SpaRTA Tracking Across Occlusions via Partitioning of 3D Clouds of Points

Andrea Cavagna, Stefania Melillo, Leonardo Parisi, and Federico Ricci-Tersenghi

Abstract—Any $3D$ tracking algorithm has to deal with occlusions: multiple targets get so close to each other that the loss of their identities becomes likely; hence, potentially affecting the very quality of the data with interrupted trajectories and identity switches. Here, we present a novel tracking method that addresses the problem of occlusions within large groups of featureless objects by means of three steps: i) it represents each target as a cloud of points in $3D$; ii) once a $3D$ cluster corresponding to an occlusion occurs, it defines a partitioning problem by introducing a cost function that uses both attractive and repulsive spatio-temporal proximity links; and iii) it minimizes the cost function through a semi-definite optimization technique specifically designed to cope with the presence of multi-minima landscapes. The algorithm is designed to work on $3D$ data regardless of the experimental method used: multicamera systems, lidars, radars, and RGB-D systems. By performing tests on public data-sets, we show that the new algorithm produces a significant improvement over the state-of-the-art tracking methods, both by reducing the number of identity switches and by increasing the accuracy of the estimated positions of the targets in real space.

Index Terms—$3D$, tracking, multi-object, occlusions, clouds of points

1 Introduction

Tracking large groups of targets in $3D$ space is a challenging topic, which is particularly relevant in the field of turbulence [1], collective animal behavior [2], [3] and social sciences [4], [5], [6] as well as in robotics [7] and autonomous mobility [8]. The technological progress of the last decades gave a boost to the development of new experimental strategies to collect $3D$ data, such as RGB–D, multicamera, lidar and radar systems. Nowadays the effort of a part of the computer vision community is directed towards finding general high-performance tracking methods.

The crucial point of all tracking algorithms is how to handle occlusions that arise every time that two or more objects get too close in $3D$ space to be detected as multiple targets. This kind of ambiguities are particularly severe when dealing with featureless objects (objects that cannot be identified by any feature such as shape or color) and with large and dense groups of targets, where the chance to get in $3D$ proximity is high. Occlusions hinder in a twofold way the quality of the retrieved trajectories: loss of one or more of the targets involved into the occlusions and a potential switch of identities.

- *A. Cavagna and S. Melillo are with Collective Behaviour in Biological Systems (CoBBS) Lab, National Research Council - Institute for Complex Systems (CNR–ISC), UOS Sapienza, Rome 00185, Italy. E-mail: andrea.cavagna@roma1.infn.it, stefania.melillo79@gmail.com.*
- *L. Parisi is with Collective Behaviour in Biological Systems (CoBBS) Lab, National Research Council - Institute for Complex Systems (CNR–ISC), UOS Sapienza, Rome, Italy, and also with the Department of Mechanical and Aerospace Engineering (DIMA), Sapienza University of Rome, Rome 00185, Italy. E-mail: leonardo.parisi@gmail.com.*
- *F. Ricci-Tersenghi is with the Physics Department, Sapienza University of Rome, Rome 00185, Italy. E-mail: federico.ricci@uniroma1.it.*

Manuscript received 29 Oct. 2018; revised 24 Sept. 2019; accepted 7 Oct. 2019. Date of publication 4 Nov. 2019; date of current version 4 Mar. 2021. (Corresponding author: Leonardo Parisi).
Recommended for acceptance by M. Betke.
Digital Object Identifier no. 10.1109/TPAMI.2019.2946796

In this paper we propose a novel tracking method called SpaRTA (Spatiotemporal Reconstruction Tracking Algorithm), which is able to solve $3D$ occlusions identifying each target during the occlusions and producing negligible switches of identities. SpaRTA is meant to work on objects detected as $3D$ clouds of points, regardless of the system used to collect the data. The core ideas of the methods are the following: i) SpaRTA reconstructs the $(3D+1)$ spatio-temporal volume (where $3D$ is the spatial dimension and $(+1)$ represents the time dimension) occupied by each target during the entire acquisition as a cloud of points; ii) when an occlusion arises, SpaRTA tackles the problem of splitting it into different objects by defining a partitioning problem that uses both attractive and repulsive links depending on the distance in space and time among the points belonging to the occlusion; iii) as the superposition of attractive and repulsive links gives rise to frustration, namely to the emergence of many local minima of the partitioning cost function, SpaRTA uses an optimization method inspired on Semi-Definite Programming (SDP) techniques developed in the context of statistical physics of disordered systems [9], to find the optimal partition (i.e., ground state of the cost function), thus finally splitting the occlusion into the actual different targets composing it.[1]

SpaRTA was tested on data of large groups of animals collected in the field with a multicamera system. This kind of data are a good benchmark for $3D$ tracking methods because large groups of animals are particularly hard to track: data are characterized by frequent occlusions, lasting several frames, and by a low spatial resolution such that targets appear as

1. For the sake of easy reading and without losing any generality, throughout the paper we will refer to occlusions to denote the geometric event of two targets getting in $3D$ proximity, but also to denote the cloud–of–points representing the $(3D+1)$ volume occupied by targets in $3D$ proximity.

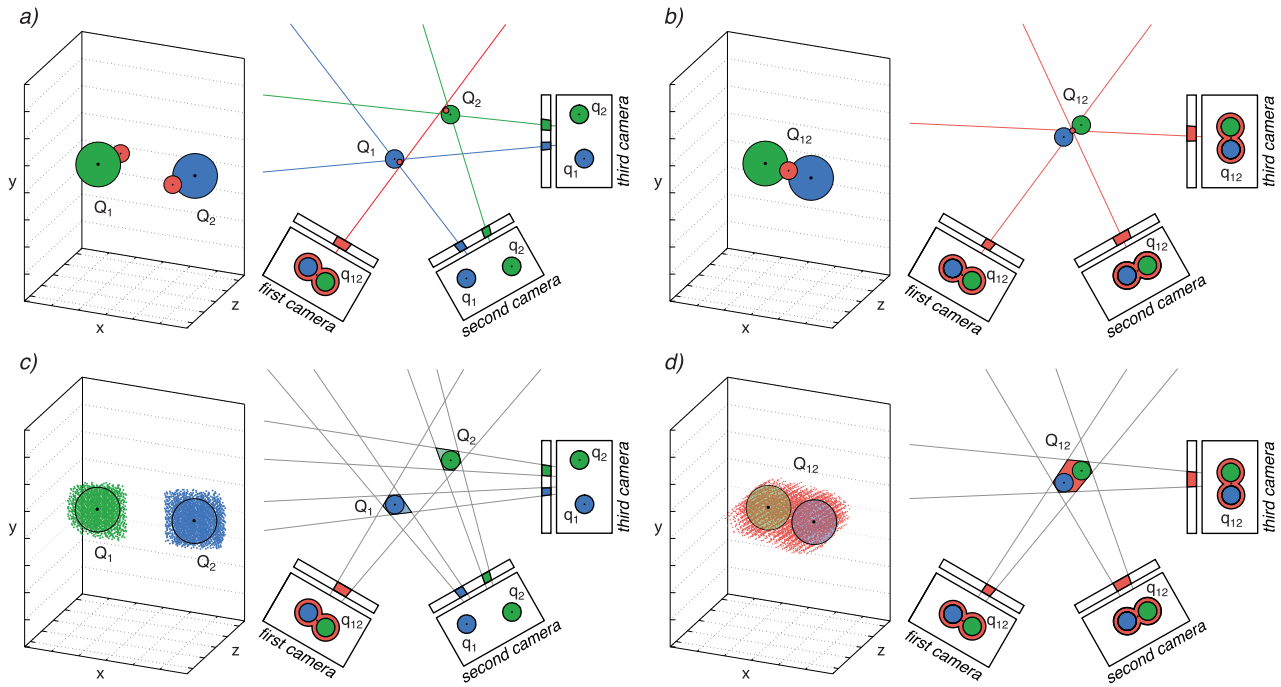

Fig. 1. *Optical occlusions: SIP versus COP representation.* Left column: $2D$ occlusions. The blue and the green targets are occluded in the image plane of the first camera and well–separated in the other two. Right column: $3D$ occlusion. The blue and the green targets are in $3D$ proximity, therefore they are occluded in all the three cameras. Top row: SIP representation. a) The information from the two cameras where the occlusion do not occur make the occluded targets to be reconstructed in two different positions (the red circles in the right graph of the panel), but with a poor accuracy: both the two $3D$ positions do not correspond to the target centers of mass. b) The two occluded targets are associated with one single $3D$ position (the red circle between the blue and the green target), with a consequent loss of the targets identities and potential switch of identities after the occlusion occurs. Bottom row: COP representation. c) The occluded targets are represented as two well-separated dense cloud of $3D$ points. d) The occluded targets are reconstructed in a single cloud of $3D$ points as big as their total volume. Identities may be retrieved by splitting the clusters in the two subsets representing the volumes of the two distinct objects, as SpaRTA does using the SDP technique described in Section 4.2.

objects without any recognizable feature. The only limitation of these data is that the production of ground truth trajectories to evaluate the tracking result is quite difficult and time–consuming, and it is then hard to give a quantitative evaluation of the quality of the resulting set of trajectories. This is the reason why there are very few public data-sets that are usable as benchmarks. To the best of our knowledge, the only available two public data-sets of featureless objects collected via a multicamera system are published by Wu et al. in [10]. We tested SpaRTA on these data-sets showing the high performance of the proposed algorithm in terms of the quality of the retrieved trajectories.

2 Related Works

Since the seminal work of Reid [11], several $3D$ tracking strategies have been proposed in the past forty years. However, despite this strong effort, only few methods are designed to track large and dense systems of featureless objects and the research on this topic is still very much ongoing, especially for what concerns the solution of occlusions.

There are two fundamentally different ways to represent the targets an algorithm wants to track: on one hand, we can associate to each target at a certain instant of time one single spatial position (typically the center of mass) – we will call this case Single Point (SIP) representation; on the other hand, we can associate to a target a dense Cloud of Points (COP), representing the full spatial volume of the target at that instant of time.

The SIP representation is typically adopted within the context of multicamera data-taking systems, in which sets of $2D$ single objects positions have to be turned into $3D$ positions and trajectories. One way to achieve this is to first track the objects in each camera and then match the $2D$ trajectories across cameras to retrieve the corresponding $3D$ trajectories (the so-called Tracking-Reconstruction (TR) route). Working in the $2D$ space of the cameras, TR algorithms have to deal with two different kind of optical occlusions: 2d–occlusions which arise when the projections of two or more objects become spatially close on the $2D$ image plane but the objects themselves are not close in the $3D$ space and 3d–occlusions which arise when multiple objects are occluded in all the cameras simultaneously, namely when multiple objects get in $3D$ proximity (see Fig. 1, top row). Both $2D$ and $3D$ occlusions produce bifurcations of the $2D$ trajectories, and hence a high intrinsic complexity due to the proliferation of the $2D$ trajectories to be matched across the cameras. Several different strategies to prune the set of $2D$ trajectories and reduce the complexity of this approach have been devised [2], [12], [13], [14], [15] and [16].

Conversely, yet still within the SIP representation, one can first reconstruct single objects turning them into $3D$ positions (by matching their identities across the cameras), and then track them in $3D$ space (the so-called Reconstruction-Tracking (RT) route). Working directly in the $3D$ space, RT methods are not affected by $2D$ occlusions that are naturally solved when matching objects across the cameras, hence their complexity is naturally quite lower than the TR

methods; however, RT strategies are typically more prone to creating false $3D$ objects. The SIP-RT approach has been explored in small groups of objects [17], [18], [19], [3], and it is not clear how these approaches perform in case of dense groups, while the most advanced SIP–RT method has been proposed in [20] and it can successfully track large groups.

The SIP representation has some drawbacks, especially in dense systems, where occlusions are frequent: the assumption behind this representation is that the $3D$ center of mass of an object corresponds to the $2D$ centers of mass of its images, which is reasonable when dealing with not–occluded targets but it fails when $2D$ and $3D$ occlusions occur, as shown in Fig. 1. More specifically, whenever two or more objects are part of a single occlusion, the SIP method associates one single position to all of them, causing: i) loss of the actual targets individual identities; ii) potential identity switches after the occlusion; iii) an inaccurate positioning of the targets in $3D$ space (see Fig. 1).

The COP representation, on the other hand, allows to associate to each object (at each instant of time) a dense cloud of $3D$ points and not only its center of mass. This dense representation reconstructs the actual volumes occupied by the detected objects: $2D$ occlusions are naturally solved, while targets in a $3D$ occlusions are associated to only one cloud of points, see Fig. 1, that may be divided into separated sub–parts each representing the volume of a single target using graph partitioning techniques. Discarding no information about the actual targets volumes, and therefore creating no simplified identities, COP representation is more suited to prevent identity switches (problems i) and ii) above) and definitely more accurate to locate the $3D$ positions of the targets. Besides, COP-based tracking (unlike SIP) is not forcibly embedded within a multicamera framework and it is therefore a significantly more general approach than SIP: indeed, COP tracking has been used in RGB–D systems [21], lidar [22] and radar [23]. In the COP context, occlusions have been tackled by different techniques, [24], [25], [26], [27], which, however are designed for the specific nature of the data to be tracked. Here, we will introduce a novel COP-based tracking method designed to be as general as possible.

3 Overview of the Method

SpaRTA works with $(3D + 1)$ (space + time) clouds of points representing the $3D$ volume occupied in time by a group of moving objects, without any limitation on the $3D$ system used to collect the data.[2] The goal of the algorithm is to partition the $(3D + 1)$ cloud of points in $(3D + 1)$ sub–clouds, each corresponding to the trajectory of a single target.

SpaRTA can be broken into the following steps:

1 – Building the graph. The cloud of $3D$ points is first clustered in space at a static level (fixed instant of time): a clustering algorithm based on the $3D$ nearest neighbor distance [29] is used to detect the well–separated dense cloud of points (clusters), which may represent the detected objects at each instant of time. These reconstructed $3D$ clusters are then connected in time through a dynamical linking procedure, see Section 4.1 and Appendix A, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPAMI.2019.2946796 for further details. In this way, we create a set of $(3D + 1)$ clouds of points representing the volumes occupied by the objects during the event, actually building the graph shown in Fig. 2 with $3D$ clusters as nodes and links as edges.

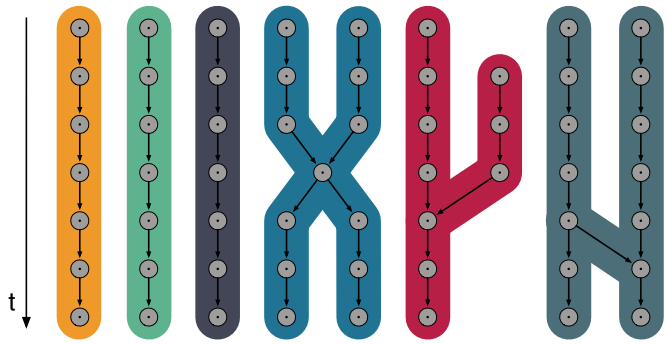

Fig. 2. *Clusters graph.* Gray circles represent $3D$ clouds of points dynamically linked via black arrows. Distinct connected components are highlighted in different colors. The first three components from the left represent trajectories of not–occluded targets, since they are made of only one–to–one linked clusters. Conversely the last three components are ambiguous because they have at least one node with more than one link from the past or to the future.

2 – Tackling the occlusions. A breadth–first search routine [30] is used to identify the connected components of the clusters graph, which should represent the trajectories of the detected targets. In the ideal situation where $3D$ occlusions do not occur, each connected component is made of one single target at each instant of time, therefore it is made only of one–to–one linked nodes, see Fig. 2. However, in the more realistic situation where $3D$ occlusions do occur, two or more objects may belong to the same connected component, sharing one or more nodes, as in the last three cases in Fig. 2. These connected components, with at least one multi-link, are due to occlusions, and they must be solved. The philosophy of SpaRTA is to break up the ambiguous connected components into different partitions, each corresponding to the trajectory of a single actual target, by defining and solving an optimization problem.

To this aim, points belonging to an ambiguous connected component are linked in space and in time: the crucial idea is to use *attractive* (positive) links connecting points that are close to each other and thus with a high probability to belong to the same target, and at the same time to penalize, with *repulsive* (negative) links, pairs of points that are too far from each other when compared with intrinsic space-time scales of the data. Once the graph is built, SpaRTA defines a cost function given by the negative sum of all links in each candidate partition, in such a way that the global minimum of this function corresponds to the optimal partitioning of the occlusion into *bona fide* $3D$ targets. The presence of both attractive and repulsive links is crucially functional in associating the correct partition to the actual targets; on the other hand, using competing links is known to increase steeply the complexity of an optimization method, by creating a proliferation of sub-optimal solutions (local minima) [31]. To deal with this problem, SpaRTA uses an optimization routine inspired by Semi-Definite Programming techniques that are

2. Note that when using a multicamera acquisition method, data are not directly obtained as $3D$ clouds of points (unlike data acquired with RGB-D, lidar or radar systems); hence, in that specific case, targets images are converted into $3D$ clouds of points through a preprocessing procedure as the one described in [28] and in Section 5.

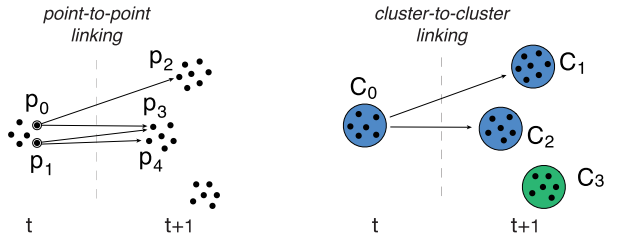

Fig. 3. *Cluster graph construction: Dynamic linking.* At a generic frame t, a point–to–point multi–linking procedure is performed: p_0 at time t is connected to p_2 and p_3 at frame $(t+1)$, while p_1 at frame t is connected with p_3 and p_4. These point–to–point links are then used to define cluster–to–cluster links: two clusters are connected if there exists at least one point–to–point link between points belonging to the two clusters. Therefore, C_0 will be linked to both C_1 and C_2 (the two points p_0 and p_1 belong to C_0 and they are both linked to points belonging to C_1 and C_2). On the opposite, C_3 does not receive any link from the past, because none of its points receive a point–to–point link.

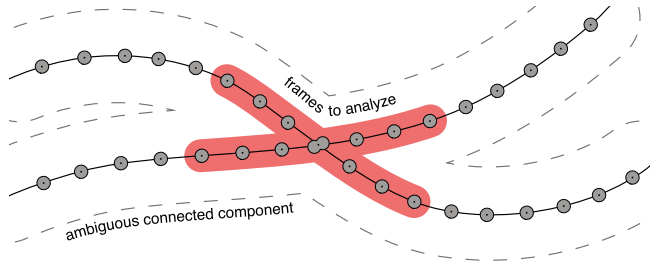

Fig. 4. *3D occlusion.* A X–shape connected component representing a $3D$ occlusion. The two objects are well–separated for the most part of the event, but they share the same cluster during the occlusion. The four branches of the X represent the two different trajectories of the two objects before and after the occlusion, which is represented as a double grey circle at the center of the X. During the SDP procedure the analysis of the ambiguous component is restricted to a quite short interval, highlighted in pink.

known to work efficiently in disordered systems whose complexity is severe [9], [32]. See Section 4.2 for details.

3 – Identifying 3D trajectories. Each non–ambiguous connected component identified in Step 2 above, represents the $3D$ volume occupied by a single target during the dynamics of the system. However, for many practical purposes it is not convenient to work with $3D$ volumes, which may be hard to be handled, and it is more desirable to associate a single $3D$ position to each object at each instant of time. Thus, we associate to each cluster its $3D$ center of mass position, i.e., average $3D$ coordinates of the cluster points, and we define the trajectories as the time sequence of the center of mass coordinates. Notice that this is *not* the same as defining object positions through their center of mass in the SIP framework, because here all occlusions have been already solved, hence the center of mass is indeed a appropriate representation to locate the target position; on the contrary, centers of mass fail in SIP whenever one object corresponds to several targets into an occlusion, as shown in Fig. 1.

4 Method Details

In this section we describe in detail how we handle the $(3D+1)$ cloud of points to build the graphs and how we solve $3D$ occlusions.

4.1 Building the Graph

The $(3D+1)$ cloud of points is first analyzed at a static level to identify well–separated clusters, which may represent single objects or multi-objects during an occlusion. To this aim, we use a standard clustering algorithm based on the $3D$ nearest neighbor distance [29]: two reconstructed $3D$ points, Q_1 and Q_2 belong to the same cluster, C, if their $3D$ distance $d(Q_1, Q_2)$ is smaller than r_1, with r_1 equal to the median of the targets nearest neighbor distance.[3]

3. To compute r_1 we should measure the distance between each reconstructed point and its first neighbor, that has a computational complexity of $O(M^2)$, with M being the average number of $3D$ points at each frame. We lower this computational complexity to $O(M)$ using the space-partitioning technique of [33]. To further reduce the computational complexity, we do not estimate r_1 on the entire set of points, but on a randomly chosen sub–set (~ 20 percent of the total reconstructed points), that we proved to be large enough to give an acceptable approximation of the median nearest neighbor distance computed on the entire set.

Once all the $3D$ clusters of points are created at each instant of time, we need to dynamically link points at subsequent instants of time, actually building the $(3D+1)$ graph. We have to be careful doing this, because missing dynamical links may result in fragmented trajectories, while extra–links increase the connectivity of the graph, creating false occlusions and making the problem difficult to be solved. We define point–to–point dynamical links using a dynamical proximity method whose only assumption is that each $3D$ point moves with a constant velocity between two consecutive instants of time, see Fig. 3. Note that the constant velocity assumption is reasonable when working on data collected at a high frame–rate, as the ones used to test SpaRTA, but it may need some refinements in a more general scenario. Once we have built point–to–point dynamical links, we use them to define cluster–to–cluster dynamical links: two clusters C_1 and C_2 are connected in time if there exists at least one point–to–point link between a point $p_1 \in C_1$ and a point $p_2 \in C_2$, see Fig. 3. For the sake of clarity, we omit here to describe all the details of the temporal linking procedure and we refer the interested reader to Appendix A, available in the online supplemental material.

4.2 Tackling the Occlusions

In this section we analyze in detail how we solve the ambiguous connected components, i.e., the occlusions. Here is the core of the method, which overcomes the occlusions yet keeping the identities of the objects involved.

We restrict our attention only to connected components made of two objects in a $3D$ occlusion for one or more frames. The more general case of more than two trajectories belonging to the same connected components can be reduced to the simplest one solving each $3D$ occlusion in a restricted frame range such that only two objects at the time are involved (under the mild assumption that no more than two objects can be involved in the same $3D$ occlusion at the same time).

In an ambiguous connected component, the trajectories of the objects (involved in a $3D$ occlusion) are well–separated for the most part of the event, sharing only few clusters, just during the occlusion. Therefore an ambiguous connected component due to a $3D$ occlusion has the X–shape shown in Fig. 4, with the 4 branches of the X representing the trajectories of the two objects before and after the occlusion, which is instead the centre of the X. The two

occluded objects are not distinguishable and they are detected as one cluster only. The goal of this step is then to identify and separate the volumes occupied by the two objects during the occlusion, i.e. to split the clusters in the two subsets representing the volumes of the two distinct objects.

To handle this situation we switch back from clusters to $3D$ cloud of points, representing the ambiguous component as a graph with its $3D$ points as nodes connected by links carrying *static* (equal time) and *dynamic* information (consecutive frames). Following the literature about graph bi–partitioning techniques [34], we address the partitioning as an energy minimization problem. Therefore, we associate an energy, H, defined as follows:

$$H = -\sum_{i,j} w_{ij} x_i x_j, \tag{1}$$

where i and j are two different points of the graph, $x_i = \pm 1$ identifies to which partition the point i belongs, and w_{ij} is the coefficient associated with the pair of points i and j: because we want to minimize H, and it has a minus sign in front of the sum, the key to solve the problem is that we must use a *positive* coefficient w_{ij} when it is highly likely that i and j belong to the same partition, i.e., i and j are at a short $3D$ distance, but it is also essential to assign a *negative* coefficient when it is likely that i and j belong to different partitions, i.e., i and j are too far from each other compared with the space–time scales of the data. Clearly, a sensible definition of w_{ij} is of paramount importance and some heuristics is inevitable in the choice. Despite this, there is a general principle to follow: $|w_{ij}|$ has to be large when i and j are likely to belong to the same partition (large and positive w_{ij}) or when i and j are likely to belong to distinct partitions (large and negative w_{ij}) and it is reasonable to have a zero value of w_{ij} whenever it is unclear what is the likely fate of i and j. To implement this scenario, we use the simplest rule, which amounts to link the coefficients w_{ij} to the spatio-temporal proximity of the points i and j.

4.2.1 Static Coefficients

We first define the coefficients between points at the same instant of time. We *statically* connect two points, i and j, at a mutual distance d_{ij}, with the following weight:

$$w_{ij}(t) = e^{-(d_{ij}/r_1)^{\beta}} - \left(\frac{d_{ij} - r_0}{r_1}\right)^2 \theta(d_{ij} - r_0) \tag{2}$$

r_1 is the median nearest neighbor distance, which is also the only natural unit of length of the system of points, hence all other lengths will be measured in units of r_1 to make all coefficients dimensionless; r_0, on the other hand, is the median size of the reconstructed clusters at that specific instant of time, $r_0 \gg r_1$; $\theta(x)$ is the Heaviside function. The actual shape of w_{ij} in Eq. (2) as a function of the space distance d_{ij} is depicted in Fig. 5: the idea is to strongly and positively connect those points which are likely to belong to the same cluster, i.e., with a mutual distance smaller than r_1, and to negatively connect those points which are likely to belong to different clusters, i.e., at a mutual distance bigger than the usual size of the objects; finally, all points with an uncertain distance, namely $r_1 < d_{ij} < r_0$, have a non-committal $w_{ij} \sim 0$. Hence, the exponent β simply rules how sharp is the elbow around r_1, and its value does not impact significantly on the results as long as $\beta \geq 2$ (we use $\beta = 2.2$); note, though, that any other sharp decay would do the job.

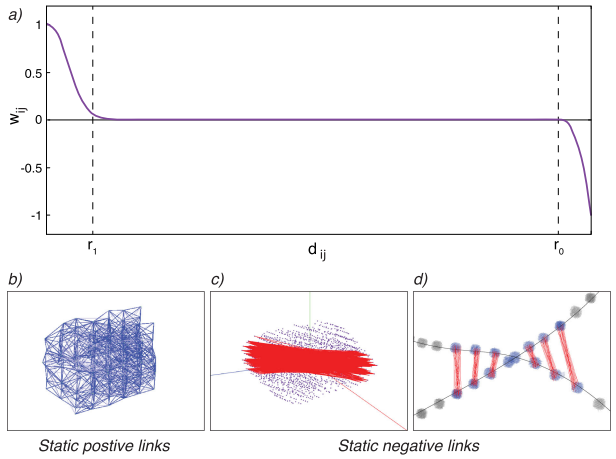

Fig. 5. *3D occlusion: static linking.* a) The static coefficient, w_{ij} between two points i and j as a function of their mutual distance d_{ij}, with r_0 and r_1 of the specific $3D$ occlusion shown in Fig. 6. b), (c), and d) Points belonging to the same frame are strongly connected if they are at a very short mutual distance (*static* positive links), while they are strongly disjointed, through a large but negative weight, when at a large mutual distance, both within the same target (*static* negative links) or in two different targets (*static* negative links between clusters).

4.2.2 Dynamic Coefficients

Using an identical philosophy, we *dynamically* connect two points, i at time t and j at time $t+1$, with the following weight:

$$w_{ij}(t, t+1) = e^{-D_{ij}/r_1}, \tag{3}$$

where,

$$D_{ij}(t, t+1) = |\vec{r}_i(t) + \Delta\vec{r}_i(t, t+1) - \vec{r}_j(t+1)| \tag{4}$$

is the distance between the extrapolated position of i at time $(t+1)$, namely $\vec{r}_i(t) + \Delta\vec{r}_i(t, t+1)$, and the position of point j at time $t+1$. The displacement $\Delta\vec{r}_i(t, t+1)$ is linearly extrapolated from past frames under the assumption that each point moves at a constant velocity between two consecutive frames. Notice that once again we have used the median nearest neighbour distance r_1 as the natural length scale of the system to make the coefficient dimensionless. The meaning of these links is to strongly connect, in time, those pairs of points belonging to the same dynamic cluster, namely the points belonging to the same branch of the ambiguous X–shape (see Figs. 4 and 6), which are likely to belong to the same partition.

4.2.3 Graph Partitioning

In order to find the partitioning which minimizes the energy H in Eq. 1 one could use standard approaches, such as integer linear programming or Montecarlo techniques; these, however, are known to fail to find the correct solution (ground state) when there are many local minima of H, which is the case of the present problem. Therefore, we choose to approach the problem by using a more robust algorithm based on Semi-Definite Programming, whose details can be found in [9], and

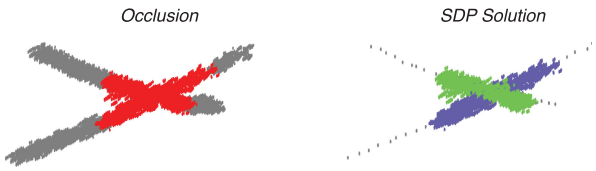

Fig. 6. *3D occlusion solution.* An example of a $3D$ occlusion from the *Davis-08 dense* dataset [10] used to test the method. On the left: the X–shape component, with the $3D$ occlusion and the sub–cloud analyzed with the SDP technique highlighted in red. On the right: the same component after the solution of the $3D$ occlusion, with the two separated trajectories highlighted in blue and green.

which successfully finds the absolute minimum of complex energy functions even in presence of multi-minima landscapes.

Once the $3D$ occlusion is solved through the SDP procedure, the ambiguous X–shaped component is divided into two partitions, connected both in space and in time, as shown in Fig. 6. The occluded potential clusters are actually split into two sub–clusters representing the volumes occupied by the two objects during the occlusion. In this way, we successfully solve the occlusion retrieving the two trajectories and maintaining the identities during the entire event. Ideally the optimization should be applied to the entire X–shape component, but this would require high computational resources and high computational time, since the minimization of the energy H in Eq.(1) is an NP–hard problem, see [35]. Hence, in order to reduce the number of variables of the problem, we restrict the optimization to a shorter interval of time, namely from few frames (3 in the specific case of the tests presented in Section 5) before the occlusion starts, to few frames after the occlusion ends, as shown in Figs. 4 and 6 (left panel) where the interval to be analyzed is highlighted in red.

Note that by restricting the optimization to a short interval of time around the occlusion, we have the advantage of successfully applying the method to connected components made of more than two targets, as the one shown in Fig. 7. Multiple connected components are generally due to occlusions occurring at different instants of time but sharing at least one target, see Fig. 7 where target A is in $3D$ occlusion with target B at frame t, and with target C at frame $t' > t$. We approach this situation solving each $3D$ occlusion in a short interval and than we identify the connected components, each corresponding to a single target (see Appendix B, available in the online supplemental material where we investigate the effect of the time interval length on the quality results of the method). Note that the results presented in Section 5 are obtained restricting the optimization to the interval of time going from 10 frames before the occlusion starts to 10 frames after the occlusion ends.

5 Contribution

The architecture of SpaRTA is based on three essential elements: i) the COP-representation; ii) the interplay between *attractive* (positive) and *repulsive* links in the graph of points to be partitioned; iii) the use of a partitioning technique SDP-based.

These three elements are not only the core of the method, but they also represent the main contributions of SpaRTA in the field of $3D$ tracking. The COP–representation is widely spread for tracking vehicles and persons, where targets are characterized by recognizable features such as color or shape,

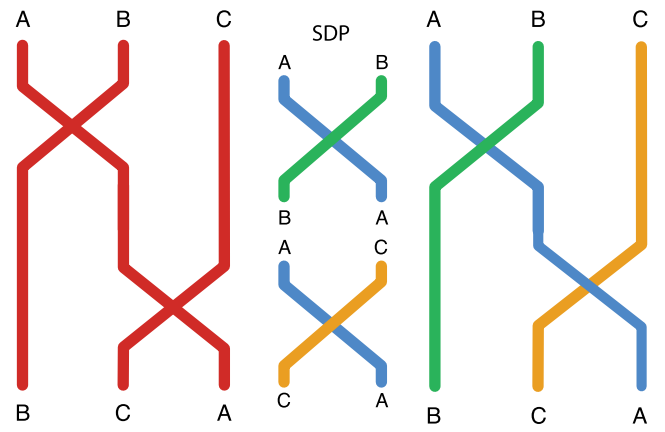

Fig. 7. *Multiple connected component.* The three targets A, B, and C are involved in the same connected component: target A is occluded at an instant of time t with B and at a different instant of time $t' > t$ with C. We identify the two occlusions and we define a minimization problems in a short interval around each occlusion. Once the occlusions are solved, we look for the connected components involved in the ambiguous original one and we find the three components (the blue, the green, and the orange) corresponding to the three different objects.

while it is not used for featureless objects where all the state–of–the–art methods (as the ones in Table 1) are based on the SIP–representation. SpaRTA introduces the representation of the targets as $3D$ clouds of points for featureless objects, therefore allowing the identification of multiple targets through an occlusion, unlike the SIP–representation where occluded objects correspond to a single $3D$ entity, see Appendix C, available in the online supplemental material for further details.

The latter two elements above, interplay of positive and negative links and use of SDP–based technique, are much more general and they represent a novelty in the field of tracking (of objects with and without features). Tracking is naturally addressed as a partitioning problem on the graph of the reconstructed objects linked in space and time, where the goal is to find the sub–parts of the graph that share some particular property, i.e., belong to the same target. To this aim, each link is associated to a coefficient that quantifies the rate of similarity between the points it connects (the higher the similarity the smaller the coefficient) and the sub–graphs are found as the ones that minimize an energy function, linear with respect of the weights of the links. In SpaRTA we propose to measure not only the similarity between two points but also their dissimilarity, using coefficients of opposite signs, and to solve the minimization problem with SDP–based techniques, since usual partitioning techniques do not work anymore due to the non–convexity of the energy function. The different sign of the coefficients push the optimization problem to find the solution with similar points in the same partition, and dissimilar points in different partitions, hence imposing harder constraints to the problem. The generality of this approach relies on the generality of the concept of similarity between points: in the particular case of featureless objects, as the one addressed by SpaRTA, the weights of the links can only be related to the distance between the connected points, because the distance in space and time is the only available information, but in a more general case, similarity and dissimilarity may be measured using information on the object features, such as color or shape, and therefore making this approach applicable to a wide class of tracking problems.

TABLE 1
Comparison of the Quality of the Trajectories Retrieved by SpaRTA and the Algorithms: MHT, SDD-MHT [10], CP(LDQD) [2], and GReTA [12] on the Public Datasets Labeled *Davis-08 sparse* and *Davis-08 dense* Published in [10]

Dataset	Algorithm	Class	MOTA (%)	IDS (#)	MT (%)	ML (%)	FM (#)
Davis–08 sparse	**SpaRTA**	**COP-RT**	**91.7**	**12**	**96.2**	**1.0**	**185**
	MHT	SIP-RT	64.1	97	96.6	0	105
	SDD-MHT	SIP-RT	78.9	126	95.2	0	145
	CP(LDQD)	SIP-RT	88.1	126	97.1	0	115
	GReTA*	SIP-TR	83.1	9	85.1	1.9	167
Davis–08 dense	**SpaRTA**	**COP-RT**	**90.4**	**11**	**91.1**	**5.4**	**123**
	MHT	SIP-RT	−32.0	355	71.9	2.5	274
	SDD-MHT	SIP-RT	44.9	444	61.1	3.0	454
	CP(LDQD)	SIP-RT	80.5	156	84.2	0.5	176
	GReTA*	SIP-TR	79.4	7	80.3	3.9	358

In the table, we report the MOTA (Multiple Object Tracking Accuracy) and also, the number of switches of identities (IDS), the percentage of mostly tracked (MT) and mostly lost (ML) trajectories corresponding to groundtruth trajectories which are correctly reconstructed respectively for more than the 80 percent and for less than the 20 percent of their time length, the number of tracks fragments (FM) corresponding to the number of times that a groundtruth trajectory, correctly reconstructed, is interrupted. A perfect tracking algorithm produces $MOTA = 100\%$, $IDS = 0$, $MT = 100\%$, $ML = 0\%$, *and* $FM = 0$. *In order to compute a match between groundtruth and reconstructed trajectories, we chose a miss/hit threshold equal to 0.3m as suggested in [10]. GReTA*: The results obtained by GReTA presented in this table are not the ones published in [12]. This is because, performing the quality evaluation of our new algorithm SpaRTA, we found a shift of one frame in the annotated file of the dataset published in [10]. We evaluated SpaRTA using the annotated file, but taking care of the time shift and for coherency we also updated the results obtained by GReTA.*

6 Experiments and Discussion

We tested the method on two public datasets, published by Wu et al. in [10], of Brazilian bats colonies emerging from a natural cave in Texas, acquired with a system of three synchronized high–speed cameras. To the best of our knowledge these are the only public $3D$ datasets of featureless objects, but they are not in the form of $(3D+1)$ cloud of points. Therefore SpaRTA cannot be directly applied to the data, but a pre–processing procedure on the images is needed. In the next two paragraphs we will give a detailed description of the pre–processing procedure and of the refinements on the method that we used to work on multicamera data, while the last two paragraphs of this section are devoted to the description of the evaluation metric used in the tests and to the discussion of the results obtained with SpaRTA and compared with 4 state–of–the–art algorithms.

6.1 From Stereo–Images to *(3D+1)* Cloud of Points

For the sake of simplicity, in the following we will refer to data collected with a system of three cameras as for the public datasets used to test SpaRTA, but the entire procedure may be easily generalized to any multicamera system.

We obtain the $(3D+1)$ cloud of points from the images performing the following steps:

1 – Segmentation. We identify the *active* pixels in the images, i.e. pixels representing the objects to be tracked, using a segmentation routine based on standard background subtraction over a sliding window (as suggested by Sobral and Vacavant in [36]), because the datasets we used to test SpaRTA consist of monochromatic images of white objects (bats) moving over a still and dark background (the sky). The segmentation routine is a crucial step that could heavily affect the results of the tracking algorithm since miss–detected as well as false detected objects may lead to fragmented or, even worst, to *ghost* trajectories, i.e., not corresponding to any real object. Despite this, we prefer to keep the segmentation routine as simple and general as possible, so that it may be applied to different systems of featureless objects with only minor modifications.

2 – Pixels matching. At each instant of time, the sets of *active* pixels, P_1, P_2 and P_3 (one for each camera) identified at the previous step, are analyzed to find the triplets $\tau = (p_1, p_2, p_3)$ of $2D$ points (pixels) $p_i \in P_i$, projections of the same $3D$ point in the three cameras. A triplet τ is considered a good match if its reprojection error, see [37], is smaller than few pixels. In principle we should reconstruct and check all the possible N^3 triplets, with N being the average number of *active* pixels in each camera. However, using the trifocal constraint [37] the number of triplets to be checked may be reduced to $O(N)$.

3 – (3D+1) cloud of points creation. All the triplets τ created at the previous step are reconstructed in the $3D$ space, using a standard DLT reconstruction procedure [37], thus forming a $(3D+1)$ cloud of points which represents the $3D$ volume occupied by all the targets during the entire event of interest.

6.2 *Ghost* Points Formation and Trajectories Removal

The pixel matching procedure described in Section 5.1 may lead to the formation of *ghost* $3D$ points, representing points not belonging to any *real* $3D$ object. *Ghost* points are a well–known artifact, as shown by Theriault et al. in [38], intrinsically due to the method that, at this level, cannot discriminate between *ghost* and correct points, as shown in Fig. 8. Therefore *ghost* $3D$ points are included in the overall $(3D+1)$ cloud of points, creating *ghost* clusters and hence *ghost* trajectories, which has to be identified and removed to not affect the quality of the retrieved solution.

For their nature, *ghost* trajectories last for very few frames, so that not ambiguous *ghost* connected components can be easily identified and eliminated removing all the trajectories shorter than a certain time length (see Appendix D, available in the online supplemental material where we investigate the effect of the threshold on the trajectory length on the quality

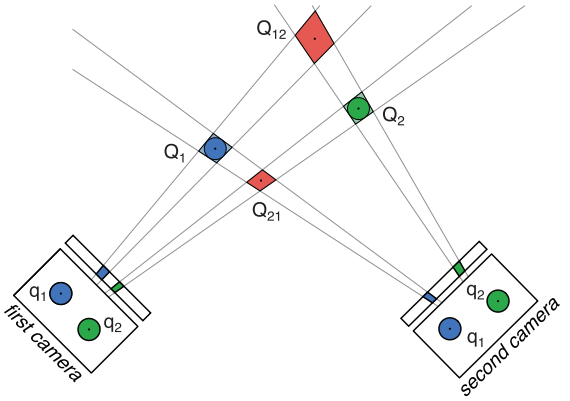

Fig. 8. *Ghost objects formation.* The blue and the green objects, namely, Q_1 and Q_2 are reprojected in the two cameras as q_1 and q_2. The pair (q_1, q_1) is a good match because the two pixels are the image of the same $3D$ point Q_1, and the pair (q_2, q_2) is a good match as well. The two pairs (q_1, q_2) and (q_2, q_1) do not represent any object of interest but they are good matches from a epipolar perspective, see [37]. The pixel matching procedure will then reconstruct the two correct objects Q_1 and Q_2 and the two *ghost* objects Q_{12} and Q_{21}. The introduction of a third camera drastically reduces the creation of *ghost* points working on triplets of pixels, but even in this case ambiguities are not completely solved.

results of the method), which has to be empirically chosen depending on the dataset. In the tests presented in Section 5 of the manuscript it is set to 10.

A more complicated situation occurs when a *ghost* and a real object get in $3D$ proximity creating a ambiguous connected component, with the typical Y–shape shown in Fig. 9 with one of the two branches (the one corresponding to the *ghost* trajectories) quite short. This kind of ambiguity is essentially due to an error in the dynamical linking procedure, which wrongly connects the *ghost* to a real cluster.

We identify the connected components with a Y–shape graph and we solve the ambiguity removing the wrong link (highlighted in red in Fig. 9) between the *ghost* and the real trajectory, actually breaking the graph into two partitions, representing a long and correct trajectory and a short *ghost* trajectory which is removed.

6.3 Evaluation Metrics

The set of trajectories retrieved by SpaRTA, see Fig. 10, is compared with the set of groudtruth trajectories (published

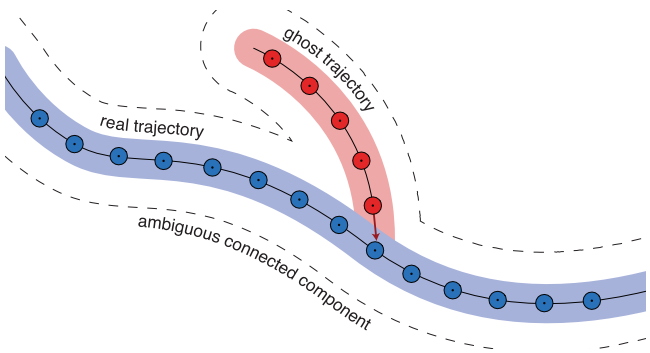

Fig. 9. *Ghost trajectories removal.* A Y–shape connected component representing a real (blue) and a *ghost* (red) trajectories occluded at one frame. This ambiguous component is due to a wrong dynamic link (the red arrow), which can be easily identified from the peculiar Y–shape of the component together with the short length of the *ghost* branch. The wrong dynamic link is detected and removed as well as the *ghost* points.

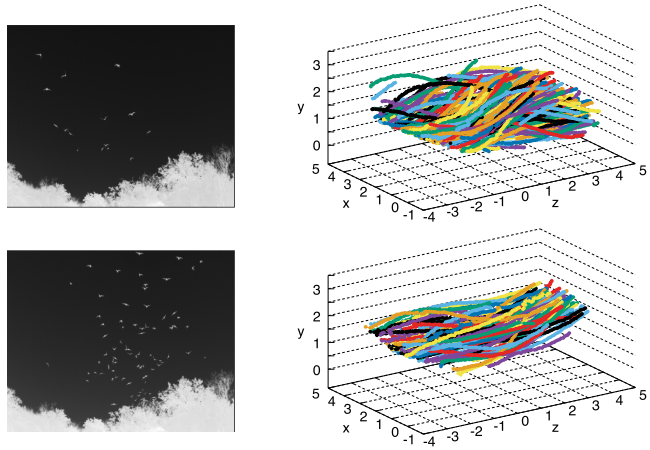

Fig. 10. *Dataset and results.* First row: on the left an image from the *Davis-08 sparse* dataset and on the right a sample of the $3D$ trajectories, each with a different color, of the same dataset reconstructed by SpaRTA. Second row: on the left an image from the *Davis-08 dense* dataset and on the right a sample of the $3D$ trajectories reconstructed by SpaRTA

by Wu et al. in [10] together with the two datasets) and its quality is assessed using the standard CLEAR MOT evaluation method proposed by Bernardin and Stiefelhagen in [39] and the metrics described in [10]. The most important metrics in this context is the so-called Multiple Object Tracking Accuracy parameter (MOTA); this quantity combines three observables, namely: i) the number of false–positive objects (reconstructed objects not belonging to the groundtruth set); ii) the number of missing objects (objects belonging to the groundtruth but not reconstructed by the algorithm); iii) the number of identity–switches (reconstructed trajectories switching between two different groundtruth identities).

MOTA can be interpreted as the fraction of groundtruth objects correctly reconstructed by the tracking method; its ideal value is equal to 100 percent (note, however, that – weirdly as it may seem – MOTA can have negative values when the number of false–positive plus miss–reconstructed objects plus identity switches exceeds the number of groundtruth objects, quite clearly a rather bad scenario). The second most important metrics is IDS (Identity Switches), which identifies how many times the identities of different actual targets are switched, this is also a rather crucial parameter, as (the consequences of IDS in data analysis are often the most severe).

The other evaluation parameters are the following: MT (Mostly tracked) – fraction of groundtruth trajectories correctly reconstructed for more than the 80 percent of their time length; ML (Mostly Lost) – fraction of groundtruth trajectories that are correctly reconstructed, but for less than the 20 percent of their time length; FM (Fragmentation) – corresponding to the number of times that a groundtruth trajectory, correctly reconstructed, is interrupted. It was not possible to evaluate the last parameter MOTP (Multiple Object Tracking Precision), which measures the average distance in the $3D$ space between the groundtruth and the reconstructed objects, because the dataset do not give the actual $3D$ positions of the targets but their estimates based on the SIP approach used by the author in [10]. Hence, with the MOTP we would have not evaluated the precision of our method, but only the difference between the $3D$ positions obtained by SpaRTA with the ones obtained in [10].

6.4 Results

The performance of SpaRTA is reported in Table 1 and compared with four other methods: MHT [10], SDD–MHT [10], CP(LDQD) [2] and GReTA [12]. The difference between the two datasets is that one of them is *sparse*, and the data sequence is rather long (1100 frames), while the second dataset is *dense*, with a far shorter sequence (200 frames), see Fig. 10.[4]

The comparison with the state–of–the–art methods shows the high performance of SpaRTA, especially in terms of tracking accuracy (MOTA), number of identity switches (IDS) and percentage of mostly tracked (MT) groundtruth trajectories. The table shows that SpaRTA performs better on the dense dataset (where SpaRTA produces the best set of trajectories) than on the sparse one.

More in detail, the set of trajectories obtained by SpaRTA shows the highest value of the MOTA parameters for both datasets: 91.7 percent for the sparse dataset and 90.4 percent for the dense one, while among the SIP methods CP(LDQD) gives the best performance with $MOTA = 88.1\%$ and $MOTA = 80.5\%$ respectively. For both datasets, SpaRTA produces a very low number of IDS (12 for the sparse and 11 for the dense dataset) comparable with the ones produced by GReTA (9 and 7 for the sparse and the dense dataset respectively) and outperforming all the SIP-RT methods. In the sparse dataset the percentage of MT obtained by SpaRTA (96.1 percent) is comparable with the 3 SIP-RT methods MHT, SDD–MHT and CP(LDQD), while on the dense dataset SpaRTA produces the highest percentage of MT (91.1 percent).

Conversely, for both datasets SpaRTA gives a high number of ML (1 and 5,4 percent respectively in the sparse and in the dense dataset.). In the sparse datasets, ML are due to targets that the segmentation routine described in Section 5.1 fails to detect especially when bats are going out the field of view of the cameras. Instead in the dense dataset the mostly lost trajectories are the groundtruth trajectories shorter than 10 frames. These trajectories are naturally lost in SpaRTA, because we remove from the final set all the trajectories shorter than 10 frames to avoid *ghost* trajectories, as decribed in Section 5.2.

Finally, on the sparse dataset SpaRTA produced the highest number of FM (185) compared with all the other methods. As for the ML, these FM are due to segmentation failures: when a miss-detection occurs the corresponding $3D$ object cannot be reconstructed, causing the interruption of the correspondent trajectory. This issue may be improved by developing a detection algorithm designed for this dataset; this, however, goes beyond the scope of the present work. On the other hand, on the dense datasets SpaRTA produces the lowest number of FM (123) confirming the overall better performance of SpaRTA on the dense dataset than on the sparse one.

7 Conclusion

We proposed a tracking method, SpaRTA (Spatiotemporal Reconstruction Tracking Algorithm), designed to track large and dense group of featureless objects, without any specific prerequisites on the $3D$ system used to acquired the data. SpaRTA works with $(3D + 1)$ dense clouds of points representing the volumes occupied in time by the targets of interest. Each cloud of points is partitioned in spatio-temporal connected components, corresponding to the trajectories of single individuals in the group. The method is designed to handle efficiently the ambiguities stemming from occlusions, i.e., objects getting too close in the $3D$ space to be detected as separated entities; to this aim SpaRTA employs an optimization routine inspired on Semi–Definite Programming techniques introduced in the field of statistical mechanics [9]. Apart from using in a proficous way for the first time SDP in the computer vision context, the true core of SpaRTA is the novel way in which the spatio-temporal graph is built: the key idea is to define an energy (or cost) function based on the use of both *attractive* and *repulsive* links between points within the cloud, in such a way to separate with relatively little numerical effort the ambiguous cases by minimizing such energy.

SpaRTA was tested on two public datasets, [10], producing high quality results in terms of correctness of the trajectories, evaluated through the standard quality parameter MOTA, producing a low rate of identity switches and a high percentage of mostly tracked groundtruth trajectories. The retrieved trajectories were compared with the four state–of–the–art tracking methods MHT [10], SDD-MHT [10], CP(LDQD) [2] and GReTA [12]. The greatest advantage of SpaRTA over the other methods is an outstanding MOTA, combined with an excellent IDS (second only to GReTA) and a very high MT. This means that, not only SpaRTA is able to achieve a quantitatively satisfying coverage of the set of trajectories, but it does that with the lowest number of false positives and a remarkably low number of identity switches compared to most other methods. At the level of data analysis, this kind of performance guarantees that the completeness of the data coverage is not jeopardized by a qualitative disruption of the results, due to the severe consequences of having wrong trajectories.

4. Tests were run on a laptop equipped with a 3.3 GHz i7 Intel processor and with 16 GB of RAM. Moreover, SpaRTA is implemented in C++ using the OpenCV library [40].

Acknowledgments

This work was supported by grants from the European Research Council, Proof of Concept Grant no. 713651. The authors would like to thank I. Giardina and M. Viale for the advice and the fruitful discussion on the new tracking strategy.

References

[1] N. T. Ouellette, H. Xu, and E. Bodenschatz, "A quantitative study of three-dimensional lagrangian particle tracking algorithms," *Experiments Fluids*, vol. 40, no. 2, pp. 301–313, 2006.
[2] Z. Wu and M. Betke, "Global optimization for coupled detection and data association in multiple object tracking," *Comput. Vis. Image Understanding*, vol. 143, pp. 25–37, 2016.
[3] X. E. Cheng, Z.-M. Qian, S. H. Wang, N. Jiang, A. Guo, and Y. Q. Chen, "A novel method for tracking individuals of fruit fly swarms flying in a laboratory flight arena," *PloS One*, vol. 10, 2015, Art. no. e0129657.
[4] M. Moussaid *et al.*, "Traffic instabilities in self-organized pedestrian crowds," *PLoS Comput. Biol.*, vol. 8, no. 3, 2012, Art. no. e1002442.
[5] L. Wen, Z. Lei, M.-C. Chang, H. Qi, and S. Lyu, "Multi-camera multi-target tracking with space-time-view hyper-graph," *Int. J. Comput. Vis.*, vol. 122, pp. 1–21, 2016.
[6] S.-I. Yu, Y. Yang, X. Li, and A. G. Hauptmann, "Long-term identity-aware multi-person tracking for surveillance video summarization," 2016, *arXiv:1604.07468*.

[7] P. Michel, J. Chestnutt, S. Kagami, K. Nishiwaki, J. Kuffner, and T. Kanade, "GPU-accelerated real-time 3D tracking for humanoid locomotion and stair climbing," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2007, pp. 463–469.
[8] A. Ess, K. Schindler, B. Leibe, and L. Van Gool, "Object detection and tracking for autonomous navigation in dynamic environments," *Int. J. Robot. Res.*, vol. 29, no. 14, pp. 1707–1725, 2010.
[9] F. Ricci-Tersenghi, A. Javanmard, and A. Montanari, "Performance of a community detection algorithm based on semidefinite programming," in *J. Phys.: Conf. Series*, vol. 699, 2016, Art. no. 012015.
[10] Z. Wu, N. Fuller, D. Theriault, and M. Betke, "A thermal infrared video benchmark for visual analysis," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, 2014, pp. 201–208.
[11] D. Reid, "An algorithm for tracking multiple targets," *IEEE Trans. Autom. Control*, vol. AC-24, no. 6, pp. 843–854, Dec. 1979.
[12] A. Attanasi *et al.*, 'GReTA-A novel global and recursive tracking algorithm in three dimensions," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 12, pp. 2451–2463, Dec. 2015.
[13] I. J. Cox and S. L. Hingorani, "An efficient implementation of reid's multiple hypothesis tracking algorithm and its evaluation for the purpose of visual tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 18, no. 2, pp. 138–150, Feb. 1996.
[14] Z. Wu, T. H. Kunz, and M. Betke, "Efficient track linking methods for track graphs using network-flow and set-cover techniques," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2011, pp. 1185–1192.
[15] H. S. Wu, Q. Zhao, D. Zou, and Y. Q. Chen, "Automated 3D trajectory measuring of large numbers of moving particles," *Opt. Express*, vol. 19, no. 8, pp. 7646–7663, 2011.
[16] Y. Liu, H. Li, and Y. Q. Chen, "Automatic tracking of a large number of moving targets in 3D," in *Proc. Eur. Conf. Comput. Vis.*, 2012, pp. 730–742.
[17] A. Tyagi, G. Potamianos, J. W. Davis, and S. M. Chu, "Fusion of multiple camera views for Kernel-based 3D tracking," in *Proc. IEEE Workshop Motion Video Comput.*, 2007, pp. 1–1.
[18] Y. Li, A. Hilton, and J. Illingworth, "A relaxation algorithm for real-time multiple view 3D-tracking," *Image Vis. Comput.*, vol. 20, no. 12, pp. 841–859, 2002.
[19] S. L. Dockstader and A. M. Tekalp, "Multiple camera fusion for multi-object tracking," in *Proc. IEEE Workshop Multi-Object Tracking*, 2001, pp. 95–102.
[20] Z. Wu, N. I. Hristov, T. H. Kunz, and M. Betke, "Tracking-reconstruction or reconstruction-tracking? comparison of two multiple hypothesis tracking approaches to interpret 3D object motion from several camera views," in *Proc. Workshop Motion Video Comput.*, 2009, pp. 1–8.
[21] J. Han, L. Shao, D. Xu, and J. Shotton, "Enhanced computer vision with microsoft kinect sensor: A review," *IEEE Trans Cybern.*, vol. 43, no. 5, pp. 1318–1334, Oct. 2013.
[22] A. Asvadi, P. Giro, P. Peixoto, and U. Nunes, "3D object tracking using RGB and LIDAR data," in *Proc. IEEE 19th Int. Conf. Intell. Transportation Syst.*, Nov. 2016, pp. 1255–1260.
[23] R. Mobus, A. Joos, and U. Kolbe, "Multi-target multi-object radar tracking," in *Proc. IEEE Intell. Vehicles Symp.*, Jun. 2003, pp. 489–494.
[24] M. Munaro, F. Basso, and E. Menegatti, "Tracking people within groups with RGB-D data," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2012, pp. 2101–2107.
[25] L. Zhang, Q. Li, M. Li, Q. Mao, and A. Nchter, "Multiple vehicle-like target tracking based on the velodyne lidar*," *IFAC Proc. Vol.*, vol. 46, no. 10, pp. 126–131, 2013.
[26] J. Choi, S. Ulbrich, B. Lichte, and M. Maurer, "Multi-target tracking using a 3D-lidar sensor for autonomous vehicles," in *Proc. 16th Int. IEEE Conf. Intell. Transp. Syst.*, Oct. 2013, pp. 881–886.
[27] R. Mobus and U. Kolbe, "Multi-target multi-object tracking, sensor fusion of radar and infrared," in *Proc. IEEE Intell. Vehicles Symp.*, Jun. 2004, pp. 732–737.
[28] A. Cavagna, C. Creato, L. Del Castello, S. Melillo, L. Parisi, and M. Viale, "Towards a tracking algorithm based on the clustering of spatio-temporal clouds of points," in *Proc. Int. Conf. Comput. Vis. Theory Appl.*, 2016, vol. 3, pp. 679–685.
[29] L. Rokach and O. Maimon, *Clustering Methods*. Boston, MA, USA: Springer, 2005, pp. 321–352.
[30] J. Hopcroft and R. Tarjan, "Algorithm 447: Efficient algorithms for graph manipulation," *Commun. ACM*, vol. 16, no. 6, pp. 372–378, 1973.
[31] M. Mezard and A. Montanari, *Information, Physics, and Computation*. London, U.K.: Oxford Univ. Press, 2009.
[32] A. Javanmard, A. Montanari, and F. Ricci-Tersenghi, "Phase transitions in semidefinite relaxations," *Proc. Nat. Acad. Sci.*, vol. 113, no. 16, pp. E2218–E2223, 2016.
[33] G. Turk, "Interactive collision detection for molecular graphics," Ph.D. dissertation, Dept. Comput. Sci., Univ. North Carolina Chapel Hill, Chapel Hill, NC, USA, 1989.
[34] Y. Boykov and V. Kolmogorov, "An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 9, pp. 1124–1137, Sept. 2004.
[35] F. Barahona, "On the computational complexity of ising spin glass models," *J. Phys. A: Math. Gen.*, vol. 15, no. 10, 1982, Art. no. 3241.
[36] A. Sobral and A. Vacavant, "A comprehensive review of background subtraction algorithms evaluated with synthetic and real videos," *Comput. Vis. Image Understanding*, vol. 122, pp. 4–21, 2014.
[37] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision. Cambridge Univ. Press*, 2003.
[38] D. H. Theriault *et al.*, "A protocol and calibration method for accurate multi-camera field videography," *J. Exp. Biol.*, vol. 217, no. 11, pp. 1843–1848, 2014.
[39] K. Bernardin and R. Stiefelhagen, "Evaluating multiple object tracking performance: The clear mot metrics," *EURASIP J. Image Video Process.*, vol. 2008, no. 1, pp. 1–10, 2008.
[40] G. Bradski and K. Adrian, "Â Learning OpenCV: Computer vision with the OpenCV library.," O'Reilly Media, Inc., 2008.

Andrea Cavagna received the PhD degree in theoretical physics from University Sapienza, Rome, Italy, in 1998. From 1999 to 2001, he worked as a postdoctoral fellow with the University of Oxford and with the University of Manchester. He is currently a researcher with the National Research Council (CNR)—Institute for Complex Systems (ISC), Rome. His research interests include statistical physics, collective animal behavior, optimization, and computer vision. Together with Irene Giardina, he leads the CoBBS group (Collective Behaviour in Biological Systems).

Stefania Melillo received the PhD degree in mathematics from University Sapienza, Rome, Italy, in 2010. She is currently a tenure researcher with the National Research Council (CNR)—Institute for Complex Systems (ISC), Rome, Italy, in the CoBBS group. Her scientific interests span from the dynamics of biological systems – experiments, 3D tracking, processing and analysis of experimental data – to different fields of applied mathematics.

Leonardo Parisi received the PhD degree in computer science from the Sapienza University of Rome, Italy, in 2016. He is currently a postdoctoral researcher with the Department of Mechanical and Aerospace Engineering (DIMA) of University Sapienza of Rome in 5SLab directed by F. Piergentili, where he applies computer vision techniques to the study of the dynamics of space debris. His scientific interests include computer vision, 3D tracking, and GPU programming.

Federico Ricci-Tersenghi received the PhD degree in physics from the Sapienza University of Rome, in 1999. From 1998 to 2001, he worked with the International Centre for Theoretical Physics. Since 2002 he has a tenured position in the Physics Department of Sapienza University, and became an associate professor in 2010. In 2009, he received the Tomassoni-Chisesi Prize. He has been a visiting professor with IHP and LPTMS, both in Paris. His research interests include statistical physics, disordered systems, statistical inference, large scale numerical simulations, development of algorithms, and computational complexity.