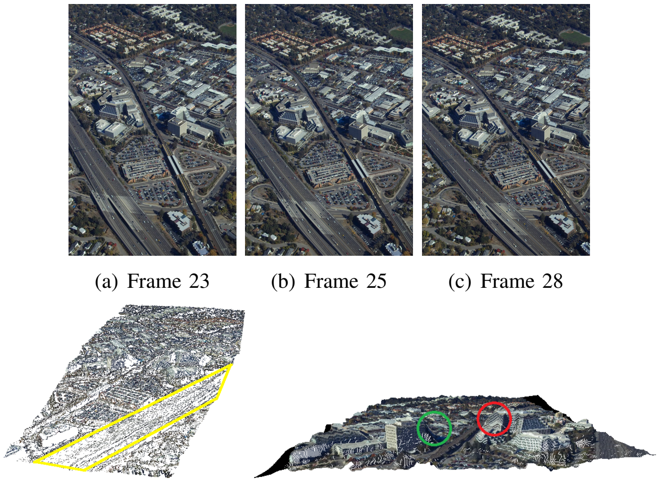
Path-Based Constraints for Accurate Scene Reconstruction from Aerial Video

Mauricio Hess-Flores*, Mark A. Duchaineau‡, and Kenneth I. Joy*
*Institute for Data Analysis and Visualization
University of California, Davis
Email: see http://www.idav.ucdavis.edu/people
‡Google, Inc.
Email: duchaine@google.com

***Abstract*—This paper discusses the constraints imposed by the path of a moving camera in multi-view sequential scene reconstruction scenarios such as in aerial video, which allow for an efficient detection and correction of inaccuracies in the feature tracking and structure computation processes. The main insight is 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. Two inter-camera and intra-camera constraints arise, which create a prediction of where all feature tracks should be located given the consensus information of all accurate tracks and cameras, which allows for the detection and correction of inaccurate feature tracks, as well as a very simple update of scene structure. This procedure differs from classical approaches such as factorization and RANSAC. In both aerial video and turntable sequences, the use of such constraints was proven to correct outlier tracks, detect and correct tracking drift, allow for a simple updating of scene structure, and improve bundle adjustment convergence.**


(a) Frame 23 (b) Frame 25 (c) Frame 28

(d) Texture-less regions (e) Occlusions and repetitive patterns

Fig. 1. Dense reconstruction (*d* and *e*) of the *Walnut Creek* dataset (input images *a-c*). Inaccuracies in the computed structure due to inaccurate dense correspondences caused by occlusions (green), repetitive patterns (red) and texture-less regions (yellow) are highlighted.

I. Introduction

The amount of work and research dealing with multi-view scene reconstruction, for example in applications such as robotics, surveillance and virtual reality, has increased substantially in the past years. 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]. One specific reconstruction scenario which has become very relevant recently is in the case of aerial video. Accurate and dense 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. For example, consider a scenario where an aircraft is flying around an urban environment, carrying an array of sensors that collects images at high frame rates as the aircraft circles around the scene over and over again. Once enough data has been collected, a semantic analysis of the scene becomes possible, and activity happening in the scene can be inferred.

In aerial video, 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 and are usually assumed known. Extrinsic parameters such as instantaneous position and orientation are usually at least roughly known due to GPS and IMU readings, respectively. In the case calibration data is not available, it can usually be estimated from feature tracks across the images, but then relies on the accuracy of such tracks. Overall, the accuracy of the final multi-view sequential reconstruction 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 calibration and structure estimation. Inaccuracies 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. Furthermore, due to the lack of ground-truth camera and/or structure data, reconstruction algorithms usually resort to non-linear *bundle adjustment* [5] parameter optimization to reduce total reprojection error. However, this can be the most expensive element in a reconstruction pipeline, despite efficient sparse implementations [6], and for convergence requires a good starting point close to the global minimum of the cost function.

Our main contribution is to present camera path-based constraints for improving feature track and structure accuracy as an intermediate step in sequential scene reconstruction, for applications such as in aerial video. The main insight is 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, or a *parallax path*, when projected onto a parallel plane. More details and results are presented in Hess-Flores et al. [7], but the entire algorithm is summarized here, and expanded with geometrical properties and the application of homography constraints to the framework. This approach introduces more and different constraints with respect to Tomasi and Kanade's classical factorization approach [8], and also differs from outlier detection through Fischler and Bolles' RANSAC algorithm [4]. Additionally, it will be shown how the same constraints allow for a simplified computation of scene structure after track correction, which is less expensive than traditional linear triangulation [9].

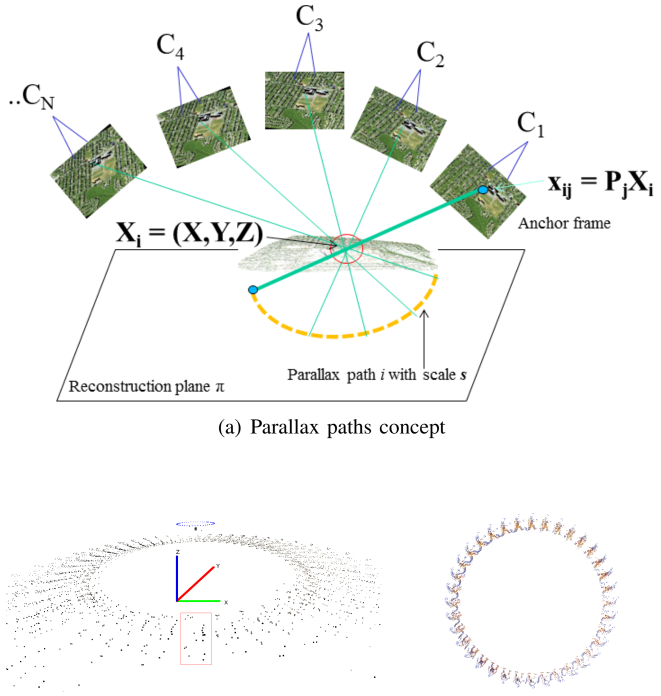
For reconstruction of a long, sequential video sequence, it is treated as a set of segment-wise, sliding window-type connected set of smaller reconstructions. For a given *segment*, beginning at its *anchor frame*, feature tracks are first computed, followed by camera calibration from these tracks if not initially available. 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. The key behind this method 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 processing 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 fixing of tracking inaccuracies at each step allows for stable sequential reconstructions, where errors are not allowed to accumulate over time.

An overview of multi-view scene reconstruction is provided in Section II. An introduction to the concept of parallax paths is provided in Section III. The introduction of path-based constraints towards improving feature tracking and structure computation is detailed in Section IV, geometrical properties of the framework are discussed in Section V, followed by results (Section VI), future work (Section VII) and conclusions (Section VIII).

II. Related Work

Typically, the input for scene reconstruction is a set of images and in some cases camera calibration information, while the output is usually a 3D point cloud along with color and/or normal information, representing scene structure. Camera parameters include intrinsics, 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 [9]. For estimating epipolar geometry information [9] between views, camera calibration (if initially unknown) and scene structure, most algorithms make use of feature tracks between images. Software packages such as *Bundler* [1] are capable of estimating all calibration and structure parameters from a set of images. This and other algorithms are based on SIFT feature detection and tracking [3], or others inspired by its concept such as SURF [10] and DAISY [11], 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 [12], such as optical flow. They can also be generated using an epipolar geometry estimate through a process known as guided matching [9]. However, dense feature tracking 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. In the case of sequential image sets, the use of a prior frame decimation [13] to filter redundant frames ensures mathematical stability for pose and structure estimation. An overview of different pose estimation methods based on feature tracking are given in Rodehorst et al. [14]. Two standard algorithms involve decomposing the epipolar geometry's *essential matrix* [9] in the case of two views, and camera resectioning to compute new camera positions from known feature tracks and scene structure [9]. Scene structure can be computed from feature tracks and projection matrices using for example linear or optimal triangulation [9]. 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 [6].

There are a number of successful general reconstruction algorithms in the literature, and comprehensive overviews and comparisons of different methods are given in Seitz et al. [16] and Strecha et al. [17]. For sequential reconstruction specifically, for example Pollefeys et al. [18] provides a complete algorithm for reconstruction from hand-held cameras, Nistér [19] deals with reconstruction from trifocal tensor hierarchies, while Fitzgibbon et al. [20] provides an approach for turn-table sequences. Our approach differs from these and other algorithms in its use of additional path-based constraints into improving feature tracking and scene structure. It is also important to note the differences with Tomasi-Kanade factorization [8]. This method can recover shape and motion from a sequence of images under weak-perspective projection, making use of the fact 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



(a) Parallax paths concept

(b) Traced parallax paths

(c) Replicas

Fig. 2. A scene point traces a unique path if projected from a camera moving along a plane onto a separate, parallel plane (*a*). Sparse reconstruction of the *dinoRing* dataset [16] with parallax paths traced on a parallel plane (*b*), with camera positions rendered in blue and one replica highlighted in red, and a top view of the obtained replicas for the *Dinosaur* dataset [21] (*c*).

of each point is globally described by the rigid transformation which the object or scene is undergoing. Our end goal of structure recovery based on a geometric constraint, while being able to deal with outliers, is similar, but 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.

III. Concept and Calculation of Parallax Paths

The primary observation behind our approach is that, for smooth and planar camera paths, parallax corresponding to a scene point (X, Y, Z) viewed by the camera is determined as a unique *parallax path* on a parallel plane, such that all viewed scene points trace equal paths up to translation and scale s, as illustrated in Fig 2(a). A parallax path for a scene point is defined as the path formed over time, on the *reconstruction plane* π, of rays from the camera center and through the point. Conversely, a scene point uniquely determines a *feature track* in a set of images. Though a parallax path can be thought of as continuous, it is really made up of discrete samples, one for each instant an image is acquired by the camera. 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*. Figs. 2(b,c) show the parallax paths created for a sparse turntable reconstruction, where each set of replicas visually resembles a 2D projection of the 3D object onto the reconstruction plane.

Mathematically, a camera moving on a plane as in Fig. 2(a), projecting rays through a set of fixed 3D positions onto a parallel plane, produces identical projected paths up to scale and translation. 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 is indeed a scaling and translation of (X_t, Y_t, Z_t).

A parallax path can be traced on π for each scene point, from its corresponding feature track and 3×4 projection matrices P_t for each camera location in time. We define $P_t = K[R|T]$ [9], 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 location, $C_t = [X_t, Y_t, Z_t, W_t]$, can be computed from P_t as its right null-space [9]. 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 (1) [9], with the right pseudo-inverse $P_t+ = P_t^T(P_tP_t^T)^{-1}$. A camera ray can be defined with two points, one being the camera center C_t and the other a point X_{kt} in space defined by the parameter λ. For the intersection point between such a ray and the reconstruction plane $\pi = (A, B, C, D)$, the value of the parametric distance 'λ' 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. Since 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 (2). Performing this ray-plane intersection for rays from the camera through each scene point at each time instant results in a discrete parallax path for each point.

$$X_{kt}(\lambda) = C_t + \lambda(P_t+)x_{kt} \ . \tag{1}$$

$$\lambda \quad = \quad \frac{-(AX_0 + BY_0 + CZ_0 + D)}{AX_d + BY_d + CZ_d} \ . \tag{2}$$

The chosen reconstruction plane must comply with a series of criteria. It should lie parallel to the best-fit plane for the set of segment camera positions, and placed such that scene structure lies in-between both planes. The only effect of the distance from π to the best-fit camera plane is an absolute scaling in parallax path coordinates on π. A non-parallel reconstruction plane would result in distorted parallax paths

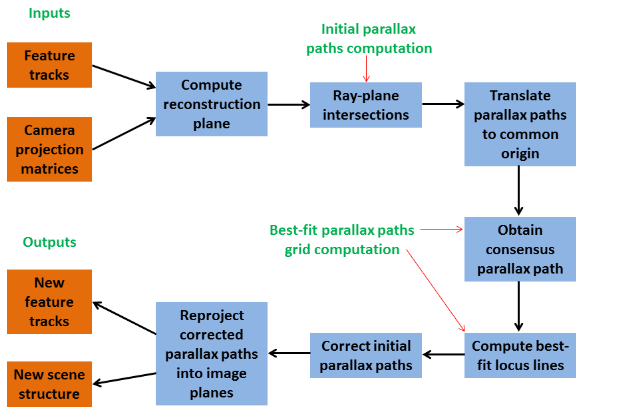

Inputs
Feature tracks
Camera projection matrices
Compute reconstruction plane
Initial parallax paths computation
Ray-plane intersections
Translate parallax paths to common origin
Best-fit parallax paths grid computation
Obtain consensus parallax path
Outputs
New feature tracks
New scene structure
Reproject corrected parallax paths into image planes
Correct initial parallax paths
Compute best-fit locus lines

Fig. 3. Flowchart for feature track correction in one segment of a sequential reconstruction [7].

up to a transformation. Also, the framework can only be used in cases where scene points do not intersect the visual hull made up of the scene and cameras, since otherwise rays would lie directly on the camera plane and parallax paths could not be traced.

IV. Path Constraint-Based Feature Track Correction

In this section, constraints arising from parallax paths will be described, as well as how these can be used to correct feature tracks and improve structure estimates. The computation of parallax paths as described in Section III is performed by *segments* of a longer camera trajectory, where movement per segment is modelled as planar. The segments should be chosen such that there is overlap and all possible feature track positions are covered by at least one segment. This allows for the global solution to feature tracking and structure computation to be is broken down into individual solves, which is very important since the amount of images in aerial video could be arbitrarily long.

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, where 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, for one segment. To each segment, anywhere along the path, there is an associated *anchor frame* where it begins, as shown in Fig. 2(a). For a given segment, the first step is to compute feature tracks for its m images, beginning at the anchor, along with the corresponding camera projection matrices, using any reconstruction algorithm discussed in the Introduction 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 III using the computed feature tracks and projection matrices.

A. Position-Invariant Reference Placement

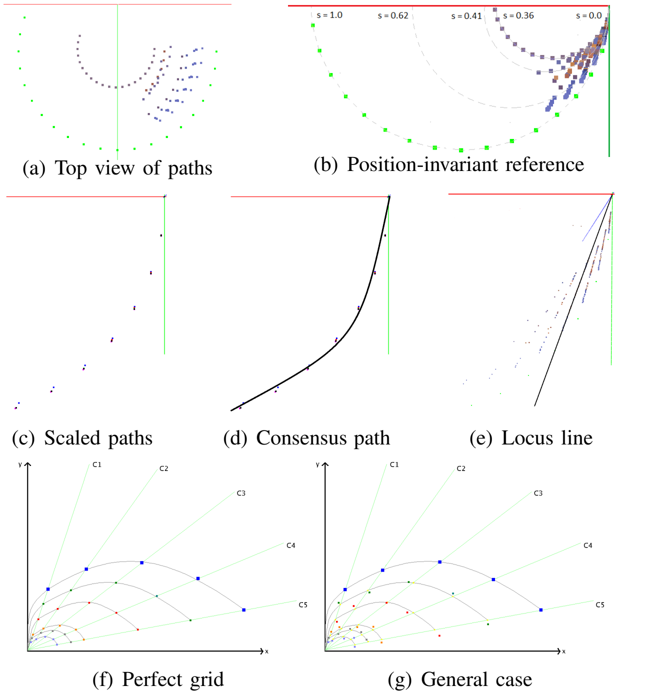
Following parallax paths computation, as shown in Fig. 4(a), the next step is to eliminate the effect of position, by placing all paths and projected cameras in a separate, 2D position-invariant reference location, such that the parallax path positions of each track at the anchor frame coordinates all coincide at the same origin. In this representation, shown in Fig. 4(b), it becomes clear to see that, in the ideal case, the position-invariant parallax paths follow the shape of the projected camera path exactly, but at different scales, since 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.

B. Inter and Intra-camera Constraints

In the ideal case, besides forming position-invariant parallax paths that are identical yet scaled versions of the camera path projected onto the plane, 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. 4(f), where parallax path positions form a *perfect parallax paths grid* at the position-invariant reference.

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. 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 an intersection '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. 4(g). The proposed 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.

Once on the position-invariant reference, as shown in Fig. 4(b) for a sample set of initial parallax paths (Fig. 4(a)), 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,



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

(c) Scaled paths (d) Consensus path (e) Locus line

(f) Perfect grid (g) General case

Fig. 4. Path-based constraint creation [7]. Parallax paths and the projected camera path (in green) are first 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. Next, the position-invariant paths are scaled (*c*), and a best-fit quadratic to the scaled paths is computed (*d*) along with locus lines, with an example shown in (*e*). Finally, the 'perfect grid' can be created, as shown in (*f*) for the ideal case. In the case of inaccurate feature tracks (*g*), deviations exist with respect to this grid. In (*f*) and (*g*), 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.

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. 4(c). 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. 4(d). 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. 4(e), 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. It forces outlier tracks to comply with the constraints imposed by the cameras and/or inlier tracks, much like in the case of epipolar geometry constraints.

C. *Feature Track Adjustment and Final Structure Computation*

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 (3) 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.

$$X_k(s_k) = (s_k)C_1 + (1 - s_k)T_{k,1} \ . \qquad (3)$$

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 [9], 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.

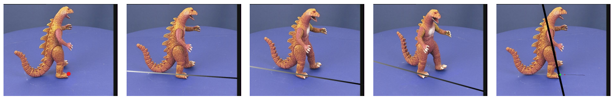
D. *Segment-Wise Concatenation*

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.

V. Geometrical Properties of Parallax Paths

It will now be discussed how the presented framework meets all 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 $F_{ij} = [P_jC_i]_x P_j P_i+$ [9].

An important observation is that a locus line on the position-invariant reference, when placed on the original reconstruction



(a) Frame 1 (b) Frame 3 (c) Frame 5 (d) Frame 7 (e) Intersection

Fig. 5. Feature track position in red (*a*) at frame 1, the anchor, and corresponding epipolar lines in frames 3, 5 and 7 (*b-d*), for the *Dinosaur* dataset. An intersection of epipolar lines is shown in (*e*).

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. An example of this is shown in Fig. 5, where the left-most image shows in red the first feature of a track, and the remaining images show the corresponding epipolar lines resulting from the set of perfect locus lines, along which the rest of the feature track positions should lie, where parallax scales are color-coded such that black corresponds to the smallest and white to the largest. 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 (4).

$$x_N = (F_{N,1} * x_1) \times (F_{N,M} * x_M) \ . \tag{4}$$

A. Homography Constraints on Scales

As previously mentioned, the scale for a scene point can vary from '0', if its position is right on the reconstruction plane, to '1', when coinciding with any of the camera centers which view it. It has been assumed so far that track scales are known, but now assume a very inaccurate track, for which a reliable range for its scale is initially unknown, or one that we wish to initialize. By making use of *homographies* [9] between pairwise consecutive camera frames, it becomes possible to achieve a much-reduced range of the scales to search over and achieve an accurate track. In experiments with aerial video, typically 90% of scales are removed after this filter.

A homography is a simpler model than epipolar geometry, allowing for a 2D prediction of a 3D movement, and since it doesn't correctly account for parallax, it generally presents a residual error for a given feature match. Let x be a 2D pixel position in an image, and x' its match in a second image. The relationship between the matches is $x' = Hx$, where H is a 3×3 homography matrix. For strictly planar scenes, x' can be recovered exactly from x. For general scenes with parallax, we can still make use of residual measurements over the set of available feature matches (obtained for example from SIFT) to create *bounds* on the expected image-to-image 2D movements of feature matches, to greatly constrain a multi-view feature track's allowed pixel movement. In fact, to determine just how accurate the homography model is to the more general epipolar geometry model, Torr's GRIC metric [22] can be used, keeping in mind that for large baselines, the homography model quickly becomes very inaccurate. In practice, after performing the parallax paths constraint computations, and taking into account the extra homography constraints, we do the following. Given an anchor position, for the remaining images we can move the resulting homography prediction position the closest distance to the locus line, and then move up or down that line within the maximum residual distance, searching only over those scales for the 'best' one. The definition of 'best' we have adopted refers to the best intensity consensus; for example where the standard deviations of intensities for candidate feature track positions take their lowest values. To show an example of our use of homography constraints, Fig. 6 shows a dense plane that was computed by initializing all image positions assuming a perfect homography, with the actual computed structure protruding from the computed plane, for the *Dinosaur* [21] dataset.

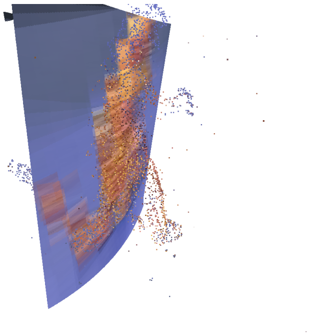


Fig. 6. Homography constraints on scale space, where a dense plane was rendered using our technique coupled with constraining homographies assuming a perfectly planar scene, for the *Dinosaur* [21] dataset. Actual scene points protrude from the computed plane.

VI. Results

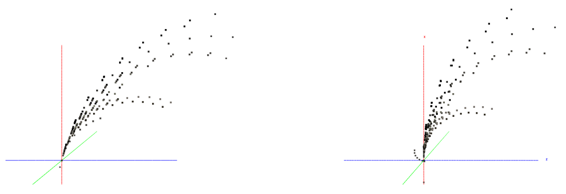
A number of tests were designed to analyze the general behavior and accuracy of the proposed feature track correction framework on smooth, continuous camera trajectories. All tests were conducted on a dual-core *Intel Core 2 Duo* machine at 2.13 GHz with 2 GB of RAM, on one thread. Both sparse and dense 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 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 [6]. The results are shown in Table I. In general, the time it takes to compute the parallax paths correction *and* run $PPBA$ is faster than TBA, and converges in less iterations, and always with a significantly lower final reprojection error. Performing bundle adjustments more efficiently per segment,

TABLE I

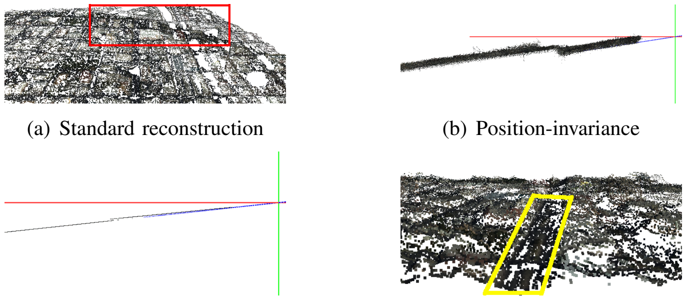
COMPARISON OF TOTAL REPROJECTION ERROR ϵ IN PIXELS, PROCESSING TIME t IN SECONDS AND ITERATIONS I OF LEVENBERG-MARQUARDT, FOR BUNDLE ADJUSTMENT APPLIED USING THE OUTPUT OF THE PROPOSED ALGORITHM ($PPBA$) VERSUS BUNDLE ADJUSTMENT APPLIED USING THE ORIGINAL FEATURE TRACKS AND STRUCTURE (TBA), ALONG WITH NUMBER OF SCENE POINTS N_{SP} [7].

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



(a) Perfect grid (b) Detected drift

Fig. 7. Perfect grid computed from a few select SIFT-based parallax paths, for the *dinoRing* dataset [16] (*a*). The positional difference between original and corrected paths (*b*) shows feature track drift detection is possible, as evidenced by greater deviations from the reconstruction plane origin that correspond to track positions for cameras farther from the anchor frame.



(a) Standard reconstruction (b) Position-invariance

(c) Best-fit grid (d) Updated reconstruction

Fig. 8. Feature track improvement for a segment of the *Stockton* dataset. Segment reconstruction without track correction (*a*), paths at the position-invariant reference (*b*), best-fit parallax paths grid (*c*) and final scene structure after correction (*d*).

after the extra correction step, further increases the robustness of the final sequential reconstruction. It is also important to note that radial distortion is not accounted for directly and that we assume mainly static scenes, though inaccurate tracks due to movers are also fixed to comply with the consensus parallax movement.

Another test dealt with analyzing the quality of the resulting feature tracks. Fig. 7(a) shows a perfect grid computed plot of SIFT-based tracks in image space. Fig. 7(b) shows the difference between the original and corrected parallax paths, for a small set of 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 Fig. 8(b), 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(d).

VII. FUTURE WORK

The presented framework has the potential to open the door to many applications, mainly in aerial imagery scenarios. One such possibility is improved dense tracking. For initialization of a new track based on the constraints, a 1D search over scales must be performed, which could involve searching for the position where image intensities best agree, and aided by using resolution pyramids, feature descriptors, and the homography constraints presented in Section V-A to provide bounds on the scales to search over. Track auto-completion, mainly with tracks which become occluded tracks and re-appear, also becomes possible by searching for parallax paths of equal scale and merging the separate track sections into a single feature track from the projected parallax path. Given a concatenated track, *virtual pixels* can be computed to indicate where a scene point is located in an image, even if it cannot actually be seen due to the effect of occlusions. Looking further, the framework could also be potentially used for the compression of both images and structure parameters, by storing only scale-based information. If used jointly with color segmentation, the joint analysis of computed tracks could make for a novel algorithm for getting accurate matches and structure over texture-less regions. Also, we're looking into the mathematical definition of a multi-view tensor based on the proposed algorithm.

VIII. CONCLUSIONS

This paper discussed the strong constraints imposed by the projection of a planar camera path onto a parallel plane, which allows for feature track outlier detection and correction along with a simple and improved structure computation, for applications such as in aerial video and turntable sequences. Analysis is performed over continuous segments of the camera's path, where intra-camera and inter-camera constraints arising from the consensus of all initial feature track and camera calibration information create a prediction of where each feature track should lie, such that outliers can be detected and corrected in a non-iterative manner, while also allowing for a simple and accurate final structure computation. Results on both real and synthetic aerial video and turntable sequences show that the framework corrects outlier tracks, detects and corrects drift, and improves scene structure, while also improving bundle adjustment convergence.

Acknowledgment

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] N. Snavely, S. M. Seitz, and R. Szeliski, “Photo tourism: exploring photo collections in 3d,” in *SIGGRAPH ’06: ACM SIGGRAPH 2006 Papers*. New York, NY, USA: ACM, 2006, pp. 835–846.

[2] M. Goesele, N. Snavely, C. Curless, H. Hoppe, and S. M. Seitz, “Multi-view stereo for community photo collections,” in *Proceedings of ICCV 2007*, 2007.

[3] D. Lowe, “Distinctive image features from scale-invariant keypoints,” *International Journal On Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[4] M. Fischler and R. Bolles, “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*, pp. 726–740, 1987.

[5] B. Triggs, P. McLauchlan, R. I. Hartley, and A. Fitzgibbon, “Bundle adjustment - a modern synthesis,” in *ICCV ’99: Proceedings of the International Workshop on Vision Algorithms*. London, UK: Springer-Verlag, 2000, pp. 298–372.

[6] M. Lourakis and A. Argyros, “The design and implementation of a generic sparse bundle adjustment software package based on the Levenberg-Marquardt algorithm,” Institute of Computer Science - FORTH, Heraklion, Crete, Greece, Tech. Rep. 340, August 2000.

[7] M. Hess-Flores, M. A. Duchaineau, and K. I. Joy, “Sequential reconstruction segment-wise feature track and structure updating based on parallax paths,” in *ACCV 2012*, in press.

[8] C. Tomasi and T. Kanade, “Shape and motion from image streams under orthography: A factorization method,” *International Journal of Computer Vision*, vol. 9, pp. 137–154, 1992. [Online]. Available: http://dx.doi.org/10.1007/BF00129684

[9] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge University Press, 2004.

[10] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, “Speeded-up robust features (SURF),” *Comput. Vis. Image Underst.*, vol. 110, pp. 346–359, June 2008.

[11] E. Tola, V. Lepetit, and P. Fua, “Daisy: an efficient dense descriptor applied to wide baseline stereo,” in *PAMI*, vol. 32, no. 5, May 2010, pp. 815–830.

[12] D. Scharstein and R. Szeliski, “A taxonomy and evaluation of dense two-frame stereo correspondence algorithms,” *International Journal On Computer Vision*, vol. 47, no. 1-3, pp. 7–42, 2002.

[13] D. Knoblauch, M. Hess-Flores, M. A. Duchaineau, K. I. Joy, and F. Kuester, “Non-parametric sequential frame decimation for scene reconstruction in low-memory streaming environments,” in *ISVC 2011*, ser. LNCS, vol. 6938, 2011, pp. 363–374.

[14] V. Rodehorst, M. Heinrichs, and O. Hellwich, “Evaluation of relative pose estimation methods for multi-camera setups,” in *International Archives of Photogrammetry and Remote Sensing (ISPRS ’08)*, Beijing, China, 2008, pp. 135–140.

[15] Q.-T. Luong, “Matrice fondamentale et auto-calibration en vision par ordinateur,” Ph.D. dissertation, Universite de Paris-Sud, Orsay, 1992.

[16] S. M. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski, “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, pp. 519–528.

[17] C. Strecha, W. von Hansen, L. J. V. Gool, P. Fua, and U. Thoennessen, “On benchmarking camera calibration and multi-view stereo for high resolution imagery,” in *CVPR’08*, 2008.

[18] M. Pollefeys, L. Van Gool, M. Vergauwen, F. Verbiest, K. Cornelis, J. Tops, and R. Koch, “Visual modeling with a hand-held camera,” *International Journal of Computer Vision*, vol. 59, pp. 207–232, 2004.

[19] D. Nistér, “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, pp. 649–663.

[20] A. W. Fitzgibbon, G. Cross, and A. Zisserman, “Automatic 3d model construction for turn-table sequences,” in *Proceedings of the European Workshop on 3D Structure from Multiple Images of Large-Scale Environments*. London, UK: Springer-Verlag, 1998, pp. 155–170.

[21] Oxford Visual Geometry Group, “Multi-view and Oxford Colleges building reconstruction,” http://www.robots.ox.ac.uk/~vgg/, August 2009.

[22] P. H. Torr, “Geometric motion segmentation and model selection,” *Philosophical Transactions: Mathematical, Physical and Engineering Sciences*, vol. 356, no. 1740, pp. 1321–1340, 1998.