Deep Non-Rigid Structure from Motion

Chen Kong

CMU-RI-TR-19-43

The Robotics Institue
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Thesis Committee:
Simon Lucey, Carnegie Mellon University, Chair
David Held, Carnegie Mellon University
Aswin Sankaranarayanan, Carnegie Mellon University
Hongdong Li, Australian National University

*Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy.*

Keywords: Nonrigid structure from motion, 3D reconstruction, hierarchical sparse coding, deep neural network, compressed sensing, dictionary learning, optimization, reconstructability.

Abstract

Non-Rigid Structure from Motion (NRSfM) refers to the problem of reconstructing cameras and the 3D point cloud of a non-rigid object from an ensemble of images with 2D correspondences. Current NRSfM algorithms are limited from two perspectives: (i) the number of images, and (ii) the type of shape variability they can handle. These difficulties stem from the inherent conflict between the condition of the system and the degrees of freedom needing to be modeled – which has hampered its practical utility for many applications within vision. In this paper we propose a novel hierarchical sparse coding model for NRSFM which can overcome (i) and (ii) to such an extent, that NRSFM can be applied to problems in vision previously thought too ill posed. Our approach is realized in practice as the training of an unsupervised deep neural network (DNN) auto-encoder with a unique architecture that is able to disentangle pose from 3D structure. Using modern deep learning computational platforms allows us to solve NRSfM problems at an unprecedented scale and shape complexity. Our approach has no 3D supervision, relying solely on 2D point correspondences. Further, our approach is also able to handle missing/occluded 2D points without the need for matrix completion. Extensive experiments demonstrate the impressive performance of our approach where we exhibit superior precision and robustness against all available state-of-the-art works in some instances by an order of magnitude. We further propose a new quality measure (based on the network weights) which circumvents the need for 3D ground-truth to ascertain the confidence we have in the reconstructability. We believe our work to be a significant advance over state-of-the-art in NRSFM.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Building an AI capable of inferring the 3D structure and pose of an object from a single image is a problem of immense importance. Training such a system using supervised learning requires a large number of labeled images, and how to obtain these labels is currently an open problem for the vision community. Rendering [36] is problematic as the synthetic images seldom match the appearance and geometry of the objects we encounter in the real-world. Hand annotation is preferable, but current strategies rely on associating the natural images with an external 3D dataset (e.g. ShapeNet [12], ModelNet [45], which we refer to as *3D supervision*). If the 3D shape dataset does not capture the variation we see in the imagery, then the problem is inherently ill-posed.

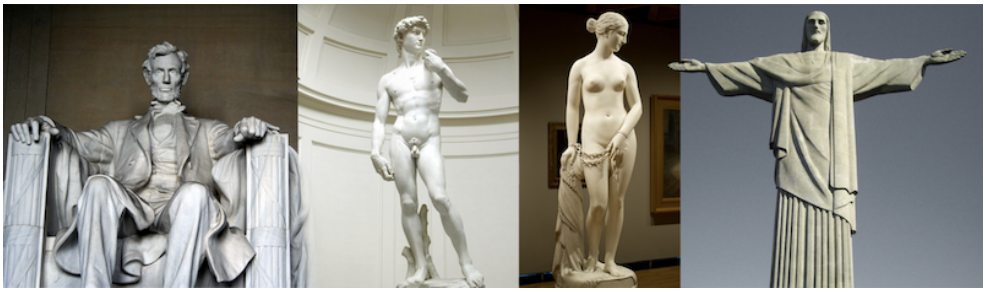


Figure 1.1: The set of 3D shapes describing an object category (e.g.statue) is inherently non-rigid, even though individual objects within the category may be rigid.

Non-Rigid Structure from Motion (NRS*f*M) offers computer vision a way out of this quandary by recovering the pose and 3D structure of an object category *solely* from hand annotated 2D landmarks with no need of 3D supervision. Classically [9], the problem of NRS*f*M has been applied to objects that move non-rigidly over time, such as the human body and face. But NRS*f*M is not restricted to non-rigid objects; it can equally be applied to rigid objects with object categories that deform non-rigidly [3, 27, 43]. Consider, for example, the four objects in Figure 1.1, instances from the visual object category "statue." Each object in isolation represents a rigid statue, but the set of all 3D shapes describing "statue" is non-rigid. In other words, each object instance can be modeled as a certain deformation from its category's general shape.

NRS*f*M is well-noted in previous research as an ill-posed problem due to the non-rigidity. This has been mainly addressed by imposing additional shape priors, e.g. low rank [9, 14], and union-of-subspaces [3, 52]. Specifically, the low-rank assumption states that the non-rigid object, a set of 3D shapes, can be approximated well by a linear combination of the same few dictionary bases (see Figure 1.2 left). The union-of-subspaces assumes that when a non-rigid object is complex, we can cluster the shape variations into clusters and apply a low-rank assumption in each shape cluster. Though these two priors achieve great success, their drawbacks considerably limit their applications: 1) low rank is only applicable to simple non-rigid objects with limited deformations and 2) union-of-subspaces relies heavily on frame clustering, which has difficulty scaling up to large image collections.

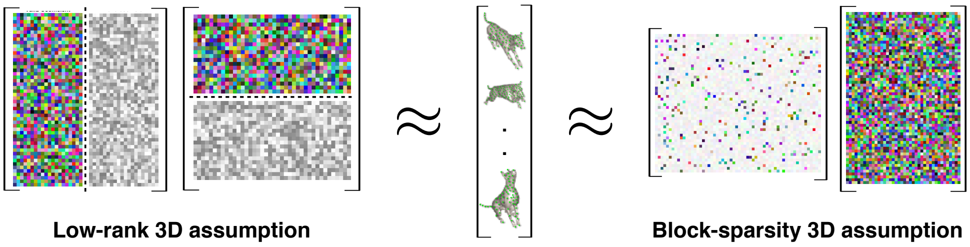


Figure 1.2: This thesis assumes a set of 3D shapes, stemming from a non-rigid 3D structure, can be approximated well by a few (i.e. K) examples of elements from an unknown basis or dictionary. Classical low-rank NRS*f*M makes a similar assumption but assumes that the same K elements within the dictionary will be used to approximate all 3D shape instances (see left). Our approach differs in this regard, where we allow for the employment of different K elements within the dictionary for each 3D instance (see right).

This thesis proposes a similar assumption to classical low-rank NRS*f*M [9, 14], where we assume each 3D shape instance can be described using only K dictionary bases, but a different set of K basis vectors can be employed for each shape instance (see Figure 1.2 right). These sets of 3D shape instances do not form a single linear subspace, they can instead be thought of as existing in a union of $\binom{L}{K}$ subspaces where L is the total number of basis vectors available. An obvious advantage of this compressible 3D structure assumption is the ability to model a much broader set of 3D structures than both low-rank and union-of-subspaces. A drawback to the assumption, however, is discovering which of the potentially very large number of $\binom{L}{K}$

subspaces best describes the actual 3D shape instance - solely from its 2D projection. In this project, this assumption is referred to as the block-sparsity assumption. Chapter 3, describes the block-sparsity assumption and proposes an innovative algorithm based on block sparse dictionary learning to solve problems that were previously deemed intractable using the low-rank assumption employed by current state-of-the-art methods [14, 23]. Chapter 4, proposes to use the Alternating Direction Method of Multipliers (ADMMs) [8] with convex relaxation to further improve the robustness and effectively optimize the dictionary learning objective.

Finally, to solve the difficulty of searching the best subspaces out of the large number of candidates, this thesis proposes a novel shape prior using *hierarchical* block-sparsity. Compared to the above block sparse prior, the hierarchical block-sparsity introduces additional layers; as such, that the sparse code of current layer is represented by the subsequent dictionary sparsely. These introduced additional layers, compared to single-layer sparse coding, are capable of controlling the number of subspaces by learning from data so that invalid subspaces are removed while sufficient subspaces remain for modeling shape variations. In other words, the number of subspaces is not solely related to $\binom{L}{K}$ but also adjusted by the additional dictionaries. Chapter 5 describes this prior and further builds a deep neural network to minimize the hierarchical block sparse dictionary learning objective.

1.1 Contributions

- We propose a novel shape prior based on sparse coding and demonstrate that the 2D projections under weak perspective cameras can be represented by the dictionary in a 2×3 block-sparse way. Based on this insight, we re-interpret NRSfM as a block sparse dictionary learning problem. We theoretically characterize the uniqueness of block sparse dictionary learning. Further, we show how the uniqueness of block sparse dictionary learning can be utilized to efficiently recover the camera motion and 3D structures.
- We propose to approximate the proposed objective by a convex relaxation and demonstrate that the proposed objective can be optimized in an iterative manner using the Alternating Direction Method of Multipliers (ADMMs) algorithm.
- We propose a novel shape prior based on hierarchical sparse coding and demonstrate that the 2D projections under weak perspective cameras can be represented by the hierarchical dictionaries in a block sparse way. Through recent theoretical innovations [32], we then show how this problem can be reinterpreted as a feed-forward Deep Neural Network(DNN) auto-encoder that can be efficiently solved through modern deep learning environments. Our employment of DNNs moves from an opaque black-box to a transparent "glass-box"

in terms of its interpretability. Our deep NRS*f*M is capable of handling hundreds of thousands of images and learning large parameterizations to model non-rigidity.

- Extensive experiments are conducted on the above three algorithms and our approach, especially the deep solution, outperforms state-of-the-art methods in the order of magnitude on a number of benchmarks. Both quantitative and qualitative results demonstrate our superior performance. Moreover, we propose a measure of model quality (using the coherence of a learned dictionary), which helps to avoid over-fitting, especially when ground-truth of training data are not available.

Chapter 2

Background

Inferring the 3D geometry of objects and camera positions of a scene/object from an ensemble of 2D projected points is known within the field of computer vision as ”structure from motion” (SfM). There are two core components associated with SfM: (i) correspondence, and (ii) inversion. Correspondence deals with the problem of determining the location of matching points across semantically similar images (e.g. statues in Figure 1.1) while rejecting points that have no matches, such as those arising from background, clutter, or occluding content. Once correspondence has been established, the inverse problem of recovering the 3D structure from the 2D point projections must be solved, requiring a priori constraints on the structure and camera (projection) matrix. This thesis focuses on the latter problem (i.e.inversion).

The field of computer vision has made significant progress on this 3D reconstruction problem for rigid scenes/objects over the last three decades. It is now capable of reconstructing entire cities using large-scale photo collections [2] and real-time visual SLAM on embedded and mobile devices [37]. However, Non-Rigid Structure from Motion (NRSfM) has long been a “poor cousin” to rigid SfM. Unlike rigid SfM, canonical NRSfM methods: (i) do not scale well when applied to large datasets, (ii) are sensitive to noise in correspondence estimation, and (iii) have found few useful applications beyond being a theoretical curiosity for computer vision. This chapter starts by introducing a factorization-style algorithm, the Tomasi Kanade factorization [38] and then from there goes through a representative NRSfM algorithm. Finally, the chapter presents the mutual coherence, its applications in the uniqueness of sparse dictionary learning and the recent progress on explaining deep neural network via convolutional sparse coding.

2.1 Tomasi and Kanade's factorization

Tomasi and Kanade's celebrated work proposes to recover the shape and motion of a rigid object under orthography without computing depth as an intermediate step. Suppose that we have F images and each image has corresponding P key points across all images. Note that all these points are visible in the image (i.e. there is no missing data in the current problem formulation). We denote (u_{fp}, v_{fp}) as the image coordinates of p-th point on f-th image, from which we define the measurement matrix in $2F \times P$ as

$$\mathbf{W} = \begin{bmatrix} u_{11} & u_{12} & \cdots & u_{1P} \\ v_{11} & v_{12} & \cdots & v_{1P} \\ \vdots & \vdots & \ddots & \vdots \\ u_{F1} & u_{F2} & \cdots & u_{FP} \\ v_{F1} & v_{F2} & \cdots & v_{FP} \end{bmatrix}. \tag{2.1}$$

Further, we denote (x_p, y_p, z_p) as the world coordinates of p-th point and then define the 3D structure matrix in $3 \times P$ as

$$\mathbf{S} = \begin{bmatrix} x_1 & x_2 & \cdots & x_P \\ y_1 & y_2 & \cdots & y_P \\ z_1 & z_2 & \cdots & z_P \end{bmatrix}. \tag{2.2}$$

Note that the Tomasi and Kanade algorithm focuses on rigid objects so the 3D structures are identical across frames; as a result, the 3D structure matrix is not related to the frame of image sequence.

Since Tomasi and Kanade focus on orthogonal projection, the camera projection for each frame degenerates to a pure rotation. Formally, denote $\mathbf{R}_f$ in shape 2×3 as the projection matrix. By concatenating all projection matrices together, we have

$$\mathbf{R} = \begin{bmatrix} \mathbf{R}_1 \\ \mathbf{R}_2 \\ \vdots \\ \mathbf{R}_F \end{bmatrix} \tag{2.3}$$

By the projection equation, it is derived that

$$\mathbf{W} = \mathbf{R}\mathbf{S}, \tag{2.4}$$

which implies that the measurement matrix is factorized into two matrices. Since the number of points and the number of frames are typically much greater than three, three dominates the rank of the measurement matrix. The observation that the rank of measurement matrix is no greater than three is the heart of Tomasi and Kanade's algorithm, which is different from bundle adjustment or stereo recosntructions.

By Singular Value Decomposition (SVD), one can always factorize the measurement matrix as

$$\mathbf{W} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T. \tag{2.5}$$

To maintain the rank-three of the measurement matrix, we can simply discard the singular values, the elements on the main diagonal of Σ, except the first three. We use tilde to denote the new factorization, i.e.

$$\mathbf{W} \approx \tilde{\mathbf{U}}\tilde{\mathbf{\Sigma}}\tilde{\mathbf{V}}^T, \tag{2.6}$$

where $\tilde{\mathbf{U}} \in \mathbb{R}^{2F\times 3}, \tilde{\Sigma} \in \mathbb{R}^{3\times 3}, \tilde{\mathbf{V}} \in \mathbb{R}^{P\times 3}$.

By denoting

$$\tilde{\mathbf{R}} = \tilde{\mathbf{U}}\left[\tilde{\mathbf{\Sigma}}\right]^{\frac{1}{2}}, \tag{2.7}$$

$$\tilde{\mathbf{S}} = \left[\tilde{\mathbf{\Sigma}}\right]^{\frac{1}{2}}\tilde{\mathbf{V}}^T, \tag{2.8}$$

it is implied that

$$\mathbf{W} \approx \tilde{\mathbf{R}}\tilde{\mathbf{S}}. \tag{2.9}$$

Until now, we have factorized the measurement matrix into two sub-matrices that have the same dimension we want. However, the decomposition is not unique: any invertible 3×3 matrix $\mathbf{G}$ could be inserted between the factorization and still maintain a valid decomposition i.e.

$$\mathbf{W} \approx \tilde{\mathbf{R}}\tilde{\mathbf{S}} = \tilde{\mathbf{R}}\mathbf{G}\mathbf{G}^{-1}\tilde{\mathbf{S}} = \hat{\mathbf{R}}\hat{\mathbf{S}}. \tag{2.10}$$

Moreover, the rotation matrix $\mathbf{R}$ is expected to satisfy orthonormal constraints:

$$\mathbf{R}_f\mathbf{R}_f^T = \mathbf{I}_2, \tag{2.11}$$

where $\mathbf{I}_2$ denotes the 2×2 identity matrix. Next, we present the Tomasi and Kanade method to recover the matrix $\mathbf{G}$ such that $\hat{\mathbf{R}}$ holds Equation 2.11.

We first divide the $2F \times 3$ matrix $\tilde{\mathbf{R}}$ into 2×3 block and denote f-th block as $\tilde{\mathbf{R}}_f$. Therefore, we want to find a G such that

$$\tilde{\mathbf{R}}_f\mathbf{G}\mathbf{G}^T\tilde{\mathbf{R}}_f^T = \mathbf{I}_2, \text{for } f = 1, 2, \cdots, F. \tag{2.12}$$

By denoting

$$\mathbf{G}\mathbf{G}^T = \mathbf{Q}, \tag{2.13}$$

it is implied that

$$\tilde{\mathbf{R}}_f \mathbf{Q} \tilde{\mathbf{R}}_f^T = \mathbf{I}_2, \text{for } f = 1, 2, \cdots, F. \tag{2.14}$$

From the property of the Kronecker product and matrix vectorization, one can vectorize the both sides of Equation 2.14 and have

$$\left(\tilde{\mathbf{R}}_f \otimes \tilde{\mathbf{R}}_f\right)\mathbf{q} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix}, \text{for } f = 1, 2, \cdots, F, \tag{2.15}$$

where $\otimes$ is the Kronecker product and $\mathbf{q} \in \mathbb{R}^9$ is the vectorization of $\mathbf{Q}$. By concatenating all equations for $f = 1, 2, 3, \cdots, F$ and solving the consequent linear system, one can obtain $\mathbf{q}$. Give $\mathbf{q}$, $\mathbf{Q}$ can be estimated by reshaping $\mathbf{q}$ back to the matrix and then $\mathbf{G}$ could be solved by SVD from Equation 2.13. Finally, the rotation matrix is

$$\hat{\mathbf{R}} = \tilde{\mathbf{R}}\mathbf{G} \tag{2.16}$$

and the 3D structure matrix is

$$\hat{\mathbf{S}} = \mathbf{G}^{-1}\tilde{\mathbf{S}}. \tag{2.17}$$

We summarize the Tomasi and Kanade's algorithm below:

```
Algorithm 1: Tomasi and Kanade's algorithm
  Data: The 2D measurement matrix $\mathbf{W}$ defined in Equation 2.1
  Result: The orthogonal camera matrix $\mathbf{R}$ defined in Equation 2.3 and the 3D structure
          matrix $\mathbf{S}$ defined in Equation 2.2
  1. Factorize $\mathbf{W}$ via SVD and keep the largest three singular value;
  2. Compute $\tilde{\mathbf{R}}, \tilde{\mathbf{S}}$ via Equation 2.7 and 2.8;
  3. Compute $\mathbf{q}$ by solving a linear system defined in Equation 2.15;
  4. Compute $\mathbf{G}$ by factorizing $\mathbf{Q}$;
  return The rotation matrix $\hat{\mathbf{R}} = \tilde{\mathbf{R}}\mathbf{G}$ and structure $\hat{\mathbf{S}} = \mathbf{G}^{-1}\tilde{\mathbf{S}}$;
```

There are two drawbacks of Tomasi and Kanade's algorithm.

- The camera assumption has to be orthogonal projection, which is seldom in real world application since scales and translations always exist in image projections. Even though there is the potential to generalize to weak-perspective projection, scales dramatically hurt the precision of the solution to the Equation 2.15.

- Tomasi and Kanade assume that all points are visible, which is seldom true either. There exists a high probability of occlusion, as well as mis-detected points during key point detection or key point annotation.

2.2 Bregler's non-rigid extension and Dai's solution

The majority of videos online are about non-rigid objects, e.g.a moving person, cute pets, or sports. This motivates researchers not only to reconstruct rigid objects from videos but also to focus on non-rigid objects. However, non-rigid objects are more challenging. Inspired by Tomasi and Kanade's algorithm introduced in above section, Bregler et al. [9] extended the idea of rank-three to low-rank. Formally, the 2D projection matrix is denoted as:

$$\mathbf{W} = \begin{bmatrix} u_{11} & u_{12} & \cdots & u_{1P} \\ v_{11} & v_{12} & \cdots & v_{1P} \\ \vdots & \vdots & \ddots & \vdots \\ u_{F1} & u_{F2} & \cdots & u_{FP} \\ v_{F1} & v_{F2} & \cdots & v_{FP} \end{bmatrix}, \tag{2.18}$$

the 3D structure as:

$$\mathbf{S} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1P} \\ y_{11} & y_{12} & \cdots & y_{1P} \\ z_{11} & z_{12} & \cdots & z_{1P} \\ \vdots & \vdots & \ddots & \vdots \\ x_{F1} & x_{F2} & \cdots & x_{FP} \\ y_{F1} & y_{F2} & \cdots & y_{FP} \\ z_{F1} & z_{F2} & \cdots & z_{FP} \end{bmatrix}, \tag{2.19}$$

and the camera rotations as:

$$\mathbf{R} = \begin{bmatrix} \mathbf{R}_1 & & & \\ & \mathbf{R}_2 & & \\ & & \ddots & \\ & & & \mathbf{R}_F \end{bmatrix}, \tag{2.20}$$

where $\mathbf{R}_f$ for $f = 1, 2, \cdots, F$ are 2×3 orthogonal matrix i.e.

$$\mathbf{R}_f \mathbf{R}_f^T = \mathbf{I}_2. \tag{2.21}$$

Therefore, the projection equation is

$$\mathbf{W} = \mathbf{R}\mathbf{S}. \tag{2.22}$$

Bregler et al.proposed to reshape the structure matrix so that each row represents a single 3D shape, where the reshaped structure matrix is

$$\mathbf{S}^\sharp = \begin{bmatrix} x_{11} & y_{11} & z_{11} & x_{12} & y_{12} & z_{12} & \cdots & x_{1P} & y_{1P} & z_{1P} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ x_{F1} & y_{F1} & z_{F1} & x_{F2} & y_{F2} & z_{F2} & \cdots & x_{FP} & y_{FP} & z_{FP} \end{bmatrix}. \tag{2.23}$$

In the case of a rigid object, 3D shapes are all identical across the shape. In other words, the rank of $\mathbf{S}^\sharp$ is one. Bregler et al.therefore proposed that the rank of $\mathbf{S}^\sharp$ of a non-rigid object is K,

$$\operatorname{rank}(\mathbf{S}^\sharp) = K, \tag{2.24}$$

where K is much smaller than the number of frames F and the number of points P. In other words, the rank of $\mathbf{S}^\sharp$ is low. Therefore, the rank of $\mathbf{S}$ is expected to be $3K$ i.e.

$$\operatorname{rank}(\mathbf{S}) = 3K. \tag{2.25}$$

This assumption is typically referred to as the low rank assumption in the field and inspires a broad range of works achieving impressive success in NRS*f*M area.

Dai et al.utilize this assumption in their CVPR best paper [13] and assert that the low rank assumption itself provides sufficient constraints so that no additional priors are needed to solve a NRS*f*M algorithm. Dai et al.also propose an algorithm for it, which follows below.

First, by the low rank assumption, Dai et al.propose to represent the 3D structure by a local subspace i.e. a set of k shape bases $\mathbf{B}_1, \mathbf{B}_2, \cdots, \mathbf{B}_K$, where $\mathbf{B}_i \in \mathbb{R}^{3\times P}$. Formally,

$$\mathbf{S} = \begin{bmatrix} c_{11}\mathbf{I}_3 & c_{12}\mathbf{I}_3 & \cdots & c_{1K}\mathbf{I}_3 \\ c_{21}\mathbf{I}_3 & c_{22}\mathbf{I}_3 & \cdots & c_{2K}\mathbf{I}_3 \\ \vdots & \vdots & \ddots & \vdots \\ c_{F1}\mathbf{I}_3 & c_{F2}\mathbf{I}_3 & \cdots & c_{FK}\mathbf{I}_3 \end{bmatrix} \begin{bmatrix} \mathbf{B}_1 \\ \mathbf{B}_2 \\ \vdots \\ \mathbf{B}_K \end{bmatrix} \tag{2.26}$$

By introducing the Kronecker product, and denoting the linear combination parameter matrix as $\mathbf{C}$ and the shape bases matrix as $\mathbf{B}$, it is identical to write

$$\mathbf{S} = (\mathbf{C} \otimes \mathbf{I}_3)\mathbf{B}. \tag{2.27}$$

By plugging Equation 2.27 into the projection matrix, we have

$$\mathbf{W} = \mathbf{R}(\mathbf{C} \otimes \mathbf{I}_3)\mathbf{B}. \tag{2.28}$$

Define

$$\mathbf{\Pi} = \mathbf{R}(\mathbf{C} \otimes \mathbf{I}_3), \tag{2.29}$$

and we can factorize the measurement matrix as

$$\mathbf{W} = \mathbf{\Pi}\mathbf{B}, \tag{2.30}$$

where $\mathbf{\Pi} \in \mathbb{R}^{2F\times 3K}, \mathbf{B} \in \mathbb{R}^{3K\times P}$. Similar to Tomasi and Kanade's algorithm, Dai et al. also utilize SVD to factorize the matrix $\mathbf{W}$ and keep the greatest $3K$ singular values:

$$\mathbf{W} \approx \hat{\mathbf{W}} = \hat{\mathbf{\Pi}}\hat{\mathbf{B}}. \tag{2.31}$$

One can see that Dai et al. also face the ambiguity of factorization; i.e. any invertible matrix $\mathbf{G} \in \mathbb{R}^{3K\times 3K}$ could be inserted between $\mathbf{\Pi}$ and $\mathbf{B}$ so that

$$\mathbf{W} = \hat{\mathbf{\Pi}}\mathbf{G}\mathbf{G}^{-1}\hat{\mathbf{B}} = \mathbf{\Pi}\mathbf{B}. \tag{2.32}$$

Dai et al. demonstrate that it is not necessary to recover $\mathbf{G}$ entirely, but only three columns are sufficient to reconstruct the camera matrices, which is the key observation in their paper.

Denote the i-th doulb erows of $\hat{\mathbf{\Pi}}$ as $\hat{\mathbf{\Pi}}_{2i-1:2i} \in \mathbb{R}^{2\times 3K}$ and the k-th triplet column of $\mathbf{G}$ as $\mathbf{G}_k \in \mathbb{R}^{3K\times 3}$. It is implied that

$$\hat{\mathbf{\Pi}}_{2i-1:2i}\mathbf{G}_k = \mathbf{\Pi} = c_{ik}\mathbf{R}_i, \text{for } i = 1, 2, \cdots, F, k = 1, 2, \cdots, K. \tag{2.33}$$

By the orthogonal constraint defined in Equation 2.21, it is expected that

$$\hat{\mathbf{\Pi}}_{2i-1:2i}\mathbf{G}_k\mathbf{G}_k^T\hat{\mathbf{\Pi}}_{2i-1:2i}^T = c_{ik}^2\mathbf{R}_i\mathbf{R}_i^T = c_{ik}^2\mathbf{I}_2. \tag{2.34}$$

By denoting $\mathbf{Q}_k$ as

$$\mathbf{Q}_k = \mathbf{G}_k\mathbf{G}_k^T, \tag{2.35}$$

we have

$$\hat{\mathbf{\Pi}}_{2i-1:2i}\mathbf{Q}_k\hat{\mathbf{\Pi}}_{2i-1:2i}^T = c_{ik}^2\mathbf{I}_2. \tag{2.36}$$

Similar to Tomasi and Kanade's derivation, we denote $\mathbf{q}_k$ as the vectorization of $\mathbf{Q}_k$ and utilize Kronecker product:

$$(\hat{\mathbf{\Pi}}_{2i-1:2i} \otimes \hat{\mathbf{\Pi}}_{2i-1:2i})\mathbf{q}_k = i\begin{bmatrix} c_{ik}^2 \\ 0 \\ 0 \\ c_{ik}^2 \end{bmatrix}, \tag{2.37}$$

Since the value of c_{ik} is unknown, the orthogonal constraint actually offers two linear equations over $\mathbf{Q}_k$

$$\begin{bmatrix} (\hat{\mathbf{\Pi}}_{2i-1:2i} \otimes \hat{\mathbf{\Pi}}_{2i-1:2i})(1,:) - (\hat{\mathbf{\Pi}}_{2i-1:2i} \otimes \hat{\mathbf{\Pi}}_{2i-1:2i})(4,:) \\ (\hat{\mathbf{\Pi}}_{2i-1:2i} \otimes \hat{\mathbf{\Pi}}_{2i-1:2i})(2,:) \end{bmatrix} \mathbf{q}_k = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \tag{2.38}$$

where $(\hat{\mathbf{\Pi}}_{2i-1:2i} \otimes \hat{\mathbf{\Pi}}_{2i-1:2i})(j,:)$ is the j-th row of $(\hat{\mathbf{\Pi}}_{2i-1:2i} \otimes \hat{\mathbf{\Pi}}_{2i-1:2i})$. For conciseness, we define capital $\mathbf{0}$ as vector filled with zero and

$$\mathbf{A}_i = \begin{bmatrix} (\hat{\mathbf{\Pi}}_{2i-1:2i} \otimes \hat{\mathbf{\Pi}}_{2i-1:2i})(1,:) - (\hat{\mathbf{\Pi}}_{2i-1:2i} \otimes \hat{\mathbf{\Pi}}_{2i-1:2i})(4,:) \\ (\hat{\mathbf{\Pi}}_{2i-1:2i} \otimes \hat{\mathbf{\Pi}}_{2i-1:2i})(2,:) \end{bmatrix}, \tag{2.39}$$

then it is written as

$$\mathbf{A}_i \mathbf{q}_k = \mathbf{0}. \tag{2.40}$$

By stacking all such equations for all frames $i = 1, 2, \cdots, F$, we finally collect all equations over $\mathbf{q}_k$ that is

$$\begin{bmatrix} \mathbf{A}_1 \\ \mathbf{A}_2 \\ \vdots \\ \mathbf{A}_F \end{bmatrix} \mathbf{q}_k = \mathbf{A}\mathbf{q}_k = \mathbf{0}. \tag{2.41}$$

By Equation 2.35, it is clear that the rank of $\mathbf{Q}$ is three i.e.

$$\operatorname{rank}(\mathbf{Q}_k) = 3. \tag{2.42}$$

However, optimization over a fixed rank is not an easy problem to solve. Therefore, Dai et al.propose to relax the problem to a trace norm minimization objective. Formally,

$$\begin{aligned} &\min_{\mathbf{Q}_k} \operatorname{trace}(\mathbf{Q}_k) \\ &\text{s.t. } \mathbf{Q}_k \succeq 0, \\ &\quad \mathbf{A}\mathbf{q}_k = \mathbf{0}, \end{aligned} \tag{2.43}$$

where $\succeq 0$ denotes semi-definite matrix. Dai et al.propose to minimize this objective by off-the-shell SDP solvers. For a higher precision, they utilize a non-linear optimization as a refinement procedure whose objective is

$$\min_{\mathbf{G}_k} \sum_{i=1}^{F} \left(1 - \frac{(\hat{\mathbf{\Pi}}_{2i-1:2i} \otimes \hat{\mathbf{\Pi}}_{2i-1:2i})(4,:)}{(\hat{\mathbf{\Pi}}_{2i-1:2i} \otimes \hat{\mathbf{\Pi}}_{2i-1:2i})(1,:)}\right)^2 + \left(2\frac{(\hat{\mathbf{\Pi}}_{2i-1:2i} \otimes \hat{\mathbf{\Pi}}_{2i-1:2i})(2,:)}{(\hat{\mathbf{\Pi}}_{2i-1:2i} \otimes \hat{\mathbf{\Pi}}_{2i-1:2i})(1,:)}\right)^2. \tag{2.44}$$

Once we obtain the corrective matrix i.e. $\mathbf{G}_k$, Dai et al.demonstrate that it is sufficient to recover camera matrices: for camera matrix on i-th image $\mathbf{R}_i$

$$\hat{\mathbf{\Pi}}_{2i-1:2i}\mathbf{G}_k = c_{ik}\mathbf{R}_i. \tag{2.45}$$

Though sign ambiguity still remains, one can estimate the value of c_{ik} and the corresponding rotation matrix $\mathbf{R}_i$,

Until now, we are able to estimate the camera matrix $\mathbf{R}$; the next job is to estimate the structure matrix $\mathbf{S}$. Dai et al.propose two algorithms to solve it.

- Pointed by work [20], the Moore-Penrose pseudo-inverse solution

  $$\mathbf{S} = \mathbf{R}^{\dagger}\mathbf{W} \tag{2.46}$$

  is a unique solution that minimizes the nuclear norm i.e. $\|\mathbf{S}\|_1$ as well as the solution to the projection equation.

- Different from the above pseudo-inverse method minimizing the rank of $\mathbf{S}$, Dai et al.propose to minimize the rank of $\mathbf{S}^{\sharp}$. Formally,

  $$\begin{aligned} &\min_{\mathbf{S}} \operatorname{rank}(\mathbf{S}^{\sharp}) \\ &\text{s.t. } \mathbf{W} = \mathbf{R}\mathbf{S}. \end{aligned} \tag{2.47}$$

  Then, Dai et al.relax the rank-minimization to nuclear-norm minimization and re-cast the objective in Lagrangian form:

  $$\min_{\mathbf{S}} \mu\|\mathbf{S}^{\sharp}\|_1 + \frac{1}{2}\|\mathbf{W} - \mathbf{R}\mathbf{S}\|_F^2, \tag{2.48}$$

  where μ is introduced as the continuation (homotopy) parameter which diminishes as the algorithm iterates. Then Dai et al.utilize proximal gradient descent to minimize the objective iteratively.

We summarize the Dai et al.'s algorithm below:

```
Algorithm 2: Dai et al.'s algorithm
  Data: The 2D measurement matrix $\mathbf{W}$
  Result: The orthogonal camera matrix $\mathbf{R}$ and the 3D structure matrix $\mathbf{S}$
  1. Factorize $\mathbf{W}$ into $\hat{\mathbf{\Pi}}$ and $\hat{\mathbf{B}}$ via SVD and keep the largest $3K$ singular value;
  2. Compute $\mathbf{q}$ by solving a SDP problem defined in 2.43;
  3. Compute corrective matrix $\mathbf{G}$ via SVD given $\mathbf{Q}$;
  4. Compute $\mathbf{R}$ via Equation 2.45;
  5. Compute $\mathbf{S}$ via either pseudo-inverse or proximal gradient descent.;
  return The rotation matrix and structure;
```

Dai et al.'s algorithm has two major problems:

1. The value of rank K has to be selected by cross validation. Given a novel sequence, one has to try several possible K for experiments and then select the K corresponding to the lowest 3D structure error. This is problematic, since in real-world applications, 3D ground truth is never approachable, otherwise it is meaningless to solve. One alternative way to cross validate is finding a K that minimizes 2D reprojection error. However, this is

problematic too. There is a high possibility that the 2D reprojection error will decrease with the increase of K because the system is less constrained and there is a higher degree of freedom to model 2D measurement when K becomes bigger, while completely failing to reconstruct the correct 3D structure.

2. The second drawback makes the first even worse; that is, Dai et al.'s algorithm is quite slow. The long running time comes from the utility of non-linear optimization, which serves an important functionality in a designed algorithm. This long running time for each parameter configuration makes cross validation even less practical.

2.3 Trajectory reconstruction

The low-rank assumption can be applied not only to shape spaces but also to trajectory space, generally considered as a dual space. One of the most representative works is Akhter et al. [6]. They propose that instead of imposing compactness (low-rank) on shape, they can impose the compactness across time, in other words, on trajectory. Formally, define

$$\mathbf{T}_x(i) = \begin{bmatrix} x_{1i} \\ x_{2i} \\ \vdots \\ x_{Fi} \end{bmatrix}, \tag{2.49}$$

$$\mathbf{T}_y(i) = \begin{bmatrix} y_{1i} \\ y_{2i} \\ \vdots \\ y_{Fi} \end{bmatrix}, \tag{2.50}$$

$$\mathbf{T}_z(i) = \begin{bmatrix} z_{1i} \\ z_{2i} \\ \vdots \\ z_{Fi} \end{bmatrix}, \tag{2.51}$$

as the X, Y, Z coordinates of the ith trajectory. Akhter then utilizes the low-rank assumption into the trajectory space; that is, each trajectory component can be approximated by a linear combination of a small number of trajectory basis:

$$\mathbf{T}_x(i) = \sum_{j=1}^{K} a_{xj}(i)\boldsymbol{\theta}^j, \tag{2.52}$$

$$\mathbf{T}_y(i) = \sum_{j=1}^{K} a_{yj}(i)\boldsymbol{\theta}^j, \tag{2.53}$$

$$\mathbf{T}_z(i) = \sum_{j=1}^{K} a_{zj}(i)\boldsymbol{\theta}^j, \tag{2.54}$$

where $\theta^j \in \mathbb{R}^F$ for $j = 1, 2, ..., K$ are trajectory bases. To build up the relationship between Akhter et al.'s method and Dai et al.'s method, we explicitly write these equations. For simplicity, we define the 3D structure as

$$\mathbf{S} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1P} \\ y_{11} & y_{12} & \cdots & y_{1P} \\ z_{11} & z_{12} & \cdots & z_{1P} \\ \vdots & \vdots & \ddots & \vdots \\ x_{F1} & x_{F2} & \cdots & x_{FP} \\ y_{F1} & y_{F2} & \cdots & y_{FP} \\ z_{F1} & z_{F2} & \cdots & z_{FP} \end{bmatrix}, \tag{2.55}$$

the bases matrix as

$$\boldsymbol{\Theta} = \begin{bmatrix} \theta_{11}\mathbf{I}_3 & \theta_{12}\mathbf{I}_3 & \cdots & \theta_{1K}\mathbf{I}_3 \\ \vdots & \vdots & \ddots & \vdots \\ \theta_{F1}\mathbf{I}_3 & \theta_{F2}\mathbf{I}_3 & \cdots & \theta_{FK}\mathbf{I}_3 \end{bmatrix}, \tag{2.56}$$

and the parameter matrix as

$$\mathbf{A} = \begin{bmatrix} a_{x1}(1) & a_{x1}(2) & \cdots & a_{x1}(P) \\ a_{y1}(1) & a_{y1}(2) & \cdots & a_{y1}(P) \\ a_{z1}(1) & a_{z1}(2) & \cdots & a_{z1}(P) \\ \vdots & \vdots & \ddots & \vdots \\ a_{xK}(1) & a_{xK}(2) & \cdots & a_{xK}(P) \\ a_{yK}(1) & a_{yK}(2) & \cdots & a_{yK}(P) \\ a_{zK}(1) & a_{zK}(2) & \cdots & a_{zK}(P) \end{bmatrix}. \tag{2.57}$$

Therefore, from Equation 2.52, 2.53 and 2.54, we have

$$\mathbf{S} = \boldsymbol{\Theta}\mathbf{A}. \tag{2.58}$$

Recall Dai et al.'s algorithm: $\boldsymbol{\Theta}$ was considered as parameters while $\mathbf{A}$ as bases. In this trajectory space, however, $\boldsymbol{\Theta}$ are trajectory bases while $\mathbf{A}$ are parameters. Based on this observation, Ahkter et al.propose that trajectory space and shape space form a certain type of duality.

To solve the problem, Ahkter et al.move forward and introduce an additional prior, positing that points moves smoothly and continuously in time. Based on this assumption, the point trajectory $\mathbf{T}_x(i), \mathbf{T}_y(i), \mathbf{T}_z(i)$ for $i = 1, 2, ..., P$ are considered as smooth and continuous. Therefore, Ahkter et al.exploit the smoothness to predefine the trajectory basis—using the Discrete Cosine Transform (DCT) basis. To demonstrate the performance of the DCT basis, they conduct experiments to compare human motion trajectory to the DCT bases. We borrow this image from [6] and shown in Figure 2.1.

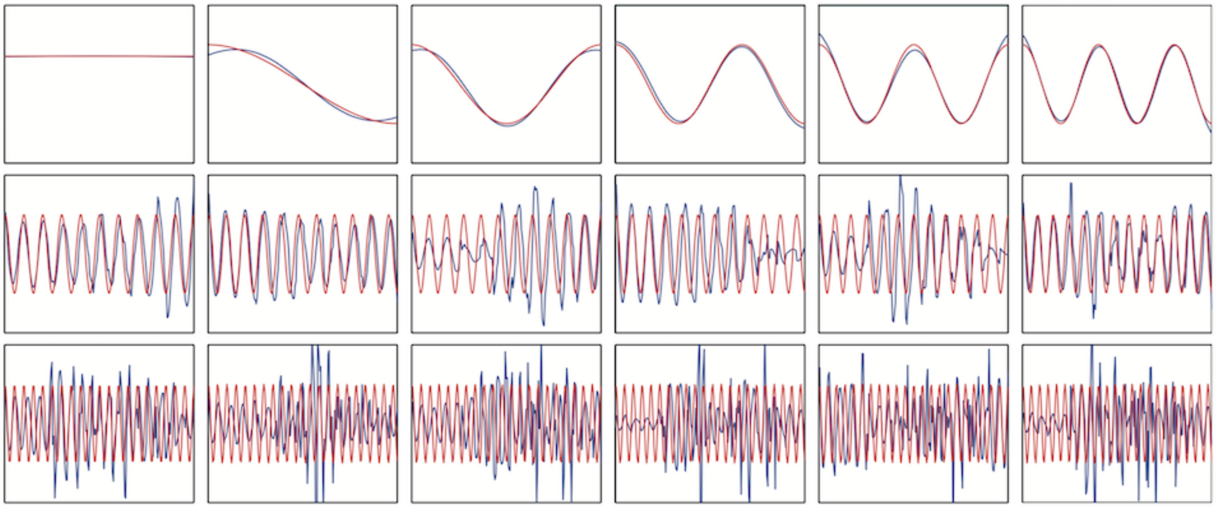


Figure 2.1: The comparison of PCA (blue) and DCT (red) as the trajectory basis for the CMU motion capture data. Here, we plot the 1st-6th, 21st-26th, and 41st-46th PCA and DCT basis. The plot shows the close resemblance between the two, especially for initial PCA basis. Some of the bases have been multiplied by -1 for better visual comparison.

With the help of DCT bases, Akhter et al. propose an algorithm similar to Dai et al.'s work. They first multiply the camera matrix

$$\mathbf{R} = \begin{bmatrix} \mathbf{R}_1 & & & \\ & \mathbf{R}_2 & & \\ & & \ddots & \\ & & & \mathbf{R}_F \end{bmatrix}, \tag{2.59}$$

to both sides of Equation 2.58, resulting in

$$\mathbf{W} = \mathbf{R}\mathbf{\Theta}\mathbf{A} = \mathbf{\Lambda}\mathbf{A}, \tag{2.60}$$

where

$$\mathbf{\Lambda} = \mathbf{R}\mathbf{\Theta} \tag{2.61}$$

is a $3F \times 3K$ matrix. The rank of $\mathbf{W}$ will be at most $3K$ similar to Dai et al.'s work. Then, by using SVD, one can factorize $\mathbf{W}$ into two parts:

$$\mathbf{W} = \hat{\mathbf{\Lambda}}\hat{\mathbf{A}}. \tag{2.62}$$

where $\hat{\mathbf{\Lambda}} \in \mathbb{R}^{2F \times 3K}$ and $\hat{\mathbf{A}} \in \mathbb{R}^{3K \times P}$. Again, this factorization is not unique and needs to estimate a corrective matrix $\mathbf{G}$ such that

$$\mathbf{\Lambda} = \hat{\mathbf{\Lambda}}\mathbf{G}, \tag{2.63}$$

$$\mathbf{A} = \mathbf{G}^{-1}\hat{\mathbf{A}}. \tag{2.64}$$

Ahkter et al.propose that instead of estimating the entire matrix of $\mathbf{G}$, actually the first three columns of $\mathbf{G}$ are sufficient to reconstruct the entire $\mathbf{\Lambda}$. Denote $\mathbf{G}_{|||}$ as the first column triple of the matrix $\mathbf{G}$. Then we have

$$\hat{\mathbf{\Lambda}}\mathbf{G}_{|||} = \begin{bmatrix} \theta_{11}\mathbf{R}_1 \\ \theta_{21}\mathbf{R}_2 \\ \vdots \\ \theta_{F1}\mathbf{R}_F \end{bmatrix}. \tag{2.65}$$

Since cameras are satisfied orthogonal constraint, i.e.

$$\mathbf{R}_i\mathbf{R}_i^T = \mathbf{I}_2, \text{for } i = 1, 2, \cdots, F, \tag{2.66}$$

we have

$$\hat{\mathbf{\Lambda}}_{2i-1:2i}\mathbf{G}_{|||}\mathbf{G}_{|||}^T\hat{\mathbf{\Lambda}}_{2i-1:2i}^T = \theta_{i1}^2\mathbf{I}_2. \tag{2.67}$$

Again, by the Kronecker product and its property, one can equally write this as

$$(\hat{\mathbf{\Lambda}}_{2i-1:2i} \otimes \hat{\mathbf{\Lambda}}_{2i-1:2i})\mathbf{q}_{|||} = \theta_{i1}^2 \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix}, \tag{2.68}$$

where

$$\mathbf{q}_{|||} = vec(\mathbf{G}_{|||}\mathbf{G}_{|||}^T). \tag{2.69}$$

Note that θ_{i1} are known since they come from predefined DCT bases. By concatenating all equations across frames, one can solve $\mathbf{q}_{|||}$ and consequently solve $\mathbf{G}_{|||}$. Given $\mathbf{G}_{|||}$, by Equation 2.65, one can estimate $\mathbf{R}_1, \mathbf{R}_2, ...\mathbf{R}_F$. Therefore, given all camera matrices and predefined DCT bases

θ_{ij}, we can compute entire $\mathbf{\Lambda}$ by

$$\mathbf{\Lambda} = \begin{bmatrix} \theta_{11}\mathbf{R}_1 & \theta_{12}\mathbf{R}_1 & \cdots & \theta_{1K}\mathbf{R}_1 \\ \theta_{21}\mathbf{R}_2 & \theta_{22}\mathbf{R}_2 & \cdots & \theta_{2K}\mathbf{R}_2 \\ \vdots & \vdots & \ddots & \vdots \\ \theta_{F1}\mathbf{R}_F & \theta_{F2}\mathbf{R}_F & \cdots & \theta_{FK}\mathbf{R}_F \end{bmatrix} \tag{2.70}$$

Given $\mathbf{\Lambda}$, one solves a linear system:

$$\mathbf{W} = \mathbf{\Lambda}\mathbf{A}, \tag{2.71}$$

to compute $\mathbf{A}$. Once $\mathbf{A}$ are computed, the 3D structure $\mathbf{S}$ is obtained by

$$\mathbf{S} = \mathbf{\Theta}\mathbf{A}. \tag{2.72}$$

We summarize Akhter et al.'s algorithm below:

```
Algorithm 3: Akhter et al.'s algorithm
  Data: The 2D measurement matrix $\mathbf{W}$
  Result: The orthogonal camera matrix $\mathbf{R}$ and the 3D structure matrix $\mathbf{S}$
  1. Factorize $\mathbf{W}$ into $\hat{\mathbf{\Lambda}}$ and $\hat{\mathbf{A}}$ via SVD and keep the largest $3K$ singular value;
  2. Compute $\mathbf{q}_{|||}$ by solving a linear system defined in 2.68;
  3. Compute corrective matrix $\mathbf{G}$ via SVD given $\mathbf{q}_{|||}$;
  4. Compute $\mathbf{R}$ via Equation 2.65;
  4. Compute $\mathbf{\Lambda}$ via Equation 2.70;
  4. Compute $\mathbf{A}$ by solving a linear system defined in Equation 2.71;
  return The rotation matrix $\mathbf{R}$ and structure $\mathbf{S} = \mathbf{\Lambda}\mathbf{A}$;
```

Akhter et al.'s algorithm is very similar to Dai et al.'s in terms of algorithm design, but the biggest difference is the additional prior—smoothness of motion—and from that the predefined bases. One of advantages of these predefined bases is to increase the performance, especially when solving a corrective matrix. Specifically, in Dai et al.'s algorithm, since shape parameters c_{ik} for $i = 1, 2, \cdots, F$ and $k = 1, 2, \cdots, K$ are unknown, orthogonal constraints on each camera defined in 2.37 solely offer two linear equations and therefore totally $2F$ equations. However, in Akhter et al.'s algorithm, orthogonal constraints in 2.68 provide $3F$ equations totally. This change is not just about adding more constraints but shifting the problem from finding the best solution in null space (Dai et al.) to a simple pseudo-inverse (Akhter et al.). With the former, it is hard to reach global optimal, but the latter has a closed-form global optimal solution.

However, introducing the predefined bases is not free food, which also brings drawbacks:

1. Assuming the smoothness of object motion directly limits Akhter et al.'s algorithm into reconstructing a single object from continuous video clips. Specifically, it is quite common to have several video clips about the same object, where reconstructing the 3D shape holistically is the optimal solution. Further, as explained in the introduction, NRS*f*M is not just about non-rigid objects but can also be equally applied to rigid objects (i.e. object category, e.g. a set of tables). In the case of object category, Akhter et al.'s algorithm completely fails.
2. Utilizing DCT bases to represent trajectory precisely requires sufficient high frequency basis components, while the low rank assumption, conversely, restricts the number of bases that are used here. This conflict dramatically restricts the type of shape variation that Akhter et al.could handle. Also, selecting proper bases via cross validation is also frustrating.

2.4 Complex Shape Recosntruction by Union of Subspaces

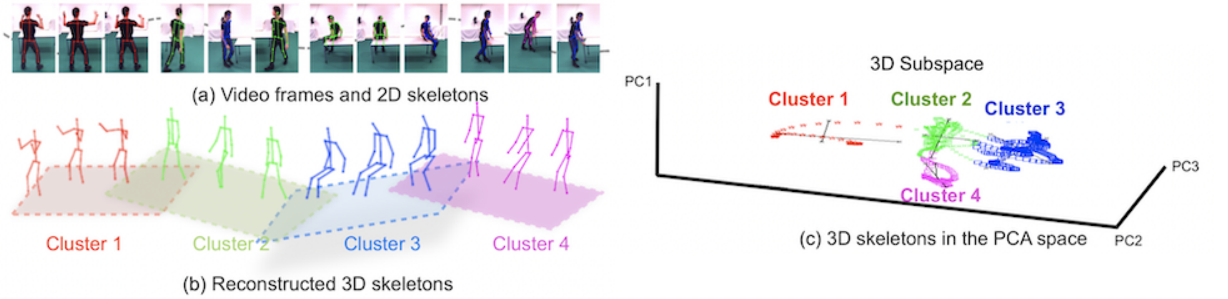

(a) Video frames and 2D skeletons
Cluster 1
Cluster 2
Cluster 3
Cluster 4
(b) Reconstructed 3D skeletons
3D Subspace
PC1
Cluster 1
Cluster 2
Cluster 3
Cluster 4
PC3
PC2
(c) 3D skeletons in the PCA space

Figure 2.2: Borrowed from [52]. An example of complex non-rigid motion using a human body. (a) A video sequence from the UMPM dataset[42] in which a subject sequentially performs actions such as: raise hand (red), walk (green), sit (blue) and stand (magenta). 2D body joints tracked in the videos are connected to form 2D skeletons in each frame. (b) Reconstructed and clustered 3D skeletons using our method. Different color represents different clusters/subspaces obtained by Zhu et al.'s method. (c) Projection of the 3D-skeletons in the local subspaces spanned by the three largest principal components (PCs). Observe that the human poses stemming from different actions adhere to separate local subspaces/clusters and the overall complex nonrigid motion lies in a union of subspaces.

The above algorithms, including Bregler et al., Dai et al.and Akhter et al.'s work, are all based on the low-rank shape assumption. From different perspectives, they demonstrate its effectiveness in reconstructing non-rigid objects. However, representing shape variations via a linear combination of a limited number of bases is obviously only capable of handling primitive or simple motions, e.g. walking, sitting, or jumping. However, what if we have a sequence of complex motions concatenating primitive motions together? Zhu et al. [52] ask this question and extend the low-rank assumption to the union of subspaces.

An obvious solution is to first group video sequences into clusters and apply Dai et al.'s algorithm in each cluster. Note that in each cluster, video frames are not necessarily continuous and thus Akhter et al.'s work fails in this case. To illustrate this idea, we borrow an image from [52] and show it in Figure 2.2. One can see that the sequence is divided into four clusters and each cluster represents a primitive motion: raising a hand, walking, sitting, and standing. Based on this idea, Zhu et al.conducted experiments to cluster video frames based on 2D annotated key points. However, their experiments demonstrate that clustering based on 2D information is less effective compared to 3D shape clustering. We also borrow one image from [52] to show the experiment results in Figure 2.3. One can see that no matter whether one uses a static camera or moving camera, 2D LLR Clustering has a substantial difference from the 3D LLR Clustering that serves as ground truth. Zhu et al.pointed out that the confusion between cluster 2 and cluster 4 is potentially caused by the information loss after camera projection.

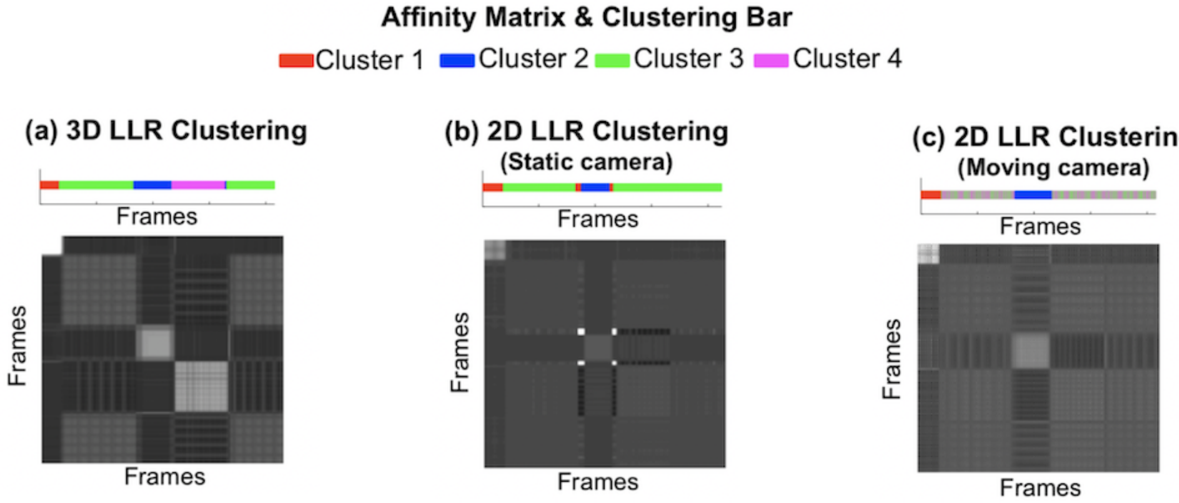


Figure 2.3: Clustering results from [52]. 3D LRR subspace clustering vs. 2D LRR subspace clustering on complex nonrigid motion.

From the observation of the 2D clustering failure, Zhu et al. move forward and propose an objective simultaneously minimizing the 3D based clustering error and 2D reprojection errors. First, Zhu et al. assumes that all cameras are known and focus on reconstructing 3D structure.

This is quite different from NRSfM work. Even though knowing the camera beforehand dramatically simplifies the problem, the theoretical contribution of using the union of subspaces is of specific interest to us. Formally, Zhu et al.defines

$$\mathbf{X} = \begin{bmatrix} x_{11} & y_{11} & z_{11} & x_{12} & y_{12} & z_{12} & \cdots & x_{1P} & y_{1P} & z_{1P} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ x_{F1} & y_{F1} & z_{F1} & x_{F2} & y_{F2} & z_{F2} & \cdots & x_{FP} & y_{FP} & z_{FP} \end{bmatrix}, \tag{2.73}$$

its reshape

$$\mathbf{X}^{\sharp} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1P} \\ y_{11} & y_{12} & \cdots & y_{1P} \\ z_{11} & z_{12} & \cdots & z_{1P} \\ \vdots & \vdots & \ddots & \vdots \\ x_{F1} & x_{F2} & \cdots & x_{FP} \\ y_{F1} & y_{F2} & \cdots & y_{FP} \\ z_{F1} & z_{F2} & \cdots & z_{FP} \end{bmatrix}, \tag{2.74}$$

and camera matrix

$$\mathbf{R} = \begin{bmatrix} \mathbf{R}_1 \\ \mathbf{R}_2 \\ \vdots \\ \mathbf{R}_F \end{bmatrix}. \tag{2.75}$$

Inspired by the Local Rank Representation (LRR), Zhu et al. introduces an affinity matrix $\mathbf{Z}$ and proposes to minimize the objective:

$$\begin{aligned} &\min_{\mathbf{X},\mathbf{Z},\mathbf{E}} \|\mathbf{Z}\|_* + \gamma\|\mathbf{X}\|_* + \lambda\|\mathbf{E}\|_1 \\ &\text{s.t. } \mathbf{X} = \mathbf{X}\mathbf{Z} \\ &\qquad \mathbf{W} = \mathbf{R}\mathbf{X}^{\sharp} + \mathbf{E}, \end{aligned} \tag{2.76}$$

where λ and γ are penalty parameters for terms $\|\mathbf{X}\|_*$ and $\|\mathbf{E}\|_1$. There are three terms in the objective:

1. Union of subspaces term: By modeling the 3D shape

$$\mathbf{X} = \mathbf{X}\mathbf{Z}, \tag{2.77}$$

   and at the same time forcing $\|\mathbf{Z}\|_*$ to be small, Zhu et al. want the 3D structure $\mathbf{X}$ to be represented by a union of local subspaces. The division of subspaces and components are recorded by the affinity matrix $\mathbf{Z}$.

2. Low-rank term: Along with the subspace prior, Zhu et al.also put a constraint on 3D structure $\mathbf{X}^\sharp$ to make it low-rank by minimizing $\|\mathbf{X}\|_*$. The nuclear norm is the tightest convex relaxation of matrix rank. However, I believe this term is theoretically unnecessary for modeling 3D structures because simultaneously using the union of subspaces prior and the low-rank prior makes the problems conflicted and does not model either prior well. Due to not seeing any experiments from [52] demonstrating the functionality of this term, we are not sure how much contribution it makes towards performance; however, I believe that it might help to maintain the robustness of the system.
3. Reprojection error term: minimizing $\|\mathbf{E}\|_1$ is to minimize the reprojection error (i.e.the error between known 2D measurement and reprojected 3D reconstructions), which is commonly seen in NRS*f*M works. The norm here is suggested by Zhu et al. to use ℓ_1 norm, i.e. the summation of absolute values of all elements; however, the performance difference between ℓ_1 and the Frobenius norm is not clear. The motivation behind the ℓ_1 norm is mentioned in the paper to boost the sparsity of error, even though it seems unconvincing.

To solve the problem, Zhu et al. utilize Augmented Lagrangian Methods (ALMs). They first introduce an auxiliary variable $\mathbf{H}$ and then write the objective function into the Langrangian formula:

$$\begin{aligned}\mathcal{L}_{\mathbf{X},\mathbf{Z},\mathbf{E},\mathbf{H}} =& \|\mathbf{Z}\|_* + \gamma\|\mathbf{X}\|_* + \lambda\|\mathbf{E}\|_1 \\ &+ <\mathbf{\Gamma}_1, \mathbf{X} - \mathbf{X}\mathbf{Z}> + \frac{\mu_1}{2}\|\mathbf{X} - \mathbf{X}\mathbf{Z}\|_F^2 \\ &+ <\mathbf{\Gamma}_2, \mathbf{W} - \mathbf{R}\mathbf{H}^\sharp - \mathbf{E}> + \frac{\mu_2}{2}\|\mathbf{W} - \mathbf{R}\mathbf{H}^\sharp - \mathbf{E}\|_F^2 \\ &+ <\mathbf{\Gamma}_3, \mathbf{X} - \mathbf{H}> + \frac{\mu_3}{2}\|\mathbf{X} - \mathbf{H}\|_F^2,\end{aligned} \tag{2.78}$$

where $\mathbf{\Gamma}_1, \mathbf{\Gamma}_2, \mathbf{\Gamma}_3$ are Lagrangian variables sharing the same shape as $\mathbf{X}, \mathbf{E}$, and $\mathbf{H}$ respectively. $< \cdot, \cdot >$ denotes the inner product between two matrices that is the summation of elements of the element-wise multiplication between two matrices. μ_1, μ_2, μ_3 are used to control the weights of the Lagrangian terms, which are expected to diminished across iterations.

Then suggested by ALMs, Zhu et al.minimize the entire objective by solving each subproblem:

- **Minimizing over** $\mathbf{X}$: In the k-th iteration,

$$\begin{aligned}\mathbf{X}^{k+1} =& \operatorname*{argmin}_{\mathbf{X}} \mathcal{L}(\mathbf{X}, \mathbf{Z}^k, \mathbf{E}^k, \mathbf{H}^k, \mathbf{\Gamma}_1^k, \mathbf{\Gamma}_2^k, \mathbf{\Gamma}_3^k) \\ =& \gamma\|\mathbf{X}\|_* + <\mathbf{\Gamma}_1^k, \mathbf{X} - \mathbf{X}\mathbf{Z}^k> + \frac{\mu_1}{2}\|\mathbf{X} - \mathbf{X}\mathbf{Z}^k\|_F^2 \\ &+ <\mathbf{\Gamma}_3^k, \mathbf{X}> + \frac{\mu_3}{2}\|\mathbf{X} - \mathbf{H}^k\|_F^2.\end{aligned} \tag{2.79}$$

This objective can be minimized by proximal gradient descent.

- **Minimizing over** $\mathbf{Z}$: In the k-th iteration,

$$\begin{aligned}\mathbf{X}^{k+1} = & \operatorname*{argmin}_{\mathbf{Z}} \mathcal{L}(\mathbf{X}^k, \mathbf{Z}, \mathbf{E}^k, \mathbf{H}^k, \mathbf{\Gamma}_1^k, \mathbf{\Gamma}_2^k, \mathbf{\Gamma}_3^k) \\ = & \|\mathbf{Z}\|_* - < \mathbf{\Gamma}_1^k, \mathbf{X}^k\mathbf{Z} > + \frac{\mu_1}{2}\|\mathbf{X}^k - \mathbf{X}^k\mathbf{Z}\|_F^2.\end{aligned} \tag{2.80}$$

  The objective can be minimized by proximal gradient descent.

- **Minimizing over** $\mathbf{E}$: In the k-th iteration,

$$\begin{aligned}\mathbf{E}^{k+1} = & \operatorname*{argmin}_{\mathbf{E}} \mathcal{L}(\mathbf{X}^k, \mathbf{Z}^k, \mathbf{E}, \mathbf{H}^k, \mathbf{\Gamma}_1^k, \mathbf{\Gamma}_2^k, \mathbf{\Gamma}_3^k) \\ = & \lambda\|\mathbf{E}\|_1 - < \mathbf{\Gamma}_2^k, \mathbf{E} > + \frac{\mu_2}{2}\|\mathbf{W} - \mathbf{R}(\mathbf{H}^k)^\sharp - \mathbf{E}\|_F^2.\end{aligned} \tag{2.81}$$

  This objective can be minimized in an element-wise way, where each element has a closed-form solution.

- **Minimizing over** $\mathbf{H}$: In the k-th iteration,

$$\begin{aligned}\mathbf{H}^{k+1} = & \operatorname*{argmin}_{\mathbf{H}} \mathcal{L}(\mathbf{X}^k, \mathbf{Z}^k, \mathbf{E}^k, \mathbf{H}, \mathbf{\Gamma}_1^k, \mathbf{\Gamma}_2^k, \mathbf{\Gamma}_3^k) \\ = & - < \mathbf{\Gamma}_2^k, \mathbf{R}\mathbf{H}^\sharp > + \frac{\mu_2}{2}\|\mathbf{W} - \mathbf{R}\mathbf{H}^\sharp - \mathbf{E}^k\|_F^2 \\ & - < \mathbf{\Gamma}_3^k, \mathbf{H} > + \frac{\mu_3}{2}\|\mathbf{X}^k - \mathbf{H}\|_F^2.\end{aligned} \tag{2.82}$$

  This objective can be directly solved by a pseudo-inverse.

- **Updating Lagrangian variables**: In the k-th iteration, each of the Lagrangian variables can be updated by

$$\mathbf{\Gamma}_1^{k+1} = \mathbf{\Gamma}_1^k + \mu_1(\mathbf{X}^k - \mathbf{X}^k\mathbf{Z}^k), \tag{2.83}$$

$$\mathbf{\Gamma}_2^{k+1} = \mathbf{\Gamma}_2^k + \mu_2(\mathbf{W} - \mathbf{R}(\mathbf{H}^k)^\sharp - \mathbf{E}^k), \tag{2.84}$$

$$\mathbf{\Gamma}_3^{k+1} = \mathbf{\Gamma}_3^k + \mu_3(\mathbf{X}^k - \mathbf{H}^k). \tag{2.85}$$

We summarize the entire procedure in Algorithm 4.

One of the advantages of Zhu et al.'s work is that the novel shape prior to the union of subspaces, dramatically increases the shape variations that NRS*f*M can handle and provides a novel insight to model large image collections. However, there are several drawbacks:

1. The work assumes that the camera matrix is known beforehand. This assumption is less practical, especially when video clips originate from online and camera information is lost. It might be argued that cameras can be reconstructed by Dai et al.'s work and then applied to Zhu et al.'s work. This is also problematic because Dai et al.have difficulty reconstructing complex motion sequences.

```
Algorithm 4: Zhu et al.'s algorithm
  Data: The 2D measurement matrix $\mathbf{W}$, camera matrix $\mathbf{R}$
  Result: The 3D structure matrix $\mathbf{X}$ and the affinity matrix $\mathbf{Z}$
  while not converge do
      - $\mathbf{X}^{k+1} = \text{argmin}_{\mathbf{X}} \mathcal{L}(\mathbf{X}, \mathbf{Z}^k, \mathbf{E}^k, \mathbf{H}^k, \mathbf{\Gamma}_1^k, \mathbf{\Gamma}_2^k, \mathbf{\Gamma}_3^k)$;
      - $\mathbf{Z}^{k+1} = \text{argmin}_{\mathbf{Z}} \mathcal{L}(\mathbf{X}^{k+1}, \mathbf{Z}, \mathbf{E}^k, \mathbf{H}^k, \mathbf{\Gamma}_1^k, \mathbf{\Gamma}_2^k, \mathbf{\Gamma}_3^k)$;
      - $\mathbf{E}^{k+1} = \text{argmin}_{\mathbf{E}} \mathcal{L}(\mathbf{X}^{k+1}, \mathbf{Z}^{k+1}, \mathbf{E}, \mathbf{H}^k, \mathbf{\Gamma}_1^k, \mathbf{\Gamma}_2^k, \mathbf{\Gamma}_3^k)$;
      - $\mathbf{H}^{k+1} = \text{argmin}_{\mathbf{H}} \mathcal{L}(\mathbf{X}^{k+1}, \mathbf{Z}^{k+1}, \mathbf{E}^{k+1}, \mathbf{H}, \mathbf{\Gamma}_1^k, \mathbf{\Gamma}_2^k, \mathbf{\Gamma}_3^k)$;
      - Update $\mathbf{\Gamma}_1, \mathbf{\Gamma}_2, \mathbf{\Gamma}_3$.
  end
  return 3D shape matrix $\mathbf{X}$ and affinity matrix $\mathbf{Z}$;
```

2. The optimization process might take a long time to converge and has difficulty finding a good local minimum. As one can see, Zhu et al.actually introduce two auxiliary variables: $\mathbf{E}$ and $\mathbf{H}$. Pointed by [10] that it tends to converge to a worse local minimum more slowly when adding more auxiliary variables. Further, as one can see, two subproblems need to use the proximal gradient descent to minimize, where the proximal gradient descent is considered as a slow process. This results in an even slower convergence of the entire algorithm.
3. Though the union of subspaces prior is capable of handling a complex motion sequence, the proposed algorithm relying on an affinity matrix cannot. The affinity matrix $\mathbf{Z}$ is a $F \times F$ matrix. Optimizing over such a matrix is not practical when the number of frames F is more than tens of thousands. This mostly restricts Zhu et al.'s algorithm into a small-scale problem and cannot scale to a large image collection.

2.5 Other Works

Quite different from what we already present above, another type of prior is the manifold assumption [23, 33] which replaces the low-rank assumption with learning a non-linear manifold. Most notable is the recent work of Gotardo and Martinez [23], who demonstrate how the "kernel trick" could be employed to model a 3D shape as a non-linear subspace. A more recent work [28] proposes an objective from a Grassmannian perspective to solve the NRS*f*M problem in a dense scenario (i.e.the number of points is large). A drawback to these approaches, however, was their reliance on additional priors, except this manifold assumption, which, for example [23] further assumes k basis constraints and [28] assumes temporal consistency (i.e.shapes move continuously along frames). These additional priors limit these approaches' viability to real-world application.

It is worth mentioning that there is some overlap between this manifold assumption and Zhu et al.'s former union of subspaces prior, as it has been demonstrated [16] that the field of manifold learning has a strong link to the recovery of compressed signals. Specifically, it has been demonstrated that a set of K sparse signals forms a K-dimensional Riemannian manifold. Further, it can be shown [16] that many manifold models can be expressed as an infinite union of subspaces.

2.6 Deep Neural Network and Sparse Coding

Sparse dictionary learning can be considered as an unsupervised learning task and divided into two sub-problems: (i) dictionary learning, and (ii) sparse code recovery. Let us consider sparse code recovery problem, where we estimate a sparse representation $\mathbf{z}$ for a measurement vector $\mathbf{x}$ given the dictionary $\mathbf{D}$, i.e.

$$\min_{\mathbf{z}} \|\mathbf{x} - \mathbf{D}\mathbf{z}\|_2^2 \quad \text{s.t. } \|\mathbf{z}\|_0 < \lambda, \tag{2.86}$$

where λ related to the trust region controls the sparsity of recovered code. One classical algorithm to recover the sparse representation is Iterative Shrinkage and Thresholding Algorithm (ISTA) [7, 15, 34]. ISTA iteratively executes the following two steps with $\mathbf{z}^{[0]} = \mathbf{0}$:

$$\mathbf{v} = \mathbf{z}^{[i]} - \alpha \mathbf{D}^T(\mathbf{D}\mathbf{z}^{[i]} - \mathbf{x}), \tag{2.87}$$

$$\mathbf{z}^{[i+1]} = \operatorname*{argmin}_{\mathbf{u}} \frac{1}{2}\|\mathbf{u} - \mathbf{v}\|_2^2 + \tau\|\mathbf{u}\|_1, \tag{2.88}$$

which first uses the gradient of $\|\mathbf{x} - \mathbf{D}\mathbf{z}\|_2^2$ to update $\mathbf{z}^{[i]}$ in step size α and then finds the closest sparse solution using an ℓ_1 convex relaxation. It can be demonstrated that the second step has a closed-form solution that is

$$\mathbf{z}^{[i+1]} = \eta(\mathbf{v}; \tau). \tag{2.89}$$

where η represents a element-wise soft-thresholding operation, formally defined as

$$\eta(x; b) = \begin{cases} x - b & \text{if } x > b, \\ x + b & \text{if } x < -b, \\ 0 & \text{otherwise.} \end{cases} \tag{2.90}$$

Therefore, ISTA can be summarized as the following recursive equation:

$$\mathbf{z}^{[i+1]} = \eta\big(\mathbf{z}^{[i]} - \alpha \mathbf{D}^T(\mathbf{D}\mathbf{z}^{[i]} - \mathbf{x}); \tau\big), \tag{2.91}$$

where τ is related to λ for controlling sparsity.

Recently, Papyan [32] proposed to use ISTA and sparse coding to reinterpret feed-forward neural networks. They argue that feed-forward passing a single-layer neural network $\mathbf{z} = \eta(\mathbf{D}^T\mathbf{x}; b)$ can be considered as one iteration of ISTA in (2.91) when setting $\alpha = 1$ and $\tau = b$. Based on this insight, the authors extend this interpretation to feed-forward neural network with N layers

$$\begin{aligned} \mathbf{z}_1 &= \eta(\mathbf{D}_1^T\mathbf{x}; b_1) \\ \mathbf{z}_2 &= \eta(\mathbf{D}_2^T\mathbf{z}_1; b_2) \\ &\vdots \\ \mathbf{z}_N &= \eta(\mathbf{D}_N^T\mathbf{z}_{N-1}; b_N) \end{aligned} \tag{2.92}$$

as executing a sequence of single-iteration ISTA, serving as an approximate solution to the hierarchical sparse coding problem: find $\{\mathbf{z}_i\}_{i=1}^N$, such that

$$\begin{aligned} \mathbf{x} &= \mathbf{D}_1\mathbf{z}_1, \quad \|\mathbf{z}_1\|_0 < \lambda_1, \\ \mathbf{z}_1 &= \mathbf{D}_2\mathbf{z}_2, \quad \|\mathbf{z}_2\|_0 < \lambda_2, \\ &\vdots \qquad , \qquad \vdots \\ \mathbf{z}_{N-1} &= \mathbf{D}_N\mathbf{z}_N, \quad \|\mathbf{z}_N\|_0 < \lambda_N, \end{aligned} \tag{2.93}$$

where the bias terms $\{b_i\}_{i=1}^N$ (in a similar manner to τ) are related to $\{\lambda_i\}_{i=1}^N$, adjusting the sparsity of recovered code. Furthermore, they reinterpret back-propagating through the deep neural network as learning the dictionaries $\{\mathbf{D}_i\}_{i=1}^N$. This connection offers a novel reinterpretation of DNNs through the lens of hierarchical sparse dictionary learning. In this paper, we extend this reinterpretation to the block sparse scenario and apply it to solving our NRS*f*M problem.

Chapter 3

Block-sparse Non-Rigid Structure from Motion

One can see that much of the research on low-rank NRS*f*M draws heavily upon the fact that one can obtain a solution to the rank constrained factorization problem

$$\underset{\mathbf{\Pi},\mathbf{B}}{\operatorname{argmin}} ||\mathbf{W} - \mathbf{\Pi}\mathbf{B}||_F^2, \quad \text{s.t. } \operatorname{rank}(\mathbf{\Pi}) = 3K \tag{3.1}$$

through a Singular Value Decomposition (SVD). Even though the SVD returns a unique solution $\{\hat{\mathbf{\Pi}}, \hat{\mathbf{B}}\}$ it is easy to demonstrate that this solution is just one of many possible solutions to $\mathbf{W} = \hat{\mathbf{\Pi}}\hat{\mathbf{B}} = \hat{\mathbf{\Pi}}\mathbf{G}\mathbf{G}^{-1}\hat{\mathbf{B}} = \mathbf{\Pi}\mathbf{B}$, where the corrective matrix $\mathbf{G}$ is any non-singular matrix. The ambiguity of this factorization is problematic for NRS*f*M problems, as additional constraints are required to obtain a unique solution.

For rigid NRS*f*M (i.e.$K = 1$), the application of camera constraints [38] is typically sufficient in order to find a correction matrix $\mathbf{G}$ that gives a unique solution. Xiao et al. [47] famously demonstrated for $K > 1$ that one cannot determine a unique $\mathbf{G}$ since the space of solutions lies in a nullspace of rank $2K^2 - K$. Akhter et al. [4] additionally demonstrated that even though $\mathbf{G}$ is not unique, any solution to $\mathbf{G}$ that satisfies the camera constraints returns a valid 3D shape and camera motion pair. This chapter will explore whether moving away from canonical rank constraints and instead assuming that $\mathbf{\Pi}$ is block-sparse could result in a far less ambiguous factorization, thus resulting in an NRS*f*M algorithm that can circumvent current theoretical and practical limitations.

3.1 Uniqueness of Block Sparse Dictionary Learning

3.1.1 Uniqueness of Sparse Dictionary Learning

The uniqueness of the Sparse Dictionary Learning (SDL) is explored in literature [25]. In general terms, the problem of SDL can be described as

$$\underset{\mathbf{D},\mathbf{Z}}{\operatorname{argmin}} \|\mathbf{X} - \mathbf{D}\mathbf{Z}\|_F^2 \quad \text{s.t. } \|\mathbf{z}_i\|_0 = K, \quad i = 1, ..., N \tag{3.2}$$

where we are trying to recover the concatenation of a sparse coefficient matrix $\mathbf{Z}$ and the dictionary basis $\mathbf{D}$ from a known set of signals in $\mathbf{X} \in \mathbb{R}^{D \times N}$. Specifically, the sparse coefficient matrix is the concatenation of $K-$sparse coefficient vectors $\mathbf{Z} = \Big[\mathbf{z}_1, \ldots, \mathbf{z}_D\Big]$, and concatenation of $\mathbf{D} = \Big[\mathbf{d}_1, \ldots, \mathbf{d}_M\Big]$ dictionary basis vectors. An important question to ask in the context of applying SDL to NRS*f*M is: how unique is the solution to Equation 3.2?

Hillar et al. [25] recently characterized the theoretical answer to this question. The authors define that if any valid solution $\{\hat{\mathbf{D}}, \hat{\mathbf{Z}}\}$ to the SDL objective in Equation 3.2 is ambiguous up to a $M \times M$ permutation matrix $\mathbf{P}$ and a diagonal invertible weighting matrix $\mathbf{\Lambda}$ such that $\hat{\mathbf{D}} = \mathbf{D}\mathbf{P}\mathbf{\Lambda}$, and $\hat{\mathbf{Z}} = \mathbf{\Lambda}^{-1}\mathbf{P}^T\mathbf{Z}$, then $\mathbf{X}$ has a *unique SDL*. Moreover, they prove theoretically that, given large enough N, the uniqueness of SDL is achieved if and only if the dictionary $\mathbf{D}$ satisfies the spark condition[1]:

$$\mathbf{D}\mathbf{z}_1 = \mathbf{D}\mathbf{z}_2 \quad \text{for } K\text{-sparse } \mathbf{z}_1, \mathbf{z}_2 \in \mathbb{R}^M \Rightarrow \mathbf{z}_1 = \mathbf{z}_2. \tag{3.3}$$

Coherence as a proxy

The spark condition provides a complete characterization of the uniqueness of SDL. However, verifying whether a matrix $\mathbf{D}$ satisfies the spark condition is an NP-hard problem, which has to visit all $\binom{M}{K}$ subspaces. It is preferable in practice to use properties of $\mathbf{D}$ that are easily computable, such as mutual coherence, which measures the largest absolute inner product between any two column vectors in the matrix, and with high probability is indicative of the spark condition of the matrix. The experimental portion of this chapter demonstrates how the coherence of a matrix can be utilized to predict the reconstructibility of a 3D structure solely from its 2D projections.

[1] Refer to [25] for the proof and a lower bound of N

3.1.2 Block Sparse Dictionary Learning and Uniqueness

As discussed in the next section, there is a strong connection between compressible NRSfM and Block Sparse Dictionary Learning (BSDL). BSDL is a generalization of the SDL objective in Equation 3.2:

$$\underset{\mathbf{D},\mathbf{Z}}{\operatorname{argmin}} ||\mathbf{X} - \mathbf{DZ}||_F^2 \quad \text{s.t. } ||\mathbf{Z}_i||_{0,\alpha} = K, \quad i = 1, ..., N/\beta, \tag{3.4}$$

where $\mathbf{Z}_i \in \mathbb{R}^{D\times\beta}$ is a submatrix of $\mathbf{Z}$, i.e.$\mathbf{Z} = \Big[\mathbf{Z}_1, ..., \mathbf{Z}_{N/\beta}\Big]$. Each $\mathbf{Z}_i$ is divided into M/α blocks of size $\alpha \times \beta$ and $\|\mathbf{Z}_i\|_{0,\alpha}$ counts the number of blocks, of which at least one element is non-zero. α and β need to be chosen such that D and M are perfectly divisible. Of particular importance in our compressible NRSfM problem is 3×2 block-sparsity, which we will describe in more detail in the next section on compressible NRSfM.

Definition 1 *If any valid solution $\{\hat{\mathbf{D}}, \hat{\mathbf{Z}}\}$ to the objective in Equation 3.4 is ambiguous only up to a $M \times M$ block permutation matrix $\mathbf{P}_\alpha$ and a block-diagonal invertible weighting matrix $\mathbf{\Lambda}_\alpha$ such that*

$$\hat{\mathbf{D}} = \mathbf{D}\mathbf{P}_\alpha\mathbf{\Lambda}_\alpha, \quad \hat{\mathbf{Z}} = \mathbf{\Lambda}_\alpha^{-1}\mathbf{P}_\alpha^T\mathbf{Z}, \tag{3.5}$$

we say $\mathbf{X}$ has a unique BSDL.

The block permutation matrix is actually defined as $\mathbf{P}_\alpha = \mathbf{P}\otimes\mathbf{I}_\alpha$ where $\mathbf{P}$ is an arbitrary $(M/\alpha)\times(M/\alpha)$ permutation matrix and $\mathbf{I}_\alpha$ is a $\alpha \times \alpha$ identity matrix. The block-diagonal invertible weighting matrix $\mathbf{\Lambda}_\alpha$ has a $\alpha \times \alpha$ block structure. We now ask the same question: what is the sufficient and necessary condition for the uniqueness of BSDL?

Theorem 1 *There exist $K\binom{M/\alpha}{K}^2$ K-block-sparse vectors $\mathbf{Z}_1, ..., \mathbf{Z}_{N/\beta}$, i.e.$N = \beta K\binom{M/\alpha}{K}^2$, such that the uniqueness of BSDL holds if and only if the matrix $\mathbf{D}$ satisfies the block spark condition:*

$$\mathbf{DZ}_1 = \mathbf{DZ}_2 \quad \textit{for } K\textit{-block-sparse } \mathbf{Z}_1, \mathbf{Z}_2 \in \mathbb{R}^{M\times\beta} \Rightarrow \mathbf{Z}_1 = \mathbf{Z}_2. \tag{3.6}$$

3.1.3 Proof

Let's first prove Theorem 1 in the case when $\beta = 1$ and once it is proven, the general case $\beta > 1$ is simple to handle. We can split sparse causes $\mathbf{Z}^i$ into $[\mathbf{z}_1^i, \cdots, \mathbf{z}_\beta^i]$, where $\mathbf{z}_j^i \in \mathbb{R}^{D\times 1}$ and then

$$\mathbf{DZ}^i = \mathbf{D}[\mathbf{z}_1^i, \cdots, \mathbf{z}_\beta^i] = \hat{\mathbf{D}}\hat{\mathbf{Z}}^i = \hat{\mathbf{D}}[\hat{\mathbf{z}}_1^i, \cdots, \hat{\mathbf{z}}_\beta^i] \tag{3.7}$$

is equivalent to $\mathbf{D}\mathbf{z}_j^i = \hat{\mathbf{D}}\mathbf{z}_j^i$, which degenerates to the situation where $\beta = 1$.

A simple case when $K = 1$

To better understand Theorem 1 and prepare for the proof in full generality, let us start from a simple case when $K = 1$. Denote $\mathbf{e}_i^L$ as a L-dimensional column vector that has one in its i-th coordinate and zeros elsewhere. For convenience, let $L = M/\alpha$. Now let us produce M block vectors

$$\mathbf{z}_j^i = (\mathbf{e}_i^L \otimes \mathbf{e}_j^\alpha), \quad i = 1, \cdots, L, \quad j = 1, \cdots, \alpha, \tag{3.8}$$

which denotes that its j-th coordinate in i-th block is one and zeros elsewhere, and $L\binom{\alpha}{2}$ block vectors $\mathbf{z}_{jk}^i = \mathbf{z}_{jk}^i + \mathbf{z}_{jk}^i$, for any i and $j \neq k$.

Now we claim that the uniqueness of BSDL in this simple case can be achieved by these $M + L\binom{\alpha}{2}$ block vectors, which is less than $K\binom{M/\alpha}{K}^2$, assuming $M \gg \alpha$.

Proof: There exists a matrix $\hat{\mathbf{D}}$ and 1-block-sparse vector $\hat{\mathbf{z}}_j^i = (\mathbf{e}_{\pi(i,j)}^L \otimes \mathbf{I}_\alpha)\boldsymbol{\lambda}_{ij}$, for some mapping $\pi : \{1, ..., L\} \times \{1, ..., \alpha\} \to \{1, ..., L\}$ and $\boldsymbol{\lambda}_{ij} \in \mathbb{R}^\alpha$, such that

$$\mathbf{D}\mathbf{z}_j^i = \mathbf{D}(\mathbf{e}_i^L \otimes \mathbf{e}_j^\alpha) = \hat{\mathbf{D}}\hat{\mathbf{z}}_j^i = \hat{\mathbf{D}}(\mathbf{e}_{\pi(i,j)}^L \otimes \mathbf{I}_\alpha)\boldsymbol{\lambda}_{ij}, \tag{3.9}$$

We claim that $\pi(i, j)$ is only dependent on i, not j. From Equation 3.9, we know that for any $j \neq k$,

$$\mathbf{D}\mathbf{z}_{jk}^i = \mathbf{D}(\mathbf{z}_j^i + \mathbf{z}_k^i) = \mathbf{D}\mathbf{z}_j^i + \mathbf{D}\mathbf{z}_k^i = \hat{\mathbf{D}}\hat{\mathbf{z}}_j^i + \hat{\mathbf{D}}\hat{\mathbf{z}}_k^i = \hat{\mathbf{D}}(\hat{\mathbf{z}}_j^i + \hat{\mathbf{z}}_k^i). \tag{3.10}$$

Since $\mathbf{z}_{jk}^i$ is 1-block-sparse, this implies that $\hat{\mathbf{z}}_j^i + \hat{\mathbf{z}}_k^i$ should also be 1-block-sparse. Therefore, $\pi(i, j) = \pi(i, k)$, that is, $\pi : \{1, ..., L\} \to \{1, ..., L\}$ and

$$\mathbf{D}(\mathbf{e}_i^L \otimes \mathbf{e}_j^\alpha) = \hat{\mathbf{D}}(\mathbf{e}_{\pi(i)}^L \otimes \mathbf{I}_\alpha)\boldsymbol{\lambda}_{ij}. \tag{3.11}$$

Let us now prove that $\boldsymbol{\Lambda}_i = [\boldsymbol{\lambda}_{i1}, \ldots, \boldsymbol{\lambda}_{i\alpha}]$ is invertible. Let $\mathbf{Z}^i = [\mathbf{z}_1^i, \ldots, \mathbf{z}_\alpha^i]$ and $\hat{\mathbf{Z}}^i = [\hat{\mathbf{z}}_1^i, \ldots, \hat{\mathbf{z}}_\alpha^i]$. From Equation 3.11, it follows that

$$\mathbf{D}\mathbf{Z}^i = \mathbf{D}[\mathbf{z}_1^i, \ldots, \mathbf{z}_\alpha^i] = \mathbf{D}[(\mathbf{e}_i^L \otimes \mathbf{e}_1^\alpha), ..., (\mathbf{e}_i^L \otimes \mathbf{e}_\alpha^\alpha)] = \mathbf{D}(\mathbf{e}_i^L \otimes \mathbf{I}_\alpha) \tag{3.12}$$

and

$$\mathbf{D}\mathbf{Z}^i = \hat{\mathbf{D}}\hat{\mathbf{Z}}^i = \hat{\mathbf{D}}(\mathbf{e}_{\pi(i)}^L \otimes \mathbf{I}_\alpha)\left[\boldsymbol{\lambda}_{i1}, ..., \boldsymbol{\lambda}_{i\alpha}\right] = \hat{\mathbf{D}}(\mathbf{e}_{\pi(i)}^L \otimes \mathbf{I}_\alpha)\boldsymbol{\Lambda}_i. \tag{3.13}$$

Therefore,

$$\mathbf{D}(\mathbf{e}_i^L \otimes \mathbf{I}_\alpha) = \hat{\mathbf{D}}(\mathbf{e}_{\pi(i)}^L \otimes \mathbf{I}_\alpha)\boldsymbol{\Lambda}_i. \tag{3.14}$$

Due to the fact that $\mathbf{D}$ satisfies the block spark condition, $\operatorname{rank}(\mathbf{D}(\mathbf{e}_i^L \otimes \mathbf{I}_\alpha)) = \alpha$. From Equation 3.14, $\operatorname{rank}(\hat{\mathbf{D}}(\mathbf{e}_{\pi(i)}^L \otimes \mathbf{I}_\alpha)\boldsymbol{\Lambda}_i) = \alpha$. We know that $\operatorname{rank}(\mathbf{X}\mathbf{Y}) \leq \min(\operatorname{rank}(\mathbf{X}), \operatorname{rank}(\mathbf{Y}))$, for any matrix $\mathbf{X}, \mathbf{Y}$. So $\operatorname{rank}(\boldsymbol{\Lambda}_i) \geq \alpha$. As $\boldsymbol{\Lambda}_i \in \mathbb{R}^{\alpha\times\alpha}$, $\operatorname{rank}(\boldsymbol{\Lambda}_i) = \alpha$.

Now, let us show π is necessarily injective. Suppose $\pi(i) = \pi(j)$, with $i \neq j$, then from Equation 3.14,

$$\mathbf{D}(\mathbf{e}_i^L \otimes \mathbf{I}_\alpha) = \hat{\mathbf{D}}(\mathbf{e}_{\pi(i)}^L \otimes \mathbf{I}_\alpha)\boldsymbol{\Lambda}_i = \hat{\mathbf{D}}(\mathbf{e}_{\pi(j)}^L \otimes \mathbf{I}_\alpha)\boldsymbol{\Lambda}_j\boldsymbol{\Lambda}_j^{-1}\boldsymbol{\Lambda}_i = \mathbf{D}(\mathbf{e}_j^L \otimes \mathbf{I}_\alpha)\boldsymbol{\Lambda}_j^{-1}\boldsymbol{\Lambda}_i. \qquad (3.15)$$

Since $\mathbf{D}$ satisfies the block spark condition, which implies $\mathbf{D}$ can never map two different 1-block-sparse vectors to the same measurement, this is possible only if $i = j$. Thus, π is injective.

Let $\mathbf{P}_\pi$ and $\mathbf{D}$ be generated by

$$\mathbf{P}_\pi = \begin{bmatrix} \mathbf{e}_{\pi(1)}^L & \dots & \mathbf{e}_{\pi(K)}^L \end{bmatrix}, \boldsymbol{\Lambda} = \begin{bmatrix} \boldsymbol{\Lambda}_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \boldsymbol{\Lambda}_L \end{bmatrix}. \qquad (3.16)$$

Since π is injective, $\mathbf{P}_\pi$ is a permutation matrix. Let us stack Equation 3.14 from left-to-right on both sides, and it follows that on left sides,

$$[\mathbf{D}(\mathbf{e}_1^L \otimes \mathbf{I}_\alpha), \dots, \mathbf{D}(\mathbf{e}_L^L \otimes \mathbf{I}_\alpha)] = \mathbf{D}, \qquad (3.17)$$

and on right sides,

$$[\hat{\mathbf{D}}(\mathbf{e}_{\pi(1)}^L \otimes \mathbf{I}_\alpha)\boldsymbol{\Lambda}_1, \dots, \hat{\mathbf{D}}(\mathbf{e}_{\pi(L)}^L \otimes \mathbf{I}_\alpha)\boldsymbol{\Lambda}_L] = \hat{\mathbf{D}}(\mathbf{P}_\pi \otimes \mathbf{I}_\alpha)\boldsymbol{\Lambda}. \qquad (3.18)$$

Hence, we proved Theorem 1 for the simple case, where $K = 1$. ■

Preparation

We use the same notation reported in [25]: Denote $[L]$ as the set $\{1, \dots, L\}$ and $\binom{[L]}{K}$ as the K-element subset of $[L]$. Moreover, let the dictionary $\mathbf{D} = [\mathbf{D}_1, \dots, \mathbf{D}_L]$ with $\mathbf{D}_i \in \mathbb{R}^{D\times\alpha}$, and denote $\mathrm{span}\{\mathbf{D}_\mathcal{S}\}$ as a subspace expanded by $\mathbf{D}_i$, $i \in \mathcal{S}$.

To prove Theorem 1 in general situations, we offer a lemma at first.

Lemma 1 *Suppose that* $\mathbf{D}$ *satisfies the block spark condition and*

$$\kappa : \binom{[L]}{K} \to \binom{[L]}{K} \qquad (3.19)$$

is a mapping with the following property: for all $\mathcal{S} \in \binom{[L]}{K}$,

$$\mathrm{span}\{\mathbf{D}_\mathcal{S}\} = \mathrm{span}\{\hat{\mathbf{D}}_{\kappa(\mathcal{S})}\}. \qquad (3.20)$$

Then, there exists a permutation matrix $\mathbf{P}_\kappa \in \mathbb{R}^{L\times L}$ *and an invertible block diagonal matrix* $\boldsymbol{\Lambda} \in \mathbb{R}^{M\times M}$ *such that* $\mathbf{D} = \hat{\mathbf{D}}(\mathbf{P}_\kappa \otimes \mathbf{I}_\alpha)\boldsymbol{\Lambda}$.

Proof: Here we demonstrate, through induction, that if our $K = 1$ case holds, then the $K > 1$ case should also hold. First, let us show function κ is injective. Suppose that $\mathcal{S}, \mathcal{S}' \in \binom{[L]}{K}$ are different and $\kappa(\mathcal{S}) = \kappa(\mathcal{S}')$ holds. Then by Equation 3.20,

$$\text{span}\{\mathbf{D}_{\mathcal{S}}\} = \text{span}\{\hat{\mathbf{D}}_{\kappa(\mathcal{S})}\} = \text{span}\{\hat{\mathbf{D}}_{\kappa(\mathcal{S}')}\} = \text{span}\{\mathbf{D}_{\mathcal{S}'}\}. \tag{3.21}$$

As $\mathbf{D}$ satisfies the block spark condition, all $K + 1$ block columns of $\mathbf{D}$ are linearly independent. From Lemma 2 (see below), it turns out that $\mathcal{S} = \mathcal{S}'$, which implies κ is injective.

Denote $\eta = \kappa^{-1}$ as the inverse of κ. Fix $\mathcal{S} = \{i_1, ..., i_{K-1}\} \in \binom{[L]}{K-1}$, and set $\mathcal{S}_1 = \mathcal{S} \cup \{p\}$ and $\mathcal{S}_2 = \mathcal{S} \cup \{q\}$ for some fixed $p, q \notin \mathcal{S}$ with $p \neq q$. Since $K < L$, $L - (K - 1) > 1$, thus, it is always possible to find such p and q. From Equation 3.20, we obtain:

$$\text{span}\{\mathbf{D}_{\eta(\mathcal{S}_1)}\} = \text{span}\{\hat{\mathbf{D}}_{\mathcal{S}_1}\}, \tag{3.22}$$

$$\text{span}\{\mathbf{D}_{\eta(\mathcal{S}_2)}\} = \text{span}\{\hat{\mathbf{D}}_{\mathcal{S}_2}\}. \tag{3.23}$$

Let us intersect Equation 3.22 and Equation 3.23, and from Lemma 3 (see below), it follows that

$$\text{span}\{\hat{\mathbf{D}}_{\mathcal{S}_1}\} \cap \text{span}\{\hat{\mathbf{D}}_{\mathcal{S}_2}\} = \text{span}\{\mathbf{D}_{\eta(\mathcal{S}_1) \cap \eta(\mathcal{S}_2)}\}. \tag{3.24}$$

Since $\text{span}\{\hat{\mathbf{D}}_{\mathcal{S}}\} \subseteq \text{span}\{\hat{\mathbf{D}}_{\mathcal{S}_1}\} \cap \text{span}\{\hat{\mathbf{D}}_{\mathcal{S}_2}\}$, it follows that $\text{span}\{\hat{\mathbf{D}}_{\mathcal{S}}\} \subseteq \text{span}\{\mathbf{D}_{\eta(\mathcal{S}_1) \cap \eta(\mathcal{S}_2)}\}$. The number of the elements in $\eta(\mathcal{S}_1) \cap \eta(\mathcal{S}_2)$ is $K - 1$, since $\eta(p) \neq \eta(q)$, with $p \neq q$, by injectivity of η. Moreover, the number of the elements in $\mathcal{S}$ is also $K - 1$, which implies that

$$\text{span}\{\hat{\mathbf{D}}_{\mathcal{S}}\} = \text{span}\{\mathbf{D}_{\eta(\mathcal{S}_1) \cap \eta(\mathcal{S}_2)}\}. \tag{3.25}$$

The association $\mathcal{S} \to \eta(\mathcal{S}_1) \cap \eta(\mathcal{S}_2)$ from Equation 3.25 defines a function $\sigma : \binom{[L]}{K-1} \to \binom{[L]}{K-1}$, with property that $\text{span}\{\hat{\mathbf{D}}_{\mathcal{S}}\} = \text{span}\{\mathbf{D}_{\sigma(\mathcal{S})}\}$.

Finally, let's show that σ is injective. Suppose $\mathcal{S}, \mathcal{S}' \in \binom{[L]}{K-1}$, and $\sigma(\mathcal{S}) = \sigma(\mathcal{S}')$, it follows that

$$\text{span}\{\hat{\mathbf{D}}_{\mathcal{S}}\} = \text{span}\{\mathbf{D}_{\sigma(\mathcal{S})}\} = \text{span}\{\mathbf{D}_{\sigma(\mathcal{S}')}\} = \text{span}\{\hat{\mathbf{D}}_{\mathcal{S}'}\}. \tag{3.26}$$

As every K block columns of $\mathbf{D}$ are linear independent, and κ is injective, every K block columns of $\hat{\mathbf{D}}$ are also linear independent. From Lemma 2, it follows that $\mathcal{S} = \mathcal{S}'$, which implies σ is injective. Hence, let $\xi = \sigma^{-1}$, with properties: for all $\mathcal{S} \in \binom{[L]}{K-1}$, $\text{span}\{\mathbf{D}_{\mathcal{S}}\} = \text{span}\{\hat{\mathbf{D}}_{\xi(\mathcal{S})}\}$. ■

Lemma 2 *If any set of $K+1$ block columns of matrix $\mathbf{D} = [\mathbf{D}_1, \ldots, \mathbf{D}_L]$ are linear independent, then for $\mathcal{S}, \mathcal{S}' \in \binom{[L]}{K}$,*

$$\text{span}\{\mathbf{D}_{\mathcal{S}}\} = \text{span}\{\mathbf{D}_{\mathcal{S}'}\} \quad \Rightarrow \quad \mathcal{S} = \mathcal{S}'. \tag{3.27}$$

Proof: Suppose that $\mathcal{S} \neq \mathcal{S}' \in \binom{[L]}{K}$ satisfying $\mathrm{span}\{\mathbf{D}_{\mathcal{S}}\} = \mathrm{span}\{\mathbf{D}_{\mathcal{S}'}\}$. Then without loss of generality, there is an $i \in \mathcal{S}$ with $i \notin \mathcal{S}'$, but atoms $\mathbf{D}_i \in \mathrm{span}\{\mathbf{D}_{\mathcal{S}'}\}$, which implies that the $K+1$ block columns indexed by $\mathcal{S}' \cup \{i\}$ are not linear-independent, a contradiction to the assumption. ■

Lemma 3 *If matrix* $\mathbf{D}$ *satisfies the block spark condition, then for* $\mathcal{S}, \mathcal{S}' \in \binom{[L]}{K}$,

$$\mathrm{span}\{\mathbf{D}_{\mathcal{S}\cap\mathcal{S}'}\} = \mathrm{span}\{\mathbf{D}_{\mathcal{S}}\} \cap \mathrm{span}\{\mathbf{D}_{\mathcal{S}'}\}. \tag{3.28}$$

Proof: The inclusion "$\subseteq$" is trivial, so let us prove "$\supseteq$". Suppose a block vector $\mathbf{x} \in \mathrm{span}\{\mathbf{D}_{\mathcal{S}}\} \cap \mathrm{span}\{\mathbf{D}_{\mathcal{S}_2}\}$. Express $\mathbf{x}$ as a linear combination of K atoms of $\mathbf{D}$ indexed by $\mathbf{S}$ and, separately, as a combination of K atoms of $\mathbf{D}$ indexed by $\mathcal{S}'$. By the block spark condition, these linear combinations must be identical. In particular, $\mathbf{x}$ was expressed as a linear combination of atoms of $\mathbf{D}$ indexed by $\mathcal{S} \cap \mathcal{S}'$, and thus is in $\mathrm{span}\{\mathbf{D}_{\mathcal{S}\cap\mathcal{S}'}\}$ ■

Proof of Theorem 1 when $\beta = 1$

First, we produce a set of $N = K\binom{M/\alpha}{K}^2$ vectors $\mathbf{s}_i \in \mathbb{R}^{\alpha K}$ in general linear position (i.e.any subset of K of them are linearly independent). One possible strategy is to produce a "Vandermonde" matrix [41]. Next, we form K-block-sparse vectors $\mathbf{z}_1, ..., \mathbf{z}_N$ by taking $\mathbf{s}_i$ for the support value of $\mathbf{z}_i$ where each possible support set is represented $K\binom{M/\alpha}{K}$ times. We claim that these $\mathbf{z}_i$ always guarantee the uniqueness of BSDL.

Proof: Suppose there exists an alternate dictionary $\hat{\mathbf{D}}$ and a set of K-block-sparse vectors $\hat{\mathbf{z}}_1, ..., \hat{\mathbf{z}}_N$ such that $\mathbf{D}\mathbf{z}_i = \mathbf{x}_i = \hat{\mathbf{D}}\hat{\mathbf{z}}_i$. As there are $K\binom{M/\alpha}{K}$ $\mathbf{x}_i$ for each support indexed by $\mathcal{S}$, the "pigeon-hole principle"[2] implies that there are at least K vectors $\hat{\mathbf{z}}_{i_1}, ..., \hat{\mathbf{z}}_{i_K}$ using the same support $\mathcal{S}'$. Thus, $\mathrm{span}\{\mathbf{x}_{i_1}, ..., \mathbf{x}_{i_K}\} \subseteq \mathrm{span}\{\hat{\mathbf{D}}_{\mathcal{S}'}\}$. By the general linear position and the block spark condition, $\mathrm{span}\{\mathbf{x}_{i_1}, ..., \mathbf{x}_{i_K}\} = \mathrm{span}\{\mathbf{D}_{\mathcal{S}}\}$. Therefore, $\mathrm{span}\{\mathbf{D}_{\mathcal{S}}\} \subseteq \mathrm{span}\{\hat{\mathbf{D}}_{\mathcal{S}'}\}$. As the dimension of $\mathrm{span}\{\hat{\mathbf{D}}_{\mathcal{S}'}\}$ is less and equal to K, $\mathrm{span}\{\mathbf{D}_{\mathcal{S}}\} = \mathrm{span}\{\hat{\mathbf{D}}_{\mathcal{S}'}\}$.

By Lemma 1, Theorem 1 is proved. ■

[2] The pigeon-hole principle states that if n items are put into m containers, with $n > m$, then at least one container must contain more than one item [11].

3.2 Modeling via Block Sparsity

Let us assume that the unknown 3D structures $\mathbf{S}^\sharp$ are compressible, that is, the 3D structure in each frame (each row of $\mathbf{S}^\sharp$) can be approximated by only K basis shapes (K rows of $\mathbf{B}^\sharp$.) Therefore, the factorization $\mathbf{S}^\sharp = \mathbf{C}^\sharp \mathbf{B}^\sharp$ results in a set of coefficients $\mathbf{C}^\sharp \in \mathbb{R}^{F \times L}$ whose rows are each K-sparse.

$$\mathbf{S}^\sharp = \mathbf{C}^\sharp \mathbf{B}^\sharp, \quad \text{s.t. } \|\mathbf{C}^\sharp_i\|_0 < K, \tag{3.29}$$

where $\|\cdot\|_0$ counts the number of active elements of argument vector/matrix and $\mathbf{C}^\sharp_i$ is the i-th row of $\mathbf{C}^\sharp$. Note that one never has access to the 3D structure $\mathbf{S}^\sharp$ a priori, only to the 2D projections $\mathbf{W}$. Interestingly, however, if we know $\mathbf{S}^\sharp$ is compressible, then from the projection equation (i.e.$\mathbf{\Pi} = \mathbf{M}(\mathbf{C}^\sharp \otimes \mathbf{I}_3)$), $\mathbf{\Pi}$ *must* be 2×3 block sparse as the camera matrix $\mathbf{M}$ is 2×3 block-diagonal. It is this insight that forms the crucial component of our algorithm. From a known measurement matrix $\mathbf{W}$ and desired K, L, one can factorize $\mathbf{W}^T$ through a 3×2 block sparse dictionary learning process. Note: for NRS*f*M $\mathbf{W} = \mathbf{\Pi}\mathbf{B}$, whereas for BSDL this would be expressed as $\mathbf{W}^T = \mathbf{B}^T\mathbf{\Pi}^T$ where $\mathbf{X} = \mathbf{W}^T, \mathbf{D} = \mathbf{B}^T$, and $\mathbf{Z} = \mathbf{\Pi}^T$.

Theorem 2 *If one can recover $\hat{\mathbf{B}}$ using a 3×2 BSDL such that $\mathbf{D} = \hat{\mathbf{B}}^T$ satisfies the block spark condition, then it can be shown that the transpose of $\hat{\mathbf{B}}^\sharp$ satisfies the canonical spark condition, where $\hat{\mathbf{B}}^\sharp$ is an $L \times 3P$ reshape of $\hat{\mathbf{B}}$. Further, for such BSDL to be unique, K must be less than or equal to $P/3 - 1$.*

Proof: Suppose two K-sparse vectors $\mathbf{z}_1$ and $\mathbf{z}_2$ such that $(\hat{\mathbf{B}}^\sharp)^T\mathbf{z}_1 = (\hat{\mathbf{B}}^\sharp)^T\mathbf{z}_2$. Then from the reshape, it follows that $\hat{\mathbf{B}}^T(\mathbf{z}_1 \otimes \mathbf{I}_3) = \hat{\mathbf{B}}^T(\mathbf{z}_2 \otimes \mathbf{I}_3)$. As $\hat{\mathbf{B}}^T$ satisfies the block spark condition, it follows that $\mathbf{z}_1 = \mathbf{z}_2$; therefore, $(\mathbf{B}^\sharp)^T$ satisfies the canonical spark condition. Further, the uniqueness of the BSDL factorization requires $\hat{\mathbf{B}}^T$ to satisfy the block spark condition. This implies that any $P \times 3(K+1)$ submatrices generated by concatenating $K+1$ block columns of $\hat{\mathbf{B}}^T$ need to be full column rank. Consider, a counterexample for contradiction: if $K = 2$, and $\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3$ are 3 linear dependent block columns of $\hat{\mathbf{B}}^T$. In addition, suppose any 2 of them are linear independent. Then the subspace spanned by $\{\mathbf{b}_1, \mathbf{b}_2\}$ is identical to one by $\{\mathbf{b}_1, \mathbf{b}_3\}$, which breaks the block spark condition. Therefore K needs to be less than or equal to $P/3 - 1$. ■

Theorem 2 actually tells us that the uniqueness of the BSDL factorization on 2D projections automatically guarantees the uniqueness of the SDL factorization on the unknown 3D structures. Interestingly, the converse is not always true. This result highlights a drawback in our proposed approach; that is, we cannot recover all compressible structures but the subsets where $\hat{\mathbf{\Pi}}$ is sufficiently sparse ($K \leq P/3 - 1$) and $\hat{\mathbf{B}}$ satisfies the block spark condition. In the experiments section, we show a strategy that can be utilized in practice to improve the incoherence of $\hat{\mathbf{B}}$ and push it to satisfy the block spark condition.

3.3 Solving via Block Sparse Dictionary Learning

In this section, we describe our BSDL algorithm that adapts K-SVD [35], OMP [40] and FOCUSS [21] to the block sparse situation respectively. However, any valid BSDL method can be employed here as long as it returns a valid factorization $\mathbf{W} = \hat{\mathbf{\Pi}}\hat{\mathbf{B}}$.

3.3.1 Block K-SVD

The objective function that block K-SVD aims to solve is

$$\begin{aligned} &\min_{\mathbf{D},\mathbf{X}} \|\mathbf{Y} - \mathbf{DX}\|_F^2 \\ &\text{s.t. } \|\mathbf{X}_i\|_{b0} \leq T_0, \text{for} i = 1, 2, ..., N. \end{aligned} \tag{3.30}$$

where $\|\cdot\|_{b0}$ counts the number of active blocks in the argument matrix.

Let's first consider the block sparse coding stage, i.e.given $\mathbf{D}$ is fixed, and find the sparse coefficient summarized in matrix $\mathbf{X}$. By the definition of the Frobenius norm, the objective function can be written as

$$\|\mathbf{Y} - \mathbf{DX}\|_F^2 = \sum_{i=1}^{N} \|\mathbf{Y}_i - \mathbf{DX}_i\|_F^2. \tag{3.31}$$

Therefore, the problem posed in (3.30) can be decoupled to N distinct problems of the form

$$\begin{aligned} &\min_{\mathbf{X}_i} \|\mathbf{Y}_i - \mathbf{DX}_i\|_F^2 \\ &\text{s.t. } \|\mathbf{X}_i\|_{b0} \leq T_0, \quad \text{for } i = 1, 2, \ldots, N. \end{aligned} \tag{3.32}$$

Each problem is adequately addressed by the pursuit algorithms, such as block-OMP.

We now turn to the second, and slightly more involved, process of updating the dictionary together with the nonzero coefficients. Denote $\mathbf{D}_k$ as the k-th atom in the dictionary $\mathbf{D}$, $\mathbf{X}_T^k$ as the k-th block row in $\mathbf{X}$, and $\mathbf{X}_k$ as the k-th block column in $\mathbf{X}$. Returning to the objective function (3.30), it can be rewritten as

$$\begin{aligned} \|\mathbf{Y} - \mathbf{DX}\|_F^2 &= \left\|\mathbf{Y} - \sum_{j=1}^{K} \mathbf{D}_j \mathbf{X}_T^j\right\|_F^2 \\ &= \left\|\left(\mathbf{Y} - \sum_{j\neq k} \mathbf{D}_j \mathbf{X}_T^j\right) - \mathbf{D}_k \mathbf{X}_T^k\right\|_F^2 \\ &= \left\|\mathbf{E}_k - \mathbf{D}_k \mathbf{X}_T^k\right\|_F^2 \end{aligned} \tag{3.33}$$

We have decomposed the multiplication $\mathbf{DX}$ into the sum of K low-rank matrices whose rank depends on the size of block defined in problem (3.30). Among those, $K-1$ matrices are fixed and only one–k-th matrix–remains in question. The matrix $\mathbf{E}_k$ stands for the modelling error for all N samples when the k-th atom is removed. Here it would be tempting to suggest the use of SVD to find alternative $\mathbf{D}_k$ and $\mathbf{X}_T^k$. The SVD finds the closest low-rank matrix (in the Frobenius norm) that approximates $\mathbf{E}_k$ and this will effectively minimize the error defined in 3.33. However, such a step would be a mistake, because the new block vector $\mathbf{X}_T^k$ is very likely to be filed, since in such an update of $\mathbf{D}_k$ we do not force the sparsity constraint. A remedy to the above problem, however, is simple and also quite intuitive. Define ω_k as the set of indices pointing to the active block in $\mathbf{X}_T^k$ i.e.Thus,

$$\omega_k = \left\{ i | 1 \leq i \leq K, \mathbf{X}_T^k(i) \text{ is active} \right\}, \tag{3.34}$$

where $\mathbf{X}_T^k(i)$ denotes the i-th block in $\mathbf{X}_T^k(i)$ denotes the size of block as $p \times q$ and $|\omega_k|$ as the number of active blocks. We then define $\mathbf{\Omega}_k$ as a matrix of size $Nq \times q|\omega_k|$ extracting the active blocks from $\mathbf{X}_T^k$ i.e.

$$\mathbf{X}_R^k = \mathbf{X}_T^k \mathbf{\Omega}_k. \tag{3.35}$$

This shrinks the block row vector $\mathbf{X}_T^k$ by discarding the zero blocks, resulting with the row vector $\mathbf{X}_R^k$ of length $q|\omega_k|$. Similarly, the multiplication

$$\mathbf{Y}_k^R = \mathbf{Y} \mathbf{\Omega}_k \tag{3.36}$$

creates a matrix that includes a subset of the examples that are currently using the $\mathbf{D}_k$ atom. The same effect happens with $\mathbf{E}_k^R = \mathbf{E}_k \mathbf{\Omega}_k$, implying a selection of error columns that correspond to examples that use the atom $\mathbf{D}_k$.

With this notation, we now return to (3.33), and suggest minimization with respect to both $\mathbf{D}_k$ and $\mathbf{X}_T^k$, but this time force the solution of $\tilde{\mathbf{X}}_T^k$ to have the same support as the original $\mathbf{X}_T^k$. This is equivalent to the minimization of

$$||\mathbf{E}_k \mathbf{\Omega}_k - \mathbf{D}_k \mathbf{X}_T^k \mathbf{\Omega}_k||_F^2 = ||\mathbf{E}_k^R - \mathbf{D}_k \mathbf{X}_R^k||_F^2 \tag{3.37}$$

and this time it can be done directly via SVD. Taking the restricted matrix $\mathbf{E}_k^R$, SVD decomposes it to $\mathbf{E}_k^R = \mathbf{U \Delta V}^T$. We define the solution for $\tilde{\mathbf{D}}_k$ as the first p columns of $\mathbf{U}$, and the coefficient vector $\tilde{\mathbf{X}}_R^k$ as the first p rows of $\mathbf{V}^T$ left multiplied by the top $p \times p$ sub-matrix in $\mathbf{\Delta}$. Note that, in this solution, we necessarily have that:

- the atoms of $\mathbf{D}$ remain normalized,
- the support of representations either stays the same or gets smaller by the possible nulling of terms.

The discussion about convergence and parallelism can be found in Rubinstein et al. [35].

Implementation Issue

Just like the original K-SVD, the proposed block K-SVD algorithm is susceptible to local minimum traps. Our experiments show that improved results can be reached if the following variations are applied.

- When using approximation methods with a fixed number of coefficients, we found that FOCUSS proves to be the best in terms of getting the best out of each iteration. However, from a run-time point of view, OMP was found to lead to a far more efficient overall algorithm. This is because FOCUSS is an iterative method to solve a re-weighted minimum norm solution, which needs several iterations to converge. In addition, the regularization parameter is decided by several FOCUSS attempts. However, OMP is a greedy method, getting the sparse solution in exactly K iterations. Obviously, it saves time to run OMP, a greedy method, than FOCUSS, an iterative method. On the other hand, as OMP is a greedy method, there is no guarantee that an optimal solution is reached by it, while FOCUSS is able to reach the optimal one with high possibility. Thus, OMP and FOCUSS is a balance between preciseness and efficiency. In our experiments, the global minimum is a more immersed need, such that preciseness is the very first thing to be considered. Thus, there is no doubt that one should exploit FOCUSS in K-SVD instead of OMP.
- When a dictionary atom is not being used "enough" relative to the number of dictionary atoms, the number of samples, and the sparsity of coefficients, it could be replaced with the least-represented signal element after being normalized. This is also suggested by Rubinstein et al.. Note that the size of dictionary atoms may differ from that of signal elements. For example, if the size of the block in a signal element is 3×2, then the size of the second dimension of a dictionary atom is 3, while the size of the second dimension of a signal element is 2. In this case, we replace the first 2 columns of a dictionary atom by the least represented signal element and fill the last column with a randomly-generated vector. One should be careful that such a newly generated dictionary atom is supposed to be *normalized* to 1, otherwise, it would never be used later.

3.3.2 Block OMP

To solve the block sparse approximation problem, we extend a regular Orthogonal Matching Pursuit (OMP) [40] to block OMP. Both of them are greedy algorithms, picking the first K atoms in the dictionary describing the signal best. Specifically, in each iteration, block OMP computes the inner product of residual and each dictionary atom left, and picks the atoms corresponding to the least inner product value. Then it computes coefficients, associates with chosen atoms, updates residual and repeats until the number of chosen atoms hits the known number K. Block OMP is efficient compared to block FOCUSS, but it succeeds only when the dictionary is sufficiently incoherent.

3.3.3 Block FOCUSS

The FOcal Underdetermined System Solver (FOCUSS), proposed by Gorodnitsky et al. [21], is an iterative method to solve the Sparse Coding problem. In experiments, FOCUSS is able to recover sparse causes from signals generated by a non-ideal dictionary, while Orthogonal Matching Pursuit (OMP) cannot. Here, to replace OMP with FOCUSS in K-SVD, we generalize FOCUSS to the block sparse situation and call it block FOCUSS.

Recall of FOCUSS

We first discuss the basic form of FOCUSS without power l or additional weight matrix W_{ak}, summarized into Algorithm 5.

```
Algorithm 5: Basic FOCUSS
  Data: the Dictionary $\mathbf{A}$, a representation $\mathbf{b}$
  Result: a sparse signal $\mathbf{x}$ satisfying $\mathbf{Ax} = \mathbf{b}$
  -Initialize $\mathbf{x}_0$ with blurred minimum norm solution;
  while not converged do
      -$\mathbf{W}_{pk} = \text{diag}(\mathbf{x}_{k-1})$;
      -$\mathbf{q}_k = (\mathbf{AW}_{pk})^\dagger \mathbf{b}$;
      -$\mathbf{x}_k = \mathbf{W}_{pk}\mathbf{q}_k$;
  end
  return $\mathbf{x} = \mathbf{x}_k$
```

The beauty of FOCUSS is to introduce the auxiliary variable $\mathbf{q}$ representing the sparse structure of vector $\mathbf{x}$ when converged: the element in $\mathbf{q}$ equals to 1 if the corresponding element in $\mathbf{x}$ is active and 0 if non-active. To understand how the introduction of this auxiliary variable helps to find the sparse solution, we consider the objective during iterations:

$$\|\mathbf{q}_k\|_2^2 = \|\mathbf{W}_{pk}^\dagger \mathbf{x}_k\|_2^2 = \sum_{i,w_i \neq 0} \left(\frac{x_i^k}{w_i^k}\right)^2 \tag{3.38}$$

The relatively large entries in $\mathbf{W}_{pk}$ reduce the contribution of the corresponding elements of $\mathbf{x}_k$ to the above cost, and vice versa [21]. The value of entries in $\mathbf{W}_{pk}$ depends on $\mathbf{x}_{k-1}$, and thus it follows that some entries in $\mathbf{x}_k$ become larger and larger but others smaller and smaller during iterations. When the algorithm converges to a stationary point, certain entries of such a solution must diminish to zero, which leads to a sparse solution.

Derivation of block FOCUSS

Denote $\mathbf{A}$ as the dictionary, $\mathbf{B}$ as the representation of signal $\mathbf{X}$ satisfying

$$\mathbf{AX} = \mathbf{B}, \tag{3.39}$$

where signal $\mathbf{X}$ is an unknown block sparse vector and each block is in shape $p \times q$. The basic idea of block FOCUSS is to design a weight matrix $\mathbf{W}_{pk}$ forcing block vector $\mathbf{X}_k$ into block sparse. A very intuitive idea is to make all elements in one block share one weight, which would force the behaviour of these elements consistent to each other. Thus, we use the Frobenius norm of the block as the weight shared by all elements inside the block. Block FOCUSS is summarized into Algorithm 6.

```
Algorithm 6: Basic block FOCUSS
  Data: the Dictionary $\mathbf{A}$, a representation $\mathbf{B}$
  Result: a sparse signal $\mathbf{X}$ satisfying $\mathbf{AX} = \mathbf{B}$
  -Initialize $\mathbf{X}_0$ with blurred minimum norm solution;
  -Denote $\mathbf{f}_k^X$ as a vector whose $i$-th entry is the Frobenius norm of $i$-th block in $\mathbf{X}_k$, and $\otimes$
   as the Kronecker product;
  while not converged do
      -Compute $\mathbf{f}_{k-1}^X$ from $\mathbf{X}_{k-1}$;
      -$\mathbf{W}_{pk} = \texttt{diag}(\mathbf{f}_{k-1}^X \otimes \mathbf{I}_p)$;
      -$\mathbf{Q}_k = (\mathbf{AW}_{pk})^\dagger \mathbf{B}$;
      -$\mathbf{X}_k = \mathbf{W}_{pk}\mathbf{Q}_k$;
  end
  return $\mathbf{X} = \mathbf{X}_k$
```

In FOCUSS, the auxiliary variable $\mathbf{q}_k$ actually represents the sparse structure of $\mathbf{x}_k$. Very similar to that, $\mathbf{Q}_k$ serves the same functionality in block FOCUSS. Specifically, denote $\mathbf{f}_k^Q$ as a vector whose i-th entry is the Frobenius norm of i-th block in $\mathbf{Q}_k$. From Algorithm 6, it is implied that each block in $\mathbf{Q}_k$, denoted by $\mathbf{Q}_k(i)$ satisfies that

$$\mathbf{Q}_k(i) = (\mathbf{W}_{pk}^\dagger \mathbf{X}_k)(i) = \frac{\mathbf{X}_k(i)}{\mathbf{f}_{k-1}^X(i)} = \frac{\mathbf{X}_k(i)}{\|\mathbf{X}_{k-1}(i)\|_F}. \tag{3.40}$$

When block FOCUSS converges, the entries in $\mathbf{f}_k^Q$, denoted by $\mathbf{f}_k^Q(i)$

$$\mathbf{f}_k^Q(i) = \frac{\|\mathbf{X}_k(i)\|_F}{\|\mathbf{X}_{k-1}(i)\|_F} \tag{3.41}$$

are either one or zero, which indicates whether the corresponding block in $\mathbf{X}_k$ is active or not. Further, the elements in $\mathbf{Q}_k(i)$ are normalized entries describing how active the corresponding element in each block is.

Implementation Issues

First, we discuss the stop criteria of block FOCUSS. There are many stop criteria we can pick up; for instances, computing the change of $\mathbf{X}_k$ during each iteration, computing the change of $\mathbf{Q}_k$ during each iteration, etc. In our implementation, we use $\mathbf{f}_k^Q$ to decide when to stop: when all entries in $\mathbf{f}_k^Q$ converge to one or zero, we say the block FOCUSS converges.

```
Algorithm 7: Complete block FOCUSS
  Data: the Dictionary $\mathbf{A}$, a representation $\mathbf{B}$
  Result: a sparse signal $\mathbf{X}$ satisfying $\mathbf{AX} = \mathbf{B}$
  -Initialize $\mathbf{X}_0$ with blurred minimum norm solution;
  -Denote $\mathbf{f}_k^X$ as a vector whose $i$-th entry is the Frobenius norm of $i$-th block in $\mathbf{X}_k$, and $\otimes$
    as the Kronecker product;
  while $\mathbf{f}_k^Q$ not converged do
      -Compute $\mathbf{f}_{k-1}^X$ from $\mathbf{X}_{k-1}$;
      -$\mathbf{W}_{pk} = \mathrm{diag}\left((\mathbf{f}_{k_1}^X)^l \otimes \mathbf{I}_p\right)$;
      -Regularization: $\mathbf{Q}_k = (\mathbf{A}\mathbf{W}_{ak}\mathbf{W}_{pk})^\dagger\mathbf{B}$;
      -Compute $\mathbf{f}_k^Q$ from $Q_k\mathbf{W}_{ak}$;
      -Hard Thresholding: $\mathbf{Q}_k = \mathcal{H}(\mathbf{Q}_k, \mathbf{f}_k^Q)$;
      -$\mathbf{X}_k = \mathbf{W}_{ak}\mathbf{W}_{pk}\mathbf{Q}_k$;
  end
  return $\mathbf{X} = \mathbf{X}_k$
```

To improve the performance of the proposed block FOCUSS, we introduce two parameters: some power l and an additional weight matrix $\mathbf{W}_{ak}$. The new algorithm is summarized in Algorithm 6.The similar idea can also be found in [21]. Specifically, a good choice of l can accelerate the convergence. The use of $\mathbf{W}_{ak}$ makes it possible to incorporate a priori information, which might extend the basin of the maximally sparse solution that we considered as the optimal solution. Note that a poorly designed $\mathbf{W}_{ak}$ may damage the functionality of auxiliary variable. We define $\mathbf{W}_{ak}$ as a block diagonal matrix where each block on the main diagonal is a scaled identity matrix $c_i\mathbf{I}$, where the scale c_i is the greatest absolute value of elements in i-th block column of $\mathbf{A}$. Therefore, by Algorithm 7, it is known that

$$\mathbf{X}_k(i) = \mathbf{W}_{ak}(i)\mathbf{W}_{pk}(i)\mathbf{Q}_k(i). \tag{3.42}$$

Thus, it is followed that

$$\mathbf{Q}_k(i) = \frac{\mathbf{X}_k(i)}{c_i\|\mathbf{X}_{k-1}(i)\|_F}. \tag{3.43}$$

By computing the limit of both sides, it is implied that

$$\lim_{k\to\infty} \|\mathbf{Q}_k(i)\|_F = \frac{1}{c_i}, \tag{3.44}$$

which maintains the functionality of the auxiliary variable $\mathbf{Q}$.

To accelerate the convergence, we introduce the hard thresholding operation. Denote ϵ as the hard thresholding operation constant. The block in $\mathbf{Q}_k$ will be truncated if corresponding entry in $\mathbf{f}_k^Q$ is less than ϵ during iterations. Consequently, since

$$\mathbf{X}_k = \mathbf{W}_{ak}\mathbf{W}_{pk}\mathbf{Q}_k, \tag{3.45}$$

the corresponding blocks in X_k are also truncated. This operation significantly saves computation, as well as provides better convergence and performance properties by eliminating the diminishing blocks.

3.3.4 Initialization via ADMMs

The BSDL factorization itself is inherently an NP-hard problem, therefore it is important to have a good initialization. We relax the BSDL objective using a block ℓ_1-norm, and solve the relaxed problem by the Alternating Direction Method of Multipliers (ADMMs) [1, 8, 10, 18]. Even though the relaxed problem is not convex either, ADMM splits the objective into several small *convex* sub-problems by introducing several auxiliary variables. A stationary point can be achieved for our ADMM initialization through the judicious choice of parameters [10].

Given data matrix $\mathbf{X}$, we want to reconstruct codebook $\mathbf{A}$ and code matrix $\mathbf{B}$, such that $\mathbf{B}$ has a $p \times q$ block sparse structure. Let's express this problem into an optimization format:

$$\begin{aligned} \min_{\mathbf{A},\mathbf{B}} \quad & \frac{1}{2}\|\mathbf{X} - \mathbf{A}\mathbf{B}\|_F^2 + \gamma\|\mathbf{B}\|_{21} \\ \text{s.t.} \quad & \|\mathbf{A}_i\|_F \leq 1 \text{ for } i = 1, \ldots, L \end{aligned} \tag{3.46}$$

where $\mathbf{X} \in \mathbb{R}^{m \times qn}$, $\mathbf{A} = \begin{bmatrix} \mathbf{A}_1 & \mathbf{A}_2 & \dots & \mathbf{A}_L \end{bmatrix} \in \mathbb{R}^{m \times pL}$, $\mathbf{A}_i \in \mathbb{R}^{m \times p}$, $\mathbf{B} \in \mathbb{R}^{pL \times qn}$. In the block matrix $\mathbf{B}$, each block that is in shape $p \times q$. $\left\| \cdot \right\|_{21}$ denotes ℓ-21 norm, which first computes ℓ_2 norm of each block, and then computes the summation of them.

ADMMs allows us to employ the two dummy variables $\mathbf{C}, \mathbf{D}$ into the optimization problem:

$$\begin{aligned} \min_{\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D}} \quad & \frac{1}{2}\left\|\mathbf{X} - \mathbf{A}\mathbf{B}\right\|_F^2 + \gamma\left\|\mathbf{C}\right\|_{21} \\ \text{s.t.} \quad & \|\mathbf{D}_i\|_F \leq 1 \text{ for } i = 1, \dots, L \\ & \mathbf{A} = \mathbf{D}, \\ & \mathbf{B} = \mathbf{C}. \end{aligned} \tag{3.47}$$

At first glance, this objective seems to buy us nothing, except to complicate things. However, we can form the augmented Lagrangian of the above objective, which becomes:

$$\begin{aligned} \mathcal{L}_{\rho,\mu}(\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}, \mathbf{\Lambda}_A, \mathbf{\Lambda}_B) = & \frac{1}{2}\left\|\mathbf{X} - \mathbf{A}\mathbf{B}\right\|_F^2 + \gamma\left\|\mathbf{C}\right\|_{21} + \frac{\rho}{2}\left\|\mathbf{B} - \mathbf{C}\right\|_F^2 + \frac{\mu}{2}\left\|\mathbf{A} - \mathbf{D}\right\|_F^2 \\ & + \text{vec}(\mathbf{\Lambda}_A)^T\text{vec}(\mathbf{A} - \mathbf{D}) + \text{vec}(\mathbf{\Lambda}_B)^T\text{vec}(\mathbf{B} - \mathbf{C}) \end{aligned}$$

where $\mathbf{\Lambda}_A \in \mathbb{R}^{m \times pL}$ and $\mathbf{\Lambda}_B \in \mathbb{R}^{pL \times qn}$ are the Lagrange multipliers, ρ and μ are penalty weighting for two auxiliary variables $\mathbf{C}$ and $\mathbf{D}$, respectively.

ADMMs consists of the iterations

$$\begin{aligned} \mathbf{A}^{k+1} &= \underset{\mathbf{A}}{\text{argmin}}\mathcal{L}_{\rho,\mu}(\mathbf{A}, \mathbf{B}^k, \mathbf{C}^k, \mathbf{D}^k) \\ \mathbf{B}^{k+1} &= \underset{\mathbf{B}}{\text{argmin}}\mathcal{L}_{\rho,\mu}(\mathbf{A}^{k+1}, \mathbf{B}, \mathbf{C}^k, \mathbf{D}^k) \\ \mathbf{C}^{k+1} &= \underset{\mathbf{C}}{\text{argmin}}\mathcal{L}_{\rho,\mu}(\mathbf{A}^{k+1}, \mathbf{B}^{k+1}, \mathbf{C}, \mathbf{D}^k) \\ \mathbf{D}^{k+1} &= \underset{\mathbf{D}}{\text{argmin}}\mathcal{L}_{\rho,\mu}(\mathbf{A}^{k+1}, \mathbf{B}^{k+1}, \mathbf{C}^{k+1}, \mathbf{D}) \\ \mathbf{\Lambda}_A^{k+1} &= \mathbf{\Lambda}_A^k + \mu(\mathbf{A}^{k+1} - \mathbf{D}^{k+1}) \\ \mathbf{\Lambda}_B^{k+1} &= \mathbf{\Lambda}_B^k + \rho(\mathbf{B}^{k+1} - \mathbf{C}^{k+1}) \end{aligned} \tag{3.48}$$

Now, let's detail each of the subproblems.

subproblem A:

$$\begin{aligned} \mathbf{A}^* = & \underset{\mathbf{A}}{\text{argmin}} \quad \mathcal{L}(\mathbf{A}; \mathbf{B}, \mathbf{C}, \mathbf{D}, \mathbf{\Lambda}_A, \mathbf{\Lambda}_B) \\ = & \underset{\mathbf{A}}{\text{argmin}} \quad \frac{1}{2}\left\|\mathbf{X} - \mathbf{A}\mathbf{B}\right\|_F^2 + \frac{\mu}{2}\left\|\mathbf{A} - \mathbf{D}\right\|_F^2 + \text{vec}(\mathbf{\Lambda}_A)^T\text{vec}(\mathbf{A}) \end{aligned} \tag{3.49}$$

Since Equation 3.49 is a quadratic problem, let's directly find optimal points where the gradient equals zero.

$$\frac{\partial \mathcal{L}}{\partial \mathbf{A}} = (\mathbf{AB} - \mathbf{X})\mathbf{B}^T + \mu(\mathbf{A} - \mathbf{D}) + \mathbf{\Lambda}_A = \mathbf{0}, \tag{3.50}$$

$$\mathbf{A}^* = (\mathbf{XB}^T + \mu\mathbf{D} - \mathbf{\Lambda}_A)(\mathbf{BB}^T + \mu\mathbf{I})^{-1}. \tag{3.51}$$

subproblem B:

$$\begin{aligned} \mathbf{B}^* &= \underset{\mathbf{B}}{\operatorname{argmin}} \quad \mathcal{L}(\mathbf{B}; \mathbf{A}, \mathbf{C}, \mathbf{D}, \mathbf{\Lambda}_A, \mathbf{\Lambda}_B) \\ &= \underset{\mathbf{B}}{\operatorname{argmin}} \quad \frac{1}{2}\left\|\mathbf{X} - \mathbf{AB}\right\|_F^2 + \frac{\rho}{2}\left\|\mathbf{B} - \mathbf{C}\right\|_F^2 + \text{vec}(\mathbf{\Lambda}_B)^T\text{vec}(\mathbf{B}) \end{aligned} \tag{3.52}$$

Since Equation 3.52 is a quadratic problem, let's directly find optimal points where the gradient equals zero.

$$\frac{\partial \mathcal{L}}{\partial \mathbf{B}} = \mathbf{A}^T(\mathbf{AB} - \mathbf{X}) + \rho(\mathbf{B} - \mathbf{C}) + \mathbf{\Lambda}_B = \mathbf{0}, \tag{3.53}$$

$$\mathbf{B}^* = (\mathbf{A}^T\mathbf{A} + \rho\mathbf{I})^{-1}(\mathbf{A}^T\mathbf{X} + \rho\mathbf{C} - \mathbf{\Lambda}_B). \tag{3.54}$$

subproblem C:

$$\begin{aligned} \mathbf{C}^* &= \underset{\mathbf{C}}{\operatorname{argmin}} \quad \mathcal{L}(\mathbf{C}; \mathbf{A}, \mathbf{B}, \mathbf{D}, \mathbf{\Lambda}_A, \mathbf{\Lambda}_B) \\ &= \underset{\mathbf{C}}{\operatorname{argmin}} \quad \gamma\left\|\mathbf{C}\right\|_{21} + \frac{\rho}{2}\left\|\mathbf{B} - \mathbf{C}\right\|_F^2 - \text{vec}(\mathbf{\Lambda}_B)^T\text{vec}(\mathbf{C}) \end{aligned} \tag{3.55}$$

Since there is no rotation of $\mathbf{C}$ in Equation 3.55, each block $\mathbf{C}_{ij} \in \mathbb{R}^{p\times q}$ of $\mathbf{C}$ can be solved independently,

$$\mathbf{C}_{ij}^* = \underset{\mathbf{C}_{ij}}{\operatorname{argmin}} \quad \gamma\left\|\mathbf{C}_{ij}\right\|_2 + \frac{\rho}{2}\left\|\mathbf{B}_{ij} - \mathbf{C}_{ij}\right\|_2^2 - \text{vec}((\mathbf{\Lambda}_B)_{ij})^T\text{vec}(\mathbf{C}_{ij}) \tag{3.56}$$

which has a closed form solution by the one-dimensional shrinkage (or soft thresholding) formula:

$$\mathbf{C}_{ij}^* = \max\left\{\left\|\mathbf{r}_{ij}\right\|_2 - \frac{\gamma}{\rho}, 0\right\}\frac{\mathbf{r}_{ij}}{\left\|\mathbf{r}_{ij}\right\|_2}, \tag{3.57}$$

where

$$\mathbf{r}_{ij} := \mathbf{B}_{ij} + \frac{1}{\rho}(\mathbf{\Lambda}_B)_{ij}. \tag{3.58}$$

subproblem D:

$$\begin{aligned} \mathbf{D}^* =& \underset{\mathbf{D}}{\operatorname{argmin}} \quad \mathcal{L}(\mathbf{D}; \mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{\Lambda}_A, \mathbf{\Lambda}_B) \\ =& \underset{\mathbf{D}}{\operatorname{argmin}} \quad \frac{\mu}{2}\left\|\mathbf{A} - \mathbf{D}\right\|_F^2 - \text{vec}(\mathbf{\Lambda}_A)^T\text{vec}(\mathbf{D}) \\ & \text{s.t.} \qquad \|\mathbf{D}_i\|_F \le 1 \text{ for } i = 1, \ldots, Lp \end{aligned} \tag{3.59}$$

Since Equation 3.59 doesn't contain any rotation, we could solve $\mathbf{D}_i$ independently,

$$\begin{aligned} \mathbf{D}_i^* = \ & \underset{\mathbf{D}_i}{\text{argmin}} \quad \frac{\mu}{2}\left\|\mathbf{A}_i - \mathbf{D}_i\right\|_F^2 - (\mathbf{\Lambda}_{A,i})^T\mathbf{D}_i \\ & \text{s.t.} \qquad \|\mathbf{D}_i\|_F \le 1 \end{aligned} \tag{3.60}$$

Further, in the quadratic term in Equation 3.60, there is no matrix left multiply $\mathbf{D}_i$, so the iso-contour of the objective function is isotropic. Hence, fortunately, we can solve the unconstrained problem, and then project it into a feasible region.

$$\widetilde{\mathbf{D}}_i^* = \mathbf{A}_i + \frac{1}{\mu}(\mathbf{\Lambda}_{A,i}) \tag{3.61}$$

$$\mathbf{D}_i^* = \frac{\widetilde{\mathbf{D}}_i^*}{\max\{\left\|\widetilde{\mathbf{D}}_i^*\right\|_F, 1\}}. \tag{3.62}$$

Penalty Update: Super linear convergence of the ADMMs may be achieved if $\mu, \rho \to \infty$. In practice, we lint the value of μ, ρ to avoid poor conditioning and numerical errors. Specifically, we adopt the following update strategy:

$$\mu^{k+1} = \begin{cases} \tau\mu^k & \text{if } \mu^k < \mu_{max} \\ \mu^k & \text{otherwise} \end{cases} \tag{3.63}$$

$$\rho^{k+1} = \begin{cases} \tau\rho^k & \text{if } \rho^k < \rho_{max} \\ \rho^k & \text{otherwise} \end{cases} \tag{3.64}$$

3.3.5 Camera and Structure Recovery

As the scale of cameras and the sizes of structures are inherently relative, we simply set the camera scale σ to unity, such that $\mathbf{M}_f\mathbf{M}_f^T = \mathbf{I}_2$. Assuming that $\mathbf{W} = \hat{\Pi}\hat{\mathbf{B}}$ has a unique BSDL, from Definition 1, the corrective matrix $\mathbf{G}$ must be of form $\mathbf{G} = (\mathbf{P} \otimes \mathbf{I}_3)\mathbf{\Lambda}$. As the permutation ambiguity has no bearing on camera motion and 3D structure, we set $\mathbf{P}$ to identity, therefore $\mathbf{G} = \mathbf{\Lambda}$.

Denote $\mathbf{G}_j$ as j-th block on diagonal of $\mathbf{G}$, and $\hat{\mathbf{\Pi}}_j, \mathbf{\Pi}_j \in \mathbb{R}^{2F\times 3}$ as the j-th coloumn-triplet of $\hat{\mathbf{\Pi}}, \mathbf{\Pi}$ respectively. From the structure of the corrective matrix, it follows that $\mathbf{\Pi}_j = \hat{\mathbf{\Pi}}_j\mathbf{G}_j$, for $j = 1, ..., L$. Define Ω_j as the set of indices pointing to the block $\hat{\mathbf{\Pi}}_{ij} \in \mathbb{R}^{2\times 3}$ that is active, i.e.$\Omega_j = \text{supp}(\hat{\mathbf{\Pi}}_j) = \{i|1 \le i \le F, \hat{\mathbf{\Pi}}_{ij} \neq \mathbf{0}\}$. If a certain Ω_j is empty, it is implied that the corresponding atom in the dictionary has never been used. We can then decrease L, and re-learn the dictionary so that Ω_j is never empty.

From the projection equation (i.e.$\mathbf{W} = \mathbf{M}(\mathbf{C}^\sharp \otimes \mathbf{I}_3)\mathbf{B} = \mathbf{\Pi}\mathbf{B}$), it is known that $\mathbf{\Pi}_{ij} = c_{ij}\mathbf{M}_i$, where c_{ij} is $ij-$th elements of $\mathbf{C}^\sharp$. Thus, since Ω_j can never be empty, $\hat{\mathbf{\Pi}}_{ij}\mathbf{G}_j = \mathbf{\Pi}_{ij} = c_{ij}\mathbf{M}_i$, for each $i \in \Omega_j$. From camera constraints, it follows that

$$\hat{\mathbf{\Pi}}_{ij}\mathbf{G}_j\mathbf{G}_j^T\hat{\mathbf{\Pi}}_{ij}^T = c_{ij}^2\mathbf{M}_i\mathbf{M}_i^T = c_{ij}^2\mathbf{I}_2, \quad i \in \Omega_j, \tag{3.65}$$

and for convenience, let $\mathbf{Q}_j = \mathbf{G}_j\mathbf{G}_j^T$. Since c_{ij} is unknown, let us eliminate it and rewrite Equation 3.65 as

$$(\hat{\mathbf{\Pi}}_{ij}\mathbf{Q}_j\hat{\mathbf{\Pi}}_{ij}^T)_{11} = (\hat{\mathbf{\Pi}}_{ij}\mathbf{Q}_j\hat{\mathbf{\Pi}}_{ij}^T)_{22}, (\hat{\mathbf{\Pi}}_{ij}\mathbf{Q}_j\hat{\mathbf{\Pi}}_{ij}^T)_{12} = 0, \tag{3.66}$$

where $(\cdot)_{ij}$ denotes the (i, j)-th elements. Now, denote $\mathbf{q}_j = \text{vec}(\mathbf{Q}_j)$ as the vectorization of $\mathbf{Q}_j$. Let us rewrite Equation 3.66 in a compact way with the fact that $\text{vec}(\hat{\mathbf{\Pi}}_{ij}\mathbf{Q}_j\hat{\mathbf{\Pi}}_{ij}) = (\hat{\mathbf{\Pi}}_{ij} \otimes \hat{\mathbf{\Pi}}_{ij})\mathbf{q}_j$:

$$\begin{bmatrix} \hat{\mathbf{\Pi}}_{ij} \otimes \hat{\mathbf{\Pi}}_{ij}(1,:) - \hat{\mathbf{\Pi}}_{ij} \otimes \hat{\mathbf{\Pi}}_{ij}(4,:) \\ \hat{\mathbf{\Pi}}_{ij} \otimes \hat{\mathbf{\Pi}}_{ij}(2,:) \end{bmatrix} \mathbf{q}_j = \mathbf{A}_{ij}\mathbf{q}_j = 0, \tag{3.67}$$

where $\hat{\mathbf{\Pi}}_{ij} \otimes \hat{\mathbf{\Pi}}_{ij}(k,:)$ denotes k-th row of $\hat{\mathbf{\Pi}}_{ij} \otimes \hat{\mathbf{\Pi}}_{ij}$. Stacking all such equations for all $i \in \Omega_j$, we obtain

$$\mathbf{A}_j\mathbf{q}_j = 0. \tag{3.68}$$

Circumventing the nullspace

One benefit of Equation 3.68 is that $\mathbf{A}_j \in \mathbb{R}^{2|\Omega_j| \times 9}$, where $|\Omega_j|$ is the number of elements in set Ω_j, with the high possibility it will be overcomplete as $F \gg L$. This result is important, as it circumvents the nullspace issue faced by low-rank NRS*f*M. This null space issue can be problematic in many practical scenarios due to its sensitivity to noise. Similar to Tomasi-Kanade's method [38], we simply pick up the eigenvector corresponding to the least eigenvalue of $\mathbf{A}_j^T\mathbf{A}_j$ and then $\mathbf{Q}_k \in \mathbb{S}_+^3$ holds automatically.

Once $\mathbf{Q}_j$ is estimated, the absolute value of c_{ij} can be computed by Equation 3.65. The sign of c_{ij}, however, is not able to be determined, which actually is an inherent ambiguity without assuming any temporal prior of camera or structures. Considering equation $\mathbf{W} = \mathbf{M}\mathbf{S}$, any block diagonal matrix blkdiag($\pm\mathbf{I}_3$) can be inserted between $\mathbf{M}$ and $\mathbf{S}$, but the compressibility assumption and camera constraint still hold. Dai *et al.* [14] *breaks* their "prior-free" assertion by restricting the camera movement between frames to at most ±90° to determine the sign of c_{ij}. In our paper, however, we claim that the absolute sign of c_{ij} cannot be determined by the current assumption, but the relative sign in each column can. Thus, the camera matrix and structures can be recovered, but up to a sign ambiguity.

Enforcing camera consistency

Let us consider the submatrix $\mathbf{G}_j$ in isolation,

$$\hat{\mathbf{\Pi}}_{ij}\mathbf{G}_j = c_{ij}\mathbf{M}_i, \quad \text{for } i \in \Omega_j. \tag{3.69}$$

One can recover the camera matrices $\{\mathbf{M}_i\}_{i\in\Omega_j}$ by solving the system of equations above. Further, if one was to then choose another $\mathbf{G}_k$ where $j \neq k$, such that one or more indexes in Ω_j are shared with Ω_k, one can equally recover the camera matrices $\{\mathbf{M}_i^*\}_{i\in\Omega_k}$. An inconsistency arises, however, such that we cannot guarantee that

$$\mathbf{M}_i^* = \mathbf{M}_i, \quad \text{for} \quad i \in \Omega_j \cap \Omega_k. \tag{3.70}$$

This inconsistency does not just occur across pairs of submatrices within $\mathbf{G}$, but actually across all possible submatrices of $\mathbf{G}$ with overlapping active blocks. We attempt to resolve this inconsistency in a recursive manner by solving for an orthonormal matrix $\mathbf{H}_k$, such that $\mathbf{M}_i^*\mathbf{H}_k = \mathbf{M}_i$. First, we choose an arbitrary $\mathbf{G}_j$ (typically the one with the most active blocks) and solve for the cameras $\{\mathbf{M}_i\}_{i\in\Gamma}$, where we initially set $\Gamma = \Omega_j$. Then we choose a $\mathbf{G}_k$ whose $|\Gamma \cap \Omega_k|$ is largest. We solve for the cameras $\{\mathbf{M}_i^*\}_{i\in\Omega_k}$, and then find an orthonormal $\mathbf{H}_k$ such that,

$$\underset{\mathbf{H}_k,\boldsymbol{\eta}}{\operatorname{argmin}} \sum_{i\in\Gamma\cap\Omega_k} \|\mathbf{M}_i - \eta_i\mathbf{M}_i^*\mathbf{H}_k\|_F \quad \text{s.t. } \mathbf{H}_k^T\mathbf{H}_k = \mathbf{I}, \eta_i = \{+1, -1\}, \tag{3.71}$$

where η_i contains the relative sign of elements in $\mathbf{C}^\sharp$ for Γ. For the element in $\mathbf{C}^\sharp$ that are not explicitly defined through $\boldsymbol{\eta}$, we set them arbitrarily to be positive. We then update $\Gamma \leftarrow \Gamma \cup \Omega_k$ and repeat the process until all cameras and relative signs in $\mathbf{C}^\sharp$ are known. The structure matrix S is then recovered by $(\mathbf{C}^\sharp \otimes \mathbf{I}_3)\mathbf{H}^{-1}\mathbf{G}^{-1}\mathbf{B}$, where $\mathbf{H}$ is a matrix with $\mathbf{H}_1, ..., \mathbf{H}_L$ on main diagonal.

3.4 Experiments

3.4.1 Compressibility

Our first experiment explores the compressibility of real 3D structures from the CMU Motion Capture dataset, where we learned various dictionaries with different dictionary size L and sparsity level K. Figure 3.1 clearly shows that the real 3D structures are modeled well by our compressibility assumption and the coherence of the learned dictionary is being controlled by balancing the approximation error. This result offers a strategy to achieve a unique BSDL factorization at the cost of approximating structures less precisely, which extends the application of our method.

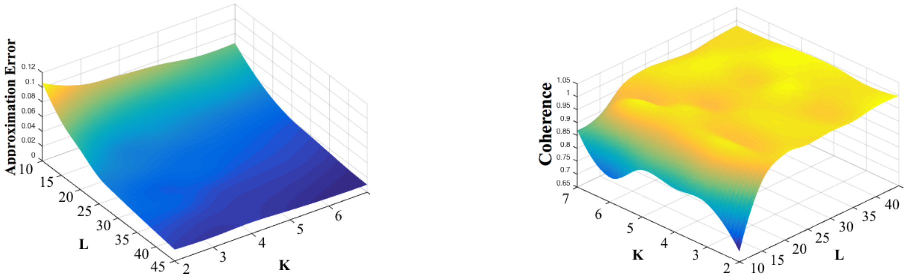


Figure 3.1: The results of SDL factorization for Motion-4 by Subject-5 in CMU Motion Capture. **Left:** The approximation error. **Right:** The coherence of a learned dictionary. With the decrease of K and L, the coherence of the learned dictionary becomes better at the cost of approximating structures less precisely.

3.4.2 Recovering temporal order

In Figure 3.2 we demonstrated that the sparse codes recovered using our method have a natural temporal coherence. This indicates our prior-less approach could be useful for the recovery of the temporal order of 3D structures in future applications. The full analysis of this phenomena is outsize of the scope of this paper.

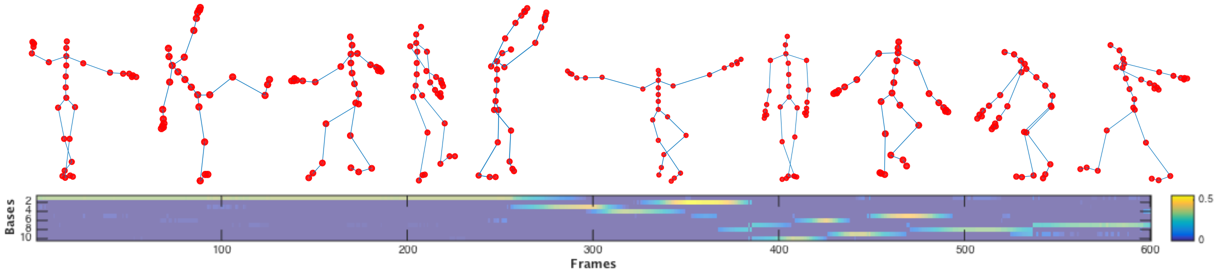


Figure 3.2: **Top:** 10 learned basis structures for Motion-4 by Subject-5 in CMU Motion Capture when $K = 2, L = 10$. These bases are learned from 3D shape sequences and identical to those learned from 2D image sequences, due to the uniqueness of BSDL. **Bottom:** The visualization of coefficients. The coefficients of each atom vary gradually in a shape of Gaussian distribution, which reveals the temporal information of video sequence. It is not used in NRSfM, but may be useful for recovering the temporal order of 3D structure in future applications.

3.4.3 High-rank performance

To verify the performance of the proposed method on high-rank and full-rank structures, we conducted experiments with synthetic data where the rank of structures is easily controlled. We utilized Dai et al.'s work as a baseline, which demonstrated that it outperforms other low-rank NRSfM methods in [14]. Note that for a fair comparison, we visit all possible rank k for [14] to ensure a best baseline estimation.

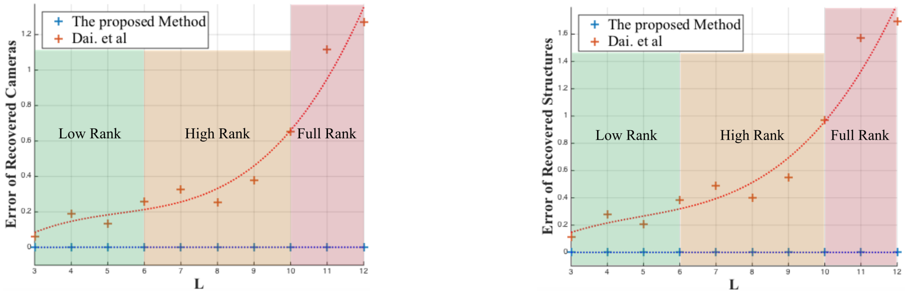


Figure 3.3: **Left:** The error of estimated camera matrix. **Right:** The error of estimated structures. The error matrices follow [5, 14, 22]. Our methods obtained nearly perfect results irrespective to the rank of structures.

The compressible structure $\mathbf{S}$, with 100 frames and 30 points in each frame, are generated by a random dictionary of size L, such that rank$(\mathbf{S}) = 3L$. We repeat the proposed method as well as Dai et al.'s method 50 times for each L from 3 to 12. The results are summarized in Figure 3.3. It is seen that our method works perfectly and robustly on structures with any rank, while the low-rank NRSfM fails in high-rank and full-rank situations. Moreover, even in a low-rank situation, the proposed method outperforms the Dai et al.'s method.

3.4.4 Noise performance

To evaluate the performance under noise, we repeat the experiments on low-rank structures (with $L = 5$) at different noise ratios, defined as $\frac{\|W-W_0\|_F}{\|W_0\|_F}$. Figure 3.4 demonstrates that our method is sensitive to noise. However, it still works no worse than Dai et al.'s method even at high noise ratios.

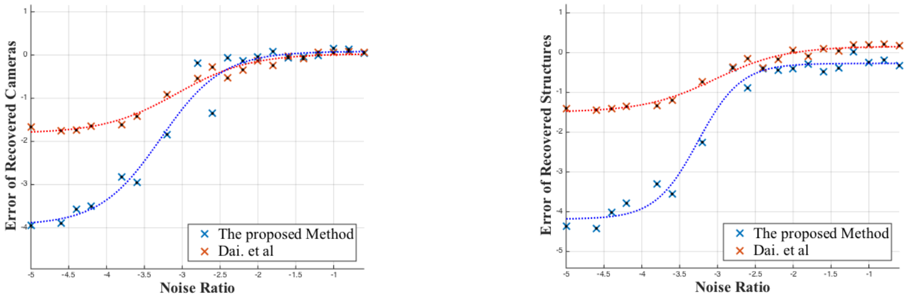


Figure 3.4: **Left:** The error of estimated camera matrix. **Right:** The error of estimated structures. Both x- and y-axis are in logarithm space. Our method is sensitive to noise, while it still works no worse than the baseline even at high noise.

3.4.5 Practical performance

The proposed method is evaluated on real compressible structures: Motion-4, -5, -6, -7, -8 by Subject-5, and Motion-2, -4 by Subject-1, Motion-5 by Subject-2, Motion-3, -4 by Subject-3 and Motion-13 by Subject-6 in CMU Motion Captures, and a Shark sequence in [39]. The visual evaluation shows that our method obtains impressive results in Figure 3.5, 3.6, 3.7, while it fails in Figure 3.8. Actually, this failure is able to be forecast even without ground truth. The coherence of the learned dictionary for sequence Shark is too poor to guarantee the uniqueness of the BSDL factorization. This insight offers an effective way to predict the reconstructibility of 3D structure when the ground truth structure is not available in practice.

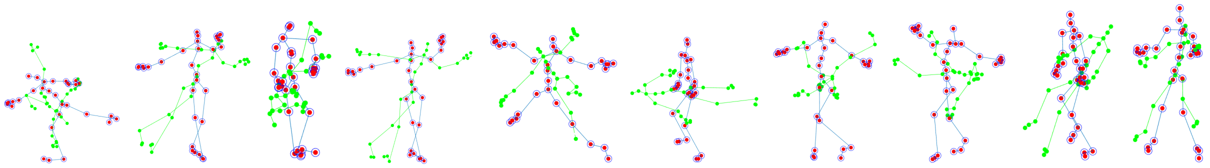


Figure 3.5: Random Sampled frames from Motion-4,-5,-6 by Subject-5.

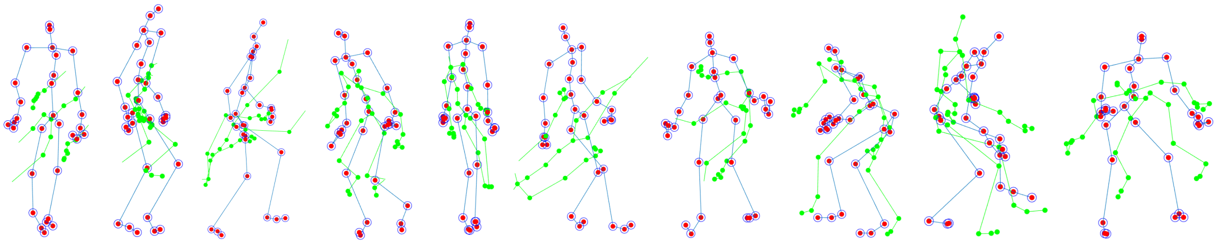


Figure 3.6: Random Sampled frames from Motion-3,-4 by Subject-3 and Motion-13 by Subject-6.

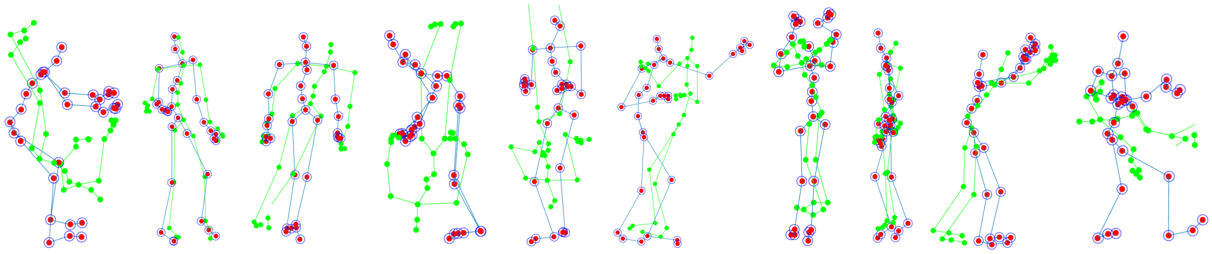


Figure 3.7: Random Sampled frames from Motion-2,-4 by Subject-1 and Motion-5 by Subject-2.

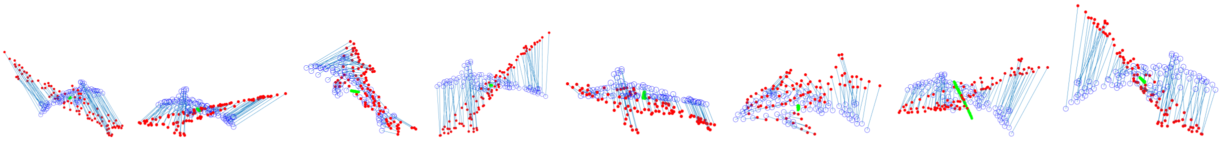


Figure 3.8: Random Sampled frames from Shark sequence.

Chapter 4

Convex Relaxation and Alternating Direction Method of Multipliers

This chapter introduces the method of Structure from Category (SfC) to infer 3D structures of objects in images stemming from the same object category. SfC is built upon the insight that the shape space describing an object category (e.g. aeroplane) is inherently non-rigid, even though individual instances of the category may be rigid. In other words, the shape of each instance can be modeled as a deformation from its category's general shape. Based on this observation, we frame SfC through an augmented sparse shape-space model that estimates the 3D shape of an object as a sparse linear combination of a set of rotated shape bases.

The proposed SfC is a *generic* and *prior-less* 3D reconstruction algorithm. Unlike current NRSfM methods, which are mainly limited to very few deformable objects (e.g. the human body and face), SfC can be generally applied on any object category, due to the non-rigid assumption of objects' shape space. Moreover, all parameters, including shape bases, sparse coefficients, and (scaled) camera motion, are jointly learned though an iterative manner, with *no* constraint on camera motion, 3D shape structure, temporal order, and deformation patterns (prior-less). Being generic and prior-less with no learning procedure in advance offers robust, large-scale 3D reconstruction for unseen object images and categories.

4.1 Problem Formulation

Inspired by the augmented sparse shape-space model [50], the 3D shape of instance f, $\mathbf{S}_f \in \mathbb{R}^{3\times P}$, can be well-approximated as a linear combination of a set of L rotated 3D *shape bases*

$\{\mathbf{B}_l\}_{l=1}^{L}$:

$$\mathbf{S}_f = \sum_{l=1}^{L} c_{fl}\mathbf{R}_{fl}\mathbf{B}_l, \tag{4.1}$$

where $\mathbf{B}_l \in \mathbb{R}^{3\times P}$, represented by the location of P key points in the 3D space, describe the object's shape space. $\mathbf{R}_{fl} \in \mathbb{R}^{3\times 3}$ and c_{fl}, respectively, refer to the rotation matrix and the coefficient of the l-th shape base and the f-th instance.

Given a set of F instances of the same object category, Eq(4.1) can be written as :

$$\begin{bmatrix}\mathbf{S}_1\\ \vdots\\ \mathbf{S}_F\end{bmatrix} = \begin{bmatrix} c_{11}\mathbf{R}_{11} & \cdots & c_{1L}\mathbf{R}_{1L}\\ \vdots & \vdots & \vdots\\ c_{F1}\mathbf{R}_{F1} & \cdots & c_{FL}\mathbf{R}_{FL}\end{bmatrix} \begin{bmatrix}\mathbf{B}_1\\ \vdots\\ \mathbf{B}_L\end{bmatrix}. \tag{4.2}$$

The projection of $\{\mathbf{S}_f\}_{f=1}^{F}$ into the image plane, $\{\mathbf{W}_f\}_{f=1}^{F}$, is computed by:

$$\begin{bmatrix}\mathbf{W}_1\\ \vdots\\ \mathbf{W}_F\end{bmatrix} = \begin{bmatrix}\mathbf{KS}_1\\ \vdots\\ \mathbf{KS}_F\end{bmatrix} + \begin{bmatrix}\mathbf{T}_1\\ \vdots\\ \mathbf{T}_F\end{bmatrix} = \begin{bmatrix} c_{11}\mathbf{KR}_{11} & \cdots & c_{1L}\mathbf{KR}_{1L}\\ \vdots & \vdots & \vdots\\ c_{F1}\mathbf{KR}_{F1} & \cdots & c_{FL}\mathbf{KR}_{FL}\end{bmatrix} \begin{bmatrix}\mathbf{B}_1\\ \vdots\\ \mathbf{B}_L\end{bmatrix} + \begin{bmatrix}\mathbf{T}_1\\ \vdots\\ \mathbf{T}_F\end{bmatrix}, \tag{4.3}$$

where we denote translation by $\mathbf{T}_f$, and projection matrix by $\mathbf{K}$. $\mathbf{W}_f \in \mathbb{R}^{2\times P}$ contains the 2D locations of P key points projected into the image plane. We consider weak-perspective cameras, which is a reasonable assumption for objects whose variation in depth is small compared to their distance from the camera; i.e.$\mathbf{K} = \begin{bmatrix}1 & 0 & 0\\ 0 & 1 & 0\end{bmatrix}$.

Denoting $\mathbf{M}_{fl} = c_{fl}\mathbf{KR}_{fl}$, Eq(4.3) can be written as:

$$\begin{bmatrix}\mathbf{W}_1\\ \vdots\\ \mathbf{W}_F\end{bmatrix} = \begin{bmatrix} \mathbf{M}_{11} & \cdots & \mathbf{M}_{1L}\\ \vdots & \vdots & \vdots\\ \mathbf{M}_{F1} & \cdots & \mathbf{M}_{FL}\end{bmatrix} \begin{bmatrix}\mathbf{B}_1\\ \vdots\\ \mathbf{B}_L\end{bmatrix} + \begin{bmatrix}\mathbf{T}_1\\ \vdots\\ \mathbf{T}_F\end{bmatrix} \tag{4.4}$$

and more concisely in the matrix form as,

$$\mathbf{W} = \mathbf{MB} + \mathbf{T} \tag{4.5}$$

The goal of S*f*C is to jointly compute $\mathbf{M}$ (projected rotation matrix), $\mathbf{B}$ (shape bases), and $\mathbf{T}$ (translation), using $\mathbf{W}$ (location of corresponding key points in a set of 2D images). This is performed by minimizing the *projection error* subject to the scaled orthogonality constraint on each $\mathbf{M}_{fl}$ and the sparsity constraint on the number of shape bases activated for each instance, which is framed as:

$$\begin{aligned}\min_{\mathbf{M},\mathbf{B},\mathbf{T}} \quad & \frac{1}{2}\Big\|\mathbf{\Gamma}\odot\big(\mathbf{MB}+\mathbf{T}\big)-\mathbf{W}\Big\|_F^2+\lambda\|\mathbf{C}\|_1 \\ \text{s.t.} \quad & \mathbf{M}_{fl}\mathbf{M}_{fl}^T=c_{fl}^2\mathbf{I}_2,\ f=1,...,F,\ l=1,...,L, \\ & \|\mathbf{B}_l\|_F=1,\ f=1,...,F,\end{aligned} \tag{4.6}$$

where $\mathbf{C}=[c_{fl}]$ and $\|\mathbf{C}\|_1$ computes the summation of ℓ_1-norm of each row in $\mathbf{C}$. $\|.\|_F$ denotes the Frobenius norm of a matrix, and $\mathbf{\Gamma}$ is a binary matrix that encodes the visibility (1) and occlusion (0) of each key point. The objective in Eq(4.6) is non-convex due to the multiplication of $\mathbf{M}$ and $\mathbf{B}$ and the orthogonality constraint on each $\mathbf{M}_{fl}$. To make the problem more convex, we utilize the relaxation strategy proposed by Zhou et al. [50] that eliminates the orthogonality constraint by replacing it with a spectral norm regularization. In this case, Eq(4.6) is relaxed as:

$$\begin{aligned}\min_{\mathbf{M},\mathbf{B},\mathbf{T}} \quad & \frac{1}{2}\Big\|\mathbf{\Gamma}\odot\big(\mathbf{MB}+\mathbf{T}\big)-\mathbf{W}\Big\|_F^2+\lambda\sum_{l,f}\|\mathbf{M}_{fl}\|_2 \\ \text{s.t.} \quad & \|\mathbf{B}_l\|_F=1,\ l=1,...,L,\end{aligned} \tag{4.7}$$

where $\|.\|_2$ here is the spectral norm of a matrix. The Alternating Direction Method of Multipliers (ADMM) [8] will be utilized to solve the objective in Eq(4.7).

4.2 Optimization via ADMM

Our proposed approach for solving Eq(4.7) involves the introduction of two auxiliary variables $\mathbf{Z}$ and $\mathbf{A}$. In this case, Eq(4.7) can be identically expressed as:

$$\begin{aligned}\min_{\mathbf{M},\mathbf{B},\mathbf{T},\mathbf{Z},\mathbf{A}} \quad & \frac{1}{2}\Big\|\mathbf{\Gamma}\odot\big(\mathbf{ZB}+\mathbf{T}\big)-\mathbf{W}\Big\|_F^2+\lambda\sum_{f,l}\|\mathbf{M}_{fl}\|_2 \\ \text{s.t.} \quad & \mathbf{M}=\mathbf{Z},\ \mathbf{A}=\mathbf{B}, \\ & \|\mathbf{A}_l\|_F=1,\ l=1,...,L.\end{aligned} \tag{4.8}$$

The augmented Lagrangian of Eq(4.8) is formulated as:

$$\begin{aligned}\mathcal{L}(\mathbf{M},\mathbf{Z},\mathbf{B},\mathbf{A},\mathbf{T},\mathbf{\Lambda},\mathbf{\Pi}) = & \frac{1}{2}\Big\|\mathbf{\Gamma}\odot\Big(\mathbf{ZB}+\mathbf{T}\Big)-\mathbf{W}\Big\|_F^2 \\ & +\lambda\sum_{f,l}\|\mathbf{M}_{fl}\|_2+\frac{\mu}{2}\Big\|\mathbf{M}-\mathbf{Z}\Big\|_F^2+\frac{\rho}{2}\Big\|\mathbf{A}-\mathbf{B}\Big\|_F^2 \\ & +\Big\langle\mathbf{\Lambda},\mathbf{M}-\mathbf{Z}\Big\rangle_F+\Big\langle\mathbf{\Pi},\mathbf{A}-\mathbf{B}\Big\rangle_F \\ \text{s.t.} \quad & \|\mathbf{A}_l\|_F=1,\ l=1,...,L,\end{aligned} \tag{4.9}$$

where $\mathbf{\Pi}, \mathbf{\Lambda}$ are Lagrangian multipliers, and μ, ρ are penalty factors to control the convergence behavior, and $< \cdot, \cdot >_F$ is a Frobenius product of two matrices.

Particularly, we utilize the Alternating Direction Method of Multipliers (ADMM) to optimize Eq(4.9). ADMM decomposes an objective into several sub-problems, and iteratively solves them till convergence occurs [8]. We detail each of the sub-problem as follows:

Sub-problem M

$$\begin{aligned} \mathbf{M}^* &= \operatorname{argmin} \mathcal{L}(\mathbf{M}; \mathbf{Z}, \mathbf{B}, \mathbf{A}, \mathbf{T}, \mathbf{\Lambda}, \mathbf{\Pi}) \\ &= \operatorname{argmin} \lambda \sum_{f,l} \|\mathbf{M}_{fl}\|_2 + \frac{\mu}{2} \Big\| \mathbf{M} - \mathbf{Z} \Big\|_F^2 + \Big\langle \mathbf{\Lambda}, \mathbf{M} - \mathbf{Z} \Big\rangle_F \end{aligned} \tag{4.10}$$

Following [50], each $\mathbf{M}_{fl}$ can be computed by using soft-thresholding:

$$\mathbf{M}^*_{fl} = \mathcal{D}_{\lambda/\mu} \left(\mathbf{Z}_{fl} - \frac{1}{\mu} \mathbf{\Lambda}_{fl} \right) \tag{4.11}$$

Sub-problem Z

$$\begin{aligned} \mathbf{Z}^* =& \operatorname{argmin} \quad \mathcal{L}(\mathbf{Z}; \mathbf{M}, \mathbf{B}, \mathbf{A}, \mathbf{T}, \mathbf{\Lambda}, \mathbf{\Pi}) \\ =& \operatorname{argmin} \quad \frac{1}{2} \Big\| \mathbf{\Gamma} \odot \Big(\mathbf{Z}\mathbf{B} + \mathbf{T} \Big) - \mathbf{W} \Big\|_F^2 + \frac{\mu}{2} \Big\| \mathbf{M} - \mathbf{Z} \Big\|_F^2 + \Big\langle \mathbf{\Lambda}, \mathbf{M} - \mathbf{Z} \Big\rangle_F \end{aligned} \tag{4.12}$$

$\mathbf{Z}^*$ is updated iteratively by gradient descent several times, where the gradient is $\Big(\mathbf{\Gamma} \odot \mathbf{\Gamma} \odot \big(\mathbf{Z}\mathbf{B} + \mathbf{T} \big) - \mathbf{W} \Big) \mathbf{B}^T - \mathbf{\Lambda} + \mu(\mathbf{Z} - \mathbf{M})$. If $\mathbf{\Gamma}$ is all ones (all key points are visible), we can compute $\mathbf{Z}^*$ easily by pseudo-inverse:

$$\mathbf{Z}^* = \Big(\mathbf{B}\mathbf{B}^T + \mu \mathbf{I} \Big)^\dagger \Big((\mathbf{W} - \mathbf{T}) \mathbf{B}^T + \mathbf{\Lambda} + \mu \mathbf{M} \Big) \tag{4.13}$$

Sub-problem B

$$\begin{aligned} \mathbf{B}^* =& \operatorname{argmin} \quad \mathcal{L}(\mathbf{B}; \mathbf{M}, \mathbf{Z}, \mathbf{A}, \mathbf{T}, \mathbf{\Lambda}, \mathbf{\Pi}) \\ =& \operatorname{argmin} \quad \frac{1}{2} \Big\| \mathbf{\Gamma} \odot \big(\mathbf{Z}\mathbf{B} + \mathbf{T} \big) - \mathbf{W} \Big\|_F^2 \\ & \qquad + \Big\langle \mathbf{\Pi}, \mathbf{A} - \mathbf{B} \Big\rangle_F + \frac{\rho}{2} \Big\| \mathbf{A} - \mathbf{B} \Big\|_F^2 \end{aligned} \tag{4.14}$$

Each column of B, corresponded to each key point p, can be independently optimized as:

$$\begin{aligned} \mathbf{B}^*_p = \operatorname{argmin} \quad & \frac{1}{2} \Big\| \operatorname{diag} \big(\mathbf{\Gamma}_p \big) \mathbf{Z}\mathbf{B}_p + \mathbf{\Gamma}_p \odot \mathbf{T}_p - \mathbf{W}_p \Big\|_2^2 \\ & + \Big\langle \mathbf{\Pi}_p, \mathbf{A}_p - \mathbf{B}_p \Big\rangle_F + \frac{\rho}{2} \Big\| \mathbf{A}_p - \mathbf{B}_p \Big\|_2^2 \end{aligned} \tag{4.15}$$

We utilized a gradient descent solver to optimize Eq (4.15) when ρ is small (Eq (4.15) is poorly conditioned). Once ρ becomes big enough, we solve $\mathbf{B}_p$ directly using a least square solver. If all entries of $\mathbf{\Gamma}$ are one, i.e.. all key points are visible, $\mathbf{B}^*$ can efficiently computed by:

$$\mathbf{B}^* = \Big(\mathbf{Z}^T\mathbf{Z} + \rho\mathbf{I}\Big)^\dagger \Big(\mathbf{Z}^T(\mathbf{W} - \mathbf{T}) + \mathbf{\Pi} + \rho\mathbf{A}\Big) \tag{4.16}$$

Sub-problem A

$$\begin{aligned} \mathbf{A}^* = &\quad \operatorname{argmin} \mathcal{L}(\mathbf{A}; \mathbf{M}, \mathbf{Z}, \mathbf{B}, \mathbf{T}, \mathbf{\Lambda}, \mathbf{\Pi}) \\ = &\quad \operatorname{argmin} \Big\langle \mathbf{\Pi}, \mathbf{A} - \mathbf{B} \Big\rangle_F + \frac{\rho}{2}\Big\|\mathbf{A} - \mathbf{B}\Big\|_F^2 \\ \text{s.t.} &\quad \big\|\mathbf{A}_l\big\|_F = 1,\ l = 1, ..., L. \end{aligned} \tag{4.17}$$

The optimal solution for Eq (4.17) can be obtained as [10],

$$\mathbf{A}_l^* = \frac{\mathbf{B}_l - 1/\rho\mathbf{\Pi}_l}{\big\|\mathbf{B}_l - 1/\rho\mathbf{\Pi}_l\big\|_F} \tag{4.18}$$

Sub-problem T

$$\mathbf{T}^* = \operatorname{argmin} \mathcal{L}(\mathbf{T}; \mathbf{M}, \mathbf{Z}, \mathbf{B}, \mathbf{A}, \mathbf{\Lambda}, \mathbf{\Pi}) = \operatorname{argmin} \frac{1}{2}\Big\|\mathbf{\Gamma} \odot \big(\mathbf{Z}\mathbf{B} + \mathbf{T}\big) - \mathbf{W}\Big\|_F^2. \tag{4.19}$$

Since all columns of $\mathbf{T} \in \mathbb{R}^{2F \times P}$, $\boldsymbol{\tau}$'s, are identical, we compute a $\boldsymbol{\tau} \in \mathbb{R}^{2F \times 1}$ by minimizing the above objective:

$$\boldsymbol{\tau}^* = \quad \operatorname{argmin} \frac{1}{2}\sum_{p=1}^{P}\Bigg\|\mathbf{\Gamma}_p \odot \big(\mathbf{Z}\mathbf{B}_p + \boldsymbol{\tau}\big) - \mathbf{W}_p\Bigg\|_2^2, \tag{4.20}$$

and optimal $\boldsymbol{\tau}$ is computed by:

$$\boldsymbol{\tau}^* = \Big(\sum_{p=1}^{P}\mathbf{W}_p - \sum_{p=1}^{P}\mathbf{\Gamma}_p \odot \mathbf{\Gamma}_p \odot \mathbf{Z}\mathbf{B}_p\Big) \oslash \Big(\sum_{p=1}^{P}\mathbf{\Gamma}_p \odot \mathbf{\Gamma}_p\Big) \tag{4.21}$$

where $\oslash$ denotes the element-wise division.

Lagrange Multiplier Update

The lagrange multipliers $\mathbf{\Pi}, \mathbf{\Lambda}$ at each iteration are updated as,

$$\begin{aligned} \mathbf{\Lambda}^{[i+1]} = &\quad \mathbf{\Lambda}^{[i]} + \mu\big(\mathbf{M}^{[i+1]} - \mathbf{Z}^{[i+1]}\big) \\ \mathbf{\Pi}^{[i+1]} = &\quad \mathbf{\Pi}^{[i]} + \rho\big(\mathbf{A}^{[i+1]} - \mathbf{B}^{[i+1]}\big) \end{aligned} \tag{4.22}$$

Penalty Update

Superlinear convergence of ADMM may be achieved by $\mu, \rho \rightarrow \infty$. In practice, we limit the value of μ, ρ to avoid poor conditions and numerical errors. Specifically, we adopt the following update strategy:

$$\begin{aligned} \mu^{[i+1]} &= \min(\mu_{max}, \beta_1 \mu^{[i]}) \\ \rho^{[i+1]} &= \min(\rho_{max}, \beta_2 \rho^{[i]}) \end{aligned} \tag{4.23}$$

We found experimentally $\mu^{[0]} = 10^{-2}$, $\rho^{[0]} = 10^{-1}$, $\beta_1(\beta_2) = 1.1$, and $\mu_{max}(\rho_{max}) = 10^5$ to perform well.

4.3 Experiments

4.3.1 Evaluation setup

This project compares the proposed method against the most notable NRSfM algorithms: the Tomasi-Kanade factorization [38], and the state-of-the-art Dai et al.'s prior-less NRSfM method [14], in terms of reprojection and reconstruction errors. The reprojection error measures the accuracy of reprojected key points: $\frac{1}{F}\sum_{i=1}^{F} \|\mathbf{W}_i - \hat{\mathbf{W}}_i\|_F$. The reconstruction error, on the other hand, evaluates the quality of estimated 3D shapes: $\frac{1}{F}\sum_{i=1}^{F} \min_\kappa \|\mathbf{S}_i - \kappa\hat{\mathbf{S}}_i\|_F$. κ (scalar) handles the scale ambiguity in camera projection.

Extensive experiments are conducted to evaluate the performance of our framework using both synthetic and natural images. For the synthetic images, we downloaded 70 CAD models of aeroplane category from the Sketchup 3D warehouse [1], and manually annotated their 3D key points. The synthetic images are simply generated by projecting random poses of these 3D models under a weak-perspective camera into the image plane. The PASCAL3D+ dataset [46] is used for the natural image experiment, which consists of 12 object categories, and each category comes with a set of annotated 3D CAD models and corresponding natural images. We utilize most of the images from all categories, except those displaying highly occluded objects. More details of the PASCAL3D+ dataset can be found in [46].

The main differences between synthetic and PASCAL3D+ images come from the camera projection and object occlusion. We utilize random *weak*-perspective projection to generate the synthetic images of the aeroplane dataset, which follows the weak-projection assumption in this paper, whilst the camera projection in the PASCAL3D+ is perspective. Moreover, all key points

[1] https://3dwarehouse.sketchup.com/

in synthetic images are visible, while some key points in the PASCAL3D+ may be occluded by the object itself or other objects.

4.3.2 3D reconstruction from synthetic images

The first experiment evaluates the performance of the proposed method on synthetic images, comparing it with the Tomasi-Kanade factorization [38] and Dai et al.'s prior-less NRS*f*M approaches [14]. The synthetic images are randomly generated from all 3D CADs of the aeroplane dataset under weak perspective projection, and these approaches are applied to reconstruct the 3D shape of each image. The predicted shapes, then, are projected into the 2D plane to compute the key points reprojection error. The result of this experiment is shown in Fig. 4.1 (top), demonstrating the superior performance of our method to the other approaches. This evaluation shows that the 3D shapes reconstructed by the proposed S*f*C not only represent the actual geometry of the objects in 3D space, but also preserve the objects' spatial configuration when projected in the image plane. The result also verifies the sensitivity of the low-rank factorization NRS*f*M algorithm, e.g.Dai et al.'s method in the real-world uncontrolled circumstances, when the shape of an object can not be modeled by very few shape bases [43].

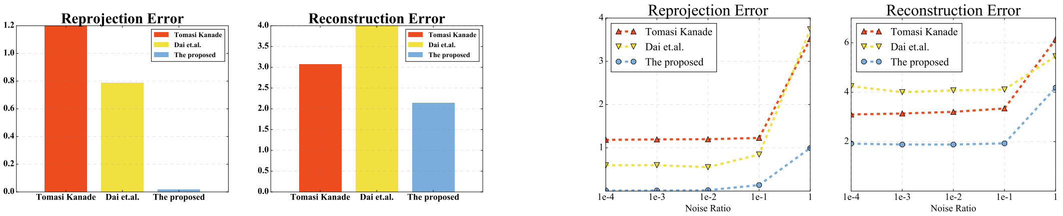


Figure 4.1: Comparing our method with the Tomasi-Kanade [38] and Dai et al. [14] methods using the synthetic images. (left) The reconstruction and reprojection errors. (right) Noise performance.

4.3.3 Noise performance

To analyse the robustness of our method against inaccurate key point detection, which is inevitable in real-world circumstances, we repeat the first experiment (using synthetic aeroplane images) with different levels of Gaussian noise added to the ground truth 2D locations. The average reconstruction and reprojection errors of ten random runs for each noise ratio is reported in Fig. 4.1 (bottom), showing that, compared to the other methods, the S*f*C method is more robust against inaccurate key point detections.

4.3.4 3D reconstruction of PASCAL3D+ dataset

To evaluate the performance of our framework over perspective projection and missing key points, we apply the proposed S*f*C approach to reconstruct 3D shapes of the PASCAL3D+ natural images. There is no additional shape and camera motion assumption given in this experiment, and images of all 12 object categories are taken under uncontrolled real-world circumstances. All images and their corresponding ground truth 3D CAD models are represented by a set of 2D and 3D annotated key points, respectively, which, together with the predicted 3D structures and their reprojected 2D key points, will be used to compute the reconstruction and reprojection errors. Since the Tomasi-Kanade factorization and Dai et al.'s method are not capable of handling occluded objects, we utilize the non-convex matrix completion via iterated soft thresholding [31] to predict the missing points for these approaches. This experiment is conducted over two different settings. In the first setting, we use the ground truth key points of each image provided by the PASCAL3D+. In the other setting, however, we adapt the SDM [48] approach for key point detection, and the predicted points are used for 3D reconstruction.

Using ground truth key points

The reprojection and reconstruction errors for each object category are summarized in Table 4.1 and showed by Fig. 4.2, where our approach outperforms the competitors and achieves the lowest reconstruction and reprojection error for each object category.

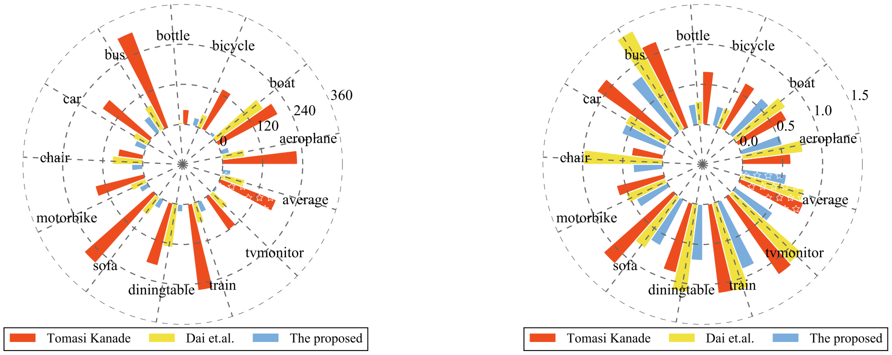


Figure 4.2: The reprojection (left) and reconstruction (right) performance of the proposed method, the Tomasi-Kanade factorization [38] and Dai et al.'s method [14] on natural images (the PASCAL3D+ dataset) with ground truth key points.

category	key points	Reprojection Error			Reconstruction Error		
		Tomasi Kanade	Dai et al.	Our method	Tomasi Kanade	Dai et al.	Our Method
aeroplane	GT	224.3925	67.7078	**24.5695**	0.6035	0.7631	**0.5257**
	detected	364.7172	282.5179	**251.0064**	0.7986	0.7465	**0.6223**
boat	GT	202.9794	174.2009	**11.1862**	0.6892	0.7609	**0.6061**
	detected	150.7320	171.5790	**133.1670**	0.7844	0.8531	**0.7497**
bicycle	GT	135.9651	41.8621	**24.7112**	0.6490	0.2568	**0.2495**
	detected	295.2249	223.5721	**207.6959**	0.7327	0.6695	**0.6351**
bottle	GT	44.4231	6.4836	**2.8315**	0.6609	0.2865	**0.2590**
	detected	108.4824	**68.6833**	69.7238	0.7087	0.4220	**0.3812**
bus	GT	304.8072	82.1719	**56.0355**	1.1427	1.3839	**0.8396**
	detected	564.3329	311.0550	**264.9117**	1.4164	1.3924	**1.1562**
car	GT	173.6506	49.5333	**35.4720**	1.1062	0.5943	**0.5808**
	detected	265.4429	173.8730	**138.6603**	0.9959	0.9636	**0.8242**
chair	GT	75.9437	91.5107	**33.0905**	0.3958	0.9887	**0.3671**
	detected	194.7178	136.9023	**117.6726**	1.0985	1.0511	**0.9338**
motorbike	GT	150.6358	48.3516	**27.1717**	0.6096	0.5252	**0.4344**
	detected	464.8820	280.3500	**264.5549**	0.7333	0.7185	**0.6887**
sofa	GT	274.9890	64.2714	**30.0575**	1.1561	0.7727	**0.6438**
	detected	416.9723	253.0140	**196.6783**	1.1198	1.1617	**1.0126**
diningtable	GT	192.5072	130.5157	**21.9391**	0.8924	1.1084	**0.6982**
	detected	258.3700	110.2296	**103.4765**	1.2404	1.1124	**1.0107**
train	GT	260.5996	61.7900	**34.2347**	1.1215	1.1316	**0.8957**
	detected	457.0754	296.3881	**213.2750**	1.2568	1.2728	**1.1799**
tvmonitor	GT	119.8794	59.2110	**6.6706**	1.1740	1.1454	**0.5653**
	detected	277.1977	100.6167	**60.0780**	0.9307	1.0412	**0.7516**
average	GT	180.0644	73.1342	**25.6642**	0.8501	0.8098	**0.5554**
	detected	318.1790	200.7318	**168.4084**	0.9847	0.9504	**0.8288**

Table 4.1: Reprojection and Reconstruction errors obtained by the Tomasi Kanade factorization [38], Dai et al.'s method [14], and our method using ground truth key points (GT) and detected key points (detected).

Using predicted key points

We adapt the Supervised Descent Method (SDM) [48], originally proposed for the task of facial landmarks alignment, to detect key points of generic objects within natural images. The main assumption of the SDM is that training samples fall into a Domain of Homogeneous Descent (DHD)[2], due to their limited pose space and appearance variation [49]. This assumption, however, is rarely valid in an object category with large intra-class appearance and poses variations that lie in multiple DHDs. To deal with this situation, we propose to employ a subset of training images with homogeneous gradient directions to train an SDM in an "on-the-fly" manner. Par-

[2] A DHD refers to optimization spaces of a function that share similar directions of gradients.

ticularly, given a test image, we use fc_7 feature from the ConvNet [36] to retrieve its M most similar samples from training images and use them to train an SDM. The training set is generated by adding Gaussian noise to the ground truth locations. After training the SDM regressors, we run them independently from M different initializations (the ground truth landmark locations of the M retrieved samples). This returns M sets of predicted key points, which will be further pruned by the mean-shift algorithm. More details of SDM training/testing can be found in [48].

The results are shown in Fig. 4.3 and Table 4.1. For both settings, using ground truth and predicted key points, our method achieves the best reconstruction and reprojection performance. The results also state that the performance of using ground truth key points is much better than the detected key points. Some qualitative results are shown in Fig. 4.4, illustrating the 3D reconstruction of two instances of each object category using ground truth key points and detected key points respectively. During the experiments, we observed that most of the failure cases are caused by the severe perspective effect (e.g.train), missing key points (e.g.sofa), and inaccurate key point detection (e.g.chair).

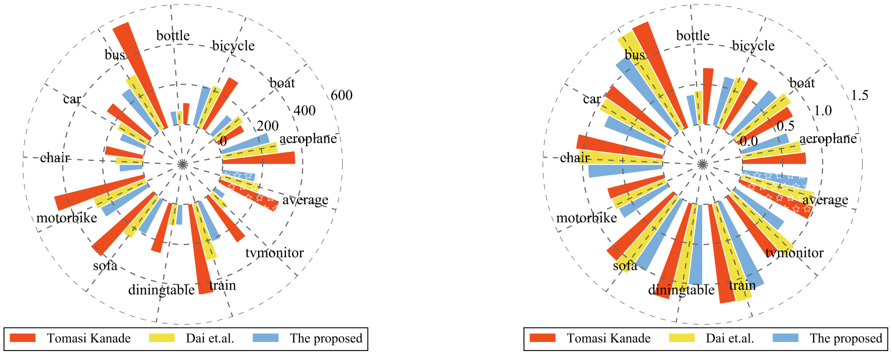


Figure 4.3: The reprojection (left) and reconstruction (right) performance of the proposed method, the Tomasi-Kanade factorization [38] and Dai et al.'s method [14] on natural images (the PASCAL3D+ dataset) with detected key points.

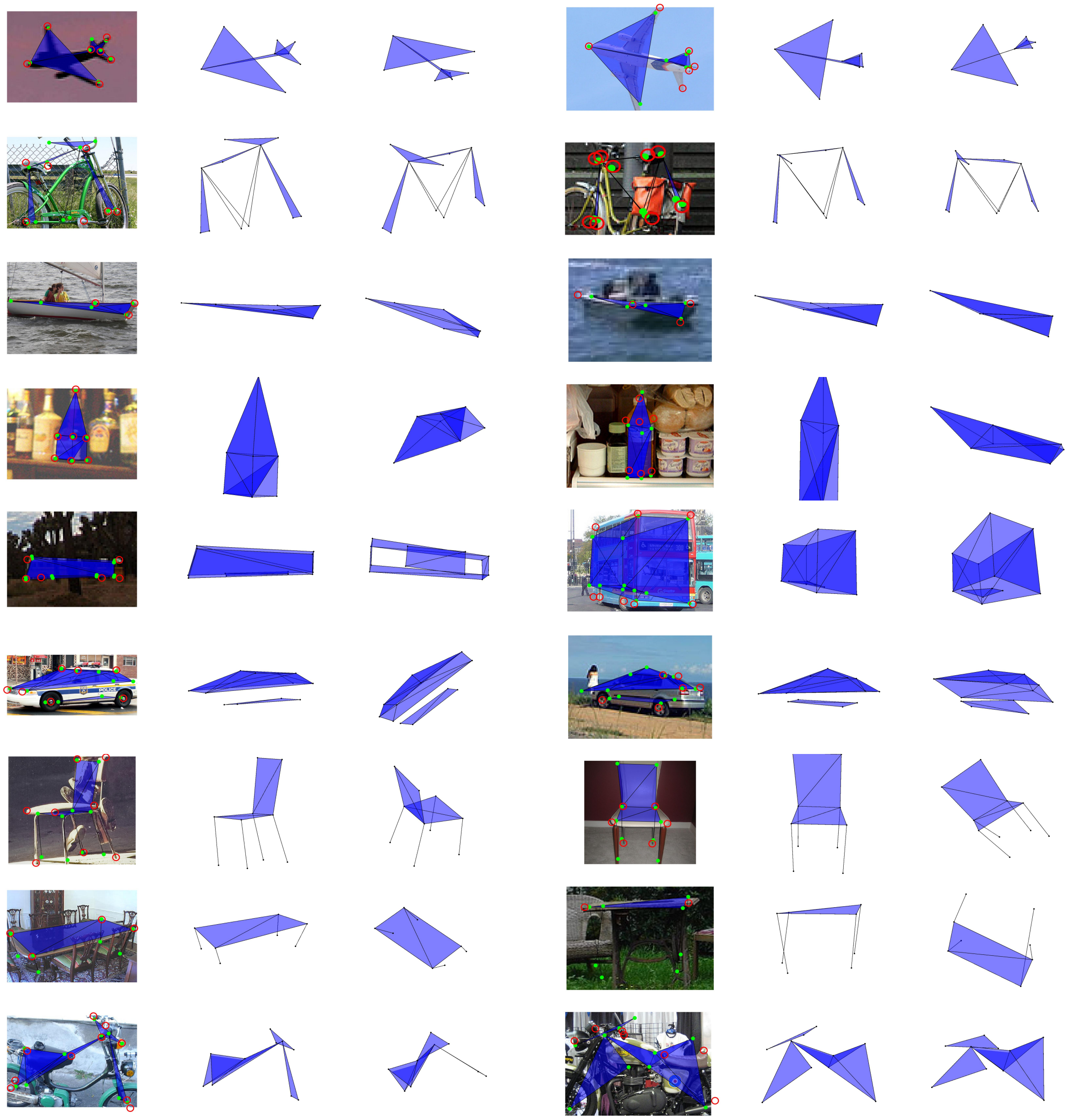


Figure 4.4: Visual evaluation of estimated structures for every category, including aeroplane, bicycle, boat, bottle, bus, car, chair, dining table, motorbike, sofa, train, and T.V. monitor. The first 3 columns use ground truth key points, while the last 3 columns use detected key points. In each triplet columns, the left columns show the images, the projection of estimated 3D shapes, the projection of estimated landmarks (green), and the ground truth landmarks (red, some are missing due to occlusion). The middle ones show the estimated 3D shapes in the same viewpoint as a camera; the right ones show a new viewpoint of the estimated 3D shapes. Two failure cases are shown in red. Best viewed in color.

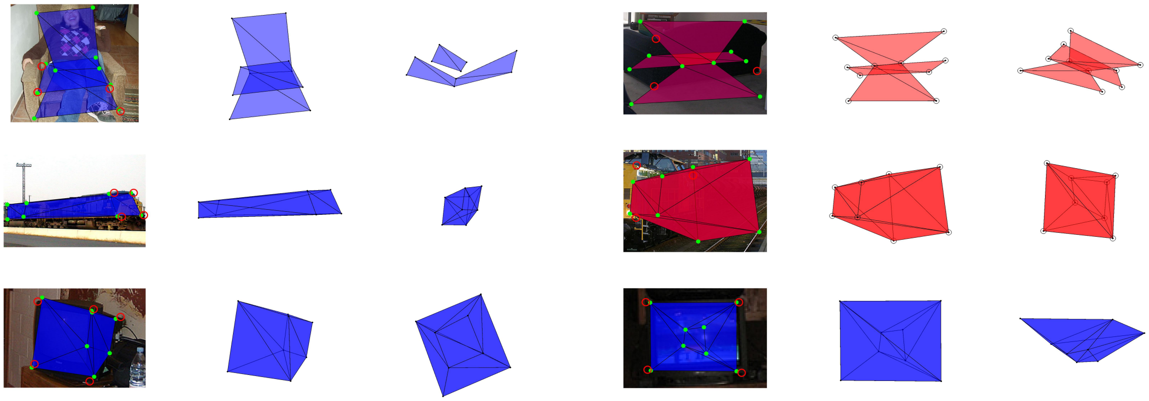


Figure 4.5: Continue Figure 4.4.

Chapter 5

Deep Non-Rigid Structure from Motion

5.1 Problem Formulation

In the context of NRS*f*M, the weak perspective projection model is a reasonable assumption since the many of objects we deal with in vision applications have a much smaller depth variation compared to their distance from the camera. We shall start from the orthogonal projection model in this section and then generalize to weak perspective projection in Section 5.4. Under orthogonal projection, NRS*f*M deals with the problem of factorizing a 2D projection matrix $\mathbf{W} \in \mathbb{R}^{P \times 2}$, given P points, as the product of a 3D shape matrix $\mathbf{S} \in \mathbb{R}^{P \times 3}$ and a camera matrix $\mathbf{M} \in \mathbb{R}^{3 \times 2}$. Formally,

$$\mathbf{W} = \mathbf{S}\mathbf{M}, \tag{5.1}$$

$$\mathbf{W} = \begin{bmatrix} u_1 & v_1 \\ u_2 & v_2 \\ \vdots & \vdots \\ u_P & v_P \end{bmatrix}, \ \mathbf{S} = \begin{bmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ \vdots & \vdots & \vdots \\ x_P & y_P & z_P \end{bmatrix}, \ \mathbf{M}^T\mathbf{M} = \mathbf{I}_2, \tag{5.2}$$

where (u_i, v_i) and (x_i, y_i, z_i) are the image and world coordinates of the i-th point respectively. The goal of NRS*f*M is to recover simultaneously the shape $\mathbf{S}$ and the camera $\mathbf{M}$ for each projection $\mathbf{W}$ in a given set $\mathbb{W}$ of 2D landmarks. In a general NRS*f*M including S*f*C, this set $\mathbb{W}$ could contain deformations of a non-rigid object or various instances from an object category.

5.2 Modeling via hierarchical sparse coding

Kong et al. [26] argued that an effective solution for NRSfM can be found by assuming the vectorization of $\mathbf{S}$ can be represented by a dictionary sparsely:

$$\mathbf{s} = \mathbf{D}\boldsymbol{\psi}, \quad \|\boldsymbol{\psi}\|_0 < \lambda \ . \tag{5.3}$$

This paper introduces additional layers and therefore a hierarchical sparse model is proposed:

$$\begin{aligned} \mathbf{s} &= \mathbf{D}_1\boldsymbol{\psi}_1, \quad \|\boldsymbol{\psi}_1\|_0 < \lambda_1, \\ \boldsymbol{\psi}_1 &= \mathbf{D}_2\boldsymbol{\psi}_2, \quad \|\boldsymbol{\psi}_2\|_0 < \lambda_2, \\ &\vdots \quad , \quad \vdots \\ \boldsymbol{\psi}_{N-1} &= \mathbf{D}_N\boldsymbol{\psi}_N, \quad \|\boldsymbol{\psi}_N\|_0 < \lambda_N, \end{aligned} \tag{5.4}$$

where $\mathbf{D}_1 \in \mathbb{R}^{3P\times K_1}, \mathbf{D}_2 \in \mathbb{R}^{K_1\times K_2}, \dots, \mathbf{D}_N \in \mathbb{R}^{K_{N-1}\times K_N}$ are hierarchical dictionaries and $\boldsymbol{\psi}_1 \in \mathbb{R}^{K_1}$, $\boldsymbol{\psi}_2 \in \mathbb{R}^{K_2}$, $\dots$, $\boldsymbol{\psi}_N \in \mathbb{R}^{K_N}$ are hierarchical sparse codes. In this prior, each non-rigid shape is represented by a sequence of dictionaries and corresponding non-negative sparse codes hierarchically. Each sparse code is determined by its lower-level neighbor and affects the next-level. The additional layers introduced by this hierarchy increase the number of variables, and thus increase the degree of freedom of the system. However, these additional layers actually result in a more constrained and thus stable sparse code recovery process.

Sparse code recovery algorithms in general attempt to solve two problems: 1) select the best subspace and 2) estimate the closest representation within the subspace. These two problems could be solved simultaneously or alternatively, but the quality of recovered sparse code highly relies on the former. If the desired subspace is given from oracle, then the sparse coding problem degenerates to a linear system. However, without knowing the size of the desired subspace, the number of valid subspaces in (5.3) is combinatorial to the number of dictionary atoms K i.e. $\sum_{n=1}^{K}\binom{K}{n}$. Selecting the best subspace out of such large number of candidates is considerably difficult, especially when using over-complete dictionaries. This reveals the conflict between the quality of sparse code recovery and the representing capacity of the dictionary, and further explains the sensitivity of [26] to non-compressible sequences.

The additional layers introduced in this paper alleviate the dilemma. In (5.4), the sparse code $\boldsymbol{\psi}_1$ is not completely free but represented by the subsequent dictionaries. Therefore, the number of subspaces is not combinatorial to K_1 but controlled by the subsequent dictionaries $\{\mathbf{D}_i\}_{i=2}^{N}$. If the subsequent dictionaries are learned properly, they could serve as a filter so that only functional subspaces remain and redundant ones are removed. This directly breaks the combinatorial explosion of the number of subspaces and consequently maintains the robustness of sparse code recovery. Based on this observation, we are able to utilize substantially over-complete dictionaries to model a highly deformable object from a large scale image collection with no worries about reconstructability and robustness.

5.2.1 Hierarchical Block Sparse Coding

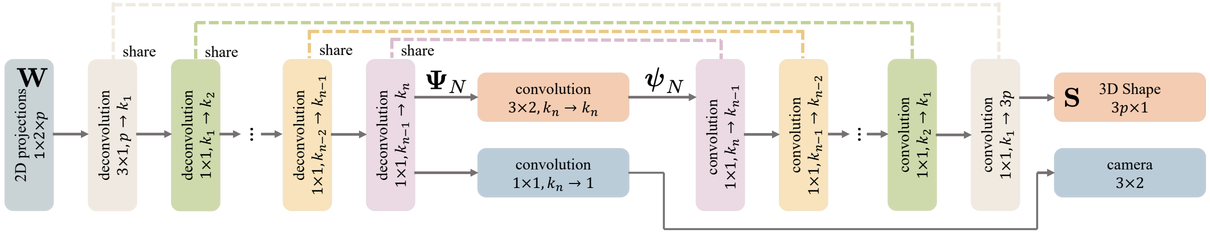
Given the proposed hierarchical sparse coding model, shown in (5.4), we now build a conduit from the 2D correspondences $\mathbf{W}$ to the proposed shape code $\{\boldsymbol{\psi}_i\}_{i=1}^{k}$. Since $\mathrm{s} \in \mathbb{R}^{3P}$ in (5.4) is the vectorization of $\mathbf{S} \in \mathbb{R}^{P\times 3}$, it can be well modeled via i.e. $\mathbf{S} = \mathbf{D}_1^{\sharp}(\boldsymbol{\psi}_1 \otimes \mathbf{I}_3)$ where $\otimes$ is the Kronecker product and $\mathbf{D}_1^{\sharp} \in \mathbb{R}^{P\times 3K_1}$ is a reshape of $\mathbf{D}_1 \in \mathbb{R}^{3P\times K_1}$ [14]. It is known that $\mathbf{AB} \otimes \mathbf{I} = (\mathbf{A} \otimes \mathbf{I})(\mathbf{B} \otimes \mathbf{I})$ given two matrices $\mathbf{A}, \mathbf{B}$, and identity matrix $\mathbf{I}$. Based on this lemma, we can derive that

$$\begin{aligned} \mathbf{S} = \mathbf{D}_1^{\sharp}(\boldsymbol{\psi}_1 \otimes \mathbf{I}_3), &\quad \|\boldsymbol{\psi}_1\|_0 < \lambda_1, \\ \boldsymbol{\psi}_1 \otimes \mathbf{I}_3 = (\mathbf{D}_2 \otimes \mathbf{I}_3)(\boldsymbol{\psi}_2 \otimes \mathbf{I}_3), &\quad \|\boldsymbol{\psi}_2\|_0 < \lambda_2, \\ \vdots \quad , &\quad \vdots \\ \boldsymbol{\psi}_{N-1} \otimes \mathbf{I}_3 = (\mathbf{D}_N \otimes \mathbf{I}_3)(\boldsymbol{\psi}_N \otimes \mathbf{I}_3), &\quad \|\boldsymbol{\psi}_N\|_0 < \lambda_N. \end{aligned} \tag{5.5}$$

Further, from (5.1), by right multiplying the camera matrix $\mathbf{M} \in \mathbb{R}^{3\times 2}$ to the both sides of (5.5) and denote $\boldsymbol{\Psi}_i = \boldsymbol{\psi}_i \otimes \mathbf{M}$, we obtain that

$$\begin{aligned} \mathbf{W} = \mathbf{D}_1^{\sharp}\boldsymbol{\Psi}_1, &\quad \|\boldsymbol{\Psi}_1\|_0^{(3\times 2)} < \lambda_1, \\ \boldsymbol{\Psi}_1 = (\mathbf{D}_2 \otimes \mathbf{I}_3)\boldsymbol{\Psi}_2, &\quad \|\boldsymbol{\Psi}_2\|_0^{(3\times 2)} < \lambda_2, \\ \vdots \quad , &\quad \vdots \\ \boldsymbol{\Psi}_{N-1} = (\mathbf{D}_N \otimes \mathbf{I}_3)\boldsymbol{\Psi}_N, &\quad \|\boldsymbol{\Psi}_N\|_0^{(3\times 2)} < \lambda_N, \end{aligned} \tag{5.6}$$

where $\|\cdot\|_0^{(3\times 2)}$ divides the argument matrix into blocks with size 3×2 and counts the number of active blocks. Since $\boldsymbol{\psi}_i$ has active elements less than λ_i, $\boldsymbol{\Psi}_i$ has active blocks less than λ_i, that is $\boldsymbol{\Psi}_i$ is block sparse. This derivation demonstrates that if the shape vector s satisfies the hierarchical sparse coding prior described by (5.4), then its 2D projection $\mathbf{W}$ must be in the format of hierarchical *block* sparse coding described by (5.6). We hereby interpret NRS*f*M as a hierarchical *block* sparse dictionary learning problem, i.e. factorizing $\mathbf{W}$ as products of hierarchical dictionaries $\{\mathbf{D}_i\}_{i=1}^{N}$ and block sparse coefficients $\{\boldsymbol{\Psi}_i\}_{i=1}^{N}$.



W
2D projections 1×2×p
share
deconvolution 3×1, $p \rightarrow k_1$
share
deconvolution 1×1, $k_1 \rightarrow k_2$
share
deconvolution 1×1, $k_{n-2} \rightarrow k_{n-1}$
share
deconvolution 1×1, $k_{n-1} \rightarrow k_n$
$\mathbf{\Psi}_N$
convolution 3×2, $k_n \rightarrow k_n$
convolution 1×1, $k_n \rightarrow 1$
$\boldsymbol{\psi}_N$
convolution 1×1, $k_n \rightarrow k_{n-1}$
convolution 1×1, $k_{n-1} \rightarrow k_{n-2}$
convolution 1×1, $k_2 \rightarrow k_1$
convolution 1×1, $k_1 \rightarrow 3p$
S
3D Shape 3p×1
camera 3×2

Figure 5.1: Architecture of our proposed deep NRSfM. The network can be divided into 1) Encoder: from 2D correspondences $\mathbf{W}$ to the hidden block sparse code $\mathbf{\Psi}_N$, 2) Bottleneck: from hidden block sparse code $\mathbf{\Psi}_N$ to hidden regular sparse code $\boldsymbol{\psi}_N$ and camera, 3) Decoder: from hidden regular sparse code $\boldsymbol{\psi}_N$ to 3D reconstructed shape $\mathbf{S}$. The encoder and decoder are intentionally designed to share convolution kernels (i.e. dictionaries) and form a symmetric formulation. The symbol $a \times b, c \rightarrow d$ refers to the convolution layer using kernel size $a \times b$ with c input channels and d output channels.

5.3 Deep Neural Network Solution

Before solving the hierarchical block sparse coding problem in (5.6), we first consider a single-layer problem:

$$\min_{\mathbf{Z}} \|\mathbf{X} - \mathbf{D}\mathbf{Z}\|_F^2 \quad \text{s.t.} \quad \|\mathbf{Z}\|_0^{(3\times 2)} < \lambda. \tag{5.7}$$

Inspired by ISTA, we propose to solve this problem by iteratively executing the following two steps:

$$\mathbf{V} = \mathbf{Z}^{[i]} - \alpha \mathbf{D}^T(\mathbf{D}\mathbf{Z}^{[i]} - \mathbf{X}), \tag{5.8}$$

$$\mathbf{Z}^{[i+1]} = \operatorname*{argmin}_{\mathbf{U}} \frac{1}{2}\|\mathbf{U} - \mathbf{V}\|_F^2 + \tau \|\mathbf{U}\|_{F1}^{(3\times 2)}, \tag{5.9}$$

where $\|\cdot\|_{F1}^{(3\times 2)}$ is defined as the summation of the Frobenius norm of each 3×2 block, serving as a convex relaxation of the block sparsity constraint. Recall the regular sparse situation in Section 2.6. Analogous to (2.89), we use an approximate solution to (5.9) for computational efficiency, i.e.

$$\mathbf{Z}^{[i+1]} = \eta(\mathbf{V}; \mathbf{b} \otimes \mathbf{1}_{3\times 2}), \tag{5.10}$$

where η represents a element-wise soft-thresholding operation defined in (2.90), $\mathbf{1}_{3\times 2}$ denotes a 3-by-2 matrix filled with one and $\mathbf{b}$ is a vector that controls the trust region for each block. Based on this approximation, a single-iteration block ISTA with step size $\alpha = 1$ can be represented by :

$$\mathbf{Z} = \eta(\mathbf{D}^T\mathbf{X}; \mathbf{b} \otimes \mathbf{1}_{3\times 2}), \tag{5.11}$$

5.3.1 Encoder

Recall from Section 2.6 that the feed-forward pass through a deep neural network can be considered as a sequence of single ISTA iterations and thus provides an approximate recovery of hierarchical sparse codes. We follow the same scheme – sequentially using single-iteration block ISTA – to solve the hierarchical block sparse coding problem (5.6) i.e.

$$\begin{aligned}\mathbf{\Psi}_1 &= \eta((\mathbf{D}_1^\sharp)^T\mathbf{W}; \mathbf{b}_1 \otimes \mathbf{1}_{3\times2}),\\ \mathbf{\Psi}_2 &= \eta((\mathbf{D}_2 \otimes \mathbf{I}_3)^T\mathbf{\Psi}_1; \mathbf{b}_2 \otimes \mathbf{1}_{3\times2}),\\ &\vdots\\ \mathbf{\Psi}_N &= \eta((\mathbf{D}_N \otimes \mathbf{I}_3)^T\mathbf{\Psi}_{N-1}; \mathbf{b}_N \otimes \mathbf{1}_{3\times2}),\end{aligned} \tag{5.12}$$

where $\{\mathbf{b}_i\}_{i=1}^N$ are learnable parameters, controlling the block sparsity. This formula composes the encoder of our proposed deep neural network.

It is worth mentioning that setting $\{\mathbf{b}_i\}_{i=1}^N$ as learnable parameters is crucial because in previous NRS*f*M algorithms – low-rank [14], union-of-subspaces [52], or block-sparsity [26] priors – the weight associated with shape regularization (e.g.low-rank or sparsity) is determined through a cumbersome and slow grid-search process. In our approach, this weighting is learned simultaneously with all other parameters, removing the need for irksome cross-validation.

5.3.2 Code and Camera Recovery

Recall that in Section 5.2.1, we define $\mathbf{\Psi} = \boldsymbol{\psi} \otimes \mathbf{M}$. By denoting the k-th block in $\mathbf{\Psi}_N$ as $\mathbf{\Psi}_N^k$ and the k-th element in $\boldsymbol{\psi}_N$ as $\boldsymbol{\psi}_N^k$. we have

$$\mathbf{\Psi}_N^k = \boldsymbol{\psi}_N^k\mathbf{M}. \tag{5.13}$$

Now, we want to estimate the regular sparse hidden code $\boldsymbol{\psi}_N$ and camera $\mathbf{M}$ given $\mathbf{\Psi}_N$. It is obvious that if one of them is known beforehand, then the other one can be solved easily. For example, if $\mathbf{M}$ is known, then $\boldsymbol{\psi}_N^k$ can be estimated by

$$\boldsymbol{\psi}_N^k = \frac{1}{6}\sum_{i=1}^{3}\sum_{j=1}^{2}\frac{[\mathbf{\Psi}_N^k]_{ij}}{[\mathbf{M}]_{ij}} = \sum_{i=1}^{3}\sum_{j=1}^{2}\frac{1}{6[\mathbf{M}]_{ij}}[\mathbf{\Psi}_N^k]_{ij}, \tag{5.14}$$

where $[\cdot]_{ij}$ denotes the ij-th element in the argument matrix. Note that actually a single element in camera M and its correspondence in $\mathbf{\Psi}_N$ are sufficient to estimate the scaler $\boldsymbol{\psi}_N^k$, but, for robust estimation, an average over all elements (3×2 block results in totally 6 elements) is utilized here. Further, if $\boldsymbol{\psi}_N$ is known, then $\mathbf{M}$ can be estimated by

$$\mathbf{M} = \frac{1}{K_N}\sum_{k=1}^{K_N}\frac{\mathbf{\Psi}_N^k}{\boldsymbol{\psi}_N^k} = \sum_{k=1}^{K_N}\frac{1}{K_N\boldsymbol{\psi}_N^k}\mathbf{\Psi}_N^k. \tag{5.15}$$

Note that a single element in $\boldsymbol{\psi}_N$ and a corresponding block in $\boldsymbol{\Psi}_N$ is again sufficient to estimate $\mathbf{M}$ but, for robustness, we utilize an average across all blocks.

In the literature of the field [6, 14, 26], these two coupled variables are mainly solved by a carefully designed algorithm that utilizes the orthonormal constraint to solve the camera first and then the sparse hidden code. However, this heuristic is quite fragile and it is even worse when the estimation of $\boldsymbol{\Psi}_N$ is bothered by noise. Further, it has difficulty deciding the sign ambiguity of each sparse code. In this paper, we propose a novel relaxation, decoupling equations (5.14) and (5.15) by introducing two learnable parameters β and γ, specifically,

$$\boldsymbol{\psi}_N^k = \sum_{i=1}^{3}\sum_{j=1}^{2} \beta_{ij}[\boldsymbol{\Psi}_N^k]_{ij}, \tag{5.16}$$

$$\mathbf{M} = \sum_{k=1}^{K_N} \gamma_k \boldsymbol{\Psi}_N^k. \tag{5.17}$$

It is clear that $\boldsymbol{\psi}$ and $\mathbf{M}$ are intrinsically linked – but our proposed approach seems to ignore this dependency. We resolve this inconsistency, however, by enforcing an orthonormal constraint for the camera in our loss function shown in Section 5.3.4. This relaxation has the further advantage of eliminating fragile heuristics and giving substantial computational savings. Figure 5.1 represents this process via convolutions for conciseness and descent visualization.

5.3.3 Decoder

Given the sparse hidden code $\boldsymbol{\psi}_N$ and hierarchical dictionaries $\{\mathbf{D}_i\}_{i=1}^N$, the 3D shape vector s could be recovered via (5.4). In practice, instead of forming a purely linear decoder, we preserve soft-thresholding in each layer. This non-linear decoder is expected to further enforce sparsity and improve robustness. Formally,

$$\begin{aligned} \boldsymbol{\psi}_{N-1} &= \eta(\mathbf{D}_N\boldsymbol{\psi}_N; \mathbf{b}'_N), \\ &\vdots \\ \boldsymbol{\psi}_1 &= \eta(\mathbf{D}_2\boldsymbol{\psi}_2; \mathbf{b}'_2), \\ \mathbf{s} &= \mathbf{D}_1^{\sharp}\boldsymbol{\psi}_1. \end{aligned} \tag{5.18}$$

This portion forms the decoder of our deep neural network.

5.3.4 Loss Function

Until now, the 3D shape S is estimated via the proposed encoder and decoder architecture given the hierarchical dictionaries, which is denoted as $\mathcal{S}\big(\mathbf{W}|\{\mathbf{D}_i\}_{i=1}^N\big)$ for simplicity. Further, the camera $\mathbf{M}$ is also estimated via the encoder and a linear combination given the dictionaries, which is denoted as $\mathcal{M}\big(\mathbf{W}|\{\mathbf{D}_i\}_{i=1}^N\big)$. Our loss function is thus defined as

$$\begin{aligned} \min_{\{\mathbf{D}\}_{i=1}^N} \sum_{\mathbf{W}\in\mathbb{W}} &\big\|\mathbf{W} - \mathcal{S}\big(\mathbf{W}|\{\mathbf{D}_i\}_{i=1}^N\big)\mathbf{U}\mathbf{V}\big\|_F \\ \text{s.t. } \mathbf{U}\Sigma\mathbf{V}^T &= \mathcal{M}\big(\mathbf{W}|\{\mathbf{D}_i\}_{i=1}^N\big), \end{aligned} \tag{5.19}$$

which is the summation of reprojection error. To ensure the success of the orthonormal constraint on the camera, we introduce the Singular Value Decomposition (SVD) to hard code the singular value of $\mathbf{M}$ to be exact ones. As mentioned in Section 5.3.2, reprojecting the estimated 3D shape via the estimated camera (i.e. left multiplying $\mathbf{M}$ to $\mathbf{S}$) implicitly re-build the bonds between the camera $\mathbf{M}$ and the sparse hidden code $\boldsymbol{\psi}_N$ (in the form of 3D shape $\mathbf{S}$).

5.3.5 Implementation Issues

The Kronecker product of identity matrix $\mathbf{I}_3$ dramatically increases the time and space complexity of our approach. To eliminate it and make parameter sharing easier in modern deep-learning environments (e.g. TensorFlow, PyTorch), we reshape the filters and features so that the matrix multiplication in each step can be equivalently computed via multi-channel convolution ($*$) and transposed convolution ($*^T$). We first reshape the 2D input correspondences $\mathbf{W}$ into a three-dimensional tensor $\mathsf{w} \in \mathbb{R}^{1\times 2\times P}$, which can be considered in the deep-learning community as a 1×2 image with P channels. Then, we reshape the first dictionary $\mathbf{D}_1^\sharp$ into a four-dimensional tensor $\mathsf{d}_1^\sharp \in \mathbb{R}^{3\times 1\times K_1\times P}$, which can be interpreted as a convolutional kernel in size 3×1 with K_1 input channels and P output channels. Therefore, we have

$$(\mathbf{D}_1^\sharp)^T\mathbf{W} = \mathsf{d}_1^\sharp *^T \mathsf{w}, \tag{5.20}$$

which helps us to maintain a uniform dictionary shape and is consequently easier to share parameters. We then reshape each dictionary $\mathbf{D}_i$ other than the first one into a four-dimensional tensor $\mathsf{d}_i \in \mathbb{R}^{1\times 1\times K_i\times K_{i-1}}$ and the hidden block sparse code $\boldsymbol{\Psi}_i$ into a three-dimensional tensor $\Psi_i \in \mathbb{R}^{3\times 2\times K_i}$. Therefore, we have

$$(\mathbf{D}_{i+1} \otimes \mathbf{I}_3)^T\boldsymbol{\Psi}_i = \mathsf{d}_{i+1} *^T \Psi_i, \tag{5.21}$$

which helps us to eliminate the Kronecker product. Finally, based on the above reshape, the dictionary-code multiplication is simplified as

$$\mathbf{D}_i\boldsymbol{\psi}_i = \mathsf{d}_i * \psi_i. \tag{5.22}$$

As for the architecture design, we only control three hyper parameters: 1) the number of dictionaries N, 2) the number of atoms in the first dictionary K_1, and 3) the number of atoms in the last dictionary K_N. We then linearly sample $K_2, \ldots, K_{N-1}$ between K_1 and K_N. As for training, we implement our neural network via TensorFlow and train it using an Adam optimizer with a learning rate exponentially decayed from 0.001.

5.3.6 Replacing Soft-thresholding via ReLU

Recall in Section 2.6, Papyan et al. replaced the soft-thresholding operator η by ReLU as a result of the non-negativity constraint. Actually, it can easily be demonstrated that a linear (block) sparse model can always be transferred equivalently to a model only using non-negative (block) sparse code i.e.

$$\mathbf{W} = \mathbf{D}\boldsymbol{\Psi} = \begin{bmatrix}\mathbf{D} & -\mathbf{D}\end{bmatrix}\begin{bmatrix}\boldsymbol{\Psi}^+ \\ -\boldsymbol{\Psi}^-\end{bmatrix}. \tag{5.23}$$

where $\boldsymbol{\Psi}^+$ and $\boldsymbol{\Psi}^-$ are positive and negative parts of $\boldsymbol{\Psi}$ respectively and $\boldsymbol{\Psi}^+ + \boldsymbol{\Psi}^- = \boldsymbol{\Psi}$. The concatenation of $\boldsymbol{\Psi}^+$ and $-\boldsymbol{\Psi}^-$ is still block sparse and now becomes non-negative. From this observation, we introduce the non-negativity constraints without the loss of generality and relax the dictionaries so that they are not bothered by mirrored structures. Interestingly, our proposed method on estimating cameras in (5.17) is compatible with the change, i.e.

$$\mathbf{M} = \sum_{k=1}^{K_N} \gamma_k \boldsymbol{\Psi}_N^k = \sum_{k=1}^{K_N} \gamma_k (\boldsymbol{\Psi}_N^k)^+ \sum_{k=1}^{K_N} -\gamma_k (-\boldsymbol{\Psi}_N^k)^-. \tag{5.24}$$

All of these enable us to utilize ReLU to replace the soft-thresholding. ReLU is good because it is closer to deep learning packages while soft-thresholding is more compact in size of parameters. An experiment comparing between soft-thresholding and ReLU is in Section 5.5.3. It is demonstrated that no discernible difference in the accuracy of reconstructions is observed. Therefore, we decide to use ReLU for the remaining sections and experiments, making our approach closer to leading techniques in deep learning and more accessible and approachable to the public.

5.4 Occlusion and Weak Perspective

5.4.1 Occlusion

It is commonly observed in real images that a certain portion of key points are occluded by other objects or the object itself. For example, we typically see two wheels of a sedan instead of four. An often-used strategy is to recover the missing entries in $\mathbf{W}$ by matrix completion before feeding it into the proposed pipeline. A commonly used shape prior for matrix completion is low-rank, even for some union-of-subspaces algorithms [3]. This is problematic.

In this paper, we derive a solution from the ISTA to handle missing entries, which turns out as a quite simple but well-functioning operation. We observe that missing entries break the first layer of encoder but once $\mathbf{\Psi}_1$ is estimated, all other layers can execute with no trouble. Based on this observation, we first introduce a diagonal matrix $\mathbf{\Omega} \in \mathbb{R}^{P\times P}$, whose element on the main diagonal is zero if the corresponding point in $\mathbf{W}$ is missing; otherwise, one and all other elements except diagonal are zeros. With the help of the mask $\mathbf{\Omega}$, the objective function w.r.t the first layer is

$$\min_{\mathbf{\Psi}_1} \|\mathbf{\Omega}(\mathbf{W} - \mathbf{D}_1^\sharp \mathbf{\Psi}_1)\|_F^2 \quad \text{s.t. } \|\mathbf{\Psi}_1\|_0^{(3\times 2)} < \lambda_1. \tag{5.25}$$

Following the same derivation in Section 5.3, a masked ISTA is to iteratively execute the following two steps:

$$\mathbf{V} = \mathbf{\Psi}_1^{[i]} - \alpha(\mathbf{D}_1^\sharp)^T \mathbf{\Omega}^T \mathbf{\Omega}(\mathbf{D}_1^\sharp \mathbf{\Psi}_1^{[i]} - \mathbf{W}), \tag{5.26}$$

$$\mathbf{\Psi}_1^{[i+1]} = \operatorname*{argmin}_{\mathbf{U}} \frac{1}{2}\|\mathbf{U} - \mathbf{V}\|_F^2 + \tau\|\mathbf{U}\|_{F1}^{(3\times 2)}, \tag{5.27}$$

By (5.10), it is implied that the single-iteration block ISTA with mask is

$$\mathbf{\Psi}_1 = \eta\big((\mathbf{D}_1^\sharp)^T \mathbf{\Omega}\mathbf{W} - \mathbf{b} \otimes \mathbf{I}_{3\times 2}\big). \tag{5.28}$$

This is equivalently to set missing entries to zero and then feed into the proposed deep neural network.

5.4.2 Scale and Translation

The main difference between weak perspective and orthogonal projection is additional scale and translation besides rotation. Due to the ambiguity between camera scale and 3D shape size, we do not solve the camera scale explicitly, but consider the scale to be one and reconstruct a scaled 3D shape. To alleviate the effect of the scale on optimization, we normalize the 2D correspondences into a unit bounding box before feeding into the proposed neural network.

Translation is not a problem and can even be eliminated when all points are visible. This is because one can always remove the camera translation by shifting the center of 2D correspondences to the image origin. However, this is not true when some correspondences are missing. Formally, $i \in \Omega$ denotes that the i-th point is visible and (u_i, v_i) is the image coordinate of the i-th point. Shifting the center of all points (where missing entries are set to zero) to the origin remains a translation residual

$$\frac{1}{n}\sum_i \begin{bmatrix} u_i \\ v_i \end{bmatrix} - \frac{1}{n}\sum_{i\in\Omega} \begin{bmatrix} u_i \\ v_i \end{bmatrix} = \frac{1}{n}\sum_{i\notin\Omega} \begin{bmatrix} u_i \\ v_i \end{bmatrix}. \tag{5.29}$$

When key points distribute closely in a cluster and a small portion of them are missing, the residual translation could be treated as some sort of noise perturbation and consequently need no further operation. Otherwise, we need to solve the translation explicitly.

Consider the camera projection with translation $\mathbf{t}$, i.e.

$$\mathbf{W} = \begin{bmatrix} u_1 & v_1 \\ u_2 & v_2 \\ \vdots & \vdots \\ u_P & v_P \end{bmatrix} = \begin{bmatrix} x_1 & y_1 & z_1 & 1 \\ x_2 & y_2 & z_2 & 1 \\ \vdots & \vdots & \vdots & \vdots \\ x_P & y_P & z_P & 1 \end{bmatrix} \begin{bmatrix} \mathbf{M} \\ \mathbf{t}^T \end{bmatrix}. \tag{5.30}$$

We could introduce an auxiliary variable ϵ

$$\mathbf{W} = \begin{bmatrix} x_1 & y_1 & z_1 & \epsilon \\ x_2 & y_2 & z_2 & \epsilon \\ \vdots & \vdots & \vdots & \vdots \\ x_P & y_P & z_P & \epsilon \end{bmatrix} \begin{bmatrix} \mathbf{M} \\ \mathbf{t}^T/\epsilon \end{bmatrix} = \tilde{\mathbf{S}}\tilde{\mathbf{M}} \tag{5.31}$$

such that $\tilde{\mathbf{S}}$ satisfies the proposed hierarchical sparse model in (5.5) after appending ones to each dictionary. Therefore, a similar neural network could be derived from a 4-by-2 block sparse ISTA as $\tilde{\mathbf{M}} \in \mathbb{R}^{4\times 2}$.

5.5 Experiments

We conduct extensive experiments to evaluate the performance of our deep solution to solving NRS*f*M and S*f*C problems. For quantitative evaluation, we follow the metric normalized mean 3D error reported in [3, 5, 14, 23]. Our implementation, processed data, and pre-trained models are publicly accessible for future comparison[1].

5.5.1 IKEA Furniture

We first apply our method to a furniture dataset, IKEA dataset [30, 44]. The IKEA dataset contains four object categories: bed, chair, sofa, and table. For each object category, we project the 3D ground-truth by the orthogonal cameras annotated from real images. Since fully annotated images are limited, we thereby augment them with 2,000 projections under randomly generated orthogonal cameras. The errors are evaluated only on frames using cameras from real images. Numbers are summarized into Table 5.1. One can observe that our method outperforms baselines

[1] https://github.com/kongchen1992/deep-nrsfm

in the order of magnitude, clearly showing the superiority of our model. For qualitative evaluation, we randomly select a frame from each object category and show these frames in Figure 5.2 against ground-truth and baselines. As shown, our reconstructed landmarks effectively depict the 3D geometry of objects and our method is able to cover subtle geometric details.

Furnitures	Bed	Chair	Sofa	Table	Average	Relative
KSTA [23]	0.069	0.158	0.066	0.217	0.128	12.19
BMM [14]	0.059	0.330	0.245	0.211	0.211	20.12
CNR [29]	0.227	0.163	0.835	0.186	0.352	33.55
NLO [17]	0.245	0.339	0.158	0.275	0.243	23.18
RIKS [24]	0.202	0.135	0.048	0.218	0.117	11.13
SPS [26]	0.971	0.946	0.955	0.280	0.788	74.96
SFC [27]	0.247	0.195	0.233	0.193	0.217	20.67
OURS	**0.004**	**0.019**	**0.005**	**0.012**	**0.010**	**1.00**

Table 5.1: Quantitative Comparison against State-Of-The-Art Algorithms using IKEA Dataset in Normalized 3D Error.

5.5.2 PASCAL3D+ Dataset

We then apply our method to the PASCAL3D+ dataset [46], which contains twelve object categories. Following the experiment setting reported in [3], we also utilize eight categories: aeroplane, bicycle, bus, car, chair, dining table, motorbike and sofa. To explore the performance in various situations, we design experiments with respect to

- Orthogonal or weak perspective projection?
- Complete or missing measurement?
- Clean data or Gaussian noise perturbed?

Totally, there are eight configurations. Specifically, for projection setting, we randomly generate rotation matrices for orthogonal projection while additionally utilizing random scale and random translation for weak perspective projection. For missing data, we randomly sample approximately 10% of points missing for each category. For noise, we corrupt 2D correspondences with a zero mean Gaussian perturbation, following the same noise ratio in [3]. For the translation

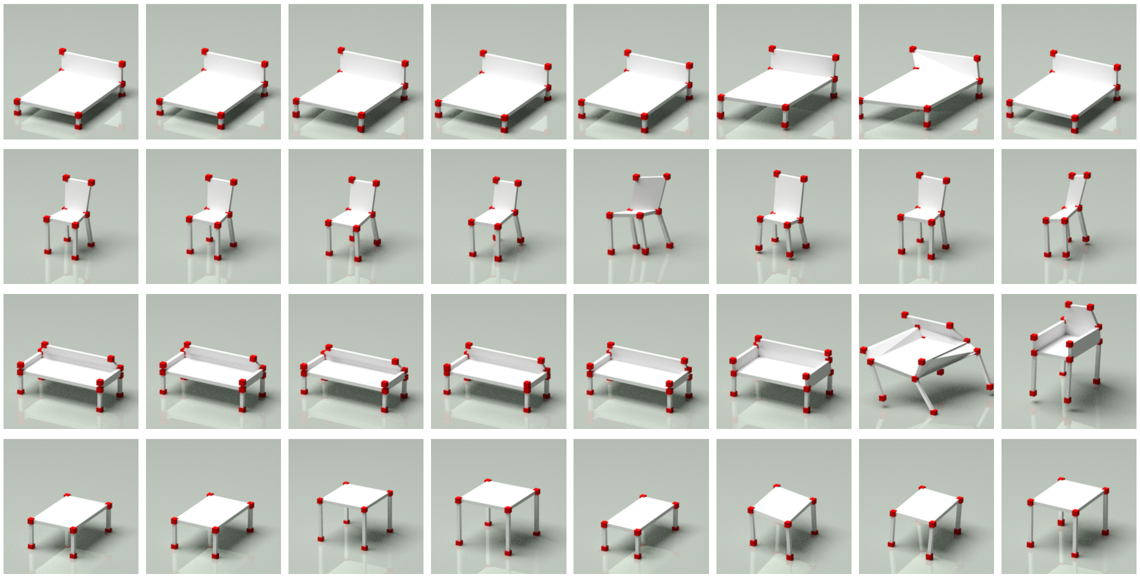


Figure 5.2: Qualitative evaluation on IKEA dataset. From top to bottom are tables, chairs, sofas and tables. From left to right are ground-truth and respectively reconstructions by ours, RIKS [24], KSTA [23], NLO [17], SFC [27], CNS [29], BMM [14]. In each rendering, red cubes are reconstructed points but the planes and bars are manually added for descent visualization.

residual, we simply treat it as noise and handle it with a 3-by-2 block sparse model. In Table 5.2 and Table 5.3, we report the normalized mean 3D error of our proposed method and state-of-the-arts: KSTA [23], RIKS [24], CNS [29], NLO [17], SFC [27], SPS [26], and BMM [14]. For readers' interest, one can compare our numbers against the Table 2 in [3] for more baselines.

From Table 5.2 and Table 5.3, one can observe that our proposed method achieves considerably more accurate reconstructions for all cases, and for some cases, more than ten times the amount of smaller 3D errors than state-of-the-arts. It clearly demonstrates the high precision of our proposed deep neural network. By comparing between clean and noisy configurations, it is shown that our proposed method has high robustness, where our method applied on noisy data even outperforms state-of-the-arts on clean data. By comparing between orthogonal and weak perspective projections, it is demonstrated that our proposed 3-by-2 block sparse model can handle scale and translation properly, even with missing data. In the configuration with missing measurement, KSTA, RIKS, BMM, CNS, and SPS use the matrix completion algorithm proposed by [22] to recover missing entries first, but our proposed method, SFC, and NLO can directly optimize over partially-visible 2D measurements, which are more capable at handling missing data. This is verified by Table 5.2 and Table 5.3, where OURS, SFC, and NLO sacrifice less performance than others when handling missing data. For qualitative evaluation, we use "motorbike"

		OURS	KSTA	RIKS	CNS	NLO	SFC	SPS	BMM
Orthogonal Projection	Complete Measurement	**0.013**	0.161	0.562	0.636	0.175	0.499	0.902	1.030
		0.003	0.249	0.826	0.732	0.285	0.370	0.959	1.247
		0.004	0.201	0.578	0.443	0.262	0.255	0.902	0.728
		0.003	0.124	0.497	0.497	0.135	0.284	0.955	1.006
		0.009	0.191	0.748	0.540	0.145	0.223	1.018	1.381
		0.030	0.244	0.778	0.549	0.234	0.220	0.707	1.351
		0.001	0.254	0.703	0.647	0.320	0.356	1.090	1.033
		0.007	0.401	0.798	0.623	0.055	0.302	0.779	1.017
	Missing Measurement	**0.033**	0.533	0.515	0.693	0.348	0.496	1.076	1.154
		0.021	0.584	0.540	0.854	0.106	0.376	1.112	1.372
		0.018	0.357	0.316	0.517	0.317	0.254	1.273	0.728
		0.010	0.400	0.334	0.598	0.089	0.286	0.918	1.014
		0.024	0.599	0.581	0.601	0.102	0.228	1.184	1.242
		0.040	0.554	0.473	0.602	0.171	0.224	1.264	1.414
		0.009	0.539	0.501	0.729	0.177	0.366	0.892	1.117
		0.015	0.573	0.567	0.728	0.911	0.301	1.214	1.171
Weak Perspective Projection	Complete Measurement	**0.034**	0.402	0.460	0.667	0.192	0.500	1.123	1.055
		0.008	0.576	0.817	0.707	0.595	0.373	1.172	1.301
		0.017	0.480	0.582	0.458	0.205	0.251	1.380	0.743
		0.015	0.369	0.573	0.504	0.175	0.284	1.090	1.051
		0.013	0.621	0.832	0.540	0.197	0.224	0.970	1.220
		0.025	0.647	0.829	0.533	0.428	0.220	0.927	1.447
		0.003	0.614	0.739	0.662	0.180	0.359	1.406	1.069
		0.022	0.609	0.792	0.632	0.070	0.295	0.976	0.980
	Missing Measurement	**0.102**	0.461	0.531	0.727	0.670	0.502	1.162	1.150
		0.048	0.499	0.572	0.875	0.115	0.372	1.312	1.279
		0.066	0.356	0.341	0.553	0.091	0.250	0.912	0.752
		0.027	0.402	0.403	0.637	0.093	0.280	0.949	0.954
		0.077	0.484	0.485	0.607	0.118	0.227	1.107	1.263
		0.091	0.463	0.465	0.594	0.174	0.232	1.210	1.229
		0.056	0.561	0.656	0.779	0.201	0.367	1.119	1.125
		0.066	0.529	0.615	0.728	0.081	0.311	1.730	1.150

Table 5.2: Quantitative Comparison against State-Of-The-Art Algorithms using PASCAL3D Dataset with no noise perturbation. In each configuration, numbers from top to bottom are for category aeroplane, bicycle, bus, car, chair, diningtable motorbike and sofa.

		OURS	KSTA	RIKS	CNS	NLO	SFC	SPS	BMM
Orthogonal Projection	Complete Measurement	**0.026**	0.175	0.583	0.626	0.167	0.518	0.761	1.177
		0.009	0.253	0.779	0.715	0.916	0.367	1.065	1.424
		0.012	0.196	0.450	0.442	0.320	0.253	1.096	0.754
		0.012	0.162	0.557	0.496	0.192	0.285	0.879	0.915
		0.028	0.190	0.668	0.554	0.107	0.224	0.927	1.251
		0.040	0.238	0.721	0.521	0.450	0.219	0.968	1.420
		0.004	0.251	0.722	0.629	0.168	0.366	0.938	1.029
		0.020	0.333	0.725	0.627	0.064	0.297	1.041	1.315
	Missing Measurement	**0.065**	0.434	0.514	0.707	0.382	0.493	0.815	1.199
		0.028	0.566	0.560	0.835	0.459	0.372	1.201	1.286
		0.057	0.364	0.323	0.526	0.079	0.245	0.791	0.743
		0.023	0.391	0.299	0.587	0.111	0.285	1.077	1.244
		0.066	0.571	0.479	0.593	0.103	0.229	1.153	1.274
		0.050	0.494	0.408	0.587	0.177	0.228	1.019	1.098
		0.032	0.523	0.528	0.730	0.154	0.363	1.100	1.157
		0.039	0.576	0.590	0.727	0.080	0.307	1.252	1.017
Weak Perspective Projection	Complete Measurement	**0.046**	0.525	0.489	0.644	0.206	0.527	0.961	1.203
		0.029	0.618	0.729	0.760	0.930	0.368	1.202	1.331
		0.044	0.384	0.443	0.443	0.666	0.248	0.820	0.739
		0.022	0.409	0.475	0.524	0.178	0.285	0.836	1.342
		0.026	0.497	0.622	0.543	0.122	0.226	1.283	1.284
		0.068	0.585	0.629	0.506	0.303	0.220	0.993	1.123
		0.018	0.607	0.789	0.671	0.159	0.362	1.101	1.019
		0.041	0.606	0.684	0.644	0.062	0.301	1.603	1.165
	Missing Measurement	**0.157**	0.449	0.571	0.737	0.742	0.493	0.984	1.220
		0.084	0.668	0.708	0.895	0.141	0.375	1.003	1.405
		0.091	0.383	0.365	0.557	0.139	0.253	0.985	0.752
		0.081	0.355	0.358	0.619	0.109	0.293	1.023	1.063
		0.122	0.522	0.434	0.601	0.123	0.224	1.037	1.263
		0.136	0.558	0.528	0.612	0.173	0.225	1.151	1.510
		0.051	0.544	0.585	0.763	0.191	0.369	1.039	1.017
		0.082	0.543	0.548	0.730	0.156	0.299	0.890	1.146

Table 5.3: Quantitative Comparison against State-Of-The-Art Algorithms using PASCAL3D Dataset with noise perturbation. In each configuration, numbers from top to bottom are for category aeroplane, bicycle, bus, car, chair, diningtable motorbike and sofa.

as an exemplar category and randomly select a frame from four configurations: 1) orthogonal+complete+noise, 2) orthogonal+missing+noise, 3) weak perspective+complete+noise, and 4) weak perspective+missing+noise, showing in Figure 5.3. One can observe that our proposed method outperforms KSTA, RIKS, CNS, and SPS obviously and beats NLO and SFC in reconstruction details, e.g. handlebar. The figure also verifies that KSTA, RIKS, CNS, and SPS break easily with missing points while ours, SFC, and NLO maintain a nice stability against missing entries.

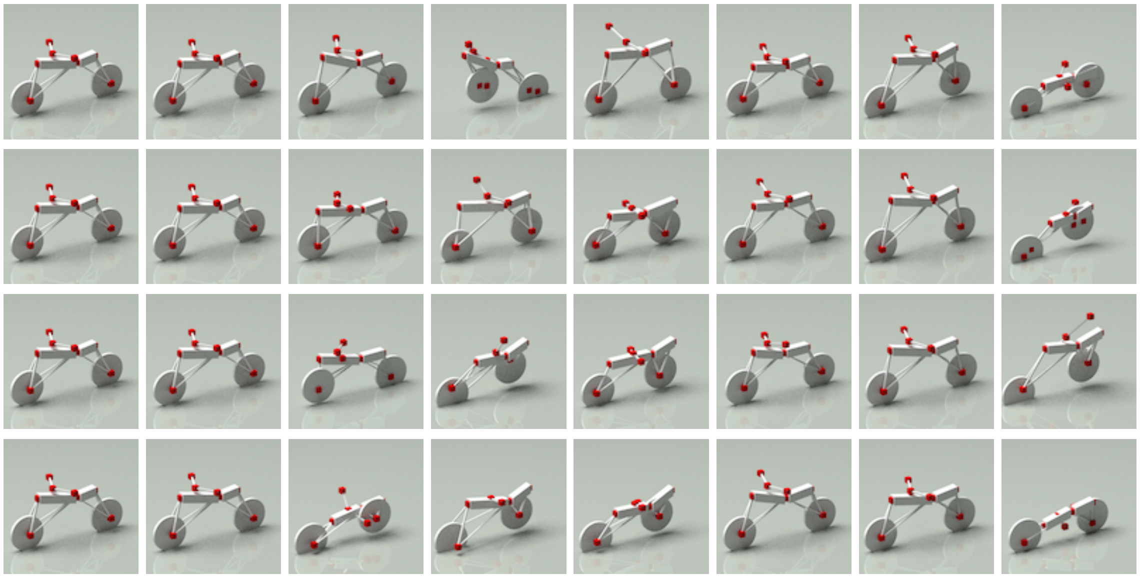


Figure 5.3: Qualitative evaluation on PASCAL3D dataset. From top to bottom are configurations 1) orthogonal projection with no missing points, 2) orthogonal projection with missing points, 3) weak perspective projection with no missing points, 4) weak perspective projection with missing points. All these four configurations are perturbed by Gaussian noise. From left to right are ground-truth, ours, KSTA [23], RIKS [24], CNS [29], NLO [17], SFC [27], SPS [26]. In each rendering of reconstruction, red cubes are reconstructed points but the planes and bars are manually added for visualization.

5.5.3 Large-Scale NRS*f*M on CMU Motion Capture

To evaluate the performance of our method on a large scale image sequence, we apply our method to solving the problem of NRS*f*M, using the CMU motion capture dataset[2]. We randomly select 10 subjects out of 144, and for each subject, we concatenate 80% of motions to form large image collections and leave the remaining 20% as unseen motions for testing generalization. In

[2] http://mocap.cs.cmu.edu/

this experiment, each subject contains more than ten thousand frames under randomly generated orthogonal projections. We compare our method against state-of-the-art methods, summarized in Table 5.4. Due to the huge volume of frames, KSTA [23], BMM [14], MUS [3], RIKS [24], and SFC [27] all fail and thus are omitted in the table. We also report the normalized mean 3D error on unseen motions, labeled as UNSEEN. From Table 5.4, one can see that our method obtains impressive reconstruction performance and outperforms all others again in every sequence. Moreover, our network generalizes well with unseen data, which implies the potential utility of our model to the application of single image 3D reconstruction. For qualitative evaluation, we randomly select a frame from each subject and render the reconstructed human skeleton in Figure 5.4, which visually verifies the impressive performance of our deep solution.

SUBJECT	OURS	CNS	NLO	SPS	UNSEEN
1	**0.176**	0.613	1.218	1.282	0.362
5	**0.221**	0.657	1.160	1.122	0.331
18	**0.082**	0.542	0.917	0.954	0.438
23	**0.054**	0.604	0.999	0.880	0.388
64	**0.082**	0.543	1.219	1.120	0.174
70	**0.040**	0.473	0.837	1.010	0.090
102	**0.116**	0.582	1.145	1.079	0.413
106	**0.114**	0.637	1.016	0.958	0.195
123	**0.041**	0.479	1.009	0.828	0.092
127	**0.095**	0.645	1.051	1.022	0.389

Table 5.4: Quantitative Comparison aginst State-Of-The-Arts using CMU Motion Capture Dataset in Normalized 3D Error

Robustness analysis

To analyze the robustness of our method, we retrain the neural network for Subject 70, using projected points perturbed by Gaussian noise. The results are summarized in Figure 5.5. The noise ratio is defined as $\|\text{noise}\|_F / \|\mathbf{W}\|_F$. One can see that the error increases slowly while adding a higher magnitude of noise; when adding up to 20% noise to image coordinates, our

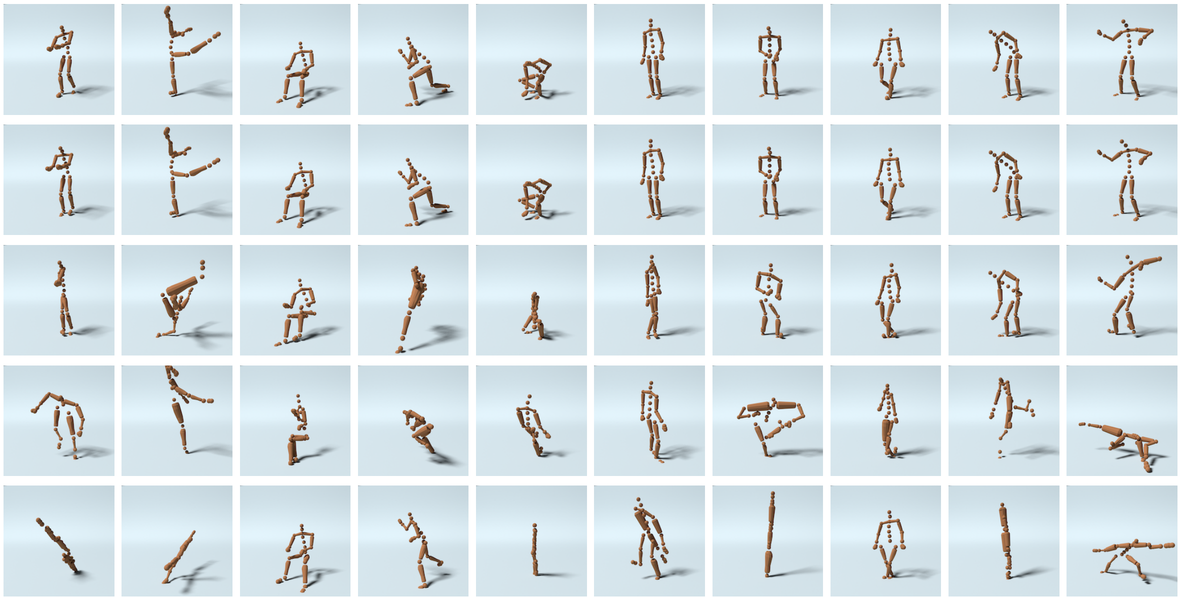


Figure 5.4: Qualitative evaluation on CMU Motion Capture dataset. From top to bottom are ground-truth, and respectively reconstructions by ours, CNS [29], SPS [26], NLO [17]. From left to right are a randomly sampled frame from subjects 1, 5, 18, 23, 64, 70, 102, 106, 123, 127. In each rendering, spheres are reconstructed landmarks but bars are for descent visualization. In each reconstruction, 3D shapes are alighted to the ground-truth by a orthonormal matrix.

method in blue still achieves better reconstruction compared to the best baseline with no noise perturbation (in red). This experiment clearly demonstrates the robustness of our model and its high accuracy against state-of-the-art works.

Explicitly solve translation

In this experiment, we verify the performance of the proposed 4-by-2 block sparse model. We focus on Subject 23, following the same experiment setting as above, except adding randomly generated translation. To avoid removing translation, we do not normalize 2D correspondences. We then apply the proposed 4-by-2 block sparse model to the data with translation and compare it to the 3-by-2 block sparse model without translation. The normalized mean 3D error of the 4-by-2 model is 0.060, which is very close to the error without translation, i.e. 0.054, and lower than state-of-the-arts without translation in the order of magnitude, as shown in Table 5.4. To give a clearer sense of the quality of the reconstructed 3D shape, we draw two cumulative error plots in Figure 5.6 that show the percentage of frames below a certain normalized mean 3D error. The two plots are mostly identical, implying the success of our 4-by-2 model.

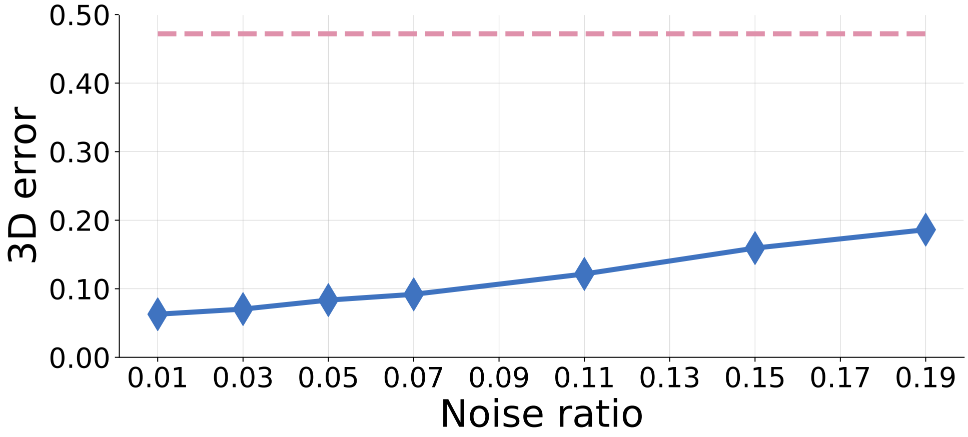


Figure 5.5: Normalized mean 3D error on CMU Motion Capture dataset with Gaussian noise perturbation. The blue solid line is ours while the red dashed line is CNS [29], the lowest error of state-of-the-arts with *no* noise perturbation.

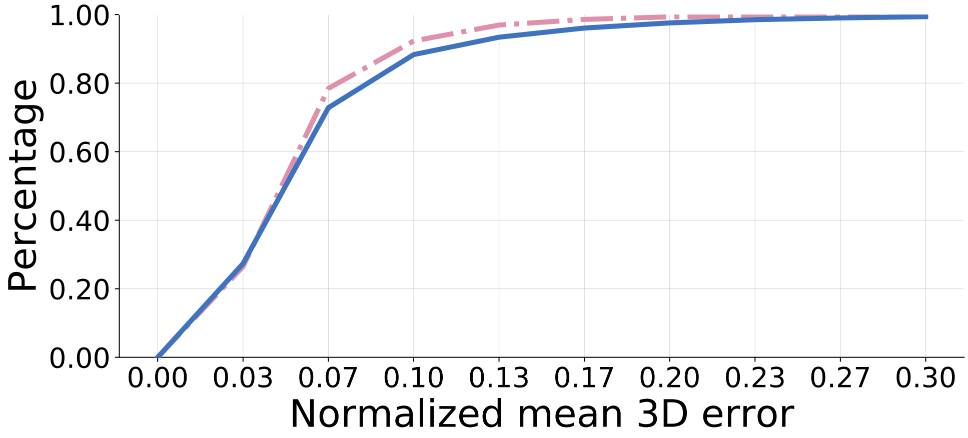


Figure 5.6: Percentage below a certain normalized mean 3D error. The blue solid line is our 4-by-2 block sparse model, proposed to solve translation explicitly. The red dashed line is our 3-by-2 block sparse model, applied on zero-centered data. These two plots are mostly identical.

Missing points

In this experiment, we explore the capability of handling missing data. We focus on Subject 23 under orthogonal projection and sequentially train and test our proposed network on data with a different percentage of missing points. Specifically, we control the maximum possible number of missing points and evaluate the performance from one to seven out of 31 total points. For example, when the maximum possible number of missing points is three, then each frame has to have one, two, or three missing points in uniform distribution. We visualize the normalized mean 3D error in each case in Figure 5.7 and append the lowest error achieved by state-of-the-arts under the complete measurement assumption as a baseline. One can see that the 3D error increases when the maximum possible number of missing points grows. However, even making approximately 20% (7/31) of points invisible, our proposed method still outperforms the best baseline with no missing points, i.e. CNS 0.604 in Table 5.4.

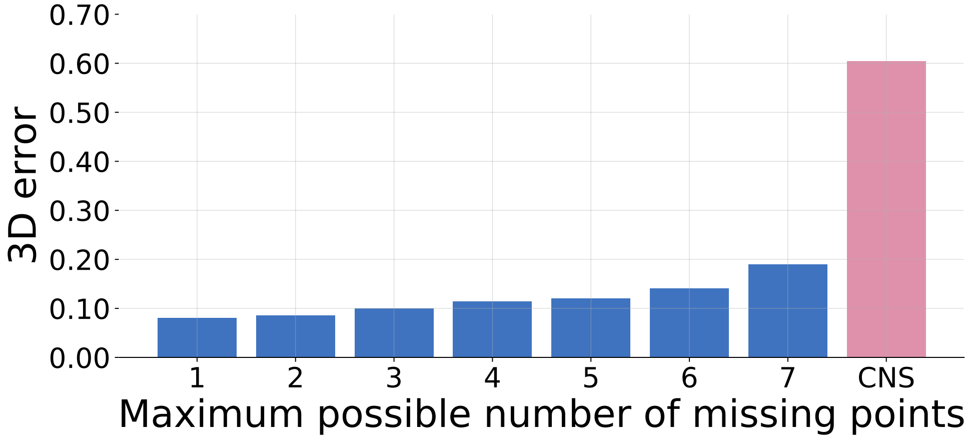


Figure 5.7: Normalized mean 3D error v.s. maximum possible number of missing points. Maximum possible number of missing points equals to three denotes every frame has to have one, two, or three missing points. The blue bar is our proposed network. The red bar is the best baseline when all points are visible, i.e. CNS in Table 5.4.

Coherence as guide

Over-fitting is commonly observed in the deep learning community, especially in the NRS*f*M area, where over-fitting to 2D correspondences will dramatically hurt the quality of reconstructions. To solve this problem, we borrow a tool from compressed sensing – mutual coherence [19]. Mutual coherence measures the similarity between atoms in a dictionary. It is often used to depict the dictionary quality and build the bounds of sparse code reconstructability. During training for each subject, we compute the normalized mean 3D error and the coherence of the last dictionary in a fixed training iteration interval. By drawing the scatter plot of the error and the coherence, we observe a strong correlation, shown in Figure 5.8. This implies that the coherence of the final dictionary could be used as a measure of model quality.

Recall the proposed block sparse model in (5.6), wherein every block sparse code $\mathbf{\Psi}_i$ is constrained by its subsequent representation and thus, the quality of code recovery depends not only on the quality of the corresponding dictionary but also the subsequent layers. However, this is not applicable to the final code $\mathbf{\Psi}_N$, making it overly reliant upon the final dictionary $\mathbf{D}_N$.

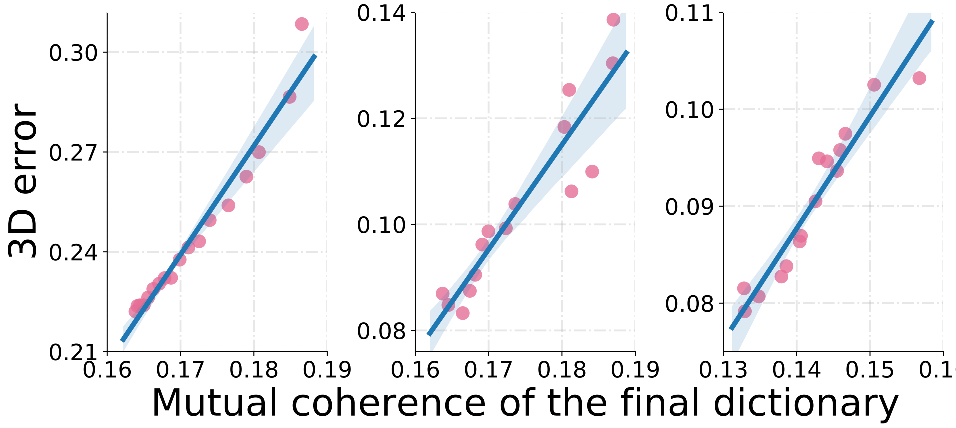


Figure 5.8: A scatter plot of the normalized mean 3D error v.s. the coherence of the final dictionary. The blue line is fitted based on the red points. Shading presents the quality of linear regression. From left to right are, respectively, for Subjects 5, 18, and 64.

From this perspective, the quality of the final dictionary measured by mutual coherence could serve as a lower bound of the entire system. With the help of the coherence, we could avoid overfitting even when 3D evaluation is not available. This improves the utility of our deep NRS*f*M in applications without 3D ground-truth.

ReLU v.s. Soft-thresholding

Theoretical analysis implies that using soft-thresholding or ReLU is expected to estimate a similar reconstructions in terms of accuracy but soft-thresholding operator tends to result in a more compact parameter size. We verify this on the large-scale dataset, CMU MoCap. We follow the same experiment setting. Specifically, we change ReLU to soft-thresholding operator and re-trained the neural network on Subject 23. Table 5.5 summarizes the normalized 3D error and the number of parameters. One can see that compared to the best baselines CNS, 0.604, these two networks show very subtle difference (0.01) in normalized 3D error while soft-thresholding network has less learnable parameters, which verifies what we predict in theory.

ACTIVATION	3D ERROR	# of PARAMETERS
ReLU	**0.054**	86787
Soft-thresholding	0.064	**61351**

Table 5.5: Quantitative Comparison between ReLU and soft-thresholding

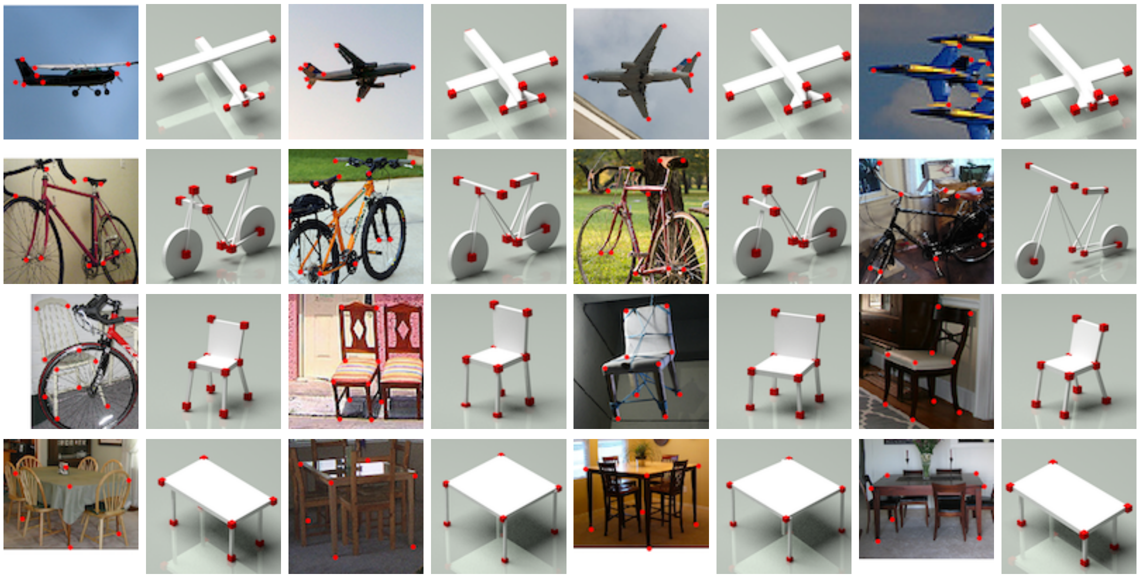


Figure 5.9: Qualitative evaluation on real images with hand-annotated 2D correspondences. Some images have missing points, due to occlusion. From top to bottom are aeroplanes, bicycles, chairs, and dining tables. For each pair, the left is an image with key points in red and the right is our reconstruction. In each rendering of reconstruction, red cubes are reconstructed points, but the planes and bars are manually added for descent visualization. Our method successfully captures shape variations presented in the images, e.g. table width-length ratio, the position of aeroplane wings, bicycle handlebar, and so forth.

5.5.4 Real Images

Our proposed network is designed for applications on large-scale image sequences of highly deformable objects, especially object categories. However, to our best knowledge, commonly-used object datasets mostly contain less than one hundred images of reasonable quality, a number which is greatly insufficient to train a neural network. For example, most objects in the PASCAL3D dataset have more than 50% occluded points. To demonstrate the performance of our proposed network, we apply the model pre-trained on synthetic images to real images with hand annotated correspondences. Due to the absence of 3D ground-truth, we qualitatively evaluate the reconstructed shapes and show them in Figure 5.9. One can see that our proposed neural network successfully reconstructs the 3D shape for each image and impressively captures the subtle shape variation presented in the image, e.g. the table width-length ratio, the position of aeroplane wings, the bicycle handlebar and so forth.

Chapter 6

Discussion and Future Work

The major limitation of our proposed methods is the weak perspective projection. The reasons for using weak perspective projection are:

1. Weak perspective projection is an appropriate approximation to perspective projection, especially when the object of interest has depth variation much smaller than the camera distance. This is fairly common in the scene of a non-rigid object, e.g. a moving person or moving animals, and of object category, e.g. chairs, tables in indoor scenes, and aeroplanes in outdoor scenes.
2. Weak perspective projection maintains a linear equation when projecting the 3D geometry into 2D images. This is crucial to most existing NRS*f*M algorithms [3, 14, 51], and ours are not exceptions.

We believe that the non-linear and cross-connected nature of perspective projection encourages the use of an iterative or recurrent solution instead of a feed-forward neural network. A lower fruit based on our propose method is to generalize our feed-forward auto-encoder to a recurrent neural network. There are two potential candidates.

6.1 Global Recurrent NRS*f*M

Global recurrent NRS*f*M refers to adding a shortcut from the output 3D structure $\mathbf{S}$ to the input 2D correpondences $\mathbf{W}$, such that passing through the network includes a fixed number of iterations or until convergence. The recurrent neural network caused by this shortcut could offer several benefits:

- Scale feedback: If one knows the camera scale from oracle, we could remove it from the projection via dividing the 2D coordinates of correspondences by the scale. A recurrent neural network allows us to obtain the camera scale from the output 3D structure $\mathbf{S}$ and the output camera $\mathbf{M}$. After eliminating the scale and passing the neural network again, a more accurate estimation is expected. Such a procedure could be repeated by a fixed number of times or until convergence.

- Linear perspective projection: The difficulty of perspective projection, non-linearly, is mostly caused by the fact that object depth is unknown. If one knows the object depth from oracle, then the perspective projection could be degenerated to a linear projection equation. A recurrent neural network directly offers the depth of an object from the reconstruction. This allows us to utilize a local feed-forward neural network to reconstruct the 3D structure and camera motions, and then correct the object depth. By repeating this process by a fixed number of iterations or until convergence, we believe an object even under perspective projection could be reconstructed with no additional prior information.
- Missing point estimation: The current strategy of recovering missing points—filling missing entries with zero and feeding into neural network—actually relies on the strong robustness of the proposed neural network. Different from it, the introduced shortcut from the output could provide a better guess of the missing correspondences. We believe that filling the missing entries with iteratively corrected 2D coordinates could help to increase the reconstruction accuracy and, moreover, enable a powerful capability of handling missing data.

6.2 Local Recurrent NRS*f*M

Recall in Chapter 5, we derived the architecture from the sparse code recovery algorithm, ISTA, resulting in an iterative solution:

$$\mathbf{V} = \mathbf{Z}^{[i]} - \alpha \mathbf{D}^T (\mathbf{D}\mathbf{Z}^{[i]} - \mathbf{X}), \tag{6.1}$$

$$\mathbf{Z}^{[i+1]} = \operatorname*{argmin}_{\mathbf{U}} \frac{1}{2} \|\mathbf{U} - \mathbf{V}\|_F^2 + \tau \|\mathbf{U}\|_{F1}^{(3\times 2)}. \tag{6.2}$$

By employing only the very first iteration, we have the expression of a single layer encoder.

It is demonstrated by our experiments that a concatenation of a single iteration of ISTA successfully fulfills our task. However, an interesting question still remains open: that is, how does the number of ISTA iterations affect the performance? We believe that additional iteration in each encoder layer could help to recover more accurate sparse hidden features and consequently increase the overall reconstruction performance.

Chapter 7

Conclusion

This thesis focused on one particular problem: non-rigid structure from motion. Classic non-rigid structure from motion focuses on the problem of reconstructing 3D shapes of non-rigid objects and recovering camera motions from a sequence of images. This thesis first characterized that non-rigid structure from motion could be equally applied to rigid objects, i.e. the object category. Then we revisited several celebrated algorithms in this area, including Tomasi-Kanade's algorithm, Bregler's low-rank assumption, Dai et al.'s prior-less algorithm, Akhter et al.'s trajectory reconstruction, and Zhu et al.'s union of subspaces strategy. Their advantages and disadvantages help readers to understand the motivation and contribution of the later-proposed algorithm.

Different from the aforementioned priors, this thesis proposed to use sparse coding as a novel prior assumption to represent non-rigid objects or an object category. Compared with low-rank priors and union-of-subspaces, the block sparse prior forms a union of a huge number of subspaces so that a much broader set of 3D structures can be modeled successfully. Based on this assumption, we demonstrated that a 3D structure under weak perspective projection could be represented in a 2×3 block-sparse way, from which the non-rigid structure from motion problem could be reinterpreted as a block sparse dictionary learning problem. To demonstrate the reconstructability, we first theoretically prove the uniqueness of block sparse dictionary learning and then practically establish algorithms to simultaneously learn a shape dictionary and block sparse representation. Once a unique 2×3 block sparse dictionary learning factorization of the 2D projections can be obtained, we showed that the 3D structure and camera motion can be recovered solely by the assumption of sparse coding.

Though they offer considerable interesting insights to the problem, the algorithms based on the uniqueness of block sparse dictionary learning are sensitive to noise in our experiments. This could be caused by the fragile nature of the uniqueness. To alleviate the problem, this thesis proposed an optimization strategy derived from alternating the direction method of multipliers to minimize the reprojection error and, at the same time, satisfy the camera orthogonal constraints and maintain the sparsity of shape representation. Experiments demonstrate that the proposed optimization algorithms are much more stable than the previous closed-form solution and outperform the previous algorithms on the object category dataset.

Next, this thesis theoretically explored the reason why block sparse prior results in a less stable system compared to others and demonstrated that the major cause can be the large number of subspaces, which makes selecting the correct subspaces substantially more difficult than low rank algorithms. Based on this insight, we proposed to use hierarchical sparse coding, replacing the regular sparse coding to represent a 3D deformable structure. From the recent progress on understanding deep neural networks via convolutional sparse coding, we designed a deep neural network serving as a hierarchical block sparse dictionary solver. Our proposed architecture is not a block-box but a transparent glass-box in terms of its interpretability. Extensive experiments demonstrated our superior performance against all state-of-the-arts on various configurations, including orthogonal projections, weak perspective projections, noise perturbations, missing points, real images, and even unseen shape variations. Our proposed hierarchical block sparse prior not only successfully avoids the previous sensitivity to noise, but also provides the capacity and efficiency to handle unprecedented scale in terms of the number of images and the types of shape variations. Finally, this thesis proposed to use the coherence of the learned dictionary as a generalization measure, i.e. metrics of reconstructability, offering a practical way to avoid over-fitting and ascertain the correctness of reconstructions in real-world applications.

Chapter 8

Bibliography

[1] Alekh Agarwal, Animashree Anandkumar, Prateek Jain, and Praneeth Netrapalli. Learning sparsely used overcomplete dictionaries via alternating minimization. *arXiv preprint arXiv:1310.7991*, 2013. 3.3.4

[2] Sameer Agarwal, Yasutaka Furukawa, Noah Snavely, Ian Simon, Brian Curless, Steven M Seitz, and Richard Szeliski. Building rome in a day. *Communications of the ACM*, 54(10): 105–112, 2011. 2

[3] Antonio Agudo, Melcior Pijoan, and Francesc Moreno-Noguer. Image collection pop-up: 3d reconstruction and clustering of rigid and non-rigid categories. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2607–2615, 2018. 1, 5.4.1, 5.5, 5.5.2, 5.5.3, 2

[4] Ijaz Akhter, Yaser Sheikh, and Sohaib Khan. In defense of orthonormality constraints for nonrigid structure from motion. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 1534–1541. IEEE, 2009. 3

[5] Ijaz Akhter, Yaser Sheikh, Sohaib Khan, and Takeo Kanade. Nonrigid structure from motion in trajectory space. In *Advances in neural information processing systems*, pages 41–48, 2009. (document), 3.3, 5.5

[6] Ijaz Akhter, Yaser Sheikh, Sohaib Khan, and Takeo Kanade. Trajectory space: A dual representation for nonrigid structure from motion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(7):1442–1456, 2011. 2.3, 2.3, 5.3.2

[7] Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm with application to wavelet-based image deblurring. In *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*, pages 693–696. IEEE, 2009. 2.6

[8] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122, 2011. 1, 3.3.4, 4.1, 4.2

[9] Christoph Bregler, Aaron Hertzmann, and Henning Biermann. Recovering non-rigid 3d shape from image streams. In *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, volume 2, pages 690–696. IEEE, 2000. 1, 1, 2.2

[10] Hilton Bristow, Anders Eriksson, and Simon Lucey. Fast convolutional sparse coding. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 391–398. IEEE, 2013. 2, 3.3.4, 4.2

[11] Richard A Brualdi. *Introductory combinatorics*. New York, 1992. 2

[12] Angel X. Chang, Thomas A. Funkhouser, Leonidas J. Guibas, Pat Hanrahan, Qi-Xing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. Shapenet: An information-rich 3d model repository. *CoRR*, abs/1512.03012, 2015. URL `http://arxiv.org/abs/1512.03012`. 1

[13] Yuchao Dai, Hongdong Li, and Mingyi He. A simple prior-free method for non-rigid structure-from-motion factorization. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2018–2025. IEEE, 2012. 2.2

[14] Yuchao Dai, Hongdong Li, and Mingyi He. A simple prior-free method for non-rigid structure-from-motion factorization. *International Journal of Computer Vision*, 107(2): 101–122, 2014. (document), 1, 1, 3.3.5, 3.4.3, 3.3, 4.3.1, 4.3.2, 4.1, 4.2, 4.1, 4.3, 5.2.1, 5.3.1, 5.3.2, 5.5, **??**, 5.2, 5.5.2, 5.5.3, 2

[15] Ingrid Daubechies, Michel Defrise, and Christine De Mol. An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences*, 57(11):1413–1457, 2004. 2.6

[16] Mark A Davenport, Marco F Duarte, Yonina C Eldar, and Gitta Kutyniok. Introduction to compressed sensing. *Preprint*, 93:1–64, 2011. 2.5

[17] Alessio Del Bue, Fabrizio Smeraldi, and Lourdes Agapito. Non-rigid structure from motion using ranklet-based tracking and non-linear optimization. *Image and Vision Computing*, 25 (3):297–310, 2007. (document), **??**, 5.2, 5.5.2, 5.3, 5.4

[18] Wei Deng, Wotao Yin, and Yin Zhang. Group sparse optimization by alternating direction method. In *SPIE Optical Engineering+ Applications*, pages 88580R–88580R. International Society for Optics and Photonics, 2013. 3.3.4

[19] David L Donoho, Michael Elad, and Vladimir N Temlyakov. Stable recovery of sparse overcomplete representations in the presence of noise. *IEEE Transactions on information theory*, 52(1):6–18, 2006. 5.5.3

[20] Yonina C Eldar and Moshe Mishali. Robust recovery of signals from a structured union of subspaces. *Information Theory, IEEE Transactions on*, 55(11):5302–5316, 2009. 2.2

[21] Irina F Gorodnitsky and Bhaskar D Rao. Sparse signal reconstruction from limited data using focuss: A re-weighted minimum norm algorithm. *Signal Processing, IEEE Transactions on*, 45(3):600–616, 1997. 3.3, 3.3.3, 3.3.3, 3.3.3

[22] Paulo FU Gotardo and Aleix M Martinez. Computing smooth time trajectories for camera and deformable shape in structure from motion with occlusion. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 33(10):2051–2065, 2011. (document), 3.3, 5.5.2

[23] Paulo FU Gotardo and Aleix M Martinez. Kernel non-rigid structure from motion. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 802–809. IEEE, 2011. (document), 1, 2.5, 5.5, **??**, 5.2, 5.5.2, 5.3, 5.5.3

[24] Onur C Hamsici, Paulo FU Gotardo, and Aleix M Martinez. Learning spatially-smooth mappings in non-rigid structure from motion. In *European Conference on Computer Vision*, pages 260–273. Springer, 2012. (document), **??**, 5.2, 5.5.2, 5.3, 5.5.3

[25] Christopher Hillar and Friedrich T Sommer. When can dictionary learning uniquely recover sparse data from subsamples? In *IEEE Transactions on Information Theory*, 2015. 3.1.1, 3.1.1, 1, 3.1.3

[26] Chen Kong and Simon Lucey. Prior-less compressible structure from motion. *Computer Vision and Pattern Recognition (CVPR)*, 2016. (document), 5.2, 5.2, 5.3.1, 5.3.2, **??**, 5.5.2, 5.3, 5.4

[27] Chen Kong, Rui Zhu, Hamed Kiani, and Simon Lucey. Structure from category: a generic and prior-less approach. *International Conference on 3DVision (3DV)*, 2016. (document), 1, **??**, 5.2, 5.5.2, 5.3, 5.5.3

[28] Suryansh Kumar, Anoop Cherian, Yuchao Dai, and Hongdong Li. Scalable dense non-rigid structure-from-motion: A grassmannian perspective. *arXiv preprint arXiv:1803.00233*, 2018. 2.5

[29] Minsik Lee, Jungchan Cho, and Songhwai Oh. Consensus of non-rigid reconstructions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4670–4678, 2016. (document), **??**, 5.2, 5.5.2, 5.3, 5.4, 5.5

[30] Joseph J. Lim, Hamed Pirsiavash, and Antonio Torralba. Parsing IKEA Objects: Fine Pose Estimation. *ICCV*, 2013. 5.5.1

[31] Angshul Majumdar and Rabab Kreidieh Ward. Some empirical advances in matrix completion. *Signal Processing*, 91(5):1334–1338, 2011. 4.3.4

[32] Vardan Papyan, Yaniv Romano, and Michael Elad. Convolutional neural networks analyzed via convolutional sparse coding. *The Journal of Machine Learning Research*, 18(1):2887–2938, 2017. 1.1, 2.6

[33] Vincent Rabaud and Serge Belongie. Re-thinking non-rigid structure from motion. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008. 2.5

[34] Christopher J Rozell, Don H Johnson, Richard G Baraniuk, and Bruno A Olshausen. Sparse coding via thresholding and local competition in neural circuits. *Neural computation*, 20 (10):2526–2563, 2008. 2.6

[35] Ron Rubinstein, Tomer Peleg, and Michael Elad. Analysis k-svd: A dictionary-learning algorithm for the analysis sparse model. *Signal Processing, IEEE Transactions on*, 61(3): 661–677, 2013. 3.3, 3.3.1

[36] Hao Su, Charles R Qi, Yangyan Li, and Leonidas J Guibas. Render for cnn: Viewpoint estimation in images using cnns trained with rendered 3d model views. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2686–2694, 2015. 1, 4.3.4

[37] Petri Tanskanen, Kalin Kolev, Lorenz Meier, Federico Camposeco, Olivier Saurer, and Marc Pollefeys. Live metric 3d reconstruction on mobile phones. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 65–72, 2013. 2

[38] Carlo Tomasi and Takeo Kanade. Shape and motion from image streams under orthography: a factorization method. *International Journal of Computer Vision*, 9(2):137–154, 1992. (document), 2, 3, 3.3.5, 4.3.1, 4.3.2, 4.1, 4.2, 4.1, 4.3

[39] Lorenzo Torresani, Aaron Hertzmann, and Christoph Bregler. Nonrigid structure-from-motion: Estimating shape and motion with hierarchical priors. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 30(5):878–892, 2008. 3.4.5

[40] Joel A Tropp and Anna C Gilbert. Signal recovery from random measurements via orthogonal matching pursuit. *Information Theory, IEEE Transactions on*, 53(12):4655–4666, 2007. 3.3, 3.3.2

[41] L Richard Turner. Inverse of the vandermonde matrix with applications. 1966. 3.1.3

[42] NP Van der Aa, Xinghan Luo, Geert-Jan Giezeman, Robby T Tan, and Remco C Veltkamp.

Umpm benchmark: A multi-person dataset with synchronized video and motion capture data for evaluation of articulated human motion and interaction. In *2011 IEEE international conference on computer vision workshops (ICCV Workshops)*, pages 1264–1269. IEEE, 2011. (document), 2.2

[43] Sara Vicente, João Carreira, Lourdes Agapito, and Jorge Batista. Reconstructing pascal voc. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 41–48. IEEE, 2014. 1, 4.3.2

[44] Jiajun Wu, Tianfan Xue, Joseph J Lim, Yuandong Tian, Joshua B Tenenbaum, Antonio Torralba, and William T Freeman. Single image 3d interpreter network. *European Conference on Computer Vision (ECCV)*, 2016. 5.5.1

[45] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1912–1920, 2015. 1

[46] Yu Xiang, Roozbeh Mottaghi, and Silvio Savarese. Beyond pascal: A benchmark for 3d object detection in the wild. In *IEEE Winter Conference on Applications of Computer Vision*, pages 75–82. IEEE, 2014. 4.3.1, 5.5.2

[47] Jing Xiao, Jinxiang Chai, and Takeo Kanade. A closed-form solution to non-rigid shape and motion recovery. *International Journal of Computer Vision*, 67(2):233–246, 2006. 3

[48] Xuehan Xiong and Fernando De la Torre. Supervised descent method and its applications to face alignment. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 532–539, 2013. 4.3.4, 4.3.4

[49] Xuehan Xiong and Fernando De la Torre. Global supervised descent method. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2664–2673, 2015. 4.3.4

[50] Xiaowei Zhou, Spyridon Leonardos, Xiaoyan Hu, and Kostas Daniilidis. 3d shape estimation from 2d landmarks: A convex relaxation approach. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4447–4455, 2015. 4.1, 4.1, 4.2

[51] Yingying Zhu and Simon Lucey. Convolutional sparse coding for trajectory reconstruction. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 37(3):529–540, 2015. 2

[52] Yingying Zhu, Dong Huang, Fernando De La Torre, and Simon Lucey. Complex non-rigid motion 3d reconstruction by union of subspaces. In *Computer Vision and Pattern Recog-*

nition (CVPR), 2014 IEEE Conference on, pages 1542–1549. IEEE, 2014. (document), 1, 2.2, 2.4, 2.3, 2, 5.3.1