Minimal Basis Representation for General Motion Segmentation

Lee Choon Meng

A THESIS SUBMITTED FOR THE DEGREE OF

Philosophiae Doctor

ECE Department, Faculty of Engineering
National University of Singapore

Declaration

I hereby declare that this thesis is my original work and it has been written by me in its entirety. I have duly acknowledged all the sources of information which have been used in the thesis.

This thesis has also not been submitted for any degree in any university previously.

Lee Choon Meng
14th July 2014

Acknowledgements

I would like to thank my advisor Cheong Loong Fah for his patience and wisdom in guiding me through this arduous PhD journey. A PhD candidature is an exciting time due to the exploring of new territories and creation of new knowledge. It can also be incredibly frustrating and forlorn, with the problem seemingly unsolvable. Yet day after day, week after week, he listened and worked through the road blocks one by one, never once abandoning me or losing patience with me. Thank you.

Abstract

While motion segmentation has been an active research area, the model selection aspect has often been neglected. Due to the difficulty of simultaneously estimating the number of motion and segmenting the trajectories, the number of motion is often assumed known.

In this thesis, we present a model selection mechanism based on finding the minimal basis subspace representation. This model selection mechanism is the enabler for our proposed general motion segmentation work that is capable of strong competitive performance for both rigid and non-rigid motion. The good performance can be attributed to the explicit modeling of overlapping subspaces by identifying the shared bases, which is also key to ensuring the recovery of a global shape in non-rigid structure from motion.

We first apply our general motion segmentation work to rigid motion segmentation by evaluating both the model selection and segmentation performance against the state-of-the-art rigid motion segmentation algorithms, using the standard Hopkins 155 and extended Hopkins 380 dataset.

These evaluations show that our work offers the best performance.

Based on this general motion segmentation work, we develop a new subspace segmentation approach to non-rigid structure from motion. This new subspace segmentation approach decomposes a complex non-rigid motion into subgroups of relatively simpler motion, which can be more easily reconstructed. Even without the benefit of ground truth, our approach compares favorably with the state-of-the-art works.

Abbreviations

NRSFM	Non-Rigid Structure From Motion
SSC	Sparse Subspace Clustering
LRR	Low Rank Representation
ALM	Augmented Lagrange Multiplier
ADMM	Alternating Direction Method of Multipliers
APG	Accelerated Proximal Gradient
ORK	Ordered Residue Kernel
MB-FLoSS	Minimal Basis Facility Location for Subspace Segmentation
LSC	Linear Subspace Spectral Clustering
KO	Kernel Optimization
AP	Affinity Propagation
SIM	Shape Interaction Matrix
GPCA	Generalized Principal Component Analysis
LSA	Linear Subspace Affinity
FLoSS	Facility Location for Subspace Segmentation
UFLP	Uncapacitated Facility Location Problem
MPBP	Max-Product Belief Propagation
CFL	Capacitated Facility Location
DCT	Discrete Cosine Transform
CSF	Column Space Fitting work
TB	Trajectory Basis
MP	Metric Projection
MDL	Minimum Description Length
EM	Expectation-Maximization
GPA	Generalized Procrustes Analysis
SPF	Simple Prior Free

Contents

List of Figures

Chapter 1

Introduction

The segmentation topic is usually associated with object segmentation. Consider the beach scene in figure 1.1. Object segmentation is the task of separating the image into different groups based on properties such as edges and saliency. Figure 1.2 is one of the many possible outcomes of segmentation. Object segmentation is difficult due to noise such as the disruption of edges and change of intensity along the edges. For example, the beach in figure 1.1 has been broken up into two segments due to the presence of the tree.

Figure 1.1: An image to be segmented

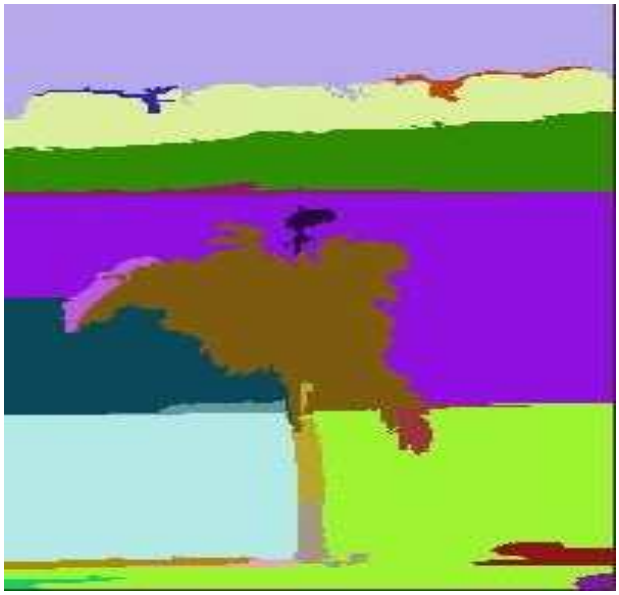


Figure 1.2: Image after segmentation

Motion segmentation is fundamentally different from object seg-

mentation. As the name suggests, the task of motion segmentation is to group rigid objects according to their motion over video frames, based on the tracked features in each frame. Figure 1.3 shows one frame of a checkerboard sequence, which consists of 26 video frames of 538 tracked points overlayed on the original image. There are three motions shown in the figure, marked in red, green and blue. The only relevant information in motion segmentation are the (x, y) coordinates of the tracked feature points over the entire video frames. The (x, y) image coordinates of the tracked feature points are stacked to form the data matrix, which is the only input to motion segmentation. The image color intensity plays no part in motion segmentation.

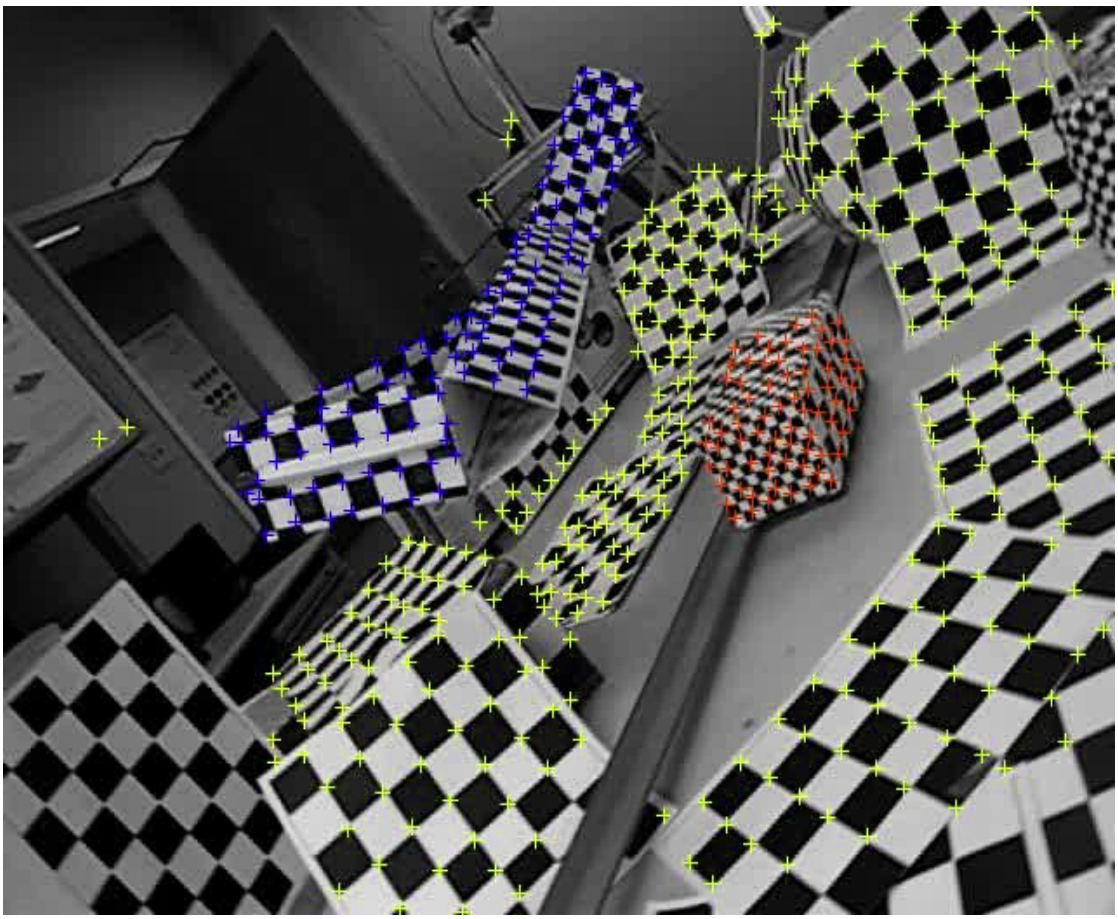


Figure 1.3: A 3 motion checkerboard sequence with tracked points

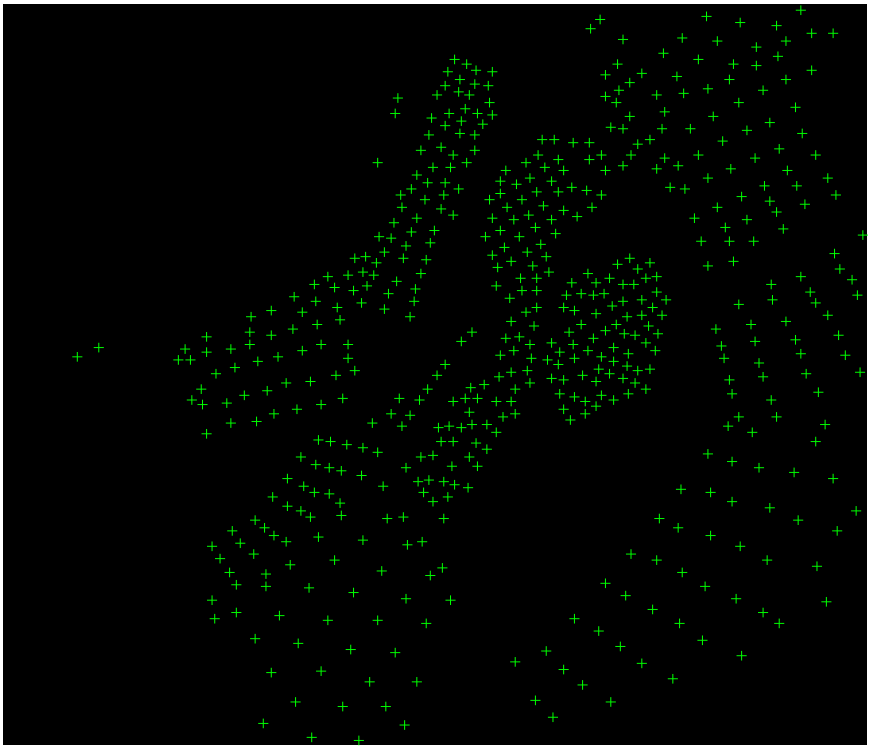


Figure 1.4: The same checkerboard sequence showing only the tracked points

Since motion segmentation relies solely on the (x, y) image coordinates of the tracked feature points, they need to be tracked reliably over the video frames. [1] is one of the commonly used feature trackers that is based on spatial intensity information. SIFT[2] is a more sophisticated feature tracker that uses a more elaborate set of feature vectors that are invariant to uniform scaling and rotation. There are many other feature trackers such as MSER and SURF. However, all these trackers face the same problem of wrong matching of the features or completely losing the features in some of the frames. The resultant data matrix will therefore likely contain missing entries or large outliers.

The clear motion grouping in figure 1.3 may convey the impression that motion segmentation is an easy problem that is readily solvable. Figure 1.4 gives a more accurate picture of the challenge of motion segmentation. Without the benefit of the underlying image and relying on the (x, y) image coordinates alone, motion segmentation looks much more daunting. With just the (x, y) image coordinates, not only is it hard to cluster the tracked feature points, it is even more challenging to find out how many motion there are in the scene. If we further take into account the tracking error present in the data matrix, the motion segmentation problem becomes a lot more formidable.

1.1 Contribution

In this thesis, we present a new, general motion segmentation framework that solves both the model selection and clustering problem. Unlike the present methods, our model selection works universally across both the rigid and the non-rigid domain, with the same set of model parameters. Our work achieves the best model selection and misclassification rates for rigid body motion segmentation.

For non-rigid structure from motion, we propose a new subspace segmentation approach based on our work. Our work is the only work that decomposes the nonrigid motions into their constituent parts automatically, and yet allows reconstruction of a global 3D shape. This allows us to handle greater deformation that might not be representable by a linear subspace. Compared to the local piecewise approaches[3][4][5], our work handles the different types of

non-rigid motion uniformly, without the need for prior information on the type of motion. For reconstruction, those non-rigid structure from motion (NRSFM) works based on shape basis (or related) representation e.g. [6][7][8][9], typically assume the availability of ground truth to achieve the optimum result. We demonstrate that we are able to achieve comparable performance without the ground truth.

There are other works such as [3] that use the same motion segmentation algorithms for both rigid and non-rigid motion. The difference between our work and theirs is that while [3] works well for non-rigid motion, the model selection performance is poor for rigid body motion segmentation. In contrast, our work performs strongly for both rigid and non-rigid motion.

Our work is also more general and less restrictive in the sense that both the articulated and deformation type of non-rigid motion are handled uniformly. This is in contrast to [3], which handles mainly articulated motions with rigid components.

1.2 Model selection difficulties

There has been a steady progress in motion segmentation ever since the seminal single body rigid structure from motion factorization work [10] allows various extensions to multi-body rigid motion segmentation. Due to the difficulty of motion segmentation, the overwhelming majority of works assume known number of motion. These motion segmentation works avoided model selection for a good reason. Model selection is inherently a difficult problem that needs the incorporation of prior knowledge. The challenge is often in the in-

corporation of this prior knowledge.

For the motion segmentation problem, overlapping motions remains a largely ignored issue. Almost all the motion segmentation works assume independent motions. This is a strong and restrictive assumption that may not hold in many of the data sequences. Overlapping motions are likely to be prevalent in articulated motions and more generally non-rigid motions.

Even for rigid motions, overlapping motions are also common. An intuitive example is the set of traffic sequences, where the cars are constrained to move along the same road, thereby sharing the same translation . Another example is the camera induced motion, which will impart the same motion to all the moving bodies.

The use of the independent subspace assumption is understandable because it simplifies the mathematical treatment and allows the use of spectral clustering as an effective segmentation tool. In many of the motion segmentation works, clustering is based on the pairwise subspace affinity between the trajectories. The overlapping motion will cause trajectories from the overlapping subspaces to have significant affinity. The use of spectral clustering ensures that the affinity between trajectories from overlapping subspaces can be treated as noise so that segmentation will be successful.

Although the use of spectral clustering for segmentation works well by regarding the overlap as noise, the situation is more serious for model selection. In spectral clustering, the number of motion is given by the number of zero eigenvalues of the Laplacian. Model selection in the spectral domain typically works by identifying the gap between

the zero and non-zero eigenvalues. The better performing methods, such as [11] and [12], are based on this principle.

The common problem these methods face is the diminishing of this eigengap due to the overlapping motion. [11] compensates for this increased ambiguity by the use of a more robust fitting function. [13] introduced a model complexity penalty to incrementally merge the over-segmented components. The idea of model complexity has been around for a while(see [14][15]), but it has only been introduced to motion segmentation recently.

Instead of treating the overlapping motion as noise and fixing this assumption a posteriori, our work models the overlapping motion upfront and make explicit the overlap. In this thesis, we show that such an approach not only gives better results, but also provides the explicit overlapping information that is desirable for NRSFM.

1.3 Minimal basis representation

At the heart of our new method is the idea of minimal basis representation. It is this minimal basis representation that allows the modeling of overlapping subspaces. This parsimonious principle is motivated by the fact that most of the real life motion sequences we deal with consist of overlapping motion. In view of the overlapping motion, the minimal basis representation is therefore seeking to find the smallest set of basis that is able to explain these overlapping subspaces. Finding the minimal basis representation is a challenging task. With the recent advances in compressive sensing and graphical model, we are able to leverage on these tools to find such minimal

basis representation.

1.4 Organization

Following this introduction chapter, we lay the foundation for our work in chapter 2. In chapter 3, we present the full treatment of our proposed new work in the context of rigid body motion segmentation, with detailed description of experiment setup, comparisons, results and analysis. In chapter 4, we outline how the same model selection mechanism can be applied to non-rigid structure from motion. Detailed experiment setup, comparisons, results and analysis are also included in this chapter. We end the thesis in chapter 5 with conclusions and some suggestions on future works.

Chapter 2

Foundation

In this chapter, we lay the foundation for our main work with the introduction of various topics that our work is built upon. We start off by reviewing two important mathematical techniques at the heart of our work - convex optimization and factor graphs.

Convex optimization and in fact, optimization in general, now plays a central role in many computer vision problems. In our work, convex optimization is important in finding the minimal basis representation solution efficiently and accurately. Compared to the ubiquitous convex optimization, the factor graph formulation of graph model is only used sporadically in computer vision. For our work, we rely on factor graphs for model selection.

These two sections are not meant as an exhaustive guide or mathematically rigorous proofs to solving convex optimization problems and message passing in factor graphs. Instead, we hope to provide some background and intuition on these techniques, so that the main thrust of our work in the latter sections can be better understood.

In the last section, we first do a quick review of orthographic

and affine camera models. We demonstrate through the commonly used affine camera model how the camera models lead naturally to the factorization framework. One particular form of factorization that is highly relevant to our work is the self-expressive representation idea presented in Sparse subspace clustering(SSC)[16] and Low Rank Representation(LRR)[11], which are important state-of-the-art motion segmentation works. Besides their strong performance in misclassification rates, they are accompanied by elegant theoretical results guaranteeing the correctness of their solution.

2.1 Convex optimization

Ever since the pioneering influential work of [17], the use of convex proxy for finding an approximate solution to NP-hard combinatorial problems in the field of compressive sensing has seen tremendous growth and applications. From this initial work, subsequent works such as [18] and [19] have profound impact on many computer vision problems. The use of convex proxies means that many of these problems can be solved efficiently and accurately. More importantly, these compressive sensing works provide the theoretical framework establishing when the solution to the convex proxies coincides with the original problem. Many computer vision problems deemed too difficult in the past can now be solved with good approximations using the convex proxies. The motion segmentation problem is one of the beneficiaries of this progress in compressive sensing.

Our minimal basis formulation requires the use of row sparsity penalty. Even though the use of convex proxies makes such formula-

tion more tractable, we still require an efficient algorithm for solving the resultant optimization program. Although there are fast and efficient off-the-shelf solvers such as CVX[20], the size of our data matrix and matrix norm formulation render these solvers unfit for use. For example, CVX is able to handle matrix sizes 30×90 comfortably whereas our data matrix is typically of the order 60×300. Such big matrices will grind CVX to a halt. One of the reasons for CVX's performance issue with large data matrices is due to the second order optimization method used. While second order optimization methods are able to solve optimization problems accurately, scalability does become an issue.

Our search for an efficient algorithm for solving our optimization problem of interest leads us to the Augmented Lagrange Multiplier(ALM) method. ALM is a first order method in the sense that it is gradient descent based, unlike the second order method where the Hessian needs to be computed.

In the following explanation on ALM, we will make use of the convex proxy formulation in our work (2.1) as a concrete illustration.

$$\begin{aligned} &\min_{C,E} \ \|C\|_{2,1} + \gamma \|E\|_{1,2} \\ &s.t. \quad \widehat{W} = \widehat{W}C + E \end{aligned} \tag{2.1}$$

where $\widehat{W}$ is a given data matrix, C is the coefficient matrix that describes how each trajectory expresses itself in terms of other trajectories, E is the column sparse outlier matrix. The detailed formulation will be given in chapter 3.

2.1.1 Augmented Lagrange multipliers(ALM)

ALM was proposed as far back as 1969 by [21] and [22], known as "method of multipliers". Our insight into ALM is based mainly on [23]. Since our formulation involves only equality constraint, we will explain ALM with equality constraints in mind. ALM with inequality constraints are more difficult. Since we do not deal with inequality constraints, ALM with inequality constraints will not be covered here.

The idea behind ALM is to move the equality constraints into the Lagrangian as quadratic penalties and then gradually increase the weight of these quadratic penalties so that the equality constraints are enforced upon convergence. Note that turning the constraints into quadratic penalties in the objective function would have been a more intuitive approach. The penalized objective function for our case is

$$f_\rho(C, E) = \|C\|_{2,1} + \gamma \|E\|_{1,2} + \frac{\rho}{2} \left\| \widehat{W}C + E - \widehat{W} \right\|^2 \tag{2.2}$$

The idea behind the quadratic penalty approach is that as $\rho \to \infty$, the quadratic penalty becomes binding and becomes the original hard constraint. The important difference, as explained in [23], is that for the Lagrangian's case, the quadratic penalty weight ρ does not need to be increased to infinity for convergence to happen and is therefore numerically more stable.

As shown in [24], the optimum primary variables and dual Lagrangian multipliers form a saddle point so ALM looks to minimize

the primary variables and maximize the dual Lagrange multipliers. Based on this augmented Lagrangian, ALM alternates between solving the primary variables as an unconstrained optimization problem and updating the Lagrange multipliers in each iteration until convergence.

For our problem of interest, the augmented Lagrangian is given by

$$\mathcal{L}(C, E, \lambda) = \|C\|_{2,1} + \gamma \|E\|_{1,2} + \langle \lambda, \widehat{W}C + E - \widehat{W} \rangle + \frac{\rho}{2} \left\| \widehat{W}C + E - \widehat{W} \right\|^2 \tag{2.3}$$

where (C,E) are the primary variables, $\|.\|_{2,1}$ is the row sparsity penalty, $\|C\|_{1,2}$ is the column sparsity penalty, λ is the Lagrange multiplier, $\widehat{W}$ is the known data matrix and ρ is the quadratic penalty weight that is increased successively with iterations.

At the k^{th} iteration, the primary variables (C,E) are updated by minimizing $\mathcal{L}$ over (C,E), with λ^k kept constant

$$(C^{k+1}, E^{k+1}) = \arg\min_{C,E} \mathcal{L}(C, E) \tag{2.4}$$

After solving for (C^{k+1}, E^{k+1}), the Lagrange multiplier is updated based on the optimality condition(KKT condition) as

$$\lambda^{k+1} = \lambda^k + \rho^k (\widehat{W}C^{k+1} + E^{k+1} - \widehat{W}) \tag{2.5}$$

The bulk of the work in ALM is in solving the unconstrained optimization problem in (2.4). The different variants of ALM lies in how (2.4) is solved and the multipliers are updated in (2.5). We follow [25] in describing the variants of ALM.

2.1.2 Closed form solutions

The popularity of ALM is in part due to the availability of closed form solutions for many of the convex penalties in (2.4). Closed form solutions can be obtained for (2.4) by introducing an auxiliary variable J

$$\begin{aligned} \min_{C,E,J} \ & \|J\|_{2,1} + \gamma \|E\|_{1,2} \\ s.t. \quad & \widehat{W} = \widehat{W}C + E \\ & C = J \end{aligned} \tag{2.6}$$

The slight complication is due to the presence of the $\widehat{W}C$ term. The introduction of an auxiliary variable will remove this complication. The price we have to pay for introducing this auxiliary variable will become apparent in the inexact ALM section 2.1.4. With this auxiliary variable, the augmented Lagrangian becomes

$$\begin{aligned} \mathcal{L}(C, E, J, \lambda) = & \|J\|_{2,1} + \gamma \|E\|_{1,2} \\ & + \langle \lambda_1, \widehat{W}C + E - \widehat{W} \rangle + \frac{\rho}{2} \left\| \widehat{W}C + E - \widehat{W} \right\|^2 \\ & + \langle \lambda_2, C - J \rangle + \frac{\rho}{2} \|C - J\|^2 \end{aligned} \tag{2.7}$$

The primary variables update step in ALM thus becomes

$$(C^{k+1}, E^{k+1}, J^{k+1}) = \arg\min_{C,E,J} \mathcal{L}(C, E, J) \tag{2.8}$$

For J, we can transform (2.7) into the following by "completing

the square" and optimize only over J, keeping the rest constant

$$\min_J \frac{1}{2} \|J - M\|_F^2 + \|J\|_{2,1} \tag{2.9}$$

where M results from "completing the square". The reason why we want to rewrite to the form in (2.9) is because a closed form solution will then exist. The closed form solution is the well-known soft thresholding operator. The minimum is given by applying the soft thresholding operator row-wise. Row J_i is updated as,

$$\bar{J}_i = \begin{cases} \left(1 - \frac{1}{\|M_i\|_2}\right) J_i & \text{if } \|M_i\|_2 > 1 \\ 0 & \text{otherwise} \end{cases} \tag{2.10}$$

When minimizing over E, (2.7) is similarly transformed into

$$\min_E \frac{1}{2} \|E - D\|_F^2 + \gamma \|E\|_{1,2} \tag{2.11}$$

where D results from completing the square, the minimum is given by applying the soft thresholding operator column-wise. Column E_i is updated as,

$$\overline{E_i} = \begin{cases} \left(1 - \frac{\gamma}{\|D_i\|_2}\right) E_i & \text{if } \|D_i\|_2 > \gamma \\ 0 & \text{otherwise} \end{cases} \tag{2.12}$$

Since C does not involve any non-smooth terms, it can be differentiated and solved in a straightforward manner.

2.1.3 Exact ALM

The exact ALM carries out the unconstrained minimization (2.8) by alternating between minimizing J, E and C using the closed form solutions until convergence. This convergence requirement usually takes many iterations, thus causing considerable slow down. The Lagrange multipliers are only updated upon convergence of these primary variables.

2.1.4 Inexact ALM

The inexact ALM is much faster compared to the exact version. Inexact ALM is solved using the alternating direction method of multipliers (ADMM) method, proposed by [26] and[27]. The main difference from exact ALM is that we no longer require convergence of the primary variables before updating the Lagrange multipliers. ADMM simply updates the primary variables once in the unconstrained minimization step (2.8) and then immediately updates the Lagrange multipliers.

When there are only two variables, convergence for ADMM can be proven if the following two assumptions hold

1. The objective functions are closed, proper and convex
2. The (un-augmented) Lagrangian has a saddle point

For our case, with the introduction of the third auxiliary variable, convergence is no longer a given. In this thesis, we have experimented and compared inexact ALM with exact ALM and APG. Inexact ALM

is the clear winner in terms of speed without sacrificing accuracy or running into convergence issue.

2.1.5 Accelerated Proximal Gradient

Instead of the alternating minimization strategy in exact ALM, Accelerated Proximal Gradient(APG) is a fast and powerful first order unconstrained minimization method that can be used. In both exact and inexact ALM, an auxiliary variable J needs to be introduced to handle the compounded C variable in the term $\frac{\rho}{2}\left\|\widehat{W}C + E - \widehat{W}\right\|^2$ in (2.3), where C is pre-multiplied by $\widehat{W}$. The close form solution only applies only for the simple form of C described in section 2.1.2.

APG is a majorization-minimization(MM) method that simplifies the compounded variables by expressing these variables in terms of the proximal gradients. MM methods works on the principle of replacing a difficult objective function with a simpler, majorizing function. At each iteration, this majorizing function is estimated and minimized until convergence. [28] has a good intuitive explanation on the majorization idea.

The APG method was proposed in [29] for solving unconstrained minimization problems involving vectors. [30] extended APG to work with matrices. In both these works, the smooth part of the objective function is majorized by a quadratic function. As long as a function is Lipschitz continuous, we can always find a Lipschitz constant so that the function is upper-bounded by a quadratic function. We will illustrate this idea with (2.3). First define the smooth part of the augmented Lagrangian as

$$f(C,E) = \langle \lambda, \widehat{W}C + E - \widehat{W} \rangle + \frac{\rho}{2} \left\| \widehat{W}C + E - \widehat{W} \right\|^2 \tag{2.13}$$

Since $f(C,E)$ in (2.13) is a quadratic function, we can do a Taylor expansion about the current estimate (C^k, E^k) up to the quadratic terms. If the Lipschitz constants are available, then $f(C,E)$ can be upper-bounded without involving the Hessian

$$\begin{aligned} f(C,E) \leq f(C^k,E^k) + \langle \nabla_C f, C - C^k \rangle + \langle \nabla_E f, E - E^k \rangle \\ + \frac{L_C}{2} \left\| C - C^k \right\|_F^2 + \frac{L_E}{2} \left\| E - E^k \right\|_F^2 \end{aligned} \tag{2.14}$$

where $\nabla_C f$ is the gradient with respect to C, $\nabla_E f$ is the gradient with respect to E, L_C and L_E are the Lipschitz constants for C and E respectively.

With this majorization, the previously compounded C is now replaced by the simple form, so that the known closed form solution can be applied. By completing the square, (2.4) can be minimized independently as

$$\min_C \ \frac{1}{2} \left\| C - G^k \right\|_F^2 + \|C\|_{2,1} \tag{2.15}$$

$$\min_E \ \frac{1}{2} \left\| E - H^k \right\|_F^2 + \|E\|_{1,2} \tag{2.16}$$

where G^k and H^k are the proximal gradients that comes from com-

pleting the square.

The Lipschitz constants can be estimated by applying the Lipschitz condition. We will show how the Lipschitz constant can be estimated in appendix 1.

2.2 Factor graphs

Due to the key role of factor graph in the model selection part of our work, we explain the important aspects of factor graph that are most relevant to our work. This short factor graph tour is based mainly on [31] and [32].

The idea behind factor graph is best explained by the use of a toy example. Consider a probability distribution function defined by the bipartite graph shown in figure 2.1(taken from [32])

$$f(x_1, x_2, x_3) = f_a(x_1, x_2) f_b(x_1, x_2) f_c(x_2, x_3) f_d(x_3) \tag{2.17}$$

Unlike the usual graph, a bipartite graph consists of two distinct kind of nodes, variable and factor(function) nodes. A variable node can only have an edge to a function node and vice versa, a function node can only have an edge to a variable node. Any undirected or directed graph can be readily converted to a factor graph. Note that the relationship between an undirected/directed graph to a factor graph is one to many i.e. there are multiple factor graph representations given a undirected/directed graph.

The factor graph reflects the underlying adjacency graph structure in figure 2.1. Our toy example function f is a product of local

functions f_a, f_b, f_c and f_d. These local functions are functions of a subset of the variables. The factor graph is therefore a bipartite graph that can be decomposed into individual 'factor'.

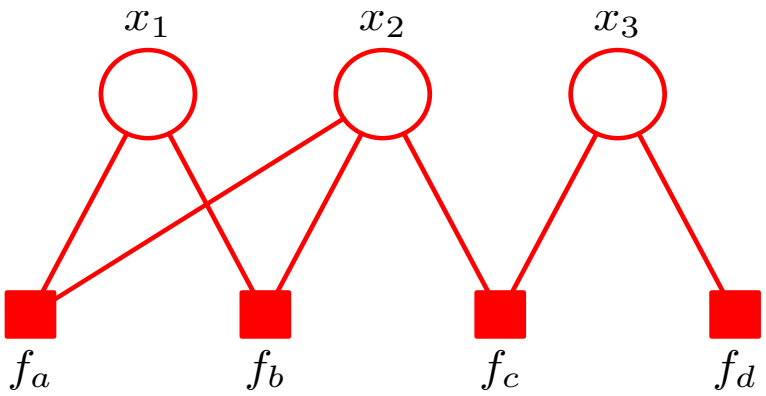


Figure 2.1: Factor graph toy example

2.2.1 Inference

Given a factor graph, we are often interested in making an inference i.e. maximizing the joint probability distribution function representing the factor graph. In our toy example, we are interested in

$$(x_1^*, x_2^*, x_3^*) = \arg \max_{x_1, x_2, x_3} f(x_1, x_2, x_3) \tag{2.18}$$

For our toy example, this maximization of the joint distribution can be expressed as

$$f(x_1^*, x_2^*, x_3^*) = \max_{x_1} \max_{x_2} \max_{x_3} f(x_1, x_2, x_3) \tag{2.19}$$

$$= \max_{x_1} \max_{x_2} \max_{x_3} f_a(x_1, x_2) f_b(x_1, x_2) f_c(x_2, x_3) f_d(x_3) \tag{2.20}$$

The key step for this joint distribution maximization is the exchanging of the max and product operators

$$f(x_1^*, x_2^*, x_3^*) = \max_{x_1} \max_{x_2} f_a(x_1, x_2) f_b(x_1, x_2) \max_{x_3} f_c(x_2, x_3) f_d(x_3) \tag{2.21}$$

[31]and [32] show that the evaluation of (2.21) can be achieved using the max-product algorithm. The max-product algorithm involves finding the max of the product of the messages(described in subsection 2.2.2) arriving at each node, and then sending this message to the parent node. This message passing is repeated for each and every node(both variable and function).

Due to the numerical instability arising from the products of small probabilities, it is often more convenient to work with the logarithm of the joint distribution, giving rise to the max-sum algorithm.

2.2.2 Message passing

At the heart of inferencing a factor graph is the concept of message passing. In this section, we explain in more detail the idea of message passing. Message passing can be best understood as the marginal of a node. The usual marginal of a node x is obtained by summing the

joint distribution over all other variables except x. Since we work with the max operator when inferencing a factor graph, the marginal is instead derived by taking the max over all other variables except x, instead of the sum.

There are two type of messages, one from a factor node to a variable node and the other from a variable node to factor node. We first consider the message from a factor node to the variable node x shown in figure 2.2. The tree structure of the graph allows the factors in the joint distribution to be partitioned into groups, with each of the neighboring factor nodes of x forming a group. We first consider the subgroup colored in cyan in figure 2.2. Let $F_s(x, X_s)$ represent the product of all the factors in the group associated with f_s, where X_s are the set of all variables in the subtree connected to x via f_s.

Since the factors in $F_s(x, X_s)$ is also described by a factor (sub)graph, $F_s(x, X_s)$ can be written as the joint probability

$$F_s(x, X_s) = f_s(x, X_s) \prod_{m \in \text{ne}(f_s) \backslash x} \mu_{x_m \to f_s}(x) \tag{2.22}$$

where $\text{ne}(f_s) \backslash x$ denotes the neighbors of x excluding x and μ_{x_m} is the message arriving at the function node f_s.

We regard the marginal of $F_s(x, X_s)$ as the message sent from f_s

to variable node x, with the marginal given by

$$\mu_{f_s \to x}(x) = \max_{X_s} F_s(x, X_s) \tag{2.23}$$

$$= \max_{X_s} f_s(x, X_s) \prod_{x_m \in \mathrm{ne}(f_s) \setminus x} \mu_{x_m \to f_s}(x) \tag{2.24}$$

Due to practical numerical consideration outlined in the section above, message passing is usually performed in the logarithm domain. The switch to the logarithm domain can be effected by taking the logarithm of f_s and changing all products to summations. The message from f_s to x in the logarithm domain thus becomes

$$\mu_{f_s \to x}(x) = \max_{X_s} \left\{ \ln f_s(x, X_s) + \sum_{x_m \in \mathrm{ne}(f_s) \setminus x} \mu_{x_m \to f_s}(x) \right\} \tag{2.25}$$

The message from a variable node x_m to a factor node f_s, as illustrated in figure 2.3, is simpler since it involves only one variable node. The message can be derived similarly as above

$$\mu_{x_m \to f_s}(x_m) = \sum_{f_l \in \mathrm{ne}(x_m) \setminus f_s} \mu_{f_l \to x_m}(x_m) \tag{2.26}$$

The max operation is therefore trivial and can be simplified to just a sum of the messages from other factor nodes.

(2.25) and (2.26) are the final message updates that are used in the implementation.

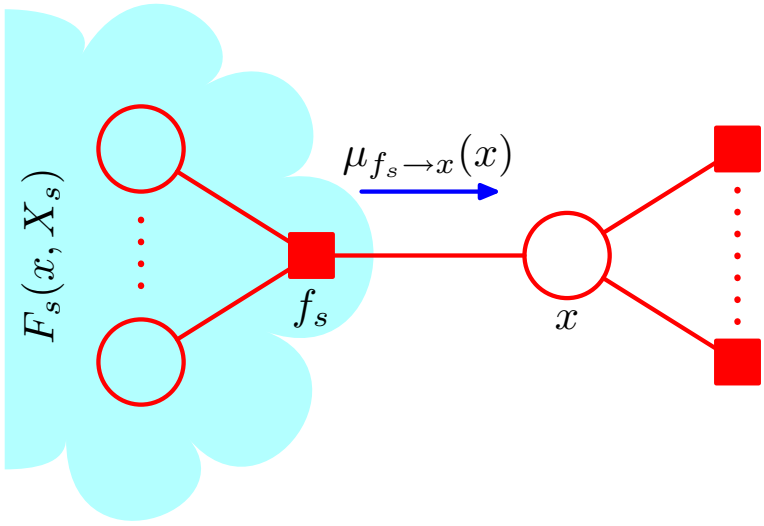


Figure 2.2: Message from function node to variable node

2.2.3 Convergence

The message updates (2.25) and (2.26) are repeated until convergence. However, there is no guarantee that all the messages will converge when there are loops in the graph. Although there is so far no theoretical guarantee, works such as [33] and [34] use damped update as a practical way to deal with this convergence issue. After computing the message update for the current iteration k, the next iteration $k+1$ message update is set to a weighted combination of the previous and current message value

$$\mu^{k+1}_{f_s \to x}(x) \leftarrow \lambda \mu^{k}_{f_s \to x}(x) + (1-\lambda)\mu^{k+1}_{f_s \to x}(x) \tag{2.27}$$

$$\mu^{k+1}_{x_m \to f_s}(x_m) \leftarrow \lambda \mu^{k}_{x_m \to f_s}(x_m) + (1-\lambda)\mu^{k+1}_{x_m \to f_s}(x_m) \tag{2.28}$$

where $0 \leq \lambda < 1$ is the damping factor. [33] and [34] both use a fairly large damping factor of $\lambda = 0.9$.

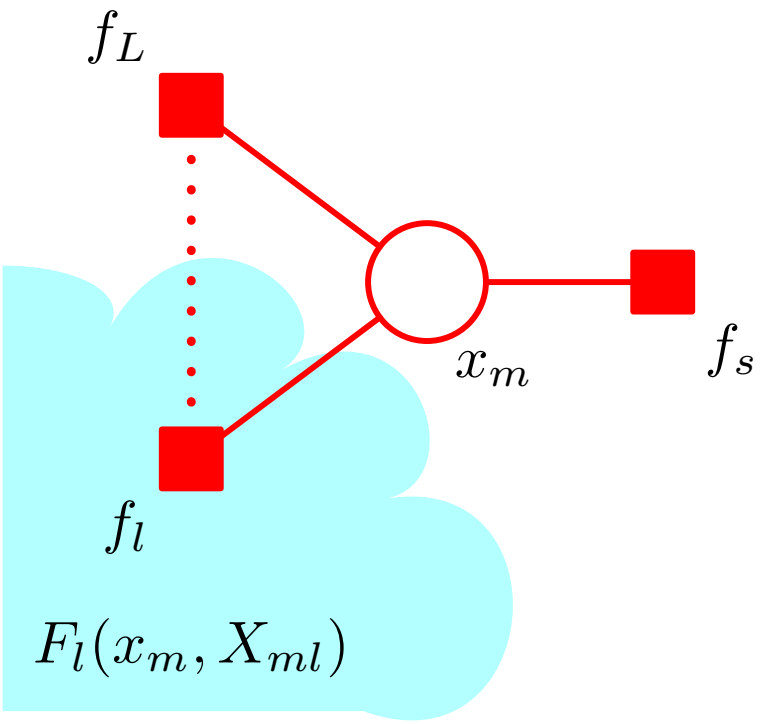


Figure 2.3: Message from variable node to function node

2.3 Factorization and self-expressive representation

2.3.1 Camera model

2.3.1.1 Orthographic camera model

Due to the ubiquitous assumption of the orthographic camera model in NRSFM, we first give a brief description of this orthographic camera model based on [35]. The orthographic camera model is the simplest camera model that takes a 3D point in homogeneous coordinate (X,Y,Z,1) to image point (X,Y,1), simply dropping the Z coordinate. The model can be represented by the homogeneous projection matrix

$$P = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.29}$$

For the case of the camera under going rotation given by the matrix $R \in \mathbb{R}^{3\times3}$ and translation $t \in \mathbb{R}^3$, then the orthographic projection is given by

$$P = \begin{bmatrix} r_1 & t_1 \\ r_2 & t_2 \\ 0 & 1 \end{bmatrix} \tag{2.30}$$

where r_1, r_2 are the first two rows of the rotation matrix R and t_1, t_2 are the first two components of the translation t. We can understand the orthographic camera model as the camera undergoing a rigid transformation only on the XY plane with the Z-axis of the camera and the 3D world coordinate reference frame being approximately aligned.

2.3.1.2 Affine camera model

Due to the use of the affine camera model in many of the motion segmentation works, we take a more detailed look at the affine camera model formulation and its extension to multiple rigid-body motions. We will show how the affine camera model leads to the factorization of the data matrix into the motion and structure components.

After the most faithful perspective camera model, the affine cam-

era is the next best camera model, generalizing orthographic, weak perspective and paraperspective camera models. The affine camera model can be described by the projection

$$\begin{bmatrix} x \\ y \end{bmatrix} = K \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{2.31}$$

where $[x\ y]^T$ is the projected image point, $\mathbb{R}^{3\times3} \ni K$ is the intrinsic camera matrix, $\mathbb{R}^{3\times3} \ni R$ is the rotation matrix, $\mathbb{R}^3 \ni t$ is the translation vector and $[X\ Y\ Z\ 1]^T$ is the 3D point in homogeneous coordinates.

If we let

$$M = K \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} \tag{2.32}$$

then the affine projection model can be written as

$$\begin{bmatrix} x \\ y \end{bmatrix} = M \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{2.33}$$

Now we look more closely at this motion matrix M. Let

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} \tag{2.34}$$

$$= \begin{bmatrix} R_1 & t_1 \\ R_2 & t_2 \end{bmatrix} \tag{2.35}$$

where R_1 and R_2 are the first two rows of the rotation matrix while t_1 and t_2 are the first two components of the translation vector. Subsequent pre-multiplication of A by the intrinsic camera matrix K results in rows of M being a linear combination of the rows of A. Due to this linear combination, the pairwise row orthogonal property of the rotation matrix no longer holds.

2.3.2 Factorization

We first consider factorization for a single rigid body and illustrate the factorization framework using the affine camera model covered in section 2.3.1.2. The given data matrix $W \in \mathbb{R}^{2F \times N}$ is formed by stacking pairs of rows of the (x, y) coordinates of N tracked image points over F frames:

$$W = \begin{pmatrix} x_{11} & x_{12} & \dots & x_{1N} \\ y_{11} & y_{12} & \dots & y_{1N} \\ \vdots & \vdots & \ddots & \vdots \\ x_{F1} & x_{F2} & \dots & x_{FN} \\ y_{F1} & y_{F2} & \dots & y_{FN} \end{pmatrix} \tag{2.36}$$

Based on the affine projection model, the data matrix can now be factorized as

$$W = MS \tag{2.37}$$

where $M \in \mathbb{R}^{2F \times 4}$ is now the motion matrix over F frames, comprising of individual frame motion matrices

$$M = \begin{pmatrix} M_1 \\ \vdots \\ M_F \end{pmatrix} \tag{2.38}$$

$S \in \mathbb{R}^{4 \times N}$ is the structure matrix in homogeneous coordinates and thus have the form

$$S = \begin{pmatrix} X_1 & \dots & X_N \\ Y_1 & \dots & Y_N \\ Z_1 & \dots & Z_N \\ 1 & \dots & 1 \end{pmatrix} \tag{2.39}$$

This factorization is not unique, since given any invertible $Q \in \mathbb{R}^{4 \times 4}$, $W = (MQ)(Q^{-1}S)$ is also a valid factorization.

For the case of multiple rigid bodies, we first consider the tracked feature points ordered by motion. We will deal with the more general case of unordered points by introducing permutation matrix. The data matrix is now given by

$$W = [W_1, \ldots, W_m] \tag{2.40}$$

where m is the number of motion. The factorization of W is now written as

$$W = MS \tag{2.41}$$

$$= [M_1, \ldots, M_m] \begin{bmatrix} S_1 & & & \\ & S_2 & & \\ & & \ddots & \\ & & & S_m \end{bmatrix} \tag{2.42}$$

where $M_i \in \mathbb{R}^{2F \times 4m}$ is the motion matrix for the i^{th} motion, $S_i \in \mathbb{R}^{4m \times N}$ being the homogeneous 3D coordinates of the tracked feature points corresponding to the i^{th} and N is the total number of tracked feature points.

For the case of unordered points, we introduce the permutation matrix $\Pi \in \mathbb{R}^{N \times N}$, that swaps the columns of W. The unordered data matrix that we observe is therefore

$$W = [W_1, \ldots, W_m]\Pi \tag{2.43}$$

$$= MS\Pi \tag{2.44}$$

2.3.3 Self expressive representation

We discuss SSC[16][36] and LRR[11][37] in greater detail because our proposed work is highly related to these works. Both SSC and LRR make use of the self-expressive property of the data matrix, in the sense that each trajectory can be represented as a linear combination of other trajectories from the same subspace.

2.3.3.1 Sparse subspace clustering

SSC regularizes the linear combination weight(or coefficient) with a sparsity penalty, thus ensuring that each trajectory uses a small number of neighbors in the same subspace for representation. Assuming the presence of Gaussian noise, SSC is formulated as

$$\min_{C_i} \ \|C_i\|_1 + \gamma \left\| \widehat{W} C_i - \widehat{W}_i \right\|_2, \quad i = 1 \dots N \tag{2.45}$$

where C_i is column i of the coefficient matrix $C \in \mathbb{R}^{N \times N}$ and W_i is the i^{th} column of the data matrix W. In the absence of noise, theorem 1 in [16] shows that this resultant coefficient matrix C is block diagonal, making it ideal for spectral clustering.

In [36], this sparse representation idea is written in matrix formulation

$$\begin{aligned} \min_{C} \ & \|C\|_1 + \frac{\lambda}{2} \|E\|_F^2 \\ s.t. \quad & \widehat{W} = \widehat{W} C + E \\ & \text{diag}(C) = 0 \end{aligned} \tag{2.46}$$

where E is the error matrix modeling noise in the data. In (2.46),

the error penalty is given in terms of the Frobenius norm, which corresponds to a Gaussian noise model. For sparse noise, the error penalty can be changed to ℓ_1 norm.

The affine camera model can be incorporated by constraining each column of C to sum up to 1, resulting in

$$\min_{C} \ \|C\|_1 + \frac{\lambda}{2}\|E\|_F^2 \tag{2.47}$$

$$s.t. \quad \widehat{W} = \widehat{W}C + E$$

$$\text{diag}(C) = 0$$

$$e^T C = e^T \tag{2.48}$$

where e is a vector of all one's and (2.48) imposes the affine constraint by requiring each column of C to sum up to 1.

Based on the representation matrix C, a symmetric affinity matrix for spectral clustering is defined as

$$A = |C| + |C^T| \tag{2.49}$$

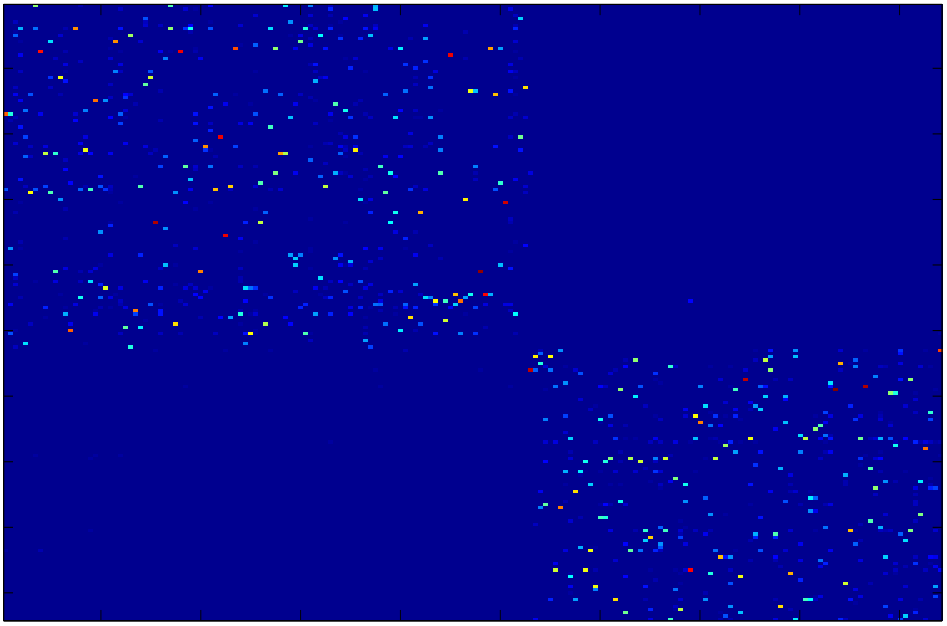


Figure 2.4: SSC representation matrix of the truck1 sequence

[38] further improves on the segmentation result in SSC by imposing spatial distance penalty. This penalty is realized through the weights matrix H in the ℓ_1 norm, resulting in

$$\begin{aligned} \min_{C} \ & \|H \odot C\|_1 + \lambda \|E\|_F \\ s.t. \ & \widehat{W} = \widehat{W}C + E \\ & \text{diag}(C) = 0 \end{aligned} \tag{2.50}$$

where $\odot$ is the Hadamard product or the element-wise matrix product.H is designed so that trajectories that are spatially close are given less penalty.

2.3.3.2 Low rank representation

LRR chooses to impose low rank penalty on the coefficient matrix C and model errors as column sparse outliers

$$\begin{aligned} \min_{C} \ & \|C\|_* + \gamma \|E\|_{2,1} \\ s.t. \quad & \widehat{W} = \widehat{W}C + E \end{aligned} \tag{2.51}$$

The affine constraint can be introduced in a similar manner like SSC. In the absence of noise, theorem 3.1 in [11] also proves that C is block diagonal. While [11] uses the same affinity matrix in (2.49), [37] obtains a better segmentation performance by defining the affinity matrix as

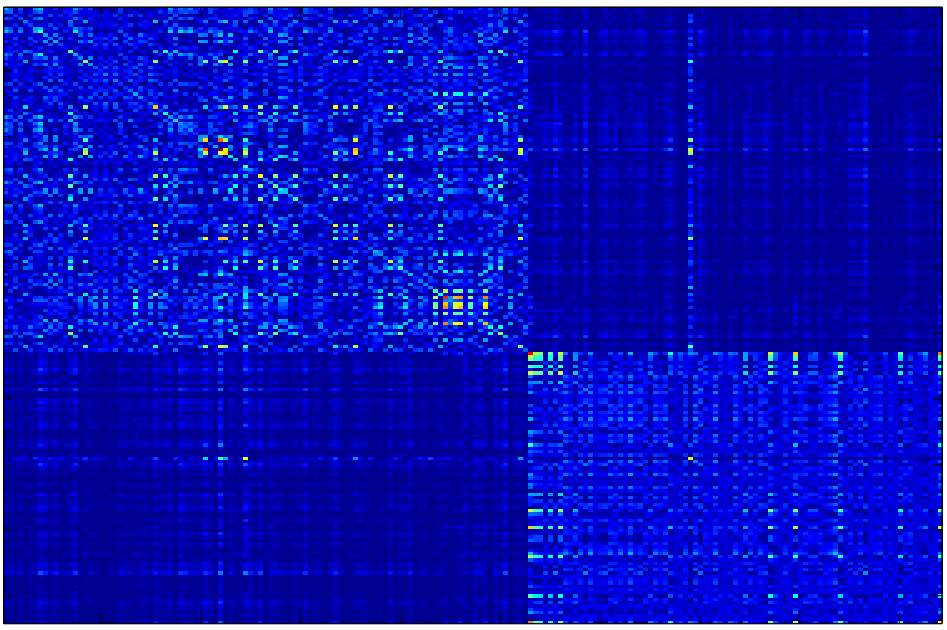


Figure 2.5: LRR representation matrix of the truck1 sequence

$$A = (UU^T)^p \tag{2.52}$$

where U is the column space of C obtained through SVD and p is an even number exponent so that any pairwise affinity value is positive. In [37], p is set to 2 but using the code provided in [39], the best segmentation performance as given in [37] is achieved by setting $p = 4$.

Figure 2.4 and 2.5 show how the different penalty influence the structure of the representation matrix C. For SSC, C is indeed block diagonal. Notably, each block contains sparse number of points due to the ℓ_1 penalty. In contrast, LRR generates a block diagonal C but each block is dense.

For model selection, LRR pursues the same strategy as ORK in counting the number of zero or near zero eigenvalues of the normalized Laplacian matrix constructed from the affinity matrix (2.52). LRR circumvents the difficulty ORK faces in determining zero eigenvalues with a more elaborate approach, mapping the eigenvalues of the normalized Laplacian σ to a soft thresholding operator

$$f_\tau(\sigma) = \begin{cases} 1 & \text{if } \sigma \geq \tau \\ \log_2(1 + \frac{\sigma^2}{\tau^2}) & \text{otherwise} \end{cases} \tag{2.53}$$

where τ is a chosen constant. [37] proposed that the number of motion can then be estimated as

$$N - \text{int}\left(\sum_{i=1}^{N} f_\tau(\sigma_i)\right) \tag{2.54}$$

where int(.) performs a rounding of the input to the nearest integer.

For the state-of-the-art performance reported in [37], the implementation given in [39] chooses τ in a more elaborate, data-dependent manner, rather than just setting it as a constant as stated in [37]. Note that algorithm 1 summarizing how τ is set is not given in [37] but found in the code given in [39].

Algorithm 1 Determining τ for LRR model selection

Input: Given singular values of the normalized Laplacian matrix $\sigma_1, \ldots \sigma_N$

1. Find $\{\hat{\sigma}\}$, the subset of σ in the range $[\max(0.036, \sigma_{N-2}), 0.09]$ i.e. $\max(0.036, \sigma_{N-2}) \leq \hat{\sigma} \leq 0.09$

if the set $\{\hat{\sigma}\}$ has only one element **then**

$\tau \leftarrow 0.08$

else

Find the index k_max corresponding to the maximum eigen gap $g_{max} = (\hat{\sigma}_k - \hat{\sigma}_{k+1})/(\hat{\sigma}_{k-1} - \hat{\sigma}_{k+1}),\ \forall\ \hat{\sigma}_k \in \{\hat{\sigma}\}$

$\tau \leftarrow (\hat{\sigma}_{k_{max}+1} + \hat{\sigma}_{k_{max}})/2$

end if

Output: τ

The model selection algorithm proposed in LRR is different from our model selection mechanism in section 3.4. The main goal of algorithm 1 is to find the largest eigen gap in a robust manner. There is no increased penalty for higher number of motion or increased model complexity. The model complexity cost in section 3.4.2 will ensure higher cost with increased model complexity.

Chapter 3

MB-FLoSS for rigid motion segmentation

3.1 Introduction

In contrast to the current motion segmentation paradigm that assumes independence between the motion subspaces, we approach the motion segmentation problem by seeking the parsimonious basis set that can represent the data. Our proposed method, Minimal Basis Facility Location for Subspace Segmentation(MB-FLoSS), solves for this parsimonious basis representation.

Our MB-FLoSS formulation explicitly looks for the overlap between subspaces in order to achieve a minimal basis representation. This parsimonious basis set is important for the performance of our model selection scheme because the sharing of basis results in savings of model complexity cost. We propose the use of affinity propagation based method to determine the number of motion. The key lies in the incorporation of a global cost model into the factor graph, serving the role of model complexity. The introduction of this global cost

model requires additional message update in the factor graph. We derive an efficient update for the new messages associated with this global cost model.

An important step in the use of affinity propagation is the subspace hypotheses generation. We use the row-sparse convex proxy solution as an initialization strategy. We further encourage the selection of subspace hypotheses with shared basis by integrating a discount scheme that lowers the factor graph facility cost based on shared basis. We verified the model selection and classification performance of our proposed method on both the original Hopkins 155 dataset and the more balanced Hopkins 380 dataset.

We motivate our work by examining the use of spectral clustering [40][41][42] in motion segmentation. Spectral clustering has proven to be an effective and robust clustering method in the motion segmentation literature. Sparse Subspace Clustering(SSC)[16], Low Rank Representation(LRR)[37] and Linear Subspace Spectral Clustering(LSC) [43] use spectral clustering for motion segmentation to achieve excellent results. These methods assume known number of motion when using spectral clustering. Recently, Ordered Residual Kernel(ORK)[12] and LRR extend the use of spectral clustering for model selection, based on the number of zero singular values in the normalized Laplacian. In the presence of noise, this is challenging because the singular values of the Laplacian are seldom zero. In fact, the gap between the supposed zero singular values and the non-zero singular values is often ill-defined. LRR came up with a robust thresholding operator in response to this difficulty and achieved

state-of-the-art performance at 78.06%[1] for the Hopkins 155 dataset, clearly with much room for improvement. This difficulty is better understood when we look at the limitations of spectral clustering below.

The appeal of spectral clustering lies in the use of local pairwise affinity information to derive global eigenvector information for clustering. Even though the construction of the affinity matrix may involve global information, the final affinity matrix only contain local pairwise similarity measure. For example, the nuclear norm regularization that LRR uses is global in nature, but the final self representation matrix describes pairwise trajectory affinity.

In [44], the fundamental limits of spectral clustering are analyzed. The two issues raised are highly relevant in the motion segmentation context. The first concern questions if the local affinity information is sufficient for global clustering. It turns out that local information is insufficient when the data consists of clusters at different scales. The second concern calls into question the use of the first k eigenvectors to find k clusters when confronted with multi-scale and multi-density clusters.

Although these limitations were discussed in the context of classification, they carry over to model selection as well. Recall that model selection in spectral clustering is based on identifying the number of zero singular values. When the complication of multi-scale, multi-density and noise set in, the number of zero singular values is different when the Laplacian is examined at different scale. The difficulty of model selection using spectral clustering can thus be understood

[1]The figure of 77.56% reported in [37] is based on 156 sequences

as ambiguity brought about by multi-scale and multi-density data clusters.

In motion segmentation, multi-scale and multi-density data clusters are very real issues that affect the performance of spectral clustering based methods. Compared to the foreground motion, the background motion tends to contain feature points that span a larger extent of their subspace(due to the greater range of depth and (x, y) location of these points). This leads to multi-scale and multi-density data clusters.

In view of the limitations of spectral clustering, we adopt an alternative paradigm for model selection and segmentation based on global trajectory-subspace distance information. Instead of reducing it to local trajectory-trajectory affinity representation, we generate a set of subspace hypotheses and compute the distance between the trajectories and the subspace hypothesis. With this measure of affinity to subspace hypotheses, model selection is based on the affinity propagation(AP)[34] framework with a judiciously chosen global cost function.

Clearly, there are several motion segmentation works[45][12][13] that are based on trajectory-subspace distance information, but not many of them develop their work for model selection. Kernel Optimization(KO) [13] is a notable exception in that it achieves a good model selection performance. However, KO's random subspace hypotheses generation strategy is different from our work. The subsequent treatment of these subspace hypotheses is also different from our approach. KO merges these subspace hypotheses in a greedy

manner, choosing the pair with the lowest kernel-target alignment at each step.

In section 3.3, we demonstrate how a minimal basis subspace hypotheses set can be generated by requiring the representation matrix to be jointly row sparse. Due to the convex relaxation artefact, the number of subspace hypotheses is far greater than the true number of subspaces. In section 3.4, we show how to incorporate a general model complexity term into the AP framework naturally and efficiently. This model complexity term is important in ensuring that the the right number of subspaces from the hypotheses set are chosen for representation. Although the subspace hypotheses set contains many overlapping subspaces, we still need to ensure the selection of those overlapping subspaces by introducing the facility cost discount scheme. We describe this discount scheme in the same section. In section 3.5, we verify our proposed work on the original and augmented Hopkins dataset, demonstrating a model selection performance significantly better than the state-of-the-art.

Our contribution is three fold. Our first contribution is in the formulation and realization of the minimal basis approach to model selection. Our method is significantly different from the current motion segmentation paradigm that uses spectral clustering. We demonstrate unequivocally the model selection strength of our proposed method.

The second contribution is the recognition, handling and leveraging of possible subspace dependencies. Whereas almost all the current better performing algorithms use subspace independence as

a starting point, treating the overlap as noise, our proposed work properly accounts for subspace dependencies by offering facility cost discount for shared basis. The use of these shared basis subspace for representation has important application in areas such as articulated motion and non-rigid structure from motion.

Lastly, we show how the introduction of a global facility cost function to the AP framework enables model selection with good performance while maintaining efficiency.

3.2 Previous works

3.2.1 Motion segmentation

Majority of the motion segmentation focuses on the classification aspect and assume known number of motion. Without assuming known number of motion, simultaneous estimation of the number of motion and the subsequent classification proves too difficult a problem for early researchers. In this section, we look at the body of important motion segmentation works that assume known number of motion.

The various state-of-the-arts motion segmentation algorithms have their roots in the factorization approach proposed by Kanade[10] for solving the rigid SfM problem. Costeira[46] extended the factorization method to multiple rigid-body segmentation by introducing the shape interaction matrix(SIM) Q, which is proven to have the block diagonal property. Segmentation is based on swapping pairs of rows and columns until Q becomes block diagonal.

Similar to our work, ORK makes use of the orthogonal distance residue of each trajectory to a set of randomly generated subspace hypotheses but differs from our work by converting this distance residue into pairwise affinities. For each trajectory, the residue is sorted in ascending order. The ordered residue kernel between a pair of trajectories is the number of hypothesis overlap in this sorted residue order. These pairwise affinities allow spectral clustering to be used for segmentation.

Generalized Principal Component Analysis(GPCA) is an algebra-geometric method that is supposedly able to segment an unknown number of subspaces of unknown and varying dimensions. GPCA represents the union of subspaces of varying dimension as a set of homogeneous polynomials. These polynomials are differentiated to obtain the basis and dimension of each subspace, thus defining each subspace. Each trajectory is then assigned to the closest subspace in terms of the orthogonal point to subspace distance.

LSA[47] is one of the few segmentation works that takes into account dependent and degenerate motions. Prior to LSA, [48] proved that the data matrix consisting of two overlapping motion will be linearly dependent. The idea behind LSA is for each trajectory to first project the data matrix onto a lower dimension through PCA, and sample the nearest neighbors to estimate its local subspace. The distance between two subspaces is measured by the principal angles. The sum of the square of the principal angles distance is converted to an affinity matrix via the radial basis function. For dependent motion, the trajectories near the intersection constitute the main

source of misclassification. In this case, LSA counts on the data distribution of the trajectories near the intersection of the dependent motions to have more neighbors from the same subspace, otherwise this trajectory will be classified wrongly. In the case of degenerate motion, even though the locally constructed subspace may not span the entire underlying subspace, it will be closer to other locally constructed subspaces from the same motion group.

Linear subspace spectral clustering(LSC)[43] is notable for providing one of the best performances amongst the surveyed methods. The idea in LSC is to seek the best ambient dimension for projecting the data matrix. In LSA, the PCA step simply projects the data matrix onto the upper bound ambient dimension $4m$. The affinity matrix in LSC is constructed from the cosine of the angles between pairs of trajectories. For the best optimal dimension, LSC proposes the relative eigen gap(in ambient dimension D)

$$r_D = \frac{\lambda_m - \lambda_{m+1}}{\lambda_{m-1} - \lambda_m} \tag{3.1}$$

This intuition is that the best choice of D is the one that leads to the best estimation of the number of motion via r_D. Since the number of motion is assumed known, the largest eigen gap will offer the best embedding for segmenting the trajectories. LSC proposes to look for the best ambient dimension in the range $m + 1$ to $4m$. The second factor in the improved performance is the use of the separating exponent α to accentuate the affinity matrix. If A_{ij} is the affinity between trajectory i and j based on the cosine of the angles

between this trajectory pair, then the accentuated affinity is $A_{ij}^{2\alpha}$.

[49] constructs a velocity profile from the data matrix. A non-negative matrix factorization of this velocity profile gives the affinity matrix between trajectories, which then serve as the input for spectral clustering.

[16] makes use of the self-expressive property of the data matrix to represent each trajectory as a linear combination of other trajectories from the same subspace. The use of the ℓ_1 penalty ensures that each trajectory uses only few other trajectories from the same subspace for representation. With the assumption of independent subspace, [16] proves that the ℓ_1 penalty will result in a block diagonal representation matrix, making it ideal for spectral clustering. [36] relaxes the independent subspace assumption and shows that SSC can succeed as long as the overlap is not too excessive. SSC offers strong competitive misclassification rate and is one of the state-of-the-art algorithm.

[38] further improves on the segmentation result in SSC by imposing spatial distance penalty. This penalty is realized by introducing a weighting matrix in the ℓ_1 norm, so that trajectories that are spatially far apart will be penalized more.

[50] raised an important connectivity issue in the SSC generated representation matrix. This connectivity issue is important since spectral clustering depends on connectivity of trajectories from the same subspace. The important result is that for subspace of dimension greater than 3, the SSC generated representation matrix can no longer guarantee block connectivity of trajectories from the

same subspace. This is highly relevant because motion segmentation is based on the affine camera model, which results in subspaces of dimension 4.

Instead of the ℓ_1 penalty, [11] imposes a low rank penalty on the representation matrix. This low rank penalty will also result in a block diagonal representation matrix. LRR is also one of the better performing motion segmentation algorithms. There are various improvements and variants of LRR. [51] proves that the representation matrix is positive semi-definite. [52] shows that in the absence of noise, the representation matrix is in fact the shape interaction matrix from [46].

[53] is one of the rare works that works with the perspective camera model for motion segmentation. The idea is to alternate between motion segmentation assuming known projective depth and estimating the projective depths from the motion segmentation of the trajectories. The projective depths can be solved by [54] and the various extensions such as [55].

3.2.2 Model selection

Kanatani's work [56] is a rare early work that focuses on the model selection aspect of motion segmentation. The idea in [56] is the use of geometric AIC and geometric MDL to balance the singular value truncation residue against a model complexity cost that is a quadratic function of rank r.

ORK derives the Laplacian matrix from the affinity matrix constructed from the pairwise ordered residue kernel and estimates the

number of motion from the number of zero or negligible Laplacian eigenvalues. However, finding a threshold to determine negligible eigenvalues that works across all the data sequences is difficult, resulting in the uncompetitive model selection performance of ORK.

GPCA and LSA propose model selection based on the effective rank detection of the data matrix $\widehat{W}$ inspired by [57]. The effective rank $\bar{r}$ is estimated as

$$\bar{r} = \arg\min_{r} \frac{\lambda_{r+1}^2}{\sum_{k=1}^{r} \lambda_k^2} + \kappa r \tag{3.2}$$

where λ_k is the k^{th} singular value of the data matrix and κ reflects the noise level in the data - the larger the data noise, the higher is the κ value. For GPCA, the data matrix is the embedded data matrix constructed by embedding the data matrix $\widehat{W}$ using the Veronese map. For LSA, the data matrix is just $\widehat{W}$. This effective rank detection method assumes that the non-zero singular values of the data matrix can be easily distinguished from the zero singular values. Due to noise, degenerate and/or dependent motion, the singular value tends to exhibit a continuous smooth spectrum, making it hard to tell the true rank. It is therefore not surprising this method performs poorly.

KO improves the ORK model selection performance by merging a set of over-segmented clusters in a greedy manner and searches for the number of clusters with the largest kernel-target alignment value. KO first over-clusters the data using ORK with a suitably chosen threshold for identifying the zero valued eigenvalues. For a pair of clusters, the discriminative power of multiple kernel learning(MKL) gives an indication of how likely this pair of clusters comes from

the same motion. The kernel-target alignment measure captures the level of discrimination quantitatively. A low kernel-target alignment value means that the cluster pair are difficult to discriminate and therefore likely to be from the same motion. At each step, the average kernel-target alignment is computed and then the cluster pair with the lowest kernel-target alignment value are chosen for merging. After merging all the clusters, the number of motion is determined as the number of clusters where the maximum average kernel-target alignment occurs.

LRR provides the state-of-the-art model selection performance based on a robust way of finding the largest eigen gap. The detail of the model selection algorithm is provided in section 2.3.3.2.

3.2.3 Affinity propagation

Affinity propagation(AP) provides an interesting comparison with spectral clustering. In affinity propagation [34][58], the goal is to look for representative data points called exemplars and cluster the rest of the data points based on similarity to the exemplars. The number of clusters is not specified in AP. Instead, the number of clusters is controlled by the preference value assigned to each data point. The preference value can be regarded as the importance of a data in terms of becoming an exemplar. If a data point has a high preference value, then it has a better chance of becoming an exemplar. As an illustration, suppose the preference value is common across all the data points. If this common preference value is large, a larger number of clusters will emerge. Vice versa, a smaller common preference value

will result in a smaller number of clusters. The affinity propagation clustering method has been applied to image categorization[59] and extended to motion segmentation in FLoSS(Facility Location for Subspace Segmentation)[45] and UFLP(Uncapacitated Facility Location Problem)[33]).

In FLoSS and UFLP, motion segmentation is formulated as an instance of the facility location(FL) problem. FL is known to be NP hard and hence difficult to solve. An approximate solution for FL can be found by performing maximum-a-posteri(MAP) inference in a probabilistic graphical model. In FLoSS, inference is based on the max-product belief propagation(MPBP) algorithm that involves local message passing. MPBP is known to converge to the MAP values of the variables on cycle-free graph. In addition to MPBP, UFLP proposed a linear programming(LP) relaxation based message passing algorithm, known as max-product linear programming(MPLP). The solution from MPLP can be augmented with a greedy algorithm that constructs a solution whose cost is at most three times the optimal for metric UFLP instances, where the customer-facility distance measure satisfies the triangle inequality, thus providing a performance guarantee.

On a related note, [60] formulated two-view motion segmentation as a facility location problem and solve it as a LP problem by relaxing the original facility location problem. Interestingly, the formulation in [60] contains many ideas similar to our work. The data-fitting term in the objective function of [60] measures how well two points comes from the same motion described by the candidate fundamental

matrices. For our case, the data-fitting term measures the residue of how well a point is described by the subspace hypotheses. More interestingly, [60] incorporates model complexity into the objective function. More specifically, [60] models the model complexity cost as a linear penalty. Just like FLoSS, [60] generates candidate fundamental matrices via random sampling.

[58] expands the scope of FL by considering Capacitated Facility Location(CFL). Each facility now has an upper bound on the number of customers it can be assigned to. The increased complexity in the consistency function now poses a potential combinatorial challenge. [58] shows that tractability can be assured by sorting the messages and consider only the top messages related to the facility capacity. The additional message update due to the global cost function in our work is made tractable and efficient by using similar techniques.

Even though both FLoSS/UFLP and our work are based on AP for solving the motion segmentation problem, there are important differences distinguishing the two works. FLoSS/UFLP solves the classification problem assuming known number of motion. Its performance has not been demonstrated on the model selection problem even though, paradoxically, the framework seems to be proposed with this problem in mind. Our proposed work capitalizes on this inherent capability of AP for model selection with the use of a more elaborate facility cost model. Furthermore, our quest for a minimal basis representation drives a more specific subspace hypotheses generation strategy. In FLoSS/UFLP, the subspace hypotheses are generated by random sampling.

[61] analyzed graphical models with high order potentials(HOP), which entails higher order interactions among the discrete variables. A particularly relevant example is the cardinality potential, whose function value is dependent on the number of variables in the subset turned on. The facility cost function we propose in section 3.4.2 is an instantiation of the cardinality function.

3.3 Hypothesis generation with minimal basis subspace representation

3.3.1 Formulation

Our subspace hypotheses generation strategy is based on finding the minimal basis subspace representation for the data matrix. Such parsimonious representation looks for basis common to the overlapping subspaces, thereby reducing the number of basis needed to explain the subspaces. This emphasis on shared basis leads naturally to the joint sparsity formulation (3.3).

As in SSC and LRR, we use the data matrix itself as the dictionary, and propose the following formulation:

$$\begin{aligned} \min_{C,E} \ & \|C\|_{2,0} + \gamma \|E\|_{0,2} \\ s.t. \quad & \widehat{W} = \widehat{W}C + E \end{aligned} \tag{3.3}$$

where $\widehat{W} \in \mathbb{R}^{2F \times N}$ is the data matrix constructed from the tracked feature trajectories, $E \in \mathbb{R}^{2F \times N}$ is the column-sparse error matrix, F is the number of frames, N is the number of tracked feature points,

$C \in \mathbb{R}^{N \times N}$ is the representation matrix, $\|.\|_{2,0}$ counts the number of non-zero rows and $\|.\|_{0,2}$ counts the number of non-zero columns.

3.3.2 Convex relaxation

Due to the combinatorial nature and therefore NP-hard nature of (3.3), we minimize the convex surrogate and model data noise as column sparse outliers, resulting in:

$$\begin{aligned} &\min_{C,E} \ \|C\|_{2,1} + \gamma \|E\|_{1,2} \\ &s.t. \quad \widehat{W} = \widehat{W}C + E \end{aligned} \tag{3.4}$$

where $\|C\|_{2,1} = \sum_{i=1}^{2F} \sqrt{\sum_{j=1}^{N} ([C]_{ij})^2}$ and $\|E\|_{1,2} = \sum_{j=1}^{N} \sqrt{\sum_{i=1}^{2F} ([E]_{ij})^2}$. (3.4) is a constrained convex program that can be solved efficiently by the Augmented Lagrange Multiplier(ALM) method[23]. We solve (3.4) using the Alternating Direction Multiplier Method(ADMM) implementation of the inexact ALM method, as in [37].

Note that our primary motivation for the joint sparsity formulation is to seek the minimal basis representation, whereas in [36], the joint sparsity regularization was introduced to ensure connectivity in the similarity graph generated by encouraging data points from the same subspace to use common representative points from the same subspace. It plays a secondary role so as not to alter the dominance of the ℓ_1 penalty in the objective function.

3.3.3 Over segmentation

While we have made the sharing of the basis evident(see figure 3.1), the relaxation artefact(and noise in the data) means that we cannot make use of this result directly to extricate the number of motions and their dependencies. As can be seen from figure 3.1, the representation matrix contains various artefacts due to the convex relaxation. While the overall two subspace structure is discernible, over segmentation is revealed in the gaps in the rows and the resultant extra rows, making the true number of motion hard to tell. There are in fact 40 subspace hypotheses generated from this convex solution.

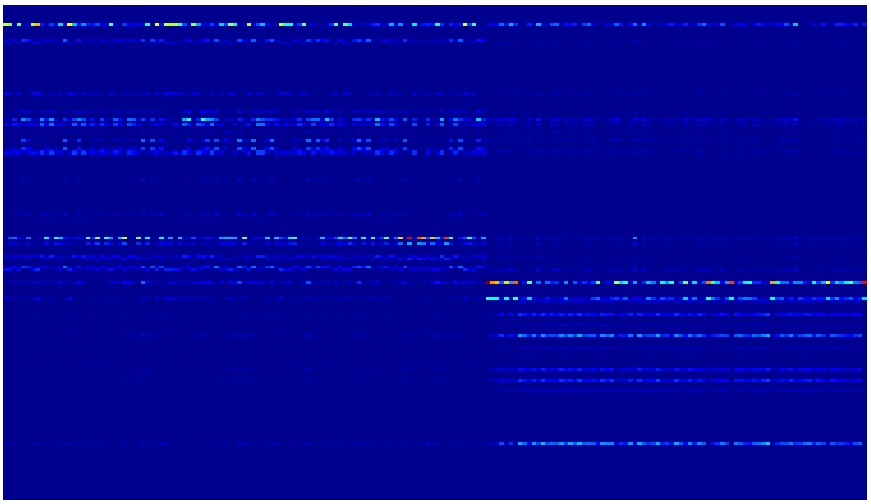


Figure 3.1: Representation matrix of the truck1 sequence

This over-segmentation phenomenon can be explained by the magnitude dependence of the $\|.\|_{2,1}$ penalty. [62] offers an excellent insight and explanation of this magnitude dependence problem in terms of $\|.\|_1$ in SSC. This magnitude dependence of the convex

proxy can be understood by considering the case when the support has large magnitude. For the counting norm $\|.\|_0$, the magnitude is irrelevant; only the support cardinality matters. For the convex proxy $\|.\|_1$, support entries with large coefficients will result in large $\|.\|_1$ value, imposing an unfair penalty. The magnitude dependence of the $\|.\|_1$ function means that trajectories from the same subspace that are nearly orthogonal will be broken up into two groups, since the large coefficients for self-expression will incur large norm penalty. This explanation also applies for the $\|.\|_{2,1}$ penalty. While some of the numerical methods like reweighted ℓ_1[63] might slightly relief the artefact problems, they do not remove the problems.

Despite the preceding comments, we have now at our disposal much more information. Each column of the coefficient matrix proposes a subspace hypothesis and carries with it a notion of AP responsibility message update to this subspace hypothesis. Row wise, the coefficient matrix indicates the importance of the subspace hypothesis, in terms of the number of trajectory that generates the subspace hypothesis. This is reminiscent of the AP availability message update from the facility. See [34] for more detail about the notion of responsibility and availability. This close relationship lends the joint sparse representation matrix well suited for subspace hypothesis generation.

3.4 Model selection

Our proposed cost model, which we term as Minimal Basis(MB)-FLoSS, is based on FloSS[45] but with important extensions. These

extensions are the facility cost model outlined in section 3.4.2 which encodes the "ecological" constraint that multiple motions are likely to be dependent, and the discount scheme in section 3.4.4.2 which ensures that facilities with overlapping basis have lower cost, translating to higher beliefs at these facilities.

Our MB-FloSS method uses the same FLoSS setup and message passing. We thus follow the notations in [64] and [45] in deriving the new message update required by our modified facility cost model.

3.4.1 FLoSS/UFLP

Due to the relevance of FLoSS/UFLP, we give a quick review here. FLoSS/UFLP formulates the facility location problem in terms of factor graph representation(fig. 3.2), consisting of variable nodes and factor nodes. This graphical model results in the following objective function:

$$F(\{h_{ij}\}) = \sum_{ij} S_{ij}(h_{ij}) + \sum_{i} I_i(h_{i:}) + \sum_{j} f_j(h_{:j}) \tag{3.5}$$

The variables nodes $h_{ij}, i = 1, \ldots, N, j = 1, \ldots, M$, are binary variables that indicate if customer(trajectory) i uses(belongs) to facility(subspace) j, where N is the number of customers and M is the number of facilities. The factor nodes evaluate potential functions over the variable nodes they are connected to.

There are three factor potential functions in FLoSS/UFLP. I_i enforces the constraint that one customer chooses one and only one facility. The notation $h_{i:}$ refers to the subset of binary variables connecting customer i to all the facilities from 1 to M. Similarly, the notation $h_{:j}$ refers to the subset of binary variables connecting all

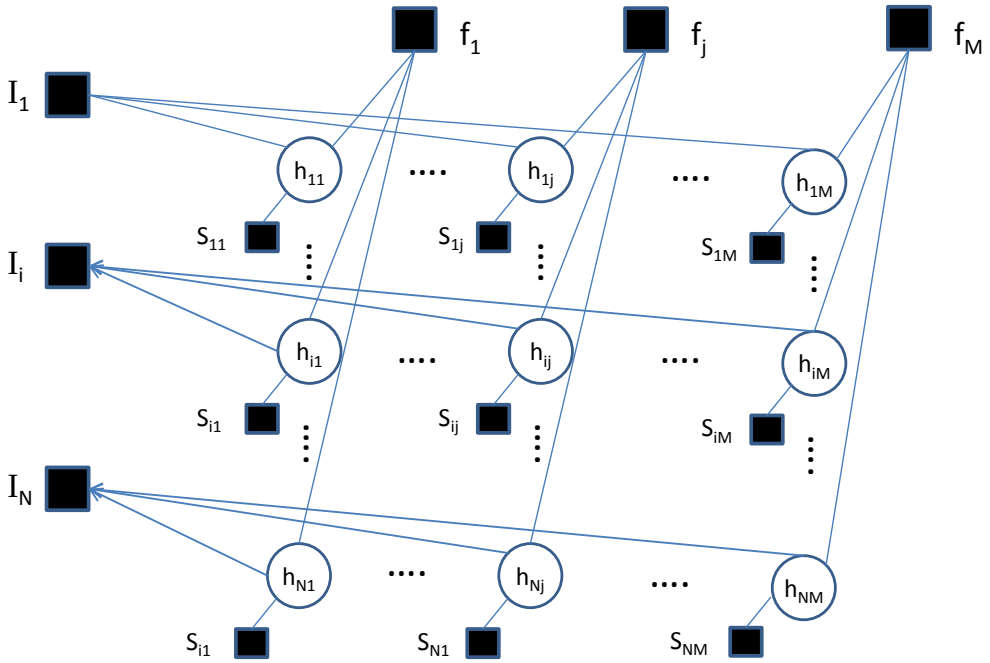


Figure 3.2: FLoSS factor graph representation

the customers from 1 to N to facility j. S_{ij} describes the distance between customer i and facility j. f_j describes the cost when facility j is turned on. Upon convergence of the message update, the binary variables $\{h_{ij}\}$ are turned on if the sum of the messages arriving at the variables are non-negative.

3.4.1.1 Local facility cost

Due to the key role of facility cost, we describe the FLoSS facility cost model so as to provide a contrast to our proposed cost model. In FLoSS, the subspace hypotheses are generated as random subsets of two, three and four trajectories, thus taking into consideration degenerate subspaces. The cost of a facility is set to be the sum of

all pairwise distances between the trajectories forming the subspace. This local cost primarily serves to balance the tendency towards the higher dimensional subspace hypotheses, since higher dimensional subspace hypotheses are able to fit the data better compared to the lower dimensional subspace hypotheses.

Unfortunately, this local cost model does not capture the actual nature of the problem very well, often resulting in the wrong number of facilities being opened. In fact, in FLoSS/UFLP, the number of motion is assumed to be known. Thus they can merge excess number of facilities opened or increase the number of facilities opened by iteratively scaling down the local cost across all facilities.

3.4.2 MB-FLoSS facility cost

To address the aforementioned shortcomings, the facility cost function we propose is a global function in the sense that it is a function of the cardinality of the number of facilities opened. Given an upper bound K on the number of motion, we propose a power law facility cost model

$$\mathcal{C} = \begin{cases} ak^p & \text{if } k \text{ facilities are opened, for } k = 1 \text{ to } K \\ \infty & \text{otherwise} \end{cases} \tag{3.6}$$

where $\mathcal{C}$ is the facility cost function and a, p are constants. Note that $\mathcal{C}$ is a monotonic increasing function of the number of opened facilities. We denote the cost of opening k facilities as $\mathcal{C}_k$. This power law cost model is motivated by the observation that in real life scenes, the larger the number of motions, the more unlikely it

is for all of them to be independent. In other words, it reflects not only the cost of increasing complexity with more models, but also the "surprise" of seeing all of them independent from one another. This cost/surprise is only attenuated if there are dependencies between the multiple motions, which will be taken care of by the discount scheme in section 3.4.4.2.

With the global facility cost function (3.6), the factor graph representation needs to be modified, as shown in figure 3.3. The facility cost potential function is now connected to the binary variables $\{e_j\}$. The number of facilities turned on is indicated by the number of $\{e_j\}$ nodes set to 1. The facility cost function $\mathcal{C}$ is therefore a function of $\{e_j\}$. This change will now necessitate message passing involving $\{e_j\}$, reflected in figure 3.4

3.4.3 Objective function

The one customer-one facility constraint remains:

$$I_i(h_{i:}) = \begin{cases} 0 & \text{if } \sum_j h_{ij} = 1 \\ -\infty & \text{otherwise} \end{cases} \tag{3.7}$$

The consistency constraint that ensures that if a customer chooses a facility, the facility gets turned on, also stays:

$$E_j(h_{:j}, e_j) = \begin{cases} 0 & \text{if } e_j = \max_i h_{ij} \\ -\infty & \text{otherwise} \end{cases} \tag{3.8}$$

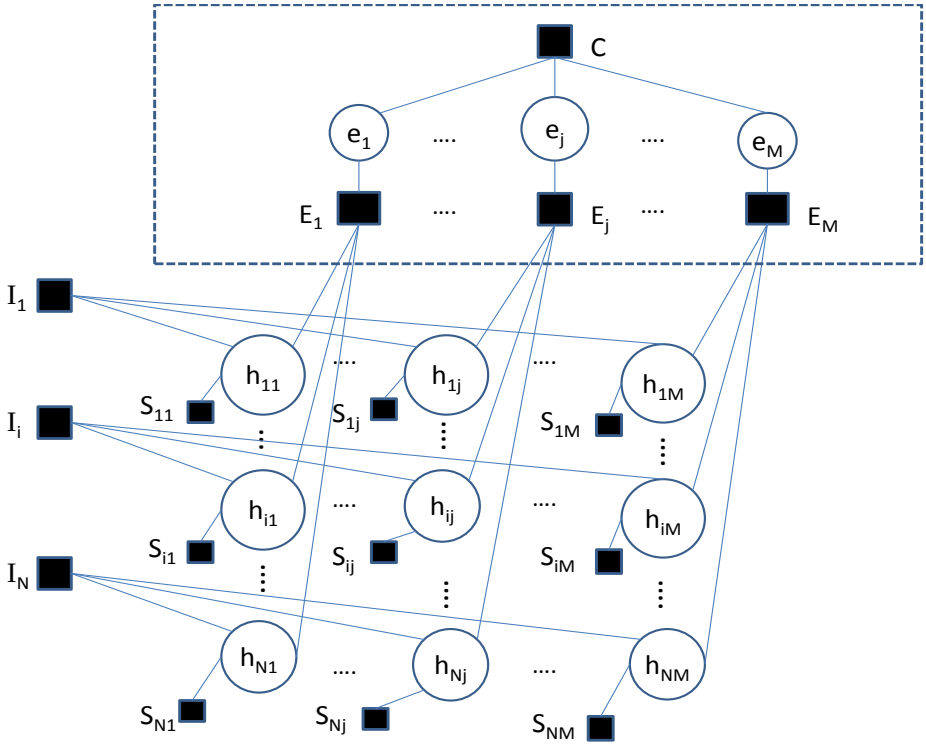


Figure 3.3: MB-FLoSS factor graph representation. The nodes in the upper rectangular box are extensions to the original FLoSS

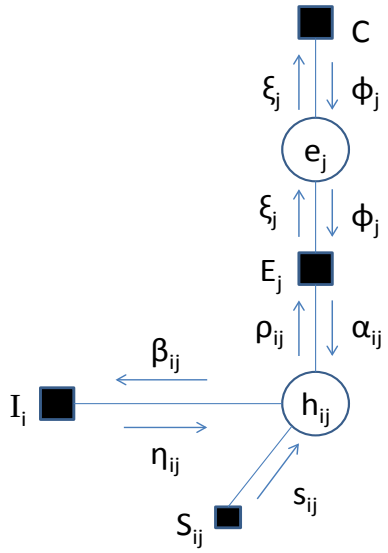


Figure 3.4: MB-FLoSS factor graph messages

The objective function to be maximized is now

$$F(\{h_{ij}\},\{e_j\}) = \sum_{ij} S_{ij}(h_{ij}) + \mathcal{C}(\{e_j\}) + \sum_i I_i(h_{i:}) + \sum_j E_j(h_{:j}, e_j) \tag{3.9}$$

3.4.4 Message passing

Since we are dealing with binary variables $\{h_{ij}\}$ and $\{e_j\}$, it appears that we need to send two-valued messages between nodes. As pointed out in [58], we only need to propagate the difference between the message values for its two possible settings. When the message passing terminates, the estimated MAP settings for each binary variable is recovered by summing all of its incoming messages. Each binary variable is set to 1 if the sum of all the incoming messages is non-negative, and 0 otherwise.

The message passing not involving $\{e_j\}$ remains the same as in FLoSS. For more detail on those messages, please refer to [33][64][65][58]. The message update for ϕ plays an important role in our work and is explained below. The message updates for ξ and α are covered in appendix B.

3.4.4.1 Message update for ϕ

Recall that we only need to send the difference between the message values corresponding to the two different settings $e_j = 0$ or $e_j = 1$. We use the notation $\phi_j(0)$ as a short hand for $\phi_j(e_j = 0)$ and similarly $\phi_j(1)$ for $\phi_j(e_j = 1)$. The message to be sent is then

$$\phi_j = \phi_j(1) - \phi_j(0) \tag{3.10}$$

where

$$\begin{aligned}\phi_j(1) &= \mu_{\mathcal{C}\to e_j}(1) \\ &= \max_{e_k, k\neq j} \left[-\mathcal{C}(e_1, .., e_j = 1, .., e_M) + \sum_{k\neq j} \xi_k(e_k) \right]\end{aligned} \tag{3.11}$$

$$\begin{aligned}\phi_j(0) &= \mu_{\mathcal{C}\to e_j}(0) \\ &= \max_{e_k, k\neq j} \left[-\mathcal{C}(e_1, .., e_j = 0, .., e_M) + \sum_{k\neq j} \xi_k(e_k) \right]\end{aligned} \tag{3.12}$$

Here we follow the same notation as chapter 2 to denote the message from the factor node $\mathcal{C}$ to the variable node e_j as $\mu_{\mathcal{C}\to e_j}$.

For (3.11), since e_j is set as 1, we are looking for the max over one, two $, \ldots, K-1$ other e_j's being turned on. For (3.12), since e_j is kept fixed as 0, we are then looking for the max over one, two $, \ldots, K$ other e_j's being turned on.

Even though (3.11) and (3.12) look combinatorial, the messages can be simplified and updated efficiently. Leveraging on the insights offered by [58], we observe that finding the max can be achieved by evaluating the sorted set $\hat{\xi}$ and the associated facility cost over the K upper bound number of facilities, where $\hat{\xi}$ is obtained by sorting

$\{\xi_j = \xi_j(1) - \xi_j(0), j = 1, \ldots, M, j \neq k\}$ in descending order. More details can be found in appendix B.

For ease of notation, we introduce the cumulative sum operator:

$$S_{ij} = \sum_{k=i}^{j} \hat{\xi}_k \tag{3.13}$$

where $\hat{\xi}_k$ is the k^{th} element in the sorted set $\hat{\xi}$. Denote the cost difference between opening i and j number of facilities as

$$\delta_{ij} = \mathcal{C}_i - \mathcal{C}_j \tag{3.14}$$

For the case of $K = 4$, which is the upper bound used in this chapter, the message update for ϕ_j in (3.10) can be shown to be

$$\phi_j = \max \begin{cases} -\max\left[S_{11},\ S_{12} - \delta_{21},\ S_{13} - \delta_{31},\ S_{14} - \delta_{41}\right] \\ -\max\left[\delta_{21},\ S_{22},\ S_{23} - \delta_{32},\ S_{24} - \delta_{42}\right] \\ -\max\left[\delta_{31} - S_{22},\ \delta_{32},\ S_{33},\ S_{34} - \delta_{43}\right] \\ -\max\left[\delta_{41} - S_{23},\ \delta_{42} - S_{33},\ \delta_{43},\ \delta_{43},\ S_{44}\right] \end{cases} \tag{3.15}$$

The indexing in (3.15) gives a hint on how the message update can be generalized for the number of motion upper bound K and is included in appendix B for further reference.

3.4.4.2 Facility cost discount scheme

To encourage the facilities to have shared basis, we modify the cost function (3.6) to implement a discount scheme: the more the number of shared basis, the greater the discount. This discounted $\mathcal{C}$ is used to compute message update in (3.10) and (3.15), thus encouraging facilities with shared basis to be chosen.

The degree of overlap in the basis is based on comparison with a reference subspace set $\mathbb{S}_{ref}$, which contains the set of opened facilities according to the current beliefs. This reference subspace is initialized as facility j whose node $\{e_j\}$ has the largest belief. The candidate set $\mathbb{S}_{can}$ from which further facilities will be drawn is initialized to be the remaining members of the entire subspace hypothesis set $\mathbb{S}$ (the belief b_j at node e_j is the sum of all the incoming messages, which is $\xi_j + \phi_j$).

The idea behind the discount scheme is to iteratively fill $\mathbb{S}_{ref}$ with K subspaces with the largest beliefs, after taking into account the facility cost discount due to overlapping subspace basis. At the i^{th} iteration, the discount is applied to the cost $\mathcal{C}_i$ computed from (3.6). The belief for each subspace in $\mathbb{S}_{can}$ is re-computed with this discounted cost. The subspace with the largest belief will then be removed from $\mathbb{S}_{can}$ and added to $\mathbb{S}_{ref}$. After filling $\mathbb{S}_{ref}$ with K subspace hypotheses, the discounted ϕ values associated with those members in $\mathbb{S}_{ref}$ replace the corresponding ϕ message update computed using (3.15). This facility cost discount scheme is summarized below:

Algorithm 2 Facility cost discount scheme

Input: subspace hypothesis set $\mathbb{S}$, upper bound on the number of motion K, discount factor η

(1) Compute the belief at each e_j by summing the incoming messages

(2) Initialize the reference subspace S_{ref} as the subspace hypothesis whose e_j has the largest belief

(3) Initialize the candidate set $\mathbb{S}_{can}$ as the remaining members in $\mathbb{S}$

for i=1 to K **do**

(4) Compute basis overlap degree d for each subspace $\in \mathbb{S}_{can}$ with the reference subspace S_{ref}

(5) For each subspace $\in \mathbb{S}_{can}$, compute the discounted cost $\mathcal{C}'_i = (1 - \eta d) \times \mathcal{C}_i$ and use this discounted cost to compute ϕ_j based on (3.15)

(6) Find the subspace with the largest belief. Remove this subspace from $\mathbb{S}_{can}$ and add it to S_{ref}

end for

Output: Discounted ϕ message updates

3.4.4.3 Message update for ξ

The message ξ_j can be interpreted as the overall responsibility to the facility j. For each facility j, let k be the index of the largest element of the set $\{\rho_{ij}, i = 1, \ldots, N\}$. The update can then be shown to be

$$\xi_j = \rho_{kj} + \sum_{i \neq k} \max(0, \rho_{ij}) \tag{3.16}$$

3.4.4.4 Message update for α

The other message update that is affected by the global facility function is α. The message update for α can be shown to be

$$\alpha_{ij} = \min[0, \sum_{i \neq k} \max(0, \rho_{ij}) + \phi_j] \tag{3.17}$$

3.4.5 Subspace hypothesis generation and selection

We provide a different subspace hypothesis generation strategy from FLoSS/UFLP. Our strategy is based on the solution to (3.4), C^*. Each column i of C^* represents the coefficients of other trajectories required to represent this trajectory i. For the case of rigid motion segmentation, since each trajectory comes from an affine subspace, it needs at most four other trajectories for representation. We therefore retain only the top four largest absolute value coefficients in each column and form a subspace hypothesis using that column. The number of subspace hypothesis M is therefore the number of unique subspace hypothesis proposed by all the trajectories.

When the MB-FLoSS message update is completed, subspace hypothesis j is chosen as a representation subspace if the belief $\xi_j + \phi_j$ at facility j is non-negative.

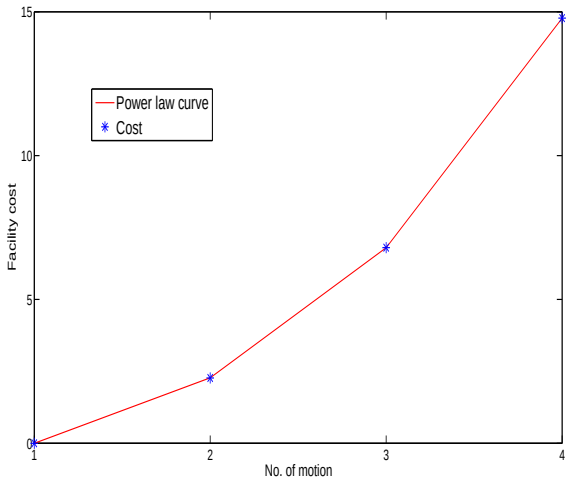
3.5 Experiments

We evaluate the performance of our proposed method on the Hopkins 155 dataset [66] and the augmented Hopkins 380 dataset. The Hopkins 155 database has established itself as the de facto standard for motion segmentation. The Hopkins 155 dataset consists of 155 sequences of feature points labeled according to their motion. There are 120 two motion sequences and 35 three motion sequences. The

dataset consists of three categories: checkerboard, traffic and articulated.

For the Hopkins 155 dataset, we base on KO and state-of-the-art LRR for comparison. For the augmented Hopkins 380 dataset, the good performance and availability of Matlab code [39] makes LRR the choice for comparison.

Model selection in LRR returns predicted number of motion in the range of $1-4$. For our facility cost model, we therefore set the upper bound K as four. The facility cost model used for the experiments is shown in figure 3.5, with the power law in (3.6) specified by $a = 0.35$ and $p = 2.7$. The discount factor η used in the facility cost discount scheme(algorithm 1) is set to 0.05.



7

Figure 3.5: Facility cost model used for the experiments

Since the number of motion is no longer known a priori, we need to generalize the misclassification rate to take into account the wrong number of motion group given by model selection. In [66], the mis-

classification rate is given by the label permutation with the lowest misclassification rate. For the generalized misclassification rate, the label permutation process is naturally extended to account for the case when the wrong number of motion group is given by model selection. Any groups (either in the segmentation result or in the ground truth) whose labels are not assigned after the label permutation process contribute to the misclassified elements. This generalized misclassification rate thus penalizes both model selection error and error in classifying the trajectories according to their motion.

We find that using SSC for classification, based on the number of motion given by the MB-FLoSS model selection gives the best overall performance. This combination is compared against the state-of-the-art LRR.

3.5.1 Augmented Hopkins 380

The need for augmenting the dataset arises from two considerations. Firstly, the model selection algorithms should work for arbitrary number of motion. In particular, for the Hopkins 155 dataset, the model selection algorithms should be tested against not just two and three motion but one motion as well. Secondly, the skewed distribution of the number of two vs. three motion sequences distorts the model selection rate, since focusing solely on two motion sequences will lead to good model selection rate. This distortion due to the uneven distribution is illustrated in [12] where [56] shows a better model selection performance by estimating two motion most of the time.

In view of these considerations, we choose to augment the Hopkins 155 dataset with one motion sequences and additional three motion sequences. The one motion sequences are derived from the original two and three motion sequences by treating each motion as a one motion sequence. For example, from the three motion sequence 1R2RC, we derive three sequences of one motion 1R2RC_g1, 1R2RC_g2, 1R2RC_g3. The additional three motion sequences are generated by concatenating the two motion traffic sequences with the foreground one motion sequences derived from the two motion traffic sequences. The summary of this augmented data in table 3.1 shows a more even distribution in terms of the number of sequence for each number of motion.

No. of motion	One	Two	Three
No. of sequence(original)	0	120	35
No. of sequence(augmented)	135	120	125

Table 3.1: Summary of the augmented Hopkins 155 dataset

3.5.2 Result

Table 3.2 shows the model selection result for the Hopkins 155 dataset. Our work enjoys an advantage over LRR and outperforms KO decisively. It is worthwhile noting that both LRR and KO show better performance for 2 motion at the expense of 3 motion whereas our proposed method handles both 2 and 3 motion more evenly.

	MB-FLoSS	LRR	KO
Overall	79.35%(123)	78.06%(121)	74.84%(116)
2 motion	81.67%(98)	84.17%(101)	82.50%(99)
3 motion	71.43%(25)	57.14%(20)	48.57%(17)

Table 3.2: No. of motion prediction rate for Hopkins 155. The number of sequences predicted correctly is shown in parenthesis

For the augmented Hopkins 380 dataset, table 3.3 shows the advantage of our proposed work over LRR more decisively. Once again, it is worth noting the more even performance of our proposed work compared to LRR.

	MB-FLoSS	LRR
Overall	83.68%(318)	81.05%(308)
1 motion	87.41%(118)	85.93%(116)
2 motion	81.67%(98)	84.17%(101)
3 motion	81.60%(102)	72.80%(91)

Table 3.3: No. of motion prediction rate for the Hopkins 380

The tracked points and basis set chosen for the checkerboard sequence 2rt3rcr_g12 are shown in figure 3.6 and 3.7.

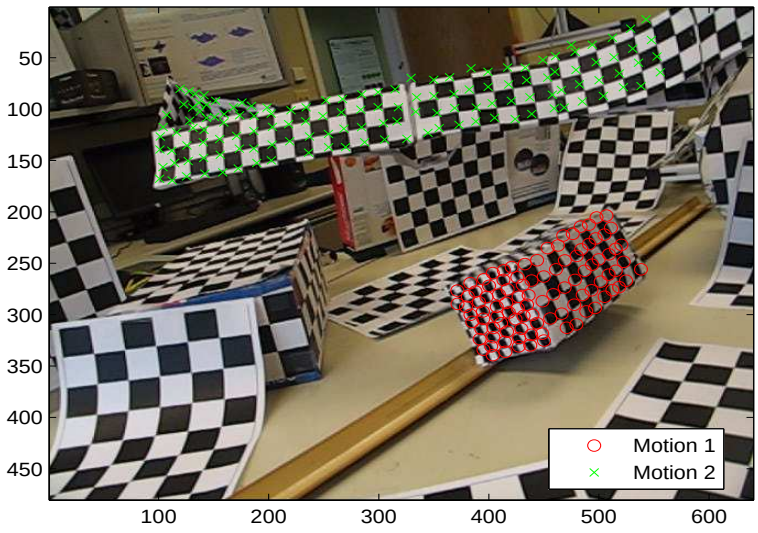


Figure 3.6: Ground truth for the checkerboard sequence 2rt3rcr_g12

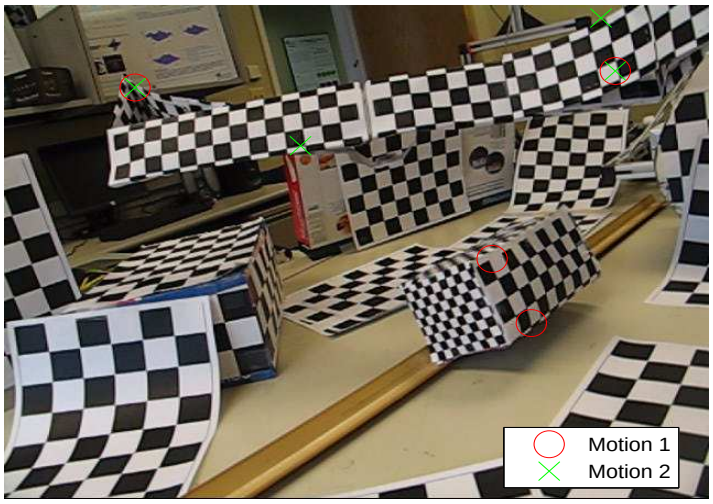


Figure 3.7: Overlapping basis for the checkerboard sequence 2rt3rcr_g12

For classification, table 3.4 shows that our proposed method com-

pares favorably to the state-of-the-art LRR.

	MB-FLoSS + SSC		LRR	
Hopkins	155	380	155	380
Overall	10.04%	8.36%	10.16%	8.98%
1 motion	-	8.74%	-	7.99%
2 motion	9.45%	9.45%	8.59%	8.59%
3 motion	12.07%	6.90%	15.51%	10.43%

Table 3.4: Generalized misclassification rate for the Hopkins 155 and 380

3.6 Conclusion

We formulated and realized the minimal basis approach to subspace segmentation and demonstrated its model selection strength. The success hinges on the use of an enhanced FLoSS framework, employing a convex relaxation formulation for subspace hypothesis generation, and a power-law facility cost with a simple discount scheme that favors overlapping subspace. Despite the added complexity due to the modified facility cost, we show how the message passing can be made tractable and efficient.

Chapter 4

Non-Rigid Structure From Motion

4.1 Introduction

Structure from motion, or the recovery of 3D object structure from video streams of 2D image data, is of fundamental importance in computer vision. The reconstructed 3D shapes serve as input to other applications such as augmented reality, object recognition and computer graphics etc. In many of these applications, non-rigid motions are arguably more common than rigid motions. Almost all animals and many mechanical objects (such as cranes, earthmovers) change shape as they move.

Compared to rigid SFM, the ill-posed or under-constraint nature of non-rigid SFM makes it a challenging problem. Besides the fact that different deforming 3D shapes can share the same image projections, the inherent basis ambiguity presented in [67] is also an important contributing factor. In our work, we propose a subspace segmentation based approach to solving the non-rigid Structure From

Motion(NRSFM) problem.

We first present the subspace segmentation approach in section 4.1.1 and then review the rich literature of NRSFM works in section 4.1.2 and 4.1.3, including both the shape basis and piecewise works. Important works that do not fall into these two categories are covered in section 4.1.4. The key ideas of MB-FLoSS applied to NRSFM is explained in section 4.2. In section 4.3, we describe the details of reconstruction based on MB-FLoSS segmentation. The experimental details are covered in section 4.4.

4.1.1 Subspace segmentation approach

Our proposed method for NRSFM based on MB-FLoSS can be viewed in the broader context of the subspace segmentation approach. The idea in the subspace segmentation approach is to decompose a non-rigid motion into constituent components based on the motion of these components. These components are then reconstructed as individual 3D patches using state-of-the-art shape basis factorization methods. Just like the piecewise approach covered in section 4.1.3, these patches are stitched back together to form a global shape.

This subspace segmentation approach is different from the piecewise method in section 4.1.3 in that the constituent components resulting from the subspace segmentation approach do not have to be rigid or to follow say, a quadratic deformation model. The emphasis in the subspace segmentation approach is on the motion coherence of the components, where by motion coherence, we mean that the shape deformation can be expressed as a linear combination of a small set

of basis shapes.

Our proposed decomposition offers the following advantages. Just like the piecewise approach e.g. [68][5][3], these constituent components can be described by simpler models of lower complexity. However, our decomposition generates a much smaller number of components and is thus much more parsimonious and natural. More importantly, our decomposition method does not assume the underlying model of each component; the number of shape basis for each component can be deduced from the decomposition automatically. To the best of our knowledge, this is the first algorithm that can decompose a non-rigid motion into its coherent parts without the use of a prior model.

4.1.2 Shape basis approach

The factorization approach to rigid Structure From Motion(SFM) advocated by [10] has proven to be the foundation for subsequent SFM works. There have been various extensions to the original rigid SFM work. [46] took the first step into motion segmentation by extending the factorization method to multiple independently moving rigid bodies. Bregler[69] first showed how the factorization approach can be extended to NRSFM with the introduction of the shape basis concept. This shape basis based factorization framework forms the core of many of the subsequent non-rigid SFM works.

In [67], the inherent shape basis ambiguity was highlighted, with the important result that the orthonormality constraints for recovering the motion matrix is insufficient in removing the fundamental

ambiguity between the shape bases and shape coefficients i.e. the shape bases and the shape coefficients cannot be recovered uniquely. This inherent ambiguity has a significant impact in shaping latter NRSFM works. [67] proposed to resolve this ambiguity by imposing additional basis constraints.

In [67] and many subsequent works, the orthogonality constraint on the motion matrix is enforced using the Gram matrix formed by triplet columns of the rectifying transformation matrix. Instead of using the Gram matrix, [70] proposed to solve for the rectifying transformation matrix directly, using all columns and not just triplets of columns. Note that the main focus of [70] is to solve for the orthogonal structure of the motion directly but does not address the ambiguity issue.

In view of the ambiguity highlighted by [67], latter works elected to add priors to resolve this ambiguity. In consideration of the fact that simple linear subspace shape models are extremely sensitive to noise, EM-PPCA[6] proposed that statistical priors should be used to constrain the parameter space. More specifically, EM-PPCA introduced priors as a Gaussian distribution on the shape coefficients. [71] shows that if shape priors are known, then the shape basis factorization can be made more reliable with the incorporation of this shape prior through the generalized SVD[72].

Instead of imposing priors on the shape coefficients, the trajectory basis(TB) work[73][7] proved the duality relationship between the shape coefficients and the shape trajectory, and imposed smoothness prior on the trajectory basis by expressing the trajectories in terms of

the Discrete Cosine Transform(DCT) representation. This trajectory basis idea is extended to the case of multiple cameras in [74]. The Column Space Fitting work(CSF)[75][8] also imposed the smooth shape trajectory prior through the DCT representation. CSF solves for the motion matrix using TB and then solve for the DCT representation of the shape basis coefficients using the column space fitting method based on the second order Levenberg-Marquardt method. [76] introduces the use of kernel into CSF to handle non-linear relationships in the coefficients.

[77] caused a re-think in the shape basis factorization paradigm by showing that ambiguity in orthonormality constraints as proved in [67] does not affect the recovery of 3D structure. The sufficiency of the orthonormality constraints in ensuring unique 3D reconstruction(up to a global rotation) means that the priors in [6][73][8] may not be necessary. The simple prior free(SPF) method proposed a shape basis factorization method with no priors, but with important constraints on metric upgrading and structure recovery.

The metric projection method(MP)[78] alternates between solving the motion matrix and the shape basis matrix. The motion matrix is the rotation matrix scaled by the shape basis coefficients. When solving for the motion matrix, MP imposes the metric constraint on each frame of the motion matrix by a metric projection onto the motion manifold i.e. ensuring that pairs of rows in each frame are orthogonal in an integral manner rather than as a post processing step. The metric projection is achieved by solving a convex relaxation of an unconstrained least-square problem, instead of the usual

geodesic approach[79][80]. In [81], the metric projection problem is formulated as a constrained bilinear optimization problem and solved through the ALM method covered in section 2.1.1.

[82] proposed a coarse to fine model by adding deformation modes or number of shape basis incrementally to achieve a low rank shape model. Such coarse to fine model will allow the deformation modes to be determined automatically. There is however no model selection scheme proposed in this work and relies on cross validation as a stopping criterion.

4.1.3 Piecewise approach

Instead of reconstruction at the global level, the piecewise approach focuses on local patches reconstruction and piecing back these 3D patches into a global shape. There are two broad categories of the piecewise approach, one focusing on strongly deforming surfaces and the other one addressing articulated motions.

While the shape basis factorization formulation works well on sequences where deformations are small deviations from a rigid principal component, it no longer holds for strongly deforming objects, such as the cloth sequence in [83] and the paper sequence in [84]. The reason is that the deformations are too complex to be explained by a global linear model. The intricate deformations would require a substantial increase in the number of shape basis used, resulting in over-fitting.

The piecewise approach first divides the object into overlapping patches and reconstruct each individual patch independently. Such

local models are often easier to construct because they require fewer parameters than the global ones. Since each local patch has fewer points to fit to the model, they are easier to optimize and less prone to over-fitting(a more complex model will require more points to fit). Since the patches are reconstructed independently(up to a depth ambiguity), they need to be stitched back together to form a global shape.

[85] proposed to decompose a global non-rigid body into local rigid patches consisting of triplets of points. The intuition is that even very complex non-rigid motions can be approximated locally by a rigid transformation involving three points. This local rigidity assumption is valid when feature points are not too distant from each other i.e. the feature points are dense. The triplet of points is the lower bound required to determine if these three points' motion is rigid under orthographic camera model. Each of these triangle is reconstructed independently by solving an orthographic 3-points-N-view rigid SFM problem.

[68] proposed a local piecewise reconstruction method for strongly deforming objects. It first constructs a mean shape of the object and divides the surface manually into overlapping patches. This overlapping patch property requires a point to have multiple label assignment. Each patch is reconstructed by using a quadratic deformation model, which consists of three modes of deformation - linear, quadratic and cross-terms. These three modes of deformation are combined with the time varying coefficients to the reconstructed shape in each frame. Since the patches are reconstructed indepen-

dently in their own reference frame, they will be reconstructed at different depths. These unconnected patches are misaligned only along the depth direction. The overlapping points are used to align the patches along the Z coordinate of the translation vector of these patches and stitch all the patches together into one global shape.

Instead of defining the patches manually, a more principled division of the strongly deforming object into local models is provided in [4]. In [4], NRSFM is formulated as a labeling problem where the number of labels and their assignment to the data points are computed simultaneously. The objective function consists of a unary term that penalizes fitting error of each point across multiple label assignment. This multiple label assignment is driven by the need for the patches to overlap so that the overlapping points can be used to stitch the patches together to form a global shape, as in [68]. Hard constraints involving interior points are imposed on the objective function to ensure that overlap occurs. Interior points are points such that the neighbors of the interior points must also belong to the same model as the interior points, but the neighbors are not necessarily interior points.

The unary fitting error term models the error from fitting the points in each patch to the same quadratic deformation model in [68]. To prevent too many patches from being formed(and therefore over-fitting), a minimum description length(MDL) cost is added as a model complexity cost to the objective function. The use of MDL as model complexity cost is common in vision problems, see for example [86][87][88]. This constrained energy function is solved

by an Expectation-Maximization(EM)[89] like approach, alternating between finding a better assignment of points to the models and fitting the models to the assigned points.

The key advantage of these local piecewise approaches is that they avoid the limitation of having to make assumption about the deformation obeying some global basis model. However, in overcoming this limitation, they go to the other extreme of breaking up the scenes into very minute patches. For many scenes and objects, even though their deformations are not explainable globally by a linear combination of basis, it can be explained by a small number of clusters of deformations, each of which spans a low-dimensional linear subspace (i.e. moves coherently). These regions or parts typically involve a much larger spatial extent than those minute meshes used in local piecewise approaches.

The key difference of our proposed method MB-FLoSS lies in that it seeks this more natural partitioning of the deformations into the much larger regions of coherently moving parts, and also automatically determines the underlying linear subspaces. For instance, even though [4] looks similar to our proposed MB-FLoSS in terms of the decomposition of a global non-rigid body into overlapping patches and the use of model complexity cost in the objective function, there is a very fundamental difference between the two. [4] tends to divide the object into numerous small patches so that the simple quadratic deformation model holds. In contrast, MB-FLoSS decomposes the global non-rigid body into much fewer sub-parts of coherent motion, which roughly correspond to the elementary parts of many moving

objects.

Articulated motion forms a specific subclass of non-rigid motions but serves as a good approximation for human motion. The piecewise approach to articulated motion[90][3][5], involves segmenting such articulated motion into rigid components and reconstructing these rigid components using the well established rigid reconstruction method[10]. These works are able to model the overlap between these rigid components that constitute the articulated motions so that a global shape can be reconstructed.

MB-FLoSS is similar to these piecewise articulated motion works in terms of the ability to generate overlapping sub-parts, so that these sub-parts can be stitched back together to form a global shape. The crucial difference that distinguishes MB-FLoSS from these works is that the sub-parts generated by MB-FLoSS no longer need to be rigid and is in fact non-rigid in general. This non-rigid modeling of the sub-parts allows a wider range of human articulated motions to be handled e.g. belly dancing where the torso is no longer a rigid motion.

In [90], RANSAC[91] is used to segment articulated motions into rigid components, removing outliers at the same time. The two types of articulated motion arising from universal and hinge joints are decomposed into the rigid constituents. Based on the relationship between these constituent components, the data matrix can be factorized and leads to 3D reconstruction.

In [3], an articulated motion is first segmented into its constituent rigid parts by the use of LSA[47] as a motion segmentation algorithm.

The number of rigid components is determined by the effective rank (3.2) of the data matrix. The dependence or overlap between any pair of the segmented motion subspaces is measurable by their minimum principal angles. From this pairwise overlapping relationship, the type/structure of the articulated motion or the kinematic chain, can be constructed.

[3] is very similar to our work in terms of the use of model selection to determine the number of motion subspaces and the subsequent segmentation into the constituent components according to their motion. The important difference is that for MB-FLoSS and more generally the subspace segmentation approach covered in section 4.1.1, the components can be non-rigid. MB-FLoSS is a more general framework in the sense that the same model selection algorithm with exactly the same parameters are used for both the rigid and non-rigid motion. In contrast, the model selection scheme for [3] only works well for the articulated motions but does not perform well for general rigid motion segmentation nor for general non-rigid motions.

The idea in [5] is to decompose an articulated motion into overlapping constituent rigid segments. The overlapping points arise naturally as points on the joints between segments. This problem is formulated and solved identically to [4] but with different model parameters and unary penalty. The fitting penalty now measures the image reprojection error. The model parameters now consists of the motion matrix and shape structure that arise from the rigid reconstruction of the segments. Compared to [4], it seems that [5] is even

more similar to MB-FLoSS. The important difference is that while [5] works on atomic rigid segments, there is no such requirement for MB-FLoSS. In MB-FLoSS, the emphasis is on components with simpler, coherent motion, regardless of whether these components are rigid or non-rigid.

It would seem that the underlying formulation of [4] is able to handle NRSFM in general. For the case of highly deformable body, [4] is applicable. For articulated motion in [5], the same formulation is applied but with different model parameters and fitting penalty. Under this formulation, the two types of non-rigid motion need to be distinguished so that the right model parameters and fitting penalty can be used. For the subspace segmentation approach, these two types of non-rigid motions are handled in the same manner.

4.1.4 Other approaches

Besides the shape basis and piecewise approaches, there is a sprinkling of works that do not fall into these two categories. In [92], there is a re-think on modeling the shape space with a linear subspace. Instead of the global linear subspace model, [92] assumes that small temporal neighborhoods of shapes are well-modeled with a linear subspace. This assumption constrains the shapes to lie on a low dimension manifold. The shape is solved using the Non-Isometric Manifold Learning algorithm[93].

Like [92], [94] is also able to handle general large nonlinear deformation. In addition, the model graph formulation in [94] allows larger consistent image clusters to be formed, so that it is able to

handle occlusions due to large variation in viewpoints, as well as making possible rigid 3D reconstruction based on the most general perspective camera model.

Both [92] and [94] assume that multiple 3D models (what is termed as rigid shape chain in [92]) can be recovered from subsets of the input image set using rigid SfM techniques. This depends on our being able to choose a subset of the input frames such that it contains only frames that are projections of the same 3D shape but taken from different viewpoints. This may not be true unless the input video is long enough and the deformation motions are somewhat repetitive in nature. Otherwise, without the ability to form these prototypical 3D shapes, the subsequent steps to model the non-rigidity in these works will break down.

As opposed to forming clusters of rigid motions in the temporal domain like in [92] and [94], our proposed work forms clusters of coherent motions in the spatial domain. The advantage of our formulation is that it is much more likely to handle a larger class of naturally occurring non-rigid motions.

Another difference stemming from our spatial partitioning is that there is no assumption of an image-wide shape basis; this assumption is still required in [92] and [94] ([92] assumes that in some small temporal neighborhood, the 3D shape lies on a linear subspace, while [94] assumes that the 3D shape is a sparse linear combination of a large number of basis shape)

Instead of imposing the rank constraint in the deformation space like what the shape basis factorization works do, [95] chooses to im-

pose the Procrustean normal distribution constraints on the motion parameters. The idea in [95] is to treat NRSFM as an alignment problem with additional constraints on the rotation matrices derived from generalized Procrustes analysis(GPA). In GPA, alignment is based on a global mean shape. It therefore depends on the objects having a main component that can be considered as rigid, as this component is crucial for the alignment step. For deforming motions(e.g. a bending motion), where there is a lack of such a rigid component, such an approach will encounter difficulties

Unlike the majority of other NRSFM works that operate in batch mode, [96] proposes an incremental approach that updates the motion and shape estimates sequentially as new video frames are acquired. In [96], there is an explicit representation of a mean rigid shape, upon which the non-rigid variations are built upon. As we have seen previously, this assumption of a rigid component restricts the type of non-rigid motion it can handle.

As a direct contrast to both [95] and [96], there is no such an assumption of a mean global rigid component in our work.

4.1.5 Contribution

The current competitive NRSFM methods can be broadly divided into the shape basis factorization approach and the piecewise approach. There are issues that need to be addressed in both these approaches. For the shape basis factorization approach, the known number of shape basis assumption is not a practical one and needs to be addressed. For the piecewise approach, articulated motions and

continuously deforming objects need different modeling parameters and treatments. Such model specificity calls for a more holistic approach that handles these two types of non-rigid motion uniformly, without having to know a priori the type of non-rigid motion.

Moreover, the shape basis approach(and other works such as [95] and [96]) explains non-rigid motion as consisting of a rigid principal component, upon which a linear combination of shape basis is built upon. While this assumption of a rigid principal component is intuitive and arises naturally in the shape basis framework, it nevertheless places restriction on the type of non-rigid motion that can be handled. Can this need for a rigid principal component be removed so that the shape basis approach can handle a wider range of non-rigid motion, but at the same time enjoy the good performance of the shape basis approach?

Our main contribution is the introduction of MB-FLoSS as a subspace segmentation approach to NRSFM. Our subspace segmentation approach is different from the piecewise articulated motion works [5] and [3], where each component has to be rigid. [3] considers mainly articulated objects with the possibility of a non-rigid part e.g. the human body consisting of limbs and torso modeled as rigid segments while the facial motion of the head is non-rigid. Our subspace segmentation approach is more general and less restrictive in the sense that each component is in general non-rigid.

As far as we know, we are the first to propose a subspace segmentation approach with non-rigid components. Our judicial choice of SPF for reconstruction ensures that our approach enjoys the best of

both worlds - reduced deformation complexity of the local piecewise approach(thus ensuring the validity of the linear shape basis model) and the parsimonious shape description as well as state-of-the-art reconstruction results from the shape basis factorization methods.

A standout feature of our work is that the unifying framework provided by MB-FLoSS handles the different types of non-rigid motion uniformly. MB-FLoSS does not differentiate between articulated motions from the continuously deforming type of motion such as face and shark. In contrast, [4] and [5] need a priori knowledge of the type of non-rigid motion in order to apply the correct model and parameters.

It is important to emphasize that MB-FLoSS is the only viable representative of the subspace segmentation approach. LRR has strong competitive performance for both model selection and segmentation that can in principle be applied to both rigid and non-rigid motion. However, the independent subspace assumption in LRR results in no overlap between the segmented subspaces. This lack of overlap prevents LRR from being a full fledged reconstruction method. The introduction section in [4] best describes this overlap requirement: "A fundamental requirement for piecewise reconstruction is the need for overlap between models to enforce global consistency...".

SSC's strong robust performance in rigid motion segmentation makes it an attractive candidate for the subspace segmentation approach. The main obstacle for SSC is the lack of an in-house model selection algorithm. A straightforward, generic spectral eigen gap

based model selection will not work well. This generic solution was adopted in ORK, giving model selection results that are uncompetitive. Such scheme will face even greater challenges for the case of non-rigid motion, since the subspaces are of varying dimension.

The first benefit of the subspace segmentation approach comes from the enabling of the shape basis factorization method as a practical NRSFM method. The state-of-the-art shape basis factorization methods all assume known number of shape basis to achieve the optimum reconstruction result. This is of course not practical because a) there is no ground truth available in real reconstruction problems b) there is no need for reconstruction if the ground truth is available. With the basis set of each subspace being made explicit in MB-FLoSS, the number of shape basis for each subspace arises naturally, allowing the number of shape basis to be inferred.

The subspace segmentation approach also has the additional advantage of circumventing the need for a rigid principle component, a central tenet in the shape basis approach. This move away from a rigid principle component is important because it expands the domain of non-rigid motion that can be handled by the shape basis framework, and at the same time leveraging fully on the shape basis framework.

The second benefit is related to the perceptual significance of the components resulting from MB-FLoSS. The human body is compositional and made up of parts, and thus we should be able to tease them apart and mentally recompose them at will. The local piecewise works such as [4] partitions at too small a scale for the patches

to be intuitive. At the other extreme, the shape basis approach does not even yield parts. There are of course circumstances where such fine partitions are more natural, such as the description of the cloth sequence[83] by [68][4].

The piecewise articulated motion works [5] and [3] decompose an articulated motion into rigid subparts. However, the elementary parts of a non-rigid motion are not always rigid. MB-FLoSS captures this concept of non-rigid elementary parts much better than other piecewise works.

The last benefit brought about by the MB-FLoSS framework stems from the more nuanced treatment of model complexity, taking into account overlap between subspaces.

4.2 MB-FLoSS

MB-FLoSS from chapter 3 is the key subspace segmentation algorithm for decomposing a non-rigid motion into sub-groups of relatively simpler motion. MB-FLoSS was applied to rigid body motion segmentation, from determining the number of motion to segmenting the trajectories according to the motion. In this chapter, we show how MB-FLoSS can be applied to non-rigid motion.

MB-FLoSS serves two functions in non-rigid SFM. The first is in determining the number of sub-groups and the number of shape basis K_i for each sub-group i. This is made possible by the explicit basis representation given by MB-FLoSS. Secondly, the eventual chosen subspace representation allows each trajectory to be assigned to its motion subspace. This segmentation thus defines each non-rigid

motion sub-groups, allowing 3D reconstruction to be carried out for each sub-group.

For NRSFM, a non-trivial adaptation is the basis definition in the hypothesis generation step. In chapter 3, due to the rigid motion and affine camera model assumption, each subspace is defined by at most four trajectories. For non-rigid motion, the number of basis in each subspace is no longer bounded by this constraint. We explain how the number of basis in each subspace can be determined in section 4.2.2.

NRSFM based on the MB-FLoSS framework consists of the following steps

1. Hypothesis generation
2. Determine the basis set for each of the hypotheses
3. Model selection to determine the number of subspaces and basis set for these representative subspaces
4. Assign the trajectories to the closest subspace thereby decomposing the global non-rigid motion into patches
5. Reconstruct 3D shapes of individual patches
6. Stitch the individual patches back together to form a global 3D structure

We will explain each of the step in detail in the following sections.

4.2.1 Number of shape basis

We will describe how the number of shape basis arise naturally from the subspace segmentation approach. In chapter 2, the idea of self expressive representation is examined in detail through the SSC and LRR works. Self expressive representation can be thought of as the factorization of the data matrix into the data matrix itself and the representation or coefficient matrix in the presence of noise

$$\widehat{W} = \widehat{W}C + E \tag{4.1}$$

where $\mathbb{R}^{2F\times N} \ni \widehat{W}$ is the data matrix, $\mathbb{R}^{N\times N} \ni C$ is the representation matrix and $\mathbb{R}^{2F\times N} \ni E$ is the error matrix. C will have different properties based on the penalty imposed on C in the problem formulation. For LRR, C will have low rank, say r, while C will have sparse number of rows for MB-FLoSS. In either case, C can be decomposed into two matrices of inner dimension r

$$C = C_1C_2 \tag{4.2}$$

where $\mathbb{R}^{N\times r} \ni C_1$ and $\mathbb{R}^{r\times N} \ni C_2$. With this decomposition, we can then do the factorization

$$\widehat{W} = \widehat{W}C_1C_2 + E \tag{4.3}$$

$$\widehat{W}' = C_1'C_2 \tag{4.4}$$

where $\mathbb{R}^{2F\times r} \ni C_1' = \widehat{W}C_1$ and $\mathbb{R}^{2F\times N} \ni \widehat{W}' = \widehat{W} - E$

If we further constrain r to be multiples of 3(by rounding, for example), say $r = 3K$ where K is an integer, then (4.4) shows that the data matrix(with the noise removed) can be factorized into two matrices of inner dimension $3K$.

As shown in (C.1), (C.2) and (C.3) in appendix C, the data matrix can be factorized as

$$\widehat{W} = \Pi B \tag{4.5}$$

where $\Pi \in \mathbb{R}^{2F \times 3K}$ is the motion matrix that contains the rotation matrices scaled by the shape coefficients and $B \in \mathbb{R}^{3K \times N}$ is the shape basis matrix.

By comparing (4.4) and (4.5), together with the constraint that r is a multiple of 3, the number of shape basis can thus be identified as K.

4.2.2 Hypothesis generation

As in chapter 3, we first solve the convex relaxation problem

$$\begin{aligned} &\min_{C} \ \|C\|_{2,1} + \gamma \|E\|_{1,2} \\ &s.t. \quad \widehat{W}C = \widehat{W} \end{aligned} \tag{4.6}$$

where $\widehat{W} \in \mathbb{R}^{2F \times N}$ is the data matrix constructed from the tracked feature trajectories, F is the number of frames, N is the number of tracked feature points, $C \in \mathbb{R}^{N \times N}$ is the representation matrix, $\|C\|_{2,1} = \sum_{i=1}^{2F} \sqrt{\sum_{j=1}^{N} ([C]_{ij})^2}$ and $\|E\|_{1,2} = \sum_{j=1}^{N} \sqrt{\sum_{i=1}^{2F} ([E]_{ij})^2}$.

Let the solution to (4.6) be C_0. Each column i of C_0 then gives

the coefficients of other trajectories used for representing trajectory i. Each column thus proposes a subspace hypothesis with the basis defined by the non-zero coefficients.

Unlike the rigid motion case in chapter 3, where each trajectory requires at most four other trajectories for representation, the number of basis for each subspace is not known a priori. Due to (4.6) being a convex relaxation of the original problem, the support in each column of C_0 is often not clear cut. This convex relaxation artefact is readily understood as the Robin Hood effect[97]. In figure 4.1, we see the distinct row structure in the representation matrix C_0, but the artefact is readily noticeable.

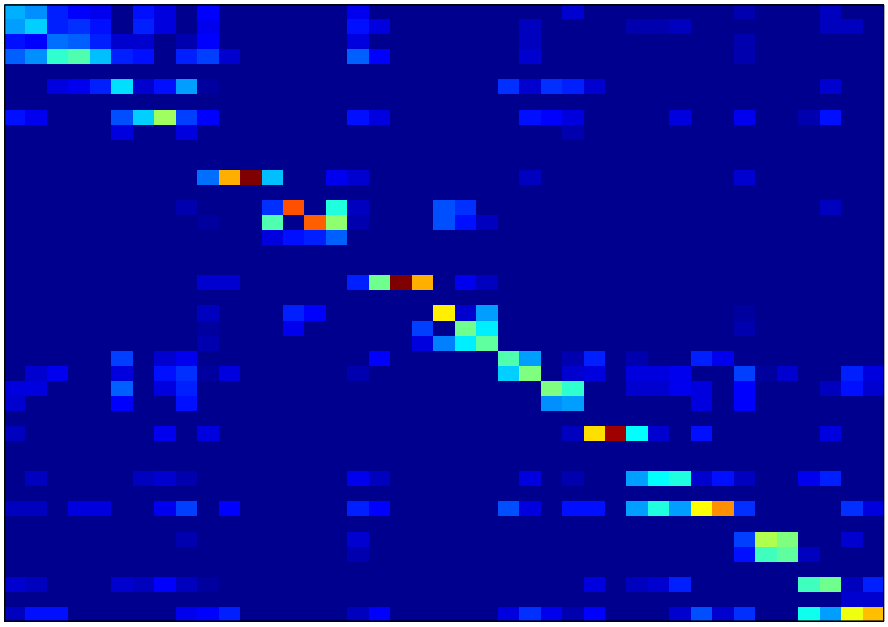


Figure 4.1: Absolute values of the representation matrix C_0 for the pickup sequence. The hotter the color i.e. the more red the color is, the larger the coefficient. The "blueness" indicates very small coefficient values

In figure 4.2, the magnitude in one column of C_0 shows a spread, instead of a sharp, well-defined gap that differentiates the support from the rest.

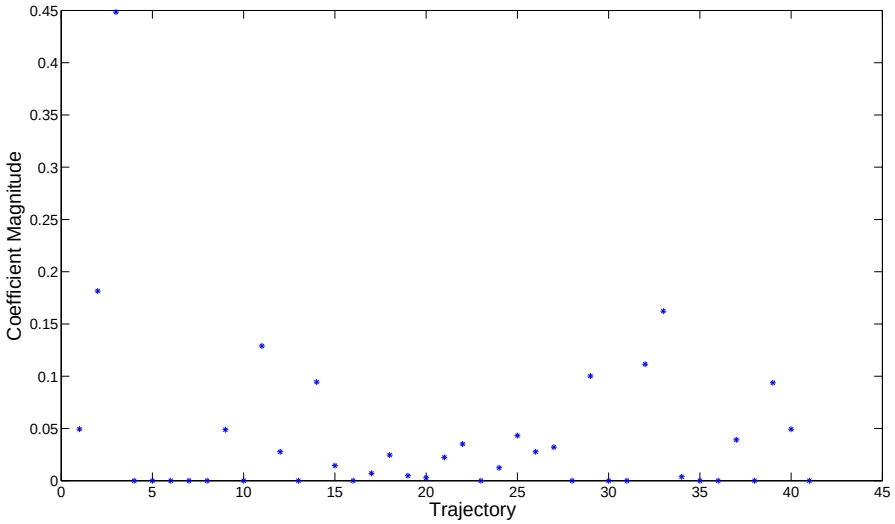


Figure 4.2: Coefficient magnitude of a column of C_0 for the pickup sequence

Recovering the true support in each column from the convex relaxation artefacts is in general a difficult task due to the spread of values, with no discernable gap to define the support. The straightforward way to set a hard threshold to recover the support. This hard threshold can be just a constant or a constant factor of the largest coefficient. We found that setting a hard threshold generally does not perform well because the threshold may not work across all the sequences.

We propose the use of k-means to automatically extract the support. It is tempting to set $k = 2$ to use k-means to extract the support but as seen in figure 4.2, even among the support set, the

typically large gap between the top coefficient and the rest means that the support with intermediate values will be wrongly omitted from the support set. Instead, we set $k = 3$ so as to include coefficients with top and intermediate values as support. The min, median and max of the coefficients are used as initializations to k-means. As seen in figure 4.3, the support recovered is more reasonable.

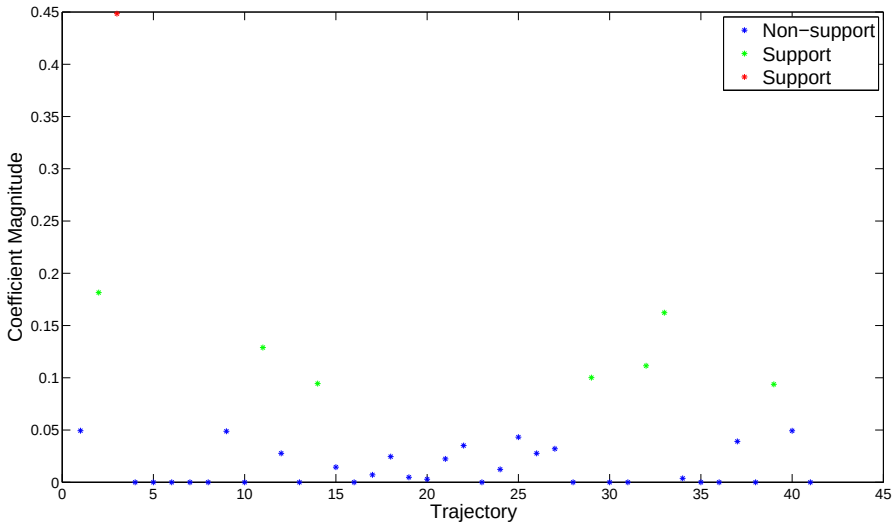


Figure 4.3: The support identified by k-means in red and green, while the non-supports are in blue

Algorithm 3 summarizes the subspace hypothesis generation step. Note that once we determine the basis set for each hypothesis, the basis set will give the dimensionality of the subspaces.

Algorithm 3 Generate subspace hypothesis set

Input: representation matrix C_0, number of tracked trajectories N

(1) Initialization: $\mathrm{H} \leftarrow \emptyset$

for i=1 to N **do**

(2) Apply k-means to column i of C_0 with $k = 3$, giving the centroid set $\{\mathcal{C}_{lo}, \mathcal{C}_{mid}, \mathcal{C}_{hi}\}$ and corresponding label $\{\mathcal{L}_{lo}, \mathcal{L}_{mid}, \mathcal{L}_{hi}\}$

(3) Construct column i support set B_i defined by all basis trajectories with labels $\in \{\mathcal{L}_{mid}, \mathcal{L}_{hi}\}$

(4) $\mathrm{H} \leftarrow \mathrm{H} \cup \mathrm{B}_i$

end for

Output: Subspace hypothesis set H

Note that the use of the row sparsity penalty in (4.6) encourages basis sharing, and thus the typically non-independent motions found among the different parts of a non-rigid motion would result in much sharing of basis among different subspace hypothesis.

4.2.3 Model selection and segmentation

The model selection and segmentation step is the key step to divide the trajectories of the full non-rigid motion into subparts according to their motion. After generating the subspace hypothesis set and the associated basis set, the same MB-FLoSS engine in chapter 3 is now used to determine

1. The number of subspaces, corresponding to the number of subparts from the decomposition of the full non-rigid motion
2. The basis sets of these representative subspaces
3. The clustering of the trajectories based on the motion subspace they belong to

Algorithm 4 describes the model selection and subspace segmentation algorithm based on MB-FLoSS. In step 1, the number of subspaces and the representative facility subspaces are established using the MB-FLoSS model selection scheme described in section 3.4. The chosen representative facility subspaces will capture and make explicit any overlap between the subspaces that is typically found in non-rigid motions. Each trajectory is segmented by assigning it to the closest representative facility subspace, measured by the closest orthogonal trajectory-subspace distance.

Algorithm 4 Model selection and subspace segmentation

Input: data matrix $\widehat{W}$, subspace hypothesis set H, convex formulation weight γ, facility cost model parameters $\{a, b\}$, number of nearest neighbors K_{nn}

1. Use MB-FLoSS to compute
   - n_S, the number of facility subspaces
   - $\{\mathrm{F}_1 \dots \mathrm{F}_{n_S}\}$, a subset of H that are chosen as representative facility subspaces
   - Segmented trajectories by assigning each trajectory to the nearest chosen subspace facilities $\{\mathrm{F}_1 \dots \mathrm{F}_{n_S}\}$, in terms of orthogonal subspace distance
2. Re-classify the basis trajectories by choosing the majority label of the K_{nn} nearest non-basis neighbors

Output: $\mathrm{T} \in \mathbb{R}^N$, trajectories classified with labels 1 to n_S

Step 2 of algorithm 4 involves reclassifying the trajectories that serve as bases. This is necessary because most of these trajectories serve as bases for multiple subspaces and therefore should be zero distance from these subspaces. The orthogonal trajectory to subspace distance criterion for classifying these trajectories is unreliable

because any difference would be due to small numerical errors. We resolve this ambiguity by giving these basis trajectories the majority label of the K_{nn} nearest neighbors. The nearest neighbor distance is defined in terms of the max distance over the entire F frames. Examples of the decomposition are shown in figure 4.11 and 4.13.

4.3 Reconstruction

With the decomposition of the non-rigid motion into patches, 3D reconstruction is now performed locally at the patch level for all the patches. There are important and subtle differences for local patch-wise 3D reconstruction as compared to the more common global 3D reconstruction. These differences need to be taken care off when stitching the patches together.

For each patch i, the patch data matrix W_i is constructed by extracting those columns of the original data matrix $\widehat{W}$ corresponding to the trajectories belonging to the motion subspace. The trajectories belonging to a motion subspace is not just those trajectories with the same label corresponding to the subspace but also the basis trajectories defining the subspace. We therefore take the union of those trajectories resulting from segmentation and the basis trajectories and use these trajectories for reconstruction.

The subspace segmentation approach focuses only on decomposing a global non-rigid body into its constituent components. A concrete algorithm is needed for reconstructing these components. Since these components are non-rigid in general, we specifically need a NRSFM algorithm for reconstruction. While any NRSFM algorithm

can be used, our choice of SPF is not only due to its good performance, but also the principle of parsimony - SPF does not impose any priors when there is no need to. Each patch is therefore reconstructed using the SPF method.

Each patch is first shifted to have zero-mean in each frame before applying SPF. This shifting operation for the patches means that the reconstructed 3D shapes for these patches are all centered at the origin, which is clearly not right. We must therefore undo the shifting operation so that these reconstructed 3D shapes are correctly placed. Based on the recovered rotation matrices from SPF, We first rotate the reconstructed 3D shape at time t back to the camera reference frame at time t. In this camera reference frame, we can undo the zero-mean shifting the data matrix previously underwent.

Another complication we have to deal with is that these reconstructed 3D shape patches have their own frame of reference. This is due to the inherent ambiguity of SFM in general, since any reconstruction is only up to a global rotation. We resolve this ambiguity by rotating all the 3D shape patches to a common reference frame, say the first camera frame.

The last ambiguity we need to resolve is the misalignment in the Z coordinate of the translation vectors of the patches. This ambiguity arose due to the loss of depth information resulting from an orthographic projection. This situation is very similar to the one in [68]. We adopt the same strategy in [68] by using shared basis trajectories between the patches to align the Z coordinate. Even though there is no guarantee of at least one shared basis between any pair of patches,

we find experimentally this is indeed the case. This is not surprising given the non-rigid motion we are dealing with; a patch's motion is unlikely to be independently of all other patches. In fact, we find the stronger condition that we can always find a patch that has at least one shared basis with all other patches. This patch is the reference patch against which the rest of the patches are aligned with. If there are more than one shared basis, we use the average of the difference between shared bases for alignment. Algorithm 5[1] gives the full 3D reconstruction procedure discussed above.

[1]There is an additional rotStruct flag in the input that is concerned with whether it is the camera or the object that moves, the latter of which would necessitate further processing. Readers who are interested in the implementation details should refer to [98] or [99]

Algorithm 5 3D shape recovery

Input: data matrix $\widehat{W}$, representative subspace facilities $\{\mathrm{F}_1 \ldots \mathrm{F}_{n_S}\}$, segmented trajectories T

for i=1 to n_S **do**

(1) Define the indices $I_i \leftarrow \mathrm{T}_i \cup \mathrm{B}_i$, where T_i are the trajectories with label i and B_i are the bases that constitute representative subspace facility F_i

(2) Construct patch W_i by assembling columns of $\widehat{W}$ corresponding to the indices I_i

(3) Shift W_i so that it has zero mean

(4) Reconstruct 3D shape $\hat{S}_i$ for patch W_i using SPF

(5) Rotate $\hat{S}_i$ back to the individual camera coordinate system

(6) Undo the zero-mean shifting in the camera coordinate system

if rotStruct **then**

(7) Rotate $\hat{S}_i$ to a common camera coordinate system

else

(8) Stay in the camera coordinate system

end if

end for

(9) Select a reference patch and align other patches with this reference patch

Output: Reconstructed 3D shape. Note that there is a boolean rotStruct flag that indicates if the camera is rotating or stationary. The reconstructed 3D shape needs to be further processed if the camera is stationary. See [98] or [99] for details.

4.4 Experiments

Unlike the rigid motion case where the Hopkins 155 dataset is the de facto benchmark, NRSFM has yet to see a standard dataset of real(as opposed to synthetic) sequences. We base the 3D reconstruction quantitative comparison on the sequences gathered in the latest work SPF. The SPF dataset consists of 8 sequences summarized in table

4.1.

Sequence	No. of frames	No. of trajectories
Drink	1102	41
Pickup	357	41
Yoga	307	41
Stretch	370	41
Dance	264	75
Face	316	40
Shark	240	91
Walking	260	55

Table 4.1: Summary of the real dataset used for the experiments

The drink, pickup, yoga, stretch and dance sequences come originally from the CMU Mocap database[100] while the face, shark and walking sequences come from [101]. Note that the shark sequence that is used in SPF and CSF comes from [101] and is different from the shark sequence in TB. The shark sequence result for TB reported in SPF is generated by running TB on the shark sequence in SPF. The drink, pickup, yoga, stretch, dance and walking sequences are articulated motions while the face and shark sequences consist of a single body with smoothly varying deformations. The yoga sequence shown in figure 4.4 is an example of articulated motion and the shark sequence shown in figure 4.5 is an example of a single body with smoothly varying deformations.

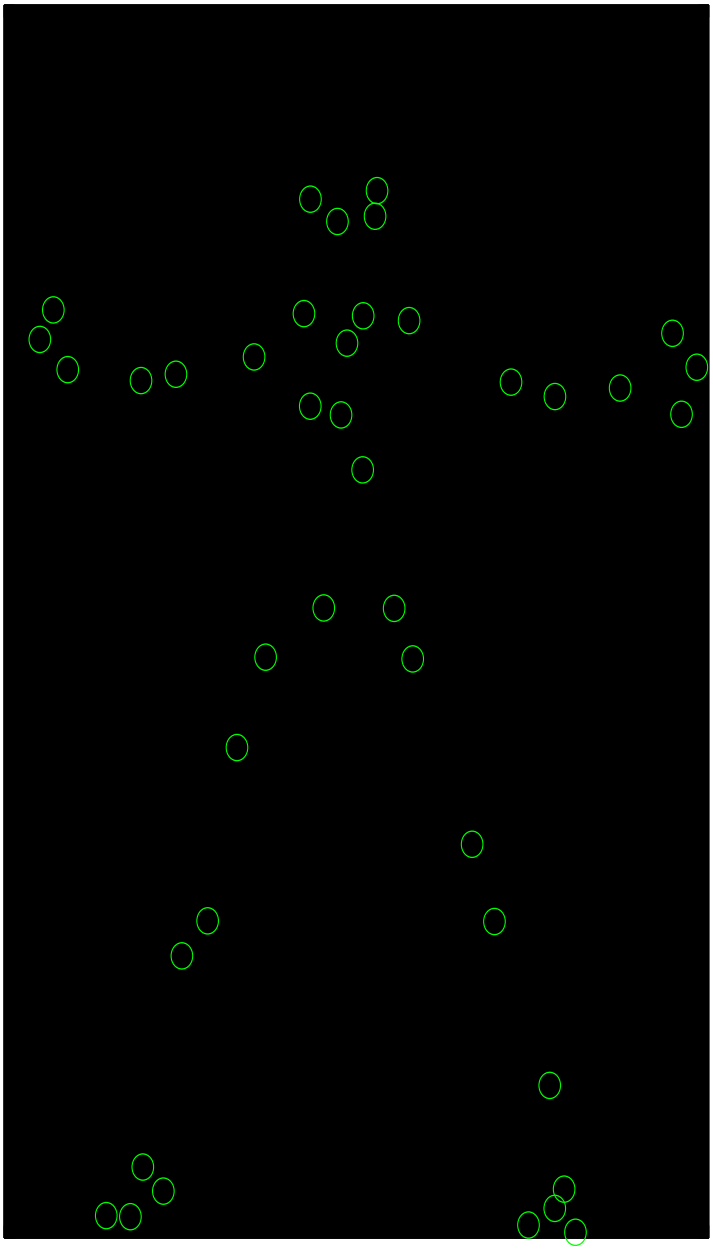


Figure 4.4: Yoga data sequence

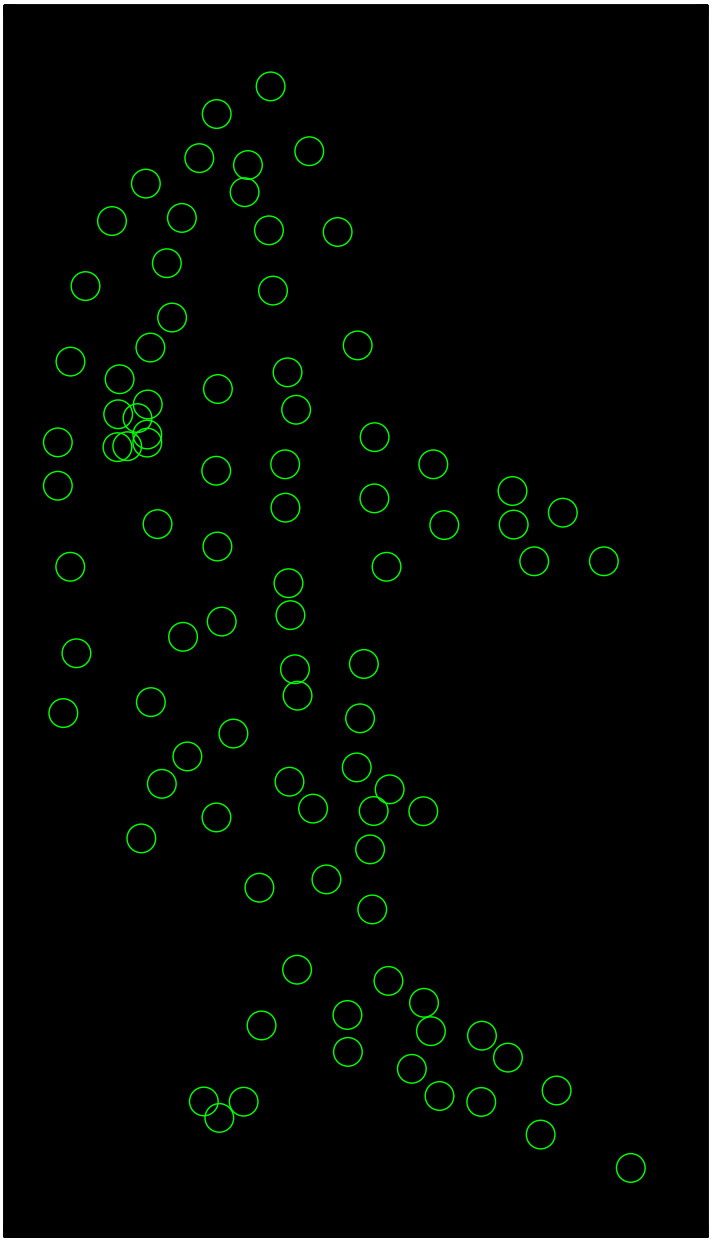


Figure 4.5: Shark data sequence

We adopt the same comparison metrics as [9][8][7] to ensure a fair

evaluation. In particular, the 3D reconstruction error e_{3D} is scaled by the average shape standard deviation

$$e_{3D} = \frac{1}{\sigma_{mean} NF} \sum_{i=1}^{F} \sqrt{\sum_{j=1}^{N} (S_{ij} - \hat{S}_{ij})^2} \tag{4.7}$$

where Sij is the j^{th} ground truth 3D point in the i^{th} frame, $\hat{S}_{ij}$ is that of the recovered shape and

$$\sigma_{mean} = \frac{1}{F} \sum_{i=1}^{F} \sigma(S_i) \tag{4.8}$$

where $\sigma(S_i)$ is the standard deviation of the 3D shape in frame i. In the implementation, this normalization step is realized by scaling the data matrix $\widehat{W}$ before reconstruction and scaling the ground truth before computing the 3D reconstruction error.

There is one important difference between our proposed method in terms of shifting the 3D reconstructed shape to be zero-mean. In [9][8][7], the data matrix $\widehat{W}$ is zero-mean shifted before reconstruction, so that the recovered shape is also zero-mean. For our patch based approach, the zero-mean shifting of the data matrices of the patches results in the various patches being zero-mean shifted. However, even after shifting back as described in section 4.3, the reconstructed global 3D shape is not zero-mean shifted. In computing the final 3D reconstruction error, the reconstructed global 3D shape needs to be zero-mean shifted.

Even though LRR was developed with rigid motion segmentation

in mind, it should in principle not be restricted to just rigid motion. As we have seen in chapter 3, LRR offers strong rigid motion segmentation performance in terms of both model selection and clustering. Here we explain how LRR can be positioned as a subspace segmentation approach to NRSFM.

LRR solves the low rank formulation to obtain a block diagonal representation matrix where each block represents an independent subspace. The number of subspace is given by the model selection scheme outlined in algorithm 1. LRR's model selection can be best described as finding the largest eigen gap in a robust manner. There is however, no increased penalty for higher number of motion or increased model complexity. For MB-FLoSS, the model complexity cost in section 3.4.2 will ensure higher cost with increased model complexity. With this model complexity cost, MB-FLoSS tends to give fewer number of subspaces compared to LRR. This is indeed observed in the experiments.

From the discussion in section 2.3.3.2, LRR solves the low rank formulation to obtain a block diagonal representation matrix where each block represents an independent subspace. The number of subspace is given by the model selection scheme outlined in algorithm 1. LRR's model selection can be best described as finding the largest eigen gap in a robust manner. There is however, no increased penalty for higher number of motion or increased model complexity. For MB-FLoSS, the model complexity cost described in section 3.4.2 will ensure higher cost with increased model complexity. With this model complexity cost, MB-FLoSS tends to give fewer number of subspaces

compared to LRR. This is indeed observed in the experiments.

We are interested in comparing how the different model selection strategies employed by MB-FLoSS and LRR affect the reconstruction accuracy. For LRR, the number of shape basis K_i for each subspace i is estimated similar to step 2 in algorithm 3. The k-means step is applied to the singular values of the data matrix $\widehat{W}_i$ of each subspace i. Since MB-FLoSS uses the same model selection and segmentation parameters for both the rigid and non-rigid cases, we do likewise for LRR for fairness consideration.

The different model selection strategies will result in different decomposition of the non-rigid body into its constituent components, both in terms of number of components and the points making up the components. These differences will likely give different reconstruction results. Furthermore, it would be interesting to evaluate quantitatively how the violation of the independent subspace assumption affects the reconstruction result, given the prevalent overlap and dependency between the subspaces in non-rigid motion.

With LRR's independent subspace assumption, there will be no overlap between the patches. Since the Z translation ambiguities cannot be resolved, individual patches cannot be stitched back together to form a reconstructed global shape. We therefore use the mean patch error to evaluate quantitatively the reconstruction error. For each patch i, the patch error e^i_{3D} is defined using (4.8). The mean patch error is then the average of all the patch error

$$e_{patch} = \sum_{i=1}^{n_S} e_{3D}^{i} \tag{4.9}$$

where n_S is the number of subspace(patch).

For both MB-FLoSS and LRR, each patch is reconstructed using SPF, since there is no in-house 3D reconstruction step in either of MB-FLoSS or LRR.

Clearly we should also compare with the classic state-of-the-art NRSFM algorithms such as SPF, CSF and TB etc. Such comparison is possible because MB-FLoSS is able to reconstruct a global 3D shape, as described in algorithm 5, instead of just disconnected patches.

We have to bear in mind that these classic state-of-the-art NRSFM algorithms have the benefit of the ground truth to determine the optimum number of shape basis K for the best reconstruction results. Even though CSF claims that K can be determined by increasing K until the orthonormality constraint of the rotation matrices holds to a pre-defined threshold, the reported reconstruction results in CSF are achieved using the ground truth. For completeness sake, we also compare against EM-PPCA and MP.

4.4.1 Number of subspace and subspace dimension

In view of the different model selection schemes for MB-FLoSS and LRR, we find it of interest to compare the number of subspaces generated from the model selection step of MB-FLoSS and LRR. Table 4.2 and figure 4.7 show the general trend that MB-FLoSS

tends to generate fewer number of subspaces compared to LRR.

With the exception of the face and shark sequences, the number of subspaces generated by LRR is significantly more than MB-FLoSS. Why the exception for the two sequences? The face and shark sequences are the only two non articulated motion sequences. They exhibit more of a continuously deforming type of motion that makes it difficult for LRR to segment them into large number of independent subspaces. For articulated motion sequences, it is easier to regard the overlap between the subspaces as noise.

Sequence	MB-FLoSS	LRR
Drink	3	8
Pick-up	4	11
Yoga	3	8
Stretch	4	13
Dance	5	11
Face	3	2
Walking	5	7
Shark	1	1

Table 4.2: Number of subspaces for MB-FLoSS and LRR

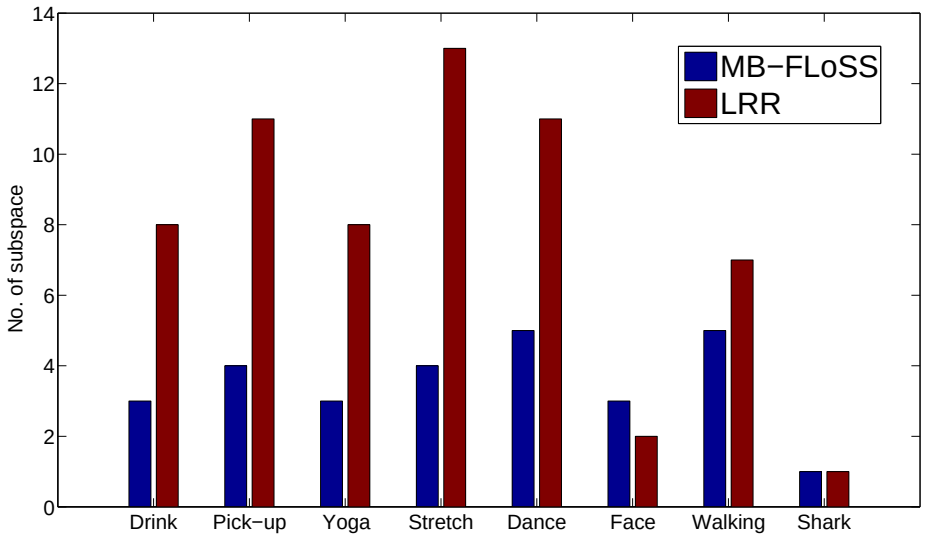


Figure 4.6: Number of subspace MB-FLoSS and LRR generate from model selection

Table 4.3 and figure 4.7 show that the average subspace dimension of MB-FLoSS is higher than LRR. For MB-FLoSS, the smaller number of subspace means each subspace needs to be higher dimension to suitably describe the motion corresponding to the subspace.

Sequence	MB-FLoSS	LRR
Drink	12.7	6.7
Pick-up	10.8	6.5
Yoga	8.3	5.6
Stretch	11.5	5.6
Dance	9.8	5.0
Face	7.0	10.0
Walking	5.6	4.4
Shark	8.0	6.0

Table 4.3: Average subspace dimension for MB-FLoSS and LRR

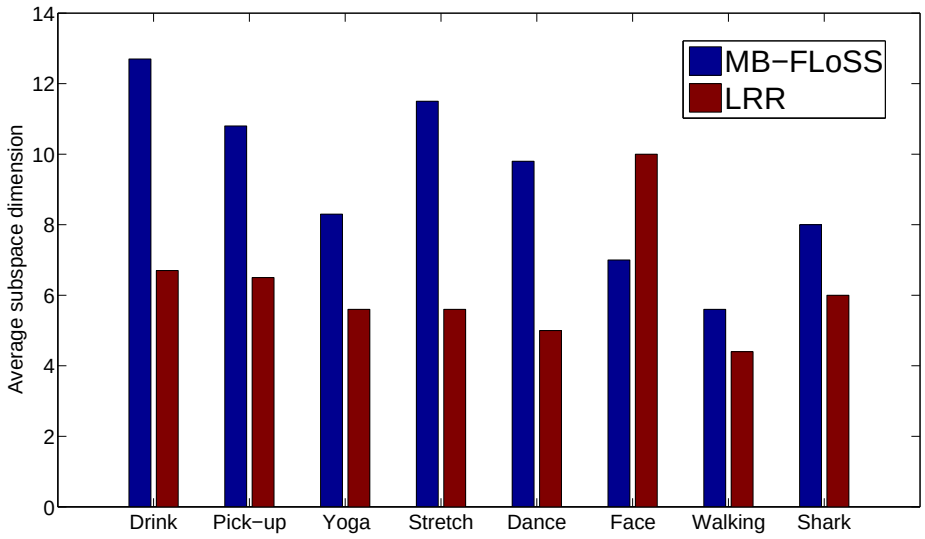


Figure 4.7: Average subspace dimension for MB-FLoSS and LRR

4.4.2 Reconstruction results

4.4.2.1 Mean patch error comparison

Figure 4.8 and table 4.4 compare the mean patch 3D reconstruction error between MB-FLoSS and LRR. Both MB-FLoSS and LRR use SPF for individual patch 3D reconstruction. The global SPF 3D reconstruction with assumed known number of shape basis is used as a baseline. Obviously the mean patch error enjoys additional degree of freedom and therefore potentially better performance when compared to a global approach like SPF. After all, there is likely to be additional error when the patches are stitched back together. But since the patches cannot be stitched back together in LRR, plus the fact that the comparison is between LRR and MB-FLoSS, the SPF ground truth serves as a useful reference and the mean patch error comparison is still meaningful.

Sequence	SPF	MB-FLoSS	LRR
Drink	0.027	0.038	0.277
Pick-up	0.173	0.196	0.170
Yoga	0.115	0.278	0.342
Stretch	0.103	0.120	0.202
Dance	0.186	0.143	0.041
Face	0.030	0.058	0.034
Walking	0.130	0.097	0.031
Shark	0.243	0.243	1.473

Table 4.4: Comparison of mean patch 3D reconstruction error of MB-FLoSS against LRR, with SPF as baseline

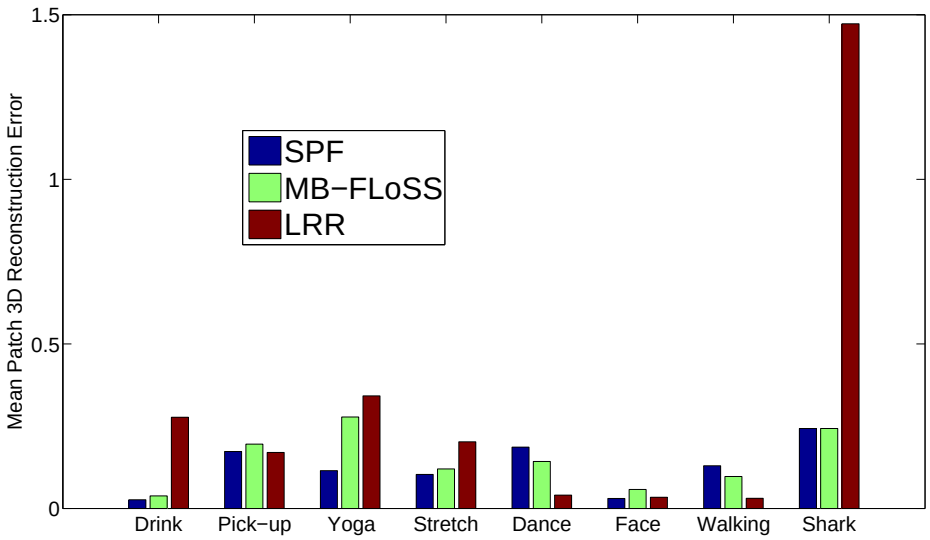


Figure 4.8: Comparison of mean patch 3D reconstruction error for LRR and MB-FLoSS, with SPF as the baseline

There are some interesting observations from these results. LRR tends to be extreme in its performance. When LRR gets it right for the dance and walking sequences, the mean patch error is spectacular

- approximately 4 times improvement over the baseline SPF results. On the other hand, when LRR gets its wrong for the shark and drink sequences, the error is off the chart - roughly 7 times worse than the baseline SPF.

MB-FLoSS tends to give a more constant performance when compared with the baseline. While MB-FLoSS gives a better performance for the dance and walking sequences compared to the SPF baseline, the improvement is not as dramatic compared to LRR. On the other end of the scale, MB-FLoSS's worst performance for the yoga sequence stands at roughly 3 times that of the baseline SPF. Significantly, this yoga sequence is MB-FLoSS's worst performance across all the sequences, but yet still outperforms LRR for this particular sequence.

From this comparison, it is fair to say that MB-FLoSS has a more consistently good performance and in general outperforms LRR. MB-FLoSS's better performance is not surprising, considering the fact that MB-FLoSS explicitly handles the overlap between subspaces in non-rigid motion.

For LRR, it seems that the model selection rule breaks down for the continuously deforming type of non-rigid motion such as the shark sequence, causing the number of shape basis to be estimated wrongly. For the shark sequence, both MB-FLoSS and LRR gave 1 subspace. MB-FLoSS estimated the number of shape basis to be 3, coinciding with the optimum number of shape basis for SPF. LRR wrongly estimated the number of shape basis to be 2, resulting in the bad reconstruction results.

4.4.2.2 Global reconstruction error comparison

The global 3D reconstruction error result is summarized in table 4.5 and figure 4.9. As we can see in table 4.5 and figure 4.9, MB-FLoSS has the same level of performance as the state-of-the-art SPF, TB and CSF for the dance, walking, shark and face sequences. However, for the other four sequences, drink, pickup, yoga and stretch, MB-FLoSS's performance does not compare as well. This is not surprising, considering the fact the additional step in stitching back the various patches will probably introduce errors. In addition, without the ground truth to choose the best number of shape basis, MB-FLoSS is unlikely to match stride for stride the performance of the state-of-the-art methods. To put things in perspective, without the benefit of the ground truth to choose the optimal number of shape basis, MB-FLoSS's performance is rather commendable.

Note that the 3D reconstruction error for the SPF block matrix method(BMM) using the Matlab code provided by the authors, as reported in table 4.5, is different from the SPF BMM reported in [9]. After clarifying with the author, it turns out that the discrepancy is due to the author modifying the Matlab code so that the sign ambiguity in the SVD step is fixed. Although this fix ensures that the code works across different Matlab version, it does however changes the 3D reconstruction error. The performance comparison in table 4.5 is based on the implementation with the SVD sign ambiguity fixed.

Sequence	EM-PPCA	MP	TB	CSF	SPF	MB-FLoSS
Drink	0.339	0.460	0.025	0.022	0.027	0.173
Pick-up	0.582	0.433	0.237	0.230	0.173	0.403
Yoga	0.810	0.804	0.162	0.147	0.115	0.337
Stretch	1.111	0.855	0.109	0.071	0.103	0.304
Dance	0.984	0.264	0.296	0.271	0.186	0.277
Face	0.033	0.036	0.044	0.036	0.030	0.058
Walking	0.492	0.561	0.395	0.186	0.130	0.182
Shark	0.050	0.157	0.180	0.008	0.243	0.243

Table 4.5: Comparison of 3D reconstruction error of MB-FLoSS against various other methods

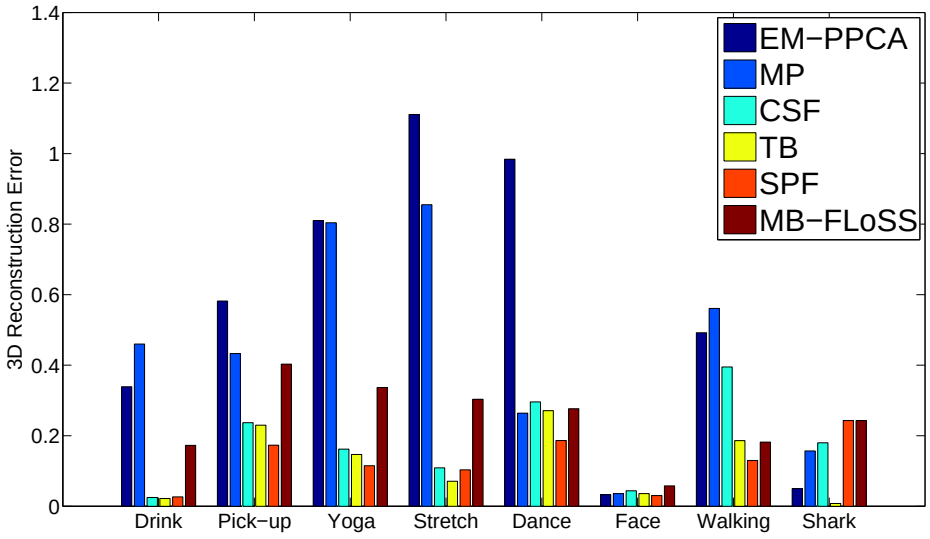


Figure 4.9: Comparison of MB-FLoSS global 3D reconstruction error against various other methods

4.4.3 Segmentation results

The segmentation of the trajectories brings up the interesting question of how the trajectories are grouped. We illustrate the segmen-

tation trend with the drink and dance sequences. Note that all the segmentations shown come from after step 2 of algorithm 4 i.e. after reclassifying the basis trajectories.

A frame of the drink sequence data matrix is shown in figure 4.10. The drink sequence involves the subject drinking using the left arm. The segmented drink sequence in figure 4.11 shows that the non-rigid motion is divided into three logical groups - the left part of the body in green, the right part of the body in blue and the lower torso in red. The left part of the body in green is segmented because of the left arm that executes the drinking motion. The lower torso is relatively stationary over the frames. The right arm move little but enough for the right arm to be segmented. Note that there is a wrongly classified blue point on the left leg.

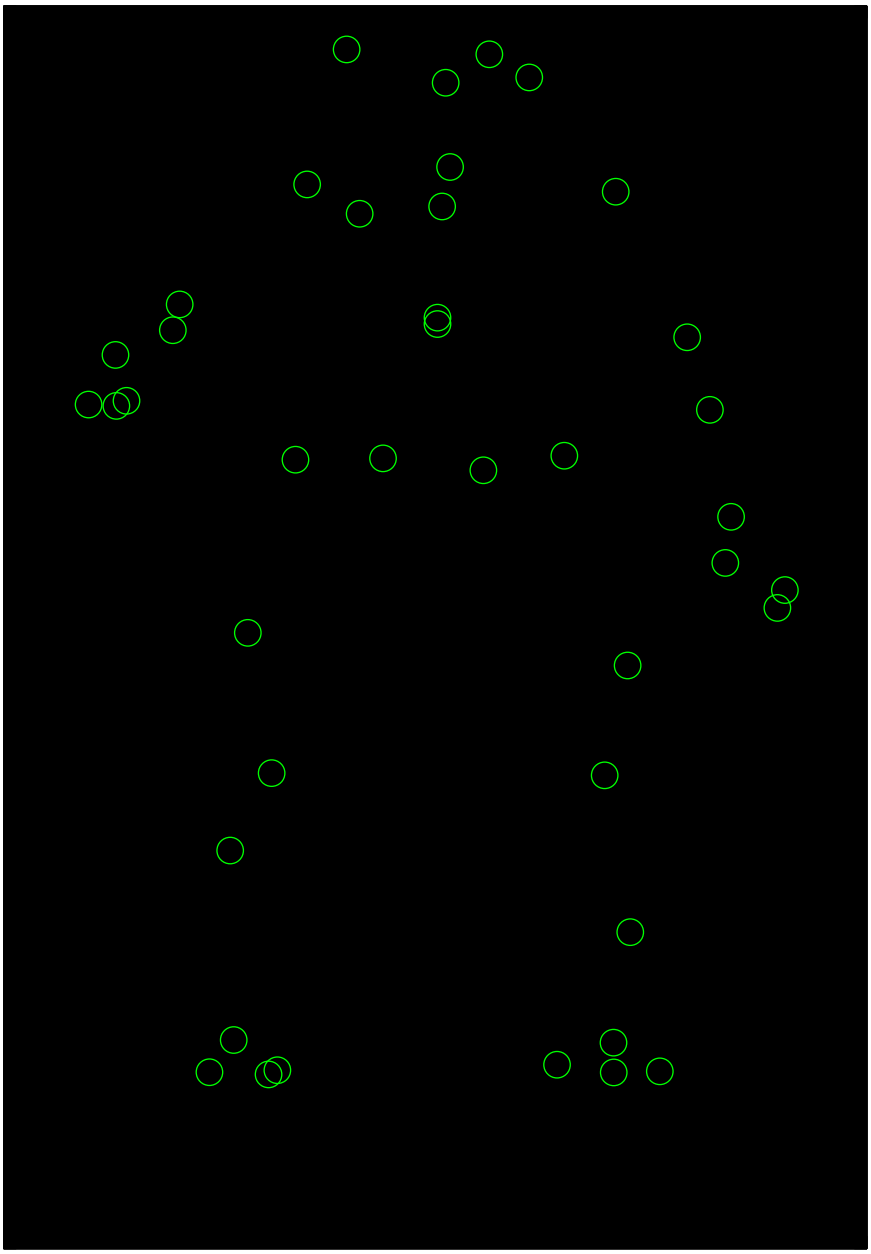


Figure 4.10: The drink sequence

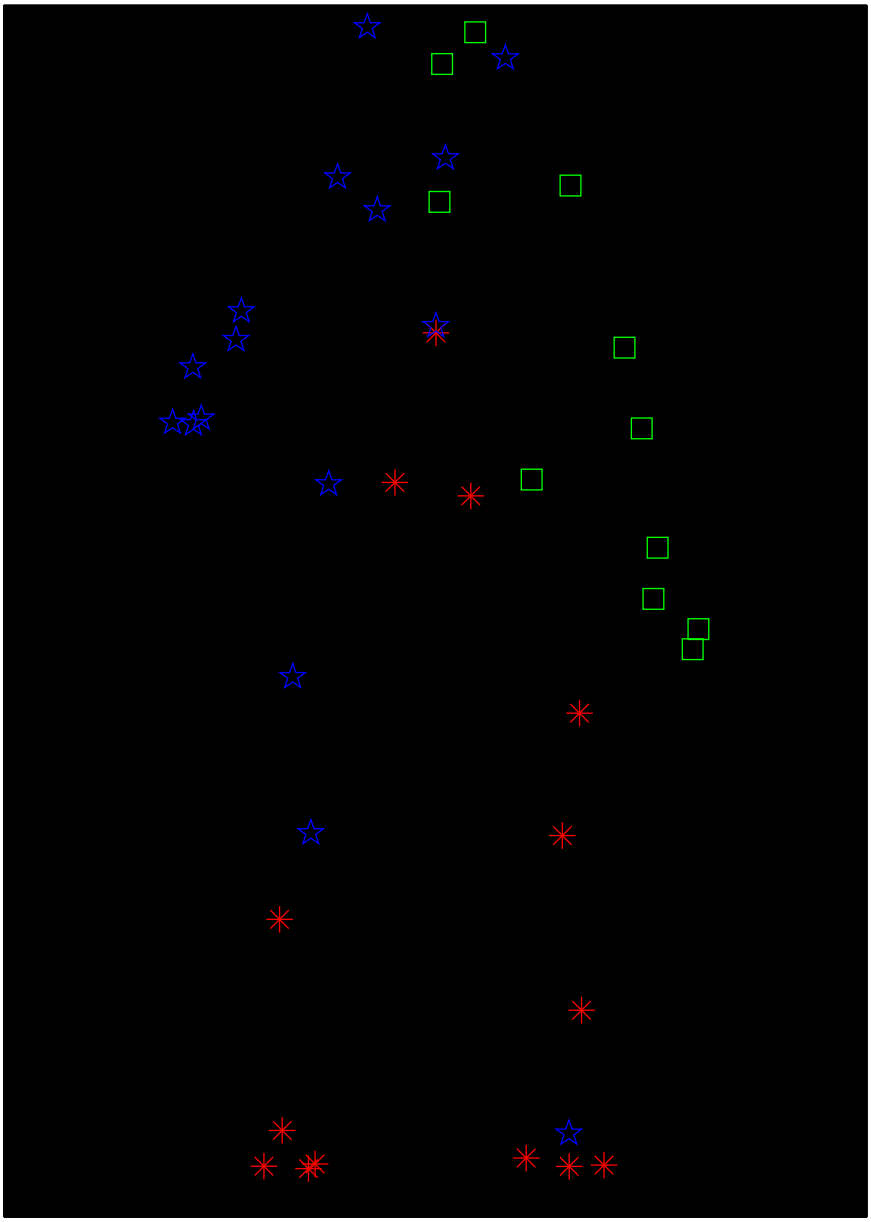


Figure 4.11: Subspace segmentation of the drink sequence using MB-FLoSS. The blue point on the left leg is due to misclassification

A frame of the dance sequence is shown in figure 4.12. The segmentation result in figure 4.13 shows the segmentation of the trajectories. The dance sequence involves large movement of the four limbs and small movement of the head. Since the four limbs are executing complex motions, it is no surprise for each limb to be segmented as a group. Even though there is movement of the head, the model complexity cost ensures that the head and the torso are treated as a single non-rigid body.

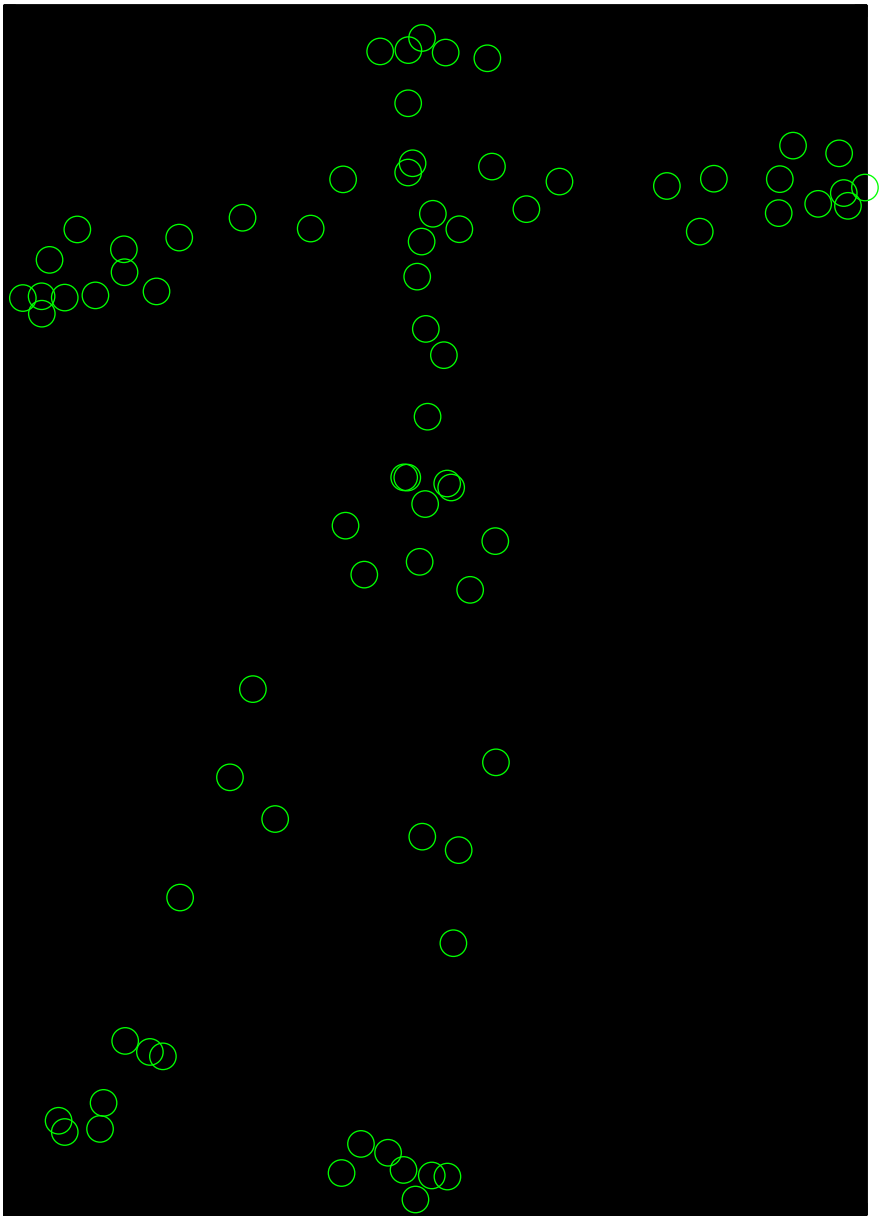


Figure 4.12: The dance sequence

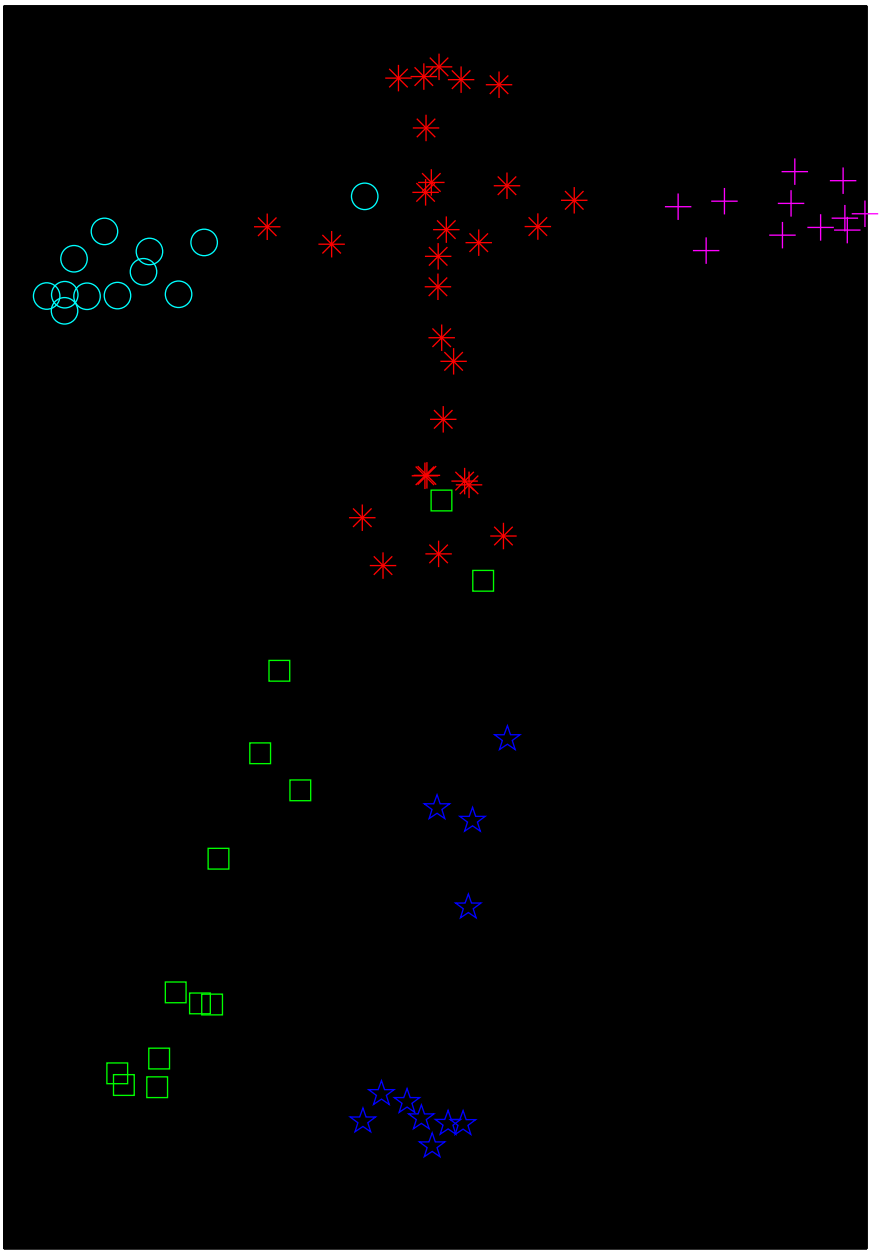


Figure 4.13: Subspace segmentation of the dance sequence using MB-FLoSS

We provide the dance sequence segmented by LRR in figure 4.14 for comparison. Note that even though the dance sequence was segmented into 11 groups in table 4.2, 3 of the groups have 2 or less trajectories. Since SPF requires at least 3 trajectories(corresponding to 1 shape basis), the 3 groups cannot be reconstructed. These 3 groups are therefore merged to the nearest subspace. LRR segments the non-rigid body into the head, torso, the limbs, with the left foot being further segmented into 2 groups. This finer division can be explained by the lack of a model complexity penalty in LRR's model selection mechanism. For example, while MB-FLoSS groups the head and torso as one group, LRR separates them into two groups.

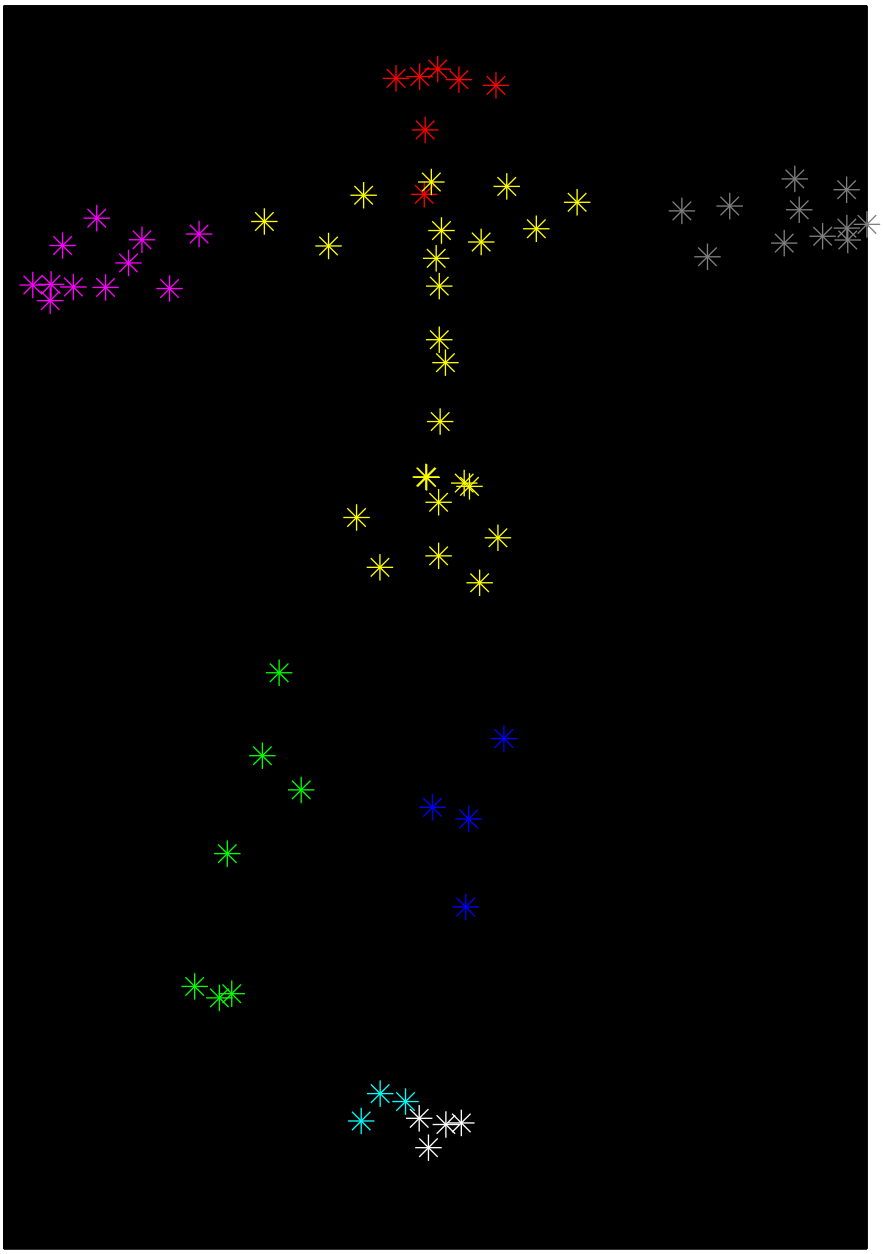


Figure 4.14: Subspace segmentation of the dance sequence using LRR

Both the segmentation by MB-FLoSS and LRR are reasonable and natural at some level. But as we have highlighted in section 4.1.5, MB-FLoSS is more flexible in the sense that it allows for non-rigid components and more importantly, allow the different parts to be assembled together to achieve global consistency.

Since MB-FLoSS identifies the trajectory basis explicitly, we are interested in which of the trajectories are chosen as basis and why they are chosen as basis. We look at the torso and right arm segments of the dance sequence in figure 4.15 for insights. Note that figure 4.13 and 4.15 come from a different frames. The torso is marked in red and the right arm is marked in blue. The trajectory bases shared by the two segments are colored in green.

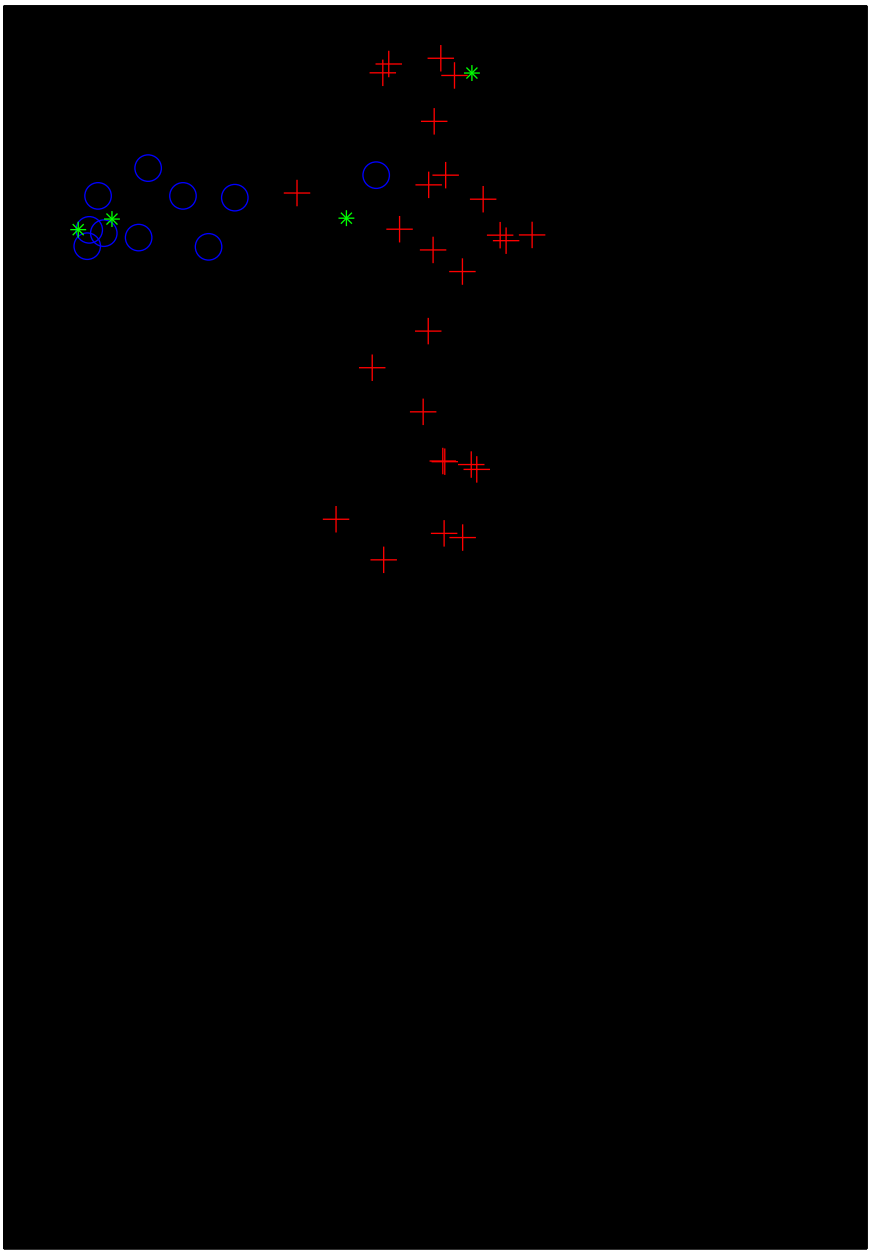


Figure 4.15: The bases shared by both the red and blue subspaces are colored in green

A natural and valid question is why is the torso trajectory serving as a basis for the right arm? And vice versa, why are the two bases on the right arm serving as bases for the torso? We can understand why this is the case by considering the fact that segments of human motion are inter-related and seldom independent. The head of the torso is obviously connected to the right arm and there will be motion correlation between the two. It is therefore not surprising that trajectories for one group serve as bases for the other group.

4.4.4 Multiple non-rigid body motion segmentation and reconstruction

In this section, we aim to verify the difference between MB-FLoSS and the shape basis approach in handling non-rigid motions without a rigid principle component. We construct this sequence without a rigid principle component by concatenating two vastly different non-rigid motion into one sequence. We identify the pickup and walking sequences as suitably different non-rigid motions that serve the purpose of our experiment. This concatenated two non-rigid body motion sequence is shown in figure 4.16.

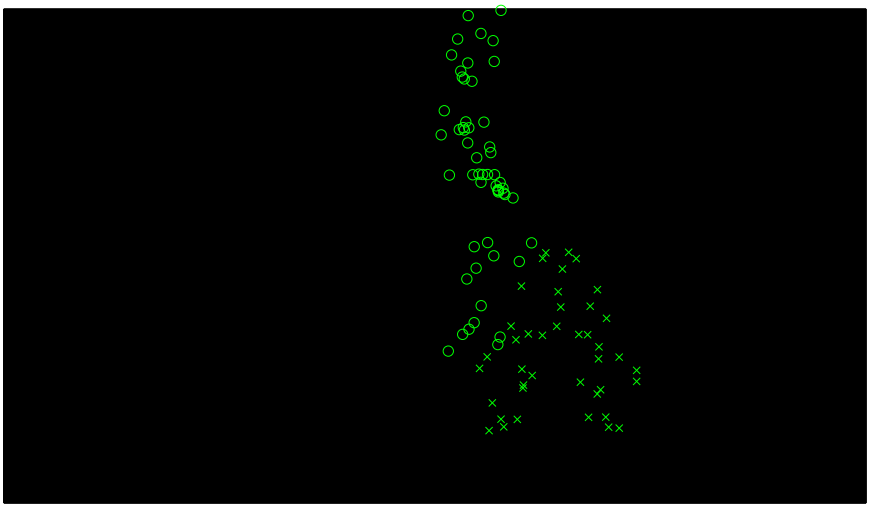


Figure 4.16: A two non-rigid body sequence obtained by concatenating the pickup and walking sequences. The walking sequence is marked in circles while the pickup sequence is marked in crosses

Our MB-FLoSS work handles such sequence as a multi-body non-rigid motion sequence. Just like the single non-rigid body's case, the trajectories are first segmented into the individual subspaces. We then iteratively merge pairs of subspaces by setting a threshold based on the maximum image distance over the entire trajectory. The intuition is that if this distance is small(below the threshold), then the pair of subspaces are likely to be from the same non-rigid motion. The threshold for this particular sequence is 0.5.

The result of the segmentation is shown in figure 4.17 which showed that the pickup motion(in red) has been successfully separated from the walking motion (in green). The pickup sequence is shown in red while the walking sequence is shown in green. After segmenting the trajectories into the individual non-rigid motion,

reconstruction can then be carried out for each non-rigid motion separately.

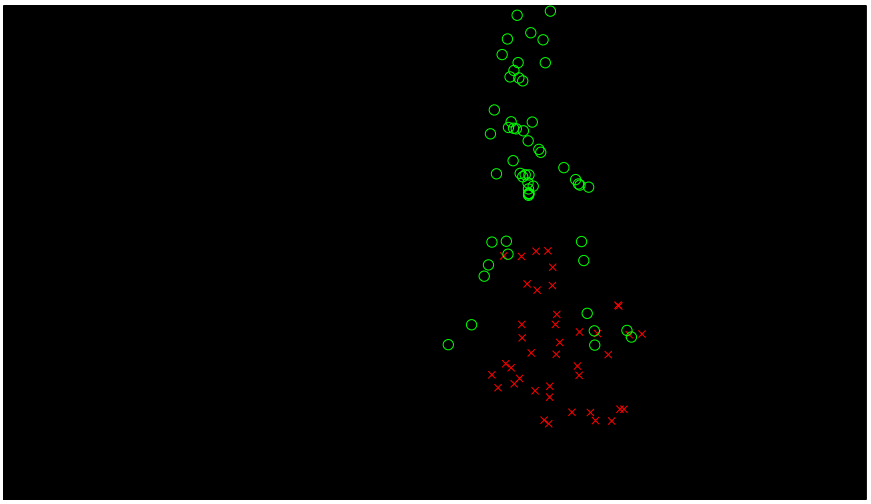


Figure 4.17: Segmentation of the concatenated pickup-walking sequence. The pickup sequence is marked in red while the walking sequence is marked in green

In this concatenated sequence, each sequence has its own motion - the camera is stationary in the walking sequence whereas the camera is rotating for the pickup sequence. If we still want to hold on to the shape basis assumption of a rigid principal component, even with these two vastly different non-rigid motion in the same sequence, the only possibility for the shape basis representation to handle this sequence is for the number of shape basis K to be large enough to treat the two different motions as a large deformation about a rigid principal component.

We wish to verify experimentally if increasing K large enough will yield good reconstruction results, thereby showing that the two

different non-rigid motions can be regarded as one non-rigid motion with large deformation about a rigid principal component. Note that K can only go up to $K_{max} = \lfloor \frac{N}{3} \rfloor$ where N is the number of trajectories. For this concatenated sequence, there are 96 points and hence $K_{max} = 32$. We choose TB for reconstructing this concatenated sequence due to its good performance and quick run-time. This good run-time performance is desirable since we have to vary K from 1 to 32 and find the K that offers the best reconstruction performance.

Table 4.6 shows that TB performs poorly in terms of reconstruction compared to MB-FLoSS. This result confirms that the two independent non-rigid motion cannot be approximated as a highly deforming non-rigid motion using the maximum number of shape basis.

	3D reconstruction error
MB-FLoSS	0.391
TB	2.717(28)

Table 4.6: Multi non-rigid body reconstruction error. For TB, the number of shape basis, shown in parenthesis, is obtained with the help of the ground truth

The reconstruction results for the individual non-rigid motion for MB-FLoSS is shown in table 4.7. Note that the reconstruction error for the pickup sequence in table 4.7 is different from table 4.5. The reason is that the pickup sequence, that has 357 frames, needs to be truncated to 260 frames so that it is the same length as the walking sequence, that has 260 frames.

	3D reconstruction error
pickup	0.671
walking	0.182

Table 4.7: The reconstruction error of the individual non-rigid motion after segmentation

From table 4.6, we can infer that MB-FLoSS is able to handle NRSFM without a rigid principal component while this lack of a rigid principal component presents a boundary where the shape basis approach breaks down.

4.5 Conclusion

In this work, we propose MB-FLoSS, a new subspace segmentation approach to NRSFM. We evaluated quantitatively the performance of MB-FLoSS against the state-of-the-art NRSFM methods and LRR, an alternative subspace segmentation based method that has competitive performance in both model selection and segmentation.

Due to the independent subspace assumption in LRR, there are no overlapping points between the subspaces. This means that LRR is only able to reconstruct each 3D patch locally but is unable to piece back these 3D patches to form a global shape. Nonetheless, we are still able to evaluate MB-FLoSS against LRR based on the mean patch error. We were able to establish that MB-FLoSS offers better performance in terms of the mean patch error compared to LRR.

Since only MB-FLoSS is able to stitch back the patches to obtain a global 3D reconstructed shape but not LRR, we compare MB-FLoSS

against the state-of-the-arts methods where the number of shape basis is assumed known. Without knowing the optimal number of shape basis, MB-FLoSS offers good reconstruction performance close to that of the baseline provided by the state-of-the-art methods. In terms of segmentation, we show that the MB-FLoSS's segmentation makes good sense and corresponds closely to human intuition. We are also able to gain insight into the shared basis between overlapping subspaces.

We pushed the boundary of the shape basis representation works by introducing a non-rigid motion sequence without a rigid principal component. Not surprisingly, this lack of a rigid principal component results in the shape basis representation works performing badly, even with the luxury of using the largest number of shape basis available. For the subspace segmentation approach, the subspaces belonging to the same non-rigid motion can be identified easily. This allows each non-rigid motion to be reconstructed independently, thus achieving good performance.

In conclusion, our new proposed subspace segmentation approach has been shown to provide strong, competitive NRSFM performance when measured against the state-of-the-art shape basis methods, but crucially, without the need for ground truth to determine the best number of shape basis or strong assumption about the presence of a rigid principle component.

Chapter 5

Summary and future works

5.1 Summary

We first motivate our thesis by highlighting the often neglected model selection aspect of motion segmentation. We explain the difficulty of model selection due to the independent subspace assumption that is inherent in the current methods. This independent subspace assumption may not be valid for many of the rigid motion sequences containing overlapping motion. For non-rigid motion, this assumption is an even bigger problem, since overlapping subspace is a given in non-rigid motion.

Instead of fixing the independent subspace assumption a posteriori, we chose to incorporate the presence of overlapping subspace into our model through the minimal basis representation. Unfortunately, this direct formulation of the minimal basis representation is NP-hard.

We solve this problem in two stages. We first obtain an approximate solution to the original problem using a convex proxy of the original problem. The use of the convex proxy comes at the price

of artefacts that prevents us from determining the basis decisively. We then make an important and non-trivial extension of the FLoSS framework to include a global model complexity cost model. This global cost model favors overlapping subspaces and is instrumental in MB-FLoSS's good performance in model selection.

For rigid body motion segmentation, we extend the de facto Hopkins 155 to Hopkins 380, testing both MB-FLoSS and LRR extensively. The experimental results show MB-FLoSS achieving a better performance than LRR. This difference in performance is not surprising, since many sequences in the Hopkins 380 data set contain overlapping motion.

We show the universality of our model selection mechanism by applying the same exact mechanism and parameters to NRSFM. This model selection mechanism is at the heart of the our new proposed subspace segmentation based approach to NRSFM. While NRSFM is a well studied problem and has seen good performance from the shape basis factorization methods, they all assume known number of shape basis. Our new approach shows good 3D reconstruction performance without knowing the optimum number of shape basis. This new approach also has the additional advantage of not requiring the presence of a rigid principal component, thereby expanding the range of non-rigid motion that can be handled by the mainstream NRSFM works.

In rigid motion segmentation, the overlapping subspace formulation improves the model selection performance. For NRSFM, the overlapping subspace formulation becomes indispensable. The over-

lapping trajectory basis allows MB-FLoSS to stitch the patches back into a reconstructed global 3D shape. For LRR, the lack of shared points means that the individual reconstructed patches cannot be stitched back together. Nevertheless, based on the mean patch error metric, we show that MB-FLoSS compares favorably against LRR.

All in all, we have shown the successful application of the same model selection mechanism in MB-FLoSS across both the rigid and non-rigid domain.

5.2 Future works

The most immediate and urgent need for future motion segmentation work is benchmark data set. The Hopkins data set has been around for quite a while and seemingly outlive its usefulness. When there are methods that can achieve sub 1% misclassification, any further improvement is insignificant and doubts will arise if improvements are due to over-fitting. In the new data set, we hope to see more varied types of motion, scenes with strong perspective effects, presence of missing data and outliers.

For NRSFM, the need for a benchmark data set is even more apparent. Currently, NRSFM mostly rely on about 8-10 sequences for experimental verification. A NRSFM's equivalent of the Hopkins data set containing more varied type of non-rigid motion will certainly help spur development, especially sequences with larger/nonlinear deformations.

A more ambitious long term goal would be the handling of missing entries and/or large outliers. While there have been exciting devel-

opments in matrix completion with sparse outliers, it is much more challenging for the motion segmentation's case. Without knowing the number of motion, filling in the missing entries in the data matrix becomes a much more difficult task, especially when there are large outliers.

On this note, we would like to conclude this thesis.

Appendix A

Appendix: Lipschitz constant derivation

We will show how the Lipschitz constant can be derived for C since E can be done similarly. For any arbitrary $C_1, C_2 \in \text{domain} f(C, E)$, the Lipschitz condition for C is

$$\left\|\nabla_C f(C_1, E^k) - \nabla_C f(C_2, E^k)\right\|_F \leq L_C \|C_1 - C_2\| \tag{A.1}$$

First we obtain an expression for the gradient of $f(.)$ with respect to C

$$\nabla_C f(C) = \lambda \widehat{W} + \rho \widehat{W}^T \left(\widehat{W} C + E^k - \widehat{W}\right) \tag{A.2}$$

Then the Lipschitz constant for C can be estimated as

$$\left\|\nabla_C f(C_1, E^k) - \nabla_C f(C_2, E^k)\right\|_F = \rho \left\|\widehat{W}^T\widehat{W}\left(C_1 - C_2\right)\right\|_F \quad \text{(A.3)}$$

$$= \rho \left\|\widehat{W}^T\widehat{W}\right\|_2 \left\|C_1 - C_2\right\|_F \quad \text{(A.4)}$$

$$= L_C \left\|C_1 - C_2\right\|_F \quad \text{(A.5)}$$

where $\left\|\widehat{W}^T\widehat{W}\right\|_2$ denotes the operator norm of $\widehat{W}^T\widehat{W}$ i.e. the largest singular value of $\widehat{W}^T\widehat{W}$ and $L_C = \rho \left\|\widehat{W}^T\widehat{W}\right\|_2$ is the estimated Lipschitz constant for C.

Appendix B

Appendix: Message passing derivation

Message update for ϕ

Following definition, the ϕ messages are written as:

$$\begin{aligned}\phi_j(1) &= \mu_{-C\to e_j}(1) \\ &= \max_{e_k, k\neq j}\left[-C(e_1,\ldots,e_j=1,\ldots,e_M)+\sum_{k\neq j}\xi_k(e_k)\right] \end{aligned} \tag{B.1}$$

$$\begin{aligned}\phi_j(0) &= \mu_{-C\to e_j}(0) \\ &= \max_{e_k, k\neq j}\left[-C(e_1,\ldots,e_j=0,\ldots,e_M)+\sum_{k\neq j}\xi_k(e_k)\right] \end{aligned} \tag{B.2}$$

$C(e_1, \ldots, e_j, \ldots, e_M)$ is effectively a feasibility function that restricts only one, two, or three facilities to be turned on. For (B.1), since e_j is set as 1, we are looking for all combinations of zero, one or two other e_j's being turned on. For (B.2), e_j is kept fixed as 0, we are then looking for all combinations of one, two, or three e_j's being turned on.

Even though (B.1) and (B.2) look combinatorial, the messages can be simplified and updated efficiently. We first observe that finding the max can be achieved by searching for the indices corresponding to the largest one, two or three $\xi_k = \xi_k(1) - \xi_k(0)$, for $k = 1, \ldots, M, k \neq j$. We sort $\{\xi_k = \xi_k(1) - \xi_k(0), k = 1, \ldots, M, k \neq j\}$ in descending order. Let $\hat{\xi}$ be the sorted $\{\xi_k = \xi_k(1) - \xi_k(0), k = 1, \ldots, M, k \neq j\}$ and the top three sorted entries be $\hat{\xi}_1, \hat{\xi}_2, \hat{\xi}_3$. Recall that the sorted set $\hat{\xi}$ and resultant top three indices exclude j and hence $\xi_j \notin \{\hat{\xi}_1, \hat{\xi}_2, \hat{\xi}_3\}$. In addition, $\hat{\xi}$ only has $M - 1$ number of entries, since index j was omitted.

For ease of notation, we define the cumulative sum operator S_{ij}:

$$S_{ij}(\{0,1\}) = \sum_{k=i}^{j} \hat{\xi}_k(\{0,1\}) \tag{B.3}$$

$$\text{e.g.} \quad S_{11}(\{0,1\}) = \hat{\xi}_1(\{0,1\}) \tag{B.4}$$

$$S_{12}(\{0,1\}) = \sum_{k=1}^{2} \hat{\xi}_k(\{0,1\}) \tag{B.5}$$

$$S_{13}(\{0,1\}) = \sum_{k=1}^{3} \hat{\xi}_k(\{0,1\}) \tag{B.6}$$

$$S_{23}(\{0,1\}) = \sum_{k=2}^{3} \hat{\xi}_k(\{0,1\}) \tag{B.7}$$

The omit cumulative sum operator $\tilde{S}_i$, where the lower index i indicates the indices from 1 to i that are omitted in the summation:

$$\tilde{S}_i(0) = \sum_{k=i+1}^{M-1} \hat{\xi}_k(0) \tag{B.8}$$

$$\text{e.g.} \quad \tilde{S}_0(0) = \sum_{k=1}^{M-1} \hat{\xi}_k(0) \tag{B.9}$$

$$\tilde{S}_1(0) = \sum_{k=2}^{M-1} \hat{\xi}_k(0) \tag{B.10}$$

$$\tilde{S}_2(0) = \sum_{k=3}^{M-1} \hat{\xi}_k(0) \tag{B.11}$$

$$\tilde{S}_3(0) = \sum_{k=4}^{M-1} \hat{\xi}_k(0) \tag{B.12}$$

In the derivations below, we will use the following identity frequently:

$$\tilde{S}_i(0) - \tilde{S}_j(0) = S_{(i+1)j}(0) \tag{B.13}$$

The differential cost between cost C_i and cost C_j is defined as

$$\delta_{ij} = C_i - C_j \tag{B.14}$$

For (B.1), since facility j is turned on, either one, two or three other facilities are turned on:

$$\phi_j(1) = \max_{e_k, k\neq j} \left[-C(e_1, \ldots, e_j = 1, \ldots, e_M) + \sum_{k\neq j} \xi_k(e_k) \right] \tag{B.15}$$

$$= \max \left[\overbrace{-C_1 + \tilde{S}_0(0)}^{\text{1 facility}}, \overbrace{-C_2 + S_{11}(1) + \tilde{S}_1(0)}^{\text{2 facilities}}, \overbrace{-C_3 + S_{12}(1) + \tilde{S}_2(0)}^{\text{3 facilities}}, \overbrace{-C_4 + S_{13}(1) + \tilde{S}_3(0)}^{\text{4 facilities}} \right] \tag{B.16}$$

For (B.2), since facility j is turned off, either one, two or three other facilities are turned on:

$$\phi_j(0) = \max_{e_k, k\neq j} \left[-C(e_1, \ldots, e_j = 0, \ldots, e_M) + \sum_{k\neq j} \xi_k(e_k) \right] \tag{B.17}$$

$$= \max \left[\overbrace{-C_1 + S_{11}(1) + \tilde{S}_1(0)}^{\text{1 facility}}, \overbrace{-C_2 + S_{12}(1) + \tilde{S}_2(0)}^{\text{2 facilities}}, \overbrace{-C_3 + S_{13}(1) + \tilde{S}_3(0)}^{\text{3 facilities}}, \overbrace{-C_4 + S_{14}(1) + \tilde{S}_4(0)}^{\text{4 facilities}} \right] \tag{B.18}$$

We change the order of evaluation in computing $\phi_j = \phi_j(1) - \phi_j(0)$ by moving $\phi_($0) into each term of $\phi_($1).

Moving $\phi_j(0)$ into the first term of $\phi_j 1$:

$$\left[-C_1 + \tilde{S}_0(0)\right] - \phi_j(0) \tag{B.19}$$

$$= \max \begin{cases} -S_{11}(1) + \tilde{S}_0(0) - \tilde{S}_1(0) \\ (C_2 - C_1) - S_{12}(1) + \tilde{S}_0(0) - \tilde{S}_2(0) \\ (C_3 - C_1) - S_{13}(1) + \tilde{S}_0(0) - \tilde{S}_3(0) \\ (C_4 - C_1) - S_{14}(1) + \tilde{S}_0(0) - \tilde{S}_4(0) \end{cases} \tag{B.20}$$

$$= \max \begin{cases} -[S_{11}(1) - S_{11}(0)] \\ \delta_{21} - [S_{12}(1) - S_{12}(0)] \\ \delta_{31} - [S_{13}(1) - S_{13}(0)] \\ \delta_{41} - [S_{14}(1) - S_{14}(0)] \end{cases} \tag{B.21}$$

$$= -\max\left[S_{11},\ S_{12} - \delta_{21},\ S_{13} - \delta_{31},\ S_{14} - \delta_{41}\right] \tag{B.22}$$

Moving $\phi_j(0)$ into the second term of $\phi_j(1)$:

$$\Big[-C_2 + S_{11}(1) + \tilde{S}_1(0)\Big] - \phi_j(0) \tag{B.23}$$

$$= \max \begin{cases} -(C_2 - C_1) \\ S_{11}(1) - S_{12}(1) + \tilde{S}_1(0) - \tilde{S}_2(0) \\ (C_3 - C_2) + S_{11}(1) - S_{13}(1) + \tilde{S}_1(0) - \tilde{S}_3(0) \\ (C_4 - C_2) + S_{11}(1) - S_{14}(1) + \tilde{S}_1(0) - \tilde{S}_4(0) \end{cases} \tag{B.24}$$

$$= \max \begin{cases} -\delta_{21} \\ -[S_{22}(1) - S_{22}(0)] \\ \delta_{32} - [S_{23}(1) - S_{23}(0)] \\ \delta_{42} - [S_{24}(1) - S_{24}(0)] \end{cases} \tag{B.25}$$

$$= -\max\left[\delta_{21},\ S_{22},\ S_{23} - \delta_{32},\ S_{24} - \delta_{42}\right] \tag{B.26}$$

Moving $\phi_j(0)$ into the third term of $\phi_j(1)$:

$$\Big[-C_3 + S_{12}(1) + \tilde{S}_2(0)\Big] - \phi_j(0) \tag{B.27}$$

$$= \max \begin{cases} -(C_3 - C_1) + S_{12}(1) - S_{11}(1) + \tilde{S}_2(0) - \tilde{S}_1(0) \\ -(C_3 - C_2) \\ S_{12}(1) - S_{13}(1) + \tilde{S}_2(0) - \tilde{S}_3(0) \\ (C_4 - C_3) + S_{12}(1) - S_{14}(1) + \tilde{S}_2(0) - \tilde{S}_4(0) \end{cases} \tag{B.28}$$

$$= \max \begin{cases} -\delta_{31} + [S_{22}(1) - S_{22}(0)] \\ -\delta_{32} \\ -[S_{33}(1) - S_{33}(0)] \\ \delta_{43} - [S_{34}(1) - S_{34}(0)] \end{cases} \tag{B.29}$$

$$= -\max\left[\delta_{31} - S_{22},\ \delta_{32},\ S_{33},\ S_{34} - \delta_{43}\right] \tag{B.30}$$

Moving $\phi_j(0)$ into the fourth term of $\phi_j(1)$:

$$\left[-C_4 + S_{14}(1) + \tilde{S}_4(0)\right] - \phi_j(0) \tag{B.31}$$

$$= \max \begin{cases} -(C_4 - C_1) + S_{13}(1) - S_{11}(1) + \tilde{S}_3(0) - \tilde{S}_1(0) \\ -(C_4 - C_2) + S_{13}(1) - S_{12}(1) + \tilde{S}_3(0) - \tilde{S}_2(0) \\ -(C_4 - C_3) \\ S_{13}(1) - S_{14}(1) + \tilde{S}_3(0) - \tilde{S}_4(0) \end{cases} \tag{B.32}$$

$$= \max \begin{cases} -\delta_{41} + [S_{23}(1) - S_{23}(0)] \\ -\delta_{42} + [S_{33}(1) - S_{33}(0)] \\ -\delta_{43} \\ -[S_{44}(1) - S_{44}(0)] \end{cases} \tag{B.33}$$

$$= -\max\left[\delta_{41} - S_{23},\ \delta_{42} - S_{33},\ \delta_{43},\ S_{44}\right] \tag{B.34}$$

The ϕ_j message update can now be simplified as

$$\phi_j = \max \begin{cases} -\max\left[\hat{\xi}_1,\ (\hat{\xi}_1 + \hat{\xi}_2) - \delta_{21},\ (\hat{\xi}_1 + \hat{\xi}_2 + \hat{\xi}_3) - \delta_{31},\ (\hat{\xi}_1 + \hat{\xi}_2 + \hat{\xi}_3 + \hat{\xi}_4) - \delta_{41}\right] \\ -\max\left[\delta_{21},\ \hat{\xi}_2,\ (\hat{\xi}_2 + \hat{\xi}_3) - \delta_{32},\ (\hat{\xi}_2 + \hat{\xi}_3 + \hat{\xi}_4) - \delta_{42}\right] \\ -\max\left[\delta_{31} - \hat{\xi}_2,\ \delta_{32},\ \hat{\xi}_3,\ (\hat{\xi}_3 + \hat{\xi}_4) - \delta_{43}\right] \\ -\max\left[\delta_{41} - (\hat{\xi}_2 + \hat{\xi}_3),\ \delta_{42} - \hat{\xi}_3,\ \delta_{43},\ \hat{\xi}_4\right] \end{cases} \tag{B.35}$$

The underlying pattern can be more easily discerned by discarding

the max symbols and presenting the various entries in a matrix:

$$\begin{pmatrix} S_{11} & S_{12}-\delta_{21} & S_{13}-\delta_{31} & S_{14}-\delta_{41} \\ \delta_{21} & S_{22} & S_{23}-\delta_{32} & S_{24}-\delta_{42} \\ \delta_{31}-S_{22} & \delta_{32} & S_{33} & S_{34}-\delta_{43} \\ \delta_{41}-S_{23} & \delta_{42}-S_{33} & \delta_{43} & S_{44} \end{pmatrix} \tag{B.36}$$

This pattern can be generalized to K facilities

$$\begin{pmatrix} S_{11} & S_{12}-\delta_{21} & \dots & \dots & \dots & \dots & \dots & S_{1K}-\delta_{K1} \\ \delta_{21} & S_{22} & S_{23}-\delta_{32} & \dots & \dots & \dots & \dots & S_{2K}-\delta_{K2} \\ \delta_{31}-S_{22} & \delta_{32} & S_{33} & S_{34}-\delta_{43} & \dots & \dots & \dots & S_{3K}-\delta_{K3} \\ \delta_{41}-S_{23} & \delta_{42}-S_{33} & \delta_{43} & S_{44} & \dots & \dots & \dots & S_{4K}-\delta_{K4} \\ \vdots & \vdots & \vdots & \ddots & \ddots & \dots & \dots & \dots \\ \vdots & \vdots & \vdots & \vdots & \ddots & \ddots & \dots & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \ddots & \dots \\ \delta_{K1}-S_{2(K-1)} & \delta_{K2}-S_{3(K-1)} & \dots & \dots & \dots & \dots & \delta_{K(K-1)} & S_{KK} \end{pmatrix} \tag{B.37}$$

Message update for ξ

$$\begin{aligned} \xi_j(1) &= \mu_{E_j \to e_j}(1) \\ &= \max_{h_{:j}} \left[E_j(h_{:j}, e_j = 1) + \sum_i \rho_{ij}(h_{ij}) \right] \end{aligned} \tag{B.38}$$

$$\xi_j(0) = \mu_{E_j \to e_j}(0)$$

$$= \max_{h_{:j}} \left[E_j(h_{:j}, e_j = 0) + \sum_i \rho_{ij}(h_{ij}) \right] \tag{B.39}$$

$$= \sum_i \rho_{ij}(0), \text{ since facility } j \text{ is not turned on} \tag{B.40}$$

$$\xi_j = \xi_j(1) - \xi_j(0) \tag{B.41}$$

$$= \max_{h_{:j}} \left[E_j(h_{:j}, e_j = 1) + \sum_i \rho_{ij}(h_{ij}) \right] - \sum_i \rho_{ij}(0) \tag{B.42}$$

$$= \max_{h_{:j}} \left[E_j(h_{:j}, e_j = 1) + \sum_i \rho_{ij}(h_{ij}) - \sum_i \rho_{ij}(0) \right] \tag{B.43}$$

$$= \rho_{kj} + \sum_{i \neq k} \max(0, \rho_{ij}) \tag{B.44}$$

since at least one facility must be turned on and k is the index of the largest ρ value.

Since e_j is turned on, one of the h_{ij} must be turned on as well, otherwise the consistency constraint (3.7) is violated. The max operation is therefore taken over all combinations of h_{ij}, with at least one of the h_{ij}'s set to 1. The one h_{ij} turned on can be readily identified as the largest value in ρ, say ρ_k. The rest of the h_{ij}'s are turned on only if the corresponding ρ_{ij} values are non-negative

For efficient Matlab implementation, ξ_j can be written as

$$\begin{aligned} & \rho_{kj} + \sum_{i \neq k} \max(0, \rho_{ij}) \\ &= \rho_{kj} - \max(0, \rho_{kj}) + \sum_{i} \max(0, \rho_{ij}) \quad \text{(B.45)} \\ &= \min(0, \rho_{kj}) + \sum_{i} \max(0, \rho_{ij}), \quad \text{(B.46)} \end{aligned}$$

using the relationship $x - \max(x, y) = \min(0, x - y)$ (B.47)

Message update for α

The other message update that is affected by the global facility function and the subspace overlap bonus function is α.

$$\alpha_{ij}(1) = \max_{h_{kj}, k \neq i} \left[E(h_{1j}, \ldots, h_{ij} = 1, \ldots, h_{Nj}, e_j) + \sum_{k \neq i} \rho_{kj}(h_{kj}) + \phi_j(e_j) \right] \quad \text{(B.48)}$$

$$= \sum_{k \neq i} \max_{h_{kj}} \rho_{kj}(h_{kj}) + \omega_j(1) \quad \text{(B.49)}$$

$$\alpha_{ij}(0) = \max_{h_{kj}, k \neq i} \left[E(h_{1j}, \ldots, h_{ij} = 0, \ldots, h_{Nj}, e_j) + \sum_{k \neq i} \rho_{kj}(h_{kj}) + \phi_j(e_j) \right] \tag{B.50}$$

$$= \max \left[\sum_{k \neq i} \max_{h_{kj}} \rho_{kj}(h_{kj}) + \phi_j(1), \sum_{k \neq i} \rho_{kj}(0) + \phi_j(0) \right] \tag{B.51}$$

$$\alpha_{ij} = \alpha_{ij}(1) - \alpha_{ij}(0) \tag{B.52}$$

$$= \min \left[0, \sum_{k \neq i} \max_{h_{kj}} \rho_{kj}(h_{kj}) + \phi_j(1) - \sum_{k \neq i} \rho_{kj}(0) - \phi_j(0) \right] \tag{B.53}$$

$$= \min \left[0, \sum_{k \neq i} \max(0, \rho_{kj}) + \phi_j \right] \tag{B.54}$$

Appendix C

Appendix: Simple prior free method

In SPF, the data matrix is decomposed into a product of the motion matrix and shape basis matrix

$$\widehat{W} = \begin{bmatrix} R_1 S_1 \\ \vdots \\ R_F S_F \end{bmatrix} \tag{C.1}$$

$$= \begin{pmatrix} c_{11}R_1 & \dots & c_{1K}R_1 \\ \vdots & \ddots & \vdots \\ c_{F1}R_F & \dots & c_{FK}R_F \end{pmatrix} \begin{pmatrix} B_1 \\ \vdots \\ B_K \end{pmatrix} \tag{C.2}$$

$$= \Pi B \tag{C.3}$$

$\Pi \in \mathbb{R}^{2F \times 3K}$ is the motion matrix that contains the rotation matrices scaled by the shape coefficients and $B \in \mathbb{R}^{3K \times N}$ is the shape basis.

The idea behind the shape basis representation is that the 3D shape in each frame can be expressed as a linear combination of the shape basis. This can be expressed as

$$S(t) = \sum_{i=1}^{K} c_i(t) B_i, \quad t = 1, \ldots, F \tag{C.4}$$

where $\mathbb{R}^{3\times 3K} \ni S(t)$ is the 3D shape in frame t, $\mathbb{R}^{3\times N} \ni B_i$'s are the shape bases, $\mathbb{R} \ni c_i(t)$ is the coefficient associated with the i^{th} shape basis in frame t and K is the number of shape basis. The full 3D shape representation in terms of the shape basis can be expressed as

$$S = (C \otimes I_3)B \tag{C.5}$$

where $\mathbb{R}^{F\times K} \ni C$ is the shape coefficient matrix, $\otimes$ is the Kronecker product, $\mathbb{R}^{3\times 3} \ni I_3$ is the 3×3 identity matrix and B is the shape basis matrix given in (C.3).

Note that each two consecutive rows of Π are the rows of the same rotation matrix for the frame repeated K times, scaled by different shape basis coefficients. For example, the first two rows of Π consist of R_1 repeated K times but scaled by different shape basis coefficients.

In SPF, the data matrix is first decomposed through SVD as $\widehat{W} = \hat{\Pi}\hat{B}$. In the rectifying transform step, the orthogonality constraint is written as

$$\hat{\Pi}_{2i-1} Q_k \hat{\Pi}_{2i-1}^T = \hat{\Pi}_{2i} Q_k \hat{\Pi}_{2i}^T \tag{C.6}$$

$$\hat{\Pi}_{2i-1} Q_k \hat{\Pi}_{2i}^T = 0, \quad i = 1 \dots F,\ k = 1 \dots K \tag{C.7}$$

where $\mathbb{R}^{3K \times 3K} \ni Q_k = G_k G_k^T$ and $\mathbb{R}^{3K \times 3} \ni G_k$ is the k^{th} column triplet of G restoring the orthogonality constraint across all frames

$$\hat{\Pi} G_k = c_{ik} R_i, \quad i = 1, \dots F,\ k = 1 \dots K \tag{C.8}$$

Note that due to the structure of Π, only one such G_k column triplet will be needed for the rectifying transformation, not the entire G. SPF formulated the orthogonality constraint in terms of null space representation

$$A q_k = 0 \tag{C.9}$$

where $\mathbb{R}^{2F \times 9K^2} \ni A$ is constructed from $\hat{\Pi}$ and $\mathbb{R}^{9K^2} \ni q_k$ is derived by vectorizing Q_k. Based on this null space representation, SPF proves the intersection theorem, that shows that Q_k is positive definite and at most rank 3. Previously, metric upgrading is achieved by solving a system of homogeneous equations. With this intersection theorem, metric upgrading is formulated as a rank minimization problem with constraints. Using nuclear norm as the convex proxy, Q_k is solved as a convex optimization problem

$$\min_{Q_k} \text{trace}(Q_k) \tag{C.10}$$

$$s.t.\ Q_k \succeq 0 \tag{C.11}$$

$$Aq_k = 0$$

After solving for Q_k, the rotation matrix of each frame $R_i,\ i = 1 \dots F$, can be recovered by scaling rows of $\hat{\Pi}_{2i-1:2i} G_k,\ i = 1 \dots F$ to unit length.

The shape recovery part in SPF is more elaborate compared to other methods. A key observation is that the shape structure matrix S is at most rank $3K$, where K is the number of shape basis. Recovering S is similarly formulated as a rank minimization problem.

$$\min_{S} \text{rank}(S) \tag{C.12}$$

$$s.t.\ \widehat{W} = RS$$

$$Aq_k = 0$$

where

$$R = \begin{bmatrix} R_1 & & \\ & \ddots & \\ & & R_F \end{bmatrix} \tag{C.13}$$

Using results from compressive sensing, S can be solved uniquely by taking the pseudo-inverse of R

$$S = R^T(RR^T)^{-1}\widehat{W} \tag{C.14}$$

Further rank constraint can be imposed on S by re-arranging S into

$$S^{\#} = \begin{bmatrix} X_{11} & \dots & X_{1N} & Y_{11} & \dots & Y_{1N} & z_{11} & \dots & z_{1N} \\ \vdots & & \vdots & \vdots & & \vdots & \vdots & & \vdots \\ X_{F1} & \dots & X_{FN} & Y_{F1} & \dots & Y_{FN} & z_{F1} & \dots & z_{FN} \end{bmatrix} \tag{C.15}$$

Since $S^{\#}$ is at most rank K, we can further refine S by solving the optimization problem

$$\begin{aligned} &\min_{S^{\#}} \operatorname{rank}(S^{\#}) \\ &s.t.\ \widehat{W} = RS \\ &\qquad S^{\#} \text{ is a re-arrangement of S} \end{aligned} \tag{C.16}$$

Once again, SPF solves this NP-hard optimization problem by using the well-known convex nuclear norm proxy.

References

[1] C. Tomasi and T. Kanade, *Shape and Motion from Image Streams: A Factorization Method. Detection and Tracking of Point Features.* School of Computer Science, Carnegie Mellon University, 1991. 5

[2] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004. 5

[3] J. Yan and M. Pollefeys, "A factorization-based approach for articulated nonrigid shape, motion and kinematic chain recovery from video," *PAMI*, vol. 30, no. 5, pp. 865–877, 2008. 6, 7, 77, 84, 85, 89, 92

[4] C. Russell, J. Fayad, and L. Agapito, "Energy based multiple model fitting for non-rigid structure from motion," in *CVPR*, pp. 3009–3016, 2011. 6, 82, 83, 85, 86, 90, 91, 92

[5] J. Fayad, C. Russell, and L. Agapito, "Automated articulated structure and 3d shape recovery from point correspondences," in *ICCV*, pp. 431–438, 2011. 6, 77, 84, 85, 86, 89, 90, 92

[6] L. Torresani, A. Hertzmann, and C. Bregler, "Nonrigid structure-from-motion: Estimating shape and motion with hierarchical priors," *PAMI*, vol. 30, no. 5, pp. 878–892, 2008. 7, 78, 79

[7] I. Akhter, Y. Sheikh, S. Khan, and T. Kanade, "Nonrigid structure from motion in trajectory space," in *NIPS*, 2008. 7, 78, 107, 108

[8] P. F. Gotardo and A. M. Martinez, "Computing smooth time trajectories for camera and deformable shape in structure from motion with occlusion," *PAMI*, vol. 33, no. 10, pp. 2051–2065, 2011. 7, 79, 107, 108

[9] Y. Dai, H. Li, and M. He, "A simple prior-free method for non-rigid structure-from-motion factorization," in *CVPR*, pp. 2018–2025, 2012. 7, 107, 108, 117

[10] C. Tomasi and T. Kanade, "Shape and motion from image streams under orthography: a factorization method," *IJCV*, vol. 9, no. 2, pp. 137–154, 1992. 7, 44, 77, 84

[11] G. Liu, Z. Lin, and Y. Yu, "Robust subspace segmentation by low-rank representation," in *ICML*, pp. 663–670, 2010. 9, 12, 33, 36, 48

[12] T. Chin, H. Wang, and D. Suter, "The ordered residual kernel for robust motion subspace clustering," in *NIPS*, 2009. 9, 40, 42, 70

[13] T. Chin, D. Suter, and H. Wang, "Multi-structure model selection via kernel optimisation," in *CVPR*, pp. 3586–3593, 2010. 9, 42

[14] K. Kanatani, "Geometric information criterion for model selection," *IJCV*, vol. 26, no. 3, pp. 171–189, 1998. 9

[15] P. H. Torr, "Geometric motion segmentation and model selection," *Philosophical Transactions of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, vol. 356, no. 1740, pp. 1321–1340, 1998. 9

[16] E. Elhamifar and R. Vidal, "Sparse subspace clustering," in *CVPR*, pp. 2790–2797, 2009. 12, 33, 40, 47

[17] E. Candes and T. Tao, "Decoding by linear programming," *IEEE Transactions on Information Theory*, vol. 51, no. 12, pp. 4203–4215, 2005. 12

[18] E. Candes and B. Recht, "Exact matrix completion via convex optimization," *Foundations of Computational Mathematics*, vol. 9, no. 6, pp. 717–772, 2009. 12

[19] E. J. Candes, X. Li, Y. Ma, and J. Wright, "Robust principal component analysis?," *Journal of the ACM (JACM)*, vol. 58, no. 3, p. 11, 2011. 12

[20] M. Grant and S. Boyd, "CVX: Matlab software for disciplined convex programming." `http://cvxr.com/cvx`, Sept. 2013. 13

[21] M. R. Hestenes, "Multiplier and gradient methods," *Journal of optimization theory and applications*, vol. 4, no. 5, pp. 303–320, 1969. 14

[22] M. J. Powell, *" A method for non-linear constraints in minimization problems".* UKAEA, 1967. 14

[23] J. Nocedal and S. Wright, *Numerical optimization.* Springer, 2006. 14, 54

[24] R. Rockafellar, *Convex analysis.* Princeton Univ Pr, 1997. 14

[25] Z. Lin, M. Chen, and Y. Ma, "The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices," *arXiv preprint arXiv:1009.5055*, 2010. 15

[26] D. Gabay and B. Mercier, "A dual algorithm for the solution of nonlinear variational problems via finite element approximation," *Computers & Mathematics with Applications*, vol. 2, no. 1, pp. 17–40, 1976. 18

[27] R. Glowinski and A. Marroco, "Sur l'approximation, par lments finis d'ordre un, et la rsolution, par pnalisation-dualit d'une classe de problmes de dirichlet non linaires," *ESAIM: Mathematical Modelling and Numerical Analysis-Modlisation Mathmatique et Analyse Numrique*, vol. 9, no. R2, pp. 41–76, 1975. 18

[28] R. Pietersz and P. J. Groenen, "Rank reduction of correlation matrices by majorization," *Quantitative Finance*, vol. 4, no. 6, pp. 649–662, 2004. 19

[29] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM Journal on Imaging Sciences*, vol. 2, no. 1, pp. 183–202, 2009. 19

[30] K. Toh and S. Yun, "An accelerated proximal gradient algorithm for nuclear norm regularized linear least squares problems," *Pacific J. Optim*, vol. 6, pp. 615–640, 2010. 19

[31] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 498–519, 2001. 21, 23

[32] C. M. Bishop, *Pattern recognition and machine learning*. Springer, 2007. 21, 23

[33] N. Lazic, B. Frey, and P. Aarabi, "Solving the uncapacitated facility location problem using message passing algorithms," in *AISTATS*, vol. 9, pp. 429–436, 2010. 26, 51, 63

[34] B. J. Frey and D. Dueck, "Clustering by passing messages between data points," *Science*, vol. 315, pp. 972–976, 2007. 26, 42, 50, 56

[35] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*, vol. 2. Cambridge Univ Press, 2000. 27

[36] E. Elhamifar and R. Vidal, "Sparse subspace clustering: Algorithm, theory, and applications," *PAMI*, 2013. 33, 47, 54

[37] G. Liu, Z. Lin, S. Yan, J. Sun, Y. Yu, and Y. Ma, "Robust recovery of subspace structures by low-rank representation," *PAMI*, vol. 34, no. 11, 2012. 33, 36, 37, 38, 40, 41, 54

[38] D.-S. Pham, S. Budhaditya, D. Phung, and S. Venkatesh, "Improved subspace clustering via exploitation of spatial constraints," in *CVPR*, pp. 550–557, 2012. 35, 47

[39] G. Liu, "Low-rank representation matlab code." https://sites.google.com/site/guangcanliu/. 37, 38, 69

[40] A. Ng, M. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," in *NIPS*, vol. 2, pp. 849–856, 2002. 40

[41] U. Von Luxburg, "A tutorial on spectral clustering," *Statistics and Computing*, vol. 17, no. 4, pp. 395–416, 2007. 40

[42] F. Chung, *Spectral graph theory*. Amer Mathematical Society, 1997. 40

[43] F. Lauer and C. Schnorr, "Spectral clustering of linear subspaces for motion segmentation," in *ICCV*, pp. 678–685, 2009. 40, 46

[44] B. Nadler and M. Galun, "Fundamental limitations of spectral clustering," in *NIPS*, vol. 19, p. 1017, 2007. 41

[45] N. Lazic, I. Givoni, B. Frey, and P. Aarabi, "Floss: Facility location for subspace segmentation," in *ICCV*, pp. 825–832, 2009. 42, 51, 56, 57

[46] J. P. Costeira and T. Kanade, "A multibody factorization method for independently moving objects," *IJCV*, vol. 29, no. 3, pp. 159–179, 1998. 44, 48, 77

[47] J. Yan and M. Pollefeys, "A general framework for motion segmentation: Independent, articulated, rigid, non-rigid, degenerate and non-degenerate," in *ECCV*, pp. 94–106, 2006. 45, 84

[48] L. Zelnik-Manor and M. Irani, "Degeneracies, dependencies and their implications in multi-body and multi-sequence factorizations," in *CVPR*, vol. 2, pp. II–287, 2003. 45

[49] A. M. Cheriyadat and R. J. Radke, "Non-negative matrix factorization of partial track data for motion segmentation," in *ICCV*, pp. 865–872, 2009. 47

[50] B. Nasihatkon and R. Hartley, "Graph connectivity in sparse subspace clustering," in *CVPR*, pp. 2137–2144, 2011. 47

[51] Y. Ni, J. Sun, X. Yuan, S. Yan, and L. Cheong, "Robust low-rank subspace segmentation with semidefinite guarantees," in *ICDM Workshops*, pp. 1179–1188, 2010. 48

[52] W. Siming and L. Zhouchen, "Analysis and improvement of low rank representation for subspace segmentation," tech. rep., 2011. 48

[53] T. Li, V. Kallem, D. Singaraju, and R. Vidal, "Projective factorization of multiple rigid-body motions," in *CVPR*, pp. 1–6, 2007. 48

[54] B. Triggs, "Factorization methods for projective structure and motion," in *CVPR*, pp. 845–851, 1996. 48

[55] J. Oliensis and R. Hartley, "Iterative extensions of the sturm/triggs algorithm: Convergence and nonconvergence," *PAMI*, vol. 29, no. 12, pp. 2217–2233, 2007. 48

[56] K. Kanatani and C. Matsunaga, "Estimating the number of independent motions for multibody motion segmentation," in *Asian Conference on Computer Vision*, pp. 7–12, 2002. 48, 70

[57] R. Vidal, Y. Ma, and J. Piazzi, "A new gpca algorithm for clustering subspaces by fitting, differentiating and dividing polynomials," in *CVPR*, vol. 1, pp. I–510, 2004. 49

[58] I. Givoni and B. Frey, "A binary variable model for affinity propagation," *Neural computation*, vol. 21, no. 6, pp. 1589–1600, 2009. 50, 52, 63, 64

[59] D. Dueck and B. Frey, "Non-metric affinity propagation for unsupervised image categorization," in *ICCV*, pp. 1–8, 2007. 51

[60] H. Li, "Two-view motion segmentation from linear programming relaxation," in *CVPR*, pp. 1–8, 2007. 51, 52

[61] D. Tarlow, I. Givoni, and R. Zemel, "Hopmap: Efficient message passing with high order potentials," *AISTATS*, 2010. 53

[62] M. Soltanolkotabi and E. Candes, "A geometric analysis of subspace clustering with outliers," *To appear in Annals of Statistics*, 2011. 55

[63] E. J. Candes, M. B. Wakin, and S. P. Boyd, "Enhancing sparsity by reweighted l1 minimization," *Journal of Fourier Analysis and Applications*, vol. 14, no. 5-6, pp. 877–905, 2008. 56

[64] I. Givoni, *Beyond Affinity Propagation: Message Passing Algorithms for Clustering*. Phd thesis, University of Toronto, 2011. 57, 63

[65] I. Givoni, C. Chung, and B. Frey, "Hierarchical affinity propagation," *arXiv preprint arXiv:1202.3722*, 2012. 63

[66] R. Tron and R. Vidal, "A benchmark for the comparison of 3-d motion segmentation algorithms," in *CVPR*, pp. 1–8, 2007. 68, 69

[67] J. Xiao, J.-x. Chai, and T. Kanade, "A closed-form solution to non-rigid shape and motion recovery," in *ECCV*, pp. 573–587, 2004. 75, 77, 78, 79

[68] J. Fayad, L. Agapito, and A. Del Bue, "Piecewise quadratic reconstruction of non-rigid surfaces from monocular sequences," in *ECCV*, pp. 297–310, 2010. 77, 81, 82, 92, 102

[69] C. Bregler, A. Hertzmann, and H. Biermann, "Recovering non-rigid 3d shape from image streams," in *CVPR*, vol. 2, pp. 690–696, 2000. 77

[70] M. Brand, "A direct method for 3d factorization of nonrigid motion observed in 2d," in *CVPR*, vol. 2, pp. 122–128, 2005. 78

[71] A. Del Bue, "A factorization approach to structure from motion with shape priors," in *CVPR*, pp. 1–8, 2008. 78

[72] A. Bjrck, *Numerical methods for least squares problems*. Siam, 1996. 78

[73] I. Akhter, Y. Sheikh, S. Khan, and T. Kanade, "Trajectory space: A dual representation for nonrigid structure from motion," *PAMI*, vol. 33, no. 7, pp. 1442–1456, 2010. 78, 79

[74] A. Zaheer, I. Akhter, M. H. Baig, S. Marzban, and S. Khan, "Multiview structure from motion in trajectory space," in *ICCV*, pp. 2447–2453, 2011. 79

[75] P. F. Gotardo and A. M. Martinez, "Non-rigid structure from motion with complementary rank-3 spaces," in *CVPR*, pp. 3065–3072, 2011. 79

[76] P. F. Gotardo and A. M. Martinez, "Kernel non-rigid structure from motion," in *ICCV*, pp. 802–809, 2011. 79

[77] I. Akhter, Y. Sheikh, and S. Khan, "In defense of orthonormality constraints for nonrigid structure from motion," in *CVPR*, pp. 1534–1541, 2009. 79

[78] M. Paladini, A. Del Bue, M. Stošic, M. Dodig, J. Xavier, and L. Agapito, "Factorization for Non-Rigid and Articulated Structure using Metric Projections," in *CVPR*, pp. 2898–2905, 2009. 79

[79] A. Edelman, T. A. Arias, and S. T. Smith, "The geometry of algorithms with orthogonality constraints," *SIAM journal on*

Matrix Analysis and Applications, vol. 20, no. 2, pp. 303–353, 1998. 80

[80] Z. Wen and W. Yin, "A feasible method for optimization with orthogonality constraints," *Mathematical Programming*, vol. 142, no. 1-2, pp. 397–434, 2013. 80

[81] A. Del Bue, J. Xavier, L. Agapito, and M. Paladini, "Bilinear modeling via augmented lagrange multipliers (balm)," *PAMI*, vol. 34, no. 8, pp. 1496–1508, 2012. 80

[82] A. Bartoli, V. Gay-Bellile, U. Castellani, J. Peyras, S. Olsen, and P. Sayd, "Coarse-to-fine low-rank structure-from-motion," in *CVPR*, pp. 1–8, 2008. 80

[83] R. White, K. Crane, and D. A. Forsyth, "Capturing and animating occluded cloth," *ACM Transactions on Graphics (TOG)*, vol. 26, no. 3, p. 34, 2007. 80, 92

[84] A. Varol, M. Salzmann, E. Tola, and P. Fua, "Template-free monocular reconstruction of deformable surfaces," in *CVPR*, pp. 1811–1818, 2009. 80

[85] J. Taylor, A. D. Jepson, and K. N. Kutulakos, "Non-rigid structure from locally-rigid motion," in *CVPR*, pp. 2761–2768, 2010. 81

[86] M. Bleyer, C. Rother, and P. Kohli, "Surface stereo with soft segmentation," pp. 1570–1577, 2010. 82

[87] A. Delong, A. Osokin, H. Isack, and Y. Boykov, "Fast approximate energy minimization with label costs," *International journal of computer vision*, vol. 96, no. 1, pp. 1–27, 2012. 82

[88] D. Hoiem, C. Rother, and J. Winn, "3d layoutcrf for multi-view object class recognition and segmentation," pp. 1–8, 2007. 82

[89] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *Journal of the Royal statistical Society*, vol. 39, no. 1, pp. 1–38, 1977. 83

[90] P. Tresadern and I. Reid, "Articulated structure from motion by factorization," in *CVPR*, vol. 2, pp. 1110–1115, 2005. 84

[91] P. H. Torr and D. W. Murray, "The development and comparison of robust methods for estimating the fundamental matrix," *IJCV*, vol. 24, no. 3, pp. 271–300, 1997. 84

[92] V. Rabaud and S. Belongie, "Re-thinking non-rigid structure from motion," in *CVPR*, pp. 1–8, 2008. 86, 87

[93] P. Dollr, V. Rabaud, and S. Belongie, "Non-isometric manifold learning: Analysis and an algorithm," in *ICML*, pp. 241–248, 2007. 86

[94] S. Zhu, L. Zhang, and B. M. Smith, "Model evolution: An incremental approach to non-rigid structure from motion," in *CVPR*, pp. 1165–1172, 2010. 86, 87

[95] M. Lee, J. Cho, C.-H. Choi, and S. Oh, "Procrustean normal distribution for non-rigid structure from motion," in *CVPR*, pp. 1280–1287, 2013. 87, 88, 89

[96] M. Paladini, A. Bartoli, and L. Agapito, "Sequential non-rigid structure-from-motion with the 3d-implicit low-rank shape model," in *ECCV*, pp. 15–28, 2010. 88, 89

[97] N. Hurley and S. Rickard, "Comparing measures of sparsity," *Information Theory, IEEE Transactions on*, vol. 55, no. 10, pp. 4723–4741, 2009. 96

[98] I. Akhter, Y. Sheikh, S. Khan, and T. Kanade, "Trajectory basis for nonrigid structure from motion project page." http://cvlab.lums.edu.pk/nrsfm/. 103, 104

[99] P. F. Gotardo and A. M. Martinez, "Csf code download." http://cbcsl.ece.ohio-state.edu/downloads.html. 103, 104

[100] "Cmu graphics lab motion capture database." http://mocap.cs.cmu.edu/. 105

[101] L. Torresani, A. Hertzmann, and C. Bregler, "Non-rigid structure-from-motion: Estimating shape and motion with hierarchical priors project page." http://www.cs.dartmouth.edu/~lorenzo/nrsfm.html. 105