In Proc. Computer Vision and Pattern Recognition, 1998 (with corrected errata).

Acquisition of a Large Pose-Mosaic Dataset*

Satyan Coorg Neel Master Seth Teller
Computer Graphics Group
MIT Laboratory for Computer Science
{satyan,neel,seth}@graphics.lcs.mit.edu

Abstract

We describe the generation of a large pose-mosaic dataset: a collection of several thousand digital images, grouped by spatial position into spherical mosaics, each annotated with estimates of the acquiring camera's 6 DOF pose (3 DOF position and 3 DOF orientation) in an absolute coordinate system.

The pose-mosaic dataset was generated by acquiring images, grouped by spatial position into nodes (essentially, spherical mosaics). A prototype mechanical pan-tilt head was manually deployed to acquire the data. Manual surveying provided initial position estimates for each node. A back-projecting scheme provided initial rotational estimates. Relative rotations within each node, along with internal camera parameters, were refined automatically by an optimization-correlation scheme. Relative translations and rotations among nodes were refined according to point correspondences, generated automatically and by a human operator. The resulting pose-imagery is self-consistent under a variety of evaluation metrics.

Pose-mosaics are useful "first-class" data objects, for example in automatic reconstruction of textured 3D CAD models which represent urban exteriors.

1 Introduction

Automatic reconstruction of textured 3D CAD models representing urban environments is an important goal of computer vision systems, and more recently of computer graphics systems. Developing such a system will require the acquisition and processing of a suitable dataset. Our approach involves *annotating* each acquired image with an estimate of 6-DOF *pose* – 3 DOF of position, and 3 DOF of orientation for the acquiring camera – through the use of instrumentation.

*Funding for this research was provided in part by the National Science Foundation under award IRI-9501937, the Advanced Research Project Agency under contract DABT63-95-C-0009, Intel Corporation and the MIT Lincoln Laboratories.

There are several advantages of such a scheme over more traditional methods (e.g., a collection of many high-resolution images). First, pose estimates are useful in organizing the images in a spatial data structure which supports proximity queries. This restricts processing to images only when they are close together, or otherwise likely to exhibit significant correlation. Second, pose information provides useful geometric constraints that can the reconstruction process. Third, the availability of *a priori* pose information prevents the computational efforts expended by the reconstruction system from growing quadratically (or worse) in the number of input images.

Existing physical instrumentation alone does not produce pose estimates sufficiently accurate for direct incorporation into reconstruction algorithms which demand pixel-level agreement between features. For example, most pan-tilt heads used to measure rotation report rotation angles accurate only to about a degree. This is one to two orders of magnitude greater than would be required by an algorithm demanding pixel-level agreement. Thus, we have developed a variety of "pose-refinement" algorithms which revise the pose-estimates based on correspondence and correlation among two or more images.

1.1 Related Work

Our work builds upon many fundamental techniques from photogrammetry and computer vision:

- *Aerial imagery* [3, 7]: Photogrammetric systems that analyze aerial imagery use pose obtained by a combination of instrumentation, manual input, and "bundle-adjustment" optimization. Acquiring similar ground-based imagery poses additional challenges: to provide a large field of view to capture all visible features and to refine initial pose estimates to predict nearby features accurately.

- *Mosaicing* [13, 20]: These techniques seamlessly stitch together multiple images taken from the same viewpoint. While such mosaics are typically used in virtual environments, we apply mosaicing techniques to refine our pose estimates and to provide a much larger field of view than would a single image. In addition, we consider the problem of registering multiple mosaics in a global coordinate system.

- *Structure from Motion* [14, 16, 18]: These techniques recover both scene structure and camera motion by analyzing correspondences in a closely-spaced image sequence (e.g., frames from a video sequence). While these techniques correlate nearby images in the sequence, significant analysis must be performed to relate images that are farther apart.

The novel ideas presented in this paper are:

- Acquisition of close-range, ground-based imagery of urban structures, annotated with absolute 6DOF pose estimates.
- Automatic correlation-based generation of spherical mosaics.
- The use of a world-space notion of adjacency to relate mosaics.
- Decoupling scene reconstruction from camera pose estimation, and using pose estimates both for initializing optimization procedures and for semi-automatic identification of feature correspondences across mosaics.

1.2 Acquisition of Raw Images and Nodes

Our dataset consists of photographs acquired by a Kodak DCS 420 digital camera mounted with fixed optical center on an indexed pan-tilt head, itself attached to a tripod base. The tripod was manually positioned at eighty-one locations among the buildings of an office complex. At each position, the camera was rotated through a predetermined "tiling" of 50-70 orientations, yielding a roughly hemispherical field of view. We call a set of images obtained from a common optical center a *node*. Initial translation estimates for each node were obtained with surveying instruments. Initial orientation estimates for each node were obtained by manual pointing of the pan-tilt head at some other node (marked by a second tripod and orange ball). These orientation estimates are expressed and manipulated as *quaternions*, which possess useful stability properties for optimization [6].

1.3 Overview

The rest of the paper is organized as follows. Section 2 describes an automatic spherical mosaicing algorithm that accurately computes relative rotations between images taken from the same position using a quaternion-based correlation maximization algorithm. Section 3 describes correspondence and optimization techniques for a global registration algorithm that uses these pose-mosaics as fundamental data objects, and Section 4 concludes.

2 Automatic Spherical Mosaicing

In this section, we use the simple relation between images taken from the same nodal point to refine relative orientations. As depth/parallax effects do not occur across images taken from a single node, a constrained 2-D projective transformation (collineation) describes the relation between any such pair of images [5]. Computing and refining the projective transformations using correlation of pixel-values is the basis of our approach to refine orientation estimates.

Our approach is closely related to the mosaicing algorithm proposed by Szeliski [13] and extended to cylindrical panoramas by McMillan [8]. Unlike McMillan [8], who computes a cylindrical panoramic image from a set of images taken with rotation around a single axis, we compute a spherical panoramic image. Szeliski & Shum in their recent papers [11, 15] also compute full-view panoramas. However, their global alignment algorithm requires a combination of both correlation-based and feature-based optimization. In contrast, we directly optimize correlation to perform global alignment, avoiding the step of identifying suitable features.

The approach we follow is to optimize a global correlation function defined for adjacent images with respect to all orientations (represented as quaternions). In addition, the algorithm also revises internal camera parameters (camera focal length and image center initially estimated using Tsai's calibration algorithm [19]) to maximize correlation. The result of the optimization is a *spherical mosaic*, a composite of all images corresponding to a single node.

The basis of the optimization is the 2-D projective transformation $\mathbf{P}_{12}$ between two images (labeled 1 and 2) taken from the same camera [5]:

$$\mathbf{P}_{12} = \mathbf{K}\mathbf{R}_2\mathbf{R}_1^{-1}\mathbf{K}^{-1}$$

which maps pixels from image 1 to pixels in image 2 by a 2-D projective transformation ($\cong$ denotes projective equality):

$$\begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix} \cong \mathbf{P}_{12} \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix}$$

where (x_1, y_1) and (x_2, y_2) are pixel positions in images 1 and 2; $\mathbf{K}$ is the 3×3 upper triangular camera-calibration matrix; and $\mathbf{R}_1$ and $\mathbf{R}_2$ are 3×3 rotation matrices.

Our algorithm uses the above relation to minimize the function:

$$C = \sum_{i,j \text{ are adjacent}} C_{ij} + C_{ji}$$

where C_{ij} is the sum-of-squared-difference error between luminance values[1] of the images under the mapping $\mathbf{P}_{ij}$:

$$C_{ij} = \sum_{x_i, y_i} (L_i(x_i, y_i) - L_j(\mathbf{P}_{ij}(x_i, y_i)))^2$$

The correlation function is computed only for pairs of adjacent images in the spherical tiling, and only for pixels of image i that map to a valid pixel of image j. This function is minimized by computing derivatives with respect to the orientations and using Levenberg-Marquadt nonlinear optimization [10] starting from the initial orientations and internal parameters. The optimization involves updating these unknowns with increments computed using the gradient $\mathbf{G}$ and (an approximation to) the Hessian $\mathbf{H}$ of the objective function, until convergence is achieved. The various steps in the computation are described in greater detail below.

For a single error term of the form:

$$e_{x,y}^2 = (L(x, y) - L'(x'', y''))^2$$

where:

$$x'' = \frac{x'}{z'} \quad y'' = \frac{y'}{z'}$$

and

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \mathbf{v}' = \mathbf{P} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \mathbf{K}\mathbf{R}'\mathbf{R}^{-1}\mathbf{K}^{-1} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

the derivatives are computed as follows (the rotation-matrix derivative is given in the appendix):

$$\frac{\partial \mathbf{v}'}{\partial \mathbf{q}} = \mathbf{K}\mathbf{R}'(\frac{\partial \mathbf{R}^{-1}}{\partial \mathbf{q}})\mathbf{v}$$

[1]In practice, we perform the optimization on band-pass luminance values instead of luminance values first, and then perform the optimization on luminance values. This results in both faster convergence and increased avoidance of false-minima.

where $\mathbf{v} = \mathbf{K}^{-1}[x, y, 1]^T$. Then, the derivative of the term $e_{x,y}$ with respect to the quaternion $\mathbf{q}$ is given by:

$$\frac{\partial x''}{\partial \mathbf{q}} = \frac{\frac{\partial x'}{\partial \mathbf{q}} - x'' \frac{\partial z'}{\partial \mathbf{q}}}{z'} \quad \frac{\partial y''}{\partial \mathbf{q}} = \frac{\frac{\partial y'}{\partial \mathbf{q}} - y'' \frac{\partial z'}{\partial \mathbf{q}}}{z'}$$

$$\frac{\partial e_{x,y}}{\partial \mathbf{q}} = \frac{\partial L'}{\partial x''} \frac{\partial x''}{\partial \mathbf{q}} + \frac{\partial L'}{\partial y''} \frac{\partial y''}{\partial \mathbf{q}} \tag{1}$$

Note that Equation 1 involves image derivatives $\frac{\partial L'}{\partial x''}$ and $\frac{\partial L'}{\partial y''}$; these are approximated using finite differences.

The gradient term corresponding to the quaternion q_i is computed using Equation 1 by accumulating over all terms that depend on $\mathbf{q}_i$:

$$\mathbf{G}_i = \sum_{x,y} e_{x,y} \frac{\partial e_{x,y}}{\partial \mathbf{q_i}}$$

Similarly, the Hessian term corresponding to two adjacent images i and j is:

$$\mathbf{H}_{ij} = \sum_{x,y} \frac{\partial e_{x,y}}{\partial \mathbf{q}_i} (\frac{\partial e_{x,y}}{\partial \mathbf{q}_j})^T$$

For a spherical tiling consisting of n images, values of $\mathbf{G}_i$ and $\mathbf{H}_{ij}$ are concatenated to yield the global $1 \times 4n$ gradient $\mathbf{G}$ and the global $4n \times 4n$ Hessian $\mathbf{H}$.

In an unconstrained optimization, the increments would be computed as $-\mathbf{H}^{-1}\mathbf{G}$ [10]. However, as quaternions are constrained to be unit vectors, the following additional constraints involving the increments $\delta\mathbf{q}_i$ are necessary:

$$\forall i : \mathbf{q}_i \cdot \delta\mathbf{q}_i = 0$$

Using Lagrange multipliers λ_i to enforce the above constraints, the equation for computing the increments becomes:

$$\begin{bmatrix} \mathbf{H} & \mathbf{Q} \\ \mathbf{Q}^T & \mathbf{0} \end{bmatrix} \begin{bmatrix} \Delta\mathbf{Q} \\ \Lambda \end{bmatrix} = - \begin{bmatrix} \mathbf{G} \\ \mathbf{0} \end{bmatrix} \tag{2}$$

where:

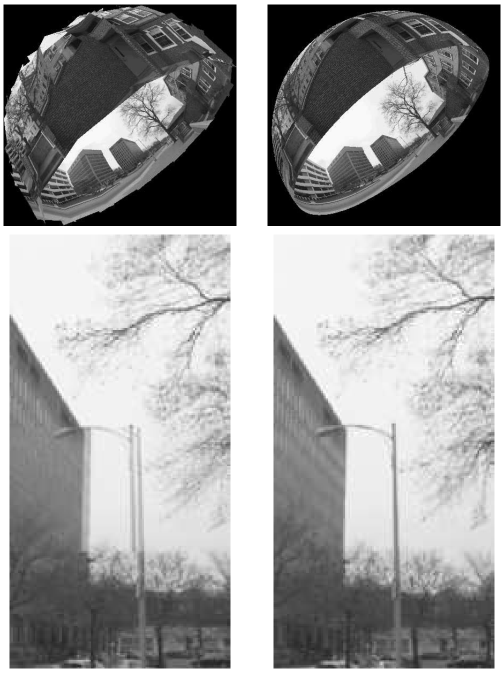
$$\mathbf{Q} = \begin{bmatrix} \mathbf{q}_1 & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \mathbf{q}_2 & \dots & \mathbf{0} \\ \vdots & \vdots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \dots & \mathbf{q}_n \end{bmatrix}, \Delta\mathbf{Q} = \begin{bmatrix} \delta\mathbf{q}_1 \\ \delta\mathbf{q}_2 \\ \vdots \\ \delta\mathbf{q}_n \end{bmatrix}, \Lambda = \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_n \end{bmatrix}$$

The optimization proceeds by solving Equation 2 for $\Delta\mathbf{Q}$ and Λ using a linear-solver[2], and then updating quaternion $\mathbf{q}_i$ to its new value (with re-normalization):

$$\frac{\mathbf{q}_i + \delta\mathbf{q}_i}{\|\mathbf{q}_i + \delta\mathbf{q}_i\|}$$

[2]Sparse matrix techniques can be used to improve the speed of the solver. However, as most of the computational effort is expended in computing the entries of $\mathbf{H}$, this optimization is not worthwhile.

Convergence in the procedure is detected when the objective function changes by less than some threshold (e.g., 0.1%).



(a) (b)

Figure 1: Part (a) shows the input images. Part (b) shows the results of the mosaicing optimization mapped onto a sphere. Details (below) show the removal of blurring after optimization.

Figure 1 shows one mosaic resulting from our optimization. Note that the input images are seamlessly blended, without any "blurring" or "ghosting" artifacts. In a batch process, our algorithm was able to successfully mosaic all (close to four thousand) input images [3] into eighty-one nodes, requiring about twenty minutes of processing per node on a 150MHz SGI O2 workstation. Internal camera parameters converged to values that differed by less than 1% across all nodes.

Grouping images into spherical mosaics has several benefits. First, it allows robust automatic estimation of internal camera parameters. Second, it effectively produces an image with a spherical field of view, eliminating the ambiguity between camera motion and camera rotation found in narrow field of view images. Third, it reduces, by a factor of about fifty, the number of degrees of freedom when optimizing global positions and orientations. All of these are important engineering advantages.

[3] A few input images were unusable due to sun flare and/or CCD oversaturation.

3 Global Mosaic Registration

The problem addressed in this section is registering initial camera pose estimates in a global coordinate system so that the image locations of 3-D points are known accurately, to within a pixel. The input to this stage is a set of nodes with estimated camera orientations and positions in a global coordinate system.

In contrast to techniques that work in projective space (e.g., [9]), we designed our algorithm to operate in Euclidean space for the following reasons. First, this makes full use of available camera-calibration and pose information. Second, the use of the Euclidean framework decreases the complexity of the optimization by eliminating extra projective variables. Third, from a practical standpoint, it is much easier to visualize and debug (using computer graphics) algorithms operating in 3-D than algorithms that operate in projective space.

For two views, registration is equivalent to computing relative orientation as in classical photogrammetry [6]. While this technique performs well for pairs of images, a disadvantage of using this approach is that computing only pairwise pose may result in global inconsistency. Instead, we use global optimization to refine pose estimates, similar to the techniques proposed in structure-from-motion [14, 16]. These approaches use correspondences (either manually specified [16] or obtained by tracking [14]) between image features to set up a global objective function, and perform an optimization using non-linear methods.

Though the problem of automatically generating correspondences is well studied in computer vision, the process tends to be very fragile, especially across disparate images. The large inter-node distance in our dataset (with baselines of tens of meters) produces fairly dissimilar images due to perspective and occlusion effects. In addition, different nodes are acquired in very different lighting conditions, further accentuating the dissimilarity.

Fortunately, as we are interested primarily in recovering accurate pose for each node, very few correspondences are necessary (five points [6] per *mosaic*). Thus, our system allows a user to manually correspond points computed by intersecting adjacent straight edges obtained by using the Canny edge detector [1]. The user's task is further simplified by our system, which uses the available pose information to generate matches automatically in most cases. The matching technique is described in Section 3.2, after the optimization described in Section 3.1.

3.1 Optimization

Formally, the pose refinement problem is as follows. Given:

- For $1 \leq i \leq m$, $\mathbf{p}'_i$ (position) and $\mathbf{q}'_i$ (orientation) estimates of the i^{th} camera pose;
- For $1 \leq i \leq m$, $1 \leq j \leq n$, unit vectors $\mathbf{r}_{ij}$ (in camera i's coordinate system) describing rays through the image feature corresponding to world-space point $\mathbf{s}_j$.

Compute:

- $\mathbf{p}_i$, $\mathbf{q}_i$ for $1 \leq i \leq m$ (the true camera poses);
- $\mathbf{s}_j$ for $1 \leq j \leq n$ (the true 3-D feature positions).

The pose-refinement algorithm performs a global optimization that refines *both* position and orientation estimates. Our approach is to use the Levenberg-Marquadt optimization to minimize the objective function described below.

Let $\mathbf{u}_{ij} = \frac{\mathbf{s}_j - \mathbf{p}_i}{\|\mathbf{s}_j - \mathbf{p}_i\|}$ represent the world-space unit vector directed from camera i to point $\mathbf{s}_j$. The same vector predicted by using the image feature is $\mathbf{v}_{ij} = \mathbf{R}_i^{-1}\mathbf{r}_{ij}$, where $\mathbf{R}_i$ is the 3×3 rotation matrix equivalent to $\mathbf{q}_i$. Their difference in the spherical image, $\mathbf{e}_{ij} = \mathbf{u}_{ij} - \mathbf{v}_{ij}$, is the residual error vector of predicting ray $\mathbf{u}_{ij}$. The objective function O is simply the sum of the squared magnitudes of the residual vectors:

$$O = \sum_{i=1}^{m} \sum_{j=1}^{n} \|\mathbf{e}_{ij}\|^2$$

To perform the optimization, we use an approach similar to "bundle-adjustment" in photogrammetry [12] that alternately refines 3-D positions and pose estimates. The advantage of this method is that fixing the 3-D positions of point features decouples the optimization of the various camera poses, each of which can be independently updated (and vice-versa). Thus, an efficient implementation of the optimization involves inverting only small constant-sized matrices. Our experiments show that the optimization requires only a few iterations, making it usable in an interactive system.

The derivatives of a single residual term (omitting subscripts) are computed as follows. Let $\mathbf{I}$ be the 3×3 identity matrix. Then, the gradient $\mathbf{G_s}$ and Hessian $\mathbf{H_s}$ with respect to a 3-D point $\mathbf{s}$ are:

$$\frac{\partial \mathbf{e}}{\partial \mathbf{s}} = \frac{\mathbf{I} - \mathbf{u}\mathbf{u}^T}{\|\mathbf{s} - \mathbf{p}\|}$$

$$\mathbf{G_s} = \left(\tfrac{\partial \mathbf{e}}{\partial \mathbf{s}}\right)^T \mathbf{e} \quad \mathbf{H_s} = \left(\tfrac{\partial \mathbf{e}}{\partial \mathbf{s}}\right)^T \tfrac{\partial \mathbf{e}}{\partial \mathbf{s}} \tag{3}$$

The gradient $\mathbf{G_{p,q}}$ and Hessian $\mathbf{H_{p,q}}$ with respect to pose parameters $\mathbf{p}$ and $\mathbf{q}$ are:

$$\tfrac{\partial \mathbf{e}}{\partial \mathbf{p}} = \tfrac{\mathbf{u}\mathbf{u}^T - \mathbf{I}}{\|\mathbf{s}-\mathbf{p}\|} \quad \tfrac{\partial \mathbf{e}}{\partial \mathbf{q}} = -\left(\tfrac{\partial \mathbf{R}^{-1}}{\partial \mathbf{q}}\right)_{\mathbf{r}} \quad \mathbf{D} = \left[\tfrac{\partial \mathbf{e}}{\partial \mathbf{p}}\ \tfrac{\partial \mathbf{e}}{\partial \mathbf{q}}\right]$$

$$\mathbf{G_{p,q}} = \mathbf{D}^T\mathbf{e} \quad \mathbf{H_{p,q}} = \mathbf{D}^T\mathbf{D} \tag{4}$$

To update the position of a 3-D point $\mathbf{s}_j$, the gradient $\mathbf{G}_{\mathbf{s}_j}$ and the Hessian $\mathbf{H}_{\mathbf{s}_j}$ are computed by accumulating Equation 3 over all cameras that "see" $\mathbf{s}_j$. To update the pose $\mathbf{p}_i, \mathbf{q}_i$ of camera i, the gradient $\mathbf{G}_{\mathbf{p}_i,\mathbf{q}_i}$ and Hessian $\mathbf{H}_{\mathbf{p}_i,\mathbf{q}_i}$ are computed by accumulating Equation 4 over all points visible to camera i. As in Section 2, Lagrange multipliers are used to enforce the unitary constraint for quaternions.

Typical residuals after optimization are about 0.001, equivalent to 0.05 degrees of error in predicting a ray, and to about 2.5 cm of error in predicting the location of a 3-D point about 25 meters from the camera.

3.2 Automatic Matching

In this section, we describe a technique that exploits the availability of pose-information to aid the user in establishing correspondences across nodes. The basic idea, similar to that proposed by Collins [2], is as follows. If any *sparse* set of points in a set of nodes is projected into 3-D rays, regions with high *incidence* of rays correspond to likely locations of 3-D features. If a 3-D feature is present in multiple nodes, then rays through the corresponding 2-D points pass near the 3-D feature, increasing its incidence count. Conversely, as the set of points are sparse, it is rare that unrelated rays pass through the same 3-D region by chance.

The matching algorithm first defines a notion of adjacency using a 2-D Delaunay triangulation of node positions projected onto the ground plane[4]. Given a nearness threshold T, the match for a selected point $\mathbf{r}$ of node i is generated as follows:

For each $\mathbf{s}_j$ in decreasing order of incidence
    If $\|\mathbf{e}_{ij}\| < T$ associate $\mathbf{r}$ with point $\mathbf{s}_j$.
If $\mathbf{r}$ is not matched
    Let minimum error $e = T$, closest 3-D point $\mathbf{s} = \phi$.
    For each Delaunay neighbor node i' of i
        For each unmatched point $\mathbf{r}'$ of node i'
            Generate $\mathbf{s}_k$ closest to rays $\mathbf{r}$ and $\mathbf{r}'$.
            If $\|\mathbf{e}_{i'k}\| < e$ then $e = \|\mathbf{e}_{i'k}\|$; $\mathbf{s} = \mathbf{s}_k$.
    If $\mathbf{s} \neq \phi$ add $\mathbf{s}$ to the 3-D point set.

[4]This spatial notion of adjacency is necessary as the usual temporal notion used in structure-from-motion is not applicable to our dataset, where nodes acquired at very different times can observe nearby 3-D regions.

Informally, given a newly selected point $\mathbf{r}$, the algorithm first searches for an existing high incidence 3-D point $\mathbf{s}_j$ that projects close to $\mathbf{r}$. If such a point is not found, the algorithm matches $\mathbf{r}$ with a close unmatched point $\mathbf{r}'$ from a neighboring node i'.

Figure 3 shows a snapshot of the global registration algorithm in action. To test this matching algorithm on our dataset, a user interactively selected five or more points (typically, building corners) in each mosaic. The algorithm automatically inferred sufficient number of matches to generate pose for over 95% of the nodes; for the remaining nodes, the user established correspondences manually. The user required about one hour to process the entire dataset (eighty-one mosaics comprising nearly four thousand images).

4 Conclusion

In this paper, we described the process of acquiring pose-mosaics, utilizing two techniques that compute accurate pose from approximate estimates. Our algorithms are simple and robust, and scale to a large set of input images.

Preliminary results indicate that our dataset can be used to reconstruct urban vertical facades [4]. Figure 2 shows some initial reconstruction results, co-located with the input spherical mosaics.

We are currently building a pose image acquisition platform using instrumentation such as Global Positioning System (GPS), inertial units, etc. [17]. The acquisition platform should enable collection of even larger pose-mosaic datasets. Also, we are investigating fully automatic matching techniques for use in the inter-mosaic registration procedure. A promising approach, exploiting domain-specific knowledge, would be to automatically detect *sky-lines* in the input nodes, and attempt to match only features of the sky-lines.

References

[1] F. J. Canny. A computational approach to edge detection. *IEEE Trans PAMI*, 8(6):679–698, 1986.

[2] R. Collins. A space-sweep approach to true multi-image matching. In *CVPR96*, pages 358–363, 1996.

[3] R. Collins, C. Jaynes, F. Stolle, X. Wang, Y. Cheng, A. Hanson, and E. Riseman. A system for automated site model acquisition. In *SPIE Proceedings Vol. 7617*, 1995.

[4] S. Coorg and S. Teller. Automatic extraction of textured vertical facades from pose imagery. Technical Report TR-729, Laboratory for Computer Science, MIT, 1998.

[5] R. Hartley. Self-calibration of stationary cameras. *IJCV*, 22(1):5–23, 1997.

[6] B. K. P. Horn. Relative orientation revisited. *Journal of the Optical Society of America A*, 8(10):1630–1638, October 1991.

[7] D. M. McKeown, C. McGlone, S. D. Cochran, Y. C. Hsieh, M. Roux, and J. Shufelt. Automatic cartographic feature extraction using photogrammetric principles. In *Digital Photogrammetry*, pages 195–212. ASPRS, 1997.

[8] L. McMillan and G. Bishop. Plenoptic modeling: An image-based rendering system. In *SIGGRAPH '95 Conference Proceedings*, pages 39–46, Aug. 1995.

[9] R. Mohr, F. Veillon, and L. Quan. Relative 3D reconstruction using multiple uncalibrated images. In *CVPR93*, pages 543–548, 1993.

[10] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C: The Art of Scientific Computing (2nd ed.)*. Cambridge University Press, Cambridge, 1992.

[11] H.-Y. Shum and R. Szeliski. Panoramic image mosaics. Technical Report MSR-TR-97-23, Microsoft Research, 1997.

[12] C. Slama. *Manual of Photogrammetry*. American Society of Photogrammetry and Remote Sensing, 1980.

[13] R. Szeliski. Video mosaics for virtual environments. *IEEE Computer Graphics and Applications*, 16(2):22–30, Mar. 1996.

[14] R. Szeliski and S. B. Kang. Recovering 3D shape and motion from image streams using non-linear least squares. *Journal of Visual Communication and Image Representation*, 5(1):10–28, 1994.

[15] R. Szeliski and H. Shum. Creating full-view panoramic mosaics and texture-mapped 3D models. In *SIGGRAPH '97 Conference Proceedings*, pages 251–258, Aug. 1997.

[16] C. J. Taylor and D. J. Kriegman. Structure and motion from line segments in multiple images. *PAMI*, 17(11):1021–1032, November 1995.

[17] S. Teller. Automatic acquisition of hierarchical, textured 3D geometric models of urban environments: Project plan. In *Proc. DARPA IUW*, May 1997.

[18] C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: a factorization method. *International Journal of Computer Vision*, 9:137–154, 1992.

[19] R. Tsai. A versatile camera calibration technique for high accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses. *IEEE Journal of Robotics and Automation*, RA-3(4), Aug. 1987.

[20] I. Zoghiami, O. Faugeras, and R. Deriche. Using geometric corners to build a 2D mosaic from a set of images. In *CVPR97*, pages 420–425, 1997.

A Rotation-Matrix Derivative

The derivative of a rotation matrix $\mathbf{R}^{-1}$ with respect to a quaternion $\mathbf{q} = [q_0, q_x, q_y, q_z]^T$ is a tensor of dimension $3 \times 4 \times 3$. Since the derivative is typically multiplied by a vector (say, $\mathbf{v} = [v_x, v_y, v_z]^T$), only its value at $\mathbf{v}$, a 3×4 matrix, is given below:

$$\begin{aligned}
a &= +q_0 v_x + q_z v_y - q_y v_z \\
b &= -q_z v_x + q_0 v_y + q_x v_z \\
c &= +q_y v_x - q_x v_y + q_0 v_z \\
d &= +q_x v_x + q_y v_y + q_z v_z \\
(\frac{\partial \mathbf{R}^{-1}}{\partial \mathbf{q}})_{\mathbf{v}} &= \begin{bmatrix} a & d & -c & b \\ b & c & d & -a \\ c & -b & a & d \end{bmatrix}
\end{aligned}$$

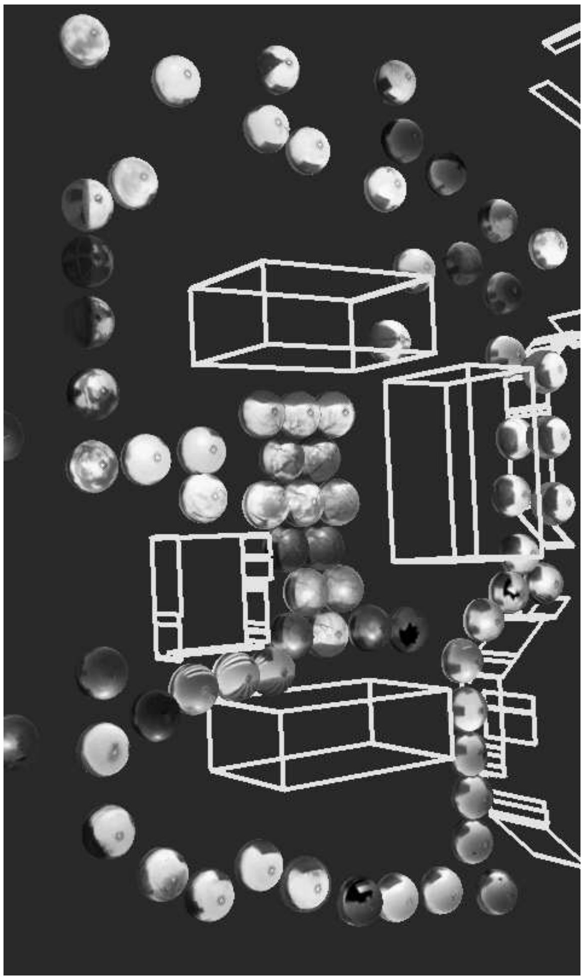
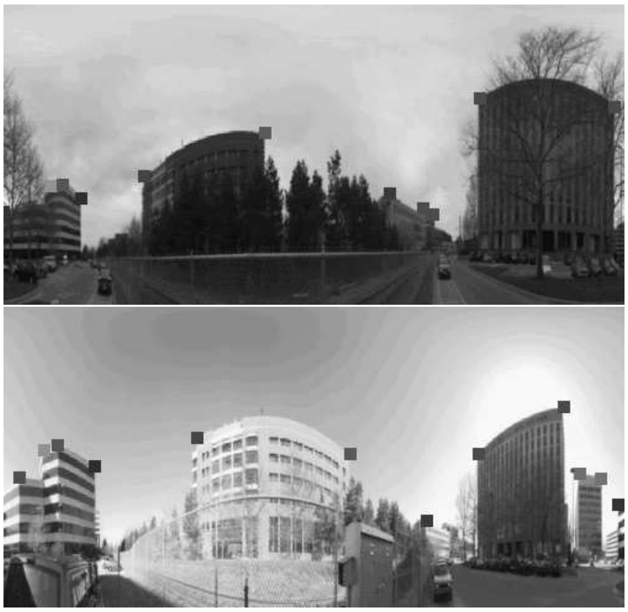
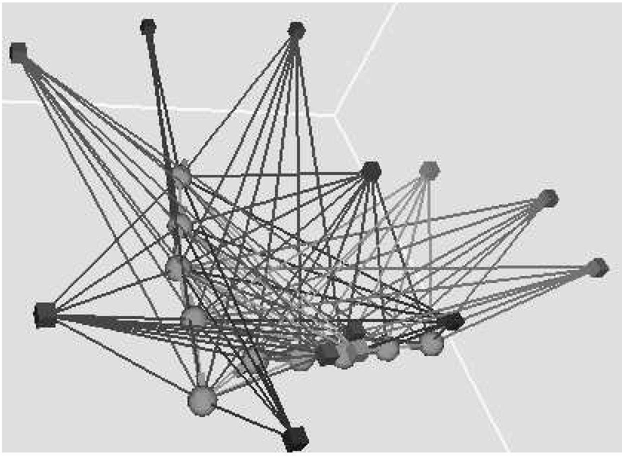


Figure 2: This figure shows reconstructed vertical facades in wireframe, co-located with input spherical mosaics.



(a)



(b)

Figure 3: This figure shows a snapshot of the global registration algorithm for ten nodes. Part (a) shows two such nodes (represented on a plane using an equal-area projection) with manually selected points (squares). The points shown have been automatically matched by the incidence counting algorithm, despite the significant dissimilarity due to perspective, occlusion, and lighting effects. Part (b) shows the locations of the ten nodes (shown as spheres) and reconstructed points (shown as boxes) in 3-D. For these nodes, automatically identified matches were sufficient to recover pose.