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

The Problem Given six or more pairs of corresponding points on two calibrated images, the accurate estimation of the essential matrix (EsM), which is a 3×3 matrix capturing the relative translation $\boldsymbol{t}$ and rotation $\mathbf{R}$ separating the two pinhole cameras, requires solving a nonlinear optimization problem subject to a set of constraints that guarantee the resulting 3×3 matrix has the structure of a valid EsM (i.e. $\mathbf{E}=[\boldsymbol{t}]_{\times}\mathbf{R}$, or equivalently $\mathrm{svd}(\mathbf{E})=\mathbf{U}\,\mathrm{diag}(1,1,0)\mathbf{V}'$, or equivalently $\mathbf{E}'\mathbf{E}\mathbf{E}'=0.5\,\mathrm{tr}(\mathbf{E}'\mathbf{E})\mathbf{E}'$). To the best of our knowledge, all existing schemes enforce the EsM constraints by performing the optimization on the manifold $\mathcal{E}$ of EsMs using either global [2] or local parametrizations [3]. No attempts were made to use the more straightforward approach of integrating the EsM constraint $\mathbf{E}'\mathbf{E}\mathbf{E}'=0.5\,\mathrm{tr}(\mathbf{E}'\mathbf{E})\mathbf{E}'$ directly into the optimization possibly because this 3×3 matrix equation as well as the homogeneity property of the EsM (i.e. $\mathbf{E}$ and $c\mathbf{E}$ represent the same EsM for all $c\neq 0$) give a total of ten (nonlinearly dependent) constraints while the number of variables in a 3×3 matrix is only nine.

Idea To avoid this problem, we propose to use adaptive penalty methods [1] to incorporate the matrix constraint into the optimization. Penalty methods relax the constraints (and so do not suffer from the too-many-constraints problem) while making violating them expensive. Assuming that $f(\boldsymbol{e})$ is the cost function measuring the (robust) algebraic or geometric fitting error of the 9-vector $\boldsymbol{e}$ corresponding to $\mathbf{E}$ and $\boldsymbol{h}_2(\boldsymbol{e})=\mathrm{vec}\{\mathbf{E}'\mathbf{E}\mathbf{E}'-0.5\,\mathrm{tr}(\mathbf{E}'\mathbf{E})\mathbf{E}'\}$ is the EsM constraint function, we define the penalty-augmented cost function $f_c(\boldsymbol{e})=f(\boldsymbol{e})+0.5c||\boldsymbol{h}_2(\boldsymbol{e})||^2$ where $c>0$ is called the penalty parameter. The two functions $f(\boldsymbol{e})$ and $f_c(\boldsymbol{e})$ are equal iff $\boldsymbol{e}\in\mathcal{E}$. Otherwise, $f_c(\boldsymbol{e})>f(\boldsymbol{e})$. Ideally, one would set c to a very high number or ∞ so that the minimizers of the original and penalty-augmented problems coincide. Such a strategy would fail to locate the (local) minimum precisely due to finite machine precision. Instead, we repeatedly compute the minimum of f_c for a gradually increasing sequence $\{c_k\}$ and we use the minimizer of f_{c_k} as an initial guess for the minimizer of $f_{c_{k+1}}$. If at iteration k the current estimate of the EsM is $\boldsymbol{e}^k$, we compute the update $\boldsymbol{\delta}^k\in\mathbb{R}^9$ on $\boldsymbol{e}^k$ by solving the following optimization problem:

$$\underset{\boldsymbol{\delta}^k\in\mathbb{R}^9}{\text{argmin}} \quad f_{c_k}(\boldsymbol{e}^k+\boldsymbol{\delta}^k)=f(\boldsymbol{e}^k+\boldsymbol{\delta}^k)+0.5c_k||\boldsymbol{h}_2(\boldsymbol{e}^k+\boldsymbol{\delta}^k)||^2, \quad (1)$$

$$\text{subject to} \quad \boldsymbol{e}^{k\prime}\boldsymbol{\delta}^k=0 \text{ (to ensure } \boldsymbol{e}^{k+1} \text{ stays away from zero).} \quad (2)$$

Solution Procedure Here we use the popular Gauss-Newton iteration to solve the above problem. In particular, we build a convex quadratic program (QP) approximation to the above problem by (a) replacing f with a convex second-order Taylor approximation $0.5\boldsymbol{\delta}^{k\prime}\mathbf{H}_f(\boldsymbol{e}^k)\boldsymbol{\delta}^k+\nabla' f(\boldsymbol{e}^k)\boldsymbol{\delta}^k+f(\boldsymbol{e}^k)$ 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:

$$\underset{\boldsymbol{\delta}^k\in\mathbb{R}^9}{\text{argmin}} \quad \frac{1}{2}\boldsymbol{\delta}^{k\prime}(\mathbf{H}_f^k+c_k\mathbf{J}_2^{k\prime}\mathbf{J}_2^k)\boldsymbol{\delta}^k+(\nabla f^k+c_k\mathbf{J}_2^{k\prime}\boldsymbol{h}_2^k)'\boldsymbol{\delta}^k+\text{const}, \quad (3)$$

$$\text{subject to} \quad \boldsymbol{e}^{k\prime}\boldsymbol{\delta}^k=0. \quad (4)$$

where $\mathbf{H}_f^k=\mathbf{H}_f(\boldsymbol{e}^k)$ and $\nabla f^k=\nabla f(\boldsymbol{e}^k)$. Introducing a scalar Lagrange multiplier v allows us to write the corresponding Lagrangian as:

$$L(\boldsymbol{\delta}^k,v)=\frac{1}{2}\boldsymbol{\delta}^{k\prime}(\mathbf{H}_f^k+c_k\mathbf{J}_2^{k\prime}\mathbf{J}_2^k)\boldsymbol{\delta}^k+(\nabla f^k+c_k\mathbf{J}_2^{k\prime}\boldsymbol{h}_2^k)'\boldsymbol{\delta}^k+v\boldsymbol{e}^{k\prime}\boldsymbol{\delta}^k+\text{const}. \quad (5)$$

The partial derivatives $\nabla_{\boldsymbol{\delta}^k}L$ and $\nabla_v L$ must be zero at the optimal $(\boldsymbol{\delta}^k,v)$ [1]. This gives rise to the following 10×10 symmetric linear system of equations:

$$\begin{bmatrix}\mathbf{H}_f^k+c_k\mathbf{J}_2^{k\prime}\mathbf{J}_2^k & \boldsymbol{e}^k\\ \boldsymbol{e}^{k\prime} & 0\end{bmatrix}\begin{pmatrix}\boldsymbol{\delta}^k\\ v\end{pmatrix}=\begin{pmatrix}-(\nabla f^k+c_k\mathbf{J}_2^{k\prime}\boldsymbol{h}_2^k)\\ 0\end{pmatrix}. \quad (6)$$

$$\text{or more compactly as } \mathbf{B}^k\boldsymbol{x}^k=\boldsymbol{b}^k. \quad (7)$$

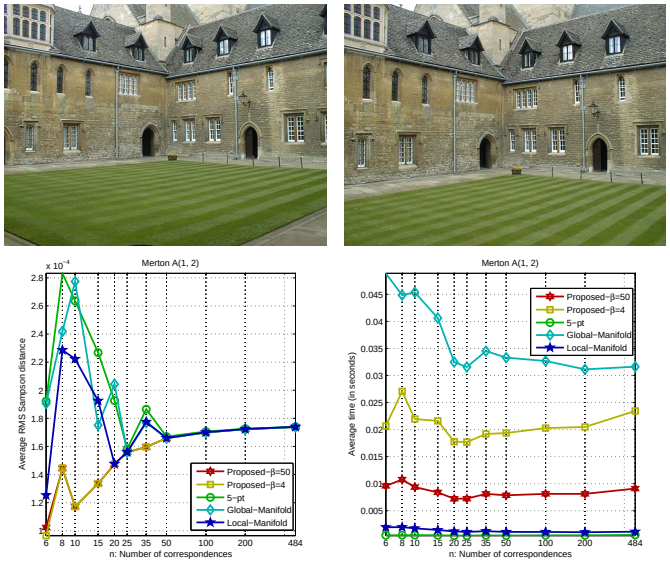

Figure 1: Row 1: (a) Merton A image 1. (b) Merton A image 2. Row 2: (a) RMS Sampson error for each point count with average taken over 75 different random subsets. (b) The average of the corresponding running times.

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. 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 [1]. 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. We set $c_0=10^{-5}$ and $c_{\max}=10^9$.

Experimental Evaluation We compared the performance of the proposed scheme and existing schemes for EsM estimation using synthetic and real data. We included 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 on robustness and speed. The other schemes included in the comparison were (a) the overdetermined five-point scheme (5-pt), (b) a manifold-based scheme using a global over-parametrization $\boldsymbol{e}:\mathbb{R}^3\times\mathbb{R}^4\to\mathcal{E}$ with the 7-D parameter vector $\boldsymbol{\theta}$ consisting of a 3-vector representing translation and a 4-D quaternion encoding rotation (Global-Manifold (GM)) [2], and (c) Helmke's intrinsic manifold scheme using the local Cayley parametrization (Local-Manifold (LM)) [3]. All schemes were set to minimize the Sampson cost function. Results for one real image pair are shown in Fig. 1. The graphs indicate that the proposed scheme (especially when $\beta=4$) achieves generally lower error curves than the rest of the schemes. GM remains the slowest scheme and LM remains the fastest iterative scheme with the proposed scheme coming in between.

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

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

[3] 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.