Fast Bilinear SfM with Side Information

Mahesh Ramachandran, Ashok Veeraraghavan and Rama Chellappa *
Center for Automation Research
University of Maryland, College Park, MD 20742
{maheshr,vashok,rama}@cfar.umd.edu

Abstract

We study the beneficial effect of side information on the Structure from Motion (SfM) estimation problem. The side information that we consider is measurement of a 'reference vector' and distance from fixed plane perpendicular to that reference vector. Firstly, we show that in the presence of this information, the SfM equations can be rewritten similar to a bilinear form in its unknowns. Secondly, we describe a fast iterative estimation procedure to recover the structure of both stationary scenes and moving objects that capitalizes on this information. We also provide a refinement procedure in order to tackle incomplete or noisy side information. We characterize the algorithm with respect to its reconstruction accuracy, memory requirements and stability. Finally, we describe two classes of commonly occurring real-world scenarios in which this algorithm will be effective: (a) presence of a dominant ground plane in the scene and (b) presence of an inertial measurement unit on board. Experiments using both real data and rigorous simulations show the efficacy of the algorithm.

1. Introduction

Structure from Motion (SfM) refers to the task of recovering the 3D structure of a scene and the motion of a camera from a video sequence. SfM has been an active area of research since Longuet-Higgins [8] eight-point algorithm. There have been several different approaches to the SfM problem and we refer the reader to [4, 9] for a comprehensive survey of the various approaches. Nevertheless, SfM is a very hard ill-posed inverse problem and very few of these algorithms provide satisfactory performance in real-world scenarios. Therefore, recent research has focused on developing SfM algorithms in the presence of additional constraints like positional information from Global Positioning System(GPS) sensors [2].

In this paper, we study the effect of additional information, in the form of measurements of a direction vector and the height of the camera center from a plane perpendicular to this vector. This type of side information is frequently available in several real-world scenarios such as when onboard inertial measurements are available or when a dominant plane is present. Inertial sensors like the inclinometer or gravitational sensors can provide sensing of a certain direction while altimeters frequently found on UAVs can provide the required height information. For example, when there is negligible camera acceleration, an accelerometer measures the gravity and we can filter out Inertial Measurement Unit (IMU) measurements to get good estimates of the gravity vector. In the case where we do not have side information but we observe a dominant plane in the scene, we can use the homographies between multiple views to obtain estimates of the ground plane normal and height using a decomposition technique. In either case, we will show that this side information constrains the ill-posed SfM problem in such a manner that the SfM equations become similar to a bilinear form in its unknowns. We then describe a fast iterative procedure much like bilinear solvers that can robustly solve for the SfM unknowns by minimizing the reprojection error.

1.1. Prior Work

This paper is related to prior work in SfM literature that assume a visible dominant plane in the scene. An important algorithm in this class is the use of the plane-plus-parallax model for the recovery of 3D depth maps [5]. The multi-view constraints imposed by a plane were used by Rother [11] and Kaucic [6] to simplify the projective reconstruction problem into a linear system of equations. Bartoli [1] derive a linear algorithm for estimating the structure of objects moving on a plane in straight lines without rotating. Reconstruction of objects moving in an unconstrained fashion was studied in detail by Fitzgibbon and Zisserman [3].

Our algorithm assumes approximate knowledge of a certain direction vector in each image of the sequence and also the altitude from a plane perpendicular to this vector. These quantities are well defined when we observe a dominant

*Partially supported by Army Research Office MURI ARMY-W911NF0410176 under the technical monitor-ship of Dr. Tom Doligalski

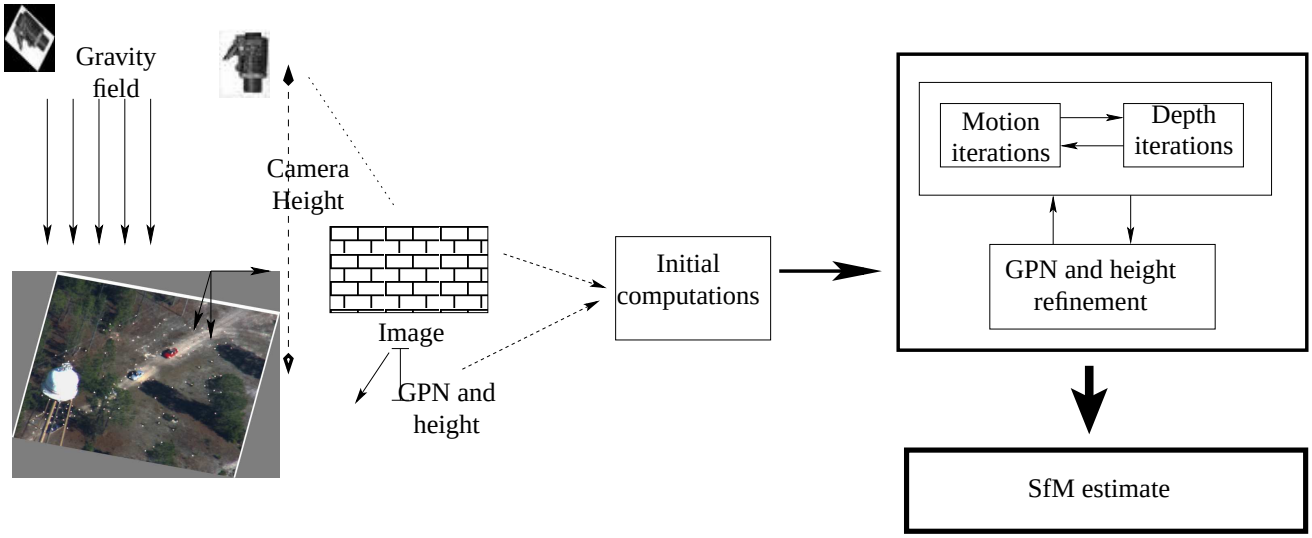


Figure 1. An illustration of the problem setting and the main computational steps. The image shows a typical scene with a ground plane and some static and moving objects on it. The gravity field is shown, which can be assumed perpendicular to the ground plane. A camera moves over the scene and gives an image sequence. We may also have measurements of the gravity vector from an IMU or sensing of the plane normal by additional means. We use these additional measurements to simplify the structure and motion computations.

plane in the scene, but the algorithm does not require the visibility of a plane (for stationary scenes). In this respect, it is quite different from all other previous approaches.

We show that using the side information, we can simplify the SfM problem and derive a fast iterative algorithm. We analyze the properties of this algorithm and show its performance with quantitative and qualitative evaluations on several real and simulated datasets. In the same framework, we provide a technique to compute the structure of moving objects on a plane.

1.2. Outline

Section 2 formulates the problem and derives the new SfM equations in the presence of additional information. In Section 3, we propose a fast iterative algorithm using the special form of the equations. Section 4 analyzes the computational complexity and memory requirements of the algorithm. Experimental results in several real scenes and quantitative comparisons with competing approaches are provided in Section 5.

2. Problem Formulation

We assume that we have measurements of a certain direction in the Camera Coordinate System (CCS) corresponding to every image in the sequence. This direction could be the gravity vector which can be sensed using inclinometers, or the normal vector to a ground plane which can be obtained using the homographies. We also assume that we have measurements or estimates of the heights of the camera along the direction of the reference vector from a plane perpendicular to the vector. We can estimate this quantity from the azimuth and the slant range of the camera, that can be measured by other sensors.

We choose a World Coordinate System (WCS) with the Z axis along the reference direction vector, and the X and Y axes perpendicular to this vector. The CCS is chosen with the Z axis along the optical camera axis and the X and Y axes along the usual image axes. The transformation between these two coordinate systems at any instant can be written as $P_w = R_{c2w}P_c + T_{c2w}$. Here, P is a point whose coordinates are represented in the WCS by P_w, and in the CCS by P_c.

A known reference vector in an unknown camera coordinate camera system fixes two degrees of freedom of the rotation matrix R_{c2w}. The unknown component is the rotation of the CCS about an axis parallel to the reference direction vector. The full rotation matrix can be shown to be split uniquely as $R_{c2w} = R_p R_g$, where R_p is the rotation along the direction vector, and R_g is along an axis perpendicular to this vector. We are now concerned with the estimation of the rotation along the direction vector (R_p), and the translations along a plane perpendicular to this vector (x and y components of T_{c2w}) in addition to the 3D locations of the world points. In the following, we refer to in-plane motion as the component of translation parallel to the $X-Y$ world plane and rotation about the Z-axis (R_p). The out-of-plane motion is the Z-axis translation (z component of T_{c2w}), and the rotation R_g that changes the reference vector orientation in the local coordinate system (CCS). We write the transformation between the WCS and the CCS as:

$$P_w^i = \begin{bmatrix} X_i \\ Y_i \\ Z_i \end{bmatrix} = R_p^{(t)} R_g^{(t)} \lambda_{ti} \begin{bmatrix} x_{ti} \\ y_{ti} \\ 1 \end{bmatrix} + T_{c2w}^{(t)} \quad (1)$$

where the camera-to-world rotation matrix has been factorized into its two components as $R_{c2w}^{(t)} = R_p^{(t)} R_g^{(t)}$. Here $T_{c2w}^{(t)} = [T_x^{(t)}, T_y^{(t)}, T_z^{(t)}]$ where $T_z^{(t)}$ is the height of the camera. $[x_i, y_i, 1]^T$ is the image feature in homogeneous coordinates, which has been normalized for the calibration matrix. From the side information, we have estimates (initial values) of $R_g^{(t)}$ and $T_z^{(t)}$. Using this information, we can rewrite (1) as:

$$P_w^i = \begin{pmatrix} \cos\theta_t & \sin\theta_t & 0 \\ -\sin\theta_t & \cos\theta_t & 0 \\ 0 & 0 & 1 \end{pmatrix} \lambda_{ti} \begin{bmatrix} u_{ti} \\ v_{ti} \\ w_{ti} \end{bmatrix} + T_{c2w}^{(t)} \quad (2)$$

where $[u_{ti}, v_{ti}, w_{ti}]^T = R_g^{(t)} [x_i, y_i, 1]^T$, and $R_g^{(t)}$ is computed from the reference vector in the side information. $R_g^{(t)}$ is the rotation matrix that rotates the reference vector expressed in the CCS to the reference vector expressed in the WCS. We rearrange (2) to get (3):

$$\lambda_{ti} \begin{bmatrix} u_{ti} \\ v_{ti} \\ w_{ti} \end{bmatrix} = \begin{pmatrix} \cos\theta_t & -\sin\theta_t & 0 \\ \sin\theta_t & \cos\theta_t & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{bmatrix} X_i - T_x^{(t)} \\ Y_i - T_y^{(t)} \\ Z_i - T_z^{(t)} \end{bmatrix} \quad (3)$$

We eliminate the projective depth λ_{ti} by taking ratios of the quantities as shown in (4)

$$\begin{bmatrix} u_{ti}/w_{ti} \\ v_{ti}/w_{ti} \end{bmatrix} = \begin{pmatrix} \cos\theta_t & -\sin\theta_t \\ \sin\theta_t & \cos\theta_t \end{pmatrix} \begin{bmatrix} \frac{X_i - T_x^{(t)}}{Z_i - T_z^{(t)}} \\ \frac{Y_i - T_y^{(t)}}{Z_i - T_z^{(t)}} \end{bmatrix} \quad (4)$$

We accumulate (4) for all the feature points in all views and write it in the factorization format as shown in (5).

$$\begin{bmatrix} \frac{u_{11}}{w_{11}} & \frac{u_{12}}{w_{12}} & \dots & \frac{u_{1n}}{w_{1n}} \\ \frac{v_{11}}{w_{11}} & \frac{v_{12}}{w_{12}} & \dots & \frac{v_{1n}}{w_{1n}} \\ \vdots & & \ddots & \dots \\ \frac{u_{m1}}{w_{m1}} & \frac{u_{m2}}{w_{m2}} & \dots & \frac{u_{mn}}{w_{mn}} \\ \frac{v_{m1}}{w_{m1}} & \frac{v_{m2}}{w_{m2}} & \dots & \frac{v_{mn}}{w_{mn}} \end{bmatrix} \begin{bmatrix} Z_1 & 0 & \dots & 0 \\ 0 & Z_2 & \dots & 0 \\ \vdots & & & \\ 0 & 0 & \dots & Z_n \end{bmatrix} -$$

$$\begin{bmatrix} T_z^{(1)} & 0 & \dots & 0 & 0 \\ 0 & T_z^{(1)} & \dots & 0 & 0 \\ \vdots & & \ddots & & \vdots \\ 0 & 0 & \dots & T_z^{(m)} & 0 \\ 0 & 0 & \dots & 0 & T_z^{(m)} \end{bmatrix} \begin{bmatrix} \frac{u_{11}}{w_{11}} & \dots & \frac{u_{1n}}{w_{1n}} \\ \frac{v_{11}}{w_{11}} & \dots & \frac{v_{1n}}{w_{1n}} \\ \vdots & \ddots & \dots \\ \frac{u_{m1}}{w_{m1}} & \dots & \frac{u_{mn}}{w_{mn}} \\ \frac{v_{m1}}{w_{m1}} & \dots & \frac{v_{mn}}{w_{mn}} \end{bmatrix}$$

$$= \begin{bmatrix} \cos\theta_1 & -\sin\theta_1 & t_x^{(1)} \\ \sin\theta_1 & \cos\theta_1 & t_y^{(1)} \\ \dots & & \\ \cos\theta_m & -\sin\theta_m & t_x^{(m)} \\ \sin\theta_m & \cos\theta_m & t_y^{(m)} \end{bmatrix} \begin{bmatrix} X_1 & \dots & X_n \\ Y_1 & \dots & Y_n \\ 1 & \dots & 1 \end{bmatrix} \quad (5)$$

In (5), m denotes the number of views, and n is the number of points. We denote the measurement matrix as A, the diagonal matrix of camera heights as $\bar{T}_z$, and the product $\bar{T}_z A = B$. We denote the diagonal matrix of heights of feature points from the ground plane as Z. The motion matrix on the right hand side is M and the shape matrix is S. We can rewrite (5) concisely as: $AZ - B = MS$. Each column of this matrix equation specifies the relation between the projections of a single point in all the views. Each pair of rows specifies the relation between the projections of all the the points in a single view. In (4), the quantities $T_x^{(t)}$ and $T_y^{(t)}$ refer to the x and y components of translation measured along the WCS axes. In (5), the variables $t_x^{(t)} = -cos\theta_t T_x^{(t)} + sin\theta_t T_y^{(t)}$ and $t_y^{(t)} = -sin\theta_t T_x^{(t)} - cos\theta_t T_y^{(t)}$ refer to the same quantities measured in the CCS axes. This change of variables is done to enable a factorization into the motion and shape matrices as shown. Note that our unknowns are the matrices $\{M, S, Z\}$ and (5) looks similar to a bilinear system in the elements of these matrices.

3. Fast Bilinear Estimation of SfM

In this section we will first consider some special cases which simplify (5) such that it is amenable to factorization or linearization. These two special cases are similar in essence to our approach and we will highlight the differences from these. Later, we will consider the general case and detail our solution which is very much like the iterative solution for solving bilinear equations and is therefore fast, accurate and stable.

3.1. Structure from Planar Motion

Let us consider a surveillance scenario where a static camera is overlooking a scene with objects (like vehicles etc.) moving on a ground plane and rotating w.r.t the plane normal. In this case, if we study (5), we notice that the matrix $\bar{T}_z$ has the constant value of the camera height h all through its diagonal and this reduces (5) to $A(Z - h) = MS$. Li and Chellappa [7] note that the measurement equation in this case is of rank-3 and show how a rank-3 factorization of the measurement matrix can be used to solve for the structure and planar motion. The algorithm cannot be directly extended to tackle non-planar motion while our FBSfM algorithm is capable of tackling arbitrary motion.

3.2. Linear Multi-View Reconstruction

Rother [11] stabilizes the images using the homographies, and chooses a projective basis where the problem becomes one of computing structure and motion of calibrated translating cameras. They derive linear equations for the camera centers and points, and solve the resulting linear system for all cameras and points simultaneously using SVD based techniques. The performance of their algorithm

is heavily dependent on the estimate of the homographies. In addition, the memory and computational requirements of the algorithm become infeasible in the case where we have a large number of frames and points.

3.3. Bilinear Algorithm for the General Case

We observe that (5) is similar to a bilinear form in the structure and in-plane motion. It is not strictly bilinear because the variables in the motion matrix are not independent (due to the presence of common cosine and sine terms in the first two columns). Our approach is to solve for the unknowns in an iterative manner, holding one set of parameters fixed while solving for the others and vice versa. We start the iterations by solving for the unknown structure.

3.3.1 Depth Iterations

For each feature point i, we pick the i^{th} column of the matrix equation (5) and write it as follows:

$$\begin{bmatrix} u_{1i}(Z_i - T_z^1)/w_{1i} \\ v_{1i}(Z_i - T_z^1)/w_{1i} \\ u_{2i}(Z_i - T_z^2)/w_{2i} \\ \vdots \\ v_{mi}(Z_i - T_z^m)/w_{mi} \end{bmatrix} = \begin{bmatrix} \cos\theta_1 & -\sin\theta_1 & t_x^1 \\ \sin\theta_1 & \cos\theta_1 & t_y^1 \\ \cdots & & \\ \cos\theta_m & -\sin\theta_m & t_x^m \\ \sin\theta_m & \cos\theta_m & t_y^m \end{bmatrix} \begin{bmatrix} X_i \\ Y_i \\ 1 \end{bmatrix} \tag{6}$$

We rearrange (6) to obtain equations of the form:

$$\begin{bmatrix} M(:,1) & M(:,2) & -A(:,i) \end{bmatrix} \begin{bmatrix} X_i \\ Y_i \\ Z_i \end{bmatrix} = -B(:,i) - M(:,3) \tag{7}$$

In (7), $M(:,1)$ and $M(:,2)$ are the first and second columns of the motion matrix, containing the cosine and sine terms. $M(:,3)$ contains the in-plane components of translation. $A(:,i)$ and $B(:,i)$ are the i^{th} columns of the matrices A and B respectively. Recall that B is the product of the heights matrix T_z, and the measurement matrix A. The above is a linear system, and we obtain a least squares solution for the unknown 3D locations.

3.3.2 Motion Iterations

We solve for the in-plane motion parameters of each view k given the 3D coordinates of the world points (obtained from the previous step). We extract the $(2k-1)^{th}$ and $(2k)^{th}$ rows from the matrix equation (5):

$$\begin{bmatrix} C(2k-1,:) \\ C(2k,:) \end{bmatrix} = \begin{bmatrix} \cos\theta_k & -\sin\theta_k & t_x^k \\ \sin\theta_k & \cos\theta_k & t_y^k \end{bmatrix} S \tag{8}$$

where, the $C = AZ - B$ denotes the left hand side of (5). We need to solve for $\{\theta_k, t_x^k, t_y^k\}$ from (8). We rearrange (8) as follows:

$$C(2k-1,:)^T = \cos\theta_k \vec{X} - \sin\theta_k \vec{Y} + t_x^k \vec{1} \tag{9}$$

$$C(2k,:)^T = \cos\theta_k \vec{Y} + \sin\theta_k \vec{X} + t_y^k \vec{1} \tag{10}$$

Here, $\vec{X}$ and $\vec{Y}$ represent the column vectors containing the X and Y coordinates of all the world points, and $\vec{1}$ is a vector of ones. First, we eliminate the translations by averaging the two sets of N equations in (9,10), and subtracting the mean from each component. This gives us new equations where the translation component has been eliminated.

$$\begin{aligned} C(2k-1,:)^T - \mu_1 &= \cos\theta_k(\vec{X} - \mu_X) - \sin\theta_k(\vec{Y} - \mu_Y) \\ C(2k,:)^T - \mu_2 &= \cos\theta_k(\vec{Y} - \mu_Y) + \sin\theta_k(\vec{X} - \mu_X) \end{aligned} \tag{11}$$

μ_1 and μ_2 are the means of $C(2k-1,:)^T$ and $C(2k,:)^T$. Equations (11) can be rewritten as:

$$\begin{bmatrix} (\vec{X} - \mu_X) & -(\vec{Y} - \mu_Y) & C(2k-1,:)^T - \mu_1 \\ (\vec{Y} - \mu_Y) & (\vec{X} - \mu_X) & C(2k,:)^T - \mu_2 \end{bmatrix} \begin{pmatrix} \cos\theta_k \\ \sin\theta_k \\ -1 \end{pmatrix} = \vec{0} \tag{12}$$

We can solve (12) using standard techniques. We find the singular vector of the system matrix with the lowest singular value, and then normalize it to make the third component as -1. Then we assign the first two components to the cosine and sine of the angle θ_k and then normalize them for consistency. After we obtain the rotation angle θ_k, the translations can be obtained as follows:

$$t_x^k = \mu_1 - \cos\theta_k \mu_X + \sin\theta_k \mu_Y \tag{13}$$

$$t_y^k = \mu_2 - \cos\theta_k \mu_Y - \sin\theta_k \mu_X \tag{14}$$

We have described a technique to iteratively estimate the 3D locations, and the in-plane components of motion. Each of the individual steps only involves solving a linear system. In practice, we iterate these two steps a certain number of times, or until a termination criterion is satisfied.

In both the depth and motion iterations, we solve linear systems that are of much smaller size compared to the size of systems that we need to solve in other techniques like bundle adjustment etc. More details on the computational requirements are given in section 4.1

3.3.3 Uncertainty in Side Information

In case the measurements of the direction vector and the camera height are not accurate, we may refine these parameters to obtain a better SfM estimate. We describe a technique to refine these quantities. From (1), we have

$$P_w^i = \begin{pmatrix} \cos\theta_t & \sin\theta_t & 0 \\ -\sin\theta_t & \cos\theta_t & 0 \\ 0 & 0 & 1 \end{pmatrix} R_g^t \lambda_{ti} \begin{bmatrix} x_{ti} \\ y_{ti} \\ 1 \end{bmatrix} + T_{cw}^{(t)} \tag{15}$$

In this step, we hold the 3D locations P_w^i and the in-plane components of motion $\{\theta_t, T_x^t, T_y^t\}$ fixed to their earlier estimates from the previous step. We perform a nonlinear refinement of the height T_z^t, and the normal vector at each frame separately. Note that this implies that at any stage, we

only refine three parameters simultaneously (in comparison to all the cameras and points for a full bundle adjustment).

We rearrange (15) by bringing the known terms on one side to obtain:

$$\begin{bmatrix} x_{ti} \\ y_{ti} \\ 1 \end{bmatrix} \times (R_g^t)^T \begin{bmatrix} X_i^{'} \\ Y_i^{'} \\ Z_i - T_z^t \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \quad (16)$$

where $X_i^{'} = \cos\theta_t X_i - \sin\theta_t Y_i$ and $Y_i^{'} = \sin\theta_t X_i + \cos\theta_t Y_i$. We refine the normal vector and height parameters by minimizing the sum of the norms of the vectors in the left hand side of (16) for all the feature points in a particular frame. We use MATLAB's implementation of the Levenberg-Marquardt algorithm in order to perform this minimization. This minimization involves only three parameters at any stage.

3.4. Motion Estimation for Moving Objects

We can incorporate the estimation of structure and motion of objects moving on a plane, in the above framework. The objects of interest can be translating and rotating on a plane, and we assume that we can measure the ground plane normal and height with respect to the same plane. We reformulate the problem into one where the object is stationary, and a separate camera is moving over each object. This approach is similar to some earlier works [12]. The world coordinate of a point on a moving object changes depending on the motion of the object, and is therefore parameterized by time along with the feature number.

$$\begin{bmatrix} X_{ti} \\ Y_{ti} \\ Z_{ti} \end{bmatrix} = \begin{pmatrix} \cos\phi_t & \sin\phi_t & 0 \\ -\sin\phi_t & \cos\phi_t & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{bmatrix} X_i \\ Y_i \\ Z_i \end{bmatrix} + \begin{bmatrix} D_x^t \\ D_y^t \\ 0 \end{bmatrix} \quad (17)$$

In (17), ϕ_t corresponds to the rotation of the object in frame t with respect to its initial orientation. (D_x^t, D_y^t) denote the displacement of the object in the WCS. The relation between image feature points and moving scene points is:

$$\begin{bmatrix} X_{ti} \\ Y_{ti} \\ Z_{ti} \end{bmatrix} = \begin{pmatrix} \cos\theta_t & \sin\theta_t & 0 \\ -\sin\theta_t & \cos\theta_t & 0 \\ 0 & 0 & 1 \end{pmatrix} \lambda_{ti} \begin{bmatrix} u_{ti} \\ v_{ti} \\ w_{ti} \end{bmatrix} + \begin{bmatrix} T_x^{(t)} \\ T_y^{(t)} \\ T_z^{(t)} \end{bmatrix} \quad (18)$$

We can rewrite (18) in the same form as for static points as follows:

$$\lambda_{ti} \begin{bmatrix} u_{ti} \\ v_{ti} \\ w_{ti} \end{bmatrix} = \begin{pmatrix} \cos\psi_t & -\sin\psi_t & 0 \\ \sin\psi_t & \cos\psi_t & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{bmatrix} X_i - C_x^t \\ Y_i - C_y^t \\ Z_i - T_z^{(t)} \end{bmatrix}$$

$$\begin{bmatrix} C_x^t \\ C_y^t \end{bmatrix} = \begin{bmatrix} \cos\phi_t & -\sin\phi_t \\ \sin\phi_t & \cos\phi_t \end{bmatrix} \begin{bmatrix} T_x^t - D_x^t \\ T_y^t - D_y^t \end{bmatrix}$$

$$\psi_t = \theta_t - \phi_t \quad (19)$$

Equations (19) say that the motion of the object can be transferred to the camera, and the object can be considered to be stationary. We can solve for (19) in the same framework as for static scene points, and the virtual camera motions can be estimated. The object motion (ϕ_t, D_x^t, D_y^t) can be solved from the real and virtual camera motions $(\theta_t, T_x^t, T_y^t, \psi_t, C_t^t, C_y^t)$ using (19).

We initialize the iterations for a moving object by assuming a fixed height for all feature points. In practice, we found that the technique was not very sensitive to this initialization in our sequences.

4. Analysis of FBSfM

4.1. Computational Complexity and Memory Requirements

The main computations in our algorithm are in the solution of the linear systems in the depth and motion iterations, and the refinement steps in the direction vector and height refinements. Suppose there are m views of n points, and that all points are visible in all views. To emphasize, this assumption is not necessary for the purposes of the algorithm but simplifies the process of quantifying the memory requirements.

As mentioned in [6], let $SVD(a,b) = 4ab^2 + 8b^3$ be the cost of carrying out a singular value decomposition of a matrix with a rows and b columns. The following are the main computational requirements:

- *Depth Iterations:* For each of the n points, this involves solving a linear system of size $2m \times 4$ which is equivalent to a total cost of $n \times SVD(2m, 4)$.
- *Motion Iterations:* For each of the m views, this involves solving a linear system of size $2n \times 3$ which is equivalent to a total cost of $m \times SVD(2n, 3)$
- *Direction vector and height refinement:* We need to update only 4 parameters and hence this requires the solution of a 4×4 linear system. For all the frames, this comes at a cost of $O(m)$ per iteration.

Totally, each iteration of FBSfM costs $200mn + 512n + 216m + O(m) \approx O(mn)$. In comparison, one full bundle adjustment takes up computations of the order of $O(nm + nm^2 + m^3)$. Rother's system involves solving a linear system of size $(3n+3m)\times(3nm)$ whose computational cost is $SVD(3n+3m, 3nm) = 108(m^3n^2 + m^2n^3) + 216m^3n^3$.

We observe that the peak computation for our individual steps increases only of the order of $O(mn)$.

4.2. Experiments on Synthetic Sequences

The results of simulation experiments described here show that FBSfM performs very consistently and advantageously over a wide range of noise levels compared to two other well-known techniques that we have compared with i.e. Rother's solution [11] and Bundle Adjustment (BA).

For FBSfM, the additional information comes from estimates of plane-normal vectors estimated from homographies. [10]. Random points were generated, some of which were chosen to lie on a fixed plane which made possible the estimates of these homographies. Random noise added to the feature points leads to noisy estimates of homographies and therefore noisy additional information.

Figs. (2a,2b) illustrate the mean reprojection error, and the deviation of these reconstruction estimates at a range of noise levels (with precise camera calibration). Over a wide range of noise levels, we see that FBSfM performs consistently and competitively. At very low noise levels, Rother's performs slightly better than FBSfM. But at realistic and high pixel noise levels, FBSfM seems to perform the best.

The results are similar in the case of high calibration errors (around 20% error in the focal lengths and camera center estimates). Note that erroneous calibration leads to erroneous initial estimates of the side information from the homographies [10]. The mean reprojection error plots (Fig. 2c) are similar, and the deviations of the estimates are lowest for FBSfM (Fig. 2d). The experiment was repeated for 70 trials under scenarios with different pixel noise levels and calibration errors as shown in Fig. (2e,2f, 2g,2h).

We observe that in different settings, both BA and Rother's give bad solutions in many cases. BA is considered the gold standard in SfM algorithms because it minimizes the reprojection error starting with a good initial solution. But it involves a highly nonlinear iterative refinement and is prone to getting stuck at local minima. The mean reprojection error of BA varies widely indicating that it may be affected by local minima unlike FBSfM which is a lot more consistent.

5. Experiments on Real Datasets

In this section we detail the experiments we performed to study the efficacy of the FBSfM algorithm. The algorithm was tested on several real datasets under both conditions - a) Scenes with a dominant plane and b) Camera augmented with inertial sensors.

5.1. Scenes with Dominant Plane

When a dominant plane is present in the scene, we can use the plane normal and the height from the plane as side information in our algorithm. We can accumulate the homographies induced by the plane from multiple views and use a decomposition technique [10] to compute the plane normals and the heights. The inter-image homographies are robustly estimated using RANSAC.

5.2. Experiments on an Indoor Handheld Sequence

We report reconstruction results on an indoor image sequence taken with a digital camera of a toy car resting on a plane. SIFT features were extracted and matched across the image sequence. Inter-image homographies were estimated robustly using RANSAC, and planar points were separated from those off the plane. The reprojection errors for the three techniques FBSfM, Rothers' and BA is shown in Table 1. Figure 3 illustrates the texture mapped 3D model of the car. This is an example of a short video sequence, with the number of keyframes being 13 and the number of feature points around 60.

	Reprojection error
FBSfM	2.79
Rother	3.40
BA	4.10

Table 1. Mean reprojection error of the reconstructions of FBSfM, BA, and Rother's for the indoor toycar sequence.

Figure 3. This figure illustrates a texture mapped 3D model of the car obtained by interpolating from the sparse structure estimates generated by FBSfM. Manually assisted feature point matches were also used to generate this result, to ensure display a full 3D model

5.3. SfM on Static Points in Aerial Video - 1

The algorithm was tested on a long real aerial video sequence. The metadata associated with the sequence made available the normal vector and the camera heights from additional sensors on-board. This information was computed from the measurements of the slant range and the azimuth, elevation and twist of the camera (which was part of the metadata). Around 1700 feature points were tracked through the sequence, which was 225 frames long. Around 30 keyframes were selected, which were the frames for which timestamped metadata was available. This sequence is a dense reconstruction problem because a large percentage of the feature points is observed in each view. Two versions of the algorithm were compared, one with the initial GPN and Tz estimates obtained from the homographies, and another with these estimates obtained from the metadata.

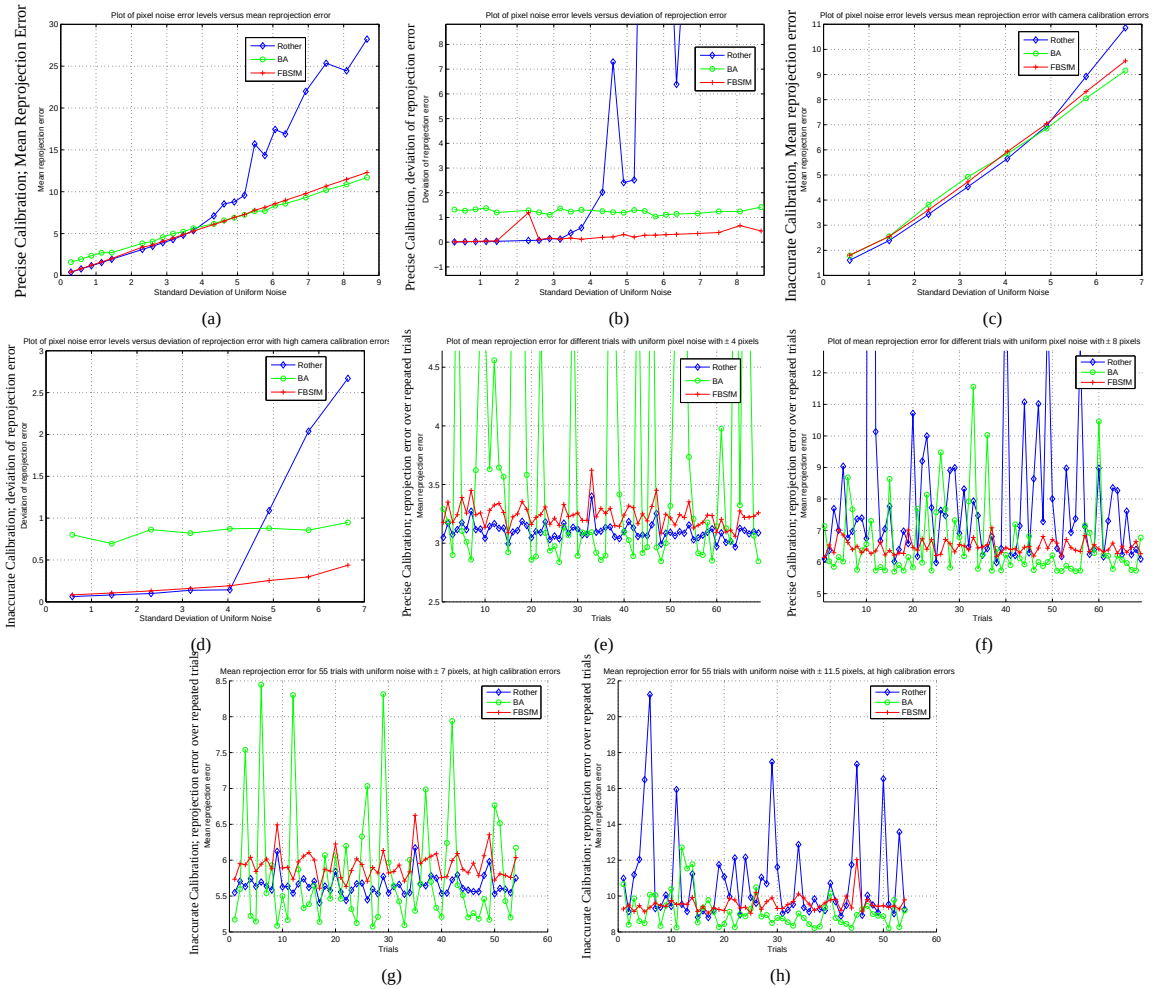


Figure 2. Plots of statistics of mean reprojection error of the reconstructions on synthetic data at various noise levels. Fig. (2a) and (2b) show the mean and variance of the estimates for the three techniques FBSfM, Rother's and Bundle Adjustment at various noise levels with precise calibration. Fig. (2c) and (2d) illustrate the same results but with a large error in calibration. Fig. (2e) and (2f) show the reprojection error for the three techniques, for 70 trials at uniformly distributed noise levels of $\pm$ 4 and $\pm$ 8 pixels respectively and for the precise calibration case. Fig. (2g) and (2h) illustrate the same results as above but with a large calibration error.

From table 2, we infer that FBSfM produces the best reconstruction when used with the metadata. With the homography-based estimates of GPN, the reprojection error was worse, and this was to a largely due to a high reprojection error in one particular frame. We feel this is because of a bad initial estimate of the GPNs and heights. The homography decomposition step is sensitive to the calibration matrix (although our algorithm is not very sensitive). Rother's requires the solution of a very large matrix equation which is highly time and memory consuming. Hence we could perform a reconstruction for only a few of the points. Bundle adjustment requires the inversion of a large Jacobian matrix. Our technique performs faster than these techniques because at any iteration, the peak memory requirement is very limited. From Fig. 4 we infer that most features are reconstructed with low error. Some features are reconstructed with a high error, but these feature tracks are erroneous outliers (mismatched correspondences etc.).

FBSfM with metadata	FBSfM	Rother	BA
2.83	10.5	24.32	5.56

Table 2. Mean reprojection error for the long aerial video sequence in which we had a lot of frames and features.

5.4. SfM on Moving Objects in Aerial Video - 2

We report experiments on reconstruction of moving objects on a planar scene. The theoretical derivations suggest that the moving object is assumed stationary, and there is a separate camera moving over it. We did not compute the inter-image homographies, but obtained the GPN and

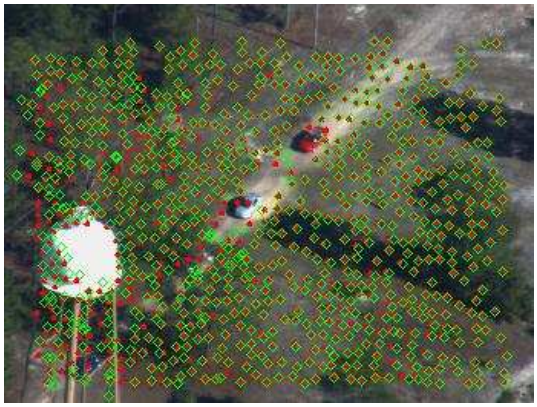
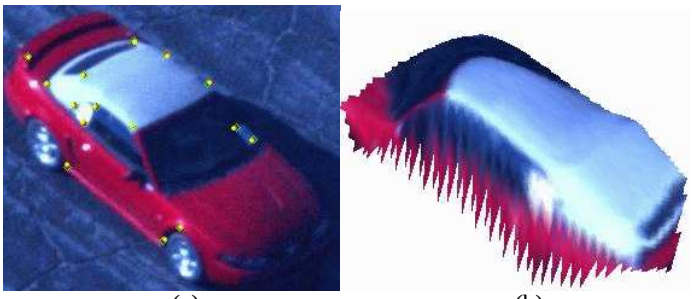


Figure 4. Detected and reprojected features in a single frame of the long aerial video sequence



(a) (b)

Figure 5. Fig (5a) shows a snapshot of a moving car in the video sequence, with the detected features shown in yellow dots. It also shows the reprojected features shown as green squares. Fig. (5b) shows the partially reconstructed 3D model of the car.

height from the metadata, and estimate the structure and in-plane motion using the bilinear algorithm without direction vector or height refinement. Fig. (5a) shows a snapshot of a moving vehicle with detected and reprojected features shown in yellow dots and green squares respectively. Fig. (5b) shows the reconstructed 3D-model. The reprojection error was around 0.7 pixels. This experiment illustrates how we can use our technique to perform SfM on objects moving on a plane, when these objects are rotating and translating on the plane, in a general way. Earlier techniques [1, 3] assume constrained object motion in order to compute the structure.

6. Conclusion

We have studied the effect of side information such as the height of the camera and the ground plane normal on the SfM estimation problem. We show that in the presence of this information, the problem can be solved easily using an iterative procedure much like bilinear solvers. Our solution does not need any joint non-linear iterative minimization and is therefore quicker than many existing techniques. We also discussed extensions in order to tackle multiple moving objects. We provide extensive experimental results to highlight the efficacy and robustness of the algorithm.

Acknowledgments

The first author wishes to thank Dr. Carlos Morimoto and Aswin Sankaranarayanan for discussions.

References

[1] A. Bartoli. The geometry of dynamic scenes - on coplanar and convergent linear motions embedded in 3d static scenes. *Computer Vision and Image Understanding*, 98(2), May.

[2] R. Carceroni, A. Kumar, and K. Daniilidis. Structure from motion with known camera positions. In *Proc. of IEEE Intl. Conf. on Computer Vision and Pattern Recognition*, pages 477–484, 2006.

[3] A. Fitzgibbon and A. Zisserman. Multibody structure and motion: 3-d reconstruction of independently moving objects. In *Proc. European Conf. on Computer Vision*, volume 98.

[4] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition, 2004.

[5] M. Irani, P. Anandan, and M. Cohen. Direct recovery of planar-parallax from multiple frames. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 24:1528–1534, 2002.

[6] R. Kaucic, R. Hartley, and N. Dano. Plane-based projective reconstruction. In *Proc. Intl. Conf. on Computer Vision*, volume 1, pages 420–427, 2002.

[7] J. Li and R. Chellappa. Structure from planar motion. *IEEE Trans. on Image Processing*, 12(1):234–778, November 2006.

[8] H. C. Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections. *Nature*, 293(1):133–135, 1981.

[9] Y. Ma, S. Soatto, J. Kosecka, and S. Sastry. *An Invitation to 3-D Vision*. Springer Verlag, ISBN: 0387008934, second edition, 2003.

[10] E. Malis and R. Cipolla. Camera calibration from unknown planar structures enforcing the multi view constraints between collineations. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 24(9):1268–1272, 2002.

[11] C. Rother and S. Carlsson. Linear multi view reconstruction and camera recovery using a reference plane. *International Journal Computer Vision*, 49(3):117–141, 2002.

[12] C. Yuan and G. Medioni. 3d reconstruction of background and objects moving on ground plane viewed from a moving camera. In *Proc. of Intl. Conf. on Computer Vision and Pattern Recognition*, 2006.