Model Evolution: An Incremental Approach to Non-Rigid Structure from Motion

Shengqi Zhu Li Zhang Brandon M. Smith
University of Wisconsin – Madison
http://www.cs.wisc.edu/~lizhang/projects/mevolve/

Abstract

In this paper, we present a new framework for non-rigid structure from motion (NRSFM) that simultaneously addresses three significant challenges: severe occlusion, perspective camera projection, and large non-linear deformation. We introduce a concept called a model graph, which greatly reduces the computational cost of discovering groups of input images that depict consistent 3D shapes. A 3D model is constructed for each input image by traversing the model graph along multiple evolutionary paths. A compressive shape representation is constructed, which (1) consolidates the multiple 3D models for each image reconstructed during model evolution and (2) reduces the number of models needed to represent the input image set. Assuming feature correspondences are known, we demonstrate our algorithm on both real and synthetic data sets that exemplify all three aforementioned challenges.

1. Introduction

Structure from Motion (SFM) is a fundamental problem in computer vision. For rigid objects, the problem is well understood [6] and impressive results have been demonstrated on large scale data sets, *e.g.* [1]. In the realm of nonrigid objects, there are several challenges that prevent existing techniques from being more widely used in practice. Two typical cases where one would want to use non-rigid structure from motion (NRSFM) are:

- Reconstructing 3D models for deformable and/or articulated targets, *e.g.*, human movement;
- Reconstructing 3D models for a collection of *similar-but-not-identical* objects in a category, *e.g.*, faces of different people.

In order to be used in practice for such applications, NRSFM techniques must robustly address the following challenges:

- Feature occlusion due to large variation of viewpoints;
- Perspective projection camera model;
- Large non-linear object deformation.

As discussed in Section 2.4, we know of no existing algorithms that can deal with all three of these challenges simultaneously. In this paper, we present a new framework for NRSFM that seeks to address all three challenges.

2. A Model Graph Formulation of NRSFM

M images $\{I_i\}_{i=1}^{M}$ are given, each represented as a collection of 2D feature points $I_i = \{\mathbf{x}_{i,j}\}$, where j is the index for the 2D feature points in I_i. In this work, we assume that some feature points may be occluded in each image, but the feature correspondences are known. That is, we know the mapping $\pi_i(\cdot)$ from the index of a 2D feature point in image i to the index of the corresponding 3D feature point in the model.

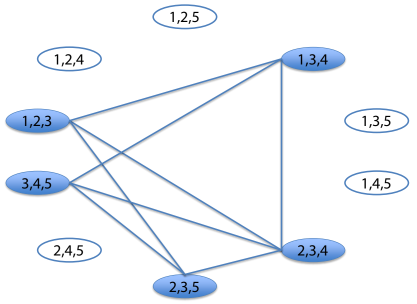


Figure 1. The concept of a Model Graph for a set of 10 input images for NRSFM. The numbers in the nodes represent image indices. Blue nodes represent clusters of 3 images that share very similar 3D shapes. An edge is defined between two nodes that differ by a single image, and 3D shapes for neighboring nodes must be very similar under this edge definition. We formulate NRSFM as an efficient Model Graph traversing problem, in which 3D models can be constructed incrementally in the presence of severe occlusion, perspective camera projection, and large non-linear deformation.

Consider a group of Q images within the input image set. This group of images may correspond to objects of very different shapes; in this case, we call it an *inconsistent* image cluster. If all images in this group happen to correspond to objects of very similar shapes, we call it a *consistent* image cluster. More precisely, we call an image cluster ε-*consistent* if rigid SFM techniques can be used to reconstruct a *full* 3D model (*i.e.*, it contains all 3D feature points of interest) from these images with a reprojection error less than ε.

Now we consider *all* ε-consistent clusters and construct a *model graph*, in which each node represents one consistent cluster and its associated 3D model. For this graph, we define an edge connecting two nodes only if the two corresponding clusters have $Q-1$ images in common; in other words, the two clusters differ by only one image. For example, for $Q = 3$, nodes representing clusters $\{I_1, I_2, I_3\}$ and $\{I_1, I_2, I_5\}$ will be connected. Figure 1 illustrates an example of a model graph for $M = 5$ and $Q = 3$. In this case, there are a total of "M choose Q," $\binom{M}{Q}$, number of clusters; we label the consistent ones in blue and connect them using our edge definition above.

A key property of our model graph is that, *if two nodes are connected by an edge, their 3D models must be similar*, because these two nodes differ by just one image (and all the rest of the $Q-1$ images are the same).

2.1. A Thought Experiment

If the computational cost is not an issue, one way to pose the NRSFM problem is simply to reconstruct all models for nodes that correspond to consistent image clusters. It is advantageous to view NRSFM this way, as it does not assume any particular deformation model (linear or nonlinear). Furthermore, the 3D model for each node can be reconstructed by existing (incremental) rigid SFM techniques, which can handle both perspective camera projection as well as feature occlusion, if the cluster size Q is not too small.

Assumption 1: To ensure that we can recover a 3D model for each input image, we assume that each image must belong to at least one ε-consistent image cluster. Intuitively, this assumes that the input images collectively observe all of the shape configurations up to the ε reprojection error.

As a counter example of this assumption, if we have a single cat face image in a large collection of human face images, this cat face cannot belong to any consistent image cluster, and therefore its 3D shape cannot be reconstructed. A similar example is an image of a crouching body within a collection of images depicting normal human walking. We can view these images as outliers in the reconstruction process.

2.2. NRSFM as Model Graph Traversal

Even if we have enough image data to cover all the shape configurations, the number of consistent clusters is exponential and the thought experiment cannot be realized in practice. Instead, we pose NRSFM as a node selection problem in the model graph: *select a subset of nodes so that their corresponding consistent clusters together cover all M input images.*

Notice that we only need a maximum of M clusters to cover the M input images. Recovering a 3D model for each node will provide at least one 3D model for each input image; an image may get more than one reconstructed 3D model if the image exists in multiple selected clusters.

The node selection problem is still difficult to solve because consistent clusters are not known *a priori* and yet we need to identify them. Given M and clusters of size Q, we need at least $\frac{M}{Q}$ consistent clusters to cover all the M input images; each cluster needs $O\left(\binom{M}{Q}\right)$ operations to be discovered, which is computationally prohibitive.

Assumption 2: The node selection process becomes efficient if we assume that the model graph is *connected*. Under this assumption, we can identify one consistent cluster as the starting node, and find other consistent clusters by traversing the model graph efficiently. At each node, we move to the next node by determining which image in the current cluster should be replaced by an unvisited image. Once we move to the next node, we can reconstruct its 3D model by slightly updating the model inherited from its predecessor.

We present the details of our graph traversal algorithm in Section 3. At the end of the traversal, we have M 3D models, one for each traversed cluster node. Each image has one or more 3D models, depending on the number of cluster nodes it belongs to. In Section 4, we present algorithms to consolidate multiple 3D models for each image and optimize a reduced set of 3D models to fit the input image set.

When is the Model Graph Connected? As the number of input images M increases, the model graph is more likely to be connected. For example, for continuously varying shapes, such as facial deformation and body articulation, if the input video is long enough, the observation of similar shape configurations in multiple images becomes more likely.

How to Reconstruct Full 3D Models? In practice, allowing feature occlusions allows the input images to have a large variety of viewpoint changes; such input data are necessary for reconstructing full 3D models. Therefore, our formulation is well suited to a large number of input images, each of which may have a nontrivial portion of important features occluded. Most previous formulations take images with limited viewpoint change (little or no occlusions) as input. Therefore, they are not suitable for reconstructing full 360 degree models.

2.3. Our Contribution

This paper makes the following technical contributions.

- A model graph formulation for NRSFM that addresses three open challenges: severe occlusion, perspective camera projection, and large non-linear deformation;
- A model evolution algorithm that traverses the model graph and reconstructs 3D models for each non-outlier input image;
- A compressive shape representation, which (1) consolidates the multiple 3D models for each image reconstructed during model evolution and (2) reduces the number of models needed to represent the input image set, and its associated bundle adjustment algorithm.

In addition to these technical contributions, our approach has a practical advantage.

- All the well-understood rigid SFM techniques are conveniently integrated into our framework as separate modules. For practitioners, there is no need to implement our approach from scratch.

2.4. Comparison with Previous Work

After the pioneering work by Bregler *et al.* [5], a number of excellent solutions [3, 12, 10, 9, 4, 14, 15, 11, 13, 2] have been proposed to solve NRSFM in the past decade. All of these method assume a linear deformation model, with the exception that the kinematic chain of human motion is modeled in [15]. Our formulation is more general, as it does not assume any form of motion *a priori*. Most of these methods also assume that there is no feature occlusion in the input image sets. For those methods that do consider occlusion [9, 11], the occlusion is modeled as an outlier process; it can handle occlusion situations like a forearm occluding features on a torso, but more severe occlusion such as a torso turning 180° away from the camera cannot be handled properly. Furthermore, all of these previous works assume affine camera projection. Images with strong perspective effects cannot be handled.

The only work, to the best of our knowledge, that can handle general large nonlinear deformation is [8], in which the authors group a set of images into rigid clusters and recover a rigid shape for each cluster as an initialization for shape manifold learning. Their clustering method favors a large number of small clusters (of size 3), and the clusters are isolated, *i.e.*, each image belongs to only one cluster. The isolation between clusters forces the authors to use a small cluster size; independently searching for each consistent cluster is computationally expensive otherwise. Furthermore, a small cluster size forces the adoption of an affine camera model and requires that all features be visible in each cluster; reconstructing a complete 3D model using perspective SFM with missing features from only 3 views is otherwise a very challenging problem.

Our formulation is opposite to [8]. We use large, overlapping image clusters. Large clusters make a perspective camera model practically feasible, and overlapping clusters avoid an independent search for each consistent cluster. Both cluster search and 3D reconstruction become efficient incremental computations. Based on this *soft* clustering, our formulation is also non-parametric in that it is not limited to any specific form of deformation.

3. Model Evolution

In this section, we present our model evolution algorithm, which traverses the model graph to discover ε-consistent nodes for 3D reconstruction. The algorithm starts from a known consistent node and traverses the graph along multiple paths, until all of the M input images are covered. During the process, the model in the starting node gradually evolves to fit the images in subsequent clusters.

3.1. Reliable and ε-Consistent Models

ε-consistency is a necessary but not a sufficient condition to reconstruct correct 3D models. Considering an extreme case, where all images in a cluster share the same viewpoint, an ε-consistent model can still be generated but does not resemble the underlying groundtruth model of this cluster. In our framework, we require an ε-consistent model to be *reliable*, defined as follows. Each 3D point is visible to at least M cameras, and all pairwise angles between these cameras are within a predefined range, $[\theta_0, \theta_1]$. In this paper, we use $M = 3$ and $[\theta_0, \theta_1] = [15^\circ, 165^\circ]$.

3.2. Seed Model Creation

Given a consistent cluster as the starting node, we first use the incremental rigid SFM [6] technique to reconstruct its 3D model. We perform projective reconstruction, followed by auto-calibration assuming zero-skew, unit aspect ratio, and principal point at origin. In the end, we perform a metric bundle adjustment to recover the 3D model for the starting node.

Assuming a single consistent cluster as a starting point is not a restrictive assumption in practice. For example, working with faces, we can easily find multiple images of one person's face; working with human body articulation, we can always find multiple images of a typical pose.

3.3. A Tree of Models

Once a seed model has been constructed, we form a candidate list $\mathcal{L}$ of input images that have not yet been used in any visited nodes. We then iterate over the following steps until $\mathcal{L}$ is empty or until no additional ε-consistent models can be generated. At each step, we want to search for a pair consisting of a visited node v and an unused image and move to a new consistent node by replacing one of v's images with the unused image. We can search exhaustively at each step to find the next move; in practice, we find the following two heuristic steps work well in our experiments.

1. For each image i in $\mathcal{L}$, we estimate its pose with respect to every model that has been constructed. We use the average reprojection error of the pose estimation as an indicator of how well the image fits with a particular model, and we use the minimum error among different models as a score for each image. RANSAC is used during pose estimation to remove feature correspondence outliers. We choose the image in $\mathcal{L}$ with the minimum score, subject to the requirement that the image has at least 12 inlier point matches and the camera is facing the model. We pair this image and its best fitting model node as the candidate pair for the next step.
2. Given the candidate pair, we want to replace one old image in the node with the new image. Intuitively, we want to remove the old image in the node that most "conflicts" with the new image. We could test every old image; instead we heuristically find this image by adding the new image to the cluster and update the model by performing a metric bundle adjustment. We pick the old image whose average reprojection error increases most compared to the error computed with respect to its previous 3D model. We remove it from the node if, after removal, the model is reliable and ε-consistent. Otherwise, we consider the old image with the second largest increase in the reprojection error and so on until we find one. We run bundle adjustment again to update the model and to make sure it is still reliable. If we cannot find such an old image, we discard the current model and put image i in $\mathcal{L}'$.
3. Remove image i from $\mathcal{L}$. If $\mathcal{L}$ is empty, reset $\mathcal{L}$ with $\mathcal{L}'$.

Note that the cost of Step 1 can be made linear in $|\mathcal{L}|$ for each iteration by saving the results of previous image-to-model pairings, so that the images in $\mathcal{L}$ need only be fit to the single newest node reconstructed in the previous iteration. A new model in Step 2 will only differ from its parent model by one image. Therefore, the 3D shape will evolve slightly as the algorithm progresses. After the evolution, we will have a tree of 3D models.

3.4. Model Reduction

After the evolution, each image is associated with one or more 3D models depending on how many visited consistent clusters it belongs to. Many of these models will be highly similar. It is desirable to reduce the number of models, both to represent the image collection and also to estimate a unique model for each image. In this subsection, we describe a coarse way to reduce the models, which will be used an starting point for the algorithm in Section 4.

Specifically, we first use a simple K-means algorithm to divide the reconstructed 3D models into K groups, where K is currently set by the user and is based on the degree to which the target object deforms and/or articulates, *i.e.*, an object that deforms significantly will require more bases than a nearly rigid object. The mean shapes of each group serves as K basis shapes for all the reconstructed 3D models.

For each image, we search for the best-fit basis shape, which has the minimum average reprojection error in pose estimation with respect to the image. We then assign this basis shape and the estimated pose to the image. Using these basis shapes and estimated poses as an initial value, we next describe a more accurate solution to 3D model reduction.

4. Compressive Shape Representation

In this section, we present a compressive shape representation to model large deformation. We seek to estimate this representation from the input image collection using the result of Section 3.4 as an initial solution.

Let $\mathcal{X} = [\mathbf{X}_1; \mathbf{X}_2; \cdots; \mathbf{X}_N]$, where each $\mathbf{X}_n \in \mathrm{R}^3$ and $\mathcal{X} \in \mathrm{R}^{3N}$, be the set of 3D feature points of an object. To model the large deformation that the object may exhibit, we assume that $\mathcal{X}$ is a *sparse* linear combination of a large number of basis shapes $\{\mathcal{Z}_k\}_{k=1}^{K}$:

$$\mathcal{X} = \sum_{k=1}^{K} c_k \mathcal{Z}_k, \tag{1}$$

where $\mathcal{Z}_k = [\mathbf{Z}_{1,k}; \mathbf{Z}_{2,k}; \cdots; \mathbf{Z}_{N,k}]$, and the coefficient vector $\mathbf{c} = [c_1; c_2; \cdots; c_K]$ has very few nonzero elements.[1]

For each image i, we use $\mathbf{K}_i$, $\mathbf{R}_i$, and $\mathbf{o}_i$ to denote its camera intrinsics, rotation, and center. Then, the 3D feature point $\mathbf{X}_n$ and the 2D feature point $\mathbf{x}_{i,n}$ on the image i satisfy the following projection equation

$$\mathbf{x}_{i,n} = \mathbf{K}_i \mathbf{R}_i \left(\mathbf{X}_n - \mathbf{o}_i \right), \tag{2}$$

where $\mathbf{X}_n = \sum_{k=1}^{K} c_{i,k} \mathbf{Z}_{n,k}$, and $\mathbf{c}_i = [c_{i,1}; c_{i,2}; \cdots; c_{i,K}]$ is the coefficient vector for this image. To recover the compressive shape model, we need to solve for a dauntingly large number of unknowns, namely $\{\mathbf{K}_i, \mathbf{R}_i, \mathbf{o}_i, \mathbf{c}_i\}$ and $\{\mathcal{Z}_k\}$, by minimizing their reprojection error. This is possible because model evolution gives good initialization for this optimization.

$$\begin{aligned} &\Psi(\{\mathbf{K}_i, \mathbf{R}_i, \mathbf{o}_i, \mathbf{c}_i\}, \{\mathcal{Z}_k\}) \\ &= \sum_{i=1}^{M} \sum_{n=1}^{N} \left\| \mathbf{x}_{i,n} - \mathbf{K}_i \mathbf{R}_i \left(\left(\sum_{k=1}^{K} c_{i,k} \mathbf{Z}_{n,k} \right) - \mathbf{o}_i \right) \right\|_2^2 . \end{aligned} \tag{3}$$

We next describe ambiguities in this problem formulation to motivate our optimization procedure. These ambiguities have been identified in the study of NRSFM using weak orthographic projection [14, 11]. We briefly summarize them below for the perspective camera projection model.

[1]Such a sparse linear representation is widely used in compressive sensing literature, which is why we call it a compressive shape representation.

Shape Translation Ambiguity If each basis shape $\mathcal{Z}_k$ is translated by a vector $\mathbf{t}_k$, such translations can be compensated by translating camera centers appropriately. Formally, the following two solutions are equivalent: $\{\mathbf{Z}_{n,k} - \mathbf{t}_k, \mathbf{o}_i\} \equiv \{\mathbf{Z}_{n,k}, \mathbf{o}_i - \sum_{k=1}^{K} c_{i,k} \mathbf{t}_k\}$, which suggests we can move the center of each basis shape to $\mathbf{0}_{3\times1}$ by using $\mathbf{t}_k = \frac{1}{N} \sum_{n=1}^{N} \mathbf{Z}_{n,k}$.

Shape Scale Ambiguity If each basis shape is scaled down by a factor α_k, such scales can be compensated by scaling up the corresponding basis coefficient by the same factor. Formally, the following two solutions are equivalent: $\{c_{i,k}, \frac{1}{\alpha_k} \mathcal{Z}_k\} \equiv \{\alpha_k c_{i,k}, \mathcal{Z}_k\}$. This ambiguity is due to the bilinear relationship between c and $\mathbf{Z}$. We can force each basis shape vector $\mathcal{Z}_k$ to have unit L_2-norm by using $\alpha_k = \|\mathcal{Z}_k\|_2$, where $\|\cdot\|_2$ denotes the L_2-norm.

Coefficient Scale Ambiguity If each basis coefficient vector is scaled down by a factor β_i, such scales can be compensated by scaling down the corresponding camera center by the same factor. Formally, the following two solutions are equivalent: $\{\mathbf{c}_i, \mathbf{o}_i\} \equiv \{\frac{1}{\beta_i} \mathbf{c}_i, \frac{1}{\beta_i} \mathbf{o}_i\}$. This ambiguity is due to the perspective projection. We can force each coefficient vector $\mathbf{c}_i$ to have unit L_2-norm by using $\beta_i = \|\mathbf{c}_i\|_2$.

Subspace Rotation Ambiguity If all coefficient vectors $\mathbf{c}_i \in \mathrm{R}^K$ are rotated by the same rotation $\mathcal{R} \in \mathrm{R}^{K\times K}$, such a rotation can be compensated by rotating all shape bases by $\mathcal{R}^{-1}$. Formally, the following two solutions are equivalent:

$$\{\mathcal{R}\mathbf{c}_i, [\mathcal{Z}_1, \cdots, \mathcal{Z}_K]\} \equiv \{\mathbf{c}_i, [\mathcal{Z}_1, \cdots, \mathcal{Z}_K]\, \mathcal{R}^{-1}\}. \tag{4}$$

Based on the above ambiguity analysis, we formulate the compressive shape reconstruction as a constrained optimization problem as follows.

$$\begin{aligned} &\Phi(\{\mathbf{K}_i, \mathbf{R}_i, \mathbf{o}_i, \mathbf{c}_i\}, \{\mathcal{Z}_k\}) \\ &= \Psi(\{\mathbf{K}_i, \mathbf{R}_i, \mathbf{o}_i, \mathbf{c}_i\}, \{\mathcal{Z}_k\}) + \tau \sum_{i=1}^{M} \|\mathbf{c}_i\|_1 \\ &\text{such that} \\ &\forall k, \quad \sum_{n=1}^{N} \mathbf{Z}_{n,k} = \mathbf{0}_{3\times1} \\ &\forall k, \quad \|\mathcal{Z}_k\|_2^2 = 1 \\ &\forall i, \quad \|\mathbf{c}_i\|_2^2 = 1 \end{aligned} \tag{5}$$

where Ψ is defined in Eq. (3), $\|\cdot\|_1$ denotes the L_1-norm and $\sum_{i=1}^{M} \|\mathbf{c}_i\|_1$ is the *sparsity regularization* that we introduce to remove subspace rotation ambiguity as follows. The three groups of constraints can remove the shape translation, shape scale, and coefficient scale ambiguities, but they cannot remove the subspace rotation ambiguity, because a subspace rotation does not change the center or the L_2-norm of any basis or the L_2-norm of any coefficient vector.

We note that, given the L_2-norm constraint $\|\mathbf{c}_i\|_2 = 1$, the L_1-norm $\|\mathbf{c}_i\|_1$ is minimized when only one element in $\mathbf{c}_i$ is 1, and 0 for all others. Therefore, in the absence of feature noise, the sparsity regularization favors shape bases whose coefficients are mostly zero except for a few; in the presence of noise, it will balance between fitting error and coefficient sparsity. Such shape bases are intuitive for human observers because they are similar to what is observed in the input images.

We note that, for any solution $\mathbf{p} = \{\mathbf{K}_i, \mathbf{R}_i, \mathbf{o}_i, \mathbf{c}_i, \mathcal{Z}_k\}$, we can always use $\{\mathbf{t}_k\}$, $\{\alpha_k\}$, and $\{\beta_i\}$ defined above to transform $\mathbf{p}$ to an equivalent solution that satisfies the three groups of constraints. We can then refine it using the following optimization procedure.

4.1. Minimization of Eq. (5)

We divide all of the unknowns in two groups: camera variables $\{\mathbf{K}_i, \mathbf{R}_i, \mathbf{t}_i, \mathbf{c}_i\}$ and shape variables $\{\mathcal{Z}_k\}$. Such a division allows us to build our solution upon existing bundle adjustment techniques [6]. Indeed, if $K = 1$, our problem is the same as the classic metric bundle adjustment problem. When $K > 1$, our problem is equivalent to a bundle adjustment problem in which each camera has an additional parameter $\mathbf{c}_i$, and each shape point has a dimensionality of $3K$ rather than 3.

Specifically, following [6], we use the *axis-angle* representation for $\mathbf{R}_i$ and the local tangent plane parameterization for the unit vector $\mathbf{c}_i$; doing so removes the constraints on $\mathbf{c}_i$. Since the number of views is often large, this parameterization makes the constrained optimization more efficient. We do not use reparameterization to remove the first two groups of constraints on shape bases, because it is not easy to find an explicit parameterization that always satisfies these constraints. Even if such a parameterization exists, we prefer to leave them as constraints because each constraint contains many variables. Explicitly parameterizing them will very likely damage the sparsity of the Hessian matrix in the optimization procedure.

At a current solution $\mathbf{p}$, we compute the Hessian matrix $\mathbf{H}$[2] and gradient vector $\mathbf{g}$ and seek to compute the update vector $\Delta\mathbf{p}$ subject to the two constraints on shape bases. We linearize these two groups of constraints as

$$\forall k, \quad \sum_{n=1}^{N} \Delta\mathbf{Z}_{n,k} = \mathbf{0}_{3\times1}, \quad \sum_{n=1}^{N} \mathbf{Z}_{n,k} \cdot \Delta\mathbf{Z}_{n,k} = 0. \qquad (6)$$

We represent these $4K$ constraints in matrix form as $\mathbf{C}\Delta\mathbf{p} = 0$ and compute $\Delta\mathbf{p}$ by solving the following linear system augmented by a Lagrangian multiplier λ

$$\begin{bmatrix} \mathbf{H} & \mathbf{C}^{\mathrm{T}} \\ \mathbf{C} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \Delta\mathbf{p} \\ \lambda \end{bmatrix} = \begin{bmatrix} \mathbf{g} \\ \mathbf{0} \end{bmatrix}. \qquad (7)$$

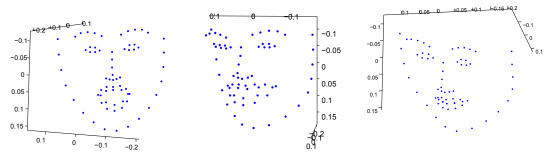
In fact, our method is the standard Sequential Quadratic Programming (SQP) method; we provide details here to illustrate how the classical sparse bundle adjustment technique can be conveniently extended to solve our problem with constraints.

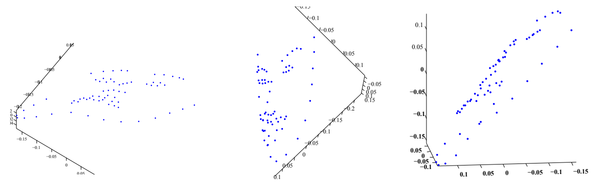
We use the Schur-Complement method [7] to solve this equation. We first compute λ by solving $(\mathbf{C}\mathbf{H}^{-1}\mathbf{C}^{\mathrm{T}})\lambda = \mathbf{C}\mathbf{H}^{-1}\mathbf{g}$ and then compute $\Delta\mathbf{p} = \mathbf{H}^{-1}\mathbf{g} - \mathbf{H}^{-1}\mathbf{C}^{\mathrm{T}}\lambda$. Note that in this process, the key intermediate results we need to compute are $\mathbf{A} = \mathbf{H}^{-1}\mathbf{C}^{\mathrm{T}}$ and $\mathbf{b} = \mathbf{H}^{-1}\mathbf{g}$. Computing the inverse of $\mathbf{H}$ is expensive, but we can compute it by solving $\mathbf{H}\mathbf{A} = \mathbf{C}^{\mathrm{T}}$ and $\mathbf{H}\mathbf{b} = \mathbf{g}$, respectively. Solving an equation like $\mathbf{H}\mathbf{f} = \mathbf{q}$ is exactly what a classic sparse bundle adjustment does. In summary, our iterative procedure works as follows:

```
1: evaluate Hessian $\mathbf{H}$, gradient $\mathbf{g}$, and constraint matrix $\mathbf{C}$;
2: compute $[\mathbf{A}, \mathbf{b}] = \mathbf{H}\backslash[\mathbf{C}^{\mathrm{T}}, \mathbf{g}]$;
3: compute $\lambda = (\mathbf{C}\mathbf{A})\backslash\mathbf{C}\mathbf{b}$;
4: compute $\Delta\mathbf{p} = \mathbf{b} - \mathbf{A}\lambda$.
```

We augment the diagonal of $\mathbf{H}$ as done in Levenberg-Marquardt [7] so the step size is automatically selected. We approximate $|x|$ as $\sqrt{x^2 + \epsilon^2} - \epsilon$ so that its derivative can be conveniently computed without the kink.

[2] For readers familiar with the notations in Hartley and Zisserman [6], $\mathbf{H}$ has the form $\begin{bmatrix} \mathbf{U}^* & \mathbf{W} \\ \mathbf{W}^{\mathrm{T}} & \mathbf{V}^* \end{bmatrix}$, which is augmented by multiplying their diagonal entries by a factor of $1 + \kappa$, for varying parameter κ.


The ground truth shape bases


Initial shape bases

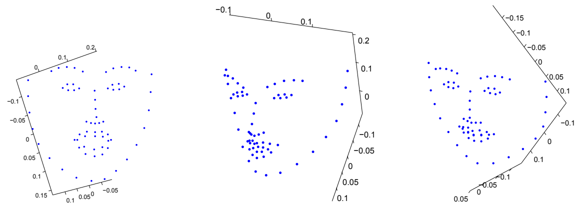

Final estimated shape bases using sparse regularization.

Figure 2. The effectiveness of sparse regularization in Eq. (5). The initial bases (middle row) span the same linear subspace as the ground truth bases do (top row). However, they require a denser set of combination coefficients. After bundle adjustment with sparse regularization, the final basis estimates (bottom row) resemble ground truth bases well because the Eq. (5) favors bases with sparse coefficients.

5. Experimental Results

We have implemented both model evolution and compressive shape bundle adjustment in C++. We created both real and synthetic data to evaluate our methods.

5.1. Effectiveness of Sparse Regularization in Eq. (5)

In this example, we demonstrate the effectiveness of the sparse regularization in Eq. (5) and show that it favors basis shapes that resemble actual shapes well, rather than an arbitrary linear combination of bases. We generate five ground truth basis shapes; due to space constraints only three are shown in Figure 2. For each basis shape, we project it to 10 different images. With these 50 images, we estimate the shape bases. We set our initial shape bases by rotating the ground truth bases by a 5×5 rotation matrix, shown in the second row of Figure 2; we also apply the inverse of the same rotation matrix to the basis coefficient vectors, as in Eq. (4). These initial bases span the same subspace as the ground truth bases, although they do not resemble natural looking faces. If we set the sparse regularization weight to $\tau = 0$ in Eq. (5), the optimization will not update the initial bases because they fit the image data perfectly. If we set $\tau = 100$, the final estimated basis shapes resemble the ground truth bases well because, with these bases, the linear combination coefficients are sparse.

5.2. SUV Example

We have collected a set of 100 SUV images from the web, containing 20 different SUV makes (Touareg, Landcruiser, *etc.*), examples of which are shown in Figure 3. Each SUV is shown in 5 images in the set. Since the viewpoints of images for each particular SUV model are randomly distributed and are often widely separated, it is difficult to reconstruct the complete 3D model for each vehicle individually using a traditional approach. We defined $N = 54$ local features (*e.g.*, door knobs, windshield corners) that are common to all SUV models in our image set. We manually marked the visible features on each of the 100 images.

There are two challenges in applying previous NRSFM methods to this data set: (1) *the large amount of missing features ($> 60\%$) due to the significant viewpoint differences* and (2) *the strong perspective effect in the images*. As discussed in Section 2.4, no previous methods have been successful in the presence of both challenges.

Indeed, we have experimented with one of the state-of-the-art methods [11] that has shown promising performance with respect to missing data. It unfortunately does not perform effectively on our data, which we believe is due to the fact that the method assumes that missing data is randomly distributed. In practice the distribution of missing data due to occlusion is structural.

Figure 4 shows the result of our method. This result verifies that the recovered compressive 3D model fits the input image features well. We use $Q = 16$ in the model evolution, $\tau = 100$, 5 basis shapes, and a maximum of 20 bundle adjustment iterations for estimating the compressive shape representations. The model evolution takes 32 seconds and the compressive shape bundle adjustment takes 1.5 minutes.

5.3. Deforming Spiral Example

In this example, we synthetically created a deformation process in which a spiral object is bent gradually from a cylinder shape to a torus shape. Such a deformation is highly non-linear. We sampled 100 points uniformly along the spiral and generated 500 *perspective* images depicting the deformation process from random viewpoints around the object. Some of the input images are shown in Figure 5. Each image has about 50 visible feature points. Occlusions are simulated by computing the inner product between line of sight and surface normal, which means they are structural rather than randomly distributed on the surface. This data set has all the challenges that existing NRSFM methods cannot handle: perspective camera projection, severe occlusion due to large viewpoint change, and large non-linear deformation.

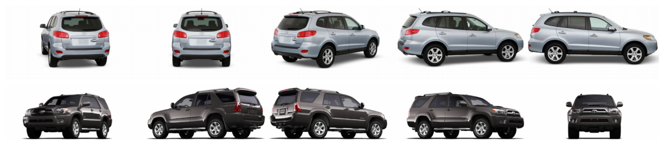


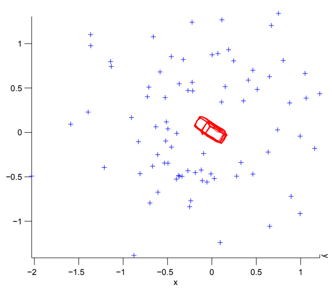
Figure 3. A subset of images used in our experiment for SUV shape reconstruction. Our full collection has 100 images in total, containing 20 different SUV makes each with 5 images. Since the viewpoints of images for each particular SUV model are randomly distributed and are often *widely separated*, it is very challenging to take a traditional approach to *individually* reconstruct the complete 3D model for each vehicle. We demonstrate that, by combining all the images together, the family of the SUV shapes can be recovered.

Figure 6 shows the result of our method. Again, the recovered compressive 3D model fits the input image features well. We use $Q = 16$ in model evolution, and $\tau = 100$, 50 basis shapes (much larger than the SUV example), a maximum of 20 bundle adjustment iterations for estimating the compressive shape representations. The model evolution takes 2.5 minutes, and the compressive shape bundle adjustment takes 2.5 hours for this case due to the larger number of points and views compared to the SUV example.

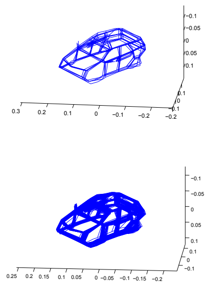
5.4. Human Skeletal Motion Example

We now evaluate our method on input data with smooth camera trajectories. We generate the input using the golf motion from the CMU Mocap database (subject #63). The motion is represented as a sequence of joint angles and we applied different skeleton parameters from 20 people (also from the same data base) on this motion and generated 20 3D motion sequences. Golf swing is a very fast motion; in order to guarantee a continuous deformation hence a connected model graph, we adaptively interpolated the 3D motion data so that the inter-frame differences are below a threshold (roughly 2% of skeleton size). Then we project each interpolated 3D sequence to a camera rotating around the subject (0.5 degree per frame) with a random initial camera position. Each sequence has about 400 frames and in total we have 7924 input frames. For each frame, we sort the 3D points based on their distance to the camera and discard the farthest 30% to simulate structural occlusion.

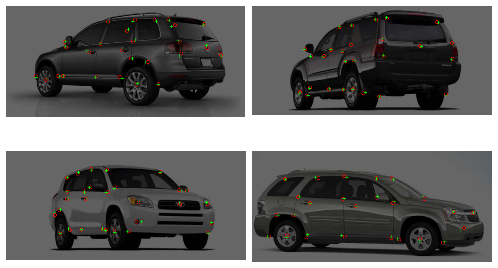
We use $Q = 16$ for this data set. After model graph traversal, we obtained 7866 3D models. We first verify that for most frames, its ground truth model is very close to at least one of its associated models. Specifically, for each frame, we compute the 3D registration error between its ground truth and its 3D reconstruction and record the min error. Figure 7 shows the histogram of this min error over all the frames, which suggests



(a)



(b)



(c)

Figure 4. Illustration of the compressive shape models recovered for SUV examples. (a) An overlay of 5 SUV basis shape (shown as the red-colored point cluster at the center) and the camera viewpoints (shown as the scattered surrounding point cloud). (b) A closeup view of the overlay of 5 basis shapes (top) and an overlay of 100 recovered SUV shapes represented by the basis shapes (bottom). Note that these shapes have been registered for illustration purposes. (c) An overlay of both reprojected 3D points (green crosses) and 2D features (red circles) on four of the input images. Most of the red crosses are invisible because they are underneath the green circles. The average reprojection error is 4-6 pixels and the scale of SUV in the images is around 300-400 pixels, so the error is about 1-2%. **Best viewed electronically.**

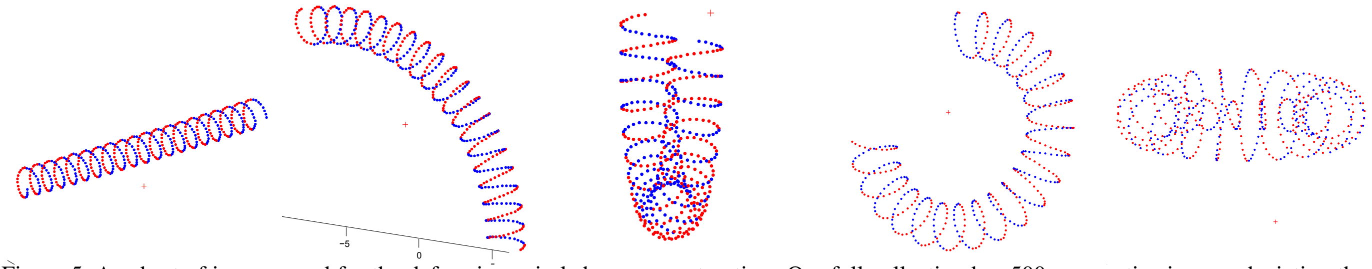


Figure 5. A subset of images used for the deforming spiral shape reconstruction. Our full collection has 500 perspective images depicting the deformation process from random viewpoints around the object. The deformation process bends a cylinder shape (leftmost image) into a torus shape (rightmost image). 100 points are sampled along the spiral, and each image sees about 50 of them due to occlusion. The occlusion is simulated by evaluating the inner product between the line of sight and the surface normal. For illustration purposes, we show visible points in blue and occluded points in red. The isolated red cross point indicates the location of the camera. This data set has all the challenges that existing NRSFM cannot handle: perspective projection, significant occlusion, and large non-linear deformation. **Best viewed electronically.**

that for more than 80% of the frames, this min error is below 0.1, about 6% of the model scale.

For the rest 20% of the frames, their ground truth models are not similar to any of their reconstructions. This is because during model graph traversal, once a frame is used to reconstruct a model, it is removed from $\mathcal{L}$. That is, a frame is used to reconstruct multiple consistent models but it may not necessarily be used to reconstruct a model that is similar to its ground truth. In practice, we found that, the ground truth models for these 20% frames can be found within the full set of 7866 reconstructed models. In particular, for each of the 20% frames, we can find one model within the 7866 reconstructed models with less than 6% 3D registration error. That is, the full set of reconstructed models and the set of ground truth models are very similar; but not every frame is associated with a model that is very close to its ground truth due to projection ambiguity. This ambiguity cannot be resolved without additional assumption, such as temporal coherence in a video. If we keep traversing the model graph (without removing images from $\mathcal{L}$), eventually each frame will be associated with a model that is very close to its ground truth; however, this will be very time-consuming. We have not run compressive bundle adjustment on this data set because of its size.

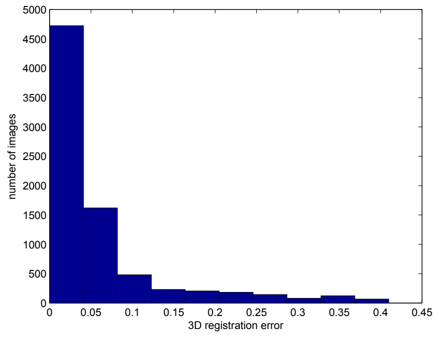


Figure 7. The histogram of the minimum alignment error between the reconstructed 3D models for each image and its ground truth model after model graph traversal. See Section 5.4 for details.

5.5. Intuitions for Choosing Q and ε

Q is the number of images in a cluster. If input images contain few occluded feature points, a small Q (such as 5) can be used; otherwise a larger Q is needed to ensure a complete model can be reconstructed from Q images.

ε is the maximum reprojection error allowed for images in a

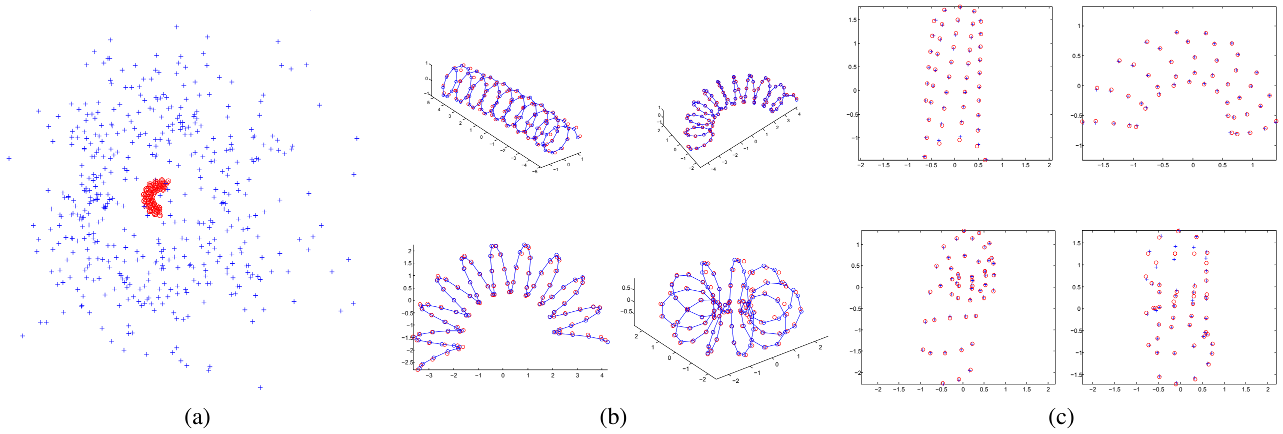


Figure 6. Illustration of the compressive shape models recovered in the deforming spiral example. (a) A recovered spiral basis shape (shown as the red-colored point cluster at the center) and the camera viewpoints (shown as the surrounding blue points). (b) An overlay of reconstructed model and ground truth model from viewpoints that are not in the input viewpoints. Note these models are registered for comparison. (c) An overlay of both reprojected 3D points (blue crosses) and 2D features (red circles). The average reprojection error ranges from 0.005-0.02 units (this is a synthetic data set; the units are arbitrary) and the scale of spiral is 3 units, so the error is about 0.2-0.7%. **Best viewed electronically.**

cluster to be considered consistent. A small ε limits the number of consistent clusters in the graph model and may cause the model graph to be unconnected; consequently our traversal algorithm may not find a model for all images. On the other hand, if ε is too large, inconsistent clusters will be mislabeled as consistent, thereby generating inaccurate models. In our experiments, ε was set to 3%-5% of the image size.

6. Discussion and Future Work

In this paper, we have introduced a new concept we call a *model graph*, which we use as a framework for NRSFM. Based on this concept, we develop a new algorithm called *model evolution* that addresses three challenges that existing NRSFM methods face: severe occlusion, perspective camera projection, and large non-linear deformation. We also propose a compressive shape representation to reduce the model complexity produced by the model evolution algorithm, and demonstrate how to extend the classic bundle adjustment technique to estimate the compressive shape representation. Our preliminary experiments are encouraging. There are three future research directions we are planning to pursue. First, our current model graph traversal is quite heuristic; we are interested in design efficient optimal algorithms to traverse the graph. Second, we would like to investigate ways to automatically identify seed clusters, and extend our approach to handle image sets that include multiple connected components, which may correspond to completely different objects. Third, we are developing robust feature matching algorithms that can be used to automatically generate input data for our NRSFM algorithm.

7. Acknowledgements

This work is supported in part by NSF IIS-0845916 and a Sloan Research Fellowship. Brandon Smith is also supported by a NSF graduate fellowship.

References

[1] S. Agarwal, N. Snavely, I. Simon, S. M. Seitz, and R. Szeliski. Building rome in a day. In *ICCV*, 2009.

[2] I. Akhter, Y. Sheikh, and S. Khan. In defense of orthonormality constraints for nonrigid structure from motion. In *CVPR*, 2009.

[3] M. Brand. Morphable 3D models from video. In *CVPR*, pages 456–463, 2001.

[4] M. Brand. A direct method for 3d factorization of nonrigid motion observed in 2d. In *CVPR*, volume 2, pages 122–128, 2005.

[5] C. Bregler, A. Hertzmann, and H. Biermann. Recovering nonrigid 3D shape from image streams. In *CVPR*, 2000.

[6] R. I. Hartley and A. Zisserman. *Multiple View Geometry*. Cambridge University Press, 2000.

[7] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer, 1999.

[8] V. Rabaud and S. Belongie. Re-thinking non-rigid structure from motion. In *CVPR*, 2008.

[9] L. Torresani and A. Hertzmann. Automatic non-rigid 3d modeling from video. In *ECCV*, pages 299–312, 2004.

[10] L. Torresani, A. Hertzmann, and C. Bregler. Learning non-rigid 3d shape from 2d motion. In *Advances in Neural Information Processing Systems*, pages 1555–1562, 2004.

[11] L. Torresani, A. Hertzmann, and C. Bregler. Non-rigid structure-from-motion: Estimating shape and motion with hierarchical priors. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 30(5):878–892, May 2008.

[12] L. Torresani, D. B. Yang, E. J. Alexander, and C. Bregler. Tracking and modeling non-rigid objects with rank constraints. In *CVPR*, pages 493–500, 2001.

[13] R. V. and B. S. Linear embeddings in non-rigid structure from motion. In *CVPR*, 2009.

[14] J. Xiao, J. Chai, and T. Kanade. A closed-form solution to non-rigid shape and motion recovery. *Int. J. on Computer Vision*, 67(2):233–246, 2006.

[15] J. Yan and M. Pollefeys. Automatic kinematic chain building from feature trajectories of articulated objects. In *CVPR*, 2006.