MITSUBISHI ELECTRIC RESEARCH LABORATORIES
http://www.merl.com

A Fast Bilinear Structure from Motion Algorithm Using a Video Sequence and Inertial Sensors

Ramachandran, M.; Veeraraghavan, A.; Chellappa, R.

TR2011-042 January 2011

Abstract

In this paper, we study the benefits of the availability of a specific form of additional information, the vertical direction (gravity) and the height of the camera, both of which can be conveniently measured using inertial sensors and a monocular video sequence for 3D urban modeling. We show that in the presence of this information, the SfM equations can be rewritten in a bilinear form. This allows us to derive a fast, robust, and scalable SfM algorithm for large scale applications. The SfM algorithm developed in this paper is experimentally demonstrated to have favorable properties compared to the sparse bundle adjustment algorithm. We provide experimental evidence indicating that the proposed algorithm converges in many cases to solutions with lower error than state-of-art implementations of bundle adjustment. We also demonstrate that for the case of large reconstruction problems, the proposed algorithm takes lesser time to reach its solution compared to bundle adjustment. We also present SfM results using our algorithm on the Google Street View research data set.

IEEE Transactions on Pattern Analysis and Machine Intelligence

A Fast Bilinear Structure from Motion Algorithm using a Video Sequence and Inertial Sensors

Mahesh Ramachandran, *Student Member, IEEE,* Ashok Veeraraghavan, *Member, IEEE,* and Rama Chellappa, *Fellow, IEEE*

***Abstract*— In this paper, we study the benefits of the availability of a specific form of additional information - the vertical direction (gravity) and the height of the camera both of which can be conveniently measured using inertial sensors, and a monocular video sequence for 3D urban modeling. We show that in the presence of this information, the SfM equations can be rewritten in a bilinear form. This allows us to derive a fast, robust, and scalable SfM algorithm for large scale applications. The SfM algorithm developed in this paper is experimentally demonstrated to have favorable properties compared to the sparse bundle adjustment algorithm. We provide experimental evidence indicating that the proposed algorithm converges in many cases to solutions with lower error than state-of-art implementations of bundle adjustment. We also demonstrate that for the case of large reconstruction problems, the proposed algorithm takes lesser time to reach its solution compared to bundle adjustment. We also present SfM results using our algorithm on the Google StreetView research dataset.**

***Index Terms*— Structure from Motion, Multiple View Geometry, Computer Vision.**

I. 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 [1] eight-point algorithm. There have been several different approaches to the SfM problem that are surveyed in [2]. The problem has gained interest because of exciting new applications like 3D urban modeling, terrain estimation from UAVs etc. Robust and scalable SfM algorithms would enable automatic building of 3D models of urban scenes from such image sequences [3].

Many image data collections have metadata containing measurements from additional sensors such as inertial sensors, global positioning systems etc. ~~These additional measurements may be used to develop new SfM algorithms to process the data collections~~. We consider the problem of SfM estimation in the presence of a specific form of measurements that are frequently available, and propose a fast, scalable and robust SfM algorithm.

We assume that ~~we have~~ measurements of the gravity vector and the height in the camera coordinate system along with every ~~image~~. These quantities can be accurately measured using inertial sensors. Under negligible camera acceleration, an accelerometer measures the gravity that can be used in a complementary filter [4] along with gyro measurements to get good estimates of the gravity vector. In the absence of these sensors, we can use the homographies induced by a world plane to derive the additional measurements [5]. Using these measurements, we show that the SfM equations reduce to a bilinear form in its unknowns. This leads to an iterative minimization algorithm that is experimentally demonstrated to converge faster from a wide class of initial solutions ~~with low error in the vertical direction and height components~~.

This work was partially supported by Army Research Office MURI ARMY-W911NF0410176 under the technical monitorship of Dr. Tom Doligalski. Mahesh Ramachandran was with the Center for Automation Research, University of Maryland, College Park, MD 20742 USA during the time of this work. He is now with Qualcomm Inc., San Diego, CA 92121 USA. (e-mail: maheshr@umiacs.umd.edu). Ashok Veeraraghavan is with Mitsubishi Electric Research Labs, Cambridge, MA 021389 USA. (e-mail: veerarag@merl.com). Rama Chellappa is with Center for Automation Research and Department of Electrical and Computer Engineering, University of Maryland, College Park, MD 20742 USA. (e-mail: rama@cfar.umd.edu).

A. Literature survey on SfM

SfM algorithms can be broadly classified into the following categories: batch techniques, minimal solutions and recursive frameworks. A comprehensive survey may be found in [2]. The proposed algorithm falls in the class of batch techniques which jointly solve for the views of all the cameras and the structure of all the points. Bundle adjustment (BA) [6] is the representative algorithm in this class and it minimizes the cumulative reprojection error of all points in all the images. This is a non-linear least squares minimization problem that is commonly solved using the Levenberg-Marquardt (LM) [7] algorithm. The ~~linear~~ system corresponding to the normal equations for LM is intractable for large reconstruction problems. To handle large problems, the sparse structure of the Hessian matrix is used for efficiently solving the normal equations resulting in the Sparse Bundle Adjustment (SBA) [8] algorithm. The conjugate gradient (CG) method [7] is another choice for optimizing the reprojection error that does not require the solution of a large system; however suitable pre-conditioners are necessary for it to work well [9]. In addition, its benefits have not yet been clearly demonstrated for large scale problems.

In spite of these advances, the current computational resources are still hard-pressed to handle very large reconstruction problems, and the performance is not satisfactory enough for real-world deployment. Therefore, recent research has focused on developing SfM algorithms in the presence of additional constraints. For instance, position information about the cameras from Global Positioning Systems (GPS) can help us solve for the parameters [10]. If inertial measurements from IMUs are available, we can reduce the ambiguities in the SfM problem [11].

B. Related Work

This paper is related to a class of alternation algorithms that solve for the structure and motion in an iterative fashion [12], [13]. These approaches solve for the projective depths along with the motion and structure matrices and result in a projective reconstruction. The projective depths are typically initialized to unity and later refined ~~iteratively~~. This has been reported to work well only in special settings where the depth of each feature point remains approximately the same throughout the sequence [2]. This does not cover many important scenarios such as roadside urban sequences or aerial videos where the altitude of the camera varies a lot. Our algorithm makes use of the bilinear form in the Euclidean frame without slack variables. Hence it does not have any restrictions on its use except that the gravity and height measurements must be available.

Oliensis and Hartley [14] point out theoretically the potential for convergence to trivial solutions for this class of algorithms. They show that [12] and [13] are unstable because they converge to nonsensical results and question their usage (even for a few iterations) as initialization techniques for the bundle adjustment algorithm. In contrast, a solution from our algorithm with low error corresponds to a non-trivial reconstruction that is good in practice. Buchanan [15] evaluated several alternation, first-order and second-order approaches for matrix factorization with applications to SfM, and compared the algorithms based on their convergence-rate and error function value at the final solution. Their key result was that for general reconstruction problems, Newton-based descent algorithms performed better than alternation approaches. We provide strong experimental evidence that with the assumed additional information, our alternation approach converge faster than other descent algorithms.

C. Contributions

The contributions of the paper are as follows.

1) ~~We~~ propose a robust and scalable SfM algorithm ~~using additional measurements~~ that is bilinear in the Euclidean frame.
2) We describe simulation results demonstrating that the proposed algorithm leads to solutions with lower error than SBA and takes ~~l~~ower time for convergence.
3) We ~~describe competitive~~ reconstruction results on the Google StreetView research dataset.

We are *not* proposing our algorithm as a substitute for the already successful BA algorithm. We emphasize that BA using LM currently remains the best known algorithm for SfM when the initial solution is favorable. We seek to provide an alternative to BA and similar approaches when the initial solution is likely to be bad (such as when it is obtained from GPS and IMU measurements with outliers). Based on the results of [15], the proposed algorithm can be used along with BA to result in a hybrid algorithm that performs better than both of them separately, starting from poor initial solutions.

II. Problem Formulation

We choose a World Coordinate System (WCS) with the Z axis along the vertical direction, and the X and Y axes are perpendicular to this axis. If a ground plane is present in the scene, the Z axis becomes the normal vector to the plane, and the X and Y axes are on the plane. The Camera Coordinate System (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 $P_w = R_{c2w}P_c + T_{c2w}$. 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 CCS 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 vector. The full rotation matrix can be shown to be split uniquely as $R_{c2w} = R_pR_g$, where R_p is the rotation along the reference vector, and R_g is along an axis perpendicular to this vector. We are now concerned with the estimation of the rotation along the reference 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 CCS.

We write the transformation between the WCS and the CCS as

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

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. P_w^i denotes the coordinates of the i^{th} point in ~~the~~ WCS. From the additional measurements, we have estimates of $R_g^{(t)}$ and $T_z^{(t)}$. Using this information, we can rewrite (1) as

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

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. $R_g^{(t)}$ is the rotation matrix that transforms the reference vector from ~~the~~ CCS to ~~the~~ WCS. We rearrange (2) to obtain (3) which relates the coordinates of the feature point in frame t ~~and feature i~~ to ~~the~~ world coordinates and ~~the~~ camera positions.

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

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

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

We rearrange (4) by multiplying both sides by $(Z_i - T_z)$ to obtain (5)

$$\begin{bmatrix} u_{ti}/w_{ti} \\ v_{ti}/w_{ti} \end{bmatrix} \cdot (Z_i - T_z^{(t)}) = \begin{pmatrix} \cos\theta_t & -\sin\theta_t \\ \sin\theta_t & \cos\theta_t \end{pmatrix} \begin{bmatrix} X_i - T_x^{(t)} \\ Y_i - T_y^{(t)} \end{bmatrix} \tag{5}$$

Assume that we have n feature points that are observed in m frames. We accumulate (5) for all the feature points in all views and write them as shown in (6). We denote the measurement matrix as A, the diagonal matrix of camera heights as $\bar{T}_z$, and the product $\bar{T}_z A = B$. The diagonal matrix of heights of feature points from the ground plane is Z. The motion matrix on the right hand side is M and the shape matrix is S. We can rewrite (6) 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 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 in ~~the~~ WCS. In (6), 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 in CCS. This change of variables is done to enable a factorization into the motion and shape matrices as shown. Our unknowns are the matrices $\{M, S, Z\}$ and (6) is bilinear. We solve for the unknown parameters (M, S, Z) by minimizing the Frobenius norm of the difference between the matrices on each side of (6). The cost function is written as

$$E = \|A \cdot Z - \bar{T}_z A - M \cdot S\|^2 \tag{7}$$

where $\|\cdot\|^2$ denotes the Frobenius norm. Since feature points may not be observed in all frames, some entries of the measurement matrix are unknown. Unlike factorization-based approaches, our algorithm can cope with missing measurements.

$$\begin{bmatrix} \frac{u_{11}}{w_{11}} & \cdots & \frac{u_{1n}}{w_{1n}} \\ \frac{v_{11}}{w_{11}} & \cdots & \frac{v_{1n}}{w_{1n}} \\ \vdots & \ddots & \vdots \\ \frac{u_{m1}}{w_{m1}} & \cdots & \frac{u_{mn}}{w_{mn}} \\ \frac{v_{m1}}{w_{m1}} & \cdots & \frac{v_{mn}}{w_{mn}} \end{bmatrix} \begin{bmatrix} Z_1 & 0 & \cdots & 0 \\ 0 & Z_2 & \cdots & 0 \\ \vdots & & \ddots & \\ 0 & 0 & \cdots & Z_n \end{bmatrix} - \begin{bmatrix} T_z^{(1)} & 0 & \cdots & 0 \\ 0 & T_z^{(1)} & \cdots & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \cdots & T_z^{(m)} \end{bmatrix} \begin{bmatrix} \frac{u_{11}}{w_{11}} & \cdots & \frac{u_{1n}}{w_{1n}} \\ \frac{v_{11}}{w_{11}} & \cdots & \frac{v_{1n}}{w_{1n}} \\ \vdots & \ddots & \vdots \\ \frac{u_{m1}}{w_{m1}} & \cdots & \frac{u_{mn}}{w_{mn}} \\ \frac{v_{m1}}{w_{m1}} & \cdots & \frac{v_{mn}}{w_{mn}} \end{bmatrix} = \begin{bmatrix} \cos\theta_1 & -\sin\theta_1 & t_x^{(1)} \\ \sin\theta_1 & \cos\theta_1 & t_y^{(1)} \\ \cdots & & \\ \cos\theta_m & -\sin\theta_m & t_x^{(m)} \\ \sin\theta_m & \cos\theta_m & t_y^{(m)} \end{bmatrix} \begin{bmatrix} X_1 & \cdots & X_n \\ Y_1 & \cdots & Y_n \\ 1 & \cdots & 1 \end{bmatrix} \tag{6}$$

III. Fast Bilinear Estimation of SfM

We solve for the unknowns by minimizing the cost function $E = \|A \cdot Z - B - M \cdot S\|^2$. We can choose from several first-order and second-order algorithms for this minimization [15]. We present an alternation-style technique to solve for the unknowns, by switching between iterations where (1) the motion parameters are kept fixed and structure parameters are estimated, with iterations where (2) the structure is fixed and motion parameters are estimated.

A. Structure Iterations

We rewrite the cost function (7) as a sum of terms corresponding to each feature point $j \in (1, \cdots, n)$ as follows: $E = \sum_{j=1}^{n} E_j^d$, where E_j^d corresponds to the cost function for the j^{th} point, and the superscript d is used to note that the total error is split into terms for each point. We pick the j^{th} column of (6) to obtain

$$E_j^d = \left\| \begin{bmatrix} u_{1j}(Z_j - T_z^1)/w_{1j} \\ v_{1j}(Z_j - T_z^1)/w_{1j} \\ u_{2j}(Z_j - T_z^2)/w_{2j} \\ \vdots \\ v_{mj}(Z_j - T_z^m)/w_{mj} \end{bmatrix} - M \cdot \begin{bmatrix} X_j \\ Y_j \\ 1 \end{bmatrix} \right\|^2 \tag{8}$$

We rewrite (8) to obtain

$$E_j^d = \left\| [M(:,1) \quad M(:,2) \quad -A(:,i)] \begin{bmatrix} X_j \\ Y_j \\ Z_j \end{bmatrix} + (B(:,j) + M(:,3)) \right\|^2 \tag{9}$$

where $M(:,1)$ 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. $A(:,j)$ and $B(:,j)$ are the j^{th} columns of ~~the~~ matrices A and B respectively. We minimize (9) w.r.t (X_j, Y_j, Z_j) at each structure iteration. The cost function is a linear system in (X_j, Y_j, Z_j) and the minimum is obtained by linear least squares.

$$[M(:,1) \quad M(:,2) \quad -A(:,j)] \begin{bmatrix} X_j \\ Y_j \\ Z_j \end{bmatrix} = -B(:,j) - M(:,3) \tag{10}$$

B. Motion Iterations

We rewrite (7) as a sum of terms corresponding to each frame as $E = \sum_{i=1}^{m} E_i^m$, where E_i^m corresponds to cost function of the i^{th} frame, and the superscript m is used to note that the total error is split into terms corresponding to ~~the~~ motion parameters of each frame. We extract the $(2i-1)^{th}$ and $(2i)^{th}$ rows from (6) to obtain

$$\begin{aligned} E_i^m &= \left\| C(2i-1,:) - \begin{bmatrix} \cos\theta_i & -\sin\theta_i & t_x^i \end{bmatrix} \cdot S \right\|^2 \\ &\quad + \left\| C(2i,:) - \begin{bmatrix} \sin\theta_i & \cos\theta_i & t_y^i \end{bmatrix} \cdot S \right\|^2 \end{aligned} \tag{11}$$

where, the $C = AZ - B$ denotes the left hand side of (6). We minimizing (11) to solve for $\{\theta_i, t_x^i, t_y^i\}$. We set the derivative of E_i^m w.r.t (t_x^i, t_y^i) to zero to obtain

$$t_x^i = \frac{1}{n}\sum_{k=1}^{n} C(2i-1,k) - \frac{1}{n}\sum_{k=1}^{n} X_k cos\theta_i + \frac{1}{n}\sum_{k=1}^{n} Y_k sin\theta_i \tag{12}$$

$$t_y^i = \frac{1}{n}\sum_{k=1}^{n} C(2i,k) - \frac{1}{n}\sum_{k=1}^{n} X_k sin\theta_i - \frac{1}{n}\sum_{k=1}^{n} Y_k cos\theta_i \tag{13}$$

Denoting μ_{2i-1} and μ_{2i} as the means of $C(2i-1,:)$ and $C(2i,:)$ respectively, and μ_X and μ_Y as the means of the X and Y coordinates of points, we write the solution of t_x^i and t_y^i as

$$t_x^i = \mu_{2i-1} - \mu_X \cos\theta_i + \mu_Y \sin\theta_i \tag{14}$$

$$t_y^i = \mu_{2i} - \mu_Y \cos\theta_i - \mu_X \sin\theta_i \tag{15}$$

We substitute the solution (15) in (11) and obtain

$$\begin{aligned} E_i^m &= \left\| \vec{X}^o \cos\theta_i - \vec{Y}^o \sin\theta_i - C^o(2i-1,:)^T \right\|^2 \\ &\quad + \left\| \vec{X}^o \sin\theta_i + \vec{Y}^o \cos\theta_i - C^o(2i,:)^T \right\|^2 \end{aligned} \tag{16}$$

where $\vec{X}$ and $\vec{Y}$ denote column vectors containing the X and Y coordinates of all the points, and $\vec{X}^o = \vec{X} - \mu_X$, $\vec{Y}^o = \vec{Y} - \mu_Y$, $C^o(2i-1,:) = C(2i-1,:) - \mu_{2i-1}$ and $C^o(2i,:) = C(2i,:) - \mu_{2i}$. We set the derivative w.r.t θ_i to zero and simplify to obtain

$$\begin{aligned} &\sum_{k=1}^{n} C^o(2i-1,k) \cdot \left(X_k^o \sin\theta_i + Y_k^o \cos\theta_i \right) + \\ &\sum_{k=1}^{n} C^o(2i,k) \cdot \left(-X_k^o \cos\theta_i + Y_k^o \sin\theta_i \right) = 0 \end{aligned} \tag{17}$$

We can simplify (17) to obtain

$$\tan\theta_i = \frac{C^o(2i,:) \cdot \vec{X}^o - C^o(2i-1,:) \cdot \vec{Y}^o}{C^o(2i-1,:) \cdot \vec{X}^o + C^o(2i,:) \cdot \vec{Y}^o} \tag{18}$$

We obtain two possible solutions for θ_i from (18). One of them is a point of local maxima and the other is a point of local minima. We choose the solution that corresponds to the local minima. The motion and structure iterations are repeated successively until a termination criterion is satisfied. The iterations are terminated if the decrease in the error function expressed as a percentage drops below a threshold or if the iteration count reaches a maximum, whichever is earlier.

C. Out-of-plane motion refinement iterations

We minimize (7) by refining the out-of-plane components of motion for each camera. We rewrite (5) as

$$\begin{bmatrix} u_{ti}/w_{ti} \\ v_{ti}/w_{ti} \end{bmatrix} \cdot (Z_i - T_z^{(t)}) = \begin{pmatrix} \cos\theta_t & -\sin\theta_t \\ \sin\theta_t & \cos\theta_t \end{pmatrix} \begin{bmatrix} X_i - T_x^{(t)} \\ Y_i - T_y^{(t)} \end{bmatrix} \tag{19}$$

where $[u_{ti}, v_{ti}, w_{ti}]^T = R_g^{(t)}[x_i, y_i, 1]^T$. We fix P_w^i and $\{\theta_t, T_x^t, T_y^t\}$ to the current estimates, and minimize the following error function

obtained from (19)

$$E_{side} = \sum_{t=1}^{m} \left\| \begin{bmatrix} u_{ti}/w_{ti} \\ v_{ti}/w_{ti} \end{bmatrix} \cdot (Z_i - T_z^{(t)}) - \begin{pmatrix} \cos\theta_t & -\sin\theta_t \\ \sin\theta_t & \cos\theta_t \end{pmatrix} \begin{bmatrix} X_i - T_x^{(t)} \\ Y_i - T_y^{(t)} \end{bmatrix} \right\|^2 \quad (20)$$

to solve for the height T_z^t and the ground plane normal vector at each frame. We use the LM algorithm [7] for solving this non-linear least squares problem. This minimization involves only three parameters at any stage, two for the out-of-plane motion and one for the camera height. In later sections we refer to the proposed method as FBSfM which is an acronym for Fast Bilinear Structure from Motion.

IV. Analysis

A. Computational Complexity and Memory Requirements

Suppose there are m views of n points with all points visible in all views. 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 [16]. The main computational requirements of the proposed algorithm are the following.

- *Depth Iterations:* For each point, this involves solving a linear system of size $2m \times 4$ which is equivalent to a total cost of $n \times SVD(2m, 4)$, where n is the number of points.
- *Motion Iterations:* For each view, this involves performing the computations in (18, 15) which is equivalent to $4n$ multiplications and $6n$ additions and is therefore $O(n)$ in computational cost. This accumulates to a total cost of $m \times O(n)$, where m is the number of views.
- *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 $m \times SVD(4,4) = 768m$ per iteration.

Totally, each iteration of FBSfM has a computational cost of $O(mn)$. In comparison, an SBA iteration has complexity $O(nm + nm^2 + m^3)$. Linear multiview reconstruction [17] involves solving a linear system of size $(3n + 3m) \times (3nm)$ whose computational cost is $SVD(3n + 3m, 3nm) = 108(m^3n^2 + m^2n^3) + 216m^3n^3$.

B. Discussion

The FBSfM algorithm has the flavor of iterative methods for projective structure from motion [12], [13]. Suppose we have n fixed points $P_1, P_2, \cdots, P_n$ observed by m cameras, we can write the projection of the j^{th} point on the i^{th} camera as $p_{ij} = \frac{1}{z_{ij}} M_i P_j$, where M_i denotes the 3×4 projective matrix associated with the i^{th} camera, and z_{ij} denotes the projective depth associated with the j^{th} point in the i^{th} camera. In a typical BA algorithm, we minimize the reprojection error which is $E = \sum_{i,j} |p_{ij} - \frac{1}{z_{ij}} M_i P_j|^2$ and solve for M_i and P_j. The objective function is highly non-linear and the results are dependent on the availability of good initialization points [6]. Hence, many authors consider the simplified (but related) objective function $E_{lin} = \sum_{i,j} |z_{ij} p_{ij} - M_i P_j|^2$. The proposed algorithm is similarly derived by scaling (4) by the $(Z_i - T_z^{(t)})$. The difference is that unlike earlier techniques, we use the additional information to leverage the bilinear form in the Euclidean frame, without using slack variables. Simulation results suggest clear advantages in speed and accuracy of the proposed algorithm when the additional information has low error.

Issue of trivial minima: Oliensis [14] pointed that existing projective bilinear alternation techniques [12], [13] converge to "trivial" solutions with nonsensical structure and motion estimates. In all our experiments, we have observed that the iterations always converge to meaningful results.

Convergence in error value: If the initial error is $E^{(0)}$ and the error after structure iterations is $E^{(1)}$, then $E^{(1)} \leq E^{(0)}$. If the error after motion iterations is $E^{(2)}$, then $E^{(2)} \leq E^{(1)}$. If $E^{(3)}$ is the error after out-of-plane iterations, then $E^{(3)} \leq E^{(2)}$. Each iteration finds solutions for the variables that decreases the error, hence the error is non-increasing. Since the error is lower-bounded by 0, the successive error values converge to a value E^*.

Convergence rate and speed of convergence: Iterations of low computational complexity do not imply faster convergence rate or lower time-to-convergence. The proposed algorithm falls within the class of alternation techniques which are susceptible to flatlining and are slower than second-order newton methods in the average case [15]. ~~We claim~~ based on strong experimental evidence that the proposed approach surprisingly violates this conventional wisdom, for the *specific* case of large-sized problems starting from a *specific* class of initial solutions with low out-of-plane motion error and high levels of in-plane motion error.

V. Simulations

A. Implementation

For BA, we use the SBA solver code [8] implemented in C. This uses LM minimization for non-linear least squares. The normal equations are solved by using the Schur complement to factor out the structure parameters and the resulting system is solved using LU decomposition ~~for the~~ update ~~to~~ the camera parameters. The proposed algorithm was also implemented in C with a MATLAB interface. The motion-structure alternation iterations and out-of-plane motion refinement iterations were written in C and the interface to switch between the two sets of iterations was written in MATLAB with system calls to the corresponding C executables. Routines from OpenCV were used to solve the linear systems corresponding to the motion iterations. The LM implementation in C [18] was used for the non-linear least squares minimization corresponding to the out-of-plane motion refinement iterations. The Conjugate gradient (CG) implementation in C [19] was used in the experiments to compare with alternation for minimizing the bilinear system (7). Computation time was measured using our implementation of a nanosecond timer. The reported times include the total time taken to execute all the operations within each C implementation except disk I/O operations.

Reconstruction problem generation: A reconstruction problem was synthesized by generating N points uniformly distributed within the cube specified by $-20 \leq X \leq 20$, $-20 \leq Y \leq 20$ and $10 \leq Z \leq 40$. The coordinates of the camera locations were uniformly distributed in the cube specified by $-25 \leq X \leq 25$, $-25 \leq Y \leq 25$, $55 \leq Z \leq 105$. The choice of dimensions is for illustration, and in practice the dimensions were scaled according to convenience. The orientations of the cameras were chosen by first generating the directions of the principal camera axes and then choosing the rotation angle of the camera around this axis. The principal camera axis was chosen by generating a random point in the $X - Y$ plane that specifies the point of intersection of the principal axis with this plane. The rotation angle of the camera axis around the principal axis was randomly chosen between 0 to 2π. This scheme for generating points and cameras ensured a wide variation in the camera matrices and point locations, to mitigate any potential bias in the reported results to the choice of reconstruction problem. Image feature points were obtained by reprojecting the 3D points on the cameras based on perspective projection. The parameters of the camera were: focal length $= 320$, principal point $= (320, 240)$ and image size $= 640 \times 480$. Moderate Gaussian noise was added to the reprojected image feature locations to simulate feature detection errors.

Height error	Ground plane vector error				
	0°	3°	6°	9°	12°
0%	0	0	0	1.2	4
3%	0.2	0.4	0.6	0.6	6
5.2%	8.6	8.8	8.4	9	12.8
13%	17.6	18.8	18.8	17.8	20
26%	19.8	21.4	22.2	20.6	19.4
39%	20.6	26.6	23.4	26.6	21.4
52.1%	27.4	29.8	27.8	30.4	23.2

TABLE I

PERCENTAGE OF RUNS ON WHICH THE SOLUTION OF ALTERNATION WAS HIGHER THAN THE MINIMUM OF THE SOLUTIONS OF CG AND LM.

SfM Initialization: Initial motion estimates are obtained by perturbing the ground truth motion. The in-plane and out-of-plane motion components are perturbed separately, with high errors in in-plane components and low errors in out-of-plane motion components. The x and y locations are displaced by a vector that is oriented in a random direction in the XY plane, and whose length is a specified fraction of the maximum dimension of the reconstruction. For example, a 70% error in the in-plane translation means that the camera center was displaced by a vector of length 0.7×50 in the XY plane. A 60° error in the in-plane rotation angle θ means that θ was perturbed in either the clockwise or counterclockwise direction by 60°. A 10% error in the out-of-plane translation means that each camera center was moved either in the positive or negative Z direction by 0.1×50. The ground plane normal is perturbed by adding a random vector to it such that the new vector makes a pre-specified angle with the original vector. The initial values for the 3D coordinates of feature points are obtained by triangulating each point using ~~the~~ image feature locations and ~~the~~ initial camera motion parameters, after solving the linear system of a structure iteration in (10)). All algorithms are initialized with the same initial solution and executed on ~~the~~ identical machines under identical loading conditions ~~when reporting comparative results~~.

B. Comparative evaluation of bilinear alternation

We compare the performance of Bilinear alternation, CG and LM for minimizing the objective function in (7). The alternation approach chosen is the motion-structure iterations as described in the paper, where (7) is minimized with respect to the in-plane motion parameters only. We analyze the results of the minimization using the three approaches for various levels of out-of-plane motion error. Conventional wisdom suggests that second-order newton methods perform best for the matrix factorization problem [15]; however the results of this experiment for the case of low out-of-plane motion error are surprising because they suggest that bilinear alternation performs fastest under the assumed operating conditions.

A reconstruction problem is obtained by generating 50 random world points and 10 cameras as described above, with 96.4% of the image feature measurements known. The ground plane normal vector at each camera location is perturbed such that the new vector makes angles of $(0^\circ, 3^\circ, 6^\circ, 9^\circ, 12^\circ)$ with the ground-truth vector. The camera centers are perturbed along the Z-axis of the WCS by errors of $(0\%, 3\%, 5.2\%, 13\%, 26\%, 39\%, 52.1\%)$. For each choice of errors in out-of-plane motion components, we compare the performance of the three algorithms on 500 runs. The in-plane motion components are perturbed by a 100% error in the $X-Y$ locations and a 90° error in the in-plane rotation angle to obtain the starting point for the three approaches. The total number of runs of the minimization for all cases was 17500.

The percentage of runs for which bilinear alternation performs suboptimally compared to LM or CG (i.e. with higher error than the best of CG or LM) is listed in Table I for each choice of out-of-plane motion errors. This suggests that for moderate levels of errors in the out-of-plane motion, alternation turns out to be a suboptimal choice of algorithm in roughly 17% of cases. At low to moderate errors where we propose the use of our algorithm, alternation turns out to be suboptimal in roughly 9% of cases. In practice, since we switch between in-plane and out-of-plane iterations, the effect of suboptimality in 9% of the cases does not adversely affect the structure and motion reconstruction.

We compared the number of iterations and times taken for each algorithm to reach convergence, where the error function decreases by less than 0.00001%. We find that alternation, CG and LM take an average time of 0.0257, 0.0550 and 2.8875 seconds respectively. In other words, CG takes 2.1424 times longer, and LM takes 112.5413 times longer than alternation to converge. The corresponding averaged number of iterations is 48.4882, 316.8153, and 96.5334 (although number of iterations do not directly compare). ~~The large~~ time taken by LM may be attributed to the large Hessian that must be inverted at each iteration. Using a sparse implementation of LM (similar to SBA), we may be able to reduce the computation time. However, we did not implement sparse LM since in a later experiment we demonstrate that the overall FBSfM algorithm took lesser time-to-convergence compared to the SBA implementation. Figure 1 shows convergence plots of the three algorithms for $(13\%, 6^\circ)$ error in out-of-plane motion. The red curves corresponding to bilinear alternation are largely below the blue CG and black LM curves. In this plot, a lower curve implies a faster convergence rate. The plots show the same trend for other choices of error levels. The strong

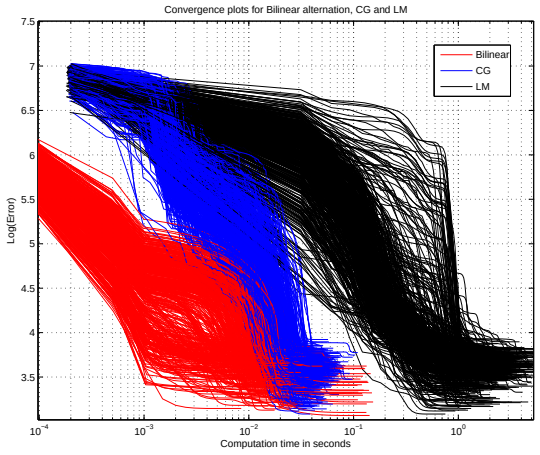

Fig. 1. This figure illustrates convergence curves plotting the Log of error versus computation time for the minimization of (7), using bilinear alternation, CG and LM. The plots for bilinear alternation are superior to the other two in terms of convergence rate because the bundle of red curves are largely below the blue and black curves. The reconstruction problem used for these plots involved 10 cameras and 50 features. We used a perturbation of $(3\%, 6^\circ)$ for the out-of-plane translation and ground plane normal angle error respectively.

advantage in computation time along with the fact that alternation converges to the same solution as CG and LM under low out-of-plane noise levels justifies its use under the operating conditions of the paper (low error in out-of-plane motion).

C. Comparison of FBSfM with SBA

We generated a reconstruction problem with 10 cameras and 50 feature points and 96% of the image measurements known. Two sets of initial camera motion solutions were generated for different choices of error levels in motion components. Set 1 had

perturbations of $(12\%, 25^\circ, 2.7\%, 2^\circ)$ in the in-plane translation, in-plane angle, vertical position and ground plane normal vector respectively. Set 2 had perturbations of $(20\%, 35^\circ, 0.5\%, 5^\circ)$. Each set had 900 different initial solutions. The structure initializations were obtained by triangulating each 3D point using the image feature locations and initial camera parameters, after solving the linear system (10) ~~of a~~ structure iteration.

In set 1, minimum error reconstruction produced by SBA was 0.3199 and that produced by our algorithm was 0.3570. In 86% of the runs, the SBA reconstruction error was lower than the minimum error produced by our algorithm. The slightly higher error of our algorithm on successful runs compared to SBA is because we minimize the algebraic as opposed to the reprojection error. On the remaining 14% of the runs, SBA produced an error of above 5. In comparison, our algorithm produced an error lower than 0.4 on 99% of the runs. This illustrates the reliability of our algorithm in producing good reconstructions when initialized from a number of random initial solutions. In set 2, 38% of the runs had a lower reprojection error for SBA than for FBSfM. The distribution of reprojection errors of both algorithms on each of the sets is illustrated in figure 2. The red curves are for set 1 and the blue curves for set 2. The figure plots the cumulative distributions of the reprojection error, hence a higher curve implies a better performing algorithm. In both sets, the curve for FBSfM is largely above that for SBA indicating better performance.

We repeated the experiments on a reconstruction problem with 45 cameras and 250 feature points, with 96.4% measurements known. We obtained 900 initial starting points by perturbing with motion error of $(30\%, 25^\circ, 3.75\%, 4^\circ)$. The initial reprojection errors ranged from 199.4611 to 3476.6 and the SBA code reported failures in minimization on 70.6% of the runs. On the remaining 29.4% of the runs on which SBA succeeded, our algorithm produced reconstructions whose final reprojection error ranged from 0.6277 to 0.7603. SBA produced an error of 0.3723 on 87.5% of the succeeded runs, and errors ranging from 9.0526 to 278.6074 on the remaining 12.5% of the succeeded runs. On the 70.6% of the original runs on which SBA reported failures, our algorithm produced reconstructions ranging from 0.6209 to 0.7848, whereas the errors of SBA ranged from 208.28 to $1.0993e+05$ (which were very close to the initial errors). These experimental results clearly demonstrate the advantage of our algorithm over SBA, because it generally avoids getting stuck in poor local minima and is more consistent in its results.

D. Comparison of convergence rates of FBSfM and SBA

We compare the convergence rates of our algorithm with SBA on a reconstruction problem with 300 cameras and 350 feature points, with 62% of the image measurements known. We generated 350 initial solutions by perturbing the ground truth motion with a 3.33% error in the in-plane translation, a 1% error in out-of-plane translation, a 15° error in the in-plane rotation angle and a 4° error in the ground plane vector. SBA converged successfully in 157 of the 350 runs, among which 138 converged to the global minimum, with a reprojection error of 1.1316. On these 138 runs, FBSfM converged to solutions with errors ranging from 1.1427 to 1.1432. Convergence of FBSfM was declared if the number of iterations exceeded 100 or if the error decreased by less than $5e-5$ or 0.0001% whichever was higher. We attribute the slightly higher reprojection error of FBSfM to the fact that it minimizes the algebraic error as opposed to the actual reprojection error. Since the final errors of FBSfM are very close to those of SBA, we believe this is very close to the global optimum. For the 138 runs described earlier, the number of iterations taken by FBSfM ranged from 93 to 102 with an average of 96.6 iterations and the total computation time ranged from 203.3 to 254.2 seconds with an average of 223.6 seconds. It is possible that because of the termination conditions, FBSfM had not converged to its global optimum in a strict sense but had exhibited flatlining beyond 100 iterations. However, for practical purposes, this is not consequential since a minimum reprojection error solution can be realized by directly minimizing the reprojection error using SBA. Accumulated over all the runs, the amount of time taken for the out-of-plane motion refinement iterations was 49.5% of the total time spent. This fraction ranged from 46% to 55.4% for all the runs.

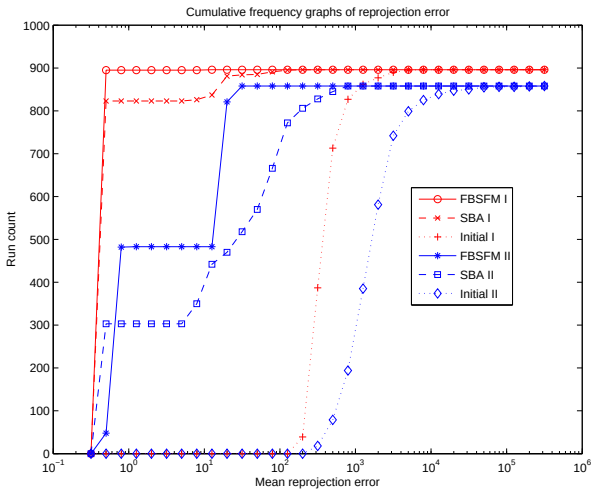

Fig. 2. This figure shows the cumulative frequency graphs of the reprojection error for SBA and FBSfM along with the error distribution of the initial solutions. The ~~problem involved~~ 10 cameras and 50 feature points. The red curves show the results for set 1 with perturbations of $(12\%, 25^\circ, 2.7\%, 2^\circ)$ in the in-plane translation, in-plane rotaion angle, out-of-plane translation and ground plane normal angle error respectively. The blue curves show the results for set 2 with initial motion error of $(20\%, 35^\circ, 0.5\%, 5^\circ)$. In both sets, the graph for FBSfM is above that of SBA indicating better performance.

Sparse BA required iterations ranging from 45 to 4639 for convergence, with an average number of iterations of 484.3. Convergence was declared if the norm of the update to the parameter vector was less than $1E-12$. The total computation time taken ranged from 157 seconds to 17864 seconds with 132 of the 138 runs requiring computation time larger than the maximum time taken by FBSfM on all the runs. Figure 3 shows the convergence plots for FBSfM (plotted in blue) and SBA (plotted in red). SBA exhibits fast convergence when the solution is very close to the global minimum. On the other hand, FBSfM exhibits slow convergence close to the minimum. Although FBSfM is faster than SBA overall, these two algorithms are good candidates to be used together in a hybrid approach. It must be noted that this advantage in computation time is larger as the problem size increases (corresponding to the size of the matrix that needs to be inverted). We have observed that for 300 cameras, FBSfM is much faster than SBA, and for 10 cameras, SBA is faster. The critical problem size above which the reduced Hessian matrix inversion or the slowness of gradient descent starts to become a disadvantage for SBA depends on the machine and the processing resources available. However, for mobile devices, we expect the memory limitation to be the bottleneck where FBSfM would find the most advantageous use (with a low critical problem size).

E. A note on alternation methods

A popular alternative to SBA that falls within the class of alternation algorithms is resection-intersection [6]. This involves

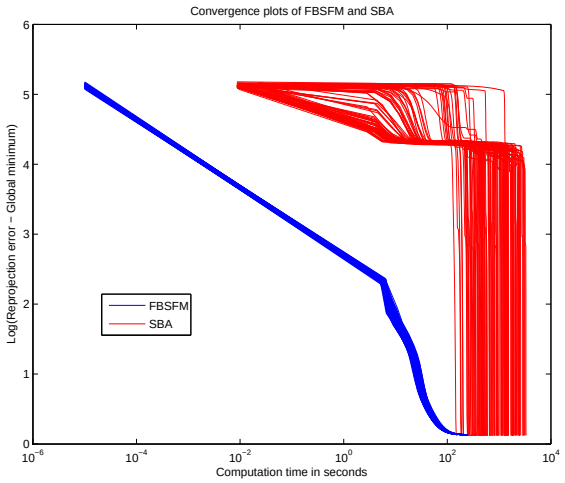

Fig. 3. This figure illustrates the convergence curves plotting the Log of the reprojection error versus computation time for FBSfM and SBA on 138 runs. The blue curves are for FBSfM and the red ones are for SBA. Note that the blue curves are clearly below the red ones indicating that FBSfM converges faster compared to SBA in this experiment.

iterations where (1) the cameras are fixed and structure variables are updated, and (2) the structure is fixed and camera matrices are updated. Each iteration is of low complexity similar to our bilinear alternation, but it involves non-linear minimization in all the iterations. The key ~~difference~~ of our work is the use of the additional measurements for the decomposition of the motion parameters that leads to a better performing algorithm for low out-of-plane noise conditions. Our experimental results do not show a clear advantage over either SBA or resection-intersection from a general class of initial solutions. However, the advantage shows up very clearly when the out-of-plane motion is known (with low error) through sensor measurements. To the best of our knowledge, we do not know of a previously proposed variant of resection-intersection for the specific setting of the paper. Earlier works [6] have evaluated the most general variant of resection-intersection and have found it to be suboptimal compared to SBA in terms of accuracy. Based on this study, we conclude that the cumulative frequency graphs of resection-intersection are expected to be below that of SBA.

VI. Experiments

A. SfM on StreetView data

The Google StreetView Research dataset consists of roadside image sequences and metadata with the locations and orientations of the camera corresponding to each image, solved using GPS and IMU measurements deployed onboard. The metadata ~~contained~~ the additional measurements in the form as required by the proposed algorithm. We chose a segment of the dataset containing 150 images when the car is moving on a single road. We obtained feature tracks by SIFT feature detection and matching, and used RANSAC and the epipolar constraint to prune out outliers. After these post-processing steps, we obtained 3145 distinct feature tracks in the sequence.

We obtained an initial solution for the camera motion by assuming a straight line motion and placing the camera centers equally distributed on the line. We initialized the out-of-plane camera rotation matrices using the direction vectors obtained from the metadata, and the in-plane rotations were all fixed to zero. The heights of the cameras were later fixed to the measurements obtained from the metadata. This provided an initial solution for the camera motion. We used this initial motion and the feature trajectories to triangulate the 3D points using structure iteration equations (10). The triangulated $3D$ locations were used as the initial structure solution. Both FBSfM and SBA were initialized with the same starting point. The initial reprojection error was 237.49. FBSfM converged to a solution with an error of 1.99 and SBA converged to a solution of 111.88. Figure 4 illustrates the top view of the reconstructed 3D points and the camera path corresponding to the FBSfM solution. When we initialized SBA with the final solution of FBSfM, the reprojection error reduced to 1.342.

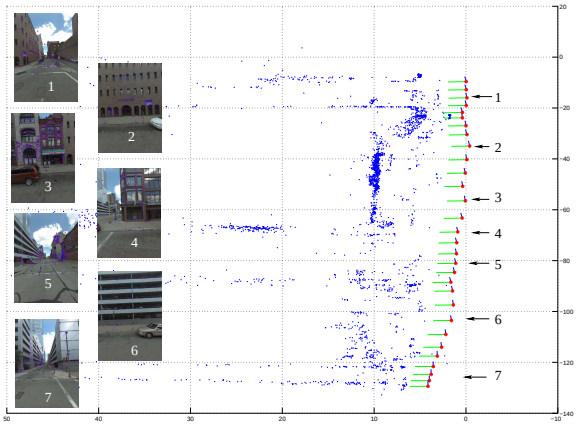

Fig. 4. This figure shows the top view of the reconstructed 3D points and the camera path obtained by solving for structure and motion from 150 images of the streetview sequence using the proposed algorithm. The red points show the camera centers and the green lines show the optical axis at each camera location. A few images in the sequence are shown in the left of the figure. We can clearly distinguish the three intersecting roads in the reconstruction, with the road in the middle approximately twice as wide as the other two roads.

Since we do not currently use sparse representations for measurement matrices, our implementation was not suited to execute multiple trials on the earlier sequence for measuring computation times. Hence we select a subsequence with lesser number of feature points. From the SfM solution with error 1.342, we selected the first 70 frames, and 755 points with reprojection error less than 3 for all frames. We used SBA for the 755 point subsequence and obtained an SfM reconstruction with error of 0.4013. We repeated the camera configuration of the 70 cameras twice by translating in the Y and Z directions by chosen distances. This produced a total of 210 cameras. We generated the feature points on the virtual cameras by reprojecting the original 3D points on each of the images and adding random noise to the pixel locations, to simulate feature detection errors. We use this SfM problem with 210 cameras, 755 points with 10% of the measurement matrix known, as the test problem for measuring computation times. The ground truth reprojection error (global optimum) for this problem was 0.3892. We generated initial solutions by perturbing the in-plane translation by 11%, out-of-plane translation by 1%, in-plane rotation angle by 7° and direction vector by 1°. Among the 15 runs of SBA that terminated successfully, the final reprojection errors of two best runs were 1.104 and 1.311, and the rest of the errors were all above 20. In contrast, the final errors of FBSfM ranged from 0.3906 to 0.3948 with the motion and structure reconstructions very close to the ground truth. The average time taken by the proposed algorithm to reach a solution was 45.957 seconds. Convergence of our algorithm was declared if the iteration count reached 100 or if the reprojection error decreased by less than $5e-5$. All trials used 100 iterations. The algorithm

probably exhibit flatlining behavior beyond 100 iterations since the error was decreasing very slowly. However, beyond this point, SBA is a better choice for minimizing the reprojection error and can be used in a hybrid approach along with FBSfM. We do not estimate the average time taken by SBA since none of the runs converged close to the global optimum. However, the time taken for the two best runs were 136.7 and 138.5 seconds. Figure 5 plots the convergence curves for both algorithms. The bundle of blue curves for FBSfM are clearly below the red ones of SBA indicating a faster overall convergence rate. The curves suggest that towards the start of the minimization, FBSfM converges faster but when the solution is close to the minima, SBA converges faster. These results illustrate

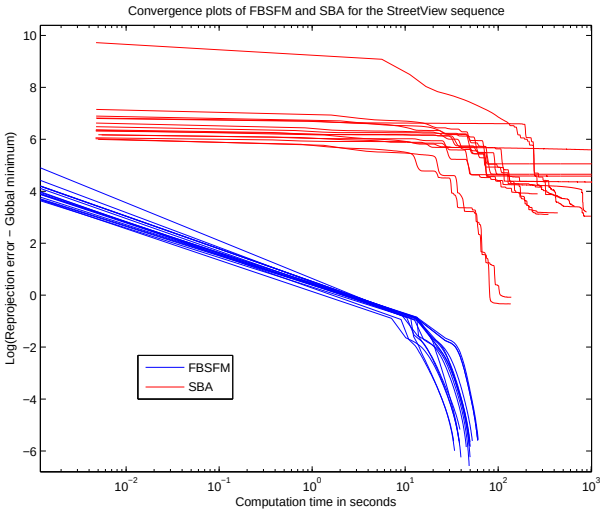

Fig. 5. This figure illustrates the convergence curves plotting the Log of the reprojection error versus computation time for FBSfM and SBA. The blue curves are for FBSfM and the red ones are for SBA. The blue curves are clearly below the red ones indicating that FBSfM converges faster compared to SBA in this experiment.

the computational advantages of FBSfM over SBA and make it a good candidate for urban modeling which involves solving large reconstruction problems.

VII. Conclusions

We discussed the importance of exploiting the available inertial measurements in the SfM estimation framework. We described a fast, robust and scalable SfM algorithm that leverages additional measurements for computing the scene structure and camera motion from a sequence of images. We described how this algorithm tackles the needs of scalability and speed required in current and future SfM applications where we work with very large datasets. We show that with the availability of measurements of the gravity vector and camera height, the SfM problem can be simplified into a bilinear form and solved using a fast scalable iterative procedure. The following are the lessons learned from the experiments.

1) FBSFM produces better solutions than SBA from initial solutions with low out-of-plane motion error. We have compared the performance of both for as much as 5° in ground plane vector error and 5% in the camera vertical position error.
2) The total time taken by FBSFM to attain convergence (with termination conditions set by a maximum number of iterations or minimum decrease in error) is less than that of SBA for large-sized problems when initialized from starting points with low error in out-of-plane motion.
3) When initialized from a large number of random starting points, FBSFM seems to converge to solutions that are lower than those produced by SBA (as is demonstrated by the cumulative frequency graphs).

Based on the above findings, our algorithm seems to be a better choice of reconstruction algorithms in many practical scenarios where it is possible to obtain the additional measurements as required by accurate sensing devices. The scalability of the algorithm also lends itself useful for large scale practical problems.

Acknowledgment

The authors would like to thank Dr. Aswin Sankaranarayanan, Dr. Carlos Morimoto and Prof. Andre Tits for their comments and discussions during the early development of this work. ~~This work was partially supported by ARL MURI ARMY-W911NF0410176 under the technical monitorship of Dr. Tom Doligalski.~~

References

[1] H. C. Longuet-Higgins, "A computer algorithm for reconstructing a scene from two projections," *Nature*, vol. 293, no. 1, pp. 133–135, 1981.
[2] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge University Press, 2004.
[3] K. Ni, D. Steedly, and F. Dellaert, "Out-of-core bundle adjustment for large-scale 3D reconstruction," in *Proc. Int'l Conf. Computer Vision, Rio de Janeiro, Brazil*, Oct. 2007.
[4] M. Euston, P. Coote, R. Mahony, J. Kim, and T. Hamel, "A complementary filter for attitude estimation of a fixed-wing UAV," in *Proc. IEEE Int'l Conf. Robots and Systems, Nice, France*, Sept. 2008.
[5] E. Malis and R. Cipolla, "Camera calibration from unknown planar structures enforcing the multi view constraints between collineations," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 24, pp. 1268–1272, Sept. 2002.
[6] B. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon, "Bundle adjustment - a modern synthesis," in *Lecture Notes in Computer Science*, vol. 1883. London, UK: Springer-Verlag, 2000.
[7] K. Madsen, H. Nielsen, and O. Tingleff, *Methods for Non-Linear Least Squares Problems*, 2nd ed. Technical University of Denmark, 2004.
[8] M. I. A. Lourakis and A. A. Argyros, "SBA: A software package for generic sparse bundle adjustment," *ACM Trans. Math. Software*, vol. 36, pp. 1–30, Jan. 2009.
[9] M. Byrod and K. Astrom, "Bundle adjustment using conjugate gradients with multiscale preconditioning," in *Proc. British Machine Vision Conference, London, UK*, Sept. 2009.
[10] R. Carceroni, A. Kumar, and K. Daniilidis, "Structure from motion with known camera positions," in *Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition, Washington DC*, vol. 1, June ~~2006~~.
[11] G. Qian, Q. Zheng, and R. Chellappa, "Reduction of inherent ambiguities in structure from motion problem using inertial data," in *Proc. Int'l Conf. Image Processing, Vancouver, Canada*, vol. 1, Sept. 2000.
[12] P. Sturm and B. Triggs, "A factorization-based algorithm for multi-image projective structure and motion," *Proc. European Conf. Computer Vision, Cambridge, UK*, vol. 1, pp. 709–720, Apr. 1996.
[13] S. Mahamud, M. Hebert, Y. Omori, and J. Ponce, "Provably-convergent iterative methods for projective structure from motion," *Proc. IEEE Conf. Computer Vision and Pattern Recognition, Honolulu, Hawaii*, vol. 1, Dec. 2001.
[14] J. Oliensis and R. Hartley, "Iterative extensions of the sturm/triggs algorithm: Convergence and nonconvergence," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 29, pp. 2217–2233, Dec. 2007.
[15] A. Buchanan and A. Fitzgibbon, "Damped Newton algorithms for matrix factorization with missing data," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition, San Diego, CA*, vol. 2, June 2005.
[16] R. Kaucic, R. Hartley, and N. Dano, "Plane-based projective reconstruction," in *Proc. Int'l Conf. Computer Vision, Vancouver, Cananda*, vol. 1, July 2001.
[17] C. Rother and S. Carlsson, "Linear multi view reconstruction and camera recovery using a reference plane," *Int'l Journal Computer Vision*, vol. 49, no. 3, pp. 117–141, Sept. 2002.
[18] M. Lourakis, "Levmar: Levenberg-Marquardt nonlinear least squares algorithms in C/C++," http://www.ics.forth.gr/∼lourakis/levmar/, Jul. 2004.
[19] W. W. Hager and H. Zhang, "CG DESCENT: A conjugate gradient method with guaranteed descent," *ACM Trans. Math. Software*, vol. 32, pp. 113–137, Mar. 2006.