Temporal Subspace Clustering for Unsupervised Action Segmentation

Larissa Clopton, Effrosyni Mavroudi, Dr. Manolis Tsakiris, Dr. Haider Ali, Dr. René Vidal

***Abstract*—Action segmentation (segmenting a continuous sequence of motion data into a set of actions) has a wide range of applications and plays a role in many problems in computer vision. We look at subspace clustering as an unsupervised approach for this task. Classical subspace clustering methods uncover relationships within the data by learning codes for the samples (i.e. frames), but in this process these methods do not consider the temporal dependency of nearby samples in motion data. In this paper, we propose two subspace clustering methods with temporal regularization terms: Temporal Sparse Subspace Clustering - Laplacian Regularization (TSSC-LR) and Temporal Sparse Subspace Clustering - Linear Combination (TSSC-LC). TSSC-LR encourages similar codings of samples within a window and TSSC-LC enforces codings within a window to be linear combinations of each other. Efficient ADMM algorithms are proposed for each method. Experiments against state-of-the-art methods on three action datasets demonstrate the effectiveness of the two proposed methods.**

I. Introduction

Motion capture is commonly used today in diverse applications, including healthcare, consumer electronics, and robotics. Dividing streams of motion data into meaningful segments (or actions) is a necessary first step before further analysis or synthesis. For example, supervised action recognition approaches depend on a large training set labeled with what actions occur and when. Few unsupervised approaches have been proposed for this task, so often this is done manually which is quite time-consuming and not scalable to larger problems. In this work, we look at subspace clustering as an unsupervised approach for action segmentation.

Subspace clustering follows the notion that high-dimensional data, for instance motion data, can be well approximated as a union of low-dimensional subspaces. Each subspace corresponds to a class, in the case of this work a specific action. Subspace clustering has emerged as a powerful technique for clustering data into multiple groups, showing impressive performance in face clustering [2], digit clustering [13], and motion segmentation [9]. The key idea in subspace clustering is to learn effective representation codings that are then used to construct an affinity matrix from which the data is grouped via spectral clustering. These representation codings often take form in a self-expressive model in which a sample is expressed as a linear combination of other samples. We can write this as $X = XC$, where $X = [x_1, x_2, ...x_N] \in \mathbb{R}^{D\times N}$ denotes the data matrix and $C \in \mathbb{R}^{N\times N}$ denotes the matrix of coefficients used to reconstruct the data (note the diagonal of this matrix will be 0). D denotes the dimension of each sample while N denotes the total number of samples.

Subspace clustering methods differ in the constraints they enforce on the coefficient/coding matrix C. For example, low-rank representation (LRR) [9] finds the lowest-rank representation of the data by minimizing the rank of C. On the other hand, sparse subspace clustering (SSC) [2] finds the sparsest representation of the data by minimizing the ℓ_1 norm of C. LRR and SSC are commonly used, representative subspace clustering methods.

Despite their impressive performance in other tasks, methods such as LRR and SSC struggle in action segmentation. Action segmentation is unique in that the motion data has temporal structure since nearby frames are likely to belong to the same action, and these representative subspace clustering methods do not take into account these relationships.

A. Our Contribution

In this paper, we propose two temporal subspace clustering methods for unsupervised action segmentation: Temporal Sparse Subspace Clustering - Laplacian Regularization (TSSC-LR) and Temporal Sparse Subspace Clustering - Linear Combination (TSSC-LC). Both methods consider the temporal relationships and local structural information found in motion data.

TSSC-LR extends SSC with a temporal Laplacian regularization term that encourages similar codings between a sample and its surrounding neighbors. TSSC-LC extends SSC with a temporal reguarlization term that more generally encourages the coding for a sample to be a linear combination of the codings from its surrounding neighbors.

The rest of the paper is organized as follows. In Section 2, we briefly discuss related works in temporal subspace clustering. Section 3 presents the details our two methods and their corresponding ADMM algorithms. Experimental results on three action datasets, all demonstrating the effectiveness of our proposed methods, are reported in Section 4. Section 5 concludes the paper and discusses opportunities for future work.

II. Related Work

In addition to the representative subspace clustering methods mentioned above, a handful of temporal subspace clustering methods have been proposed.

Ordered Subspace Clustering (OSC) [11] extends SSC by introducing a temporal regularization term that penalizes differences between consecutive codings via a matrix R, a lower triangular matrix with -1 on the diagonal and 1 on the second diagonal.

Temporal Subspace Clustering (TSC) [8] uses a similar temporal Laplacian regularization term to TSSC-LR, but it differs from TSSC-LR in two important ways: 1) TSC does not extend SSC, minimizing the Frobenius norm instead of the ℓ_1 norm. 2) TSC employs dictionary learning in place of the self-expressive model, which the authors claim allows for more expressive codings.

Temporal Smoothness Sequential Subspace Clustering (TSSSC) [10] derives a temporal regularization term motivated by the concept of temporal predictability as measured through a short-term moving average vs. a long-term moving average. This term enforces local smoothness in the codings. TSSSC also employs dictionary learning.

III. Our Approach

This section describes each of our temporal regularization terms in more detail, defines the respective objective function, and presents the corresponding ADMM algorithm to solve that objective function. Both methods extend SSC, so in each case the first term captures the reconstruction error, the second term encourages a sparse representation, and the third term enforces temporal regularization.

A. Temporal Sparse Subspace Clustering - Laplacian Regularization (TSSC-LR)

TSSC-LR utilizes a temporal Laplacian regularization term based on the observation that since nearby samples are likely similar, their new representations in the coding space should also be close. This term thus penalizes differences in the codings within a defined window. Given the coding matrix C, we define the temporal Laplacian regularization function as

$$f(C) = \frac{1}{2}\sum_{i=1}^{N}\sum_{i=1}^{N} w_{ij}||c_i - c_j||_2^2 = tr(CL_TC^T) \quad (1)$$

where L_T is a temporal Laplacian matrix, $L_T = D - W$, $D_{ii} = \sum_{j=1}^{N} w_{ij}$, and W is a weight matrix that captures temporal relationships in X. Let s denote the number of neighbors to consider for each sample (i.e. window size), then an element in W is calculated as

$$w_{ij} = \begin{cases} 1 & \text{if } |i-j| \le \frac{s}{2} \\ 0 & \text{otherwise} \end{cases}$$

Adjoining this term to a reconstruction error term and a sparsity term leads to the following objective function:

$$\min_{C} \frac{1}{2}||X - XC||_F^2 + \lambda_1||C||_1 + \frac{\lambda_2}{2} tr(CL_TC^T) \quad (2)$$

where λ_1 and λ_2 are trade-off parameters to balance the different terms.

To solve the objective function (2), we devise an alternating direction method of multipliers (ADMM) optimization algorithm. To facilitate optimization, we consider an equivalent form of (2):

$$\begin{aligned} &\min_{C,Z} \frac{1}{2}||X - XZ||_F^2 + \lambda_1||C||_1 + \frac{\lambda_2}{2} tr(ZL_TZ^T) \\ &\quad \text{s.t. } Z = C \end{aligned} \quad (3)$$

where Z is an auxiliary variable.

The augmented Lagrangian of (3) is

$$\mathcal{L}(C,Z,\Lambda) = \frac{1}{2}||X - XZ||_F^2 + \lambda_1||C||_1 + \frac{\lambda_2}{2}tr(ZL_TZ^T) + \langle\Lambda, Z - C\rangle + \frac{\mu}{2}||Z - C||_F^2 \tag{4}$$

where Λ is a matrix of Lagrange multipliers and $\mu > 0$ is a penalty parameter.

The ADMM algorithm for (4) is derived by alternatively minimizing $\mathcal{L}$ with respect to Z (C fixed), then with respect to C (Z fixed), and finally updating Λ based on the residual $Z - C$.

Update Z when fixing others. The problem (4) becomes

$$\min_Z \frac{1}{2}||X - XZ||_F^2 + \frac{\lambda_2}{2}tr(ZL_TZ^T) + \langle\Lambda, Z - C\rangle + \frac{\mu}{2}||Z - C||_F^2 \tag{5}$$

By setting the derivative of (5) with respect to Z to zero, we obtain the following equation:

$$(X^TX + \mu I)Z + Z(\lambda_2 L_T) = X^TX + \mu(C - \frac{\Lambda}{\mu}) \tag{6}$$

This is a standard Sylvester equation (AZ + ZB = C) which we solve using MATLAB's sylvester function.

Update C when fixing others. To find the optimal C given Z and Λ, we compute

$$C = S_{\frac{\lambda_1}{\mu}}(Z + \frac{\Lambda}{\mu}) \tag{7}$$

where S_ϵ is the shrinking threshold operator applied to each element in the matrix ([12], section 2.3).

The above process is repeated until convergence, further details of the ADMM algorithm can be found in Algorithm 1.

B. Temporal Sparse Subspace Clustering - Linear Combination (TSSC-LC)

The temporal regularization term for TSSC-LC is also motivated by the fact that nearby samples are likely similar; however, rather than encouraging that the codings within a window are the same, this method more generally enforces that these codings are linear combinations of each other. This is accomplished by minimizing the expression $||C - CM||_F^2$,

Algorithm 1 TSSC-LR

initialize: $C^0 = 0, \Lambda^0 = 0, \mu > 0$
while not converged **do**
    solve for Z^{k+1} (Sylvester equation):
    $(X^TX + \mu I)Z^{k+1} + Z^{k+1}(\lambda_2 L_T)$
    $= X^TX + \mu(C^k - \frac{\Lambda^k}{\mu})$
    $C^{k+1} = S_{\frac{\lambda_1}{\mu}}(Z^{k+1} + \frac{\Lambda^k}{\mu})$
    $\Lambda^{k+1} = \Lambda^k + \mu(Z^{k+1} - C^{k+1})$
end while
Input: data X, Output: coefficients C

where M assigns equal weight to past and future codings within the window (again defined by the parameter s). For example, a window size of 4 with 5 samples produces the following matrix:

$$M = \begin{bmatrix} 0 & 0.25 & 0.25 & 0 & 0 \\ 0.25 & 0 & 0.25 & 0.25 & 0 \\ 0.25 & 0.25 & 0 & 0.25 & 0.25 \\ 0 & 0.25 & 0.25 & 0 & 0.25 \\ 0 & 0 & 0.25 & 0.25 & 0 \end{bmatrix}$$

Other sophisticated graph weighting algorithms can also be applied to M to attain better performance.

Replacing the temporal Laplacian regularization term from TSSC-LR with this new term leads to the following objective function:

$$\min_C \frac{1}{2}||X - XC||_F^2 + \lambda_1||C||_1 + \frac{\lambda_2}{2}||C - CM||_F^2 \tag{8}$$

where again λ_1 and λ_2 are trade-off parameters to balance the different terms.

To solve the objective function (8), we devise a new alternating direction method of multipliers (ADMM) optimization algorithm. Similar to the previous method, to facilitate optimization, we consider an equivalent form of (8):

$$\min_{C,Z} \frac{1}{2}||X - XZ||_F^2 + \lambda_1||C||_1 + \frac{\lambda_2}{2}||Z - ZM||_F^2 \quad \text{s.t. } Z = C \tag{9}$$

where Z is an auxiliary variable.

The augmented Lagrangian of (9) is

$$\mathcal{L}(C,Z,\Lambda) = \frac{1}{2}||X - XZ||_F^2 + \lambda_1||C||_1 + \frac{\lambda_2}{2}||Z - ZM||_F^2 + \langle \Lambda, Z - C \rangle + \frac{\mu}{2}||Z - C||_F^2 \quad (10)$$

Update Z when fixing others. The problem (10) becomes

$$\min_Z \frac{1}{2}||X - XZ||_F^2 + \frac{\lambda_2}{2}||Z - ZM||_F^2 + \langle \Lambda, Z - C \rangle + \frac{\mu}{2}||Z - C||_F^2 \quad (11)$$

By setting the derivative of (11) with respect to Z to zero, we obtain the following equation:

$$(X^T X + \mu I)Z + Z(\lambda_2 (I - M)(I - M)^T) = X^T X + \mu(C - \frac{\Lambda}{\mu}) \quad (12)$$

This is also a standard sylvester equation.

Update C when fixing others. As before, we compute C as

$$C = S_{\frac{\lambda_1}{\mu}}(Z + \frac{\Lambda}{\mu}) \quad (13)$$

The above process is repeated until convergence, further details of the ADMM algorithm can be in Algorithm 2.

Algorithm 2 TSSC-LC

1: **initialize:** $C^0 = 0, \Lambda^0 = 0, \mu > 0$
2: **while** not converged **do**
3: solve for Z^{k+1} (Sylvester equation):
4: $(X^T X + \mu I)Z^{k+1} + Z^{k+1}(\lambda_2 (I - M)(I - M)^T)$
5: $= X^T X + \mu(C^k - \frac{\Lambda^k}{\mu})$
6: $C^{k+1} = S_{\frac{\lambda_1}{\mu}}(Z^{k+1} + \frac{\Lambda^k}{\mu})$
7: $\Lambda^{k+1} = \Lambda^k + \mu(Z^{k+1} - C^{k+1})$
8: **end while**
Input: data X, Output: coefficients C

C. Clustering

The coefficient matrix C is used to construct an affinity graph G for subspace clustering. In classical subspace clustering methods such as LRR and SSC, $G = \frac{|C| + |C|^T}{2}$. However, for time series data we can take advantage of the fact that neighbors are highly correlated to each other by using

$$G(i,j) = \frac{c_i^T c_j}{||c_i||_2 ||c_j||_2} \quad (14)$$

as the similarity measurement to construct G. Spectral clustering is then performed on G to produce the temporal clustering results.

IV. Experiments and Analysis

A. Datasets

We compared TSSC-LR and TSSC-LC against state-of-the-art subspace clustering methods on three action datasets. While there are a number of datasets used to assess performance of temporal subspace clustering methods in action segmentation, they are often recorded in a range of modalities. Here, we look at action datasets that contain skeleton/kinematic data.

1) CMU MOCAP: We use the first 14 sequences from subject 86 from the Carnegie Mellon University Motion Capture (CMU MOCAP) [1] database. Each sequence has up to 10 actions, including walk, squat, run, stand, jump, drink, and punch. The skeleton data was recorded at 120 Hz with a Vicon motion capture system consisting of 12 infrared MX-40 cameras. To speed up computation, we downsampled the data by a rate of 20, resulting in 6 frames per second. We also normalized each sample to be of unit norm.

2) CMU MAD: The Carnegie Mellon University Multimodal Action Database (CMU MAD) [6] contains RGB video, 3D depth, and skeleton data (3D coordinates of 20 joints) for 20 subjects performing two trials of the same sequence of actions. In each trial, the subject performs 35 actions that range from full-body motion (run, crouch, jump) to lower-body motion (kick) to upper-body motion (throw, basketball dribble, baseball swing). The skeleton data was recorded with a Microsoft Kinect. To speed up computation, we downsampled the data by a rate of 4. Following [3], we also normalized the skeleton data values to be between 0 and 1.

3) JIGSAWS: The JHU-ISI Gesture and Skill Assessment Working Set (JIGSAWS) [4] was captured from the *da Vinci* Surgical System as 8 surgeons performed 5 trials of the suturing task. Example actions in this task include grab needle and pass needle through ”tissue”. JIGSAWS contains video data from an endoscopic camera as well as kinematic data (cartesian positions, orientations, velocities, angular velocities, and gripper angles of the manipulators). The kinematic data was recorded at 30 Hz and we downsampled the data by a rate of 5, resulting in 6 frames per second. We also normalized each sample to be of unit norm.

B. Evaluation Criteria

The performance of the subspace clustering methods are assessed with two metrics. First, accuracy reports the percentage of correctly clustered frames. Second, a segmental edit score [7] is built based on the number of insertions/deletions/replacements required to match the order of actions between the experimental result and ground truth. The second metric is useful as it more strongly penalizes over-segmentation errors.

MATLAB implementations for LRR, SSC, and OSC came from the open source Subkit package (https://github.com/sjtrny/SubKit).

C. Results

1) CMU MOCAP: For TSSC-LR, the parameters λ_1, λ_2, and s are empirically set to 0.05, 10 and 12, respectively. For TSSC-LC, the parameters λ_1, λ_2, and s are empirically set to 0.1, 300 and 20, respectively. Table 1 reports the average accuracy and edit score. We observe that both methods greatly improve upon the accuracy of representative methods such as LRR and SSC, and they also outperform ACA. TSSC-LR and TSSC-LC perform similarly on this dataset. Figure 1 visualizes the clustering results of the different subspace clustering methods on an example sequence. The temporal subspace clustering methods correct for the over-segmentation seen in LRR and SSC.

Table 1: Accuracy and Edit Score on CMU MOCAP

Method	Accuracy	Edit Score
LRR [9]	65.08	16.86
SSC [2]	76.65	45.55
ACA [15]	84.50[1]	-
TSSC-LR	**85.28**	**40.40**
TSSC-LC	**86.08**	**37.68**

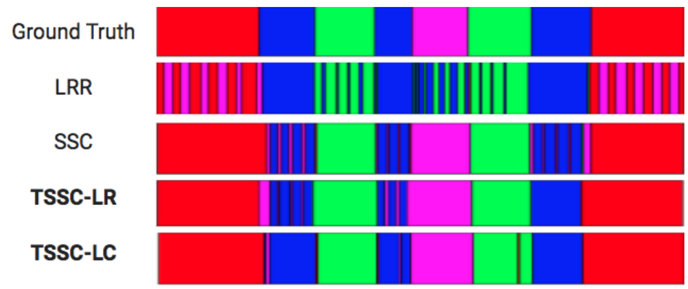

Fig. 1: Visual comparison of subspace clustering methods on an example sequence from CMU MOCAP.

2) CMU MAD: For TSSC-LR, the parameters λ_1, λ_2, and s are empirically set to 0.1, 10 and 8, respectively. For TSSC-LC, the parameters λ_1, λ_2, and s are empirically set to 0.5, 300 and 8, respectively. Table 2 reports the average accuracy and edit score (note that some reported values have an advantage as they use both RGB and skeleton data). Using skeleton data alone, only TAA outperforms TSSC-LR and TSSC-LC. TSSC-LR and TSSC-LC notably outperform the representative LRR and SSC in both accuracy and edit score (visually shown in Figure 2), with TSSC-LC performing the best in both metrics.

Table 2: Accuracy and Edit Score on CMU MAD

Method	Accuracy	Edit Score
LRR [9]	36.93	17.27
SSC [2]	33.64	12.02
OSC [11]	38.867[2]	-
TSC [8]	71.331[3]	-
TAA [3]	69.287[4]	-
TSSC-LR	**65.89**	**55.08**
TSSC-LC	**66.98**	**81.04**

[1] [12], section 11.4

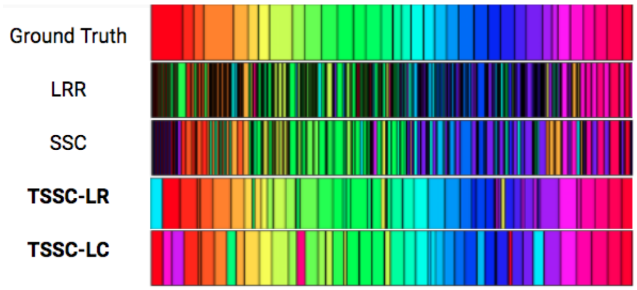

Fig. 2: Visual comparison of subspace clustering methods on an example sequence from CMU MAD.

3) JIGSAWS: For TSSC-LR, the parameters λ_1, λ_2, and s are empirically set to 0.3, 10 and 8, respectively. TSSC-LC has yet to be tested on this dataset. Table 3 reports the average accuracy and edit score, showing that TSSC-LR outperforms all other methods in both metrics. As reflected in both Table 3 and Figure 3, this is a more challenging dataset.

Table 2: Accuracy and Edit Score on JIGSAWS

Method	Accuracy	Edit Score
LRR [9]	45.70	20.77
SSC [2]	44.97	24.18
OSC [11]	46.89	22.62
TSSC-LR	**49.66**	**32.86**

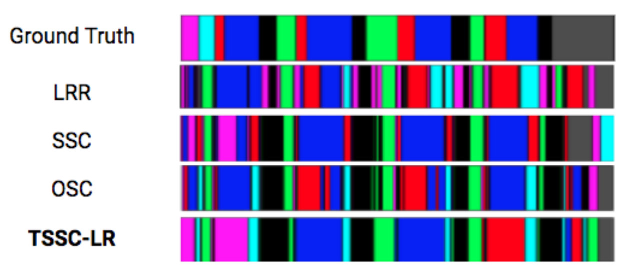

Fig. 3: Visual comparison of subspace clustering methods on an example sequence from JIGSAWS.

V. Conclusions and Future Work

In this work, we propose two temporal subspace clustering methods for unsupervised action segmentation. TSSC-LR applies a temporal Laplacian regularization term that encourages similar codings within a window of samples. TSSC-LC applies a more general temporal regularization term that encourages the coding for a sample to be a linear combination of the codings for its surrounding neighbors. We design efficient ADMM algorithms to solve each problem.

These methods were evaluated against state-of-the-art methods on three action datasets. TSSC-LR and TSSC-LC achieve the highest accuracies on CMU MOCAP, with TSSC-LC performing slightly better. On CMU MAD using only skeleton data, TSSC-LR and TSSC-LC are outperformed only by TAA. For this dataset, TSSC-LC improves upon TSSC-LR in both accuracy and edit score. TSSC-LR outperforms all methods on JIGSAWS, demonstrating the highest accuracy and edit score. These results illustrate the effectiveness of our methods and may suggest advantages of TSSC-LC over TSSC-LR.

More recent methods such as TSC and TSSSC utilize dictionary learning in place of the self-expressive model used in this paper, and future work will look into the effect of this additional step. This entails direct comparisons of our methods to TSC and TSSSC as well as adding dictionary learning to TSSC-LC.

Acknowledgment

This work was supported by NSF award 1447822. I would like to thank the CSMR REU at Johns Hopkins University for accepting me as a participant in their program, in particular Anita Sampath for coordinating everything. Thank you Dr. Vidal for providing me with the opportunity to work in your lab.

References

[1] CMU Motion Capture Database. http://mocap.cs.cmu.edu/.

[2] E. Elhamifar and R. Vidal. Sparse subspace clustering. In *CVPR*, 2009.

[3] E. Fotiadou, Y. Panagakis, and M. Pantic. Temporal Archetypal Analysis for Action Segmentation. In *Automatic Face and Gesture Recognition*, 2017.

[4] Y. Gao et al. JHU-ISI gesture and skill assessment working set (JIGSAWS): A surgical activity dataset for human motion modeling. In *MICCAI*, 2014.

[5] B. Gholami and V. Pavlovic. Probabilistic Temporal Subspace Clustering. In *CVPR*, 2017.

[6] D. Huang et al. Sequential max-margin event detectors. In *European conference on computer vision*, 2014.

[2] [3], RGB + skeleton

[3] [3], RGB + skeleton

[4] [3], skeleton only

[7] C. Lea, R. Vidal, and G.D. Hager. Learning convolutional action primitives for fine-grained action recognition. In *Robotics and Automation (ICRA)*, 2016.

[8] S. Li, K. Li, and Y. Fu. Temporal subspace clustering for human motion segmentation. In *IEEE Conference on Computer Vision*, 2015.

[9] G. Liu, Z. Lin, and Y. Yu. Robust subspace segmentation by low-rank representation. In *ICML*, 2010.

[10] H. Liu, J. Cheng, and F. Weng. Sequential Subspace Clustering via Temporal Smoothness. In *Automatic Face and Gesture Recognition*, 2017.

[11] S. Tierney, J. Gao, and Y. Guo. Subspace clustering for sequential data. In *CVPR*, 2014.

[12] R. Vidal, Y. Ma, and S. Sastry. Generalized principal component analysis (GPCA). In *IEEE transactions on pattern analysis and machine intelligence*, 2005.

[13] S. Wang, B. Tu, C. Xu, and Z. Zhang. Exact subspace clustering in linear time. In *AAAI*, 2014.

[14] F. Wu, Y. Hu, J. Gao, Y. Sun, and B. Yin. Ordered subspace clustering with block-diagonal priors. In *IEEE transactions on cybernetics*, 2016.

[15] F. Zhou, F. D. la Torre, and J. K. Hodgins. Aligned cluster analysis for temporal segmentation of human motion. In *Automatic Face and Gesture Recognition*, 2008.

APPENDICES

A. Research Ethics

Our research did not directly involve human participants so no explicit measures needed to be taken in that regard. All of the motion data we used was gathered for previous work in a responsible manner and openly available to download. Any results or code pulled from other sources is appropriately documented. The parameters for the other methods shown in this paper are optimized for the best performance in order to fairly compare to our methods. The figures in the paper are also chosen to be representative of the data as a whole.

B. Value of the Program

Through my research project, I now know more about the research process and its challenges for more abstract work in computer science. For example, it is hard to compare performance of different algorithms on different sets of data, especially since this more abstract work can at times be hard to interpret. In addition, I had to learn how to effectively communicate these abstract ideas in a variety of formats a brief poster, a more in-depth presentation, and a comprehensive research paper. Those communication skills will be key in expressing my ideas in both future college enrollment and future employment. More specifically, in my work I gained mathematical skills in matrix calculus and optimization techniques that will continue to be relevant in my future study/work in machine learning. Overall, I have become more comfortable reading technical research papers that contain complex functions and equations. Lastly, I found it particularly interesting to gain exposure to a wide range of problems in computer vision by learning about other peoples work in the lab and attending related talks. From this, I have a better idea of the opportunities available to me for research in graduate school.

C. Overview of the Program

The people that I met during the program were all great, that was probably the biggest highlight. My PI Ren was highly intelligent/knowledgeable and always had helpful input. I was impressed by his ability to sit through meetings all day, devoting time to each of his students, and discuss the wide range of topics being studied in his lab. My graduate mentor Efi was also awesome, she was always available for questions and genuinely understood what it meant to be a mentor. I also got along really well with my suitemates in the dorm, it was nice to be in a room of four. Finally, Anita was great in making sure everything was well-organized and well-communicated. Outside of people, I enjoyed the field trips and appreciated the resources we had in preparing for the poster session and presentation.

That being said, I believe the presentation class could be improved in its structure. I think the class could have started a couple weeks later and maybe had each class only last 2 hours rather than 3. It was certainly good to think about these things early, but in some cases it was too early (for example having to present our results section before we really had any final results). I think time could be saved and spent more productively if we split into small groups and only presented within the small group. This would also allow for more detailed feedback from fellow REU participants and then we could send updated outlines to whoever teaches the class so they can more closely look at them to give even further feedback. On a related note, I felt that completing a poster, presentation, and formal paper at the end of the program was difficult. I would have found it more doable to only complete two of the three.