A Condition Number for Joint Optimization of Cycle-Consistent Networks

Leonidas Guibas[1], Qixing Huang[2], and Zhenxiao Liang[2]

[1]Stanford University
[2]The University of Texas at Austin

Abstract

A recent trend in optimizing maps such as dense correspondences between objects or neural networks between pairs of domains is to optimize them jointly. In this context, there is a natural *cycle-consistency* constraint, which regularizes composite maps associated with cycles, i.e., they are forced to be identity maps. However, as there is an exponential number of cycles in a graph, how to sample a subset of cycles becomes critical for efficient and effective enforcement of the cycle-consistency constraint. This paper presents an algorithm that select a subset of weighted cycles to minimize a condition number of the induced joint optimization problem. Experimental results on benchmark datasets justify the effectiveness of our approach for optimizing dense correspondences between 3D shapes and neural networks for predicting dense image flows.

1 Introduction

Maps between sets are important mathematical quantities. Depending on the definition of sets, maps can take different forms. Examples include dense correspondences between image pixels [28, 23], sparse correspondences between feature points [27], vertex correspondences between social or biological networks [13], and rigid transformations between archaeological pieces [16]. In the deep learning era, the concept of maps naturally extends to neural networks between different domains. A fundamental challenge in map computation is that there is only limited information between pairs of objects/domains for map computation, and the resulting maps are often noisy and incorrect, particularly between relevant but dissimilar objects. A recent trend in map computation seeks to address this issue by performing map synchronization, which jointly optimizes maps among a collection of related objects/domains to improve the maps between pairs of objects/domains in isolation [29, 24, 15, 14, 39, 9, 17, 11, 19, 45, 34]. In this context, there is a natural constraint called cycle-consistency, which states that composite maps along cycles should be the identity map. When maps admit matrix representations, state-of-the-art techniques [14, 41, 9, 41, 36, 36, 5, 4] formulate map synchronization as recovering a low-rank matrix, where the pairwise maps computed between pairs of objects are noisy measurements of the blocks of this matrix. This paradigm enjoys tight exact recovery conditions as well as empirical success (e.g.,[14, 41, 9, 41, 36, 36, 5, 4]).

In this paper, we focus on the case where maps between objects/domains do not admit matrix representations (c.f. [45]), which is a popular setting for neural networks between domains. To jointly optimize neural networks in this context, one has to enforce the original cycle-consistency constraint. The technical challenge, though, is that the number of cycles in a graph may be exponential in the number of vertices. In other words, we have to develop a strategy to effectively sample a subset of cycles to enforce the cycle-consistency constraint. The goal for cycle selection is three-fold: *completeness*, *conciseness*, and *stability*. Completeness stands for the fact that enforcing the cycle-consistency constraint on the selected cycles induces the cycle-consistency property among all cycles in the graph. Conciseness means both the size of the cycle-set and the length of each cycle should be small. Stability concerns the convergence behavior when solving the induce joint optimization problem, e.g., joint learning of a network of neural networks. In particular, stability is crucial for

33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.

cycle-set selection as many cycle sets satisfy the first two objectives, yet the numerical behavior of the induced optimization problem turns out drastically different. To date, cycle selection is mostly done manually or uses approaches that only consider the first two objectives. In contrast, we introduce an automatic approach that takes all three objectives into account for cycle selection.

Our cycle selection approach first establishes a stability score based on a condition number of the Hessian matrix of the induced optimization problem. This condition number dictates the local convergence rate as well as the convergence radius of the induced optimization problem. Our approach then combines semidefinite programming and importance sampling to select a small subset of cycles from an over-complete set of cycles to minimize the condition number.

We have evaluated our approach on a variety of settings of optimizing cycle-consistent maps, including dense correspondences across hundreds of shapes and neural networks for predicting dense flows across natural images. Example results show that cycle selection not only improves the convergence rate of the induced optimization problem but also leads to maps with improved map quality.

2 Related Works

Optimizing maps among a collection of objects/domains is a fundamental problem across many different scientific domains. In the following, we review works that focus on various optimization techniques, which are most relevant to the context of this paper.

Cycle-consistency constraint. The foundation for joint map computation is the so-called cycle-consistency constraint [20, 14], which states that the composition of correct maps along cycles should be the identity map. There are two widely used formulations of the cycle-consistency constraint. Low-rank based techniques utilize matrix representations of maps and the equivalence between cycle-consistent maps and the fact that the matrix that stores pair-wise maps in blocks is low-rank and/or semidefinite (c.f.[14]). This equivalence leads to a simple formulation of joint map computation via low-rank matrix recovery [41, 24, 15, 14, 17, 10, 48, 36, 26, 18, 3, 33, 32, 1, 6, 2]. Such techniques enjoy both empirical success and exact recovery conditions. However, a fundamental limitation of such techniques is that there must exist matrix map representations, and such assumptions are not always true, e.g., when neural networks encode maps.

Another category of methods utilizes spanning trees [20, 16]. In a modern context of joint map computation, i.e., recovering accurate maps from maps computed between pairs of objects in isolation, one can seek to recover the correct maps by computing the minimum spanning tree where the induced maps agree with the input maps as much as possible. Most recent combinatorial optimization techniques are based on sampling inconsistent cycles [44, 29, 47]. They formulate map synchronization as removing maps so that each inconsistent cycle contains at least one removed map. However, both techniques are most suited for the task of pruning incorrect maps. They are not suitable for optimizing maps continuously. In contrast, the approach described in this paper combines the strength of both formulations. Precisely, we still formulate map synchronization by minimizing an objective function that combines an observation term and a prior term. The observation term evaluates the quality of each map based on the training data. The difference is in the regularization term, where we directly enforce the consistency of maps along cycles. This approach is suitable for diverse map representations. However, a fundamental challenge is to obtain a concise and effective cycle-basis, which is the main focus of this paper.

Cycle-basis. Cycle-basis is a well studied topic on undirected graphs (c.f.[22]). In the standard setting, a cycle-basis consists of a minimum set of cycles of a graph, where all other cycles in this graph are linear combinations of the cycles in this graph. In this paper, we extend this notion to cycle-consistency bases. The goal is to compute a minimum set of cycles, where enforcing consistency along these cycles induces consistency along all cycles of the input graph. Although cycle-consistency bases are equivalent, enforcing them for map computation exhibits different behavior. The primary goal of this paper is to properly define the condition number of cycle-consistency basis and develop efficient ways to optimize in the space of cycle-consistency bases to minimize the condition number.

3 Cycle-Consistency Bases

In this section, we define map graphs and cycle-consistency bases. In Section 4, we discuss how to optimize cycle-consistency bases for joint map optimization. Note that due to the space constraint, we defer the proofs to the supplemental material.

We first define the notion of a map graph along an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. Since parametric maps (e.g., neural networks) are oriented, we let the edges in $\mathcal{E}$ be oriented. We say $\mathcal{G}$ is undirected if and only if $\forall (i,j) \in \mathcal{E}$, $(j,i) \in \mathcal{E}$.

Definition 1 *We define a map graph $\mathcal{F}$ as an attributed undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where $V = \{v_1, \ldots, v_{|\mathcal{V}|}\}$. Each vertex $v_i \in \mathcal{V}$ is associated with a domain $\mathcal{D}_i$. Each edge $e = (i,j) \in \mathcal{E}$ is associated with a map $f_{ij} : \mathcal{D}_i \rightarrow \mathcal{D}_j$. In this paper, we assume $\mathcal{G}$ is connected. Note that when defining cycle-consistency bases later, we always assume f_{ij} is an isomorphism between $\mathcal{D}_i$ and $\mathcal{D}_j$, and $f_{ji} = f_{ij}^{-1}$.*

To define cycle-consistency bases on $\mathcal{G}$, we introduce composite maps along cycles of $\mathcal{G}$.

Definition 2 *Consider a cycle $c = (i_1, \cdots, i_k i_1)$ along $\mathcal{G}$. We define the composite map along c induced from a map graph $\mathcal{F}$ as*

$$f_c = f_{i_k i_1} \circ \cdots \circ f_{i_1 i_2}. \tag{1}$$

In this paper, we are talking about a cycle. We always assume it has no repeating edges.

Definition 3 *Given a map graph $\mathcal{F}$ associated with a graph $\mathcal{G}$, let $\mathcal{C}$ be a cycle set of $\mathcal{G}$. We say $\mathcal{F}$ is cycle-consistent on $\mathcal{C}$, if*

$$f_c = Id_{\mathcal{D}_{i_1}}, \quad \forall c = (i_1 \cdots i_k i_1) \in \mathcal{C}. \tag{2}$$

Here Id_X refers to the identity mapping from X to itself. Let $\overline{\mathcal{C}}$ collects all cycles of $\mathcal{G}$. We say $\mathcal{F}$ is cycle-consistent, if it is cycle-consistent on $\overline{\mathcal{C}}$.

Remark 1 *Note that due to the bi-directional consistency in Def. 1, (2) is independent of the starting vertex i_1. In fact, it induces*

$$f_{i_l \cdots i_k i_1 \cdots i_l} = Id_{\mathcal{D}_{i_l}}, \quad 1 \leq l \leq k.$$

Since $\overline{\mathcal{C}}$ contains an exponential number of cycles, a natural question is whether we can choose a small subset of cycles $\mathcal{C}$ so that for every map graph $\mathcal{F}$ that is cycle-consistent on $\mathcal{C}$, it induces the cycle-consistency of $\mathcal{F}$. To this end, we need to define the notion of induction:

Definition 4 *Consider a cycle set $\mathcal{C}$ and a cycle $c \notin \mathcal{C}$. We say $\mathcal{C}$* induces *c if there exists an ordered cycle set $c_1, \cdots, c_K \in \mathcal{C}$ and intermediate simple cycles $c^{(k)}, 1 \leq k \leq K$ so that (1) $c^{(1)} = c_1$, (2) $c^{(K)} = c$, and (3) $c^{(k)} = c^{(k-1)} \oplus c_k$, i.e., $c^{(k)}$ is generated by adding new edges in c_k to $c^{(k-1)}$ while removing their common edges.*

An immediate consequence of Def. 4 is the following:

Fact 1 *Given a map graph $\mathcal{F}$, a cycle set $\mathcal{C}$ and another cycle c. If (1) $\mathcal{F}$ is cycle-consistent on $\mathcal{C}$, and (2) $\mathcal{C}$ induces c. Then $\mathcal{F}$ is cycle-consistent on $\{c\}$.*

Remark 2 *It is necessary for Def. 4 to require that $c^{(k)}, 1 \leq k \leq K$ are simple cycles. We provide a counter example in the supplemental material.*

The following proposition shows that Def. 4 is complete.

Proposition 1 *Suppose a map graph $\mathcal{F}$ is cycle-consistent on a cycle set $\mathcal{C}$. If a cycle c can not be induced from $\mathcal{C}$ using the procedure described in Def. 4, then $\mathcal{F}$ may not be cycle-consistent on $\{c\}$.*

Now we define the notion of cycle-consistency basis:

Definition 5 *We say a cycle set $\mathcal{C}$ is a cycle-consistency basis if it induces all other cycles of $\overline{\mathcal{C}}$.*

The following proposition characterizes the minimum size of cycle-consistency bases and a procedure for constructing a category of cycle-consistency bases with minimum size.

Proposition 2 *The minimum size of a cycle-consistency basis on a connected graph $\mathcal{G}$ is $|\mathcal{E}| - |\mathcal{V}| + 1$. Moreover, we can construct a minimal cycle-consistency basis from a spanning tree $\mathcal{T} \subset \mathcal{E}$ of $\mathcal{G}$, i.e., by creating a cycle c_e for each edge pair $e = (i,j) \in \mathcal{E} \setminus \mathcal{T}$, where $c_e = (i,j) \sim p_{ji}$, where p_{ji} is the unique path from j to i on $\mathcal{T}$.*

Remark 3 *A difference between cycle-consistency bases on undirected graphs and path-invariance bases on directed graphs is that the minimum size of cycle-consistency bases is known and is upper bounded by the number of edges. In contrast, computing the minimum size of path-invariance bases of a given directed graph remains an open problem (c.f.[45]).*

Connections to cycle bases [22]. When talking about cycles of an undirected graph, there exists a related notion of cycle bases [22]. The difference between cycle-consistency bases and cycle bases lies in the induction procedure. Specifically, cycle bases utilize a vector that collects edge indicators, i.e., each edge has an orientation, and each cycle corresponds to a sparse vector whose elements are in $\{1, 0\}$, which represent edges in c and the other edges. The induction procedure takes the form of linear combinations of indicator vectors. Depending on the weights for linear combination, cycle bases fall into zero-one cycle bases, integral cycle bases and general cycle-bases, which correspond to $\{-1, 0, 1\}$, integer and real weights, respectively. Please refer to [22] for more details.

It is easy to see that one can use linear combinations of vectors with binary weights to encode the induction procedure of cycle-consistency bases. On the other hand, the reverse is not true.

Proposition 3 *A cycle-consistency basis is a zero-one cycle basis. A zero-one cycle basis may not be a cycle-consistency basis.*

Remark 4 *Unlike the situation that we can verify a zero-one cycle basis by checking independence within polynomial time (c.f. [22]), we conjecture that verifying whether a cycle set forms a cycle-consistency basis is NP-hard. In light of this, when optimizing a cycle set we enforce its cycle-consistency property by adding cycles to a minimum cycle-consistency basis.*

4 Cycle Consistency Basis Optimization for Joint Map Optimization

In this section, we present an approach that optimizes a cycle-consistency basis for a given joint map optimization task. Specifically, we assume each edge $(i, j) \in \mathcal{E}$ is associated with a parametric map $f_{ij}^{\theta_{ij}} : \mathcal{D}_i \rightarrow \mathcal{D}_j$, where θ_{ij} denotes the parameters of this map. We also assume we have a subset of edges $\mathcal{E}_0 \subseteq \mathcal{E}$. For each edge $e \in \mathcal{E}_0$, we have a loss term denoted as $l_{ij}(\theta_{ij})$ (e.g., we may only have training data among a subset of edges). We assume $\mathcal{G}_0 = (\mathcal{V}, \mathcal{E}_0)$ forms a connected graph. Otherwise, we have insufficient constraints to determine all parametric maps.

Our main idea is to pre-compute a super set of cycles $\mathcal{C}_{\text{sup}}$ (See Section 4.2 for details) and formulate cycle-consistency basis optimization as determining a cycle set $\mathcal{C}$, where $\mathcal{C}_{\min} \subseteq \mathcal{C} \subseteq \mathcal{C}_{\text{sup}}$, and a weight $w_c > 0$ of each cycle $c \in \mathcal{C}$, for solving the following joint map optimization problem:

$$\sum_{(i,j)\in\mathcal{E}_0} l_{ij}(\theta_{ij}) + \sum_{c=(i_1\cdots i_k i_1)\in\mathcal{C}} w_c l_{i_1}(f_c^{\Theta}, Id_{\mathcal{D}_{i_1}}) \tag{3}$$

Here $f_c^{\Theta} = f_{i_k i_1}^{\theta_{i_k i_1}} \circ \cdots \circ f_{i_1 i_2}^{\theta}$ is the composite map along c and $l_i(\cdot, \cdot)$ denotes a loss-term for comparing self-maps on $\mathcal{D}_i$. For example, $l_i(f, f') := E_{\boldsymbol{x}\sim p} d^2_{\mathcal{D}_i}(f(\boldsymbol{x}), f'(\boldsymbol{x}))$, where $d_{\mathcal{D}_i}(\cdot, \cdot)$ is a distance metric of $\mathcal{D}_i$, and where p is an empirical distribution on $\mathcal{D}_i$. Note that (3) essentially enforces the cycle-consistency constraint along $\mathcal{C}$.

In the reminder of this section, Section 4.1 introduces a condition number of (3); Section 4.2 describes how to minimize this condition number by selecting and weighting cycles.

4.1 A Condition Number for Cycle-Consistency Map Optimization

We begin with a simple setting of translation synchronization [18], where pairwise parametric maps are given by translations. We then discuss how to generalize this definition to neural networks. Note that for the particular task of translation synchronization, there exist many other formulations, e.g., [21, 18]. Our goal here is simply to motivate the definition of the condition number. Specifically, consider a pre-computed translation $t_{ij}^0 \in \mathbb{R}$ for each edge $e = (i, j) \in \mathcal{E}_0$. Our goal is to recover translations $t_{ij}, (i, j) \in \mathcal{E}$ by solving the following quadratic minimization problem:

$$\min_{\{t_{ij},(i,j)\in\mathcal{E}\}} \sum_{(i,j)\in\mathcal{E}_0} (t_{ij} - t_{ij}^0)^2 + \sum_{c=(i_1\cdots i_k i_1)\in\mathcal{C}} w_c(\sum t_{i_l i_{i+1}})^2. \tag{4}$$

where we set $l_{k+1} := l_1$. Given an ordering of the edge set $\mathcal{E}$, let $\boldsymbol{v}_e \in \{0, 1\}^{|\mathcal{E}|}$ be the indicator vector for edge e. With $\boldsymbol{v}_c = \sum_{l=1}^{k} \boldsymbol{v}_{(i_l, i_{l+1})}$ we denote the indicator vector for cycle $c = (i_1 \cdots i_k i_1)$. Let

$\boldsymbol{t}_0 = \sum_{(i,j)\in\mathcal{E}^0} \boldsymbol{v}_{(i,j)} t_{ij}^0$ and $\boldsymbol{t} = \sum_{(i,j)\in\mathcal{E}} \boldsymbol{v}_{(i,j)} t_{ij}$. We can rewrite (4) in the matrix form as

$$\min_{\boldsymbol{t}} \ \boldsymbol{t}^T H \boldsymbol{t} - 2\boldsymbol{t}^T \boldsymbol{t}^0 + \|\boldsymbol{t}^0\|^2, \qquad H := \sum_{e\in\mathcal{E}^0} \boldsymbol{v}_e \boldsymbol{v}_e^T + \sum_{c\in\mathcal{C}} w_c \boldsymbol{v}_c \boldsymbol{v}_c^T \tag{5}$$

When solving (5) using gradient-based techniques (which is the case for neural networks), their convergence rates are generally relevant to the condition number $\kappa(H) := \lambda_{\max}(H)/\lambda_{\min}(H)$. For example, steepest descent with exact line search admits a linear convergence rate of $(\kappa(H)-1)/(\kappa(H)+1)$, and this argument also applies to the local convergence behavior of non-linear objective functions (c.f. Sec.1.3 of [7]). In addition, the deviation between the optimal solution $\boldsymbol{t}^\star$ and the ground truth solution $\boldsymbol{t}^{gt}$ is $\|\boldsymbol{t}^\star - \boldsymbol{t}^{gt}\| = \|H^{-1}\boldsymbol{e}\| \leq \frac{1}{\lambda_{\min}(H)}\|\boldsymbol{e}\|$, where $\boldsymbol{e} = \boldsymbol{t}_0 - \boldsymbol{t}_0^{gt}$ is the vector that encodes the error in the input. Minimizing the $\kappa(H)$ usually leads to an increased value of $\lambda_{\min}$, which reduces the error in the output. Thus, we proceed with the following definition:

Definition 6 *We define the condition number of (3) as the condition number $\kappa(H)$.*

Def. 6 also generalizes to other parametric maps:

Theorem 4.1 ***(Informal)*** *Under mild conditions, the condition number of the Hessian matrix at a local optimal of (3) is $O(s\kappa(H))$, where s depends on quantities of individual $f_{ij}^{\theta_{ij}}$.*

Another factor related to an efficient optimization of (3) is the size of $\mathcal{C}$. As there is some fixed overhead for implementing the constraint associated with each cycle, we favor that the size of $\mathcal{C}$ is small. In summary, our goal for cycle-consistency basis optimization is to reduce the condition number of H while simultaneously to minimize the size of the resulting cycle-consistency basis.

4.2 Algorithm for Cycle-Consistency Basis Construction

Our approach for cycle-consistency basis optimization proceeds in three steps. The first step generates the super cycle set $\mathcal{C}_{\text{sup}}$. The second step solves a semi-definite program to optimize weights $w_c, c \in \mathcal{C}_{\text{sup}}$ to minimize the condition number of H. The final setup controls the size of the resulting cycle-consistency basis via importance sampling.

$\mathcal{C}_{\text{sup}}$ generation. We construct $\mathcal{C}_{\text{sup}}$ by computing the breadth-first spanning tree $\mathcal{T}(v_i)$ rooted at each vertex $v_i \in \mathcal{V}$. For each spanning-tree $\mathcal{T}(v_i)$, we use the procedure described in Prop. 2 to construct a minimum cycle-consistency basis $\mathcal{C}(v_i)$. We define $\mathcal{C}_{\text{sup}} := \cup_{v_i\in\mathcal{V}}\mathcal{C}(v_i)$. We set $\mathcal{C}_{\min}$ as the cycle-consistency basis with minimum depth.

The resulting $\mathcal{C}_{\text{sup}}$ has two desired properties. First, the cycles in $\mathcal{C}_{\text{sup}}$ are kept as short as possible. For example, if $\mathcal{G}$ is a clique, then $\mathcal{C}_{\text{sup}}$ only contains the desired 3-cycles. Second, if $\mathcal{G}$ is sparse, then $\mathcal{C}_{\text{sup}}$ contains a mixture of short and long cycles. These long cycles can address the issue of accumulated errors if we only enforce the cycle-consistency constraint along short cycles.

Weight optimization. As the condition number of H is minimized if it is a scalar multiple of the identity matrix, we formulate the following semidefinite program for optimizing cycle weights:

$$\begin{aligned}
\min_{w_c \geq 0, s_1, s_2} \quad & \alpha s_2 - s_1 \\
\text{subject to} \quad & (C1): s_1 I \preceq \sum_{e\in\mathcal{E}^0} \boldsymbol{v}_e \boldsymbol{v}_e^T + \sum_{c\in\mathcal{C}_{\text{sup}}} w_c \boldsymbol{v}_c \boldsymbol{v}_c^T \preceq s_2 I \\
& (C2): \sum_{c\in\mathcal{C}_{\text{sup}}} |\boldsymbol{v}_c|^2 w_c = \lambda, \qquad (C3): w_c \geq \delta, \forall c \in \mathcal{C}_{\min}
\end{aligned} \tag{6}$$

Here λ characterizes the trade-off between the loss terms and the regularization terms and α is a super parameter. To figure out the motivation of introducing such a parameter, recall that the definition of condition number is $\frac{\lambda_{\max}}{\lambda_{\min}}$ where $\lambda_{\max}, \lambda_{\min}$ are maximal and minimal eigenvalues of the weighted matrix sum respectively. But optimizing the ratio of two eigenvalues is non-convex in terms of variables $\{w_c\}$, thus we in turn try to reduce the gap between $\lambda_{\max}$ and $\lambda_{\min}$. Different α's would provide different outcomes of eigen-ratio. Hence we test over a set of α's and look for the one with optimal eigen-ratio. In fact it is easy to see that as $\alpha^* = \max \frac{\lambda_{\min}}{\lambda_{\max}}$ the objective function $\alpha^* s_2 - s_1$ would have minimum 0 at the point which minimizes $\frac{\lambda_{\max}}{\lambda_{\min}}$ as well. In most cases where the optimal ratio is in the order of magnitude near 1, so that the setting $\alpha = 1$ would provide a good enough approximation.

Moreover, (C3) ensures that edges in the minimum cycle consistency basis $\mathcal{C}_{\min}$ are also selected. In our experiments, we set $\lambda = 8|\mathcal{E}_0|/|\mathcal{E}|$ and $\delta = \lambda/(8|\mathcal{E}|)$. We solve (6) using alternating direction method of multipliers (See the supplemental material for details)

As the cycle indicators, which are sparse vectors, tend to be orthogonal to each other. The solution to (6) typically turns a matrix with small condition number.

Theorem 4.2 *(**Informal**) Let $s_1^\star$ and $s_2^\star$ be the optimal solution to (6). Then $\max_c w_c \leq s_2^\star$, and both $s_2^\star$ and $s_2^\star - s_1^\star$ are upper bounded by quantities that are related to an uniform score of the spherical Voronoi diagram of copies of $\{\boldsymbol{v}_e, e \in \mathcal{E}_0\} \cup \{\frac{\boldsymbol{v}_c}{\|\boldsymbol{v}_c\|}, c \in \mathcal{C}_{\text{sup}}\}$.*

Importance sampling. Although the semidefinite program described in (6) controls the condition number of H, it does not control the size of the cycle sets with positive weights. Thus we seek to select a subset of cycles $\mathcal{C}_{\text{sample}} \subset \mathcal{C}_{\text{active}} := \mathcal{C}_{\text{sup}} \setminus \mathcal{C}_{\min}$ and compute new weights $\overline{w}_c, c \in \mathcal{C}_{\text{sample}}$, so that

$$\sum_{c \in \mathcal{C}_{\text{sample}}} \overline{w}_c \boldsymbol{v}_c \boldsymbol{v}_c^T \approx \sum_{c \in \mathcal{C}_{\text{active}}} \overline{w}_c \boldsymbol{v}_c \boldsymbol{v}_c^T. \tag{7}$$

We achieve this goal through sampling. Specifically, consider a desired size L for $\mathcal{C}_{\text{sample}}$. Let $w_{\max} = \max\limits_{c \in \mathcal{C}_{\text{active}}} w_c$. Choose the maximum $\alpha \leq 1$ so that $L \leq \sum\limits_{c \in \mathcal{C}_{\text{active}}} (w_c/w_{\max})^\alpha$. We define an independent random variable x_c and a modified weight $\overline{w}_c$ for each cycle $c \in \mathcal{C}_{\text{active}}$:

$$x_c = \left\{ \begin{array}{cc} 1 & \text{with probability } p_c \\ 0 & \text{with probability } 1 - p_c \end{array} \right. \qquad \overline{w}_c = w_c/p_c, \quad p_c := L \cdot w_c^\alpha / \sum_{c \in \mathcal{C}_{\text{active}}} w_c^\alpha. \tag{8}$$

To generate $\mathcal{C}_{\text{sample}}$, we simply sample $\mathcal{C}_{\text{active}}$ according x_c. It is easy to check that

$$E[|\mathcal{C}_{\text{sample}}|] = L, \qquad E[\sum_{c \in \mathcal{C}_{\text{sample}}} \overline{w}_c \boldsymbol{v}_c \boldsymbol{v}_c^T] = \sum_{c \in \mathcal{C}_{\text{active}}} w_c \boldsymbol{v}_c \boldsymbol{v}_c^T.$$

In the following, we provide concentration inequalities on both quantities:

Theorem 4.3 *Given the sampling procedure described in (8) with standard deviation*

$$\sigma_1 := \sqrt{\sum_c p_c(1 - p_c)}, \quad \sigma_2 := \sqrt{\sum_c p_c(1 - p_c)\overline{w}_c^2}$$

and condition

$$\sigma_1^2 = \Omega(1), \quad \sigma_2^2 = \Omega\big((\max_c \overline{w}_c)^2\big),$$

we have with probability at least $1 - O(1/poly(n))$,

$$||\mathcal{C}_{\text{sample}}| - L| \leq O(\log n)\sigma_1 \tag{9}$$

$$\| \sum_{c \in \mathcal{C}_{\text{sample}}} \overline{w}_c \boldsymbol{v}_c \boldsymbol{v}_c^T - \sum_{c \in \mathcal{C}_{\text{active}}} w_c \boldsymbol{v}_c \boldsymbol{v}_c^T \| \leq O(\log n)\sigma_2 \tag{10}$$

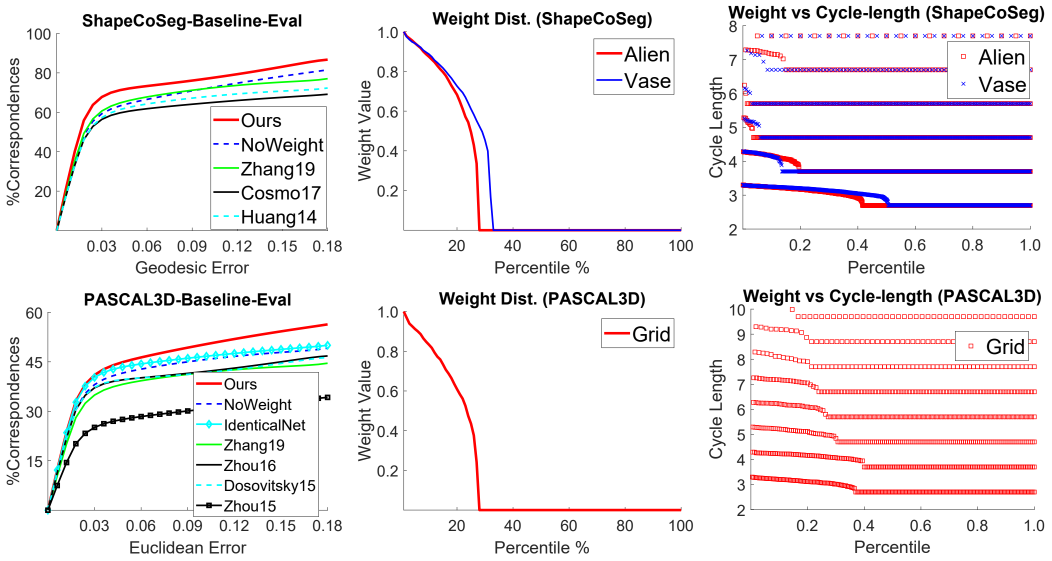
Note that (10), which utilizes $\text{rank}(\boldsymbol{v}_c \boldsymbol{v}_c^T) = 1$, is sharper than general concentration bounds [38]. To be more precise, the known bound contains an extra multiplicative term $O(\log d)$ where d is the dimension of matrices involved. In our case $d = O(|\mathcal{E}|)$.

5 Experimental Evaluation

In this section, we present an experimental evaluation of our joint map optimization approach in two application settings: consistent shape maps (c.f. [24, 14, 17, 11]) and dense image correspondence using neural networks (c.f. [46, 45]).

5.1 Consistent Shape Correspondences

Experimental setup. Similar to [39, 40, 17], we encode the map from one shape S_i and another shape S_j as a function map $X_{ij} : \mathcal{F}(S_i) \rightarrow \mathcal{F}(S_j)$ [31]. The same as [17], we choose each functional space $\mathcal{F}(S_i)$ as the linear space spanned by smallest $m = 30$ eigenvectors of the co-tangent mesh



(a) (b) (c)

Figure 1: The top and bottom rows show quantitative evaluations on the ShapeCoSeg [42] and the PASCAL3D [43], respectively. (a) Cumulative distributions of geodesic errors (ShapeCoSeg) and Euclidean errors (PASCAL3D) of predicted feature correspondences of our approach and baseline approaches. (b) Distributions of cycle weights. (c) Distributions of cycle weights per cycle-length

Laplacian. A functional map $X_{ij} \in \mathbb{R}^{m\times m}$ essentially encodes a linear map from $\mathcal{F}(S_i)$ to $\mathcal{F}(S_j)$. We refer to [31] for more details about functional maps.

The evaluation considers two shape collections from ShapeCoSeg [42]: Alien (200 shapes) and Vase (300 shapes). For each shape collection, we construct $\mathcal{G}$ by connecting every shape with $k = 25$ randomly chosen shapes. The edge set $\mathcal{E}_0$ collects for each shape the $k_0 = 9$ closest shapes among the neighbors specified by $\mathcal{E}$ via the GPS descriptor [35]. For each edge $e = (i,j) \in \mathcal{E}_0$, we compute dense correspondences from S_i to S_j using Blended Intrinsic Maps [25]. We then convert this map into the corresponding functional map X_{ij}^0 using [31].

For joint map computation, we obtain improved functional maps $X_{ij}, (i,j) \in \mathcal{E}$ by minimizing

$$\frac{1}{|\mathcal{E}|}\sum_{(i,j)\in\mathcal{E}}\|X_{ij}-X_{ij}^{in}\|_{\mathcal{F}}^2+\lambda\sum_{c=(i_1\cdots i_{|c|}i_1)\in\mathcal{C}} w_c\|X_{i_{|c|}i_1}\cdots X_{i_1 i_2}-I_m\|_{\mathcal{F}}^2 \qquad (11)$$

For numerical optimization, we start from the identity map $X_{ij} = I_m, (i,j) \in \mathcal{E}$ and apply steepest descent with exact line search [30]. We run 3000 iteration on each dataset. After optimization, we generate maps between all pairs of shapes by composing maps along shortest paths on $\mathcal{G}$.

Analysis of results. To evaluate the quality of shape maps, we report the cumulative distribution (or CD) of normalized geodesic error e_{geo} of predicted feature correspondences (c.f [25]. We compare our approach with three state-of-the-art joint shape matching approaches: Huang14 [17], Cosmo17 [11] and Zhang19 [45]. As shown in Figure 1(a), our approach leads to noticeable performance gains from Huang14 and Cosmo17 that leverage low-rank relaxations of the cycle-consistency constraint (c.f. [14]). An explanation is that when the observations are sparse, these low-rank relaxations become loose. In contrast, enforcing the cycle-consistency constraint exactly offers strong regularization. Our approach also outperforms [45] (i.e., by 4.9% when $e_{geo} = 0.1$), which employs a relevant path-invariance constraint by treating $\mathcal{G}$ as a directed graph. As we will discuss immediately, the improvement comes from weighted cycles.

Analysis of cycle weighting. As show in Figure 1(a), weighting the cycles has a significant impact on the quality of the optimized maps. When solving (11) with equal weight $\lambda/|\mathcal{C}_{\text{sup}}|$, the CD percentage drops by 5.2% (when $e_{geo} = 0.15$). Note that using more iterations does not close the gap, as there is still a 2.4% difference even after 30000 iterations. This gap also justifies the argument that reducing the condition number alleviates the amplified errors in the solution of (11) that are caused by X_{ij}^0.

Figure 1(b) plots the distribution of cycle weights returned by the SDP formulation. We can see that the SDP formulation leads to sparse and relatively uniform cycle weights, indicating the effectiveness of our approach for selecting important cycles. Moreover, most cycles with positive weights are short (See Figure 1(c)). This behavior coincides with the intuition that $\mathcal{G}$ is a dense graph, and utilizing short cycles for optimizing (11) is sufficient.

5.2 Consistent Neural Networks among Multiple Domains

Experimental setup. In this setting, we consider the task of predicting dense flows between image objects using a neural network (c.f. [12, 46]. We perform experimental evaluation on 12 rigid categories from PASCAL3D [43]. For each category, we construct a map graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where each vertex $v \in \mathcal{V}$ represents image objects viewed from similar camera poses, and where each edge represents a dense flow neural network between some adjacent vertex pairs (to be discussed shortly). In our experiments, we generate $\mathcal{V}$ by first picking the dominant view of each category [43] and then sampling a grid of 5×5 camera poses. This grid is centered at the dominant view, its two axes align with the latitude and longitude, and its spacing is $22.5°$. Similar to ([46]), we consider both real images from PASCAL3D [43] and synthetic images from ShapeNet [8] for training. For each training image, we allocate it to four closest vertices in terms of camera poses. We connect an edge between two vertices if the angular distance between their camera poses is less than $35°$. All edges use the same network architecture [46]. However, we allow them to take different weights to learn specific features associated with each camera pose pair. Moreover, we set $\mathcal{E} = \mathcal{E}_0$.

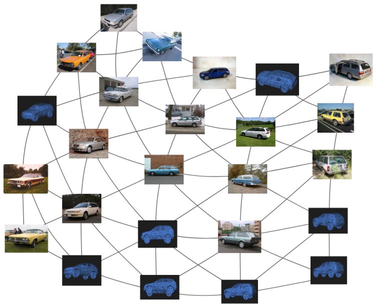


Figure 2: Map graph for image flow.

We apply (3) to jointly learn the neural networks associated with each edge. Inspired by [12], we use synthetic images to define the loss term associated with each edge. In contrast, the cycle-consistency constraint is enforced on real images. To initialize the neural networks, we first pre-train a single network using synthetic images. We then pass the pre-trained weights for all networks. The same as joint shape matching, we generate dense image flows between all pairs of images by composing neural networks along shortest paths on $\mathcal{G}$.

During testing time, we use [37] to predict a camera pose for each image and associate it with the closest vertex of $\mathcal{G}$. Given two images, we extract the corresponding network to predict dense correspondences.

Analysis of results. For experimental evaluation, we report cumulative distributions of normalized Euclidean error e_{euc} (with respect to the $\max$(width, height)) of predicted feature correspondences (c.f [47]. We compare our approach with four state-of-the-art data-driven dense image flow approaches: Zhou15 [47], Dosovitskiy15 [12],Zhou16 [46], Zhang19 [45]. As shown in Figure 1(a), our approach leads to noticeable performance gains from Zhou15, which only utilizes real images. Likewise, our approach also significantly outperforms Dosovitskiy15 trained on synthetic data alone (i.e., by 8.1% when $e_{euc} = 0.1$. This encouraging result shows the potential of leveraging the self-supervision constraint on real images. In addition, our approach is also superior to [46] and [45]. Such improvements are attributed to allowing network parameters to vary across different edges — enforcing identical network parameters results in a 3.2% drop in the CD percentage.

Analysis of cycle weighting. Similar to the case of joint shape matching, using uniform weights to solve (3) leads to a 4.3% drop in the CD percentage, which again shows the advantages of weighing the cycles for both the convergence behavior and the robustness of the solution. Moreover, the distribution of cycle weights is similar to that of joint shape matching (See Figure 1(b)), where the solution of SDP returns sparse and uniform cycle weights.

A unique characteristic of this application is that with a relatively sparse graph, the selected cycles contain a few long cycles (See Figure 1(c)). One explanation is that if all selected cycles are short, then the composite networks along long cycles may suffer from accumulated errors. As a consequence, the composite networks between non-adjacent vertices may drift.

Acknowledgement. Qixing Huang would like to acknowledge support from NSF DMS-1700234, a Gift from Snap Research, and a hardware Donation from NVIDIA. Leonidas Guibas would like to acknowledge NSF grant DMS-1546206, a grant from the Stanford-Toyota AI center, and a Vannevar Bush Faculty Fellowship.

References

[1] Federica Arrigoni, Andrea Fusiello, and Beatrice Rossi. Camera motion from group synchronization. In *3D Vision (3DV), 2016 Fourth International Conference on*, pages 546–555. IEEE, 2016.

[2] Federica Arrigoni, Luca Magri, Beatrice Rossi, Pasqualina Fragneto, and Andrea Fusiello. Robust absolute rotation estimation via low-rank and sparse matrix decomposition. In *3D Vision (3DV), 2014 2nd International Conference on*, volume 1, pages 491–498. IEEE, 2014.

[3] Federica Arrigoni, Beatrice Rossi, and Andrea Fusiello. Spectral synchronization of multiple views in se (3). *SIAM Journal on Imaging Sciences*, 9(4):1963–1990, 2016.

[4] Chandrajit Bajaj, Tingran Gao, Zihang He, Qixing Huang, and Zhenxiao Liang. SMAC: Simultaneous mapping and clustering using spectral decompositions. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 324–333, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018. PMLR.

[5] Afonso S. Bandeira, Nicolas Boumal, and Amit Singer. Tightness of the maximum likelihood semidefinite relaxation for angular synchronization. *Math. Program.*, 163(1-2):145–167, May 2017.

[6] Florian Bernard, Johan Thunberg, Peter Gemmar, Frank Hertel, Andreas Husch, and Jorge Goncalves. A solution for multi-alignment by transformation synchronisation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2161–2169, 2015.

[7] D.P. Bertsekas. *Nonlinear Programming*. Athena Scientific, 1999.

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

[9] Yuxin Chen, Leonidas J. Guibas, and Qi-Xing Huang. Near-optimal joint object matching via convex relaxation. In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, pages 100–108, Beijing, China, 2014. JMLR, Inc.

[10] Yuxin Chen, Leonidas J. Guibas, and Qi-Xing Huang. Near-optimal joint object matching via convex relaxation. In *ICML*, pages 100–108, 2014.

[11] Luca Cosmo, Emanuele Rodolà, Andrea Albarelli, Facundo Mémoli, and Daniel Cremers. Consistent partial matching of shape collections via sparse modeling. *Comput. Graph. Forum*, 36(1):209–221, 2017.

[12] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Häusser, Caner Hazirbas, Vladimir Golkov, Patrick van der Smagt, Daniel Cremers, and Thomas Brox. Flownet: Learning optical flow with convolutional networks. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 2758–2766, 2015.

[13] Somaye Hashemifar, Qixing Huang, and Jinbo Xu. Joint alignment of multiple protein-protein interaction networks via convex optimization. *Journal of Computational Biology*, 23(11):903–911, 2016.

[14] Qi-Xing Huang and Leonidas Guibas. Consistent shape maps via semidefinite programming. In *Proceedings of the Eleventh Eurographics/ACMSIGGRAPH Symposium on Geometry Processing*, SGP '13, pages 177–186, Aire-la-Ville, Switzerland, Switzerland, 2013. Eurographics Association.

[15] Qi-Xing Huang, Guo-Xin Zhang, Lin Gao, Shi-Min Hu, Adrian Butscher, and Leonidas Guibas. An optimization approach for extracting and encoding consistent maps in a shape collection. *ACM Trans. Graph.*, 31(6):167:1–167:11, November 2012.

[16] Qixing Huang, Simon Flöry, Natasha Gelfand, Michael Hofer, and Helmut Pottmann. Reassembling fractured objects by geometric matching. *ACM Trans. Graph.*, 25(3):569–578, July 2006.

[17] Qixing Huang, Fan Wang, and Leonidas Guibas. Functional map networks for analyzing and exploring large shape collections. *ACM Transactions on Graphics*, 33(4):36:1–36:11, July 2014.

[18] Xiangru Huang, Zhenxiao Liang, Chandrajit Bajaj, and Qixing Huang. Translation synchronization via truncated least squares. In *NIPS*, 2017.

[19] Xiangru Huang, Zhenxiao Liang, Xiaowei Zhou, Yao Xie, Leonidas J. Guibas, and Qixing Huang. Learning transformation synchronization. *CoRR*, abs/1901.09458, 2019.

[20] Daniel Huber. *Automatic Three-dimensional Modeling from Reality*. PhD thesis, Carnegie Mellon University, Pittsburgh, PA, December 2002.

[21] Xiaoye Jiang, Lek-Heng Lim, Yuan Yao, and Yinyu Ye. Statistical ranking and combinatorial hodge theory. *Math. Program.*, 127(1):203–244, March 2011.

[22] Telikepalli Kavitha, Christian Liebchen, Kurt Mehlhorn, Dimitrios Michail, Romeo Rizzi, Torsten Ueckerdt, and Katharina A. Zweig. Survey: Cycle bases in graphs characterization, algorithms, complexity, and applications. *Comput. Sci. Rev.*, 3(4):199–243, November 2009.

[23] Jaechul Kim, Ce Liu, Fei Sha, and Kristen Grauman. Deformable spatial pyramid matching for fast dense correspondences. In *CVPR*, pages 2307–2314. IEEE Computer Society, 2013.

[24] Vladimir Kim, Wilmot Li, Niloy Mitra, Stephen DiVerdi, and Thomas Funkhouser. Exploring collections of 3d models using fuzzy correspondences. *ACM Trans. Graph.*, 31(4):54:1–54:11, July 2012.

[25] Vladimir G. Kim, Yaron Lipman, and Thomas Funkhouser. Blended intrinsic maps. *ACM Trans. Graph.*, 30(4):79:1–79:12, July 2011.

[26] Spyridon Leonardos, Xiaowei Zhou, and Kostas Daniilidis. Distributed consistent data association via permutation synchronization. In *ICRA*, pages 2645–2652. IEEE, 2017.

[27] Marius Leordeanu and Martial Hebert. A spectral technique for correspondence problems using pairwise constraints. In *Proceedings of the Tenth IEEE International Conference on Computer Vision - Volume 2*, ICCV '05, pages 1482–1489, Washington, DC, USA, 2005. IEEE Computer Society.

[28] Ce Liu, Jenny Yuen, and Antonio Torralba. Sift flow: Dense correspondence across scenes and its applications. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(5):978–994, May 2011.

[29] Andy Nguyen, Mirela Ben-Chen, Katarzyna Welnicka, Yinyu Ye, and Leonidas Guibas. An Optimization Approach to Improving Collections of Shape Maps. *Computer Graphics Forum*, 30:1481–1491, 2011.

[30] Jorge Nocedal and Stephen J. Wright. *Numerical optimization*. Springer series in operations research and financial engineering. Springer, New York, NY, 2. ed. edition, 2006.

[31] Maks Ovsjanikov, Mirela Ben-Chen, Justin Solomon, Adrian Butscher, and Leonidas J. Guibas. Functional maps: a flexible representation of maps between shapes. *ACM Trans. Graph.*, 31(4):30:1–30:11, 2012.

[32] Deepti Pachauri, Risi Kondor, Gautam Sargur, and Vikas Singh. Permutation diffusion maps (PDM) with application to the image association problem in computer vision. In *NIPS*, pages 541–549, 2014.

[33] Deepti Pachauri, Risi Kondor, and Vikas Singh. Solving the multi-way matching problem by permutation synchronization. In *NIPS*, pages 1860–1868, 2013.

[34] David M. Rosen, Luca Carlone, Afonso S. Bandeira, and John J. Leonard. Se-sync: A certifiably correct algorithm for synchronization over the special euclidean group. *I. J. Robotics Res.*, 38(2-3), 2019.

[35] Raif M. Rustamov. Laplace-beltrami eigenfunctions for deformation invariant shape representation. In *Proceedings of the Fifth Eurographics Symposium on Geometry Processing*, SGP '07, pages 225–233, Aire-la-Ville, Switzerland, Switzerland, 2007. Eurographics Association.

[36] Yanyao Shen, Qixing Huang, Nati Srebro, and Sujay Sanghavi. Normalized spectral map synchronization. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 4925–4933. Curran Associates, Inc., Barcelona, Spain, 2016.

[37] Hao Su, Charles Ruizhongtai Qi, Yangyan Li, and Leonidas J. Guibas. Render for CNN: viewpoint estimation in images using cnns trained with rendered 3d model views. In *ICCV*, pages 2686–2694. IEEE Computer Society, 2015.

[38] Joel A. Tropp. An introduction to matrix concentration inequalities. *Found. Trends Mach. Learn.*, 8(1-2):1–230, May 2015.

[39] Fan Wang, Qixing Huang, and Leonidas J. Guibas. Image co-segmentation via consistent functional maps. In *Proceedings of the 2013 IEEE International Conference on Computer Vision*, ICCV '13, pages 849–856, Washington, DC, USA, 2013. IEEE Computer Society.

[40] Fan Wang, Qixing Huang, Maks Ovsjanikov, and Leonidas J. Guibas. Unsupervised multi-class joint image segmentation. In *CVPR*, pages 3142–3149. IEEE Computer Society, 2014.

[41] Lanhui Wang and Amit Singer. Exact and stable recovery of rotations for robust synchronization. *Information and Inference: A Journal of the IMA*, 2:145–193, December 2013.

[42] Yunhai Wang, Shmulik Asafi, Oliver van Kaick, Hao Zhang, Daniel Cohen-Or, and Baoquan Chen. Active co-analysis of a set of shapes. *ACM Trans. Graph.*, 31(6):165:1–165:10, November 2012.

[43] Yu Xiang, Roozbeh Mottaghi, and Silvio Savarese. Beyond PASCAL: A benchmark for 3d object detection in the wild. In *WACV*, pages 75–82. IEEE Computer Society, 2014.

[44] Christopher Zach, Manfred Klopschitz, and Marc Pollefeys. Disambiguating visual relations using loop constraints. In *CVPR*, pages 1426–1433. IEEE Computer Society, 2010.

[45] Zaiwei Zhang, Zhenxiao Liang, Lemeng Wu, Xiaowei Zhou, and Qixing Huang. Path-invariant map networks. *CoRR*, abs/1812.11647, 2018.

[46] Tinghui Zhou, Philipp Krähenbühl, Mathieu Aubry, Qi-Xing Huang, and Alexei A. Efros. Learning dense correspondence via 3d-guided cycle consistency. In *CVPR*, pages 117–126, 2016.

[47] Tinghui Zhou, Yong Jae Lee, Stella X. Yu, and Alexei A. Efros. Flowweb: Joint image set alignment by weaving consistent, pixel-wise correspondences. In *CVPR*, pages 1191–1200. IEEE Computer Society, 2015.

[48] Xiaowei Zhou, Menglong Zhu, and Kostas Daniilidis. Multi-image matching via fast alternating minimization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4032–4040, 2015.