GTT-Net: Learned Generalized Trajectory Triangulation

Xiangyu Xu Enrique Dunn
Stevens Institute of Technology
{xxu24, edunn}@stevens.edu

Abstract

We present GTT-Net, a supervised learning framework for the reconstruction of sparse dynamic 3D geometry. We build on a graph-theoretic formulation of the generalized trajectory triangulation problem, where non-concurrent multi-view imaging geometry is known but global image sequencing is not provided. GTT-Net learns pairwise affinities modeling the spatio-temporal relationships among our input observations and leverages them to determine 3D geometry estimates. Experiments reconstructing 3D motion-capture sequences show GTT-Net outperforms the state of the art in terms of accuracy and robustness. Within the context of articulated motion reconstruction, our proposed architecture is 1) able to learn and enforce semantic 3D motion priors for shared training and test domains, while being 2) able to generalize its performance across different training and test domains. Moreover, GTT-Net provides a computationally streamlined framework for trajectory triangulation with applications to multi-instance reconstruction and event segmentation.

1. Introduction

Trajectory triangulation aims to estimate multi-view sparse dynamic 3D geometry in the absence of concurrent observations. Recent advances in modeling and estimating the spatio-temporal relationships among 2D observations have yielded solutions with increasing generality and effectiveness. However, such research efforts have focused on developing and exploiting geometric insights and formulations, relegating the analysis of higher-order semantic relationships among the geometric entities being estimated. This work addresses the data-driven explicit characterization and modeling of these properties within the context of generalized trajectory triangulation.

Learning to encode generic spatio-temporal relationships hinges on the geometric reference being used and the scope of the analysis. The choice of geometric reference typically poses a dichotomy between Eulerian (e.g. field approach) vs. Lagrangian (e.g. particle approach) representations,

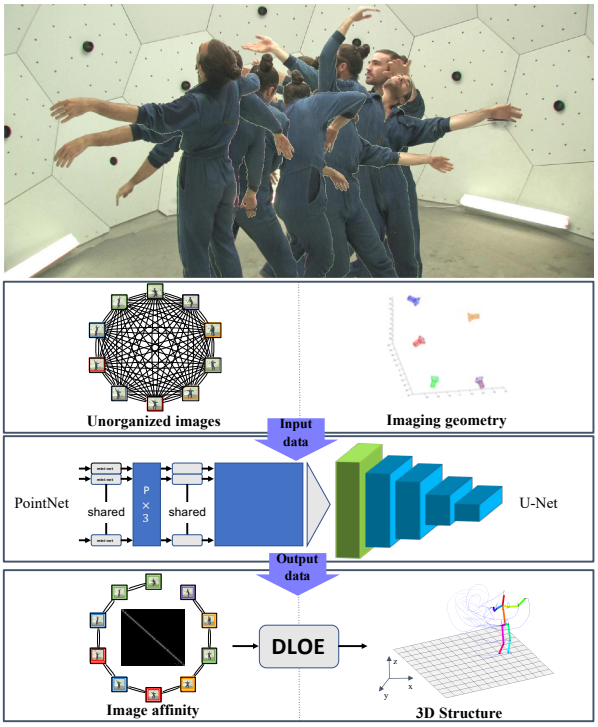


Figure 1: GTT-Net Workflow. Input camera poses and 2D features are mapped to a latent space encoding a pairwise affinity matrix leveraged to estimate 3D geometry.

where the former defines interactions among rigidly structured adjacency-based neighborhoods (e.g. voxel laticces), the latter defines interactions based on generic notions of proximity (e.g. nearest-neighbor graphs). Although scope is tightly coupled to these interaction mechanisms, the efficiency vs. comprehensiveness trade-offs between local and global analysis, determine the efficacy of the learned models and representations. We target a discrete-continuous local-global middle ground by 1) learning to approximate pairwise affinities over all estimated geometric elements, through 2) the use of sparse continuous convolutions.

Along these lines, the recent framework for generalized trajectory triangulation (GTT) described in [40], poses the

estimation of such relationships in terms of the iterative continuous optimization of a graph-theoretic representation. However, said optimization offers relatively slow convergence and provides no straightforward mechanisms for codifying internal shape constraints or sequence-level motion priors. This work focuses on learning to synthesize a global shape affinity matrix directly from input 3D geometry to integrate with and leveraging the representation and formulation used in [40], see Fig. 1. Our contributions are:

- A learning-based solution to the joint reconstruction and sequencing problems from multi-view image capture.
- A generalizable learning and representation framework applicable across diverse input shape domains.
- An efficient and flexible cascaded training framework applicable across diverse types of supervisory information.

2. Related work

2.1. Trajectory Triangulation

Trajectory triangulation operates on the premise of known cameras. However, the lack of concurrency requires enforcing estimation constraints to discriminate among the space of solutions compliant with the input observations.

Motion priors. Avidan and Shashua [6] enforced analytical linear and conical motion constraints upon the estimated 3D point trajectories from monocular capture. Extensions to these motion priors, include [5, 6, 13, 31, 30, 22]. Vo et al. [36] used physics-based motion priors such as least kinetic energy, to formulate a bundle adjustment framework for jointly optimizing static and dynamic 3D structure, camera poses and cross-capture temporal offsets.

Spatio-temporal smoothness. Enforcing spatio-temporal smoothness on the geometric estimation process [23, 24, 44, 45, 35, 42, 43, 36, 33, 34] has shown to be an effective approach to leverage temporally dense capture, such as those obtained by multiple video observers. Park et al. [23] parameterize a 3D trajectory in terms of linear combinations of a set of Direct Cosine Transform trajectory bases and optimize for each coefficient weight. In [24], Park et al. improve their method by selecting a small number of DCT bases according to N-fold a cross validation method to avoid low reconstructability cases. Zhu et al. [44] improve this result by adding a set of manual keyframes and adding L_1-norm regularization to their optimization to force sparsity on the DCT basis, instead of N-fold cross validation. Valmadre et al. [35] modify the reconstructability analysis for the trajectory basis solutions and propose two solutions: reducing trajectory bases by setting a gain threshold and applying a high-pass filter. Zheng et al. [43, 42] reconstructed dynamic 3D structure observed by multiple unsynchronized cameras with partial sequencing information, by assuming a self-expressive motion prior and implementing a bi-convex optimization problem. Recent works explicitly model and solve for relationships among dynamic 3D estimates and their spatio-temporal data associations [2, 3, 4, 1]. Along these lines, Xu et al. [40] used a graph-based formulation jointly estimating dynamic 3D structure and its corresponding discrete Laplace operator to reduce reliance on the temporal density and uniformity of the input data.

2.2. Learning for Sparse Dynamic 3D Geometry

Structured 3D Data Representations. Relevant to our problem, some early CNN-based approaches to 3D processing [12, 18] map the 3D representations onto a 2D space, where traditional CNN machinery is deployed. Such representations forgo accurate modeling of the geometric relationships lost or warped during projection. Performing 3D convolutions on volumetric representations [11, 19, 26, 29, 39] encodes 3D positional information and adjacency relations, but may quantize the representation space, leading alternatively, to data merging or sparsity. Riegler et al. [29] addressed this limitation by implementing 3D convolutions on data organized on an oct-tree data structure.

Unstructured 3D Data Representations. Qi et al. [25] worked on unstructured data enforcing network invariance to different permutations of the input feature by aggregating global information through max pooling. PointNet++ [27] improved performance by capturing local structure information. Wang et al. [38] propose a continuous convolutional neural network, which similarly to 2D convolutions, computes feature maps in terms of weighted sums of the input features. The use of a multi-layer perceptron (MLP) enabled adaptive weight determination based on geometric similarity. Boulch [10] computes a denser weighting function which takes into account the entire kernel.

Deep learning for Dynamic 3D reconstruction. Recently, network architectures have been proposed for the NRSfM problem. Kong et al. [16, 17] propose an unsupervised auto-encoder neural network to solve NRSfM problem under an orthogonal camera model by relying on a multi-layer sparse coding framework assumption. Wang et al. [37] developed a similar multi-layer sparse coding framework with improved generalization to weak and strong perspective camera models, along with increased robustness to missing data. Novotny et al. [21] learned a deep network to unambiguously factorize 3D structure and viewpoints by forcing consistency via canonicalization. Bai et al. proposed an end-to-end deep network [8] targeting multi-view 3D facial reconstruction. Another unsupervised end-to-end deep network [32] is introduced by Sidhu, which proposes the first dense neural NRSfM approach.

3. Generalized Trajectory Triangulation

The goal of generalized trajectory triangulation (GTT) is to recover time-varying 3D structure from a set of 2D observations with known imaging geometry, but absent of global

sequencing relations among input capture frames. Accordingly, GTT may be deemed a structure-only variation of the general non-rigid structure from motion problem (NRSfM).

A graph-theoretic formulation. A structure-motion graph representation has been recently presented in [40], where nodes are mapped from input images and have 3D geometry as attributes, while edges have the pairwise affinities as weights. Based on this representation, the GTT problem can be formulated as jointly estimating dynamic 3D geometry with the graph's Laplacian matrix, given by

$$\mathbb{L} = diag(\mathbb{A} \cdot \mathbf{1}) - \mathbb{A} \tag{1}$$

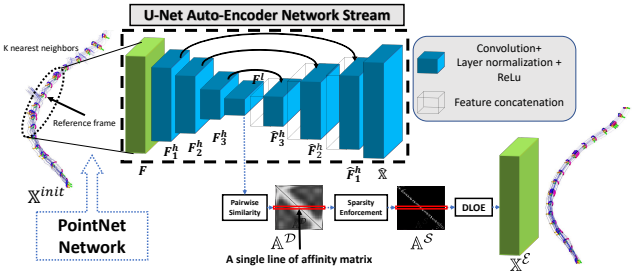
where $\mathbb{A}$ is the graph's affinity matrix, whose values $\mathbb{A}_{ij}$ correspond to the edge weights $e_{ij} \in \mathbb{R}_{\geq 0}$, characterizing the spatio-temporal relationships among 3D estimates $\mathbb{X}$. This generalization of the self-expressive motion prior [43], yields a non-convex optimization problem of the form

$$\min_{\mathbb{X},\mathbb{L}} \quad \mathcal{S}(\mathbb{L}\mathbb{X}) + \mathcal{T}\left(\mathbb{X}^{\top}\mathbb{L}\mathbb{X}\right) + \mathcal{R}(\mathbb{L}, \Theta) + \mathcal{O}(\mathbb{X}, \Theta), \tag{2}$$

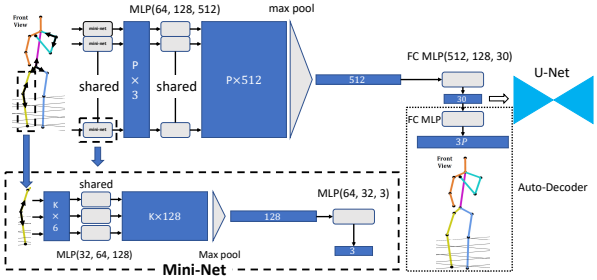
where $\Theta = \{\{\mathbf{x}_{np}\}, \{\mathbf{K}_n\}, \{\mathbf{M}_n\}\}$ denotes the aggregation of all input 2D observations and their camera parameters, $\mathcal{O}(\cdot)$ is a data term based on reprojection error, while $\mathcal{S}(\cdot)$, $\mathcal{T}(\cdot)$, and $\mathcal{R}(\cdot)$, are regularizers controlling, respectively, anisotropic smoothness, topological compactness, and multi-view reconstructability. Variables $\mathbb{X}$ and $\mathbb{L}$ are solved alternatively. That is, for fixed $\mathbb{L}$, 3D structure $\mathbb{X}$ is estimated by unconstrained quadratic programming; while for fixed $\mathbb{X}$, $\mathbb{L}$ is estimated by a linearly constrained quadratic problem. We refer readers to the original publication for further details [40]. While the above formulation achieved state of the art accuracy and robustness, its explicit full graph analysis limits its computational scalability. GTT-Net aims to alleviate this limitation by developing an encoder-decoder framework directly mapping the input 3D geometry $\mathbb{X}$ to the discrete Lapalace operator $\mathbb{L}$.

4. GTT-Net

As presented in [40], global dependencies required for affinity matrix optimization impose a computational bottleneck. GTT-Net learns to directly estimate these affinity values from input data. From an initial geometry $\mathbb{X}^{init}$, we learn a latent space F^l encoding the affinity among input 3D shapes. A sparse affinity matrix $\mathbb{A}^{\mathcal{S}}$ decoded from this latent space is fed to a differentiable quadratic optimization module to determine a refined dynamic geometry estimate $\mathbb{X}^{\mathcal{E}}$. We use data augmentation to explicitly target equivariance w.r.t. relevant input capture variants and perturbations. To accelerate training, we utilize cascaded training leveraging supervisory loss functions of increasing complexity.



(a) GTT-Net architecture



(b) PointNet Auto-encoder

Figure 2: (a) GTT-Net combines a U-Net learning a latent space from input features and affinity learning layers decoding a pairwise affinity values. (b) An optional PointNet auto-encoder maps input 3D shape structure to an abstract representation with fixed dimensionality.

4.1. Network Architecture

Parameterizing Input Geometry. A time-varying set of P 3D points $\mathbf{X}_{np}$ is observed in N images $\mathcal{I}_n$ captured by unsynchronized perspective cameras with known intrinsic and extrinsic matrices $\mathbf{K}_n$ and $\mathbf{M}_n$. 3D points are denoted as $\mathbf{X}_{np}$, while their image projections are $\mathbf{x}_{np}$. The set of all 3D points to estimate is represented by a $N \times 3P$ matrix

$$\mathbb{X} = \begin{bmatrix} \mathbf{X}_{11} & \dots & \mathbf{X}_{1P} \\ \vdots & \ddots & \vdots \\ \mathbf{X}_{N1} & \dots & \mathbf{X}_{NP} \end{bmatrix} \tag{3}$$

where $\mathbf{X}_{np}$ represents a 3D point's coordinates. Each row of $\mathbb{X}$ aggregates the P 3D points captured in frame n and constitutes a per-frame shape descriptor from which to estimate affinities. The input matrix $\mathbb{X}^{init}$ is estimated through pseudo-triangulation of viewing rays associated with $\mathbf{x}_{np}$.

Parametric Continuous Convolution Layers. Based on [38] and [10], we perform approximated continuous convolution operations on a given feature descriptor x as

$$h(x) = \int_{-\infty}^{+\infty} f(y)g(x-y)dy \approx \sum_{j \in \mathcal{N}_K^x}^{K} \frac{1}{K} f(y_j)g(x-y_j) \tag{4}$$

where y_j is one the K nearest neighbors of x, f is the feature map value function and g is a convolution kernel func-

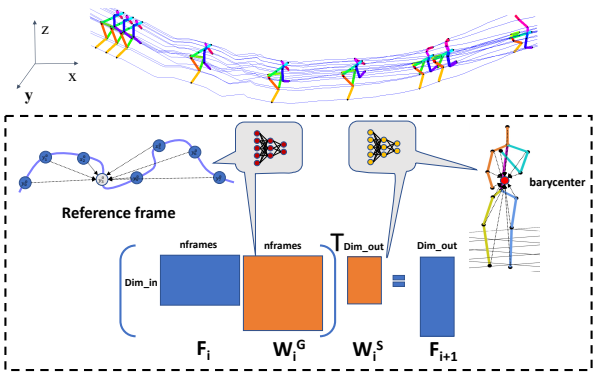


Figure 3: Each U-Net layer learns two types of continuous convolution filters: One applied among shape descriptors along the entire motion trajectory (W_i^G) and another based on intra-shape 3D point geometry (W_i^S).

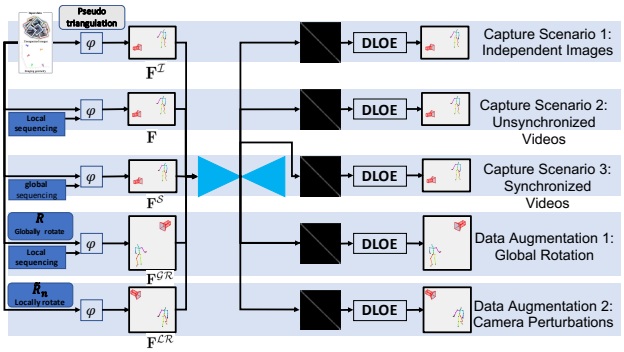


Figure 4: Five different variants of the input feature are generated and the network trained with shared weights. The convolution support domain is determined independently for each input variant.

tion approximated by a multi-layer perceptron (MLP)

$$g(x - y_j; \theta) = MLP(x - y_j; \theta). \quad (5)$$

This yields continuous output values using a finite set of learned weight parameters θ. We learn two types of filters for each layer, see Fig. 3. The first operates on the N single-frame descriptors and their K nearest neighbors, defining the support neighborhood w.r.t. spatio-temporal proximity among their shapes. Filter values are determined by geometry difference between shapes according to Eq. 5. The second operates on single-coordinate whole-trajectory descriptors and defines the support neighborhood domain w.r.t. intra-shape geometry (i.e. per-component proximity to their barycenter). Filter values are determined by geometry difference between joints.

U-Net Auto-Encoder Network Stream. We learn a latent space $\mathbf{F}^l$ using a U-Net encoder-decoder to perform dimensionality reduction through continuous parametric convolutions, see Fig. 2a. For translation and scale invariance across different input data, we apply layer normalization [7] for input and hidden layers by subtracting the mean μ, dividing by the standard deviation σ for each feature channel, while scaling and shifting by learnable parameters γ and β.

$$\hat{x}_{d,i} = \frac{x_{d,j} - \mu_d}{\sqrt{\sigma_d + \epsilon}} \gamma_d + \beta_d \quad (6)$$

An affinity matrix $\mathbb{A}^{\mathcal{D}}$ is computed in closed form as the pairwise similarity between latent space features $\mathbf{F}^l$ by

$$\mathbb{A}^{\mathcal{D}}_{nm} = \frac{1}{(1 + exp||\mathbf{F}^l_n - \mathbf{F}^l_m||)} \quad (7)$$

Unlike a regular graph affinity matrix, $\mathbb{A}^{\mathcal{D}}$ does not encode a graph's local connectivity. $\mathbb{A}^{\mathcal{D}}$ is sparsified into $\mathbb{A}^{\mathcal{S}}$ through a layer retaining the Q-highest affinity values among the per-feature convolution support domain $\mathcal{N}^x_K$. Empirically, we found Q=2 yielded the best performance (see Fig. 9b) and enforced this selection criteria deterministically. Finally, $\mathbb{A}^{\mathcal{S}}$ is fed into a differentiable instance of the Discrete Laplace Operator Estimator framework [40], denoted as a DLOE-layer, to estimate output 3D geometry $\mathbb{X}^{\mathcal{E}}$.

PointNet Network Stream. To allow for input shapes having different number of 3D points, we integrate a PointNet network [25] to provide a fixed-sized input into our U-Net, see Fig. 2b, We normalize each shape by subtracting its barycenter before PointNet maps it to a 30 dimension feature. To retain spatial separation among shapes we interpret PointNet's output as 10 virtual 3D points, add back the original barycenter, and feed them to the U-Net.

4.2. Supervisory Data

Depending on the capture scenario, complete or partial sequencing priors (e.g. sequencing among frames belonging to the same camera or video stream) may be available. As GTT-Net encodes these priors in terms of the support domain $\mathcal{N}^x_K$ used for continuous convolution, we only need to train a single network instance that is inclusive of all such variations. We explicitly instantiate such input prior variations within our training data and to account for capture variability, we perform data augmentation tailored to our formulation as in Fig. 4. We inject Gaussian noise to the 2D features $\mathbf{x}_{np}$ to account for feature localization ambiguity and apply geometric transformations to the ground truth data to account for capture variability.

Capture Scenario 1: Independent Images. Independent imagery provides no sequencing information. The convolution support domain for shape descriptors is determined by the spatial distribution of our initial 3D geometry $\mathbb{X}^{init}$, which is computed by exhaustive pseudo-triangulation of sparse 2D features. Once a rough 3D geometry is estimated per each frame, we compute the per-frame K-nearest neigh-

bors by the combination of triangulation error and viewing ray convergence analysis to eliminate frames with reduced camera baseline and unreliable triangulation. This input feature variant is denoted as $\mathbf{F}^{\mathcal{I}}$.

Capture Scenario 2: Unsynchronized Videos. For unsynchronized videos, sequencing priors are available for each independent video stream, allowing us to summarily eliminate from the support domain any frames from the same stream, and frames within another stream that are not adjacent among themselves. These constraints mitigate repetitive and/or self-intersecting 3D motions. The initialized input feature is defined as $\mathbf{F}$.

Capture Scenario 3: Synchronized Videos. For synchronized videos,[1] global sequencing is known and we can readily determine the K-nearest neighbors as those elements temporally adjacent to a given reference video frame. Also, pseudo-triangulation efficiency and reliability can benefit from guidance from the known sequencing info. This input feature variant is denoted as $\mathbf{F}^{\mathcal{S}}$.

Data Augmentation 1: Global Structure Rotation. Feature normalization in our encoder layers mitigates global scale and displacement variations. To promote rotational invariance we generate augmented input instances by randomly rotating initial 3D structure and camera poses jointly. While this transformation does not change input 2D feature locations, it targets the generalization of 3D and sequencing estimates. This input feature variant is denoted as $\mathbf{F}^{\mathcal{GR}}$.

Data Augmentation 2: Camera Perturbations. We inject structured perturbations to our input by randomly rotating and translating the camera pose of each frame. Since this transformation changes the imaging geometry, it alters the input 2D features used to initialize both 3D structure, and the K nearest neighbors associated to each frame. This input feature variant is denoted as $\mathbf{F}^{\mathcal{LR}}$.

4.3. Loss functions

U-Net Reconstruction Loss. To train our U-Net auto-encoder, we penalize the difference between the input and the reconstructed output maps, which correspond, respectively, to the initialized 3D structure $\mathbb{X}^{init}$ and a decoded 3D structure $\hat{\mathbb{X}}$. We penalize the differences between each hidden feature map $\mathbf{F}_i^h$ inside the encoder and the symmetrically corresponding hidden feature map $\hat{\mathbf{F}}_i^h$ in the decoder as in Fig. 2a. The loss function is written as,

$$\ell^{\text{Æ}} = \frac{1}{NP}(||\mathbb{X}^{init} - \hat{\mathbb{X}}||_F^2 + \sum_i^{d-1} ||\mathbf{F}_i^h - \hat{\mathbf{F}}_i^h||_F^2), \quad (8)$$

where $d = 4$ is the depth of the encoder and decoder layers.

(Pseudo) Ground Truth Affinity Loss. Ground truth affinity matrix optimization is computationally intractable (NP-hard). Hence, we use ground truth sequencing to generate a proxy (pseudo) ground truth affinity matrix $\mathbb{A}^{\mathcal{G}}$ having affinity values $\mathbb{A}_{i,j}^{\mathcal{G}} = 1$ for temporally consecutive frames and zero otherwise. If ground truth structure is available, we estimate real-valued affinities through optimization as in [40]. The reconstruction accuracy training by these two kinds of (pseudo) ground truth affinity matrix are compared in Fig. 8a. We penalize the difference between $\mathbb{A}^{\mathcal{D}}$ and $\mathbb{A}^{\mathcal{G}}$.

$$\ell^{\mathcal{A}} = \frac{1}{N}\sum_i^d ||\mathbb{A}^{\mathcal{D}} - \mathbb{A}^{\mathcal{G}}||_F^2 \quad (9)$$

3D Reconstruction Loss. Given the affinity matrix $\mathbb{A}^{\mathcal{S}}$ estimated by GTT-Net, we generate the corresponding Laplacian matrix as in Eq.(1) and estimate the 3D geometry $\mathbb{X}^{\mathcal{E}}$ by solving a quadratic programming problem. We penalize the 3D structure estimation error w.r.t. ground truth $\mathbb{X}^{\mathcal{G}}$ as

$$\ell^{\mathcal{X}} = \frac{1}{NP}||\mathbb{X}^{\mathcal{E}} - \mathbb{X}^{\mathcal{G}}||_F^2 \quad (10)$$

Smoothness Loss. In the absence of ground truth 3D structure $\mathbb{X}^{\mathcal{G}}$, we penalize the first and second terms in Eq.2, to foster local smoothness and linear topological structure.

$$\ell^{\mathcal{S}} = \mathcal{S}(\mathbb{L}\mathbb{X}) + \mathcal{T}(\mathbb{X}^{\top}\mathbb{L}\mathbb{X}) \quad (11)$$

PointNet Auto-encoder reconstruction Loss. If the PointNet stream is considered, we penalize the difference between it's input $\mathbb{X}^{init}$ and output map reconstructed by a domain-specific decoder $\hat{\mathbb{X}}^P$. In this scenario, the input to our U-Net is PointNet's fixed-dimension latent space.

$$\ell^{\mathcal{P}} = \frac{1}{NP}(||\mathbb{X}^{init} - \hat{\mathbb{X}}^P||_F^2) \quad (12)$$

4.4. A Cascaded Supervision Strategy.

The loss functions just described address a diversity of performance aspects we aim to control through supervision. However, they impose different levels supervisory specificity as well as computational burden. In order to streamline the training process, we partition it into sequential stages, each one of them considering supervisory loss functions of increasing specificity and complexity.

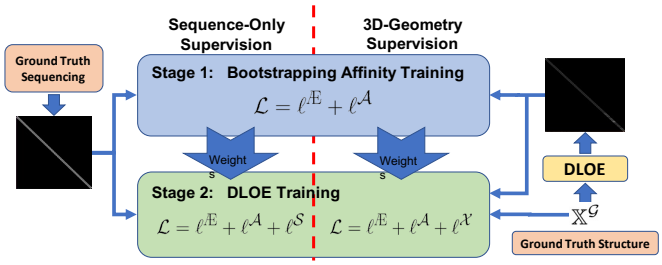
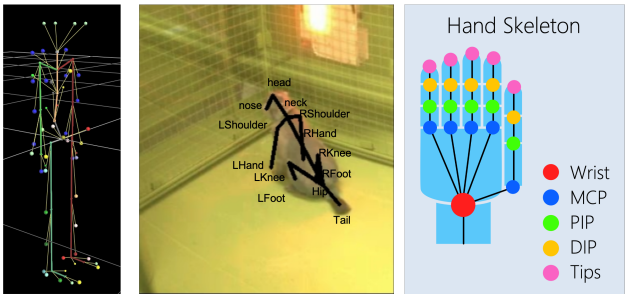


Figure 5: Cascaded Supervision Strategy.

[1] Synchronization denotes temporal alignment, not capture concurrency



(a) Human (b) Monkey (c) hand

Figure 6: PointNet auto-encoder is trained on three diverse articulated motion datasets: human, monkey and hands.

We aim to bootstrap the training process using efficient weak supervision and later improve upon the quality of the results by incorporating more targeted and computationally burdensome loss functions. We observed that strong supervision based on the output of the DLOE layers, while being the most effective, significantly slowed down convergence rate and increased the processing time for each epoch. Accordingly, DLOE-based supervision is used to fine-tune affinity estimation and omitted during initial training epochs. We now describe our 2-stage cascaded approach, shown in Fig. 5.

Stage 1: Bootstrap Affinity Supervision. Stage 1 only enforces sequencing constraints and relies on the $\ell^{\mathcal{AE}}$ and $\ell^{\mathcal{A}}$ loss functions. The goal is to accurately learn to auto-encode U-Net's input signal, while effectively learning pair-wise affinity. For sequencing-only supervision, the binary version of $\mathbb{A}^{\mathcal{G}}$ is used to target the identification of temporal neighborhoods. Conversely, if ground truth 3D geometry $\mathbb{X}^{\mathcal{G}}$ is available, the continuous version of $\mathbb{A}^{\mathcal{G}}$ is used to target fine-grain affinity estimation.

Stage 2: DLOE-based Supervision. Stage 2 leverages the DLOE model to enforce geometric regularization on the output 3D structure. For sequencing-only supervision, we enforce the smoothness loss function $\ell^{\mathcal{S}}$, to learn affinity values in $\mathbb{A}^{\mathcal{S}}$ yielding smooth 3D trajectories. For training instances where $\mathbb{X}^{\mathcal{G}}$ is available, we replace $\ell^{\mathcal{S}}$ with a 3D reconstruction loss $\ell^{\mathcal{X}}$ for fully supervised learning.

5. Experiments

5.1. Motion Capture Datasets

We use motion capture data [20] of 130 human 3D motions for 31 joints with frame rates of 120 Hz. We choose 10 of the 130 motions, each averaging ∼300 frames for testing. We generate training datasets by randomly choosing from the remaining 120 datasets with varying levels of 2D noise, frames rates and percent of joints missing. We simulate four virtual cameras with 1000×1000 resolution and 1000 focal length. Dynamic 3D joint positions are projected on them as 2D observations at a distance of 3m. For each 3D motion in both training and testing datasets, temporal sampling is performed at 30Hz and concurrent observations are systematically avoided to ensure all cameras are unsynchronized. We show results of 3D reconstruction accuracy comparisons in Fig. 7. GTT-Net is compared against discrete Laplace operator estimation (DLOE)[40], self-expressive dictionary learning (SEDL)[43], trajectory basis (TB)[24], high-pass filter (HPF)[35] and the pseudo-triangulation approach in Sec.4.2. SEDL requires partial sequencing information. TB and HPF require complete ground truth sequencing.

Varying 2D noise. We randomly add 2D Guassian noise with standard deviation from 1 to 5 pixels to our observations. Fig. 7a shows GTT-Net is competitive with other methods across all sequencing information conditions. When full sequencing info is available, GTT-Net outperforms geometry-only methods (e.g. DLOE), indicating we learn improved affinity relations to triangulate 3D trajectories. Even without any sequencing info, GTT-Net outperforms methods leveraging global sequencing.

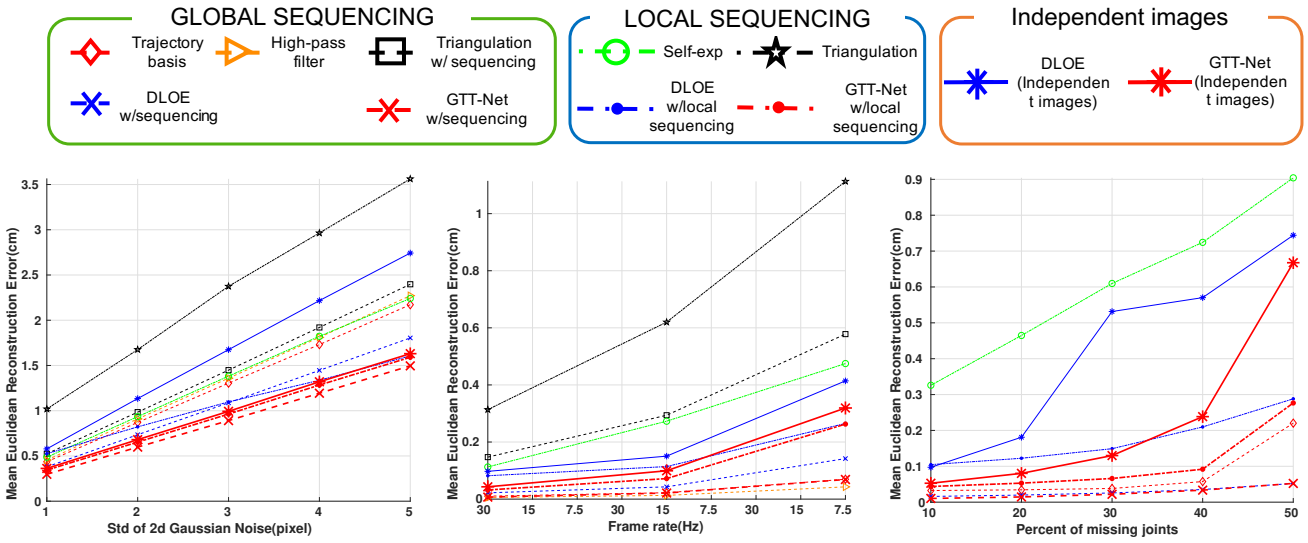
Varying frame rates. We simulate lower frame rate conditions by downsampling the 2D capture to 7.5Hz, 15Hz and 30Hz. As shown in Fig. 7b, our method performs better than DLOE on the conditions of partial sequencing information and no sequencing information. Working with full sequencing information, our method is still competitive.

Missing data. We randomly decimate 3D joints at rates varying from 10% to 50%, and compare GTT-Net's robustness against missing and/or occluded input features, see Fig. 7c. Only DLOE, SEDL and TB are able to operate having missing joints. The robustness of GTT-Net is competitive in all sequencing information conditions.

Ablation of cascaded training. Fig. 8a compares the reconstruction error distribution among the different stages in our cascaded training strategy. We include a self-supervised version using only $\ell^{\mathcal{AE}}$ and $\ell^{\mathcal{A}}$ loss functions without external data. Surprisingly, self-supervised training is strongly competitive with full training cascade results, although subject to grater variability.

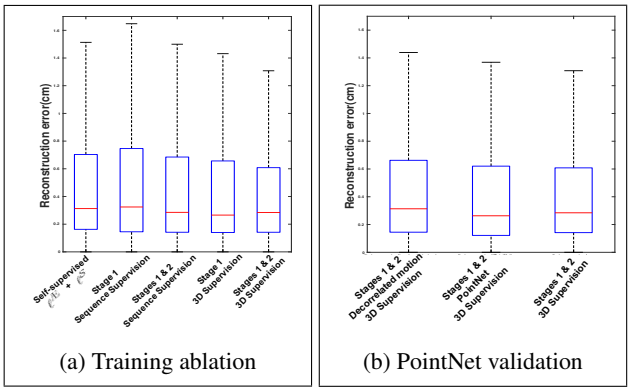
PointNet network validation. The PointNet-enabled variant of GTT-Net is trained on different datasets of articulated 3D motion, such as monkeys[9], hands[41] and humans[2], see Fig. 6, all having different joint topology compared to the testing data. In Fig. 8b, we compare the reconstruction error distribution of three GTT-Net variants: 1) a Multi-domain PointNet-enabled GTT-Net, 2) a Single-Domain 3D-Supervised GTT-Net and 3) a Single-Domain 3D-Supervised GTT-Net where random rigid motions are applied to individual joint 3D trajectories to decorrelate their motion from the original motion semantics. Our Point-

[2]CMU Mocap (http://mocap.cs.cmu.edu/)



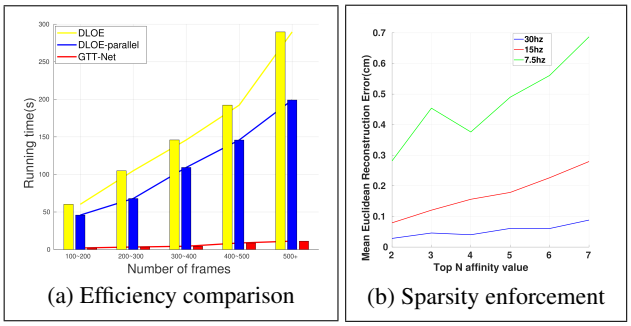
(a) Noise level (b) Frame rates (c) Missing points

Figure 7: (a) 3D Reconstruction error of the motion capture datasets under different level of 2D noise, (b) frames rates and (c) different percent of missing frames



(a) Training ablation (b) PointNet validation

Figure 8: Reconstruction error distributions. (a) Different training cascade variations. (b) Different GTT-Net variants.



(a) Efficiency comparison (b) Sparsity enforcement

Figure 9: (a) Computational efficiency comparision with DLOE [40]. (b) Reconstruction error for different sparsity levels (i.e. keeping top N affinity value for each row).

Net variant outperforms the variant training on decorrelated input 3D motion and is competitive with the Single-Domain 3D-Supervised variant even though our PointNet variant was not exposed to the test domain during training. The fact that training on decorrelated motion provides inferior performance, indicates our GTT-Net framework effectively learns to enforce general 3D motion semantics when estimating inter-shape affinities.

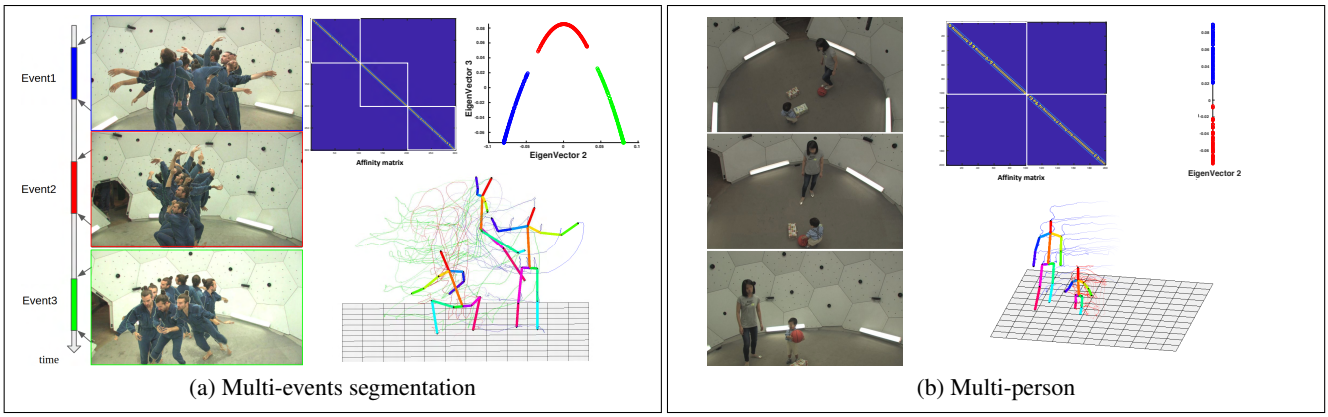
Computational advantages over [40]: GTT-Net is over an order of magnitude ($\sim$ 30X average speedup) faster than the open-source version of [40] when estimating a single full-graph affinity matrix across different sequence lengths, while consistently being more accurate as in Fig. 9a.

5.2. Cross-Domain Multi-view Video Datasets

Experiments on different 3D shapes classes illustrate the generality of our PointNet-enabled GTT-Net variant. The multi-view Human Ski [28] and Dog [15] datasets were unsynchronized and their provided 2D features were input to GTT-Net. Fig. 11 illustrates our qualitative results. GTT-Net was not exposed to either test domain during training.

5.3. Panoptic Studio dataset

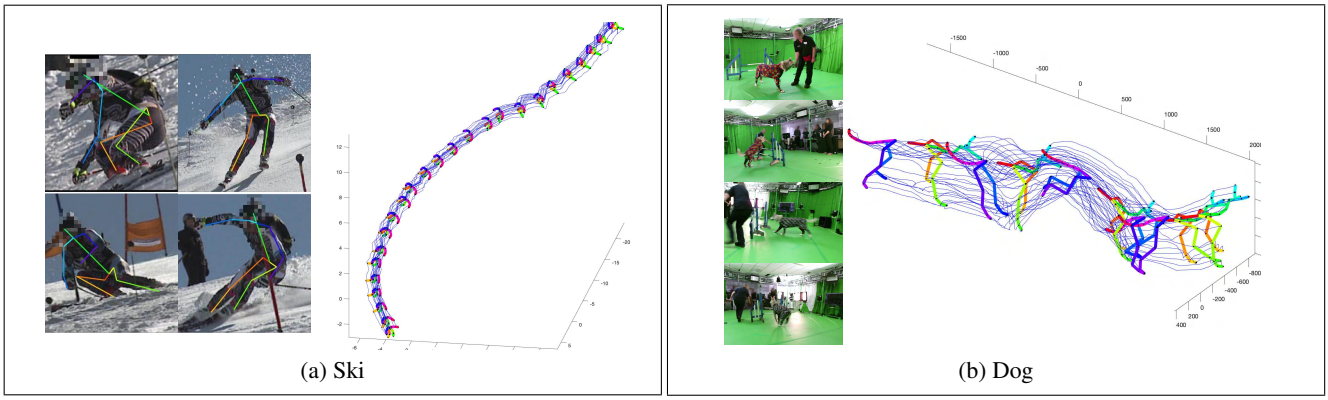
CMU Panoptic Studio dataset [14] contains synchronized multi-view videos, 2D human joint estimates and camera poses. We sample video frames to generate multi-view unsynchronized image streams. Again, as the dataset-provided sparse shape feature inputs (i.e. skeleton joints)



(a) Multi-events segmentation

(b) Multi-person

Figure 10: (a) Result on a disjoint dancing scene. The affinity matrix and spectral analysis show the three segments (b) This scenario includes an adult and a toddler which are clustered by the affinity matrix and corresponding spectral analysis.



(a) Ski

(b) Dog

Figure 11: Qualitative results on unsynchronized multi-view video capture. GTT-Net was not trained on the test domain.

are different from the 31-dimensional sparse shape features used for training, we use the PointNet variant of GTT-Net.

Application to Event Segmentation. For multi-view videos capturing temporally separated events, our goal is to jointly reconstruct the dynamic 3D structure and segment all events based on the estimated affinity matrix $\mathbb{A}^{\mathcal{S}}$. $\mathbb{A}^{\mathcal{S}}$ described a graph with multiple connected components, each of which corresponds to a separate event. For each segmented event, the sequencing of its constituting images was directly extracted from the affinity matrix. From top right in Fig. 10a, we can notice the chain-like structure for each event by performing spectral analysis on the affinity matrix.

Application to Multi-Target Scenarios. We consider the case where multiple shapes are captured in multi-view cameras, but the shape's correspondence among the images is not available. Given N images $\{\mathcal{I}_n\}$ with maximal M shape captured, the goal is to reconstruct the aggregated dynamic 3D structure $\mathbb{X}_{i,:} \in \mathbb{R}^{3MP}$.We propose a solution for this case based on GTT-Net: **1**) We separately create virtual frames $\{\tilde{\mathcal{I}}_{n,m}\}$(each observing P 3D points) for each of the subjects captured in original images. **2**) Execute GTT-Net on the (up to NM) new virtual images to reconstruct the 3D structure and generate the corresponding affinity matrix as in the single shape case. **3**) Cluster individual objects based on the affinity matrix by any standard clustering method. **4**) Merge estimated 3D shapes originating from the same image. **5**) Run GTT-Net on the N original input images with aggregated shape feature to refine the decoupled event reconstructions from step 2. Fig. 10b shows our results for a two-target scenario.

6. Conclusion

GTT-Net uses supervised learning to estimate pairwise spatio-temporal affinities and compute dynamic 3D geometry from image observations. Our framework allows for a diverse set of training scenarios and leverages them on a cascaded supervision strategy to both improve training efficiency and be adaptive to available supervisory information. Moreover, the proposed system is robustly applicable across different shape-trajectory domains, while outperforming the current state of the art.

References

[1] Antonio Agudo. Segmentation and 3d reconstruction of non-rigid shape from rgb video. In *2020 IEEE International Conference on Image Processing (ICIP)*, pages 2845–2849. IEEE, 2020.

[2] Antonio Agudo and Francese Moreno-Noguer. Deformable motion 3d reconstruction by union of regularized subspaces. In *2018 25th IEEE International Conference on Image Processing (ICIP)*, pages 2930–2934. IEEE, 2018.

[3] Antonio Agudo and Francesc Moreno-Noguer. A scalable, efficient, and accurate solution to non-rigid structure from motion. *Computer Vision and Image Understanding*, 167:121–133, 2018.

[4] Antonio Agudo and Francesc Moreno-Noguer. Robust spatio-temporal clustering and reconstruction of multiple deformable bodies. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(4):971–984, 2019.

[5] Shai Avidan and Amnon Shashua. Trajectory triangulation of lines: Reconstruction of a 3d point moving along a line from a monocular image sequence. In *Proceedings of the 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, pages 62–66. IEEE, 1999.

[6] Shai Avidan and Amnon Shashua. Trajectory triangulation: 3d reconstruction of moving points from a monocular image sequence. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (4):348–357, 2000.

[7] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[8] Ziqian Bai, Zhaopeng Cui, Jamal Ahmed Rahim, Xiaoming Liu, and Ping Tan. Deep facial non-rigid multi-view stereo. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[9] Praneet C Bala, Benjamin R Eisenreich, Seng Bum Michael Yoo, Benjamin Y Hayden, Hyun Soo Park, and Jan Zimmermann. Openmonkeystudio: Automated markerless pose estimation in freely moving macaques. *bioRxiv*, 2020.

[10] Alexandre Boulch. Generalizing discrete convolutions for unstructured point clouds. In *3DOR*, pages 71–78, 2019.

[11] Benjamin Graham and Laurens van der Maaten. Submanifold sparse convolutional networks. *arXiv preprint arXiv:1706.01307*, 2017.

[12] Saurabh Gupta, Ross Girshick, Pablo Arbeláez, and Jitendra Malik. Learning rich features from rgb-d images for object detection and segmentation. In *European conference on computer vision*, pages 345–360. Springer, 2014.

[13] Mei Han and Takeo Kanade. Reconstruction of a scene with multiple linearly moving objects. *International Journal of Computer Vision*, 59(3):285–300, 2004.

[14] Hanbyul Joo, Tomas Simon, Xulong Li, Hao Liu, Lei Tan, Lin Gui, Sean Banerjee, Timothy Godisart, Bart Nabbe, Iain Matthews, et al. Panoptic studio: A massively multiview system for social interaction capture. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(1):190–204, 2017.

[15] Sinead Kearney, Wenbin Li, Martin Parsons, Kwang In Kim, and Darren Cosker. Rgbd-dog: Predicting canine pose from rgbd sensors. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[16] Chen Kong and Simon Lucey. Deep non-rigid structure from motion. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1558–1567, 2019.

[17] Chen Kong and Simon Lucey. Deep non-rigid structure from motion with missing data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[18] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

[19] Daniel Maturana and Sebastian Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 922–928. IEEE, 2015.

[20] M. Müller, T. Röder, M. Clausen, B. Eberhardt, B. Krüger, and A. Weber. Documentation mocap database hdm05. Technical Report CG-2007-2, Universität Bonn, June 2007.

[21] David Novotny, Nikhila Ravi, Benjamin Graham, Natalia Neverova, and Andrea Vedaldi. C3dpo: Canonical 3d pose networks for non-rigid structure from motion. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7688–7697, 2019.

[22] Hyun Soo Park and Yaser Sheikh. 3d reconstruction of a smooth articulated trajectory from a monocular image sequence. In *2011 International Conference on Computer Vision*, pages 201–208. IEEE, 2011.

[23] Hyun Soo Park, Takaaki Shiratori, Iain Matthews, and Yaser Sheikh. 3d reconstruction of a moving point from a series of 2d projections. In *European Conference on Computer Vision*, pages 158–171. Springer, 2010.

[24] Hyun Soo Park, Takaaki Shiratori, Iain Matthews, and Yaser Sheikh. 3d trajectory reconstruction under perspective projection. *International Journal of Computer Vision*, 115(2):115–135, 2015.

[25] Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[26] Charles R Qi, Hao Su, Matthias Nießner, Angela Dai, Mengyuan Yan, and Leonidas J Guibas. Volumetric and multi-view cnns for object classification on 3d data. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5648–5656, 2016.

[27] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in neural information processing systems*, pages 5099–5108, 2017.

[28] Helge Rhodin, Jörg Spörri, Isinsu Katircioglu, Victor Constantin, Frédéric Meyer, Erich Müller, Mathieu Salzmann, and Pascal Fua. Learning monocular 3d human pose estimation from multi-view images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8437–8446, 2018.

[29] Gernot Riegler, Ali Osman Ulusoy, and Andreas Geiger. Octnet: Learning deep 3d representations at high resolutions.

In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3577–3586, 2017.

[30] Dana Segal and Amnon Shashua. 3d reconstruction from tangent-of-sight measurements of a moving object seen from a moving camera. In *European Conference on Computer Vision*, pages 621–631. Springer, 2000.

[31] Amnon Shashua, Shai Avidan, and Michael Werman. Trajectory triangulation over conic section. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 1, pages 330–336. IEEE, 1999.

[32] Vikramjit Sidhu, Edgar Tretschk, Vladislav Golyanik, Antonio Agudo, and Christian Theobalt. Neural dense non-rigid structure from motion with latent space constraints. In *European Conference on Computer Vision (ECCV)*, 2020.

[33] Tomas Simon, Jack Valmadre, Iain Matthews, and Yaser Sheikh. Separable spatiotemporal priors for convex reconstruction of time-varying 3d point clouds. In *European Conference on Computer Vision*, pages 204–219. Springer, 2014.

[34] Tomas Simon, Jack Valmadre, Iain Matthews, and Yaser Sheikh. Kronecker-markov prior for dynamic 3d reconstruction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(11):2201–2214, 2017.

[35] Jack Valmadre and Simon Lucey. General trajectory prior for non-rigid reconstruction. In *Proceedings of 2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1394–1401. IEEE, 2012.

[36] Minh Vo, Srinivasa G Narasimhan, and Yaser Sheikh. Spatiotemporal bundle adjustment for dynamic 3d reconstruction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1710–1718, 2016.

[37] Chaoyang Wang, Chen-Hsuan Lin, and Simon Lucey. Deep nrsfm++: Towards 3d reconstruction in the wild. *arXiv preprint arXiv:2001.10090*, 2020.

[38] Shenlong Wang, Simon Suo, Wei-Chiu Ma, Andrei Pokrovsky, and Raquel Urtasun. Deep parametric continuous convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2589–2597, 2018.

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

[40] Xiangyu Xu and Enrique Dunn. Discrete laplace operator estimation for dynamic 3d reconstruction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[41] Shanxin Yuan, Qi Ye, Bjorn Stenger, Siddhant Jain, and Tae-Kyun Kim. Bighand2. 2m benchmark: Hand pose dataset and state of the art analysis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4866–4874, 2017.

[42] Enliang Zheng, Dinghuang Ji, Enrique Dunn, and Jan-Michael Frahm. Sparse dynamic 3d reconstruction from unsynchronized videos. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4435–4443, 2015.

[43] Enliang Zheng, Dinghuang Ji, Enrique Dunn, and Jan-Michael Frahm. Self-expressive dictionary learning for dynamic 3d reconstruction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(9):2223–2237, 2017.

[44] Yingying Zhu, M Cox, and S Lucey. 3d motion reconstruction for real-world camera motion. In *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE Computer Society, 2011.

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