Title

Sequential Reconstruction Segment-Wise Feature Track and Structure Updating Based on Parallax Paths

Authors

Hess-Flores, Mauricio
Duchaineau, Mark A.
Joy, Kenneth I.

Publication Date

2012

Peer reviewed

Sequential Reconstruction Segment-Wise Feature Track and Structure Updating Based on Parallax Paths

Mauricio Hess-Flores[1], Mark A. Duchaineau[2], and Kenneth I. Joy[1]

[1] Institute for Data Analysis and Visualization, University of California, Davis, USA
mhessf@ucdavis.edu, kenneth.i.joy@gmail.com
[2] Lawrence Livermore National Laboratory, Livermore, CA, USA**
duchaine@google.com

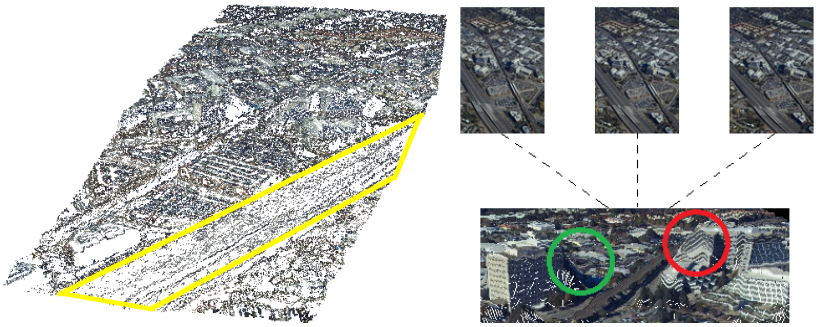
Abstract. This paper presents a novel method for multi-view sequential scene reconstruction scenarios such as in aerial video, that exploits the constraints imposed by the path of a moving camera to allow for a new way of detecting and correcting inaccuracies in the feature tracking and structure computation processes. The main contribution of this paper is to show that for short, planar segments of a continuous camera trajectory, parallax movement corresponding to a viewed scene point should ideally form a scaled and translated version of this trajectory when projected onto a parallel plane. This creates two constraints, which differ from those of standard factorization, that allow for the detection and correction of inaccurate feature tracks and to improve scene structure. Results are shown for real and synthetic aerial video and turntable sequences, where the proposed method was shown to correct outlier tracks, detect and correct tracking drift, and allow for a novel improvement of scene structure, additionally resulting in an improved convergence for bundle adjustment optimization.

1 Introduction

During the past years there has been a surge in the amount of work dealing with multi-view reconstruction of scenes, for example in applications such as robotics, surveillance and virtual reality. For the reconstruction of general scenes, state-of-the-art algorithms [1, 2] provide very accurate feature tracking, camera poses and scene structure, based mainly on sparse feature detection and matching, such as with the SIFT algorithm [3] and others inspired by its concept. However, one specific scenario for reconstruction is aerial video. Accurate models developed from aerial video can form a base for large-scale multi-sensor networks that support activities in detection, surveillance, tracking, registration, terrain modelling and ultimately semantic scene analysis. Time-effective, accurate and in some cases dense scene models are needed for such purposes.

In such cases, image acquisition is sequential and camera movement is smooth, such that it can be modelled as planar by segments, and in general is parallel to the dominant plane of the scene. Additionally, intrinsic parameters such as focal length and principal point remain constant. These are usually assumed known, and generally extrinsic parameters such as instantaneous position and orientation are at least roughly known due

** This author is now at Google, Inc.



(a) Dense reconstruction (b) Close-up view of errors

Fig. 1. Reconstruction (*a* and *b*) of the *Walnut Creek* dataset (sample images in *b*), based on dense correspondences. Examples of inaccuracies in the computed structure due to occlusions (green), repetitive patterns (red) and texture-less regions (yellow) are highlighted.

to GPS and IMU readings, respectively. In the case calibration data is not available, it can usually be estimated by making use of feature tracks across the images, but then relies on the accuracy of such tracks. Even if camera parameters are known, and are accurate, the accuracy of the final multi-view sequential reconstruction still relies fundamentally on accurate feature tracking. Due to varying lighting conditions, occlusions, repetitive patterns and other issues, feature tracks may not be perfect and this skews subsequent calibration and structure estimation. Inaccuracies may remain even after applying robust estimation procedures and outlier detection, such as with RANSAC [4]. An example of the effect of such errors on a reconstruction based on dense feature tracking is shown in Fig. 1, which reveals regions that were reconstructed inaccurately due to tracking errors. Furthermore, due to the lack of ground-truth information for the cameras and/or the structure, reconstruction algorithms usually resort to non-linear optimization of parameters to reduce total reprojection error, which is the most meaningful geometric measure of accuracy in the lack of ground-truth. However, this can be the most expensive element in a reconstruction pipeline, despite efficient sparse implementations [5]. Furthermore, such *bundle adjustment* requires a good enough starting point close to the global minimum for convergence.

The main contribution of this paper is to present a novel technique for improving feature track accuracy as an intermediate step in sequential multi-view scene reconstruction, for applications such as in aerial video. It is based on using the camera path as a source of strong and additional constraints in feature tracking, such that tracks that do not meet the constraints can be corrected, resulting in an improved scene structure. This introduces more and different constraints with respect to the classical factorization approach [6]. As will be shown, such improvements are evidenced by a reduction in total reprojection error and an improvement in the convergence of bundle adjustment. Additionally, it will be shown how the same framework allows for a simplified structure computation, which is less expensive than traditional linear triangulation [7].

The algorithm treats reconstruction of a sequential video sequence, such as aerial or turntable, as a set of segment-wise, sliding window-type connected set of smaller

reconstructions. For a given *segment*, beginning at its *anchor frame*, feature tracks and camera poses (if not available from GPS/IMU) are first computed. Then, rays from the segment's camera centers and through all computed feature track positions are intersected with a plane that lies parallel to the best-fit plane for the segment's cameras. As will be discussed, taking such a sample 'slice' of ray space makes it possible to infer the overall harmony of all feature tracks in consensus, where it becomes easy to identify where there may be specific, individual errors. The power behind this algorithm is that it uses *consensus information* from all tracks and the camera path to introduce additional, strong constraints into feature tracking. If the cameras or at least some of the initial feature tracks are accurate, inaccurate tracks can be corrected to comply with the consensus parallax movement defined by the cameras and accurate tracks. Scene structure for the segment can then be computed through a very simple procedure. Such segment-wise track computation and immediate evaluation can be used as the building block for sequential reconstruction under more general camera motions, as many can be well-approximated by planar motions over small segments. The cleaning up of tracking inaccuracies at each step allows for stable sequential reconstructions, where errors are not allowed to accumulate over time, which also has a positive effect on bundle adjustment, whose convergence is improved with more accurate tracks and structure as input.

An overview of general and sequential multi-view reconstruction is provided in Section 2. An introduction to the concept of parallax paths is provided in Section 3. The application of parallax paths to improve feature tracking and structure computation is detailed in Section 4, followed by results (Section 5) and conclusions (Section 6).

2 Related Work

For scene reconstruction, the input is a set of images and in some cases camera calibration information, while the output is typically a 3D point cloud along with color and/or normal information, representing scene structure. Camera parameters include intrinsic parameters, such as focal length, skew and principal point, as well as extrinsic or pose parameters of absolute position and orientation, and radial distortion. Intrinsics and extrinsics can be encapsulated in 3×4 projection matrices for each camera [7]. For estimating the epipolar geometry [7] between views, camera calibration and scene structure, most algorithms make use of feature tracks between images. Software packages such as *Bundler* [1] are capable of estimating tracks and all parameters from a set of images. This and other algorithms are based on SIFT feature detection and tracking [3], but there are a number of other sparse and dense methods in the literature. Dense tracking assigns a correspondence in a destination image to each source image position, and can be computed through a variety of methods [8], such as optical flow. Dense approaches especially suffer from issues such as occlusions, repetitive patterns, texture-less regions and illumination changes, which dramatically affect the quality of the tracks and reconstruction. An overview of different pose estimation methods based on feature tracking are given in Rodehorst et al. [9]. Scene structure can be computed from feature tracks and projection matrices using for example linear or optimal triangulation [7]. Once pose and structure estimates are available, a common fine-tuning step is to perform a bundle adjustment, where the total reprojection error of all computed

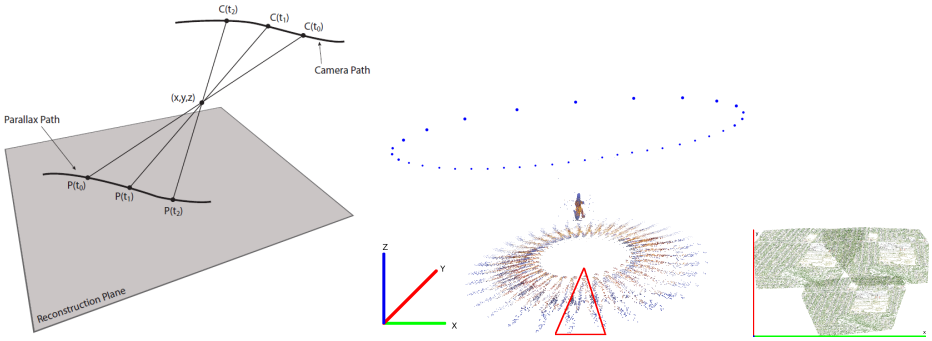
3D points in all cameras is minimized using non-linear techniques [5].

There are a number of successful general reconstruction algorithms in the literature, and comprehensive overviews and comparisons are given in Seitz et al. [10] and Strecha et al. [11]. As for sequential reconstruction algorithms, Pollefeys et al. [12] provides a method for reconstruction from hand-held cameras, Nistér [13] deals with reconstruction from trifocal tensor hierarchies, while Fitzgibbon et al. [14] provides an approach for turn-table sequences. In general, our approach differs from these and other algorithms in its use of additional constraints arising from the camera's motion into solving for feature tracking and scene structure. It is also important to note its differences with Tomasi-Kanade factorization [6]. This method can recover shape and motion from a sequence of images under weak-perspective projection. The main observation is that if feature tracks of scene points are collected in a measurement matrix, scene point trajectories reside in a certain subspace. This matrix is of reduced rank because tracks for scene points are constrained, as the motion of each point is globally described by the rigid transformation which the object or scene is undergoing. The end goal of structure recovery based on a geometric constraint, as well as the handling of outliers, is of similar nature to our work. However, our approach differs substantially in that we use two constraints under full perspective projection, can correct the feature tracks themselves using a non-algebraic solve, and use correction information to efficiently compute scene structure. It also differs from RANSAC [4], where one can for example estimate epipolar geometry and cameras accurately even in the presence of some outliers, but structure computation for uncorrected tracks would still be inaccurate.

3 Introduction to Parallax Paths

This section will further introduce the concept of parallax paths. The primary observation is that, for smooth and planar camera trajectories, parallax movement corresponding to a scene point viewed by the camera is uniquely determined as a unique *parallax path* on a parallel plane, such that all viewed scene points trace equal parallax paths up to translation and scale. A parallax path for a scene point is defined as the path formed on a plane, over time, of rays from the camera center and through the point. Dually, a scene point uniquely determines a *feature track* in a set of images. The concept is shown in Fig. 2(a). The plane onto which parallax paths are created will be referred to as the *reconstruction plane*, π. Though a parallax path can be thought of as continuous, it is really made up of discrete samples, where the camera's time resolution (frames per second) determines how close adjacent samples are on the traced path. For a given camera position in time, the set of parallax path positions it traces on π, corresponding to every scene point viewed at that instant in time, are defined as *replicas*. To illustrate this concept, Fig. 2(b) shows the parallax paths created for a sparse turntable reconstruction. Notice how each set of replicas in Fig. 2(b-c) visually resembles a 2D projection of the 3D object onto the reconstruction plane. Some conditions exist on what reconstruction plane should be used, and this will be discussed further in Sec. 3.1.

Mathematically, it is straightforward to show that a camera point moving on a plane as in Fig. 2(a), shooting rays through a set of fixed 3D positions onto a parallel plane, produces identical projected paths up to scale and translation. The 3D positions ef-



(a) Parallax paths concept (b) Traced parallax paths (c) Replicas

Fig. 2. Each 3D scene point, when projected from the cameras viewing it onto a plane, traces a unique path as the camera moves (*a*). Sparse ground-truth reconstruction of the *Dinosaur* dataset [15] with parallax paths and replicas (one highlighted in red) traced on the plane $Z = -1$ (*b*), with camera positions at $Z = 0$ rendered in blue, and a top view of replicas on $Z = 10$ obtained for the *Campus* synthetic aerial dataset (*c*).

fectively act like pinhole cameras. Let Z be the axis perpendicular to the two parallel planes, with X and Y axes tangent to the planes, and $(X_t, Y_t, 1)$ be the camera point at time t, with $Z = 0$ set as the camera plane. Then for 3D point (X, Y, Z) the projection of the camera position is $(X', Y', Z') = (X/Z, Y/Z, 1/Z - 1) + (1 - 1/Z)) * (X_t, Y_t, Z_t)$, which corresponds to a scaling and translation of (X_t, Y_t, Z_t).

3.1 Parallax Path Calculation

In practice, a discrete parallax path for each scene point can be computed directly from 3×4 projection matrices P_t for each camera location in time and its corresponding feature track. In our coordinate representation, we define each projection matrix as $P_t = K[R|T]$ [7], where K corresponds to the camera's fixed 3×3 intrinsic calibration matrix, while R_t is its absolute orientation matrix and T_t its absolute translation at time t. Each camera center position, $C_t = [X_t, Y_t, Z_t, W_t]$, can be computed from P_t as in Eq. 1, where p_j corresponds to the j_{th} column of P_t [7]. For any k_{th} feature track, a ray from camera center position C_t and through its pixel coordinates x_{kt} on the image plane at time t can be computed parametrically per Eq. 2 [7]. The right pseudo-inverse P_t+ of projection matrix P_t is computed from $P_t+ = P_t^T(P_t P_t^T)^{-1}$. Since a ray can be defined with two points, one will always be the camera center C_t and the other a point X_{kt} in space defined by the parameter λ.

$$C_t = [det([p_2, p_3, p_4]), -det([p_1, p_3, p_4]), det([p_1, p_2, p_4]), -det([p_1, p_2, p_3])] \quad (1)$$

$$X_{kt}(\lambda) = C_t + \lambda(P_t+)x_{kt} \quad (2)$$

To compute the intersection between such a ray and the reconstruction plane $\pi = (A, B, C, D)$, the value of the parametric distance 'λ' along the ray is computed, for

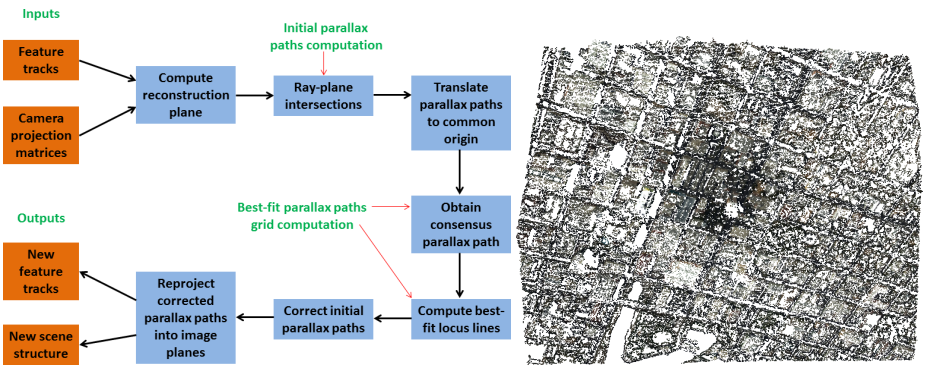
which the intersection is achieved. Let the ray $R(\lambda) = R_0 + \lambda R_d$, $\lambda > 0$, such that $R_0 = [X_0, Y_0, Z_0]$ corresponds to the camera center coordinates C_t at time t and $R_d = [X_d, Y_d, Z_d]$ is some point along the ray. If the plane is defined as $AX+BY+CZ+D = 0$, then $A(X_0 + X_d\lambda) + B(Y_0 + Y_d\lambda) + (Z_0 + Z_d\lambda) + D = 0$, which yields the value for 'λ' shown in Eq. 3. Performing this ray-plane intersection for rays from the camera through a scene point at each time instant results in a discrete parallax path for that point. Parallax paths can then be computed for all available scene points.

$$\lambda = \frac{-(AX_0 + BY_0 + CZ_0 + D)}{AX_d + BY_d + CZ_d} \tag{3}$$

The use of a proper reconstruction plane is key, and must comply with a series of criteria. First, it should lie parallel to the best-fit plane for the set of segment camera positions. It should be placed such that scene structure lies in-between both planes, with the only effect of its distance to the best-fit camera plane being an absolute scaling in parallax path coordinates on the reconstruction plane; all relative positioning remains constant. Notice how a non-parallel reconstruction plane would result in distorted parallax paths up to a transformation, while with a perpendicular plane information about the true camera trajectory is lost. It is also important to note that the framework can only be used in cases such as in aerial video and certain turntable sequences, where scene points do not intersect the visual hull made up of the scene and cameras, since otherwise a suitable reconstruction plane cannot exist for constructing parallax paths, because rays would lie directly on the camera plane.

4 Feature Track Correction Based on Parallax Paths

In this section, it will be described how the parallax paths framework can be used to correct feature tracks and compute structure in sequential reconstruction. The computation of parallax paths as described in Section 3.1 is performed in *segments* of a longer camera trajectory, where movement for a segment is mainly planar. The segments should be chosen such that there is overlap and all possible feature track positions are covered by at least one segment. Such processing can be thought of as a 'sliding window' type of correction. If the number of segments is increased, feature track and structure correction for a given segment involves fewer tracks and should be computed faster, so total processing time is unaffected. Thus, the global solution to feature tracking and structure computation is broken down into individual sub-problems. This is key since the amount of images in aerial video could be arbitrarily long, yet each sequential solve can be kept small. For now, focus will be on explaining the correction process for one particular segment, though concatenation across segments is necessary for long sequences. For one segment, the process is summarized as follows. The first step is to place all computed parallax paths in a 2D position-invariant reference, such that paths only differ in scale. In that reference frame, a set of best-fit *locus lines* and a *consensus path* can be computed. Parallax paths are then corrected to fit the best-fit lines and path, the actual correction is applied on parallax paths positions over the original reconstruction plane, and reprojected into each camera's image plane to obtain new, corrected pixel feature track coordinates. A flowchart for the correction process is shown in Fig. 3(a).



(a) Flowchart for feature track correction (b) Corrected structure, top view

Fig. 3. Flowchart for track correction based on parallax paths, for one segment of a sequential reconstruction (*a*). An example corrected structure is shown in (*b*) for the *Stockton* dataset.

4.1 Segment-Based Parallax Paths Computation

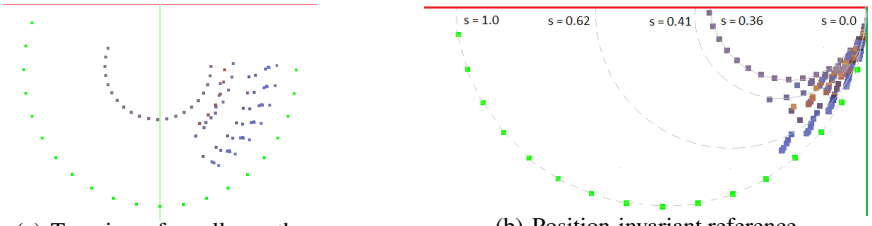
To each segment, anywhere along the sequential reconstrution, there is an associated *anchor frame* where it begins. For a given segment, the first step is to compute feature tracks for its m images, beginning at the anchor. Any algorithm discussed in the Introduction can be used, which can also compute camera intrinsics (assumed constant throughout) and extrinsics if these are not initially available, or with software such as *Bundler* [1]. Accurate projection matrices are key towards our algorithm's success, since inaccuracies will skew the obtained parallax paths. Next, a reconstruction plane is chosen parallel to the segment's best-fit camera plane. Now, parallax paths can be computed as in Section 3.1 using the computed feature tracks and projection matrices.

4.2 Parallax Path Analysis on a Position-Invariant Reference

Once parallax paths have been computed for the segment, as shown in Fig. 4(a), the next step is to eliminate the effect of position such that parallax paths differ only by scale. To this end, all parallax paths and projected cameras are placed in a separate, 2D position-invariant reference location, such that the parallax path positions of each track at the anchor frame coordinates all coincide at the same origin. In this representation, shown in Fig. 4(b), it becomes clear to see that the position-invariant parallax paths follow the shape of the projected camera path exactly, but at different *scales*. This is key towards our algorithm, as parallax path position and scale uniquely define the parallax of each scene point. As discussed next, in general this situation will not occur, and inaccuracies in the shape and scale of parallax paths will be present.

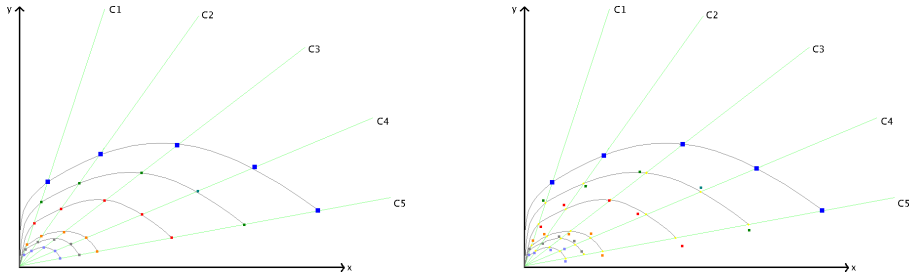
4.3 Enforcement of Inter and Intra-camera Constraints

In the position-invariant reference described in Section 4.2, for perfect cameras and feature tracks, parallax paths on the position-invariant reference form identical yet scaled



(a) Top view of parallax paths (b) Position-invariant reference

Fig. 4. Parallax paths and projected camera path (in green) obtained on a reconstruction plane for a set of scene points (*a*). The paths placed in a 2D position-invariant reference are shown in (*b*), where continuous curves for a few paths depict that they only vary in scale s.

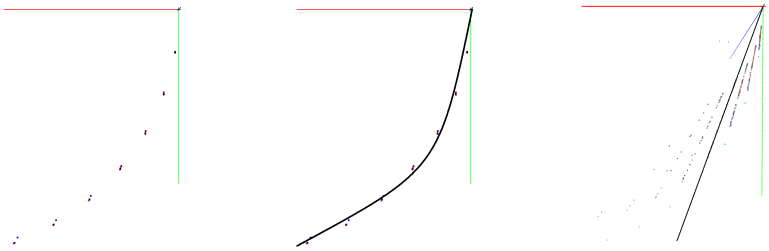


(a) Perfect case (b) General case

Fig. 5. Parallax paths and projected camera centers on the position-invariant reference, for perfect cameras and feature tracks, form a 'perfect grid' (*a*). In the case of inaccurate feature tracks (*b*), deviations exist with respect to this grid. In this example, position-invariant parallax paths are shown for five scene points, where discrete path positions are drawn in a specific color and joined by continuous light-grey curves. Each locus line is drawn in light green for each of five cameras C_1 to C_5. The projected camera path appears as a discrete set of larger blue squares, joined here by a light-grey curve. Notice how each curve is a scaled version of the projected camera path.

versions of the camera path projected onto the plane. Furthermore, all features seen by a given camera yield parallax path positions that are collinear, along *locus lines*. An *inter-camera* parallax path constraint holds for all cameras involved in a given feature track, while an *intra-camera* locus line constraint holds for the features from all tracks that are seen by a given camera. This concept is illustrated in Fig. 5(a), where parallax path positions form a *perfect parallax paths grid* at the position-invariant reference.

A closer analysis of Fig. 4(b) and Fig. 5(a) reveals another very important concept: in the position-invariant reference, we have proven that all *replicas*, corresponding to parallax path positions traced for all scene points seen by the same camera, lie along the same line along with the projected camera center, known as a *locus line*. In a perfect setting, reprojection error is zero for a scene point whose position-invariant parallax paths lie along such lines, across all cameras that view it. Notice that movement is possible along locus lines, such that reprojection error can be maintained at zero for a scene point with respect to any of the cameras, but then the *shape* of the parallax path, and



(a) Scaled paths (b) Consensus path (c) Locus line

Fig. 6. Scaled position-invariant parallax paths (*a*), best-fit quadratic to the scaled paths (*b*) and example of a locus line for a perfect parallax paths grid (*c*).

its *scale*, are violated. The very power of the parallax paths technique lies in the fact that the inter-camera parallax path and intra-camera locus line constraints jointly create a 'grid' over the position-invariant reference, which in the perfect case associate both the exact parallax scale and perfect feature track for a scene point, and this principle is the main concept behind the proposed feature track correction scheme. In the general case, however, feature tracks are inaccurate, such that position-invariant parallax path positions will not lie on the perfect grid. This is shown in Fig. 5(b). The novel feature tracks correction procedure essentially comes down to creating a *best-fit parallax paths grid* from all the available inter-camera and intra-camera consensus information, such that the resulting grid defines adjusted feature tracks.

The creation of a best-fit parallax paths grid is a two-step process. The first step is to obtain a *consensus parallax path*. Since our algorithm does not alter camera parameters, the consensus path *is* the position-invariant projection of the camera path. Alternatively, to relax the strong dependency on camera accuracy, in order to achieve a consensus path that is not necessarily the exact projection of the camera path, first all parallax paths on the position-invariant reference are scaled such that they match the scale of the projected camera trajectory, as shown in Fig. 6(a). Next, a best-fit curve to the obtained positions is obtained, for example by obtaining the best-fit quadratic or cubic to the set of equal-scale paths, which yields low residual errors over short, smooth trajectories, as shown in Fig. 6(b). For each position-invariant parallax path, the consensus path is then scaled such that the residual error with respect to the original path is minimized, and this defines the final parallax scale for the corresponding scene point.

The second step is to compute a locus line corresponding to each camera. An example of a locus line is shown in black in Fig. 6(c), for a perfect grid. Since our algorithm does not alter camera parameters, such lines are a direct function of the cameras, defined between the origin of the position-invariant reference and the camera projection's position on this reference. Alternatively, to relax the strong dependency on camera accuracy, a robust line-fitting technique can be used, for example linear regression embedded in RANSAC [4]. Finally, the best-fit grid results from intersecting the locus lines with the scaled consensus parallax path at each position-invariant parallax path location. The power of this grid is that it forces outlier tracks to comply with the constraints imposed by the cameras and/or inlier tracks. In fact, parallax paths are essentially a constraint

on the entire state space of a structure from motion problem. With ground-truth values for all parameters, the parallax paths will fit the model perfectly, which is also true of epipolar constraints on feature matches. For scene positions along each anchor frame ray, having zero reprojection error means the corresponding points in all images are on the respective epipolar lines. Ideally, if this were true for all scene points, this implies that all camera parameters and tracks are perfect, and all scene points will form perfectly matching parallax paths, up to scale and translation.

4.4 Adjustment of Feature Tracks and Scene Structure

Once the best-fit parallax paths grid has been created, the difference between the original position-invariant paths and the grid is computed. Finally, this difference is applied on the original reconstruction plane parallax paths. Each corrected path position is then reprojected to each respective camera, in order to obtain corrected feature tracks.

Another advantage of this framework is that it allows for a very simple update of scene structure. For the k_{th} corrected feature track, the corresponding scene point X_k can be computed in terms of its previously-recovered scale s_k as shown in Eq. 4 using the corrected parallax path coordinates on the reconstruction plane for the anchor camera, $T_{k,1}$, and anchor camera center C_1, which uses simple interpolation assuming a scale of '0' at the reconstruction plane and '1' right at the camera center's position.

$$X_k(s_k) = (s_k)C_1 + (1 - s_k)T_{k,1} \tag{4}$$

Given that rays through corrected tracks now intersect exactly in space, this is much more simple than having to use for example multi-view linear triangulation [7], where a system of the form $AX = 0$ is solved for a best-fit 3D position, with an A matrix of size $2N \times 4$ for N cameras, using for example Singular Value Decomposition.

Concatenation of Corrections Across Segments The discussion so far has focused on describing the algorithm for one segment. However, concatenation of corrections across neighboring segments is straightforward. The parallax paths correction can be performed totally independently for different segments, but always making sure and adjusting any feature tracks that span multiple adjacent segments such that they always have the same parallax scales, since each scene point corresponding to a track has a unique parallax movement as seen by the total set of cameras.

Relation with Epipolar Geometry Finally, it will be shown how the presented framework is geometrically valid as it meets the necessary epipolar geometry constraints. Given projection matrices for each of the cameras in a segment, it is possible to extract pairwise fundamental matrices F_{ij} between any camera pairs. In general, let P_i be the projection matrix for the first camera of a pair, P_j for the second camera, P_i+ is the pseudo-inverse of P_i and C_i is the camera center for the first camera. The fundamental matrix between the two views is then given by $F_{ij} = [P_jC_i]_xP_jP_i+$ [7].

The first result is that a locus line on the position-invariant reference, when placed on the original reconstruction plane such that it intersects the parallax path position

corresponding to a feature track position seen in a given camera, reprojects on that camera's image plane as an epipolar line, associated to the fundamental matrix computed between itself and the anchor camera, as well as the anchor frame feature track position. Furthermore, if locus lines are computed for any two cameras C_N and C_M and a pixel feature track position x_M is known for the M_{th} camera, the exact corresponding feature track position x_N in the N_{th} image is given by an intersection of epipolar lines with respect to both camera M and anchor camera 1, as shown in Eq. 5. This relationship is very powerful, as it allows parallax paths to 'induce' very accurate epipolar geometry-based estimation even at very long baselines.

$$x_N = (F_{N,1} * x_1) \times (F_{N,M} * x_M) \tag{5}$$

5 Results

A number of tests were designed to test the general behavior and accuracy of the proposed feature track correction framework on a number of real and synthetic scenes representative of smooth, continuous camera trajectories. This includes an analysis of the accuracy of corrected feature tracks and scene structure after correction. All tests were conducted on a dual-core *Intel Core 2 Duo* machine at 2.13 GHz with 2 GB of RAM, on one thread. Both sparse and dense feature tracking were analyzed.

One way to test the overall harmony of the resulting feature tracks and structure after correction is to compare the number of iterations, processing time and total reprojection error after applying bundle adjustment on the corrected set, referred to as $PPBA$, as opposed to applying it on the original feature tracks and structure, which will be referred to as TBA. The cost function to minimize, total pixel reprojection error, is the sum of squares of the reprojection error of each scene point with respect to each of its corresponding feature track positions, summed over all scene points. The sparse SBA implementation of bundle adjustment was used [5]. The results are shown in Table 1. In general, the time it takes to compute the parallax paths correction *and* run $PPBA$ is faster than TBA, and converges in less iterations, with a lower final reprojection error. The combined time for *PPBA* plus parallax paths correction is less than in *TBA* for *Dinosaur* (70 ms) and *dinoRing* [10] (20 ms). For *Stockton*, an aerial video sequence, it's slightly greater (2.07 s) for the sparse case, but lower for the dense case. For *fountain-P11* [11] it's also greater (0.98 s); notice that this is not an aerial or turntable sequence but we were still able to apply our method. In all of these cases, however, the final reprojection error was significantly smaller, even if timing was sometimes higher. For the *Palmdale* aerial sequence, a reconstruction was tested for which bundle adjustment does not even begin to iterate due to the high initial reprojection error, to see how our algorithm would behave with very inaccurate cameras. In this case, the algorithm failed to improve the tracks, as evidenced in Table 1. Our system is capable of cleaning up even large tracking errors, but not with very inaccurate cameras, as this skews creation of the best-fit grid for detecting and correcting such errors. However, if using robust techniques such as RANSAC to estimate the cameras from initial feature tracks despite some outliers, or GPS/IMU coordinates, this is generally not an issue. Also, we assume mainly static scenes, and don't account for movers directly, though inaccurate

tracks due to movers are also fixed to comply with the consensus parallax movement.

It is very important to note that since the feature track and structure correction fits a consistent geometry with the cameras, bundle adjustment converges faster and all it really does is incrementally fine-tune the cameras and structure further. This can include radial distortion, which our technique does not account for explicitly. On the other hand, bundle adjustment applied using the original feature tracks tries to minimize reprojection error but based on tracks that potentially have errors themselves, and thus a zero reprojection error is possible without having an accurate structure in a ground-truth sense, since it is optimized to inaccurate tracks. Performing bundle adjustments more efficiently per segment, after the extra correction step, further increases the robustness of the final sequential reconstruction.

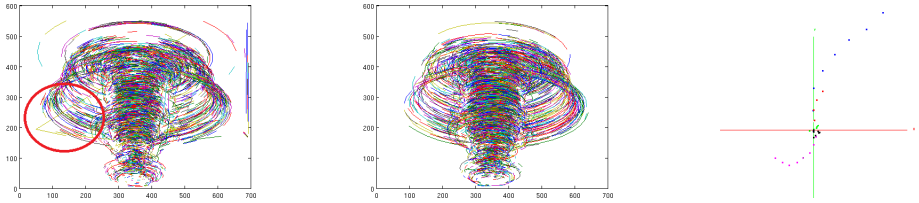
Another test dealt with analyzing the quality of the resulting feature tracks. Fig. 7(a)

Table 1. Comparison of total reprojection error ϵ in pixels, processing time t in seconds and iterations I of Levenberg-Marquardt, for bundle adjustment applied using the output of the proposed algorithm ($PPBA$) versus bundle adjustment applied using the original feature tracks and structure (TBA), along with number of scene points N_{SP}.

Dataset	PPBA ϵ (px)	PPBA t (s)	I_{PP}	TBA ϵ (px)	TBA t (s)	I_T	N_{SP}
Stockton	0.126	1.45	26	4.991	1.58	27	4991
Stockton dense	0.003	25.35	29	0.1041	27.73	31	151098
fountain-P11	0.232	0.80	82	4.851	0.32	31	1219
Dinosaur	1.208e-09	0.04	17	2.256	0.09	39	257
dinoRing	0.009	0.01	18	6.929	0.03	29	92
Palmdale	178.32	0.02	1	165.094	0.01	1	3978

shows a plot of SIFT-based tracks in image space. The pixel locations of every feature that makes up a track are joined by curves of the same color, such that curves that do not follow the general flow are inaccurate. Notice how some of the SIFT-based tracks are inaccurate. Fig. 7(b) shows the resulting feature tracks after the parallax paths-based correction for the corresponding segment, where the flow of tracks is more smooth. Fig. 7(c) shows the difference between the original and corrected parallax paths, for a small set of those feature tracks. The greatest differences are obtained for paths corresponding to track positions whose cameras lie farthest from the anchor, caused by the build-up of errors due to drifting in feature tracking. Besides correcting very inaccurate tracks, the proposed algorithm also detects and prevents such drifting for any track, allowing for error minimization in concatenation across segments and the ability to process very long image sequences without accumulating significant tracking errors.

Finally, the positive effect of the proposed correction on scene reconstruction can be shown. In Fig. 8, notice how outlier tracks, as evidenced in Figs. 8(c,e), are corrected to fit the geometry of the good tracks and cameras. It can also be seen that inaccuracies in structure, such as the dip highlighted in red in Fig. 8(a), are corrected with our method, resulting in a better structure such as the smooth road in Fig. 8(f).



(a) Original tracks (b) Corrected tracks (c) Detected drift

Fig. 7. Image locations of initial SIFT-based feature tracks (*a*) with errors marked in red and their parallax paths correction (*b*) for a turntable reconstruction. The positional difference between original and corrected parallax paths for a few select ones (*c*) shows that the algorithm detects feature track drifting, as evidenced by greater deviations from the reconstruction plane origin that correspond to track positions for cameras farther from the anchor frame.

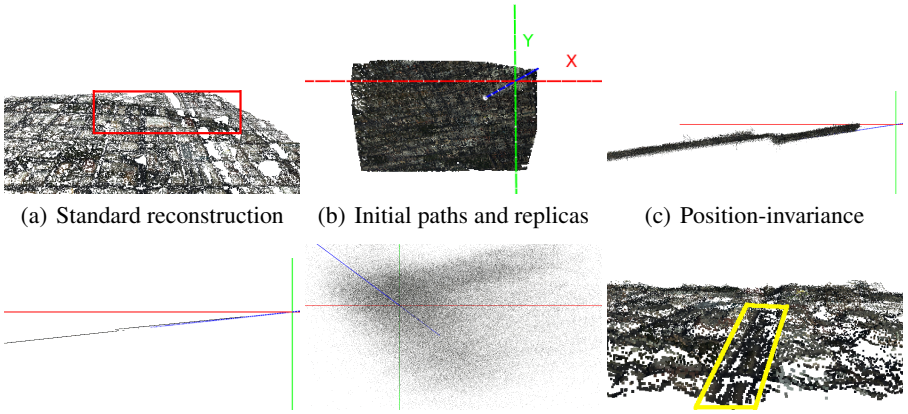
6 Conclusions

This paper presented a novel framework that makes use of the strong constraints imposed by the projection of a camera moving on a plane onto a parallel plane to allow for segment-wise feature track correction and scene structure computation, for applications such as in reconstruction from aerial video. Over sequential segments of the camera's path, intra-camera and inter-camera constraints imposed by the path allow for the detection and non-iterative correction of inaccurate tracks, leading to an improved final scene structure. Results were demonstrated for real and synthetic aerial video and turntable sequences, where the practical, efficient and inexpensive proposed method was shown to correct outlier tracks, detect and correct drift, and allow for structure improvement, while improving convergence for bundle adjustment optimization.

Acknowledgements. This work was supported in part by the Department of Energy, National Nuclear Security Agency through Contract No. DE-GG52-09NA29355. This work was performed in part under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344.

References

1. Snavely, N., Seitz, S.M., Szeliski, R.: Photo tourism: exploring photo collections in 3d. In: SIGGRAPH '06: ACM SIGGRAPH 2006 Papers, New York, NY, USA, ACM (2006) 835–846
2. Goesele, M., Snavely, N., Curless, C., Hoppe, H., Seitz, S.M.: Multi-view stereo for community photo collections. In: Proceedings of ICCV 2007. (2007)
3. Lowe, D.: Distinctive image features from scale-invariant keypoints. International Journal On Computer Vision **60** (2004) 91–110
4. Fischler, M., Bolles, R.: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Readings in computer vision: issues, problems, principles, and paradigms (1987) 726–740
5. Lourakis, M., Argyros, A.: The design and implementation of a generic sparse bundle adjustment software package based on the Levenberg-Marquardt algorithm. Technical Report 340, Institute of Computer Science - FORTH, Heraklion, Crete, Greece (2000)



(a) Standard reconstruction (b) Initial paths and replicas (c) Position-invariance

(d) Best-fit grid (e) Path differences from grid (f) Updated reconstruction

Fig. 8. Correction process for a segment of the *Stockton* dataset. Segment reconstruction without track correction (*a*), top view of computed closely-spaced replicas at $Z = -30$ (*b*), paths at the position-invariant reference (*c*), best-fit parallax paths grid (*d*) and difference between the initial paths and the grid (*e*), and final scene structure after correction (*f*).

6. Tomasi, C., Kanade, T.: Shape and motion from image streams under orthography: A factorization method. International Journal of Computer Vision **9** (1992) 137–154
7. Hartley, R.I., Zisserman, A.: Multiple View Geometry in Computer Vision. 2nd edn. Cambridge University Press (2004)
8. Scharstein, D., Szeliski, R.: A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. International Journal On Computer Vision **47** (2002) 7–42
9. Rodehorst, V., Heinrichs, M., Hellwich, O.: Evaluation of relative pose estimation methods for multi-camera setups. In: International Archives of Photogrammetry and Remote Sensing (ISPRS '08), Beijing, China (2008) 135–140
10. Seitz, S.M., Curless, B., Diebel, J., Scharstein, D., Szeliski, R.: A comparison and evaluation of multi-view stereo reconstruction algorithms. In: CVPR '06: Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Washington, DC, USA, IEEE Computer Society (2006) 519–528
11. Strecha, C., von Hansen, W., Gool, L.J.V., Fua, P., Thoennessen, U.: On benchmarking camera calibration and multi-view stereo for high resolution imagery. In: CVPR'08. (2008)
12. Pollefeys, M., Van Gool, L., Vergauwen, M., Verbiest, F., Cornelis, K., Tops, J., Koch, R.: Visual modeling with a hand-held camera. International Journal of Computer Vision **59** (2004) 207–232
13. Nistér, D.: Reconstruction from uncalibrated sequences with a hierarchy of trifocal tensors. In: Proceedings of the 6th European Conference on Computer Vision-Part I, London, UK, Springer-Verlag (2000) 649–663
14. Fitzgibbon, A.W., Cross, G., Zisserman, A.: Automatic 3d model construction for turntable sequences. In: Proceedings of the European Workshop on 3D Structure from Multiple Images of Large-Scale Environments, London, UK, Springer-Verlag (1998) 155–170
15. Oxford Visual Geometry Group: Multi-view and Oxford Colleges building reconstruction. `http://www.robots.ox.ac.uk/~vgg/` (2009)