fits S. These algorithms can be categorized as non-iterative or iterative. The eight-point [7, 11] (when $|S| \geq 8$) and the overdetermined five-point [14, 18] schemes are examples of the non-iterative schemes for EsM estimation. Despite their efficiency and widely available implementations, most of the time they are used to initialize more complicated schemes to improve the accuracy of the estimates.

Iterative schemes for EsM estimation generally lead to more accurate results. While the minimization of the nonlinear geometric cost functions is one reason these schemes work generally better [12], the other reason is that they take into account the special structure of the manifold $\mathcal{E}$ of valid EsMs through different ways with varying degrees of success. The simplest way is to project the current estimate $\mathbf{E}^k$ onto $\mathcal{E}$ after each iteration [17]. Another popular way is to use a global parametrization of $\mathcal{E}$ [7, 17, 19]. While computationally simple, such global parametrizations can have degeneracies and convergence issues since they

ignore the fact the EsM manifold is only locally diffeomorphic to $\mathbb{R}^5$ [12]. Ma et al. [12] proposed a Newton algorithm for the intrinsic optimization over $\mathcal{E}$. Helmke et al. [8] identified convergence issues with the algorithm in [12] and proposed a hybrid Newton/Gauss-Newton algorithm for the intrinsic minimization over $\mathcal{E}$ in addition to an alternative set of simpler parametrizations that may be used interchangeably with the one proposed in [12].

This paper presents an iterative scheme for estimating EsMs that can minimize algebraic as well as geometric cost functions. The proposed scheme integrates the EsM constraints into the optimization algorithm using penalty methods [2], which are conceptually simpler to understand than Riemannian manifold-based algorithms (such as [12] and [8]). Our experiments show that the proposed APF algorithm outperforms Helmke's local manifold scheme [8], in terms of accuracy, and the Levenberg-Marquardt-based global manifold scheme [7, 12, 19] in terms of both accuracy and speed.

Section 2 of this paper reviews the relevant mathematical background while Section 3 describes and provides analytical derivatives for the cost and constraint functions necessary for our work. We present our APF-based scheme in Section 4. The proposed scheme is evaluated and the obtained results are discussed in Section 5. Concluding remarks and future work are described in Section 6.

2 Mathematical Background

Here we list the basic properties of epipolar geometry and the EsM necessary for understanding the proposed schemes. Detailed derivations of these basic properties can be found in [5, 7, 13].

Notation The transpose of a matrix $\mathbf{E}$ is denoted by $\mathbf{E}'$. If $\boldsymbol{t} \in \mathbb{R}^3$, we use $[\boldsymbol{t}]_\times$ to denote the corresponding skew-symmetric matrix which permits expressing the cross-product as a matrix-vector product, i.e. $\boldsymbol{t} \times \boldsymbol{x} = [\boldsymbol{t}]_\times \boldsymbol{x}\ \forall \boldsymbol{t},\ \boldsymbol{x} \in \mathbb{R}^3$. We will occasionally use the 9-vector $\boldsymbol{e}^k$ to refer to $\mathrm{vec}\{\mathbf{E}^k\}$, the column-wise vectorization of the kth estimate of $\mathbf{E}$. Similarly, $\mathrm{mat}\{\boldsymbol{e}^k\}$ gives the 3×3 matrix $\mathbf{E}^k$ corresponding to $\boldsymbol{e}^k$. If $\boldsymbol{x} \in \mathbb{R}^m$, we define the homogeneous form $\tilde{\boldsymbol{x}} \in \mathbb{R}^{m+1}$ as:

$$\tilde{\boldsymbol{x}} = \begin{pmatrix} \boldsymbol{x} \\ 1 \end{pmatrix}. \tag{1}$$

The subscript l (resp. r) in $\tilde{\boldsymbol{p}}_l$ (resp. $\tilde{\boldsymbol{p}}_r$) indicates that the vector should appear at the left (resp. right) side of a quadratic form (e.g. $\tilde{\boldsymbol{p}}_l' \mathbf{E} \tilde{\boldsymbol{p}}_r$).

The Epipolar Constraint Suppose we have two calibrated pinhole cameras observing a scene at the same time from two different locations. The epipolar constraint expresses the fact that the perspective projections $\boldsymbol{p}_l$ and $\boldsymbol{p}_r$ of any 3D point $\boldsymbol{P}$ onto the image planes of camera 1 and camera 2 must satisfy the following algebraic relationship [5, 7]:

$$\tilde{\boldsymbol{p}}_l' [\boldsymbol{t}]_\times \mathbf{R} \tilde{\boldsymbol{p}}_r = \tilde{\boldsymbol{p}}_l' \mathbf{E} \tilde{\boldsymbol{p}}_r = 0. \tag{2}$$

where $\boldsymbol{t}$ is the relative translation vector of camera 2, $\mathbf{R}$ is the 3×3 relative orientation of camera 2, and $\mathbf{E} = [\boldsymbol{t}]_\times \mathbf{R}$ is the 3×3 essential matrix (EsM). The pair of related projections $(\boldsymbol{p}_l, \boldsymbol{p}_r)$ is referred to as a correspondence or match. If we know a set of $n \geq 5$ (calibrated) matches between two cameras with unknown poses, we can use these n correspondences to obtain an estimate of $\mathbf{E}$ from which (and other conditions [14]) the relative pose can be recovered (except for the scale of $\boldsymbol{t}$).

Essential Matrix Characterization A 3x3 matrix $\mathbf{E}$ is an EsM if and only if it can be written as the product $[\boldsymbol{t}]_\times \mathbf{R}$ for some non-zero vector $\boldsymbol{t} \in \mathbb{R}^3$ and a 3x3 rotation matrix

$\mathbf{R}$ [5, 7]. We hereafter use $\mathcal{E} \subset \mathbb{R}^{3\times3}$ to refer to the set/manifold of all EsMs. Due to the homogeneity of the epipolar constraint (2), an EsM has only five degrees of freedom (dofs) rather than six dofs: three dofs for $\mathbf{R}$ and only two dofs for the orientation of $\boldsymbol{t}$.

Equivalently, $\mathbf{E}$ is an EsM if and only if it has one singular value being zero and the other two being equal [5, 7, 8]. In other words, the singular value decomposition (SVD) of the normalized version of $\mathbf{E} = \mathbf{U}\mathbf{D_0}\mathbf{V}'$ where $\mathbf{D}_0 = \sqrt{0.5}\,\mathrm{diag}(1,1,0)$. This characterization provides an easy way to obtain an EsM $\mathbf{E}$ from any 3x3 matrix $\mathbf{B}$ that may not be an EsM. Basically, one computes the SVD of $\mathbf{B} = \mathbf{U_B}\mathbf{D_B}\mathbf{V}'_\mathbf{B}$ and then sets $\mathbf{E} = \mathbf{U_B}\mathbf{D_0}\mathbf{V}'_\mathbf{B}$. We refer to this process as the SVD-correction. It yields from $\mathcal{E}$ the closest EsM to $\mathbf{B}$ (under the Frobenius norm) [8]. This is the standard method to correct estimates of $\mathbf{E}$ obtained by schemes that either do not enforce the EsM constraints (e.g. the eight-point scheme [7, 11]) or need to compensate for round-off errors (e.g. the five-point algorithm [6, 14, 18]).

A third equivalent characterization for a *non-zero* matrix $\mathbf{E}$ to be an EsM is given by the following 3×3 matrix equation [6, 14, 18]:

$$\mathbf{E}'\mathbf{E}\mathbf{E}' = 0.5\,\mathrm{tr}(\mathbf{E}'\mathbf{E})\mathbf{E}'. \tag{3}$$

Intuitively, this equation implies that every row (or column) of $\mathbf{E}$ is a singular vector of $\mathbf{E}$ with singular value $s = \sqrt{0.5}\,\mathrm{tr}(\mathbf{E}'\mathbf{E})$. It follows that exactly two rows (or columns) of $\mathbf{E}$ are independent (and so $\mathbf{E}$ must be singular). In addition, the two non-zero singular values of $\mathbf{E}$ are equal to s. Eq. (3) provides a redundant set of nine cubic polynomial equations where the coupling structure is nonlinear. We hereafter refer to this matrix equation as the matrix constraint (MC). Along with the homogeneous property, we use the MC to define the constraints for estimating the EsM using the penalty-based scheme described in this paper. The MC, homogeneity and the zero-determinant property have been used in most of the popular five-point EsM estimation schemes [6, 14, 18].

For the penalty-based scheme proposed later in Section 4 and for our experiments, we use $d_\mathcal{E}(\mathbf{E})$ to denote the Frobenius distance of the normalized version of the non-zero 3×3 matrix $\mathbf{E}$ and the closest EsM $\hat{\mathbf{E}} \in \mathcal{E}$ (assuming the SVD of $\mathbf{E} = \mathbf{U}\mathbf{D}\mathbf{V}'$) [7, 8]:

$$d_\mathcal{E}(\mathbf{E}) \quad = \quad \left|\left| \frac{1}{||\mathbf{D}||_F}\mathbf{D} - \mathbf{D}_0 \right|\right|_F . \tag{4}$$

Error Criteria and Sum-of-Squares Cost Functions If a correspondence $(\tilde{\boldsymbol{p}}_l, \tilde{\boldsymbol{p}}_r)$ violates the epipolar constraint (2), the algebraic distance (error) $d_r(\boldsymbol{p}_l, \boldsymbol{p}_r) = \tilde{\boldsymbol{p}}'_l\mathbf{E}\tilde{\boldsymbol{p}}_r$ is the simplest way to quantify the violation even though it has been shown to be biased [5, 7]. It can also be written in the dot-product form:

$$d_r(\boldsymbol{p}_l, \boldsymbol{p}_r) = \tilde{\boldsymbol{p}}'_l\mathbf{E}\tilde{\boldsymbol{p}}_r = (\tilde{\boldsymbol{p}}_l \otimes \tilde{\boldsymbol{p}}_r)'\boldsymbol{e}. \tag{5}$$

where $(\tilde{\boldsymbol{p}}_l \otimes \tilde{\boldsymbol{p}}_r)' = \begin{bmatrix} x_l\tilde{\boldsymbol{p}}'_r & y_l\tilde{\boldsymbol{p}}'_r & \tilde{\boldsymbol{p}}'_r \end{bmatrix}$ is the transpose of the Kronecker product of $\tilde{\boldsymbol{p}}_l$ and $\tilde{\boldsymbol{p}}_r$.

Assuming the noise in each coordinate of $(\boldsymbol{p}_l, \boldsymbol{p}_r)$ is IID zero-mean Gaussian, the maximum likelihood (ML) estimate $(\hat{\boldsymbol{p}}_l, \hat{\boldsymbol{p}}_r)$ of the true correspondence is the minimizer of $||\hat{\boldsymbol{p}}_l - \boldsymbol{p}_l||^2 + ||\hat{\boldsymbol{p}}_r - \boldsymbol{p}_r||^2$ subject to $\tilde{\hat{\boldsymbol{p}}}'_l\mathbf{E}\tilde{\hat{\boldsymbol{p}}}_r = 0$ [20]. The reprojection error d_E measures the distance from $(\boldsymbol{p}_l, \boldsymbol{p}_r)$ to its ML estimate and is well approximated by its first-order approximation d_s known as the Sampson distance and which has the following closed form [4, 20]

$$d_s(\boldsymbol{p}_l, \boldsymbol{p}_r) = \frac{d_r(\boldsymbol{p}_l, \boldsymbol{p}_r)}{g(\mathbf{E})} = \frac{\tilde{\boldsymbol{p}}'_l\mathbf{E}\tilde{\boldsymbol{p}}_r}{\sqrt{||\mathbf{D}_1\mathbf{E}\tilde{\boldsymbol{p}}_r||^2 + ||\mathbf{D}_1\mathbf{E}'\tilde{\boldsymbol{p}}_l||^2}}. \tag{6}$$

where $\mathbf{D}_1 = \text{diag}(1,1,0)$.

Derivatives of d_r and d_s with respect to $\boldsymbol{e} = \text{vec}\{\mathbf{E}\}$ are used in this paper. Since d_r is linear, the derivative $\nabla d_r = \nabla_{\boldsymbol{e}} d_r \in \mathbb{R}^9$ is easily found to be independent of $\boldsymbol{e}$:

$$\nabla d_r(\boldsymbol{e}) = \tilde{\boldsymbol{p}}_l \otimes \tilde{\boldsymbol{p}}_r. \tag{7}$$

The derivative $\nabla d_s \in \mathbb{R}^9$ is derived, using the rules in [15], as

$$\nabla d_s(\boldsymbol{e}) = \text{vec}\left\{\nabla_{\mathbf{E}} d_s(\boldsymbol{E})\right\} = \text{vec}\left\{\frac{1}{g(\mathbf{E})}\left[\tilde{\boldsymbol{p}}_l\tilde{\boldsymbol{p}}_r' - \frac{d_s(\mathbf{E})}{g(\mathbf{E})}\left(\mathbf{D}_1\mathbf{E}\tilde{\boldsymbol{p}}_r\tilde{\boldsymbol{p}}_r' + \tilde{\boldsymbol{p}}_l\tilde{\boldsymbol{p}}_l'\mathbf{E}\mathbf{D}_1\right)\right]\right\}. \tag{8}$$

3 Essential Matrix Estimation Problem

Given $n \geq 6$ calibrated, noisy (but outlier-free) correspondences $\{\boldsymbol{p}_l^i, \boldsymbol{p}_r^i\}_{i=1}^n$, we wish to estimate an EsM that adheres as much as possible to the observed data. To this end, we solve a least-squares optimization problem of the following form:

$$\underset{\boldsymbol{e}}{\text{argmin}} \qquad f(\boldsymbol{e}) = 0.5\sum_{i=1}^{n} d_i^2(\boldsymbol{e}) = 0.5\,||\boldsymbol{d}(\boldsymbol{e})||^2, \tag{9}$$

$$\text{subject to} \qquad \boldsymbol{h}(\boldsymbol{e}) = \mathbf{0}. \tag{10}$$

where $\boldsymbol{d}$ is a vector in $\mathbb{R}^n$ stacking the individual d_i's. The residual d_i can either be $d_{r_i} = d_r(\boldsymbol{p}_l^i, \boldsymbol{p}_r^i)$ or $d_{s_i} = d_s(\boldsymbol{p}_l^i, \boldsymbol{p}_r^i)$. The equality constraint function $\boldsymbol{h} : \mathbb{R}^9 \to \mathbb{R}^p$ (where p is the number of constraints) is defined such that $\boldsymbol{h}(\boldsymbol{e}) = \mathbf{0}$ iff $\mathbf{E} = \text{mat}\{\boldsymbol{e}\} \in \mathcal{E}$. Eq. (9, 10) is the main optimization problem solved in this paper.

Algebraic Residual When the residual $d = d_r$, we refer to f as the algebraic cost function C_r. In this case, $f = C_r$ is quadratic (and convex) in $\boldsymbol{e}$. However, the constrained optimization problem is still not convex because the equality constraint function $\boldsymbol{h} : \mathbb{R}^9 \to \mathbb{R}^p$ is not linear. The gradient $\nabla C_r(\boldsymbol{e})$ and the Hessian $\mathbf{H}_f(\boldsymbol{e}_k) = \mathbf{H}_{C_r}(\boldsymbol{e}_k)$ are given as:

$$\nabla C_r(\boldsymbol{e}) = \sum_{i=1}^{n}(\nabla d_{r_i}(\boldsymbol{e}))d_{r_i}(\boldsymbol{e}) = (\sum_{i=1}^{n}(\tilde{\boldsymbol{p}}_l^i \otimes \tilde{\boldsymbol{p}}_r^i)(\tilde{\boldsymbol{p}}_l^i \otimes \tilde{\boldsymbol{p}}_r^i)')\boldsymbol{e} = \mathbf{A}\boldsymbol{e}, \quad \mathbf{H}_{C_r}(\boldsymbol{e}_k) = \mathbf{A}. \tag{11}$$

where $\mathbf{A} = \sum_{i=1}^{n}(\tilde{\boldsymbol{p}}_l^i \otimes \tilde{\boldsymbol{p}}_r^i)(\tilde{\boldsymbol{p}}_l^i \otimes \tilde{\boldsymbol{p}}_r^i)'$ is a 9 x 9 symmetric positive (semi-)definite matrix commonly known as the moment matrix. If C_r is used in an iterative estimation algorithm, the Hessian can be pre-computed once and used in all iterations [3, 7]. The gradient can be efficiently computed each iteration in constant time independent of n.

Geometric Residual When the residual $d = d_s$, we refer to f as the geometric (Sampson) cost function C_s. Unlike C_r, C_s is neither quadratic nor convex in $\boldsymbol{e}$. Assuming that we know a guess $\boldsymbol{e}^k$ and we wish to find the best update $\boldsymbol{\delta}^k \in \mathbb{R}^9$, we convexify C_s by replacing $d_s(\boldsymbol{e}^k + \boldsymbol{\delta}^k)$ by the first-order Taylor approximation at $\boldsymbol{e}^k$ given by $d_s(\boldsymbol{e}^k) + \nabla' d_s(\boldsymbol{e}^k)\boldsymbol{\delta}^k$. With this approximation, the gradient and approximate Hessian at $\boldsymbol{e}^k$ are given by:

$$\nabla f(\boldsymbol{e}^k) = \nabla C_s(\boldsymbol{e}^k) = \sum_{i=1}^{n} d_{s_i}(\boldsymbol{e}^k)\nabla d_{s_i}(\boldsymbol{e}^k), \quad \mathbf{H}_f(\boldsymbol{e}^k) = \sum_{i=1}^{n} \nabla d_{s_i}(\boldsymbol{e}^k)\nabla' d_{s_i}(\boldsymbol{e}^k). \tag{12}$$

Form of The Constraint Function $\boldsymbol{h}$ The constraint function $\boldsymbol{h} : \mathbb{R}^9 \to \mathbb{R}^{10}$ used in this paper consists of two parts. The first part consists of a scalar function that ensures that the

EsM is non-zero. For example, $\boldsymbol{h}_1(\boldsymbol{e}) = ||\boldsymbol{e}||^2 - 1$ ensures that the EsM has a norm equal to 1. We use a slightly different way of imposing the non-zero property in our APF scheme. We defer the description of this alternative way till Section 4. For now, the derivative of $\boldsymbol{h}_1$ at $\boldsymbol{e}^k$ is given by

$$\nabla \boldsymbol{h}_1(\boldsymbol{e}^k) = 2\boldsymbol{e}^k. \tag{13}$$

The second part of $\boldsymbol{h}$ consists of the nine cubic functions derived from (3):

$$\boldsymbol{h}_2(\boldsymbol{e}) = \text{vec}\left\{\mathbf{E}'\mathbf{E}\mathbf{E}' - 0.5\,\text{tr}(\mathbf{E}'\mathbf{E})\mathbf{E}'\right\} \tag{14}$$

This ensures that $\mathbf{E}$ is singular with the two other singular values equivalent. The derivative $\frac{\partial \boldsymbol{h}_2}{\partial \mathbf{E}_{ij}}$ at $\boldsymbol{e}^k$ is given by

$$\frac{\partial \boldsymbol{h}_2}{\partial \mathbf{E}_{ij}} = \text{vec}\left\{\mathbf{I}^{ji}\mathbf{E}^k\mathbf{E}^{k\prime} + \mathbf{E}^{k\prime}\mathbf{I}^{ij}\mathbf{E}^{k\prime} + \mathbf{E}^{k\prime}\mathbf{E}^k\mathbf{I}^{ji} - \mathbf{E}^k_{ij}\mathbf{E}^k - 0.5\,\text{tr}(\mathbf{E}^{k\prime}\mathbf{E}^k)\mathbf{I}^{ji}\right\}. \tag{15}$$

where $\mathbf{I}^{ij}$ is the 3×3 matrix that has all zeros except a value of 1 at the entry (i,j), for $i,j \in \{1,2,3\}$. The 10×9 Jacobian $\mathbf{J}^k_{\boldsymbol{h}} = \mathbf{J}_{\boldsymbol{h}}(\boldsymbol{e}^k)$ of the complete function $\boldsymbol{h}$ at $\boldsymbol{e}^k$ is given by

$$\mathbf{J}^k_{\boldsymbol{h}} = \begin{bmatrix} \nabla' \boldsymbol{h}^k_1 \\ \mathbf{J}^k_2 \end{bmatrix} = \begin{bmatrix} & & & & 2\boldsymbol{e}^{k\prime} & & & & \\ \frac{\partial \boldsymbol{h}_2}{\partial \mathbf{E}_{11}} & \frac{\partial \boldsymbol{h}_2}{\partial \mathbf{E}_{21}} & \frac{\partial \boldsymbol{h}_2}{\partial \mathbf{E}_{31}} & \frac{\partial \boldsymbol{h}_2}{\partial \mathbf{E}_{12}} & \frac{\partial \boldsymbol{h}_2}{\partial \mathbf{E}_{22}} & \frac{\partial \boldsymbol{h}_2}{\partial \mathbf{E}_{32}} & \frac{\partial \boldsymbol{h}_2}{\partial \mathbf{E}_{13}} & \frac{\partial \boldsymbol{h}_2}{\partial \mathbf{E}_{23}} & \frac{\partial \boldsymbol{h}_2}{\partial \mathbf{E}_{33}} \end{bmatrix}. \tag{16}$$

4 Adaptive Penalty Formulations (APFs)

A penalty formulation provides an intuitive way of enforcing constraints during optimization [2]. The idea is to relax the constraints of the problem while making violating them expensive. This is done by augmenting the cost function $f(\boldsymbol{e})$ with a penalty term $q(\boldsymbol{e})$ that incurs very high cost when $\boldsymbol{h}(\boldsymbol{e}) \neq \mathbf{0}$ and evaluates to 0 when $\boldsymbol{h}(\boldsymbol{e}) = \mathbf{0}$. There are different choices for the penalty function q [2]. In this work, we use the quadratic penalty $q_c(\boldsymbol{e}) = 0.5c\,||\boldsymbol{h}(\boldsymbol{e})||^2$ (with $c > 0$) which is popular in nonlinear programming, computer vision and machine learning [2, 10, 16]. If we let $f_c(\boldsymbol{e}) = f(\boldsymbol{e}) + q_c(\boldsymbol{e})$ denote the APF cost function, we see that $f_c(\boldsymbol{e}) = f(\boldsymbol{e})\ \ \forall \boldsymbol{e} \in \mathcal{E}$ since $q_c(\boldsymbol{e}) = 0 \quad \forall \boldsymbol{e} \in \mathcal{E}$. In addition, if we let $\boldsymbol{e}^*_c = \text{argmin}_{\boldsymbol{e}\in\mathbb{R}^9} f_c(\boldsymbol{e})$ and $\boldsymbol{e}^* = \text{argmin}_{\boldsymbol{e}\in\mathcal{E}} f(\boldsymbol{e})$, we can see that

$$f_c(\boldsymbol{e}^*_c) \leq f_c(\boldsymbol{e}^*) \quad = f(\boldsymbol{e}^*) + q_c(\boldsymbol{e}^*) \quad = f(\boldsymbol{e}^*). \tag{17}$$

and that $f(\boldsymbol{e}^*_c) \leq f(\boldsymbol{e}^*_c) + q_c(\boldsymbol{e}^*_c) = f_c(\boldsymbol{e}^*_c) \leq f(\boldsymbol{e}^*)$. In other words, the minimizer $\boldsymbol{e}^*_c$ of the penalty-augmented function f_c has a lower (or equal) value of the original cost f than the minimizer of f subject to $\boldsymbol{e} \in \mathcal{E}$. Theoretically, we can make the optimal sets of the PF and the original EsM function f equivalent by choosing c close to ∞ which makes $q(\boldsymbol{e}) = \infty\ \forall \boldsymbol{e} \notin \mathcal{E}$ and in which case $f_c(\boldsymbol{e}_g) < f_c(\boldsymbol{e}_b) = \infty\ \forall \boldsymbol{e}_g \in \mathcal{E}, \boldsymbol{e}_b \notin \mathcal{E}$. In practice, we repeatedly compute the minimum $\boldsymbol{e}^*_c$ of f_c for a gradually increasing sequence $\{c_k\}$ to avoid numerical conditioning problems [2].

We use the quadratic penalty method in our scheme to enforce the MC constraint $\boldsymbol{h}_2(\boldsymbol{e}) = \mathbf{0}$. We keep $\boldsymbol{e}$ away from zero by requiring the update $\boldsymbol{\delta}^k$ to satisfy $< \boldsymbol{e}^k, \boldsymbol{\delta}^k >= 0$. Assuming $\boldsymbol{e}^k \neq \mathbf{0}$, it is easy to see that $||\boldsymbol{e}^{k+1}||^2 = ||\boldsymbol{e}^k + \boldsymbol{\delta}^k||^2 = ||\boldsymbol{e}^k||^2 + ||\boldsymbol{\delta}^k||^2 \geq ||\boldsymbol{e}^k||^2 > 0$. This strategy results in an update $\boldsymbol{\delta}^k$ that is orthogonal to $\boldsymbol{e}^k$ or, equivalently, $\boldsymbol{\delta}^k$ has a zero radial component along $\boldsymbol{e}^k$. This is desirable in practice because the exact scale of $\boldsymbol{e}^k$ is insignificant

(as long as it is non-zero) and radial updates along $\boldsymbol{e}^k$ merely adjust its scale rather than improving the solution.

The penalty formulation of the problem is given by

$$\underset{\boldsymbol{\delta}^k \in \mathbb{R}^9}{\operatorname{argmin}} f_{c_k}(\boldsymbol{e}^k + \boldsymbol{\delta}^k) = f(\boldsymbol{e}^k + \boldsymbol{\delta}^k) + 0.5 c_k ||\boldsymbol{h}_2(\boldsymbol{e}^k + \boldsymbol{\delta}^k)||^2, \text{ subject to } \boldsymbol{e}^{k\prime}\boldsymbol{\delta}^k = 0. \tag{18}$$

where $f : \mathbb{R}^9 \to \mathbb{R}$ is the cost function.

We then build a convex quadratic program (QP) approximation to the above problem by (a) replacing f with a convex second-order Taylor approximation $0.5\boldsymbol{\delta}^{k\prime}\mathbf{H}_f(\boldsymbol{e}^k)\boldsymbol{\delta}^k + \boldsymbol{\nabla}' f(\boldsymbol{e}^k)\boldsymbol{\delta}^k + f(\boldsymbol{e}^k)$ and (b) replacing $\boldsymbol{h}_2(\boldsymbol{e}^k + \boldsymbol{\delta}^k)$ with a linear Taylor approximation $\boldsymbol{h}_2^k + \mathbf{J}_2^k\boldsymbol{\delta}^k$ where $\boldsymbol{h}_2^k = \boldsymbol{h}_2(\boldsymbol{e}^k)$. The resulting QP is given by:

$$\underset{\boldsymbol{\delta}^k \in \mathbb{R}^9}{\operatorname{argmin}} \quad \hat{f}_{c_k}(\boldsymbol{e}^k + \boldsymbol{\delta}^k) = \frac{1}{2}\boldsymbol{\delta}^{k\prime}(\mathbf{H}_f^k + c_k\mathbf{J}_2^{k\prime}\mathbf{J}_2^k)\boldsymbol{\delta}^k + (\boldsymbol{\nabla} f^k + c_k\mathbf{J}_2^{k\prime}\boldsymbol{h}_2^k)'\boldsymbol{\delta}^k + \text{const}, \tag{19}$$

$$\text{subject to} \quad \boldsymbol{e}^{k\prime}\boldsymbol{\delta}^k = 0. \tag{20}$$

where $\mathbf{H}_f^k = \mathbf{H}_f(\boldsymbol{e}^k)$ and $\boldsymbol{\nabla} f^k = \boldsymbol{\nabla} f(\boldsymbol{e}^k)$. Both $\boldsymbol{\nabla} f^k$ and $\mathbf{H}_f^k$ can be computed from (11) when $f = C_r$ or from (12) when $f = C_s$. Introducing a scalar Lagrange multiplier v allows us to write the corresponding Lagrangian as:

$$L(\boldsymbol{\delta}^k, v) = \frac{1}{2}\boldsymbol{\delta}^{k\prime}(\mathbf{H}_f^k + c_k\mathbf{J}_2^{k\prime}\mathbf{J}_2^k)\boldsymbol{\delta}^k + (\boldsymbol{\nabla} f^k + c_k\mathbf{J}_2^{k\prime}\boldsymbol{h}_2^k)'\boldsymbol{\delta}^k + v\boldsymbol{e}^{k\prime}\boldsymbol{\delta}^k + \text{const}. \tag{21}$$

The partial derivatives $\boldsymbol{\nabla}_{\boldsymbol{\delta}^k} L$ and $\boldsymbol{\nabla}_v L$ must be zero at the optimal $(\boldsymbol{\delta}^k, v)$ [2]. This gives rise to the following 10×10 symmetric linear system of equations:

$$\begin{bmatrix} \mathbf{H}_f^k + c_k\mathbf{J}_2^{k\prime}\mathbf{J}_2^k & \boldsymbol{e}^k \\ \boldsymbol{e}^{k\prime} & 0 \end{bmatrix} \begin{pmatrix} \boldsymbol{\delta}^k \\ v \end{pmatrix} = \begin{pmatrix} -(\boldsymbol{\nabla} f^k + c_k\mathbf{J}_2^{k\prime}\boldsymbol{h}_2^k) \\ 0 \end{pmatrix}. \tag{22}$$

$$\text{or more compactly as } \mathbf{B}^k\boldsymbol{x}^k = \boldsymbol{b}^k. \tag{23}$$

Rather than using the LDL or LU factorizations, we use the SVD factorization of $\mathbf{B}^k = \mathbf{U}\mathbf{S}\mathbf{V}'$ to solve for $\boldsymbol{x}^k$ as it is more numerically stable [9]. This is recommended because the block $\mathbf{H}_f^k + c_k\mathbf{J}_2^{k\prime}\mathbf{J}_2^k$ usually has a relatively high condition number. We then use $\boldsymbol{\delta}^k$ to compute the new estimate $\boldsymbol{e}^{k+1} = \boldsymbol{e}^k + \boldsymbol{\delta}^k$.

Controlling The Penalty Parameter Finding an effective strategy for adapting the penalty parameter c_k is the most critical ingredient for the success of a penalty-based algorithm [2, 10]. We consider updating c_k only if (a) we have done enough iterations (at least 3) with the current value of c_k to ensure the solution $\boldsymbol{e}^k$ has achieved some progress with the current value of c_k, and (b) the drop in the value of $||\boldsymbol{h}_2(\boldsymbol{e}^{k+1})||^2$ is found to be not adequate, i.e. $||\boldsymbol{h}_2(\boldsymbol{e}^{k+1})||^2 > \gamma||\boldsymbol{h}_2(\boldsymbol{e}^k)||^2$ where we set $\gamma = 0.5$. If any of the two conditions is not met, we keep $c_{k+1} = c_k$. Otherwise, we use the update rule $c_{k+1} = \min(\beta c_k, c_{\max})$ where the *penalty multiplier* $\beta > 1$ controls the speed and the robustness of the convergence (this is demonstrated empirically in Section 5). We set $c_0 = 10^{-5}$ and $c_{\max} = 10^9$.

Terminating Iterations We stop iterating if a pre-set maximum number of iterations is reached or if $\boldsymbol{e}^k$ converges (i.e. $||\boldsymbol{\delta}^k||^2 \le s_1 = 10^{-14}$) and the limit point is close enough to $\mathcal{E}$ (i.e. $d_{\mathcal{E}}(\boldsymbol{e}^k + \boldsymbol{\delta}^k) \le s_2 = 10^{-9}$).

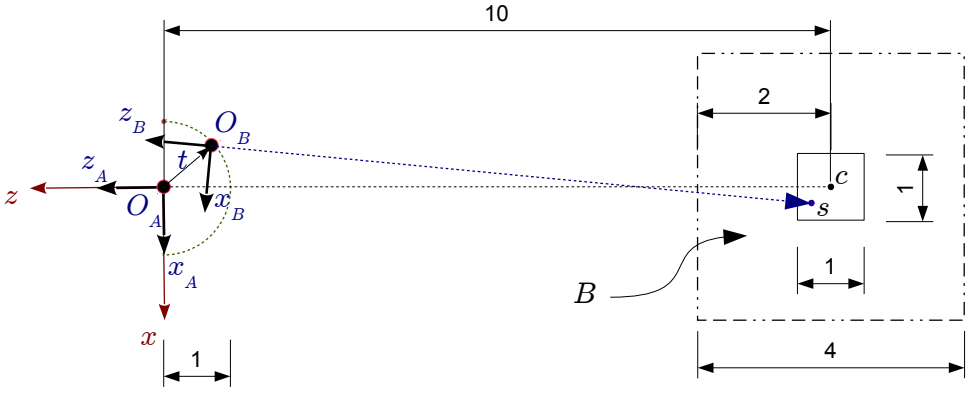

Figure 1: The $x-z$ planar view of the configuration of the synthetic scenes used in our experiments. The points are generated uniformly inside the cube B centered at $c = (0,0,-10)'$ having side length 4. The frame of camera A is fixed at the origin whereas the location O_B of camera B is chosen uniformly on the hemisphere with radius 1 in front of camera A. Camera B is oriented to have zero roll and such that it fixates at a random point $\boldsymbol{s}$ in the vicinity of $\boldsymbol{c}$. In particular, we set $\boldsymbol{s} = \boldsymbol{c} + \boldsymbol{u}$ where $\boldsymbol{u}$ is uniformly chosen from $[-0.5, 0.5]^3$. The two cameras are assumed to have equal focal length $f = 1000$ and square pixels.

5 Experimental Evaluation

We compare the performance of the proposed scheme and existing schemes for EsM estimation using synthetic and real data. We include in the comparison two instances of the proposed penalty-based algorithm: one with the penalty multiplier $\beta = 50$ (labeled as Proposed-$\beta = 50$) and another with $\beta = 4$ (Proposed-$\beta = 4$) to demonstrate the effect of the penalty multiplier β, which controls the rate at which the penalty parameter c_k is grown over time, on robustness and speed. The other schemes included in the comparison are (a) the overdetermined five-point scheme (5-pt) [13], (b) a manifold-based scheme that uses a global over-parametrization $\boldsymbol{e} : \mathbb{R}^3 \times \mathbb{R}^4 \rightarrow \mathcal{E}$ where the 7-D parameter vector $\boldsymbol{\theta}$ consists of a 3-vector representing translation and a 4-D quaternion encoding rotation (Global-Manifold) [7], and (c) Helmke's intrinsic manifold scheme using the local Cayley parametrization (Local-Manifold) [8]. We tried the other parametrizations from [8] but we included only the results of the Cayley parametrization as they all produce slightly differing results with the Cayley parameterization being occassionally the most robust.

All schemes are set to minimize the Sampson cost function. The Global-Manifold scheme uses Levenberg-Marquardt algorithm [5, 7] for the minimization. Among the possible candidate solutions generated by the five-point scheme, we select the candidate that yields the lowest RMS Sampson error. This is used in the comparison and is also used to initialize the rest of the schemes.

We excluded the eight-point scheme from the comparison as it yields very inaccurate results compared to the other schemes. Showing its results in the plots would make the curves of all other schemes get too close together to be easily compared.

Parameter Settings For all iterative schemes, we set the maximum number of iterations to 1000 and the convergence threshold $s_1 = 10^{-14}$. We ran all experiments using MATLAB on a Dell Precision T3600 workstation running Windows 7.

Metrics For each estimated solution $\mathbf{E}$ by a given scheme, we normalize $\mathbf{E}$ so that $||\mathbf{E}||_F^2 = 1$. Next, we measure the EsM manifold distance $d_{\mathcal{E}}(\mathbf{E})$ to assess how well the

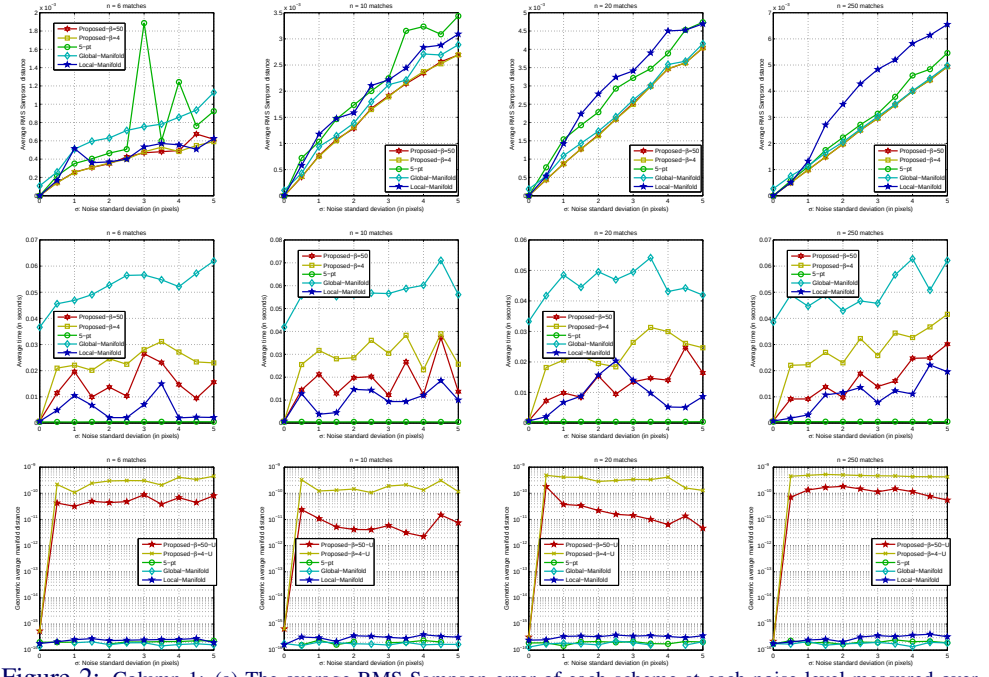

Figure 2: Column 1: (a) The average RMS Sampson error of each scheme at each noise level measured over 75 random scenes with $n = 6$ correspondences. (b) The average running time taken by each scheme for each noise level for $n = 6$. (c) The geometric mean of the manifold distances $d_{\mathcal{E}}$ of the estimates obtained by each scheme for each noise level for $n = 6$. The suffix '-U' in the labels of the proposed schemes denotes the absence of any SVD correction before measuring $d_{\mathcal{E}}$. Columns 2, 3, 4: The same plots of Column 1 but for $n = 10, 20$, and 250, respectively.

result $\mathbf{E}$ adheres to the EsM constraints. We then SVD-correct $\mathbf{E}$ and use it to compute the root-mean-square (RMS) of the Sampson distance d_s on the current subset of points. We also record the running time taken by each scheme. When we get multiple measurements from repeated experiments with a given scheme, we summarize these measurements with the arithmetic mean except for the manifold distance which is summarized by the geometric mean.

5.1 Synthetic Data

We generate 75 two-view scenes each involving a different random relative pose (and so a different EsM) and a different random set of n 3D points. The scenes follow the configuration shown in Fig. 1. For each of the 75 scenes, we project the 3D points on the two cameras and add noise to the pixel coordinates of the resulting correspondences. The noise added to each coordinate is IID Gaussian with zero mean and a standard deviation σ pixels. Different noise standard deviations between 0 and 5 pixels are tried with the correspondences of each scene. After calibrating the correspondences, we run each scheme on the 75 noisy scenes at each noise level and record its mean RMS Sampson error, mean running time and geometric mean of the manifold distance. The obtained results are graphed in Fig. 2 for $n = 6, 10, 20$, and 250 correspondences.

The graphs indicate that the lowest error is almost always achieved by the proposed penalty scheme. This is the case for the various values of n. The graphs also show that the

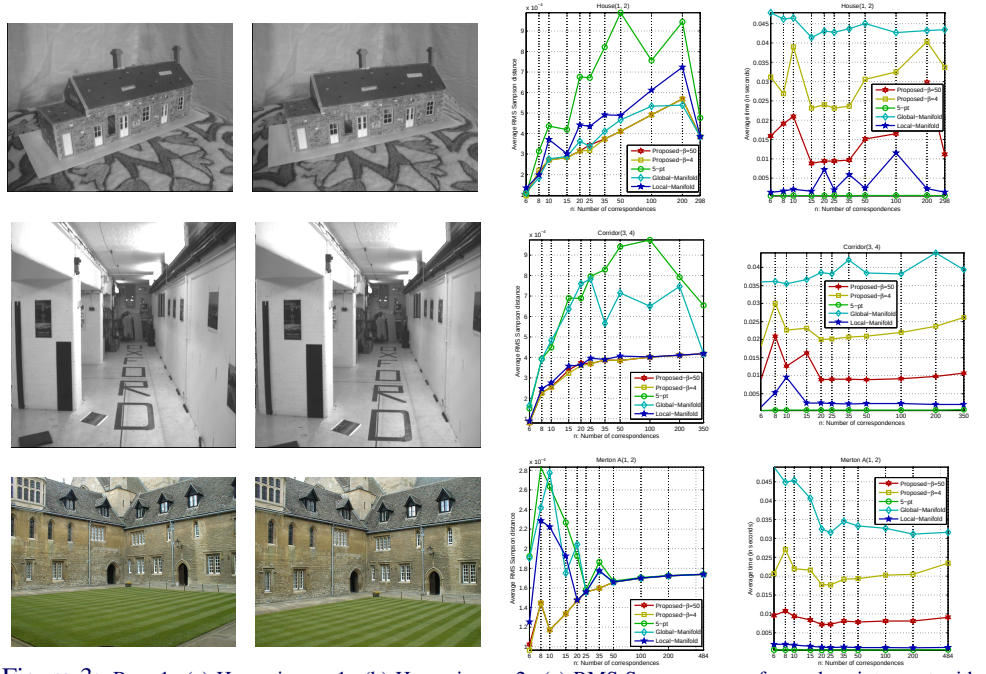

Figure 3: Row 1: (a) House image 1. (b) House image 2. (c) RMS Sampson error for each point count with average taken over 75 different random subsets. (d) The average of the corresponding running times. Row 2: (a) Corridor image 3. (b) Corridor image 4. (c) RMS Sampson error for each point count with average taken over 75 different random subsets. (d) The average of the corresponding running times. Row 3: (a) Merton A image 1. (b) Merton A image 2. (c) RMS Sampson error for each point count with average taken over 75 different random subsets. (d) The average of the corresponding running times.

Global-Manifold (GM), which is the slowest, produces more accurate results than the Local-Manifold (LM) scheme proposed by [8] for all n except $n = 6$. This may be explained by the fact that LM uses iteratively reweighted least-squares which results in fixed points that are not necessarily critical points of the Sampson cost function. Another explanation is that it uses a hybrid Newton/Gauss-Newton iterative scheme which is faster but less robust than the the Levenberg-Marquardt scheme employed by GM. The graphs also show that increasing the penalty multiplier β from 4 to 50 enhances the speed of the penalty-based scheme at the expense of a slight loss of robustness (see the sudden jump of the average RMS error of Proposed-$\beta = 50$ at $\sigma = 4.5$ for $n = 6$). The graphs also indicate that the penalty-based methods succeed in converging to estimates with manifold distances consistently below 10^{-9} (which is the convergence threshold used in our implementation).

5.2 Real Data

The Oxford dataset [21] offers a number of image sequences and provides a set of (noisy) coordinates of corresponding points between the images. The availability of the intrinsic camera parameters allows us to calibrate the coordinates and subsequently run EsM estimation. We use 3 pairs of images from 3 different sequences to compare the different schemes. For each pair, we try various numbers of correspondences (in addition to using the full set of correspondences) and form 75 random subsets of correspondences for each count considered. We then run the various EsM estimation techniques on these subsets and compute the average

RMS Sampson error and running time of each scheme. The results are shown in Fig. 3.

The graphs indicate that the proposed scheme (especially when $\beta = 4$) achieves generally lower error curves than the rest of the schemes except at a few locations ($n = 8,200$ in the House pair and $n = 20$ in the Corridor pair). While the difference in performance at these few locations is relatively small, it indicates that the proposed scheme may converge to local minima like other iterative schemes. When all the correspondences are used for estimation, all iterative schemes produce the same result unlike the five-point scheme which gives higher errors in the House and Corridor sequences. GM remains the slowest scheme and LM remains the fastest iterative scheme with the proposed scheme coming in between.

6 Concluding Remarks and Future Work

We have presented an iterative scheme for EsM estimation that augments the cost function with quadratic penalties to integrate the EsM constraints into the optimization. The proposed scheme can be used to minimize various types of cost functions as described in Section 4 although results were reported for the Sampson cost function only due to space limitation. We have also described a strategy for updating the penalty parameter c_k and have empirically demonstrated that the speed-robustness trade-off associated with selecting the speed of growing c_k. Experiments on synthetic and real data indicate the superiority of our scheme.

It is worth noting that we have attempted to use the Augmented Lagrangian Method (ALM) which is generally faster than penalty-based methods but faced a difficulty due to the fact that the set of EsM constraints are non-linearly dependent and outnumber the dimension of the EsM. One possible space for improvement is finding/designing alternative sets of constraints for EsMs that are less redundant than the set currently used in the paper and which can effectively be integrated into ALM or APFs for EsM estimation. Another direction of future work is investigating if APFs (or ALM) can be used effectively with other related problems (such as trifocal tensor estimation [7]) and seeing if they could improve on the state-of-the-art methods.

Acknowledgment

The authors would like to thank Venu Madhav Govindu for fruitful discussions and comments. These comments, as well as the comments of the anonymous reviewers, have helped improve the quality of this paper.

References

[1] Sameer Agarwal, Yasutaka Furukawa, Noah Snavely, Ian Simon, Brian Curless, Steven M Seitz, and Richard Szeliski. Building Rome in a day. In *Int'l Conf. on Comput. Vision (ICCV)*, 2009.

[2] Dimitri P. Bertsekas. *Nonlinear programming*. Athena Scientific, 1999.

[3] Mohammed E Fathy, Ashraf S Hussein, and Mohammed F Tolba. Simple, fast and accurate estimation of the fundamental matrix using the extended eight-point schemes. In *Proc. British Machine Vision Conf. (BMVC)*, 2010.

[4] Mohammed E Fathy, Ashraf S Hussein, and Mohammed F Tolba. Fundamental matrix estimation: A study of error criteria. *Pattern Recognition Letters*, 32(2):383–391, 2011.

[5] David A Forsyth and Jean Ponce. *Computer Vision: A Modern Approach*. Prentice Hall, August 2002. ISBN 0130851981.

[6] Richard Hartley and Hongdong Li. An efficient hidden variable approach to minimal-case camera motion estimation. *IEEE Trans. Pattern Anal. Machine Intell.*, 34(12): 2303–2314, 2012.

[7] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[8] Uwe Helmke, Knut Hüper, Pei Yean Lee, and John Moore. Essential matrix estimation using gauss-newton iterations on a manifold. *Int'l J. Comput. Vision*, 74(2):117–136, 2007.

[9] David Ronald Kincaid and Elliott Ward Cheney. *Numerical analysis: mathematics of scientific computing*, volume 2. American Mathematical Society, 2002.

[10] Zhouchen Lin, Risheng Liu, and Zhixun Su. Linearized alternating direction method with adaptive penalty for low-rank representation. In *NIPS*, volume 2, page 6, 2011.

[11] HC Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections. *Nature*, 293:133–135, 1981.

[12] Yi Ma, Jana Košecká, and Shankar Sastry. Optimization criteria and geometric algorithms for motion and structure estimation. *Int'l J. Comput. Vision*, 44(3):219–249, 2001.

[13] Yi Ma, Stefano Soatto, Jana Kosecka, and Shankar S. Sastry. *An Invitation to 3-D Vision: From Images to Geometric Models*. Springer, 2004.

[14] David Nistér. An efficient solution to the five-point relative pose problem. *IEEE Trans. Pattern Anal. Machine Intell.*, 26(6):756–777, 2004. ISSN 0162-8828.

[15] Kaare Brandt Petersen and Michael Syskind Pedersen. The matrix cookbook. *Technical University of Denmark*, pages 7–15, 2008.

[16] Xiang Ren and Zhouchen Lin. Linearized alternating direction method with adaptive penalty and warm starts for fast solving transform invariant low-rank textures. *Int'l J. Comput. Vision*, 104(1):1–14, 2013.

[17] Stefano Soatto, Ruggero Frezza, and Pietro Perona. Motion estimation on the essential manifold. *Proc. European Conf. on Comput. Vision (ECCV)*, pages 60–72, 1994.

[18] Henrik Stewenius, Christopher Engels, and David Nistér. Recent developments on direct relative orientation. *ISPRS J. of Photogrammetry and Remote Sensing*, 60(4): 284–294, 2006.

[19] Camillo J Taylor and David Kriegman. Structure and motion from line segments in multiple images. *IEEE Trans. Pattern Anal. Machine Intell.*, 17(11):1021–1032, 1995.

[20] Philip HS Torr and David W Murray. The development and comparison of robust methods for estimating the fundamental matrix. *Int'l J. Comput. Vision*, 24(3):271–300, 1997.

[21] Tomás Werner and Andrew Zisserman. New techniques for automated architectural reconstruction from photographs. In *Proc. European Conf. on Comput. Vision (ECCV)*, pages 541–555. 2002.