Online Unsupervised Learning of the 3D Kinematic Structure of Arbitrary Rigid Bodies

Urbano Miguel Nunes and Yiannis Demiris
Personal Robotics Lab, Imperial College London, UK
{um.nunes, y.demiris}@imperial.ac.uk

Abstract

This work addresses the problem of 3D kinematic structure learning of arbitrary articulated rigid bodies from RGB-D data sequences. Typically, this problem is addressed by offline methods that process a batch of frames, assuming that complete point trajectories are available. However, this approach is not feasible when considering scenarios that require continuity and fluidity, for instance, human-robot interaction. In contrast, we propose to tackle this problem in an online unsupervised fashion, by recursively maintaining the metric distance of the scene's 3D structure, while achieving real-time performance. The influence of noise is mitigated by building a similarity measure based on a linear embedding representation and incorporating this representation into the original metric distance. The kinematic structure is then estimated based on a combination of implicit motion and spatial properties. The proposed approach achieves competitive performance both quantitatively and qualitatively in terms of estimation accuracy, even compared to offline methods.

1. Introduction

The kinematic structure of an articulated rigid body provides a compact and meaningful representation, which is useful for robot manipulation tasks and object kinematic recognition [21], finding kinematic correspondences between objects [6], interactively perceiving and acting with the environment [3], among others. Thus, it is an active research topic in computer vision [5, 17, 23] and robotics [14]. Model-based kinematic structure learning methods have shown to be superior in domains where the model of the scene is known [4]. However, when this is not the case, unsupervised methods are commonly used, and the problem of kinematic structure learning is divided into two well-known problems, namely motion segmentation of rigid bodies [7, 8, 11] and kinematic structure generation based on motion and/or other cues [5, 23]. We adopt the unsupervised approach, as we intend to estimate the kinematic structure of arbitrary articulated rigid objects.

There have been significant contributions to kinematic structure estimation using images from monocular RGB cameras [5, 23]. Typically, these methods rely on tracking complete feature point trajectories to perform motion segmentation and are sensitive to outliers from the background. For example, to bridge the latter limitation, Chang and Demiris [5] included an adaptive object boundary generation to distinguish the actual object from the background. This problem can be relatively easily overcome using RGB-D sensors, which acquire 3D data where foreground/background segmentation is trivial.

Existing kinematic structure learning methods are offline and/or are not feasible for real-time applications, such as perceiving and interacting with objects in the environment [3] by learning kinematic correspondences [6]. Thus, to the best of our knowledge, we propose the first approach that can maintain the 3D structure estimated in real-time and in an online fashion. Based on the observation that the distance between points belonging to the same rigid body is constant [18], we maintain the motion information implicitly for each incoming frame. We then compute the similarity measure between each point, accommodating for lost points and noise (*e.g.* due to occlusion) by devising an effective policy to incorporate new points while maintaining an even distribution of points tracked. Finally, the underlying kinematic structure can be estimated on demand. An illustration of our method in action is presented in Fig. 1.

Although the proposed method is independent of the actual source of 3D data, for simplicity we focus on data acquired by an RGB-D sensor. We could also consider a pipeline where the 3D shape of an object is estimated from monocular images [1, 2] and the corresponding 3D points are provided as inputs for the proposed method. The resultant scale ambiguity would not be an issue, since our approach relies on relative distances.

Main contributions: 1) To the best of our knowledge, this is the first approach that directly tackles the problem of online and real-time 3D kinematic structure learning of arbitrary articulated rigid bodies; 2) We naturally handle lost points and occurring noise, and thus the method does

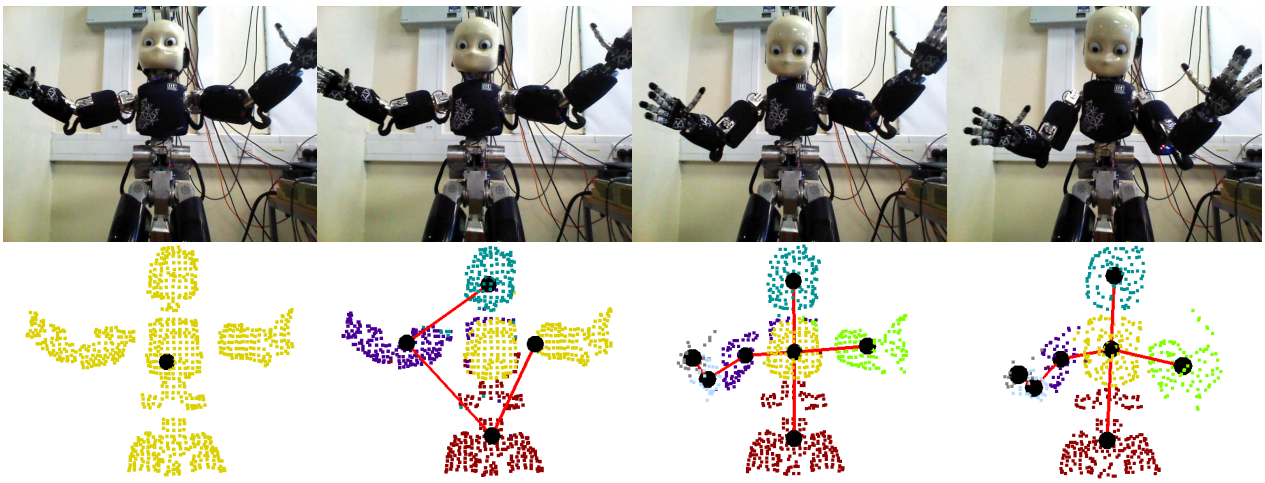


Figure 1: Frames 19, 24, 162 and 499 of the *iCub* sequence (first row) and the corresponding 3D kinematic structure estimation by the proposed method (second row). Until around frame 19, no significant motion is observed and thus the whole object is segmented. By frame 24, some motions are perceived and an initial sketch of the 3D kinematic structure is built. At frame 162, the right arm has been correctly segmented into three parts, with some points around the torso remaining misclassified, until they are correctly labelled, which is seen in frame 499. Fig. best viewed in colour.

not depend on complete point trajectories; 3) We propose to combine implicit motion and biharmonic distance [13] to build more plausible kinematic structures, since the object's motion and spatial properties are considered.

2. Related work

The literature is grouped into the three parts most related to the main contributions of this work: 1) motion segmentation from 3D data, 2) kinematic structure learning based on 3D data and 3) online motion segmentation methods.

Motion segmentation based on 3D data: Using data from an RGB-D camera, Perera and Barnes [18] propose an approach to motion segmentation that does not explicitly estimate the underlying motion. The main observation is that the Euclidean distance between points that share the same motion does not vary. They build a similarity measure based on the corresponding standard deviation, where a thresholded binary decision is made accordingly (*i.e.* if the standard deviation is below a given threshold, then the corresponding points share the same motion). We adopt the same observation as the basis of our method, but do not perform any binary decision to build the similarity measure and consequently do not need a threshold to group points. Furthermore, (dis)similarity measures based on standard deviation are known to be sensitive to outliers and are not suitable for long-term scenarios, as they stabilise at a certain point.

Zografos *et al*. [29] propose a method that resorted to group theoretical invariants, where the canonical representation of points belonging to the same motion may be recovered by a unique alignment, corresponding to the associated rigid transformation. Although comparably fast and accurate, they require the number of motions to be given. Also, with the aim of estimating rigid transformations to perform motion segmentation, Judd *et al*. [11] apply a multimotion fitting technique, where the point trajectories and motion models are jointly estimated, based on given observations and prior estimated models. The number of motions is estimated through an iterative procedure where motion candidates are proposed and merged. Nunes and Demiris [16] apply subspace clustering methods in the context of 3D motion segmentation, where they additionally incorporated an adaptive spectral clustering method to estimate the number of rigid motions. In contrast to the method that follows, [16] cannot handle lost points, which constrains its usage to scenarios where complete point trajectories are available.

Kinematic structure estimation based on 3D data: Zhang *et al*. [26] perform a non-rigid matching between consecutive frames to keep track of the point trajectories. They then build a similarity matrix based on the standard deviation of the pairwise distances between points similarly to [18], and determine the kinematic structure of the object generating the minimum spanning tree, whose weights are based on the max-min Euclidean pairwise distances between each estimated part. In line with [5], we advocate to combine the topological structure of the object, based on the biharmonic distance [13] and implicit motion. Tzionas and Gall [22] propose a method to reconstruct articulated models that focus on tracking points based on deformable meshes, applying spectral clustering to the corresponding point trajectories with a threshold to estimate the number of body parts. Yuan *et al*. [24] present an approach that propagates and merges all models that were segmented within a given sequence. One of the limitations of [24] lies in the fact that it cannot robustly handle large displacements between frames, since points might get lost. We mitigate this issue by naturally handling lost points.

Online motion segmentation: The problem of motion segmentation in an online fashion is a non-trivial problem, mainly due to: 1) only information of current and past frames is available, 2) current and past frame information must be summarized, and 3) point trajectories may not span an entire sequence. Elqursh and Elgammal [8] propose an online motion segmentation method, where current and past frame information is maintained by considering motion and spatial affinities of 2D feature point trajectories. They then apply label propagation based on Markov random walks on the graph whose edges' weights are given by the similarities previously considered. We consider a similar approach, where instead the label propagation is carried by the *label spreading* [28] method, in order to allow adjustments to previously labelled points. Kang and Chung [12] propose a method that relies on building a spatial neighbourhood in the image plane for each point and inferring the underlying affine motion based on multiple motion hypotheses from randomly selected subsets of temporally linked points. Both approaches rely on 2D feature points, which may be insufficient to describe general case scenarios.

3. Methodology

We propose a method that considers the information of incoming frames in an online fashion to estimate the kinematic structure of arbitrary articulated rigid bodies. Only the following assumptions are made: 1) a 3D point cloud of the body is assumed to be given (which can be provided by any suitable sensor or by a sequential 3D shape reconstruction method from monocular images [1]) and 2) we assume the body is piecewise rigid, which implies that the pairwise Euclidean distance[1] between points belonging to the same body segment is constant throughout all frames; on the contrary, the pairwise Euclidean distance between points that belong to different body parts changes, provided sufficient relative motion is observed. Based on these assumptions, we design an algorithm to estimate the kinematic structure of arbitrary articulated rigid bodies. First, each point is tracked based on the scene flow [10] and the pairwise Euclidean distance between the points from the point cloud is computed for each incoming frame. The accumulated variation of pairwise Euclidean distance (AVPED) is obtained and used to build the correspondent similarity measure. *Label spreading* [28] is then applied so that points are segmented into distinct body parts. Finally, the kinematic structure of the object is generated as the corresponding graph's minimum spanning tree.

3.1. Notation

N is the number of points considered, c^f is the number of estimated body parts at frame $f \in \mathbb{N}$ and n_g^f is the number of points belonging to body part $g = \{1, \dots, c^f\}$. Given a point cloud P^f, the 3D position of the i-th point p_i^f is defined as $\mathbf{x}_i^f \in \mathbb{R}^3$, where $i = \{1, \dots, N\}$. $\mathbf{D}^f = [d_{ij}^f] \in \mathbb{R}^{N \times N}$ denotes the pairwise Euclidean distance between all points, $\tilde{\mathbf{D}}^f = [\tilde{d}_{ij}^f] \in \mathbb{R}^{N \times N}$ denotes the AVPED and $\mathbf{W}^f = [w_{ij}^f] \in \mathbb{R}^{N \times N}$ represents the similarity matrix built. The g-th body segment estimated is composed of a subset of points $S_g^f \subseteq P^f$, where $S_g^f \cap S_h^f = \emptyset, g \neq h$. The symbols $\odot$ and $|\cdot|$ denote the Hadamard product and the Euclidean distance, respectively. The superscript regarding the frame will be omitted when only one frame is being considered.

3.2. Formulation

At any frame, we aim to estimate the kinematic structure of an articulated rigid body, given its point cloud and respective 3D point positions

$$\mathbf{X}^f = \begin{bmatrix} \mathbf{x}_1^f & \mathbf{x}_2^f & \dots & \mathbf{x}_N^f \end{bmatrix} \quad \in \mathbb{R}^{3 \times N}. \tag{1}$$

We can compute the pairwise Euclidean distance $\mathbf{D}^f$ between all points and then obtain the AVPED

$$\tilde{\mathbf{D}}^f = \sum_{j=1}^{f} \frac{|\mathbf{D}^j - \mathbf{D}^{j-1}|}{t_s} = \tilde{\mathbf{D}}^{f-1} + \frac{|\mathbf{D}^f - \mathbf{D}^{f-1}|}{t_s}, \tag{2}$$

where t_s is the sampling period. With this formulation, the AVPED can also be interpreted as an accumulation of the absolute temporal rate of change in pairwise Euclidean distances. Note that by construction the AVPED defines a proper Euclidean distance matrix, which will be used to build a similarity matrix between every point considered and to cluster them. In addition, this formulation does not require the static camera assumption, since the motion is captured implicitly with respect to relative pairwise distances. In other words, only the relative distance matters and since every point would be subject to the same apparent motion, the pairwise distance perceived would be the same, regardless of the camera motion.

3.3. Building the similarity

The proposed method relies in the fact that the pairwise Euclidean distances between points belonging to the same body segment remain constant throughout the frames, whereas they vary for points belonging to different body parts provided enough evidence

$$\begin{cases} \tilde{d}_{ij} \leq \epsilon & \text{if } p_i, p_j \in S_g \\ \tilde{d}_{ij} > \epsilon & \text{otherwise} \end{cases}. \tag{3}$$

Thus, a natural construction of a similarity matrix is to associate points belonging to the same body part if the AVPED is near zero

$$\hat{w}_{ij} = \begin{cases} 1 & \text{if } \tilde{d}_{ij} \leq \epsilon \\ 0 & \text{otherwise} \end{cases}, \tag{4}$$

[1] We refer to the Euclidean distance as the l_2-norm.

and thresholding accordingly. Perera and Barnes [18] proposed a similar method, where instead they constructed this matrix considering the standard deviation of the pairwise distances between points. However, such an approach has a major limitation: a global threshold must be provided, which in practice, may depend on the observed scene or on the sensor measurement's errors. Typically, a different value must be manually found and set for each scenario, which can be cumbersome.

Instead, we embed the AVPED in a feature space and use the resultant representation to build the similarity matrix $\mathbf{W}$. In particular, since AVPED defines a proper Euclidean distance matrix, a configuration $\mathbf{Z} \in \mathbb{R}^{N \times k}, k \leq N$ can be found, such that the original Euclidean distances are preserved and we have that [19, 27]

$$\mathbf{B} = -\frac{1}{2}\mathbf{J}\left(\tilde{\mathbf{D}} \odot \tilde{\mathbf{D}}\right)\mathbf{J} = \mathbf{Z}\mathbf{Z}^{\mathsf{T}} \quad \in \mathbb{R}^{N \times N} \tag{5}$$

is the positive definite matrix of inner products of the underlying configuration $\mathbf{Z}$, where $\mathbf{J} = \mathbf{I} - \frac{1}{N}\mathbf{1}\mathbf{1}^{\mathsf{T}} \in \mathbb{R}^{N \times N}$, $\mathbf{I} \in \mathbb{R}^{N \times N}$ is the identity matrix and $\mathbf{1} \in \mathbb{R}^{N}$ is a vector of ones. $\mathbf{B}$ can be factorised by its eigendecomposition as

$$\mathbf{B} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{\mathsf{T}}, \tag{6}$$

where $\mathbf{V}$ is the orthogonal matrix of eigenvectors corresponding to the eigenvalues $(\lambda_1, \lambda_2, \ldots, \lambda_N)$ sorted in descending order in the diagonal of $\mathbf{\Lambda}$. Since $\mathbf{B}$ is positive definite, we may obtain a k-dimensional ($k \leq N$ non-zero eigenvalues) representation of $\tilde{\mathbf{D}}$ as

$$\mathbf{Z} = \mathbf{V}_k \mathbf{\Lambda}_k^{\frac{1}{2}}, \tag{7}$$

where $\mathbf{V}_k \in \mathbb{R}^{N \times k}$ is a matrix of the k eigenvectors corresponding to the first k leading eigenvalues contained in the diagonal of $\mathbf{\Lambda}_k \in \mathbb{R}^{k \times k}$. Finally, we build a similarity matrix $\mathbf{W}$, whose entries w_{ij} are given by an exponential radial basis function kernel

$$w_{ij} = \exp\left[-\left(\frac{|\mathbf{z}_i - \mathbf{z}_j|}{\gamma}\right)^2\right], \tag{8}$$

where $\mathbf{z}_i \in \mathbb{R}^k$ corresponds to the i-th row of $\mathbf{Z}$ and $\gamma \in \mathbb{R}_+$ is related to the kernel parameter. The resulting matrix will closely match Eq. (4), where points belonging to the same segment are associated, since their AVPED is almost zero.

3.4. Clustering by *label spreading*

Having built the similarity matrix $\mathbf{W}$, we can now employ any suitable clustering algorithm (*e.g.* spectral clustering [15, 25]), because the continuity of the scene is maintained by the AVPED. Although this works, the final results may exhibit discontinuities between frames, since we would not be taking full advantage of previous segmentation results [8]. Thus, in order to take into consideration previous labelling information, we use *label spreading* [28] on the graph defined by $\mathbf{W}$, which also allows re-labelling depending on a combination of past information and current graph structure. New labels are obtained by performing normalized cuts [20] on each sub-graph corresponding to existing labels. The approach is briefly detailed as follows.

***Label spreading*:** Based on the similarity matrix $\mathbf{W}^f$, we compute its normalized graph Laplacian as

$$\mathbf{L} = \mathbf{\Delta}^{-\frac{1}{2}}\mathbf{W}^f\mathbf{\Delta}^{-\frac{1}{2}} \quad \in \mathbb{R}^{N \times N}, \tag{9}$$

where $\mathbf{\Delta} \in \mathbb{R}^{N \times N}$ is the diagonal degree matrix of $\mathbf{W}^f$, whose elements are given by $\delta_i = \sum_j w_{ij}$. Then, the new labels $\mathbf{Y}^f \in \mathbb{R}^{N \times c^f}$ are obtained given previous labels $\mathbf{Y}^{f-1} \in \mathbb{R}^{N \times c^{f-1}}$ with each row having the probability of point p_i belonging to label g as follows

$$\mathbf{Y}^f = (1-\mu)(\mathbf{I} - \mu\mathbf{L})^{-1}\mathbf{Y}^{f-1}, \tag{10}$$

where $\mu \in [0, 1)$. The parameter μ controls how much the graph structure influences future labels in comparison to previous labels (*e.g.* when $\mu \to 1$ the graph structure is the dominant factor).

Obtaining new labels: Since *label spreading* only propagates existing labels, we need to be able to find new ones, *e.g.* new segments may emerge. Points belonging to an emergent new label would be assigned some existing label, increasing its intra-cluster variation. This means we could perform a normalized cut to each label's sub-graph and evaluate if its cost is below some threshold τ. Given the respective similarity sub-matrix $\mathbf{W}_g$, we compute its normalized graph Laplacian $\mathbf{L}_g$ according to Eq. (9), and obtain the eigenvector corresponding to the second largest eigenvalue[2]. The eigenvector is then thresholded according to the average of its values and the normalized cut cost is obtained. If this cost is below τ, then we split the cluster, obtaining a new one; otherwise, we keep it intact.

3.5. Building the kinematic structure

The kinematic structure is represented as the minimum spanning tree of a non cyclic graph $\mathcal{G} = (V, E)$ [5, 23]. Each segment's center $\mathbf{m}_g$ is treated as a vertex and the proximity between each pair of vertices (*i.e.* edges' weights) is given as

$$E_{ij} = \beta\tilde{d}_{ij} + (1-\beta)\hat{d}_{ij}, \tag{11}$$

where $\tilde{d}_{ij}$ is the AVPED and $\hat{d}_{ij}$ corresponds to the biharmonic distance [13] between segments' centers i and

[2]The Laplacian can also be computed as $\mathbf{L} = \mathbf{I} - \mathbf{\Delta}^{-\frac{1}{2}}\mathbf{W}\mathbf{\Delta}^{-\frac{1}{2}}$, the only difference being the values of the eigenvalues, from λ_i to $1 - \lambda_i$; hence the selection of the eigenvector corresponding to the second largest eigenvalue, instead of the second smallest one.

j, respectively. This distance provides a balance between geodesic distances for local neighbourhoods and large distances across the global structure. Each edge's weight is thus a combination of the AVPED, which stores motion information, and the biharmonic distance, which contains information about the shape and spatial structure of the body.

3.6. Handling lost points

While observing the scene, some points may get lost, *e.g*. due to tracking/measurement errors or some body part naturally disappearing from the field of view. Thus, the proposed method must be able to deal with these occurrences, as in a real-world scenario, such that their influence is minimized. One can observe that the AVPED, according to Eq. (2), is directly responsible for maintaining information about the scene. Therefore, we propose to handle lost points during the computation of $\tilde{\mathbf{D}}$. First, each point cloud is represented in an octree data structure [9], which allows us to have some knowledge about the 3D spatial structure of the scene and manipulate it accordingly. Whenever a point is lost, we then search for the region with fewer points and randomly select a point to start tracking. The implementation is detailed as follows.

Octree representation of a 3D point cloud: An octree representation is an efficient tree-based data structure, where each branch node represents a cube volume in a 3D space, known as a voxel. Each node is sub-divided into eight children-nodes, until some stopping criteria is met, *e.g*. maximum depth or intra-resolution. Nodes that do not have children-nodes are designated by leaf nodes and all the original points are mapped onto these node. This representation allows different levels of branch depths, which leads to efficient data processing.

Maintaining an even distribution of points tracked: In particular, we are interested in maintaining information about the density of the 3D distribution of the points tracked, in order to keep an evenly balanced representation. The number of neighbours of each point is computed as

$$\rho_i = \#\text{neighbourhood}(i) = \#\left\{p_j : |\mathbf{x}_i - \mathbf{x}_j| < r\right\}. \tag{12}$$

Essentially, we count the number of points inside a sphere with radius r and center $\mathbf{x}_i$, *i.e*. the 3D position of the point of interest. This statistic allows us to have a simple, yet effective, policy to keep an even distribution of points tracked during the entire scene: if this number is greater than some threshold (*i.e*. $\rho_i > \rho_{\text{thresh}}$), it means too many points are concentrated in a given region and some should be discarded; afterwards, we search for the region with the lowest density and randomly select one point from it.

Incorporating new points: In general, when a new point starts to be tracked, no previous information is known about it, since we have no prior knowledge about its respective AVPED from previous frames $\tilde{\mathbf{d}}_i^{f-1}$, as in Eq. (2). When this is the case, we initialize $\tilde{\mathbf{d}}_i^{f-1}$ to zero and let the algorithm run as if this was the first frame for this particular point. However, a new point could be selected in a region where previously tracked points are located. In line with our second assumption (please refer to Section 3), we presume that this point will have close similarity to the new neighbouring points, especially if they belong to the same body part. Thus, we advocate that its AVPED should be initialized depending on the AVPED of its neighbouring points. In particular, we propose a weighted average as follows:

$$\tilde{\mathbf{d}}_i^{f-1} = \frac{\sum_j a_j \tilde{\mathbf{d}}_j^{f-1}}{\sum_j a_j} \quad \forall j \in \text{neighbourhood}(i), \tag{13}$$

where $a_j = \exp\left(-\alpha|\mathbf{x}_i^f - \mathbf{x}_j^f|\right)$ and $\alpha \in \mathbb{R}_+$. This means that the contribution of closer points is more significant.

Note that lost points may be seen as a special case that is naturally handled, since whenever a point being tracked is lost, we randomly select another point from a lower density region and follow the procedure described above.

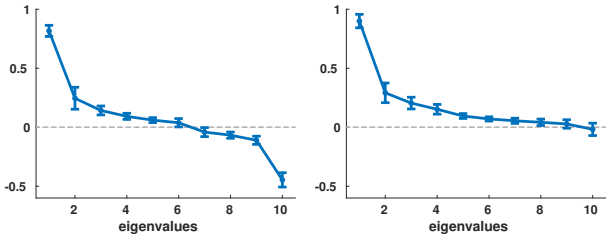
3.7. Handling noise

Noise must be a fundamental consideration when dealing with real-world scenarios, *e.g*. due to measurement's errors or imprecise point tracking, since it may significantly influence the method's overall performance. We propose to handle this issue, by observing that the original distance matrix $\tilde{\mathbf{D}}$ must be a proper Euclidean distance matrix by construction, according to Eq. (2). Consequently, $\mathbf{B}$ in Eq. (5) is positive definite, *i.e*. eigenvalues are in $[0, 1]$. However, in practice, noise may introduce non-significant directions to the representation $\mathbf{Z}$ or even make $\tilde{\mathbf{D}}$ non-metric [19]. Formally, this means that $\mathbf{B}$ will have small and/or negative eigenvalues. The latter case is easily handled, since we know $\tilde{\mathbf{D}}$ is a proper Euclidean distance matrix and consequently $\mathbf{B}$ must have non-negative eigenvalues. Therefore, we simply disregard any contribution from negative eigenvalues. The former case, however, has no immediate solution, since the significance of each eigenvalue contribution, even if small, may depend on the particular scene.

We could solely employ the spectrum transformations as described above to obtain a representation for each incoming frame. However, the effects of the noise would still be present and accumulating in the original distance matrix $\tilde{\mathbf{D}}$ possibly reaching a point where the noise contribution would be dominant. The noise effects can be reduced by incorporating the spectrum transformations into the original distance matrix. We have that [19]

$$\tilde{\mathbf{D}} \odot \tilde{\mathbf{D}} = \mathbf{b}\mathbf{1}^\mathsf{T} + \mathbf{1}\mathbf{b}^\mathsf{T} - 2\mathbf{B}, \tag{14}$$

where $\mathbf{b} \in \mathbb{R}^N$ is a vector containing the diagonal elements of $\mathbf{B}$, as defined in Eq. (5). Thus, a noiseless distance matrix, denoted as $\tilde{\mathbf{D}}_l$, can be obtained by only considering



(a) Without incorporation. (b) With incorporation.

Figure 2: Effect of incorporating the spectrum transformation into the original distance matrix: a) the large negative eigenvalue reflects the significant noise contribution; b) incorporating the spectrum transformation, the noise contribution is negligible.

the contribution of the l leading eigenvalues: first, based on Eq. (6), $\mathbf{B}_l$ is obtained, where only the l leading eigenvalues and corresponding eigenvectors are considered; then, we compute $\tilde{\mathbf{D}}_l$ according to Eq. (14), where $\mathbf{B}_l$ is used instead; finally, in the next frame, we consider $\tilde{\mathbf{D}}_l$ in Eq. (2) instead. Fig. 2 illustrates the effect of considering the spectrum transformation into the original distance matrix.

4. Experimental results

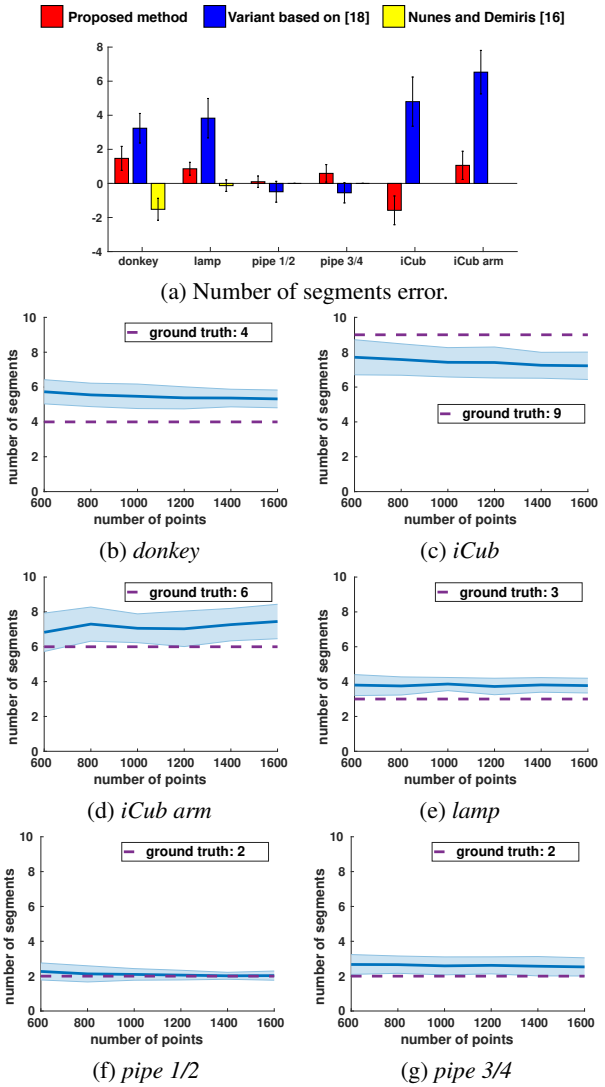
The proposed method was evaluated on the dataset provided by [22], as well as two newly recorded sequences. All experiments were performed using a PC with an Intel Core i7-8700k CPU @ 3.7Ghz (x6) and 32GB of RAM, using a C++ implementation[3]. The same parameters were used across all the experiments and were set as follows: $\mu = 0.75$, $\gamma = \frac{e^{-2}}{N^2}\sum_{i,j}\tilde{d}_{ij}$, $\alpha = 50$, $\rho_{\text{thresh}} = \left\lceil \frac{2}{N}\sum_i \rho_i \right\rceil$, r as the octree resolution, $l = 3$, $\tau = 0.025$ and $\beta = 0.275$. Also, we compare our method with the 3D subspace clustering based method [16] and a variant of our method where the similarity is built based on [18] in an online fashion. We note that since the 3D subspace clustering method [16] can not handle occlusion, we did not compare it for our newly recorded sequences, as they exhibit some occlusion. Additional results are provided in the supplementary material, which we encourage the reader to explore.

Quantitative assessment: Five metrics were evaluated: precision P_{ij}, recall R_{ij}, f-measure F_{ij}, number of segments estimated and execution time per frame. Given the segment estimated S_i and the corresponding ground-truth S_j^{GT}, the precision, recall and f-measure are given by

$$P_{ij} = \frac{|S_i \cap S_j^{\text{GT}}|}{|S_i|}, \; R_{ij} = \frac{|S_i \cap S_j^{\text{GT}}|}{|S_j^{\text{GT}}|}, \; F_{ij} = \frac{2P_{ij}R_{ij}}{P_{ij}+R_{ij}}, \tag{15}$$

which try to capture the trade-off between false positives and misses [17]. The overall evaluation of these metrics follows the description in [17], where the Hungarian method is used to find the best allocation of segments to the ground-truth and empty segments are introduced in case there are

[3]Code publicly available: www.imperial.ac.uk/personal-robotics.



(a) Number of segments error.

(b) *donkey* (c) *iCub*

(d) *iCub arm* (e) *lamp*

(f) *pipe 1/2* (g) *pipe 3/4*

Figure 3: (a) Comparison of number of segments error for 1000 points initially sub-sampled and selected randomly. (b)-(g) Number of motion segments estimated by the proposed method in function of the number of points sub-sampled.

fewer segments estimated (*i.e.* recall is zero and precision is defined as one). Each metric is evaluated for one hundred trials and for each trial the points are randomly selected.

Fig. 3 shows the results obtained for the number of segments estimated. We see that the proposed method achieves comparable performance with the offline method [16]. For some objects there is a constant bias, which may indicate that either some motions are too subtle and not detected, as seen in *iCub* sequence, or that severe point tracking errors introduce additional motions, as seen in *donkey* and *lamp* sequences. Also, the number of points considered does not significantly influence the overall number of segments estimated. This means that it is possible to sub-sample fewer points, without compromising the overall performance.

Fig. 4 presents the average cumulative computational

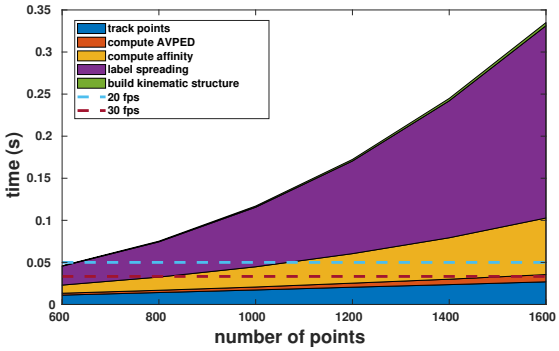


Figure 4: Cumulative computational time per frame in function of the number of points considered. Tracking points and computing the AVPED can be done in real-time considering up-to around 1400 points. Note that the time spent dealing with lost points and noise is also included, respectively.

time per frame that is required by the proposed method. We can observe that the AVPED can be maintained in real-time considering up-to around 1400 points being tracked (we also have to include the time spent tracking the points). This is the only computation required for each incoming frame, since the AVPED is responsible of maintaining information across the entire scene. The rest of computation, *i.e*. affinity, *label spreading* and kinematic structure estimation, can be done on demand and/or in parallel, not necessarily for each frame, depending on the computational resources, since they do not influence the information that is preserved by AVPED. Thus, the proposed method is suitable for real-time applications.

In Fig. 5, we show the comparison of precision, recall and f-measure metrics with other methods. These results are obtained from the last frame of each sequence. We highlight that, even though our method is online, since the estimation is always only based on current and past information, we achieve comparable results with the offline method [16] (and even outperform for the *donkey* sequence), which by design takes into consideration the entire sequence. The variant based on [18], where the similarity matrix is built based on the standard deviation of the pairwise distances, achieves the lowest performance. One reason to this might be due to the fact that this distance gradually stagnates as the number of frames increases. On the contrary, our method does not have such a limitation, since the AVPED does not stagnate if motions keep occurring. This means that motions will be segmented, regardless of when they happen, which leads to higher overall performance and not impairing the idea of online motion segmentation.

Qualitative assessment: Fig. 6 presents some qualitative results for the last frame of each sequence. We can see that the proposed method produces plausible kinematic structures compared to the ground truth, whilst being online and running in real-time. The results obtained for the variant based on [18] seem to corroborate the fact that the similarity built from the standard deviation of the pairwise distances can not accommodate for motions that appear later in the sequence (*e.g*. the head of the *donkey* is the last part to move and it is not segmented). Furthermore, the results seem to indicate that the variant based on [18] yields a noisier segmentation. It is also worth mentioning that the bias exhibited for some sequences by the proposed method in the number of motions segmented in Fig. 3 can now be better understood. For instance, the number of segments of *iCub* is consistently underestimated by around two segments; however, we can see from Fig. 6 that the three segments of the left arm are estimated as being one segment, which explains the bias. One possible reason might be that the motions exposed are not significant and/or too subtle and the method can not distinguish them. Even in these cases, the kinematic structures estimated are plausible.

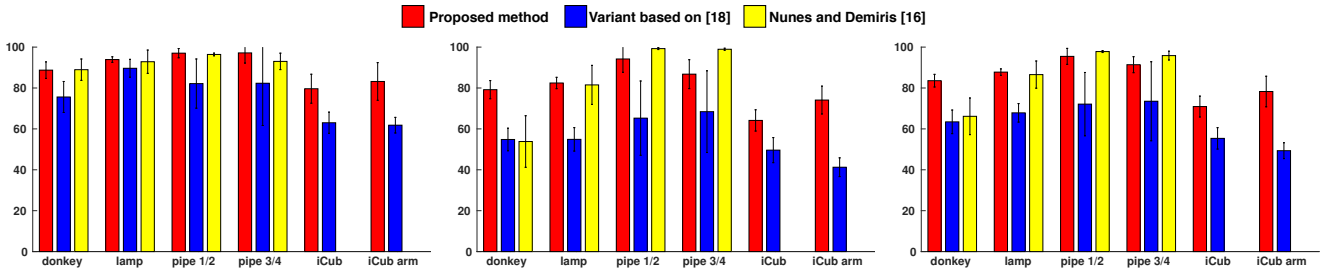
5. Conclusion and future work

An unsupervised 3D online kinematic structure learning method was proposed, which can naturally handle noise and lost points. To the best of our knowledge, this is the first method that estimates arbitrary 3D kinematic structures in an online fashion and in real-time. The experimental results show that its performance is comparable to offline methods and the kinematic structures produced are plausible and compact representations of the structure of each object. Point tracking is still an open problem in the computer vision field, *e.g*. due to large displacements and occlusions. Even though we devised an approach that mitigates the impact of lost points and noise, this is a component of our work that can still be improved. Thus, as future work, we plan on bridging this component, by not explicitly associating points between consecutive frames.

We envision our method to be applied in real-world scenarios, such as human-robot interaction. Contrary to offline methods that can only process a batch at a time, the proposed method can run continuously, which is crucial in enabling fluid human-robot interactions. For instance, enabling a humanoid robot to adaptively learn kinematic properties of surrounding objects, either by demonstration or exploration, for better object manipulation and grasping.

Acknowledgements

Urbano Miguel Nunes was supported by the Portuguese Foundation for Science and Technology under Doctoral Grant with reference SFRH/BD/130732/2017. Yiannis Demiris is supported by a Royal Academy of Engineering Chair in Emerging Technologies. This research was supported in part by EPRSC Grant EP/S032398/1. The authors thank the reviewers for their insightful feedback, and the members of the Personal Robotics Laboratory at Imperial College London for their support.



(a) Precision (%) (b) Recall (%) (c) F-Measure (%)

Figure 5: Precision, recall and f-measure metrics comparison. All results are obtained from one hundred trials, where 1000 points are initially sub-sampled and selected randomly. We do not compare *iCub* and *iCub arm* sequences with the method proposed in [16] because it cannot handle lost points and occlusions.

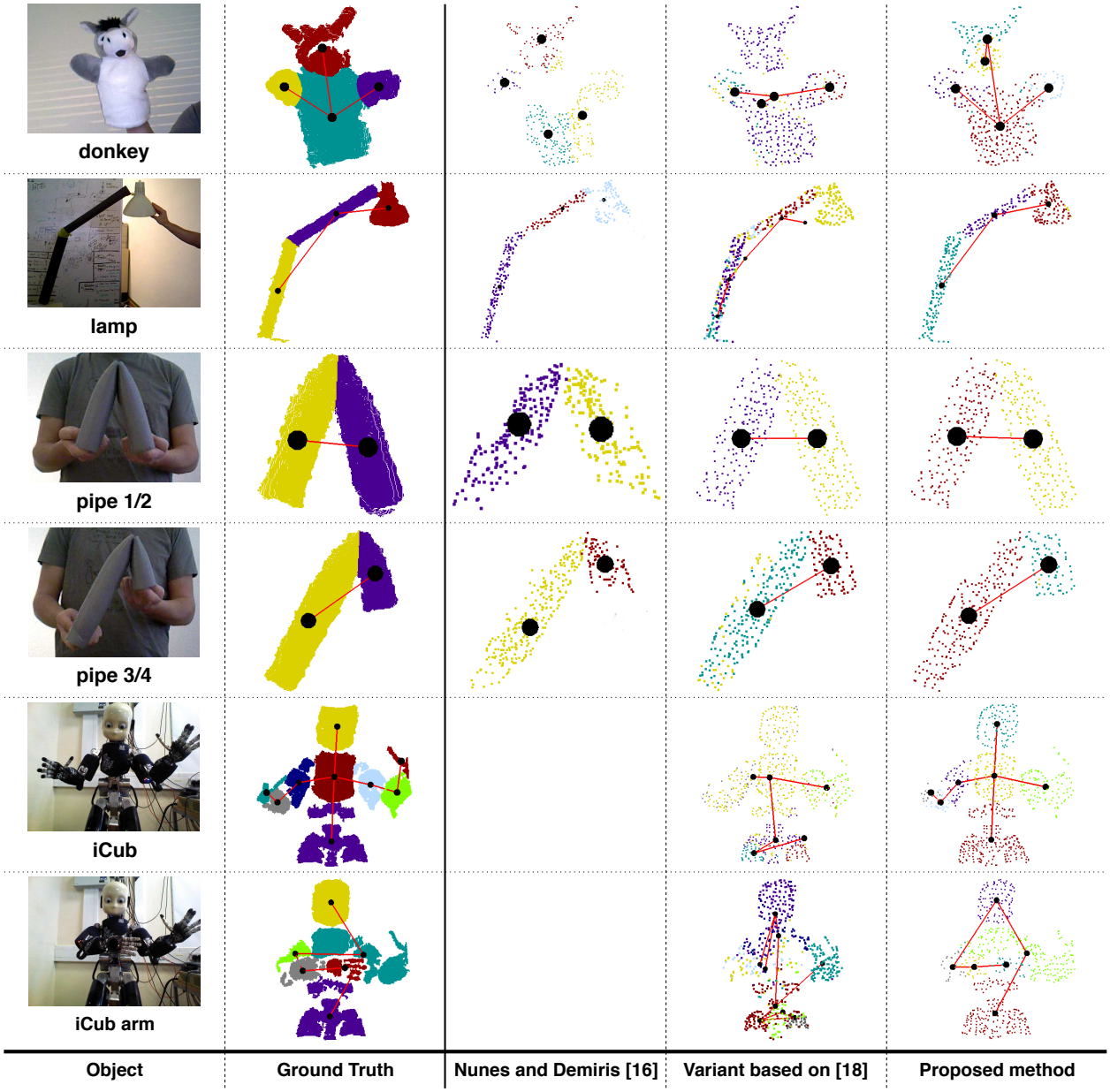


Figure 6: Qualitative results. These results shown are obtained for the last frame of each sequence. The proposed method learns plausible 3D kinematic structures of the objects. Fig. best viewed in colour.

References

[1] Antonio Agudo and Francesc Moreno-Noguer. Combining local-physical and global-statistical models for sequential deformable shape from motion. *International Journal of Computer Vision*, 122(2):371–387, 2017.

[2] 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.

[3] Jeannette Bohg, Karol Hausman, Bharath Sankaran, Oliver Brock, Danica Kragic, Stefan Schaal, and Gaurav S Sukhatme. Interactive perception: Leveraging action in perception and perception in action. *IEEE Transactions on Robotics*, 33(6):1273–1291, 2017.

[4] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 7291–7299, 2017.

[5] Hyung Jin Chang and Yiannis Demiris. Highly articulated kinematic structure estimation combining motion and skeleton information. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(9):2165–2179, 2018.

[6] Hyung Jin Chang, Tobias Fischer, Maxime Petit, Martina Zambelli, and Yiannis Demiris. Learning kinematic structure correspondences using multi-order similarities. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12):2920–2934, 2018.

[7] Ehsan Elhamifar and Rene Vidal. Sparse subspace clustering: Algorithm, theory, and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(11):2765–2781, 2013.

[8] Ali Elqursh and Ahmed Elgammal. Online motion segmentation using dynamic label propagation. In *IEEE International Conference on Computer Vision*, pages 2008–2015, 2013.

[9] Yan Huang, Jingliang Peng, C. C. Jay Kuo, and M Gopi. A generic scheme for progressive point cloud coding. *IEEE Transactions on Visualization and Computer Graphics*, 14(2):440–453, 2008.

[10] Mariano Jaimez, Mohamed Souiai, Javier Gonzalez-Jimenez, and Daniel Cremers. A primal-dual framework for real-time dense rgb-d scene flow. In *IEEE International Conference on Robotics and Automation*, pages 98–104, 2015.

[11] Kevin M. Judd, Jonathan D. Gammell, and Paul Newman. Multimotion visual odometry (mvo): Simultaneous estimation of camera and third-party motions. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3949–3956, 2018.

[12] Jungwon Kang and Myung Jin Chung. Fast online motion segmentation through multi-temporal interval motion analysis. *IEICE Transactions on Information and Systems*, 98(2):479–484, 2015.

[13] Yaron Lipman, Raif M Rustamov, and Thomas A Funkhouser. Biharmonic distance. *ACM Transactions on Graphics*, 29(3):1–11, 2010.

[14] Roberto Martín-Martín, Sebastian Höfer, and Oliver Brock. An integrated approach to visual perception of articulated objects. In *IEEE International Conference on Robotics and Automation*, pages 5091–5097, 2016.

[15] Andrew Y. Ng, Michael Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems*, pages 849–856, 2002.

[16] Urbano Miguel Nunes and Yiannis Demiris. 3d motion segmentation of articulated rigid bodies based on rgb-d data. In *Proceedings of the British Machine Vision Conference*, 2018.

[17] Peter Ochs, Jitendra Malik, and Thomas Brox. Segmentation of moving objects by long term video analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(6):1187–1200, 2014.

[18] Samunda Perera and Nick Barnes. Maximal cliques based rigid body motion segmentation with a rgb-d camera. In *Asian Conference on Computer Vision*, pages 120–133, 2013.

[19] Elżbieta Pękalska, Pavel Paclík, and Robert PW Duin. A generalized kernel approach to dissimilarity-based classification. *Journal of Machine Learning Research*, 2:175–211, 2001.

[20] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[21] Jürgen Sturm, Cyrill Stachniss, and Wolfram Burgard. A probabilistic framework for learning kinematic models of articulated objects. *Journal of Artificial Intelligence Research*, 41:477–526, 2011.

[22] Dimitrios Tzionas and Juergen Gall. Reconstructing articulated rigged models from rgb-d videos. In *European Conference on Computer Vision Workshops*, pages 620–633, 2016.

[23] Jingyu Yan and Marc Pollefeys. A factorization-based approach for articulated nonrigid shape, motion and kinematic chain recovery from video. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(5):865–877, 2008.

[24] Qing Yuan, Guiqing Li, Kai Xu, Xudong Chen, and Hui Huang. Space-time co-segmentation of articulated point cloud sequences. *Computer Graphics Forum*, 35(2):419–429, 2016.

[25] Lihi Zelnik-Manor and Pietro Perona. Self-tuning spectral clustering. In *Advances in Neural Information Processing Systems*, pages 1601–1608, 2005.

[26] Quanshi Zhang, Xuan Song, Xiaowei Shao, Ryosuke Shibasaki, and Huijing Zhao. Unsupervised skeleton extraction and motion capture from 3d deformable matching. *Neurocomputing*, 100:170–182, 2013.

[27] Yin Zhang and Zhi-Hua Zhou. Non-metric label propagation. In *21st International Joint Conference on Artificial Intelligence*, pages 1357–1362, 2009.

[28] Dengyong Zhou, Olivier Bousquet, Thomas Lal, Jason Weston, and Bernhard Schölkopf. Learning with local and global consistency. In *Advances in Neural Information Processing Systems*, pages 321–328, 2004.

[29] Vasileios Zografos, Reiner Lenz, Erik Ringaby, Michael Felsberg, and Klas Nordberg. Fast segmentation of sparse 3d point trajectories using group theoretical invariants. In *Asian Conference on Computer Vision*, pages 675–691, 2015.