Street View Motion-from-Structure-from-Motion

Bryan Klingner
bmk@google.com

David Martin
royal@google.com

James Roseborough
jrosebor@google.com

Abstract

We describe a structure-from-motion framework that handles "generalized" cameras, such as moving rolling-shutter cameras, and works at an unprecedented scale—billions of images covering millions of linear kilometers of roads—by exploiting a good relative pose prior along vehicle paths. We exhibit a planet-scale, appearance-augmented point cloud constructed with our framework and demonstrate its practical use in correcting the pose of a street-level image collection.

1. Introduction

Google Street View has a repository of billions of 2D images captured with rolling-shutter camera rigs along vehicle trajectories. Although we use GPS and inertial sensors to estimate the pose of this imagery, it still contains low-frequency error due to challenging GPS environments in cities and elsewhere. To improve the pose of these images, we have extended traditional Structure from Motion (SfM) techniques to construct a point-based model of the street-level world where each point carries both its geometric position as well as its local appearance from several views (see Figure 1). We use the appearance information from this model to find corresponding 3D points viewed from nearby images, and the geometric information to align the cameras that view them, thereby globally correcting the imagery's pose: motion-from-structure-from-motion.

To create our SfM model, we must overcome two important challenges not encountered in typical SfM problems: rolling-shutter cameras and planet-wide scale. We use the same tool to tackle both: a good initial estimate of the local vehicle trajectory. Although GPS may introduce global error of many meters in our estimate of the trajectory, the local shape is determined almost entirely by integration of inertial sensors. These sensors are trustworthy over short distances and they give us an accurate estimate of the vehicle's motion during image capture as well as the relative pose of images nearby each other in the trajectory, as depicted in Figure 2. In Sections 3 and 4, we present a generalized camera model that uses this local pose to handle rolling shutters in the context of projection, triangulation, and bundle adjustment. Then, in Section 5, we use local pose to overcome superlinear matching cost and achieve planet scale. In Section 6, we descibe how we apply the resulting appearance-augmented SfM model to reduce the global pose error of our imagery by more than 85% in the densest urban environments in the world.

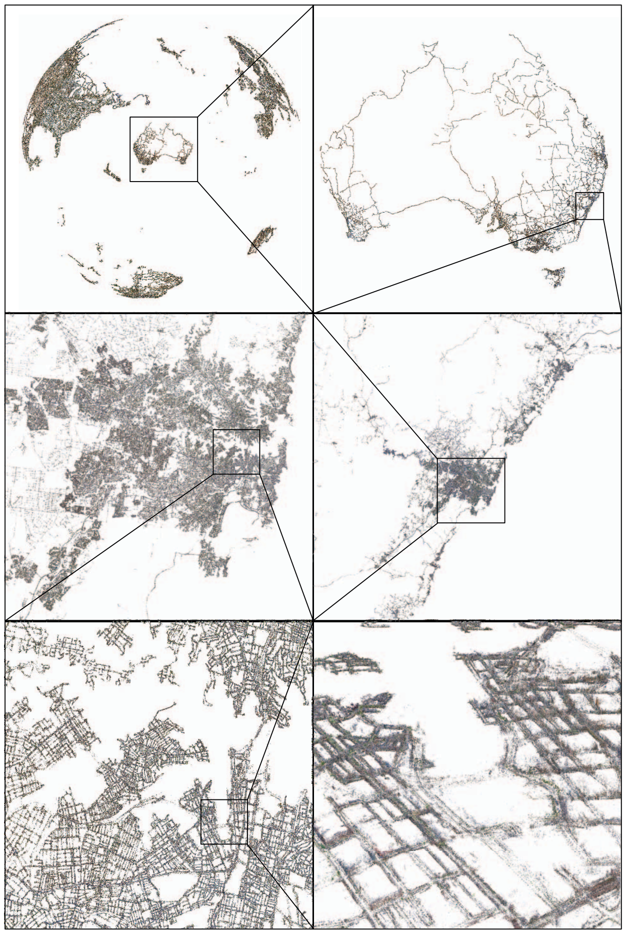
Figure 1. An appearance-augmented point cloud comprising 404 billion tracked features, computed from street-level imagery. Every point in the cloud carries with it local appearance descriptors from at least three different viewpoints. Frames zoom successively closer to a detail of Sydney harbor.

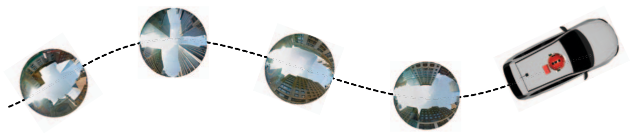
Figure 2. The vehicle's path establishes good relative pose and natural connectivity for the panoramic imagery we capture.

2. Related work

Perspective cameras and SfM for perspective cameras are well studied [14]. Generalized cameras, with arbitrary relationships between points in the world and the image, were formalized by Grossberg and Nayar [12]. A corresponding generalized epipolar constraint and optical flow are treated by Pless [22]. Rigorous efforts to solve for generalized camera models exist: 8 solutions for 3 camera rays to intersect 3 known world points [21], 64 solutions for 6 corresponding camera ray pairs [25]. The latter technique can be used with RANSAC for motion estimation between two generalized cameras, but the method is complex and ultimately unnecessary with a good relative pose prior, which we possess.

Rolling-shutter cameras are a particularly common type of generalized camera, and have a literature of their own. Moving, rolling-shutter cameras are studied and modeled by Geyer et al. [10]. Much work has been done to recover object motion and shape from rolling-shutter images and video taken from fixed viewpoints [3] [4] [20]. Complementary work has been done to remove rolling shutter distortion from images and video created when the camera moves [18] [7] [9] [5] [13].

Perhaps the most directly related work on bundle adjustment is that of Hedborg et al. [15], which aims to bundle adjust rolling-shutter video sequences. As we do, it approximates the complex rolling-shutter camera model as locally linear near points of interest. We consider static images instead of video, which makes tracking harder, but we have the advantage of starting with a better high-frequency pose prior.

Large-scale (i.e., many-camera) SfM reconstruction is a hot topic in computer vision research, with most efforts aiming to tackle ever more cameras in a single problem [24] [2]. Our approach is unlike these efforts: we deliberately limit our BA to a modestly sized (1500-camera) window. Global accuracy is achieved through loop closing as described in Section 6.

Our use of appearance-augmented 3D points is not unprecedented. Such points are employed for for image localization relative to an SfM model by Se et al. [23], Gordon & Lowe [11], Irschara et al. [16], and Li et al. [17]. In those works, each 3D point is augmented with SIFT descriptors (perhaps averaged, perhaps quantized). We similarly augment our 3D SfM points with descriptors from all views. Instead of matching an image to a 3D model, we match 3D models to each other for the purpose of loop closing.

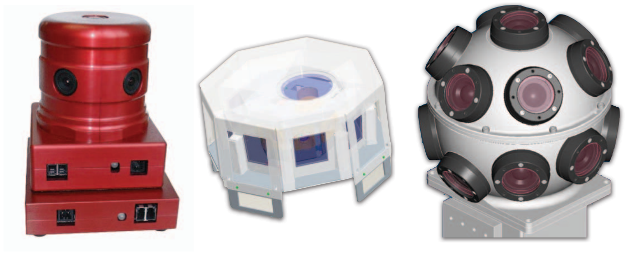
Figure 3. Example panoramic rosettes with 6, 9, and 15 cameras.

3. Generalized camera model

Most of our images come from *rolling-shutter* cameras. Rolling-shutter cameras do not capture all pixels of an image instantaneously. Instead, the exposure "rolls" across the image sensor, often column-by-column. If the camera moves while the shutter rolls, different pixels in the image have different projection centers. Moving, rolling-shutter cameras are an example of a *generalized camera* [12].

In addition, our camera rigs consist of a rigid assembly, or *rosette*, of cameras. Figure 3 shows some of our camera rigs, with rosettes composed of between 6 and 15 cameras. Figure 4 shows how the rolling shutter complicates pixel projection for the 15 camera rig. In our rigs, rosette intrinsics and rolling shutter timings are calibrated. We require a generalized camera model for these moving rosettes of calibrated rolling-shutter cameras.

Notation: Let x_a denote a point x in frame "a" and let ${}_\mathrm{b}\mathsf{T}_\mathrm{a}$ denote a possibly non-linear transform T from frame "a" to frame "b". Thus, $x_\mathrm{b} = {}_\mathrm{b}\mathsf{T}_\mathrm{a} \cdot x_\mathrm{a}$, $\mathrm{inv}\, {}_\mathrm{b}\mathsf{T}_\mathrm{a} = {}_\mathrm{a}\mathsf{T}_\mathrm{b}$, ${}_\mathrm{c}\mathsf{T}_\mathrm{b} \cdot {}_\mathrm{b}\mathsf{T}_\mathrm{a} = {}_\mathrm{c}\mathsf{T}_\mathrm{a}$, and so on.

We may write the generalized camera model as a non-

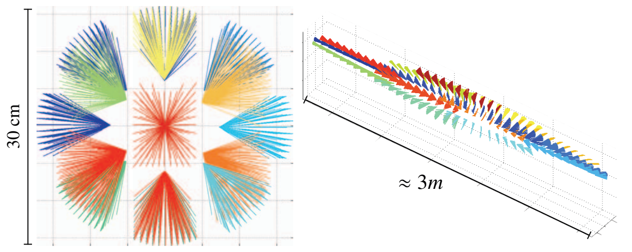

Figure 4. Left: a visualization of pixel rays from our 15-camera rosette, with each camera's rays assigned a different color. Right: the same rays when the rosette undergoes typical vehicle motion (30kph). Each pixel has a different projection center, in this case spread over several meters of vehicle trajectory. Traditional multi-view geometry algorithms do not work with such cameras.

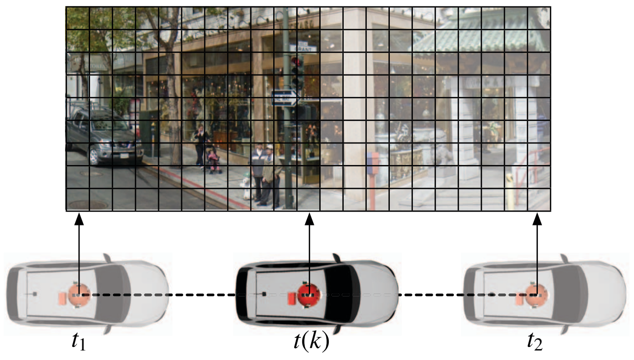

Figure 5. A rolling shutter exposure that starts at t_1 and ends at t_2. The vehicle moves so each image column may have a different projection center.

linear transform ${}_{\mathrm{im}}\mathrm{T}_{\mathrm{w}}(t)$ from the world frame "w" to the image frame "im". Due to the rolling shutter, this transform is a function of time t:

$$x_{\mathrm{im}} = {}_{\mathrm{im}}\mathrm{T}_{\mathrm{w}}(t) \cdot x_{\mathrm{w}} \tag{1}$$

We decompose the world-to-image transform in order to separate the non-linear component (lens model) from the time-dependent component (rosette pose):

$$ {}_{\mathrm{im}}\mathrm{T}_{\mathrm{w}}(t) = \overbrace{{}_{\mathrm{im}}\mathrm{T}_{\mathrm{c}}}^{\text{Lens Model}} \cdot \underbrace{{}_{\mathrm{c}}\mathrm{T}_{\mathrm{r}} \cdot \overbrace{{}_{\mathrm{r}}\mathrm{T}_{\mathrm{w}}(t)}^{\text{Rosette Pose}}}_{\text{Camera Pose}} \tag{2}$$

where "c" is the camera frame and "r" is the rosette frame. The non-linear transform ${}_{\mathrm{im}}\mathrm{T}_{\mathrm{c}}$ represents the lens model.

All frames are in motion, but the image and camera frames are fixed relative to the rosette. Thus, the only time-dependent component of the camera model is the world-to-rosette transform ${}_{\mathrm{r}}\mathrm{T}_{\mathrm{w}}(t)$ because the rosette is moving rigidly through the world frame. The quantity ${}_{\mathrm{r}}\mathrm{T}_{\mathrm{w}}(t)$ represents the 6 DOF pose of the rosette in the world frame.

The rolling shutter model relates pixel coordinates and time as some function $t(x_{\mathrm{im}})$. In our case, $t(x_{\mathrm{im}})$ is a linear function of the column index (see Figure 5).

3.1. Projection

Mapping from the image frame to the world frame is straightforward. The image point x_{im} yields an exposure time $t(x_{\mathrm{im}})$ from the rolling shutter, which in turn sets the rosette pose ${}_{\mathrm{w}}\mathrm{T}_{\mathrm{r}}$:

$$x_{\mathrm{w}} = {}_{\mathrm{w}}\mathrm{T}_{\mathrm{r}}(t(x_{\mathrm{im}})) \cdot {}_{\mathrm{r}}\mathrm{T}_{\mathrm{c}} \cdot {}_{\mathrm{c}}\mathrm{T}_{\mathrm{im}} \cdot x_{\mathrm{im}} \tag{3}$$

Projection from the world frame into the image, however, is not well defined for a generalized camera. Some world points may be imaged multiple times, whereas other world points may not be imaged at all. Figure 6 depicts this issue

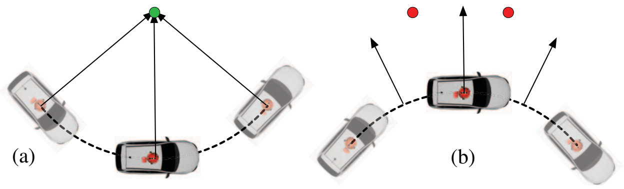

Figure 6. Because the projection center may move in a generalized camera, some points may be exposed multiple times (a) or not at all (b), making world-to-camera projection ill-defined.

geometrically. In a rolling-shutter camera in particular, *image coordinates and time are interchangeable*. The world-to-image projection equation is therefore implicit in x_{im}:

$$x_{\mathrm{im}} = {}_{\mathrm{im}}\mathrm{T}_{\mathrm{c}} \cdot {}_{\mathrm{c}}\mathrm{T}_{\mathrm{r}} \cdot {}_{\mathrm{r}}\mathrm{T}_{\mathrm{w}}(t(x_{\mathrm{im}})) \cdot x_{\mathrm{w}} \tag{4}$$

In practice, if the speed of the camera in the world is slow relative to the speed of the rolling shutter *across objects in the scene*, which is generally the case for a vehicle-mounted camera that rotates slowly, the mapping is well behaved. The circularity in the equation may be broken by estimating the exposure time and iterating on the solution. Instead, in the following sections, we show how the generalized camera model may be approximated effectively by local linearization at feature locations.

3.2. Triangulation

A fundamental operation in bundle adjustment is triangulation from multiple views. Given multiple image observations $\hat{x}^k_{\mathrm{im}}$ (superscripts hereafter dropped for clarity) of an unknown 3D world point, we wish to find the world point x_{w} that minimizes reprojection error:

$$\underset{x_{\mathrm{w}}}{\operatorname{argmin}} \sum_{\text{views}} \|x_{\mathrm{im}} - \hat{x}_{\mathrm{im}}\|^2 \ \text{ s.t. Equation 4} \tag{5}$$

The constraint is implicit in x_{im}, but we can make the constraint explicit with a simple assumption. Because rolling shutters tend to be fast, $t(x_{\mathrm{im}})$ is a slowly changing function and therefore $t(x_{\mathrm{im}}) \approx t(\hat{x}_{\mathrm{im}})$. Then we may avoid computing $t(x_{\mathrm{im}})$ entirely: the projection of $\hat{x}_{\mathrm{w}}$ into each camera may be done at the *a priori* known exposure times $t(\hat{x}_{\mathrm{im}})$ for each feature location $\hat{x}_{\mathrm{im}}$. The optimization problem is now:

$$\underset{x_{\mathrm{w}}}{\operatorname{argmin}} \sum_{\text{views}} \| \underbrace{\overbrace{{}_{\mathrm{im}}\mathrm{T}_{\mathrm{c}}}^{\text{Lens}} \cdot \overbrace{{}_{\mathrm{c}}\mathrm{T}_{\mathrm{r}} \cdot {}_{\mathrm{r}}\mathrm{T}_{\mathrm{w}}(t(\hat{x}_{\mathrm{im}}))}^{\text{Camera Pose}} \cdot x_{\mathrm{w}}}_{x_{\mathrm{im}}} - \hat{x}_{\mathrm{im}}\|^2 \tag{6}$$

which is a standard triangulation problem with known camera poses. As the rolling shutter approaches an instantaneous shutter, $t(x_{\mathrm{im}})$ becomes constant, and the equation simplifies to that of standard multi-view triangulation.

This approximation degrades as $\hat{x}_{\mathrm{im}}$ and x_{im} diverge, but, critically for optimization, *the approximation is exact at*

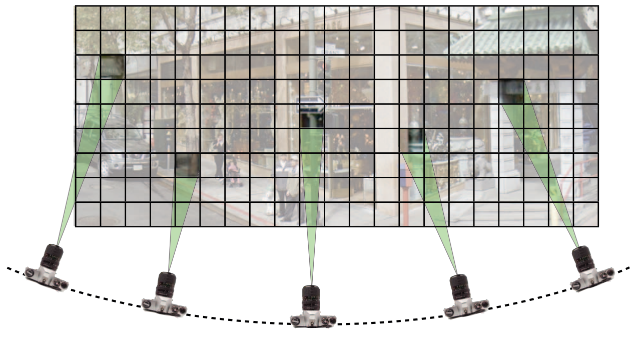

Figure 7. We establish a virtual, linear "feature camera" at each feature whose optical axis is the projection of the feature center through the generalized camera model.

the feature location. Moreover, the approximation diverges slowly in practice: For our cameras, over a very large feature of diameter 100 pixels, lens distortion varies by a few percent and pose varies by ~1 cm at 30 kph.

3.3. Feature cameras

In addition to the simplification that $t(x_{\text{im}}) \approx t(\hat{x}_{\text{im}})$, we may further simplify triangulation by linearizing the (smooth) lens model at each feature location. The monolithic, generalized camera model is thereby shattered into a constellation of simple, global-shutter, linear perspective *feature cameras*, depicted in Figure 7. This constellation would appear as a denuded porcupine,[1] like that in Figure 4 but with only the quills that survive feature tracking. To project a world point into a feature camera,

$$x_{\text{f}} = \underbrace{{}_{\text{f}}\mathsf{T}_{\text{c}} \cdot {}_{\text{c}}\mathsf{T}_{\text{r}} \cdot {}_{\text{r}}\mathsf{T}_{\text{w}}(t(\hat{x}_{\text{im}}))}_{{}_{\text{f}}\mathsf{T}_{\text{w}}} \cdot x_{\text{w}} \tag{7}$$

where the ${}_{\text{f}}\mathsf{T}_{\text{w}}$ transform is both linear and constant given the rosette pose. It is as if each feature has its own linear lens. The feature frame "f" is centered on the feature point $\hat{x}_{\text{im}}$, so if $\hat{x}_{\text{im}}$ is an exact view of x_{w}, then x_{f} is $(0,0)$. Thus, the projection coordinates x_{f} directly yield reprojection error, and the triangulation problem is simply:

$$\underset{x_{\text{w}}}{\operatorname{argmin}} \sum_{\text{views}} \|x_{\text{f}}\|^2 \quad \text{s.t. Equation 7} \tag{8}$$

To control for varying magnification across the image due to lens distortion, we sample the angular resolution of the lens model at the feature location and scale the feature camera focal length so that all feature camera frames have compatible pixel scale.

4. Generalized bundle adjustment

The previous section provides the essential elements required for bundle adjustment with general cameras. However, in contrast to global shutter cameras, the triangulation sub-problems of bundle adjustment are coupled through the camera trajectory ${}_{\text{r}}\mathsf{T}_{\text{w}}(t)$. Using feature cameras, the bundle adjustment optimization for generalized cameras is:

$$\underset{\{x_{\text{w}},\, {}_{\text{r}}\mathsf{T}_{\text{w}}(t)\}}{\operatorname{argmin}} \sum_{\text{points}} \sum_{\text{views}} \|x_{\text{f}}\|^2 \quad \text{s.t. Equation 7} \tag{9}$$

The twist with bundle adjustment using generalized cameras is that the camera trajectory influences the camera model. With traditional cameras, only the *instantaneous* pose of the camera is used (as camera extrinsics). For generalized cameras, the *trajectory* of the camera during the rolling shutter becomes part of the camera *intrinsics*. In the context of bundle adjustment and triangulation, one must choose a representation for the trajectory that constrains the feature cameras' relative poses.

In the work of Hedborg et al. [15], for rolling-shutter video, the trajectory is represented by a single key pose for each frame, with the pose during the rolling shutter linearly interpolated between successive key poses. In general, one may insert as many key poses as the tracked features support.

If a relative pose prior is available, for example from a calibrated IMU, then the prior may be used to constrain bundle adjustment in a variety of ways depending on the accuracy of the prior and the nature of the camera motion. For example, one may model low-order deviations from the relative pose prior as a way of both constraining the number of free pose variables and regularizing the bundled pose.

We use calibrated camera rigs with an IMU rigidly attached, mounted on a vehicle. From a separate pose optimization step, which is outside the scope of this paper but locally dominated by the IMU, we have accurate relative pose on the timescale of the rosette exposure. We therefore "bake in" to the camera model the relative pose that spans the rosette exposure so that bundling does not adjust the known high frequencies of pose.

To that end, we factor ${}_{\text{r}}\mathsf{T}_{\text{w}}(t(\hat{x}_{\text{im}}))$ in Equation 7 by introducing a time-dependent *nominal rosette frame* "n" and a nominal rosette exposure time t_n:

$$x_{\text{f}} = {}_{\text{f}}\mathsf{T}_{\text{c}} \cdot {}_{\text{c}}\mathsf{T}_{\text{r}} \cdot \overbrace{{}_{\text{r}}\mathsf{T}_{\text{n}}(t(\hat{x}_{\text{im}}) - t_n) \cdot {}_{\text{n}}\mathsf{T}_{\text{w}}(t_n)}^{{}_{\text{r}}\mathsf{T}_{\text{w}}(t(\hat{x}_{\text{im}}))} \cdot x_{\text{w}} \tag{10}$$

The quantity $t(\hat{x}_{\text{im}}) - t_n$ represents the rolling shutter delay from the nominal rosette exposure time t_n for pixel $\hat{x}_{\text{im}}$. Because we are using a feature camera, $t(\hat{x}_{\text{im}})$ is constant and the lens model is linear. Thus, all transform components in Equation 10 are linear. Combining the quantities that are contant during bundling yields:

$$x_{\text{f}} = \overbrace{{}_{\text{f}}\mathsf{T}_{\text{n}}(t(\hat{x}_{\text{im}}) - t_n)}^{\text{Feature Camera}} \cdot \overbrace{{}_{\text{n}}\mathsf{T}_{\text{w}}(t_n)}^{\text{Rosette Pose}} \cdot x_{\text{w}} \tag{11}$$

[1]Not shown in the expurgated version of this paper.

and during bundling we solve for the nominal rosette poses ${}_{\mathrm{n}}\mathsf{T}_{\mathrm{w}}(t_n)$ rather than for the full camera trajectory ${}_{\mathrm{r}}\mathsf{T}_{\mathrm{w}}(t)$:

$$\underset{\{x_{\mathrm{w}},\, {}_{\mathrm{n}}\mathsf{T}_{\mathrm{w}}(t_n)\}}{\operatorname{argmin}} \sum_{\text{points}} \sum_{\text{views}} \|x_{\mathrm{f}}\|^2 \;\; \text{s.t. Equation 11} \tag{12}$$

This now has the form of traditional bundle adjustment with linear cameras:

- The first term ${}_{\mathrm{f}}\mathsf{T}_{\mathrm{n}}(t(\hat{x}_{\mathrm{im}}) - t_n)$ represents the linear feature camera model for pixel $\hat{x}_{\mathrm{im}}$ in the nominal rosette frame. The relative pose of the feature camera from t_n to $t(\hat{x}_{\mathrm{im}})$ due to motion during the rolling shutter is baked into this term.

- The second term represents the 6 DOF pose of the rosette in the world frame at the constant nominal rosette exposure time t_n; this pose is a free variable.

- The third term x_{w} is the world point, also a free variable.

The result x_{f} is the projection of the world point into the feature camera for pixel $\hat{x}_{\mathrm{im}}$; because the feature camera is centered on pixel $\hat{x}_{\mathrm{im}}$, x_{f} is literally the reprojection residual.

With this approach, all the feature cameras within each panorama form a rigid assembly (porcupine) given by the relative pose prior. This assembly may move rigidly during bundling by changing the nominal rosette pose. Thus, the highest frequencies of pose are baked into the camera model; the medium frequency errors are reduced by bundling; the lowest frequency errors remain and must be addressed via loop closing. Figure 8 shows an example point cloud before and after bundle adjustment. Section 6 discusses loop closing.

5. SfM at scale with feature cameras

The previous section assembles many of the pieces required for SfM with generalized cameras, in particular the use of feature cameras and a relative pose prior for bundle adjustment. In this section, we leverage the relative pose prior for the other half of SfM, track generation.

5.1. Managing scale with vehicle paths

The scale of SfM is limited by the complexity of matching features between images to establish tracks. For n unposed proximal images, $O(n^2)$ image pairs generally must be compared to identify matches. As the density of imagery increases, this superlinear factor dominates computation, so we must sidestep it to achieve planet scale.

We capture images using vehicles outfitted with sensors in addition to cameras: accelerometers, gyroscopes, wheel speed sensors and GPS receivers. We fuse this sensor data to establish an initial trajectory for the vehicle uninformed by the imagery. This initial pose has two interesting properties:

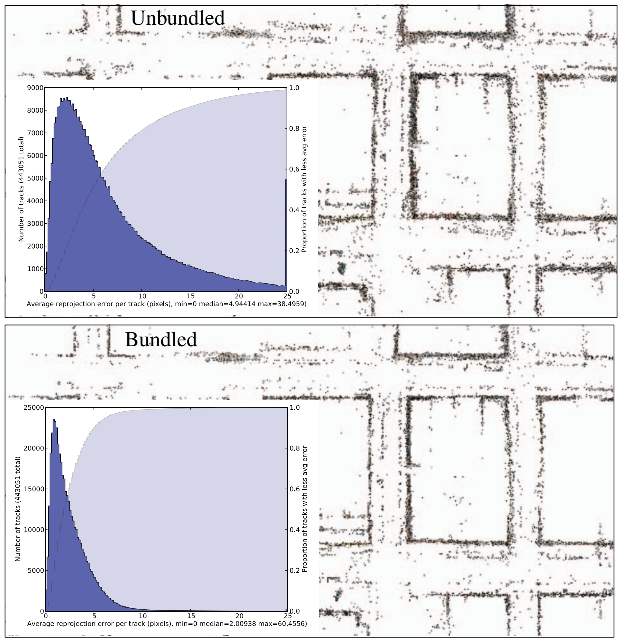

Figure 8. A comparison of an unbundled (top) to bundled (bottom) SfM point cloud, including top-down, block-level details of the clouds and inset histograms of reprojection error for all tracked feature points.

- The *absolute pose* of the vehicle path is accurate to about 10 meters in the worst case, due to multipath GPS issues in dense urban cores.

- The *relative pose* along the vehicle path is extremely accurate (sub-centimeter) because it is dominated by calibrated IMU integration over short time scales.

We leverage this second strength of our pose prior to scale our SfM reconstruction. Similar to SfM with video, the natural linear path of the vehicle trajectory establishes potential connectivity between images: only images within a fixed window of each other along the path are considered for joint participation in image tracks. This puts a constant bound on matching and tracking effort per image.

5.2. Tracking

A strong relative pose prior lets us avoid visual odometry via RANSAC-based relative pose estimation, which is often the first step in an SfM system. Instead, we get odometry from independent sensors. The relative pose is good enough for triangulation along this trajectory, so we can bootstrap the system by *tracking* features rather than by matching images.

First, we extract invariant local features similar to SIFT [19], SURF [6], and HOG [8] from each image and construct feature cameras for each feature. We link features into tracks—a track represents a 3D world point—along the trajectory incrementally. Given incoming tracks from previous panoramas in a trailing window, and given the set of features in the current panorama, a track is extended only if we have both visual and geometric confirmation:

- **Visual confirmation**: The feature visually matches a feature in the track better than any other track. Matching is strict: the two features must be mutual 1-NN matches between the two images, and they must also pass a Lowe threshold test [19] of 0.7.[2]

- **Geometric confirmation**: The track triangulates with a 3D triangulation error below 1 meter and a reprojection error below 1 degree. Triangulation is done using standard methods with the track's set of feature cameras. The pose prior permits geometric verification of matches without RANSAC-based model estimation or heuristic outlier rejection.

Further, all 2-view tracks are dropped because they are too likely subject to aliasing due to epipolar geometry. This leaves a few hundred high precision, 3+ view tracked features per panorama. We downsample images to mitigate scale; we can generate 10x as many tracked features in exchange for increased computation.

It is interesting to add that, apart from the Lowe thresholding and discarding tracks closer than 2 meters from the camera, we employ no additional thresholds to suppress false positives. There are no ad hoc heuristics employed to suppress matches on the vehicle, below the ground, on moving objects, or in the sky. The combination of visual matching with geometric confirmation to the pose prior eliminates virtually all noise because all features in a track must be similar in appearance *and* conform to the geometric dictates of the camera model and vehicle trajectory.

5.3. Augmented 3D Points

For each track we retain the image features with the triangulated 3D world point. Thus, each of our triangulated 3D world points has appearance information (in the form of scale/orientation/lighting invariant feature descriptors) for 3+ views. This is not relevant for bundle adjustment, but it is immensely useful for image-based loop closing as discussed in Section 6. We name these 3D points that retain multiview appearance information "augmented 3D points."

6. Results

Using the camera model and pose prior described in Sections 3-5, we have created an appearance-augmented, 3D SfM point cloud for a substantial subset of all Street View imagery, comprising over 404 billion tracked feature points from 1.5 trillion unique viewpoints in 9.2 billion panoramic images. The cloud, shown in Figure 1, covers a substantial portion of the world. The points are dense enough to capture street-level details such as road markings, traffic signs and business facades, as shown in Figure 11. Noise is low; the distribution of reprojection error for our tracked points is illustrated in Figure 12.

In practice, we spend about 10 seconds per 4-megapixel panorama to extract, track, and bundle adjust features on a single modern CPU core. The majority of this time is spent on pixel I/O from disk. The largest concurrent bundle adjustment problem solved is about 1500 cameras (a 100-panorama window of a 15-camera rosette). We use the open-source Ceres nonlinear least-squares solver [1] to compute our BA solution. Each window of input data is independent and many can be processed in parallel. All told, about 2000 core-years were required to compute the worldwide SfM cloud.

6.1. Loop closing with SfM constraints

The primary application of our SfM model is the correction of global pose error in our vehicle trajectory and hence our image collection. As mentioned in Section 5, GPS noise can result in global pose error of many meters, especially in dense urban areas. Figures 9a and 10a depict such error.

We can correct this error by establishing relative pose constraints between pairs of panoramas that capture overlapping views of the street-level world. First, we identify candidate panorama pairs by the proximity of their initial poses, culling pairs to limit the linear density along trajectories. Each panorama in the pair is associated with a set of augmented 3D points from our SfM model, namely, all the points that the panorama views. These points form a local "constellation" around the panorama. Using the descriptors associated with each 3D point, we build up correspondences

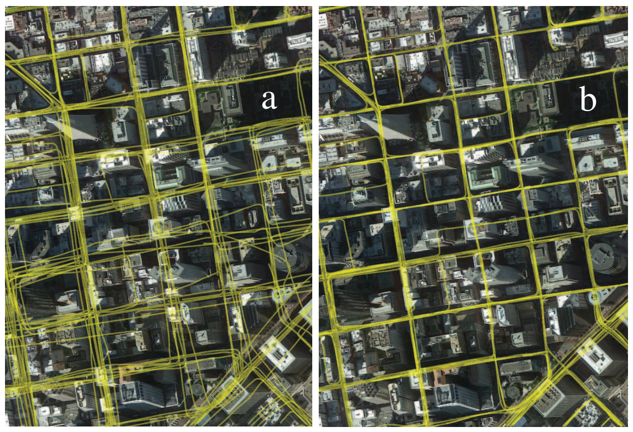

Figure 9. A comparison of vehicle paths in downtown San Francisco before (a) and after (b) SfM-based correction.

[2]We call this highly stable matching method *BFF matching*.

between the two panoramas' constellations using the same "BFF" matching scheme described in Section 5.2. We then use Umeyama's method [26] in a RANSAC loop to find the least-squares-best rigid transform that aligns the most corresponding points in the two constellations. This aligning transform yields a compact constraint describing the relative position and orientation of the two panoramas.

We repeat this process for all candidate panorama pairs, generating billions of relative pose constraints linking geographically proximal panoramas that may have been captured minutes, days, or years apart in time. We add these constraints to our sensor data and constraints derived from other sources, and repeat our optimization of the vehicle trajectories until convergence (about 8 iterations). The result, shown in Figures 9b and 10b, is a dramatic reduction in the global error of our vehicle trajectories.

How much does the error improve quantitatively? It's difficult to answer this question conclusively, as we lack ground truth pose for our vehicle trajectories. One way to get an idea, however, is to compare the residual error of the SfM constraints before and after their inclusion in the iterative optimization of vehicle trajectories. Because the SfM constraints are jointly optimized with data from over a dozen independent sources, it is unlikely that they will dominate the solution. Table 1 compares the 10th, 50th, and 90th percentile residuals for SfM constraints before and after optimization in some of the world's most challenging urban environments.

6.2. Limitations

As described in Section 3, we rely heavily on a good local pose estimate to make our complex, rolling-shutter camera systems useable for SfM. When local pose degrades—usually due to failure of inertial sensors—tracking collapses and we cannot construct our SfM model. Further, our SfM points are quite sparse (hundreds per panorama) because of downsampling and our strict checks on visual and geometric coherence. A denser model could be constructed at the cost of more computation or more noise, or both.

6.3. Conclusions

We have overcome the challenges of rolling-shutter cameras and global scale to construct an appearance-augmented SfM model of the street-level world. We have demonstrated the practical use of this model for correcting the pose of the Street View image collection. The model may be used for other applications, such as: global camera localization (by matching a query image against our point cloud), creation of appearance-enhanced 3D models of street-level scenes for navigation and annotation, and alignment of aerial and street-level imagery.

	10th percentile		50th percentile		90th percentile	
	before	after	before	after	before	after
Position	0.30 m	0.03 m	1.5 m	0.2 m	5.0 m	0.8 m
Orientation	0.05°	0.03°	0.3°	0.2°	2.0°	1.2°

Table 1. SfM constraint residual error before and after inclusion in pose optimization for the subset of trajectories in San Francisco, New York, Hong Kong, Singapore, and Seoul.

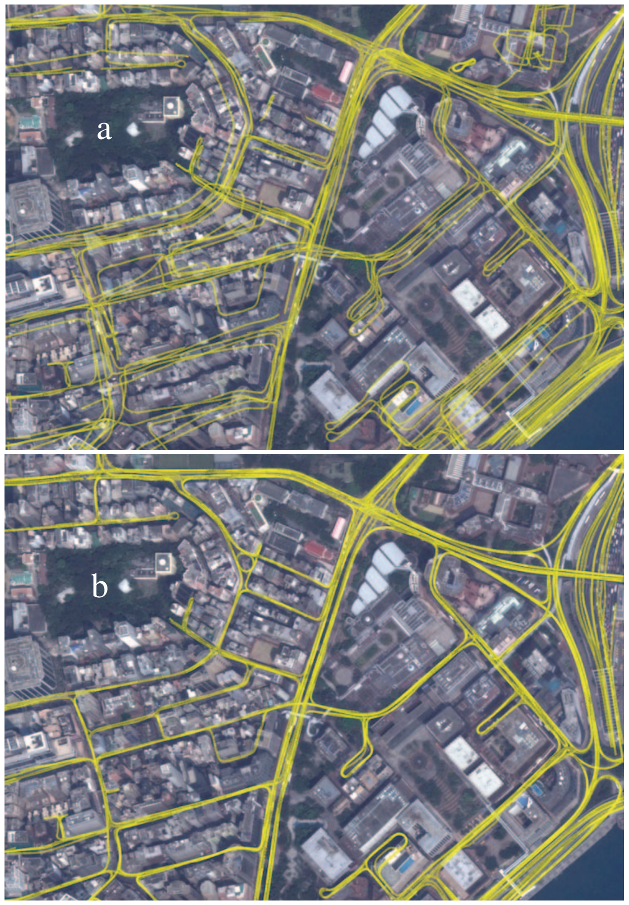

Figure 10. A comparison of vehicle paths in downtown Hong Kong before (a) and after (b) SfM-based correction.

References

[1] S. Agarwal, K. Mierle, and Others. Ceres solver. https://code.google.com/p/ceres-solver/. 6

[2] S. Agarwal, N. Snavely, I. Simon, S. M. Seitz, and R. Szeliski. Building rome in a day. In *ICCV*, pages 72–79. IEEE, 2009. 2

[3] O. Ait-Aider, A. Bartoli, and N. Andreff. Kinematics from lines in a single rolling shutter image. In *CVPR*, 2007. 2

[4] O. Ait-Aider and F. Berry. Structure and kinematics triangulation with a rolling shutter stereo rig. In *ICCV*, pages 1835–1840, 2009. 2

[5] S. Baker, E. Bennett, S. B. Kang, and R. Szeliski. Removing rolling shutter wobble. In *CVPR*, pages 2392–2399, 2010. 2

[6] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. Speeded-up robust features (surf). *Comput. Vis. Image Underst.*, 110(3):346–359, June 2008. 5

[7] D. Bradley, B. Atcheson, I. Ihrke, and W. Heidrich. Synchronization and rolling shutter compensation for consumer video camera arrays. In *CVPR Workshops*, pages 1–8, 2009. 2

[8] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, CVPR '05, pages 886–893, Washington, DC, USA, 2005. IEEE Computer Society. 5

[9] P. Forssen and E. Ringaby. Rectifying rolling shutter video from hand-held devices. In *CVPR*, pages 507–514, 2010. 2

[10] C. Geyer, M. Meingast, and S. Sastry. Geometric models of rolling-shutter cameras. *OMNIVIS*, 2005. 2

[11] I. Gordon and D. G. Lowe. What and where: 3d object recognition with accurate pose. In *CLOR06*, pages 67–82, 2006. 2

[12] M. D. Grossberg and S. K. Nayar. A general imaging model and a method for finding its parameters. In *ICCV*, pages 108–115, 2001. 2

[13] M. Grundmann, V. Kwatra, D. Castro, and I. Essa. Calibration-free rolling shutter removal. In *ICCP*, pages 1–8, 2012. 2

[14] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2nd edition, 2003. 2

[15] J. Hedborg, P.-E. Forssén, M. Felsberg, and E. Ringaby. Rolling shutter bundle adjustment. In *CVPR*, pages 1434–1441. IEEE, 2012. 2, 4

[16] A. Irschara, C. Zach, J.-M. Frahm, and H. Bischof. From structure-from-motion point clouds to fast location recognition. In *CVPR*, pages 2599–2606, 2009. 2

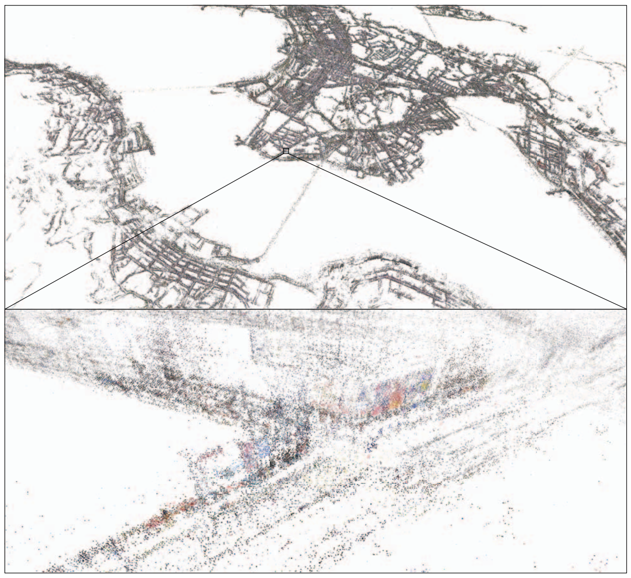

Figure 11. Our point cloud in downtown Hong Kong (top), with a detail (bottom) showing street-level features like road markings, building facades, and billboards.

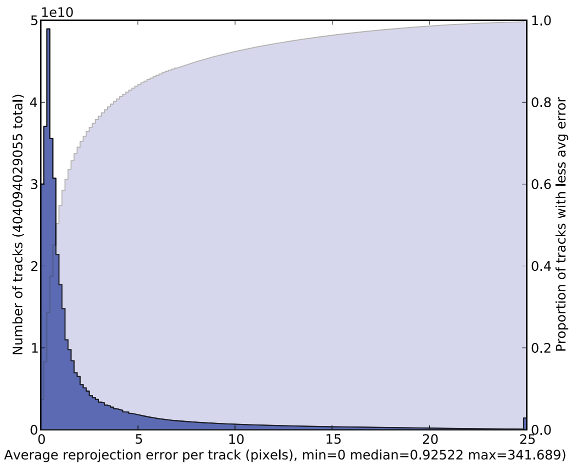

Figure 12. A histogram of reprojection error, in pixels, for each of the 404 billion tracked features in our augmented point cloud. We downsample our imagery so that it all shares a common angular resolution of about 8 pixels per degree.

[17] Y. Li, N. Snavely, and D. P. Huttenlocher. Location recognition using prioritized feature matching. In *ECCV*, ECCV'10, pages 791–804, Berlin, Heidelberg, 2010. Springer-Verlag. 2

[18] C.-K. Liang, L.-W. Chang, and H. H. Chen. Analysis and compensation of rolling shutter effect. *IEEE Transactions on Image Processing*, 17(8):1323–1330, 2008. 2

[19] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60:91–110, 2004. 5, 6

[20] L. Magerand and A. Bartoli. A generic rolling shutter camera model and its application to dynamic pose estimation. In *3DPVT*, 2010. 2

[21] D. Nistér and H. Stewénius. A minimal solution to the generalized 3-point pose problem. *Journal of Mathematical Imaging and Vision*, 2006. 2

[22] R. Pless. Using many cameras as one. In *CVPR*, pages 587–593, 2003. 2

[23] S. Se, D. Lowe, and J. Little. Vision-based mobile robot localization and mapping using scale-invariant features. In *Proceedings of ICRA 2001.*, volume 2, pages 2051–2058 vol.2, 2001. 2

[24] N. Snavely, S. M. Seitz, and R. Szeliski. Photo tourism: exploring photo collections in 3d. In *ACM transactions on graphics*, volume 25, pages 835–846. ACM, 2006. 2

[25] H. Stewénius, D. Nistér, M. Oskarsson, and K. Åström. Solutions to minimal generalized relative pose problems. In *Workshop on Omnidirectional Vision*, 2005. 2

[26] S. Umeyama. Least-squares estimation of transformation parameters between two point patterns. *Pattern Analysis and Machine Intelligence*, 13(4):376–380, 1991. 7