HAL
open science

Factorization Methods for Projective Structure and Motion

Bill Triggs

To cite this version:

Bill Triggs. Factorization Methods for Projective Structure and Motion. International Conference on Computer Vision & Pattern Recognition (CVPR '96), Jun 1996, San Francisco, United States. pp.845–851, 10.1109/CVPR.1996.517170 . inria-00548364

HAL Id: inria-00548364

https://inria.hal.science/inria-00548364v1

Submitted on 20 Dec 2010

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Factorization Methods for Projective Structure and Motion

Bill Triggs
INRIA Rhône-Alpes,
655 avenue de l'Europe, 38330 Montbonnot Saint-Martin, France.
Bill.Triggs@inrialpes.fr http://www.inrialpes.fr/MOVI/Triggs

Abstract

This paper describes a family of factorization-based algorithms that recover 3D projective structure and motion from multiple uncalibrated perspective images of 3D points and lines. They can be viewed as generalizations of the Tomasi-Kanade algorithm from affine to fully perspective cameras, and from points to lines. They make no restrictive assumptions about scene or camera geometry, and unlike most existing reconstruction methods they do not rely on 'privileged' points or images. All of the available image data is used, and each feature in each image is treated uniformly. The key to projective factorization is the recovery of a consistent set of *projective depths* (scale factors) for the image points: this is done using fundamental matrices and epipoles estimated from the image data. We compare the performance of the new techniques with several existing ones, and also describe an approximate factorization method that gives similar results to SVD-based factorization, but runs much more quickly for large problems.

Keywords: Multi-image Structure, Projective Reconstruction, Matrix Factorization.

1 Introduction

There has been considerable progress on scene reconstruction from multiple images in the last few years, aimed at applications ranging from very precise industrial measurement systems with several fixed cameras, to approximate structure and motion from real time video for active robot navigation. One can usefully begin by ignoring the issues of camera calibration and metric structure, initially recovering the scene up to an overall projective transformation and only later adding metric information if needed [5, 10, 1]. The key result is that projective reconstruction is the best that can be done without calibration or metric information about the scene, and that it is possible from at least two views of point-scenes or three views of line-scenes [2, 3, 8, 6].

Most current reconstruction methods either work *only* for the minimal number of views (typically two), or single out a few 'privileged' views for initialization before bootstrapping themselves to the multi-view case [5, 10, 9]. For robustness and accuracy, there is a need for methods that uniformly take account of all the data in all the images, without making restrictive special assumptions or relying on privileged features or images for initialization. The orthographic and paraperspective structure/motion factorization methods of Tomasi, Kanade and Poelman [17, 11] partially fulfill these requirements, but they only apply when the camera projections are well approximated by affine mappings. This happens only for cameras viewing small, distant scenes, which is seldom the case in practice. Factorization methods for perspective images are needed, however it has not been clear how to find the unknown projective scale factors of the image measurements that are required for this. (In the affine case the scales are constant and can be eliminated).

As part of the current blossoming of interest in multi-image reconstruction, Shashua [14] recently extended the well-known two-image epipolar constraint to a trilinear constraint between matching points in three images. Hartley [6] showed that this constraint also applies to lines in three images, and Faugeras & Mourrain [4] and I [18, 19] completed that corner of the puzzle by systematically studying the constraints for lines and points in any number of images. A key aspect of the viewpoint presented in [18, 19] is that projective reconstruction is essentially a matter of recovering a coherent set of **projective depths** — projective scale factors that represent the depth information lost during image projection. These are exactly the missing factorization scales mentioned above. They satisfy a set of consistency conditions called 'joint image reconstruction equations' [18], that link them together via the corresponding image point coordinates and the various inter-image matching tensors.

In the MOVI group, we have recently been developing projective structure and motion algorithms based on this 'projective depth' picture. Several of these methods use the factorization paradigm, and so can be viewed as generalizations of the Tomasi-Kanade method from affine to fully perspective projections. However they also require a depth recovery phase that is not present in the affine case. The basic reconstruction method for point images was introduced in [15]. The current paper extends this in several directions, and presents a detailed assessment of the performance of the new methods in comparison to existing techniques such as Tomasi-Kanade factorization and Levenberg-Marquardt nonlinear least squares. Per-

To appear in CVPR'96. This work was supported by an EC HCM grant and INRIA Rhône-Alpes. I would like to thank Peter Sturm and Richard Hartley for enlightening discussions.

haps the most significant result in the paper is the extension of the method to work for lines as well as points, but I will also show how the factorization can be iteratively 'polished' (with results similar to nonlinear least squares iteration), and how any factorization-based method can be speeded up significantly for large problems, by using an approximate fixed-rank factorization technique in place of the Singular Value Decomposition.

The factorization paradigm has two key attractions that are only enhanced by moving from the affine to the projective case: *(i)* All of the data in all of the images is treated uniformly — there is no need to single out 'privileged' features or images for special treatment; *(ii)* No initialization is required and convergence is virtually guaranteed by the nature of the numerical methods used. Factorization also has some well known disadvantages:

1) Every primitive must be visible in every image. This is unrealistic in practice given occlusion and extraction and tracking failures.

2) It is not possible to incorporate a full statistical error model for the image data, although some sort of implicit least-squares trade-off *is* made.

3) It is not clear how to incorporate additional points or images incrementally: the whole calculation must be redone.

4) SVD-based factorization is slow for large problems.

Only the speed problem will be considered here. SVD is slow because it was designed for general, full rank matrices. For matrices of fixed low rank r (as here, where the rank is 3 for the affine method or 4 for the projective one), approximate factorizations can be computed in time $\mathcal{O}(mnr)$, *i.e.* directly proportional to the size of the input data.

The Tomasi-Kanade 'hallucination' process can be used to work around missing data [17], as in the affine case. However this greatly complicates the method and dilutes some of its principal benefits. There is no obvious solution to the error modelling problem, beyond using the factorization to initialize a nonlinear least squares routine (as is done in some of the experiments below). It would probably be possible to develop incremental factorization update methods, although there do not seem to be any in the standard numerical algebra literature.

The rest of the paper outlines the theory of projective factorization for points and lines, describes the final algorithms and implementation, reports on experimental results using synthetic and real data, and concludes with a discussion. The full theory of projective depth recovery applies equally to two, three and four image matching tensors, but throughout this paper I will concentrate on the two-image (fundamental matrix) case for simplicity. The underlying theory for the higher valency cases can be found in [18].

2 Point Reconstruction

We need to recover 3D structure (point locations) and motion (camera calibrations and locations) from m uncalibrated perspective images of a scene containing n 3D points. Without further information it is only possible to reconstruct the scene up to an overall projective transformation [2, 8], so we will work in homogeneous coordinates with respect to arbitrary projective coordinate frames. Let $\mathbf{X}_\mathrm{p}$ ($p = 1, \ldots, n$) be the unknown homogeneous 3D point vectors, $\mathbf{P}_\mathrm{i}$ ($i = 1, \ldots, m$) the unknown 3×4 image projections, and $\mathbf{x}_\mathrm{ip}$ the measured homogeneous image point vectors. Modulo some scale factors λ_ip, the image points are projected from the world points: $\lambda_\mathrm{ip}\, \mathbf{x}_\mathrm{ip} = \mathbf{P}_\mathrm{i}\, \mathbf{X}_\mathrm{p}$. Each object is defined only up to rescaling. The λ's 'cancel out' the arbitrary scales of the image points, but there is still the freedom to: (*i*) arbitrarily rescale each world point $\mathbf{X}_\mathrm{p}$ and each projection $\mathbf{P}_\mathrm{i}$; (*ii*) apply an arbitrary nonsingular 4×4 projective deformation $\mathbf{T}$: $\mathbf{X}_\mathrm{p} \to \mathbf{T}\mathbf{X}_\mathrm{p}$, $\mathbf{P}_\mathrm{i} \to \mathbf{P}_\mathrm{i}\mathbf{T}^{-1}$. Modulo changes of the λ_ip, the image projections are invariant under both of these transformations.

The scale factors λ_ip will be called **projective depths**. With correctly normalized points and projections they become true optical depths, *i.e.* orthogonal distances from the focal planes of the cameras. (NB: this is not the same as Shashua's 'projective depth' [13]). In general, $m + n - 1$ projective depths can be set arbitrarily by choosing appropriate scales for the $\mathbf{X}_\mathrm{p}$ and $\mathbf{P}_\mathrm{i}$. However, once this is done the remaining $(m-1)(n-1)$ degrees of freedom contain real information that can be used for 3D reconstruction: taken as a whole the projective depths have a strong internal coherence. In fact, [18, 19] argues that just as the key to calibrated stereo reconstruction is the recovery of Euclidean depth, the essence of projective reconstruction is precisely the recovery of a coherent set of projective depths modulo overall projection and world point rescalings. Once this is done, reconstruction reduces to choosing a projective basis for a certain abstract three dimensional 'joint image' subspace, and reading off point coordinates with respect to it.

2.1 Factorization

Gather the point projections into a single $3m \times n$ matrix equation:

$$\mathbf{W} \equiv \begin{pmatrix} \lambda_{11}\,\mathbf{x}_{11} & \lambda_{12}\,\mathbf{x}_{12} & \cdots & \lambda_{1\mathrm{n}}\,\mathbf{x}_{1\mathrm{n}} \\ \lambda_{21}\,\mathbf{x}_{21} & \lambda_{22}\,\mathbf{x}_{22} & \cdots & \lambda_{2\mathrm{n}}\,\mathbf{x}_{2\mathrm{n}} \\ \vdots & \vdots & \ddots & \vdots \\ \lambda_{\mathrm{m}1}\,\mathbf{x}_{\mathrm{m}1} & \lambda_{\mathrm{m}2}\,\mathbf{x}_{\mathrm{m}2} & \cdots & \lambda_{\mathrm{mn}}\,\mathbf{x}_{\mathrm{mn}} \end{pmatrix} = \begin{pmatrix} \mathbf{P}_1 \\ \mathbf{P}_2 \\ \vdots \\ \mathbf{P}_\mathrm{m} \end{pmatrix} \begin{pmatrix} \mathbf{X}_1 & \mathbf{X}_2 & \cdots & \mathbf{X}_\mathrm{n} \end{pmatrix}$$

Hence, with a consistent set of projective depths the **rescaled measurement matrix** $\mathbf{W}$ has rank at most 4. Any rank 4 ma-

trix can be factorized into some $3m \times 4$ matrix of 'projections' multiplying a $4 \times n$ matrix of 'points' as shown, and any such factorization corresponds to a valid projective reconstruction: the freedom in factorization is exactly a 4×4 nonsingular linear transformation $\mathbf{P} \rightarrow \mathbf{P}\,\mathbf{T}^{-1},\ \mathbf{X} \rightarrow \mathbf{T}\,\mathbf{X}$, which can be regarded as a projective transformation of the reconstructed 3D space.

One practical method of factorizing $\mathbf{W}$ is the Singular Value Decomposition [12]. This decomposes an arbitrary $k \times l$ matrix $\mathbf{W}_{\mathrm{k}\times\mathrm{l}}$ of rank r into a product $\mathbf{W}_{\mathrm{k}\times\mathrm{l}} = \mathbf{U}_{\mathrm{k}\times\mathrm{r}}\,\mathbf{D}_{\mathrm{r}\times\mathrm{r}}\,\mathbf{V}_{\mathrm{l}\times\mathrm{r}}^{\top}$, where the columns of $\mathbf{V}_{\mathrm{l}\times\mathrm{r}}$ and $\mathbf{U}_{\mathrm{k}\times\mathrm{r}}$ are orthonormal bases for the input (co-kernel) and output (range) spaces of $\mathbf{W}_{\mathrm{k}\times\mathrm{l}}$, and $\mathbf{D}_{\mathrm{r}\times\mathrm{r}}$ is a diagonal matrix of positive decreasing 'singular values'. The decomposition is unique when the singular values are distinct, and can be computed stably and reliably in time $\mathcal{O}(kl\min(k,l))$. The matrix $\mathbf{D}$ of singular values can be absorbed into either $\mathbf{U}$ or $\mathbf{V}$ to give a decomposition of the projection/point form $\mathbf{P}\mathbf{X}$. (I absorb it into $\mathbf{V}$ to form $\mathbf{X}$).

The SVD has been used by Tomasi, Kanade and Poelman [17, 11] for their affine (orthographic and paraperspective) reconstruction techniques. The current application can be viewed as a generalization of these methods to projective reconstruction. The projective case leads to slightly larger matrices ($3m \times n$ rank 4 as opposed to $2m \times n$ rank 3), but is actually simpler than the affine case as there is no need to subtract translation terms or apply nonlinear constraints to guarantee the orthogonality of the projection matrices.

Ideally, one would like to find reconstructions in time $\mathcal{O}(mn)$ (the size of the input data). SVD is a factor of $\mathcal{O}(\min(3m,n))$ slower than this, which can be significant if there are many points and images. Although SVD is probably near-optimal for full-rank matrices, rank r matrices can be factorized in 'output sensitive' time $\mathcal{O}(mnr)$. I have experimented with one such 'fixed rank' method, and find it to be almost as accurate as SVD and significantly faster for large problems. The method repeatedly sweeps the matrix, at each sweep guessing and subtracting a column-vector that 'explains' as much as possible of the residual error in the matrix columns. A rank r matrix is factorized in r sweeps. When the matrix is not exactly of rank r the guesses are not quite optimal and it is useful to include further sweeps (say $2r$ in total) and then SVD the matrix of extracted columns to estimate the best r combinations of them.

2.2 Projective Depth Recovery

The above factorization techniques can *only* be used if a self-consistent set of projective depths λ_{ip} can be found. The key technical advance that makes this work possible is a practical method for estimating these using fundamental matrices and epipoles obtained from the image data. The full theory can be found in [18], which also describes how to use trivalent and quadrivalent matching tensors for depth recovery. Here we briefly sketch the fundamental matrix case. The image projections $\lambda_{\mathrm{ip}}\,\mathbf{x}_{\mathrm{ip}} = \mathbf{P}_{\mathrm{i}}\,\mathbf{X}_{\mathrm{p}}$ imply that the 6×5 matrix

$$\left(\begin{array}{c|c} \mathbf{P}_{\mathrm{i}} & \lambda_{\mathrm{ip}}\,\mathbf{x}_{\mathrm{ip}} \\ \mathbf{P}_{\mathrm{j}} & \lambda_{\mathrm{jp}}\,\mathbf{x}_{\mathrm{jp}} \end{array}\right) = \left(\begin{array}{c} \mathbf{P}_{\mathrm{i}} \\ \mathbf{P}_{\mathrm{j}} \end{array}\right) \left(\begin{array}{c|c} \mathbf{I}_{4\times 4} & \mathbf{X}_{\mathrm{p}} \end{array}\right)$$

has rank at most 4, so all of its 5×5 minors vanish. Expanding by cofactors in the last column gives homogeneous linear equations in the components of $\lambda_{\mathrm{ip}}\,\mathbf{x}_{\mathrm{ip}}$ and $\lambda_{\mathrm{jp}}\,\mathbf{x}_{\mathrm{jp}}$, with coefficients that are 4×4 determinants of projection matrix rows. These turn out to be the expressions for the fundamental matrix $\mathbf{F}_{\mathrm{ij}}$ and epipole $\mathbf{e}_{\mathrm{ji}}$ of camera j in image i in terms of projection matrix components [19, 4]. The result is the **projective depth recovery equation**:

$$(\mathbf{F}_{\mathrm{ij}}\,\mathbf{x}_{\mathrm{jp}})\,\lambda_{\mathrm{jp}} \;=\; (\mathbf{e}_{\mathrm{ji}} \wedge \mathbf{x}_{\mathrm{ip}})\,\lambda_{\mathrm{ip}} \qquad (1)$$

This says two things: (*i*) The epipolar line of $\mathbf{x}_{\mathrm{jp}}$ in image i is the same as the line through the corresponding point $\mathbf{x}_{\mathrm{ip}}$ and epipole $\mathbf{e}_{\mathrm{ji}}$ (as is well known); (*ii*) *With the correct projective depths and scalings for* $\mathbf{F}_{\mathrm{ij}}$ *and* $\mathbf{e}_{\mathrm{ji}}$*, the two terms have exactly the same size*. The equality is exact, not just up to scale. This is the new result that allows us to recover projective depths using fundamental matrices and epipoles. Analogous results based on higher order matching tensors can be found in [18].

It is straightforward to recover projective depths using (1). Each instance of it linearly relates the depths of a single 3D point in two images. By estimating a sufficient number of fundamental matrices and epipoles, we can amass a system of homogeneous linear equations that allows the complete set of depths for a given point to be found, up to an arbitrary overall scale factor. At a minimum, this can be done by selecting any set of $m-1$ equations that link the m images into a single connected graph. With such a non-redundant set of equations the depths for each point p can be found trivially by chaining together the solutions for each image, starting from some arbitrary initial value such as $\lambda_{1\mathrm{p}} = 1$. Solving the depth recovery equation in least squares gives a simple recursion relation for λ_{ip} in terms of λ_{jp} :

$$\lambda_{\mathrm{ip}} \;:=\; \frac{(\mathbf{e}_{\mathrm{ji}} \wedge \mathbf{x}_{\mathrm{ip}}) \cdot (\mathbf{F}_{\mathrm{ij}}\,\mathbf{x}_{\mathrm{jp}})}{\|\mathbf{e}_{\mathrm{ji}} \wedge \mathbf{x}_{\mathrm{ip}}\|^2}\,\lambda_{\mathrm{jp}}$$

If additional depth recovery equations are used, this simple recursion must be replaced by a redundant (and hence potentially more robust) homogeneous linear system. However, care is needed. The depth recovery equations are sensitive to the scale factors chosen for the $\mathbf{F}$'s and $\mathbf{e}$'s, and these can not be recovered directly from the image data. This is irrelevant when a single chain of equations is used, as rescalings of $\mathbf{F}$ and $\mathbf{e}$ affect all points equally and hence amount to rescalings of the corresponding projection matrices. However with redundant equations it is essential to choose a mutually self-consistent set of scales for the $\mathbf{F}$'s and $\mathbf{e}$'s. I will not describe this process here, except to note that the consistency condition is the Grassmann identity $\mathbf{F}_{\mathrm{kj}}\mathbf{e}_{\mathrm{ij}} = \mathbf{e}_{\mathrm{ik}} \wedge \mathbf{e}_{\mathrm{jk}}$ [18].

It is still unclear what the best trade-off between economy and robustness is for depth recovery. This paper considers only

two simple non-redundant choices: either the images are taken pairwise in sequence, $\mathbf{F}_{21}, \mathbf{F}_{32}, \ldots, \mathbf{F}_{\mathrm{m\,m-1}}$, or all subsequent images are scaled in parallel from the first, $\mathbf{F}_{21}, \mathbf{F}_{31}, \ldots, \mathbf{F}_{\mathrm{m1}}$. It might seem that long chains of rescalings would prove numerically unstable, but in practice depth recovery is surprisingly well conditioned. Both serial and parallel chains work very well despite their non-redundancy and chain length or reliance on a 'key' image. The two methods give similar results except when there are many (>40) images, when the shorter chains of the parallel system become more robust. Both are stable even when epipolar *point* transfer is ill-conditioned (*e.g.* for a camera moving in a straight line, when the epipolar lines of different images coincide): the image observations act as stable 'anchors' for the transfer process.

Balancing: A further point is that with arbitrary choices of scale for the fundamental matrices and epipoles, the average size of the recovered depths might tend to increase or decrease exponentially during the solution-chaining process. Theoretically this is not a problem as the overall scales are arbitrary, but it could easily make the factorization phase numerically ill-conditioned. To counter this the recovered matrix of projective depths must be balanced after it has been built, by judicious overall row and column rescalings. The process is very simple. The image points are normalized on input, so ideally all of the scale factors λ_{ip} should have roughly the same order of magnitude, $\mathcal{O}(1)$ say. For each point the depths are estimated as above, and then: (*i*) each row (image) of the estimated depth matrix is rescaled to have length $\sqrt{n}$; (*ii*) each column (point) of the resulting matrix is rescaled to length $\sqrt{m}$. This process is repeated until it roughly converges, which happens very quickly (within 2–3 iterations).

3 Line Reconstruction

3D lines can also be reconstructed using the above techniques. A line $\mathbf{L}$ can be represented by any two 3D points lying on it, say $\mathbf{Y}$ and $\mathbf{Z}$. In image i, $\mathbf{L}$ projects to some image line $\mathbf{l}_{\mathrm{i}}$ and $\mathbf{Y}$ and $\mathbf{Z}$ project to image points $\mathbf{y}_{\mathrm{i}}$ and $\mathbf{z}_{\mathrm{i}}$ lying on $\mathbf{l}_{\mathrm{i}}$. The points $\{\mathbf{y}_{\mathrm{i}} | i = 1, \ldots, m\}$ are in epipolar correspondence, so they can be used in the depth recovery equation (1) to reconstruct $\mathbf{Y}$, and similarly for $\mathbf{Z}$. The representatives $\mathbf{Y}$ and $\mathbf{Z}$ can be fixed implicitly by choosing $\mathbf{y}_1$ and $\mathbf{z}_1$ arbitrarily on $\mathbf{l}_1$ in the first image, and using the epipolar constraint to transfer these to the corresponding points in the remaining images: $\mathbf{y}_{\mathrm{i}}$ lies on both $\mathbf{l}_{\mathrm{i}}$ and the epipolar line of $\mathbf{y}_1$, so is located at their intersection.

In fact, epipolar transfer and depth recovery can be done in one step. Let $\mathbf{y}_{\mathrm{i}}$ stand for the *rescaled* via points $\mathbf{P}_{\mathrm{i}}\mathbf{Y}$. Substitute these into equation (1), cross-product with $\mathbf{l}_{\mathrm{i}}$, expand, and simplify using $\mathbf{l}_{\mathrm{i}} \cdot \mathbf{y}_{\mathrm{i}} = 0$:

$$\begin{aligned} \mathbf{l}_{\mathrm{i}} \wedge (\mathbf{F}_{\mathrm{ij}}\, \mathbf{y}_{\mathrm{j}}) &= \mathbf{l}_{\mathrm{i}} \wedge (\mathbf{e}_{\mathrm{ji}} \wedge \mathbf{y}_{\mathrm{i}}) \\ &= -\,(\mathbf{l}_{\mathrm{i}} \cdot \mathbf{e}_{\mathrm{ji}})\, \mathbf{y}_{\mathrm{i}} + (\mathbf{l}_{\mathrm{i}} \cdot \mathbf{y}_{\mathrm{i}})\, \mathbf{e}_{\mathrm{ji}} \\ &= -\,(\mathbf{l}_{\mathrm{i}} \cdot \mathbf{e}_{\mathrm{ji}})\, \mathbf{y}_{\mathrm{i}} \qquad (2) \end{aligned}$$

Up to a factor of $\mathbf{l}_{\mathrm{i}} \cdot \mathbf{e}_{\mathrm{ji}}$, *the intersection* $\mathbf{l}_{\mathrm{i}} \wedge (\mathbf{F}_{\mathrm{ij}}\, \mathbf{y}_{\mathrm{j}})$ *of* $\mathbf{l}_{\mathrm{i}}$ *with the epipolar line of* $\mathbf{y}_{\mathrm{j}}$ *automatically gives the correct projective depth for reconstruction.* Hence, factorization-based line reconstruction can be implemented by choosing a suitable (widely spaced) pair of via-points on each line in the first image, and then chaining together instances of equation (2) to find the corresponding, correctly scaled via-points in the other images. The required fundamental matrices can not be found directly from line matches, but they can be estimated from point matches, or from the trilinear line matching constraints (trivalent tensor) [6, 14, 4, 19, 18]. Alternatively, the trivalent tensor can be used directly: in tensorial notation [18], the trivalent via-point transfer equation is ${\mathrm{l}_{\mathrm{B_k}}}\, \mathbf{G}_{\mathrm{C_j}}{}^{\mathrm{A_i B_k}}\, \mathbf{y}^{\mathrm{C_j}} = ({\mathrm{l}_{\mathrm{B_k}}}\, \mathbf{e}_{\mathrm{j}}^{\mathrm{B_k}})\mathbf{y}^{\mathrm{A_i}}$.

As with points, redundant equations may be included if and only if a self-consistent normalization is chosen for the fundamental matrices and epipoles. For numerical stability, it is essential to balance the resulting via-points (*i.e.* depth estimates). This works with the $3m \times 2n_{\text{lines}}$ '$\mathbf{W}$' matrix of via-points, iteratively rescaling all coordinates of each image (triple of rows) and all coordinates of each line (pair of columns) until an approximate equilibrium is reached, where the overall mean square size of each coordinate is $\mathcal{O}(1)$ in each case. To ensure that the via-points representing each line are on average well separated, I also orthonormalize the two $3m$-component column vectors for each line with respect to one another. The via-point equations (2) are linear and hence invariant with respect to this, but it does of course change the 3D representatives $\mathbf{Y}$ and $\mathbf{Z}$ recovered for each line.

4 Implementation

This section summarizes the complete algorithm for factorization-based 3D projective reconstruction from image points and lines, and discusses a few important implementation details and variants. The algorithm goes as follows:

0) Extract and match points and lines across all images.

1) Standardize all image coordinates (see below).

2) Estimate a set of fundamental matrices and epipoles sufficient to chain all the images together (*e.g.* using point matches).

3) For each point, estimate the projective depths using equation (1). Build and balance the depth matrix λ_{ip}, and use it to build the rescaled point measurement matrix $\mathbf{W}$.

4) For each line choose two via-points and transfer them to the other images using the transfer equations (2). Build and balance the rescaled line via-point matrix.

5) Combine the line and point measurement matrices into a $3m \times (n_{\text{points}} + 2n_{\text{lines}})$ data matrix and factorize it using either SVD or the fixed-rank method. Recover 3D projective structure (point and via-point coordinates) and motion (projection matrices) from the factorization.

6) Un-standardize the projection matrices (see below).

Complexity: The algorithm is dominated by the $\mathcal{O}(mn \min(3m, n))$ SVD step if this is used, while if an approximate factorization is used it is proportional to the input data size $\mathcal{O}(mn)$.

Standardization: To get acceptable results from the above algorithm, it is *absolutely essential* to work in a well-adapted image coordinate system. The basic idea is to choose working coordinates that reflect the least squares trade-offs implicit in the factorization algorithm. This is standard practice in numerical analysis, but it does not seem to have been widely known in vision until Hartley [7] pointed out its importance for fundamental matrix estimation. The exact scheme used is not critical, provided that the homogeneous working coordinates are all of the same order of magnitude. I currently prefer to scale the image into the unit square $[-1, 1] \times [-1, 1]$, homogenize, and then normalize the resulting homogeneous 3-vectors to unit length $x^2 + y^2 + z^2 = 1$. This simple scheme works very well in practice. The normalization applies to line vectors as well as point ones, and behaves well even for points (*e.g.* epipoles) near the line at infinity. After reconstruction, the camera projections need to be un-standardized by multiplying by the inverse transformation.

4.1 Generalizations & Variants

I have implemented and experimented with a number of variants of the above algorithm, the more promising of which are featured in the experiments described below.

Iterative Factorization: The projective depths depend on the 3D structure, which in turn derives from the depths. The reconstruction can be iteratively improved by reprojecting to refine the depth estimates and then re-factorizing. For points one finds the component of the reprojected 3D point vector along each image vector, while for lines the reprojected via-point is perturbed orthogonally to lie on the measured image line. With SVD-based factorization and standardized image coordinates the iteration turns out to be extremely stable, and always improves the recovered structure slightly (often significantly for lines). For points, one can even start with arbitrary initial depths (say the affine ones $\lambda_{\mathrm{ip}} = 1$) and iterate to convergence. This requires no fundamental matrices or depth recovery equations and converges reliably in practice, although it can be rather slow if started far from the true solution.

Nonlinear Least Squares: The 'linear' factorization-based projective reconstruction methods described above are a suitable starting point for more refined nonlinear least-squares estimation. This can take account of image point error models, camera calibrations, or Euclidean constraints, as in the work of Szeliski and Kang [16], Hartley [5] and Mohr, Boufama and Brand [10]. The standard workhorse for such problems is Levenberg-Marquardt iteration [12], so for comparison with the linear methods I have implemented simple L-M based projective reconstruction algorithms. These can be initialized from either fixed-rank or SVD-based factorizations. For lines the recovered structure is often improved significantly, while for points the improvement over the linear methods is usually small.

Affine Factorization: To illustrate the advantages of projective factorization over the original Tomasi-Kanade-Poelman work [17, 11], I have also implemented affine SVD-based point reconstruction. This gives rather poor results in the below experiments because the perspective distortions are quite large.

5 Experiments

To quantify the performance of the various algorithms, I have run a large number of simulations using synthetic data, and also tested the algorithms on manually matched primitives derived from real images. There is only space for a very brief summary here, more details can be found in [20].

The simulations are based on trial scenes consisting of random 3D points and lines in the unit cube $[-1, 1] \times [-1, 1] \times [-1, 1]$, perturbed by uniform noise and viewed by identical perspective cameras in various arrangements. In the graphs shown here, the cameras are spaced uniformly along a 90 degree arc of radius 2 in the equatorial plane of the scene, and are directed towards the scene centre (*i.e.* there is a large baseline and significant perspective distortion). Reconstruction error is measured over 50 trials, after least-squares projective alignment with the true 3D structure. Mean errors are reported for points, while for lines there are always outliers so median errors are used[1].

Fundamental matrices and epipoles are estimated using the linear least squares method with all the available point matches, followed by a supplementary SVD to project the fundamental matrices to rank 2 and find the epipoles. In standardized coordinates this method performs very well [7], and it has not proved necessary to refine the results with a nonlinear method. Unless otherwise noted, the projective depths of points are recovered by chaining sequentially through the images: $\mathbf{F}_{12}, \mathbf{F}_{23}, \ldots, \mathbf{F}_{\mathrm{m-1\ m}}$. A parallel chain $\mathbf{F}_{12}, \mathbf{F}_{13}, \ldots, \mathbf{F}_{1\ \mathrm{m}}$ usually gives similar results. For lines in more than a few images, the parallel chain is superior and is used by default.

Fig. 1 shows the sensitivity of various point and line reconstruction methods to image noise, number of views, and number of scene primitives (points or lines). The methods shown are: *points:* fundamental matrix depth recovery with SVD and fixed-rank factorization, iterated SVD and nonlinear least-squares initialized from SVD; *lines:* fundamental matrix and trilinear parallel and serial via-point transfer followed by SVD, iterated SVD, and SVD plus nonlinear least-squares.

[1] The image of a line passing near the optical centre of a camera is extremely sensitive to small 3D perturbations. Also, if the camera centres lie in a plane (as here), all lines in that plane have the same image, so such lines can not be uniquely reconstructed (*c.f.* axial points for cameras lying in a line; in this case, only lines skew with the axis can be reconstructed).

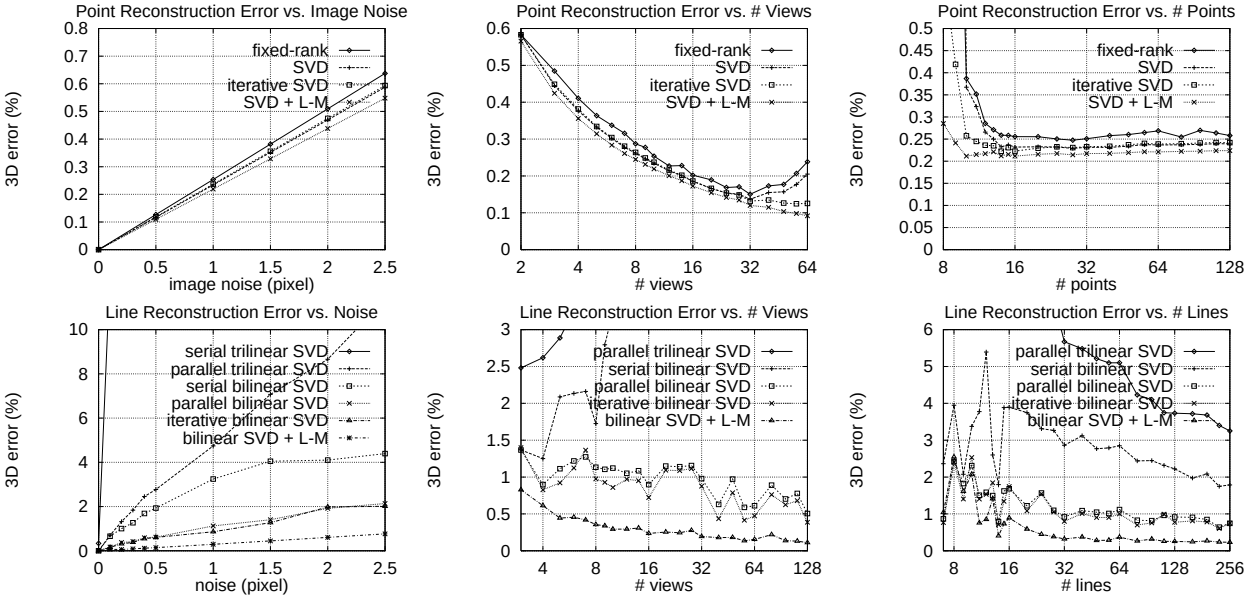

Figure 1: Mean 3D reconstruction error for points and lines, *vs.* noise, number of views and number of primitives. Defaults: ± 1 pixel noise; 10 views; 50 primitives.

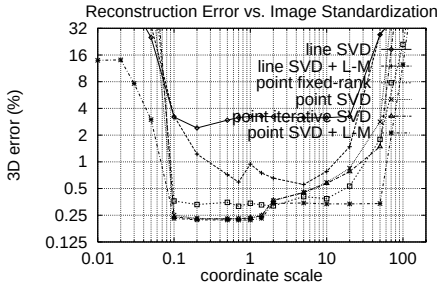

Figure 2: Reconstruction error *vs.* image standardization.

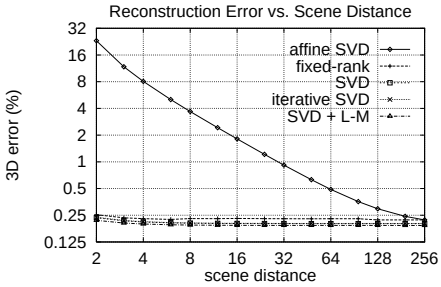

Figure 3: Projective and affine reconstruction *vs.* scene distance.

All of the point methods are very stable. Their errors vary linearly with noise and decrease as more points or views are added. There is not much difference in precision, but generally the fixed-rank method is slightly less accurate (but significantly faster) than SVD. Iterating the SVD makes a small improvement, and nonlinear least-squares is slightly more accurate again. Serial depth recovery chains become ill-conditioned when more than 30-40 images are chained: beyond this parallel chaining is advised.

Line reconstruction is less stable. Only the least-squares methods consistently give reconstruction errors commensurate with the input noise. Parallel F-matrix transfer plus factorization is a factor of 2 or more worse than this, and serial transfer is worse again. Iterative factorization helps a little, but the use of a nonlinear least-squares routine is still advisable. Any of these methods are accurate enough for reliable initialization of the least-squares iteration. If my implementation is correct, trilinear transfer based reconstruction is too sensitive to noise to be useful (this requires confirmation). For all of the above methods, there are outliers corresponding to lines that either can not be reconstructed uniquely, or are very sensitive to small 3D perturbations.

The importance of standardization is illustrated in fig. 2, where the image coordinates are standardized to $\mathcal{O}(\text{scale})$ rather than $\mathcal{O}(1)$ before reconstruction. Pixel coordinates correspond to a scale of 256 and give errors hundreds of times worse than well-standardized coordinates. The rapid increase in error at scales below 0.1 is caused by floating-point truncation error.

Fig. 3 illustrates the advantages of using perspective rather than affine reconstruction, for a camera driving in a 90 degree arc around a scene at various distances. Clearly, the affine approximation introduces a considerable amount of systematic error even for quite distant scenes. Projective factorization is stable and accurate even for distant scenes: even in these cases, the only real advantage of affine factorization is the fact that it is 2-3 times faster.

I have also run the point-based algorithms on several data se-

quences extracted from real images. Without the ground truth it is hard to be precise, but the final aligned reconstructions seem qualitatively accurate and in good agreement with the results obtained using synthetic data.

6 Discussion & Conclusions

Within the limitations of the factorization paradigm, factorization-based projective reconstruction seems quite successful. For points, the methods studied have proved simple, stable, and surprisingly accurate. For lines the situation is less clear: the methods work, but least-squares refinement often improves the results significantly. As with any line reconstruction, there are always outliers, especially when the cameras are collinear or coplanar.

Fixed-rank factorization works well, although (as might be expected) SVD always produces slightly more accurate results. The savings in run time over SVD probably only become significant for quite large problems (say more than 40 images and 100 points), but in these cases they can become very substantial.

This paper presents only the first few members of a large family of reconstruction techniques, based on the recovery of projective depths or scale factors. Future work will expand on this. There are analogous factorization methods using higher matching tensors, and also methods that reconstruct the projection matrices directly from matching tensors without factorization (and hence do not require tokens to be tracked through every image). All of these allow various trade-offs between redundancy, computation and implementation effort. I am also investigating numerical factorization methods that can handle missing data and incremental updates gracefully, and alternatives to Levenberg-Marquardt refinement (which I feel is not well suited to nonlinear least-squares reconstruction).

Summary: Projective structure and motion can be recovered from multiple perspective images of a scene consisting of points and lines, by estimating fundamental matrices and epipoles from the image data, using these to rescale the image measurements, and then factorizing the resulting rescaled measurement matrix using either SVD or a fast approximate factorization algorithm.

References

[1] P. Beardsley, I. Reid, A. Zisserman, and D. Murray. Active visual navigation using non-metric structure. In E. Grimson, editor, *IEEE Int. Conf. Computer Vision*, pages 58–64, Cambridge, MA, June 1995.

[2] O. Faugeras. What can be seen in three dimensions with an uncalibrated stereo rig? In G. Sandini, editor, *European Conf. Computer Vision*, Santa Margherita Ligure, Italy, May 1992. Springer-Verlag.

[3] O. Faugeras, Q.-T. Luong, and S. J. Maybank. Camera self calibration: Theory and experiments. In G. Sandini, editor, *European Conf. Computer Vision*, Santa Margherita Ligure, Italy, May 1992. Springer-Verlag.

[4] O. Faugeras and B. Mourrain. On the geometry and algebra of the point and line correspondences between n images. In E. Grimson, editor, *IEEE Int. Conf. Computer Vision*, pages 951–6, Cambridge, MA, June 1995.

[5] R. Hartley. Euclidean reconstruction from multiple views. In 2nd *Europe-U.S. Workshop on Invariance*, pages 237–56, Ponta Delgada, Azores, October 1993.

[6] R. Hartley. Lines and points in three views – an integrated approach. In *Image Understanding Workshop*, Monterey, California, November 1994.

[7] R. Hartley. In defence of the 8-point algorithm. In E. Grimson, editor, *IEEE Int. Conf. Computer Vision*, pages 1064–70, Cambridge, MA, June 1995.

[8] R. Hartley, R. Gupta, and T. Chang. Stereo from uncalibrated cameras. In *IEEE Conf. Computer Vision & Pattern Recognition*, pages 761–4, Urbana-Champaign, Illinois, 1992.

[9] P. F. McLauchlan and D. W. Murray. A unifying framework for structure and motion recovery from image sequences. In E. Grimson, editor, *IEEE Int. Conf. Computer Vision*, pages 314–20, Cambridge, MA, June 1995.

[10] R. Mohr, B. Boufama, and P. Brand. Accurate projective reconstruction. In 2nd *Europe-U.S. Workshop on Invariance*, page 257, Ponta Delgada, Azores, October 1993.

[11] C. J. Poelman and T. Kanade. A parapersective factorization method for shape and motion recovery. In J-O. Eklundh, editor, *European Conf. Computer Vision*, pages 97–108, Stockholm, 1994. Springer-Verlag.

[12] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, 1992. 2nd edition.

[13] A. Shashua. Projective structure from uncalibrated images: Structure from motion and recognition. *IEEE Trans. Pattern Analysis & Machine Intelligence*, 16(8), 1994.

[14] A. Shashua. Algebraic functions for recognition. *IEEE Trans. Pattern Analysis & Machine Intelligence*, 1995.

[15] P. Sturm and B. Triggs. A factorization based algorithm for multi-image projective structure and motion. In *European Conf. Computer Vision*, Cambridge, England, 1996. Springer-Verlag.

[16] R. Szeliski and S. B. Kang. Recovering 3d shape and motion from image streams using nonlinear least squares. In *IEEE Conf. Computer Vision & Pattern Recognition*, page 752, 1993.

[17] C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: a factorization method. *Int. J. Computer Vision*, 9(2):137–54, 1992.

[18] B. Triggs. The geometry of projective reconstruction I: Matching constraints and the joint image. Submitted to *Int. J. Computer Vision*, March 1995.

[19] B. Triggs. Matching constraints and the joint image. In E. Grimson, editor, *IEEE Int. Conf. Computer Vision*, pages 338–43, Cambridge, MA, June 1995.

[20] B. Triggs. New methods for projective reconstruction. In preparation, 1996.