Unsupervised 3D Object Recognition and Reconstruction in Unordered Datasets

M. Brown and D. G. Lowe
Department of Computer Science
University of British Columbia
{mbrown|lowe}@cs.ubc.ca

Abstract

This paper presents a system for fully automatic recognition and reconstruction of 3D objects in image databases. We pose the object recognition problem as one of finding consistent matches between all images, subject to the constraint that the images were taken from a perspective camera. We assume that the objects or scenes are rigid. For each image we associate a camera matrix, which is parameterised by rotation, translation and focal length. We use invariant local features to find matches between all images, and the RANSAC algorithm to find those that are consistent with the fundamental matrix. Objects are recognised as subsets of matching images. We then solve for the structure and motion of each object, using a sparse bundle adjustment algorithm. Our results demonstrate that it is possible to recognise and reconstruct 3D objects from an unordered image database with no user input at all.

1 Introduction

Object recognition and structure and motion recovery are two long standing problems in computer vision. The structure and motion (SAM) problem has reached a degree of maturity, with several commercial offerings [1, 2], in addition to an extensive research literature [17, 8, 13]. Object recognition is also well studied but remains an extremely active research area, with recent advances in image features and probabilistic modelling inspiring previously unexplored areas such as object class recognition [7]. Invariant local features have emerged as an invaluable tool in tackling the ubiquitous image correspondence problem. By using descriptors that are invariant not just to translation, but also to rotation [16], scale [9] and affine warping [3, 12, 11], invariant features provide much more robust matching than previous correlation based methods.

Until recently, the majority of object recognition algorithms have depended upon some form of training phase [9, 18]. However, algorithms have been developed recently that operate in an unsupervised manner on an image dataset. Such algorithms look for structure in the data, using, for example, a probabilistic 'constellation model' [7] or geometric constraints arising from the image formation process [5, 15, 14]. Our work is in the spirit of the latter. We operate in an unsupervised setting on an unordered image dataset, and pose the object recognition problem as one of finding matches that are consistent views of some 3D scene.

The remainder of this paper is structured as follows. In section 2 we describe our invariant feature extraction and matching scheme. Section 3 describes the geometric constraints used to find correct image matches. Section 4 describes the sparse bundle adjustment algorithm used to solve jointly for the cameras and structure. Section 5 demonstrates results of object recognition and reconstruction on a test dataset, and section 6 presents conclusions and ideas for future work.

2 Feature Matching

The features we use are SIFT (Scale Invariant Feature Transform) features [10]. These locate interest points at maxima/minima of a difference of Gaussian function in scale-space. Each interest point has an associated orientation, which is the peak of a histogram of local orientations. This gives a similarity invariant frame in which a descriptor vector is sampled. Though a simple pixel resampling would be similarity invariant, the descriptor vector actually consists of spatially accumulated gradient measurements. This spatial accumulation is important for shift invariance, since the interest point locations are typically accurate in the 1-3 pixel range [6]. Illumination invariance is achieved by using gradients (which eliminates bias) and normalising the descriptor vector (which eliminates gain).

Once features have been extracted from all n images (linear time), they must be matched. Since multiple images may view the same point in the world, each feature is matched to k nearest neighbours (typically $k = 4$). This can be done in $O(n \log n)$ time by using a k-d tree to find approximate nearest neighbours [4].

2.1 Feature Space Outlier Rejection

We perform feature space outlier rejection to remove incorrect matches. It has been found that comparing the distance of a potential match to the distance of the best incorrect match is an effective strategy for outlier rejection [10, 6]. Suppose that the number of images that overlap a given point in the world is $n_{overlap}$. In an ordered list of nearest-neighbour matches, we assume that the first $n_{overlap} - 1$ elements are potentially correct, but that the $n_{overlap}$ element is an incorrect match. We denote the match distance of the $n_{overlap}$ element as $e_{outlier}$, as it is the best matching outlier. In order to verify a match, we compare the match distance of a potential correct match e_{match} to the outlier distance, accepting the match if

$$e_{match} < 0.8 \times e_{outlier}$$

Typically we use a value $n_{overlap} = 5$.

3 Image Matching

During this stage, the objective is to find all matching images, that is, those that view a common subset of 3D points. Connected sets of image matches will later become 3D models.

From the feature matching step, we have identified images with a large number of matches between them. Since each image could potentially match every other one, this problem appears at first to be quadratic in the number of images. However, we have found it necessary to match each image only to a small number of neighbouring images in order to get good solutions for the camera positions. We consider a constant number m images, that have the greatest number of (unconstrained) feature matches to the current image, as potential image matches (we use $m = 6$).

We parameterize each camera using 7 parameters. These are a rotation vector $\mathbf{\Theta}_i = \begin{bmatrix}\theta_{i1} & \theta_{i2} & \theta_{i3}\end{bmatrix}$, translation $\mathbf{t}_i = \begin{bmatrix}t_{i1} & t_{i2} & t_{i3}\end{bmatrix}$ and focal length f_i. The calibration matrix is then

$$\mathbf{K}_i = \begin{bmatrix} f_i & 0 & 0 \\ 0 & f_i & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

and the rotation matrix (using exponential representation)

$$\mathbf{R}_i = e^{[\boldsymbol{\theta}_i]_\times}, \quad [\boldsymbol{\theta}_i]_\times = \begin{bmatrix} 0 & -\theta_{i3} & \theta_{i2} \\ \theta_{i3} & 0 & -\theta_{i1} \\ -\theta_{i2} & \theta_{i1} & 0 \end{bmatrix}$$

Each pairwise image match adds four constraints on the camera parameters whilst adding three unknown structure parameters $\mathbf{X} = \begin{bmatrix}X_1 & X_2 & X_3\end{bmatrix}$

$$\begin{aligned} \tilde{\mathbf{u}}_i &= \mathbf{K}_i \mathbf{X}_{ci} \\ \tilde{\mathbf{u}}_j &= \mathbf{K}_j \mathbf{X}_{cj} \\ \mathbf{X}_{ci} &= \mathbf{R}_i \mathbf{X} + \mathbf{t}_i \\ \mathbf{X}_{cj} &= \mathbf{R}_j \mathbf{X} + \mathbf{t}_j \end{aligned}$$

where $\tilde{\mathbf{u}}_i, \tilde{\mathbf{u}}_j$ are the homogeneous image positions in camera i and j respectively.

The single remaining constraint (4 equations minus 3 unknowns = 1 constraint) expresses the fact that the two camera rays $\tilde{\mathbf{p}}_i$, $\tilde{\mathbf{p}}_j$ and the translation vector between camera centres $\mathbf{t}_{ij}$ are coplanar, and hence their scalar triple product is equal to zero

$$\tilde{\mathbf{p}}_i^T [\mathbf{t}_{ij}]_\times \tilde{\mathbf{p}}_j = 0 \tag{1}$$

Writing $\tilde{\mathbf{p}}_i, \tilde{\mathbf{p}}_j$ and $\mathbf{t}_{ij}$ in terms of camera parameters

$$\begin{aligned} \tilde{\mathbf{p}}_i &= \mathbf{R}_i^T \mathbf{K}_i^{-1} \tilde{\mathbf{u}}_i \\ \tilde{\mathbf{p}}_j &= \mathbf{R}_j^T \mathbf{K}_j^{-1} \tilde{\mathbf{u}}_j \\ \mathbf{t}_{ij} &= \mathbf{R}_j^T \mathbf{t}_j - \mathbf{R}_i^T \mathbf{t}_i \end{aligned}$$

and substituting in equation 1 gives

$$\tilde{\mathbf{u}}_i^T \mathbf{F}_{ij} \tilde{\mathbf{u}}_j = 0 \tag{2}$$

where

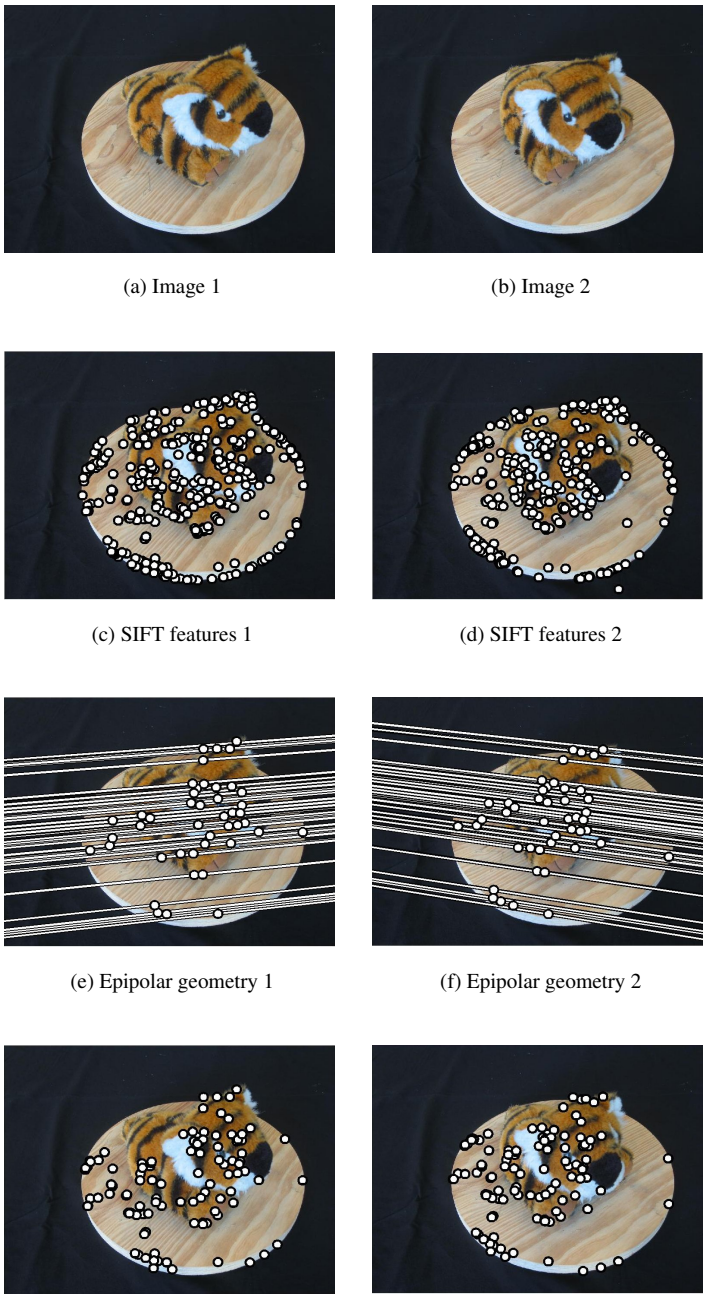
$$\mathbf{F}_{ij} = \mathbf{K}_i^{-T} \mathbf{R}_i [\mathbf{R}_j^T \mathbf{t}_j - \mathbf{R}_i^T \mathbf{t}_i]_\times \mathbf{R}_j^T \mathbf{K}_j^{-1}$$

This is the well known epipolar constraint. Image matching entails robust estimation of the fundamental matrix $\mathbf{F}_{ij}$. Since equation 2 is non-linear in the camera parameters, it is commonplace to relax the non-linear constraints and estimate a general 3×3 matrix $\mathbf{F}_{ij}$. This enables a closed form solution via SVD.

We use RANSAC to robustly estimate $\mathbf{F}$ and hence find a set of inliers that have consistent epipolar geometry. An image match is declared if the number of RANSAC inliers $n_{inliers} > n_{match}$, where the minimum number of matches n_{match} is a constant (typically around 20). Future work will add a more principled probabilistic model for image match verification. 3D objects/scenes are identified as connected components of image matches.

4 Bundle Adjustment

Given a set of geometrically consistent matches, we use bundle adjustment to solve for the camera and structure parameters jointly. In contrast to other approaches [8, 13] that begin with a projective reconstruction and later refine to a



(a) Image 1 (b) Image 2

(c) SIFT features 1 (d) SIFT features 2

(e) Epipolar geometry 1 (f) Epipolar geometry 2

(g) RANSAC inliers 1 (h) RANSAC inliers 2

Figure 1. Finding sets of consistent matches using SIFT and RANSAC. SIFT features are extracted from all input images, and each feature is matched to k = 4 nearest neighbours. Outliers are first rejected by thresholding against the distance of an incorrect match (section (2.1)), before RANSAC is used to find a final set of inliers that are consistent with the fundamental matrix. For this pair of 1024 $\times$ 768 input images, there were 365 SIFT features in image 1 and 379 in image 2. Of the initial feature matches, 133 matches remained after feature space outlier rejection, and there were 103 matches in the final solution after using RANSAC.

metric reconstruction, we solve directly for the metric structure and camera parameters. The cameras are added one by one, starting with the best matching pair. We have found that initialising each new camera with the rotation, translation and focal length of the best matching image works well, even if the images have different rotation and scale (see example in figure 2). To cope with Necker reversal, we first run bundle adjustment on the initial image pair, noting the final value of the error function (section 4.1). We then swap the camera positions, and flip the 3D point depths, before repeating bundle adjustment. This normally converges to a different local minimum. We retain the solution that minimises the error function.

4.1 Sparse Bundle Adjustment

Each connected component of feature matches defines a 3D point $\mathbf{X}_j$, and our error function is the sum squared error between the projected 3D point and the measured feature position

$$e = \sum_{i \epsilon \mathcal{I}} \sum_{j \epsilon \mathcal{X}(i)} f(\mathbf{r}_{ij})^2 \tag{3}$$

where $\mathcal{I}$ is the set of all images, $\mathcal{X}(i)$ is the set of 3D points projecting to image i, and $\mathbf{r}_{ij}$ is the residual error in image i for 3D point j. The residual $\mathbf{r}_{ij}$ is the difference between the measured feature position and projected 3D point

$$\mathbf{r}_{ij} = \mathbf{m}_{ij} - \mathbf{u}_{ij}$$

where $\mathbf{m}_{ij}$ is the measured feature position, and $\mathbf{u}_{ij}$ is the projection of point $\mathbf{X}_j$ in image i

$$\tilde{\mathbf{u}}_{ij} = \mathbf{K}_i(\mathbf{R}_i\mathbf{X}_j + \mathbf{t}_i)$$

We use a robust error function $f(\mathbf{x}) = \sqrt{g(\mathbf{x})}$ where $g(\mathbf{x})$ is the Huber robust error function

$$g(\mathbf{x}) = \begin{cases} |\mathbf{x}|^2, & \text{if } |\mathbf{x}| < \sigma \\ 2\sigma|\mathbf{x}| - \sigma^2, & \text{if } |\mathbf{x}| \geq \sigma \end{cases}$$

The outlier distance σ is set at 3 standard deviations of the current (un-normalised) residual error. This error function combines the fast convergence properties of an L_2 norm optimisation scheme for inliers (distance less than σ), with the robustness of an L_1 norm scheme for outliers (distance greater than σ).

We use the Levenberg-Marquardt algorithm to solve this non-linear least squares problem. Each iteration step is of the form

$$\mathbf{\Phi} = (\mathbf{J}^T\mathbf{J} + \sigma^2\mathbf{C}_p^{-1})^{-1}\mathbf{J}^T\mathbf{r} \tag{4}$$

where $\mathbf{\Phi} = [\mathbf{\Theta}, \mathbf{X}]$ is the vector of camera ($\mathbf{\Theta}$) and structure ($\mathbf{X}$) parameters, $\mathbf{r}$ is the vector of residuals and $\mathbf{J} = \partial\mathbf{r}/\partial\mathbf{\Phi}$. The jacobean $\mathbf{J}$ is an $M \times N$ matrix, where M is the number of measurements (twice the number of features), and $N = n_\Theta + n_X$ is the number of camera (n_Θ) and structure (n_X) parameters (7 for each camera plus 3 for each 3D point). The prior covariance matrix $\mathbf{C}_p$ is set such that the standard deviation of angles is $\sigma_\theta = \pi/16$, translations $\sigma_t = 0.01$, focal lengths $\sigma_f = \bar{f}/100$ and 3D points $\sigma_X = 0.1$. Although one could in principle solve equation 4 directly, to do so would ignore the sparse structure of the problem, and be very inefficient.

Firstly, the matrix $\mathbf{J}$ is mostly zeros (since the derivatives of residuals for image i are zero except with respect to the parameters of image i), so the elements of $\mathbf{J}^T\mathbf{J}$ should be computed directly, instead of computing $\mathbf{J}$ first. Examining the structure of $\mathbf{J}^T\mathbf{J}$

$$\mathbf{J}^T\mathbf{J} = \left[\begin{array}{c|c} \frac{\partial\mathbf{r}}{\partial\mathbf{\Theta}}^T\frac{\partial\mathbf{r}}{\partial\mathbf{\Theta}} & \frac{\partial\mathbf{r}}{\partial\mathbf{\Theta}}^T\frac{\partial\mathbf{r}}{\partial\mathbf{X}} \\ \hline \frac{\partial\mathbf{r}}{\partial\mathbf{X}}^T\frac{\partial\mathbf{r}}{\partial\mathbf{\Theta}} & \frac{\partial\mathbf{r}}{\partial\mathbf{X}}^T\frac{\partial\mathbf{r}}{\partial\mathbf{X}} \end{array}\right] = \left[\begin{array}{c|c} \mathbf{C}_\mathbf{\Theta}^{-1} & \mathbf{C}_{\mathbf{\Theta X}}^{-1} \\ \hline \mathbf{C}_{\mathbf{\Theta X}}^{-T} & \mathbf{C}_\mathbf{X}^{-1} \end{array}\right]$$

where the camera parameter inverse covariance matrix

$$\mathbf{C}_\mathbf{\Theta}^{-1} = \begin{bmatrix} \sum_j \frac{\partial\mathbf{r}_{1j}}{\partial\mathbf{\Theta}_1}^T\frac{\partial\mathbf{r}_{1j}}{\partial\mathbf{\Theta}_1} & 0 & \cdots \\ 0 & \sum_j \frac{\partial\mathbf{r}_{2j}}{\partial\mathbf{\Theta}_2}^T\frac{\partial\mathbf{r}_{2j}}{\partial\mathbf{\Theta}_2} & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix}$$

is block diagonal, consisting of 7×7 blocks, and the structure parameter inverse covariance matrix

$$\mathbf{C}_\mathbf{X}^{-1} = \begin{bmatrix} \sum_i \frac{\partial\mathbf{r}_{i1}}{\partial\mathbf{X}_1}^T\frac{\partial\mathbf{r}_{i1}}{\partial\mathbf{X}_1} & 0 & \cdots \\ 0 & \sum_i \frac{\partial\mathbf{r}_{i2}}{\partial\mathbf{X}_2}^T\frac{\partial\mathbf{r}_{i2}}{\partial\mathbf{X}_2} & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix}$$

is also block diagonal, consisting of 3×3 blocks. The camera/structure cross covariance is a full matrix

$$\mathbf{C}_{\mathbf{\Theta X}}^{-1} = \begin{bmatrix} \frac{\partial\mathbf{r}_{11}}{\partial\mathbf{\Theta}_1}^T\frac{\partial\mathbf{r}_{11}}{\partial\mathbf{X}_1} & \frac{\partial\mathbf{r}_{12}}{\partial\mathbf{\Theta}_1}^T\frac{\partial\mathbf{r}_{12}}{\partial\mathbf{X}_2} & \cdots \\ \frac{\partial\mathbf{r}_{21}}{\partial\mathbf{\Theta}_2}^T\frac{\partial\mathbf{r}_{21}}{\partial\mathbf{X}_1} & \frac{\partial\mathbf{r}_{22}}{\partial\mathbf{\Theta}_2}^T\frac{\partial\mathbf{r}_{22}}{\partial\mathbf{X}_2} & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix}$$

but consists of a single multiplication for each element (the covariance between image i and point j depends only on the residual of point j in image i). Computing $\mathbf{J}^T\mathbf{J}$ by explicit multiplication of $\mathbf{J}$ would take $O(MN^2)$ operations. However, $\mathbf{J}^T\mathbf{J}$ can in fact be computed in $O(n_\Theta n_X)$ operations (the cost of computing $\mathbf{C}_{\mathbf{\Theta X}}^{-1}$).

Secondly, the matrix inversion involving $\mathbf{J}^T\mathbf{J}$ need not be computed explicitly ($O(N^3)$) due to the sparse structure

of $\mathbf{J}^T\mathbf{J}$. This sparseness reflects the loose coupling inbetween cameras, and inbetween 3D points, in the error function of equation 3. The cameras are independent given the 3D structure parameters, and the 3D points are independent given the cameras. Equation 4 may be rewritten

$$\begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{B}^T & \mathbf{C} \end{bmatrix} \begin{bmatrix} \mathbf{\Theta} \\ \mathbf{X} \end{bmatrix} = \begin{bmatrix} e_{\mathbf{\Theta}} \\ e_{\mathbf{X}} \end{bmatrix} \tag{5}$$

where

$$\begin{aligned} \mathbf{A} &= \mathbf{C}_{\mathbf{\Theta}}^{-1} + \sigma^2 \mathbf{C}_{p_{\mathbf{\Theta}}}^{-1} \\ \mathbf{C} &= \mathbf{C}_{\mathbf{X}}^{-1} + \sigma^2 \mathbf{C}_{p_{\mathbf{X}}}^{-1} \\ \mathbf{B} &= \mathbf{C}_{\mathbf{\Theta}\mathbf{X}}^{-1} \\ \mathbf{e}_{\mathbf{\Theta}} &= \frac{\partial \mathbf{r}}{\partial \mathbf{\Theta}}^T \mathbf{r} \\ \mathbf{e}_{\mathbf{X}} &= \frac{\partial \mathbf{r}}{\partial \mathbf{X}}^T \mathbf{r} \end{aligned}$$

and

$$\mathbf{C}_p^{-1} = \begin{bmatrix} \mathbf{C}_{p_{\mathbf{\Theta}}}^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{C}_{p_{\mathbf{X}}}^{-1} \end{bmatrix}$$

Multiplying both sides of equation 5 by $\begin{bmatrix} \mathbf{I} & -\mathbf{B}\mathbf{C}^{-1} \\ \mathbf{0} & \mathbf{I} \end{bmatrix}$ gives

$$\begin{bmatrix} \mathbf{A} - \mathbf{B}\mathbf{C}^{-1}\mathbf{B}^T & \mathbf{0} \\ \mathbf{B}^T & \mathbf{C} \end{bmatrix} \begin{bmatrix} \mathbf{\Theta} \\ \mathbf{X} \end{bmatrix} = \begin{bmatrix} e_{\mathbf{\Theta}} - \mathbf{B}\mathbf{C}^{-1}e_{\mathbf{X}} \\ e_{\mathbf{X}} \end{bmatrix}$$

which eliminates $\mathbf{X}$ from the solution for $\mathbf{\Theta}$.

This gives an $n_\Theta \times n_\Theta$ linear system to solve for the camera parameters $\mathbf{\Theta}$. The resulting value of $\mathbf{\Theta}$ can be substituted into the linear system for $\mathbf{X}$, which reduces to independent 3×3 linear systems for each 3D point. The most expensive stage in this process is (potentially) in computation of the left-hand side elimination product $\mathbf{B}\mathbf{C}^{-1}\mathbf{B}^T$. The ijth element is given by

$$(\mathbf{B}\mathbf{C}^{-1}\mathbf{B}^T)_{ij} = \sum_k \mathbf{B}_{ik}\mathbf{C}_{kk}\mathbf{B}_{kj}$$

where $\mathbf{B}_{ij} = \frac{\partial \mathbf{r}_{ij}}{\partial \mathbf{\Theta}_i}^T \frac{\partial \mathbf{r}_{ij}}{\partial \mathbf{X}_j}$ is a 7×3 matrix (number of parameters per camera $\times$ number of parameters per 3D point) and $\mathbf{C}_{kk} = \sum_i \frac{\partial \mathbf{r}_{ik}}{\partial \mathbf{X}_k}^T \frac{\partial \mathbf{r}_{ik}}{\partial \mathbf{X}_k}$ is a 3×3 matrix. This summation may involve in the worst case M terms (if every 3D point is imaged in every camera). Hence the worst case complexity of sparse bundle adjustment is $O(Mn_\Theta^2)$. Note that this is still much cheaper than it would be were $\mathbf{C}$ not block diagonal. If $\mathbf{C}$ were a general $n_X \times n_X$ matrix the cost of this elimination step would be $O(n_X^3)$. However, the cost of bundle adjustment is usually much less than $O(Mn_\Theta^2)$. This is because the terms $\mathbf{B}_{ij}$ are zero unless point j is viewed in camera i. This means that each summation above involves only a constant number of 3D points for each camera. In this case, the complexity of sparse bundle adjustment is $O(mn_\Theta^2)$, where m is the number of residuals in each image. The best case complexity (given small m) is $O(n_\Theta^3)$, which is the cost of solving the linear system for the camera parameters.

Hence the total computational cost for one step of sparse bundle adjustment is now $O(mn_\Theta^2)$, reduced from $O(MN^2)$ for naive solution of the normal equations. Note that $n_\Theta \ll N$ since the number of camera parameters n_Θ is very much less than the number of structure parameters n_X ($N = n_\Theta + n_X$). This is a very significant reduction in practice. For example, with 10 cameras ($n_\Theta = 70$), and 1000 3D points ($n_X = 3000$), sparse bundle adjustment would be about $(N/n_\Theta)^2 = (3070/70)^2 \approx 2000$ times faster than naive bundle adjustment. Furthermore, if a constant number of 3D points are imaged by each camera, the cost would be further reduced.

4.2 Analytical Computation of Derivatives

Derivatives are computed analytically via the chain rule, for example

$$\frac{\partial \mathbf{u}_{ij}}{\partial \theta_{i1}} = \frac{\partial \mathbf{u}_{ij}}{\partial \tilde{\mathbf{u}}_{ij}} \frac{\partial \tilde{\mathbf{u}}_{ij}}{\partial \theta_{i1}}$$

where

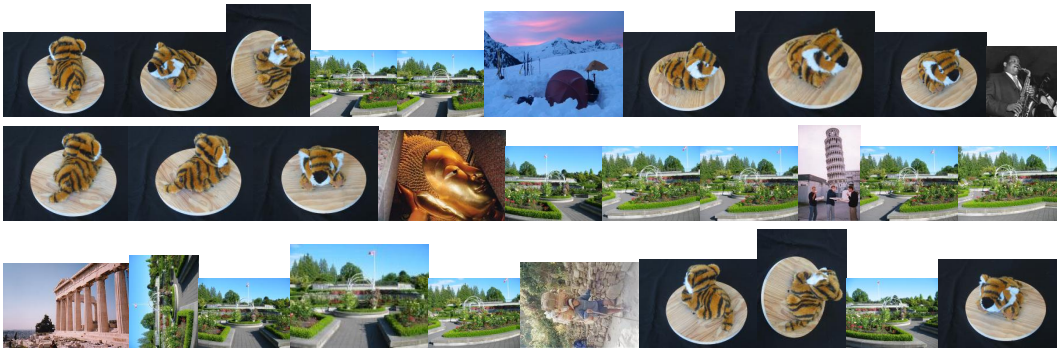
$$\frac{\partial \mathbf{u}_{ij}}{\partial \tilde{\mathbf{u}}_{ij}} = \frac{\partial \begin{bmatrix} x/z & y/z \end{bmatrix}}{\partial \begin{bmatrix} x & y & z \end{bmatrix}} = \begin{bmatrix} 1/z & 0 & -x/z^2 \\ 0 & 1/z & -y/z^2 \end{bmatrix}$$

and

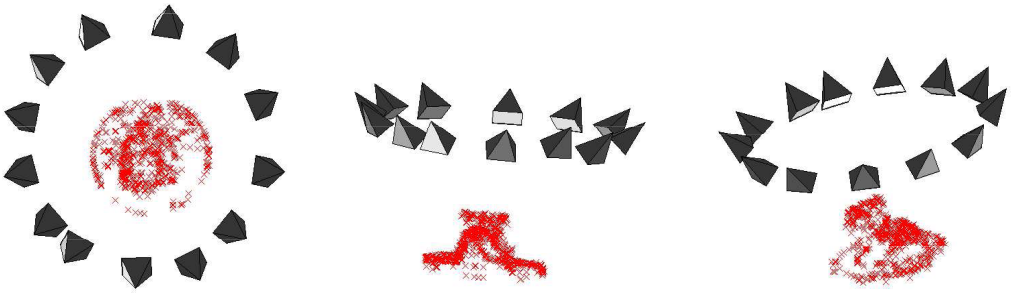
$$\begin{aligned} \frac{\partial \tilde{\mathbf{u}}_{ij}}{\partial \theta_{i1}} &= \mathbf{K}_i \frac{\partial \mathbf{R}_i}{\partial \theta_{i1}} \mathbf{X}_j \\ \frac{\partial \mathbf{R}_i}{\partial \theta_{i1}} &= \frac{\partial}{\partial \theta_{i1}} e^{[\boldsymbol{\theta_i}]_\times} = e^{[\boldsymbol{\theta_i}]_\times} \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix} \end{aligned}$$

5 Results

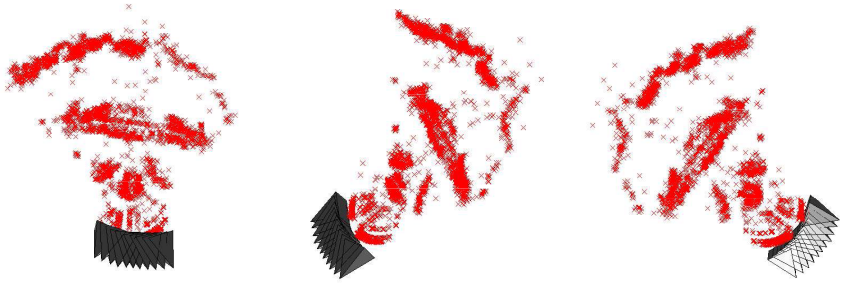
Figure 2 shows typical operation of the object recognition algorithm. A set of images containing 2 objects and 6 distractor images was input. The algorithm detected 2 connected components of image matches and 6 unmatched images, and output the 2 reconstructed 3D models. The complete algorithm ran in 556 seconds on a 2.8GHz PC. About half of the computation time was spent in bundle adjustment. Another example of fully automatic structure and motion estimation is given in figure 3.



(a) Input images

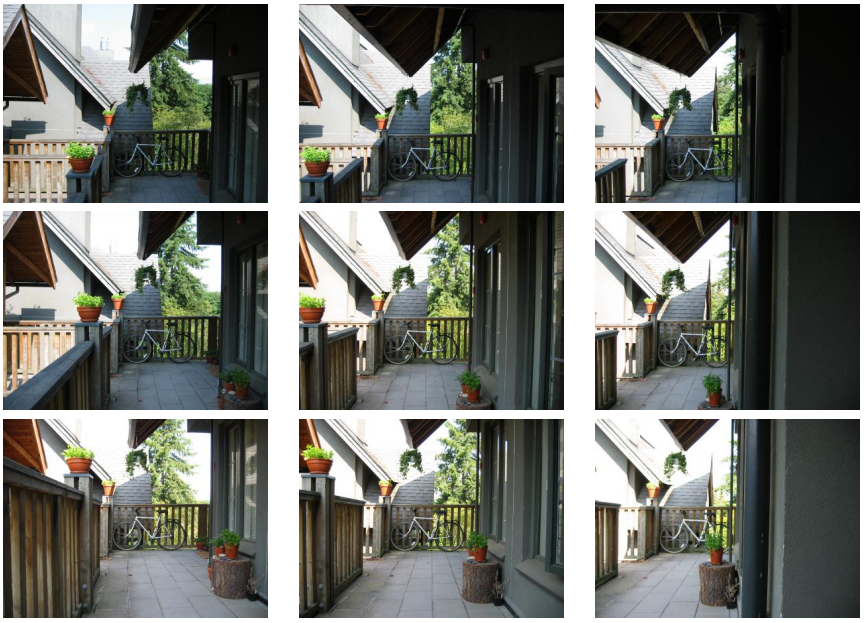


(b) Output 3D model 1 - Tiger

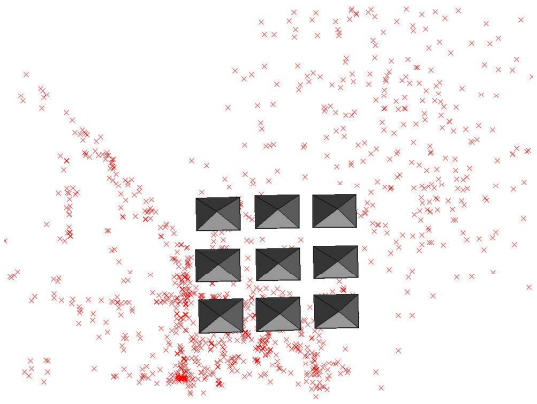


(c) Output 3D model 2 - Rosegarden

Figure 2. Fully automatic object recognition and 3D reconstruction. Note that despite the incorrect ordering, rotation and scale changes, and distractor images in the input, the system is able to successfully recognise the two consistent objects and perform 3D reconstruction. The Tiger sequence consisted of 13 images and yielded a 3D model with 675 points. The Rosegarden sequence consisted of 11 images and the 3D model contained 1351 points. The whole process of feature matching, image matching and bundle adjustment took a total of 556 seconds, of which 230 seconds were spent during bundle adjustment. The tests were run using a MATLAB implementation on a 2.8GHz Pentium processor.



(a) Input images



(b) Output 3D model

Figure 3. Fully automatic Structure and Motion (SAM) estimation for a 3 $\times$ 3 array of cameras. In this example the input images were 800 $\times$ 600, and a 3D model of 1684 points was computed. The total computation time for feature extraction and matching, image matching and bundle adjustment was 293 seconds.

Algorithm: 3D Object Recognition/Reconstruction

Input: n unordered images

I. Extract SIFT features from all n images

II. Find k nearest-neighbours for each feature using a k-d tree

III. For each image:
   (i) Select m candidate matching images (with the maximum number of feature matches to this image)
   (ii) Find geometrically consistent feature matches using RANSAC to solve for the fundamental matrix between pairs of images
   (iii) (Future work) Verify image matches using probabilistic model

IV. Find connected components of image matches

V. For each connected component:
   (i) Perform sparse bundle adjustment to solve for the rotation $\theta_1, \theta_2, \theta_3$, translation t_1, t_2, t_3 and focal length f of all cameras, and pointwise 3D geometry
   (ii) (Future Work) Compute dense depth estimates, triangulate, texture map etc.

Output: 3D model(s)

6 Conclusions

We have presented a fully automatic 3D object recognition and reconstruction system. Our system starts by extracting SIFT features from a collection of images, and recognises 3D scenes as geometrically consistent sets of feature matches. We perform bundle adjustment for metric structure directly, without the initial projective reconstruction common to other approaches. We have found that initialising each new camera with the same parameters as the best matching camera gives no problems with convergence.

Future work will develop a principled model for verifying correct image matches, and enhance the output 3D models by using dense depth estimation, triangulation and texture mapping.

References

[1] 2D3. http://www.2d3.com.

[2] REALVIZ. http://www.realviz.com.

[3] A. Baumberg. Reliable Feature Matching Across Widely Separated Views. In *Proceedings of the Interational Conference on Computer Vision and Pattern Recognition*, pages 774–781, 2000.

[4] J. Beis and D. Lowe. Shape Indexing using Approximate Nearest-Neighbor Search in High-Dimensional Spaces. In *Proceedings of the Interational Conference on Computer Vision and Pattern Recognition*, pages 1000–1006, 1997.

[5] M. Brown and D. Lowe. Recognising panoramas. In *Proceedings of the 9th International Conference on Computer Vision*, volume 2, pages 1218–1225, Nice, October 2003.

[6] M. Brown, R. Szeliski, and S. Winder. Multi-Image Matching using Multi-Scale Oriented Patches. In *Proceedings of the Interational Conference on Computer Vision and Pattern Recognition*, San Diego, June 2005.

[7] R. Fergus, P. Perona, and A. Zisserman. Object Class Recognition by Unsupervised Scale-Invariant Learning. In *Proceedings of the Interational Conference on Computer Vision and Pattern Recognition*, 2003.

[8] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521623049, 2000.

[9] D. Lowe. Object Recognition from Local Scale-Invariant Features. In *Proceedings of the International Conference on Computer Vision*, pages 1150–1157, Corfu, Greece, September 1999.

[10] D. G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[11] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust Wide Baseline Stereo from Maximally Stable Extremal Regions. In *British Machine Vision Conference*, 2002.

[12] K. Mikolajczyk and C. Schmid. An Affine Invariant Interest Point Detector. In *Proceedings of the European Conference on Computer Vision*, 2002.

[13] M. Pollefeys. 3D Modelling from Images. In *Proceedings of the European Conference on Computer Vision*, Dublin, June 2000.

[14] F. Rothganger, S. Lazebnik, C. Schmid, and J. Ponce. 3D Object Modelling and Recognition using Affine-Invariant Patches and Multi-View Spatial Constraints. In *Proceedings of the Interational Conference on Computer Vision and Pattern Recognition*, pages 272–277, Madison, WI, June 2003.

[15] F. Schaffalitzky and A. Zisserman. Multi-view Matching for Unordered Image Sets, or "How Do I Organise My Holiday Snaps?". In *Proceedings of the European Conference on Computer Vision*, pages 414–431, 2002.

[16] C. Schmid and R. Mohr. Local Grayvalue Invariants for Image Retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(5):530–535, May 1997.

[17] R. Szeliski and S. B. Kang. Recovering 3D Shape and Motion from Image Streams using Non-Linear Least Squares. Technical Report CRL 93/3, Digital Equipment Corporation, Cambridge Research Laboratory, March 1993.

[18] P. Viola and M. Jones. Rapid Object Detection using a Boosted Cascade of Simple Features. In *Proceedings of the Interational Conference on Computer Vision and Pattern Recognition*, December 2001.