

An application-based characterization of dynamical distance geometry problems

Antonio Mucherino, Jérémy Omer, Ludovic Hoyet, Paolo Robuffo Giordano, Franck Multon

▶ **To cite this version:**

Antonio Mucherino, Jérémy Omer, Ludovic Hoyet, Paolo Robuffo Giordano, Franck Multon. An application-based characterization of dynamical distance geometry problems. Optimization Letters, 2020, 14 (2), pp.493-507. 10.1007/s11590-018-1302-6 . hal-01846265

HAL Id: hal-01846265
https://inria.hal.science/hal-01846265

Submitted on 25 Sep 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Noname manuscript No.
(will be inserted by the editor)

An application-based characterization of dynamical distance geometry problems

Antonio Mucherino[1], **Jeremy Omer**[2],
Ludovic Hoyet[3], **Paolo Robuffo Giordano**[4],
Franck Multon[5]

the date of receipt and acceptance should be inserted later

Abstract The dynamical Distance Geometry Problem (dynDGP) is the problem of finding a realization in a Euclidean space of a weighted undirected graph G representing an animation by relative distances, so that the distances between realized vertices are as close as possible to the edge weights. In the dynDGP, the vertex set of the graph G is the set product of V, representing certain objects, and T, representing time as a sequence of discrete steps. We suppose moreover that distance information is given together with the priority of every distance value. The dynDGP is a special class of the DGP where the dynamics of the problem comes to play an important role. In this work, we propose an application-based characterization of dynDGP instances, where the main criteria are the presence or absence of a skeletal structure, and the rigidity of such a skeletal structure. Examples of considered applications include: multi-robot coordination, crowd simulations, and human motion retargeting.

1 Introduction

Given a simple weighted undirected graph G and a positive integer K, the Distance Geometry Problem (DGP) asks whether there exists a realization x of G into a Euclidean space $\mathbb{R}^K$ so that a predefined number of distance constraints, involving pairs of realized vertices, are satisfied [18]. Saxe proved in 1979 that this problem is NP-hard [30].

Surveys and collections (see for example [18,19,21,24]) have recently been appearing in the scientific literature, showing a consistent interest in this topic from the research community. The DGP has several applications, including sensor network localization [12] and protein structure determination [16]. Most "traditional" applications of the DGP are based on a static representation of the problem: even

[1]Univ Rennes, Inria, CNRS, IRISA, Rennes, France. Email: antonio.mucherino@irisa.fr
[2]Univ Rennes, INSA Rennes, France. Email: jeremy.omer@insa-rennes.fr
[3]Univ Rennes, Inria, CNRS, IRISA, Rennes, France. Email: ludovic.hoyet@inria.fr
[4]Univ Rennes, Inria, CNRS, IRISA, Rennes, France. Email: paolo.robuffo_giordano@irisa.fr
[5]Univ Rennes, Inria, M2S, University of Rennes 2, Rennes, France. Email: franck.multon@irisa.fr

if protein functions may be associated to a protein dynamics, only energetically stable conformations of proteins are generally searched; and even in sensor networks, where sensors may change their positions over time, the solution of the problem is generally attempted for one network snapshot per time. More recently, an approach for extending DG to dynamical applications was proposed in [22].

In order to consider the temporal component in DGPs, we suppose that the vertex set of the graph G is the set product of two sets: the set V, which contains a predefined number of objects, and the set $T \subset \mathbb{N}_+$, consisting of the first $n = |T|$ integer positive numbers, which represents time as a sequence of discrete steps. A vertex of the graph is an ordered pair $(v,t) \in V \times T$, representing a given object v at a certain time t (in the following, we will use the notation v_t for the vertex (v,t) of G). The edge set E contains edges $\{u_q, v_t\}$, whose weights provide information about the distance between two vertices u and v at times q and t, respectively.

We suppose that two weight functions are associated to the graph G, and that both functions assign nonnegative weights to the edges in E. The function

$$\delta : \{u_q, v_t\} \in E \longrightarrow \delta(u_q, v_t) \in \mathbb{R}_+$$

associates a distance value to every pair of vertices of $V \times T$ belonging to the edge set E. We suppose that, for every t and $q \in T$, and for every u and $v \in V$, we have that $\delta(u_q, v_t) = \delta(v_t, u_q)$. Moreover, the function

$$\pi : \{u_q, v_t\} \in E \longrightarrow \pi(u_q, v_t) \in \mathbb{R}_+$$

assigns, to the edge $\{u_q, v_t\}$, a nonnegative value representing the "importance" of the distances $\delta(u_q, v_t)$, where higher values indicate higher importance. We also refer to π as the *priority level* of the distance δ. We say that the graph $G = (V \times T, E, (\delta, \pi))$ represents an instance of the dynamical DGP (dynDGP) [22].

Given a graph G, finding an optimal solution of a dynDGP consists in identifying a realization

$$x : V \times T \longrightarrow \mathbb{R}^K$$

of the graph such that the following penalty function is minimized:

$$\sigma(x) = \frac{1}{2} \sum_{\{u_q, v_t\} \in E} \pi(u_q, v_t) \left(\|x_u^q - x_v^t\| - \delta(u_q, v_t) \right)^2,$$

where $||\cdot||$ denotes the Euclidean norm, and x_v^t indicates the position in $\mathbb{R}^K$ of the object v at time t. The *stress* function σ was initially proposed in Kruskal's works on multidimensional scaling [13] in 1964, and successively studied and adapted by de Leeuw [4] in 1984. In [4], it was proved that σ is differentiable at x if and only if $||x_u^q - x_v^t|| > 0$ for all edges $\{u_q, v_t\}$ such that $\pi(u_q, v_t)\delta(u_q, v_t) > 0$. In dynDGP applications, the realization x represents an *animation* of the objects in V over the time steps in T.

The simplest approach to the dynDGP would be to tackle it as a classical (*static*) DGP, where the fact that distances may concern the same object at two different times, or two different objects at the same time, is neglected. However, this information can be actually exploited for creating ad-hoc instances of the dynDGP, as well as in the

solution methods. By reviewing some recent literature, we found out that different kinds of applications can be give rise to problems that can in fact be formulated as a dynDGP.

The dynDGP instances for the applications that we consider in this work can be constructed from known initial animations x. Generally, such animations are given by the trajectories x_v^t of the objects $v \in V$ for all times $t \in T$. Our approach consists in representing such animations by the relative distances between pairs of vertices u_q and v_t, so that they can subsequently be easily *manipulated* by introducing new distance constraints. The set of distances represented by the edges in E can be divided in two subsets:

- the set E^*, consisting of edges related to distance measurements obtained from the initial animations;
- the set E^+, consisting of edges related to newly introduced distances.

In general, modified or introduced distances are incompatible with the original ones, and as a consequence a higher priority π needs to be associated to them. We propose a characterization of these instances on the basis of the presence and of the properties of the possible skeletal structures (see Section 2) that can be found in the graph G.

The rest of the paper is organized as follows. In Section 2, we will analyze some particular subgraphs of G related to dynDGP instances, by focusing on the concept of graph rigidity. The focus of Section 3 will be on dynDGP instances having no skeletal structure (some preliminary computational examples will be shown in Section 3.2). Section 4 will instead be devoted to dynDGP instances admitting a skeletal structure, and we will separate our discussion for non-rigid (see Section 4.1) and rigid skeletal structures (see Section 4.4). An initial illustrative computational experiment will be given for this case in Section 4.2. Finally, Section 5 will conclude the paper.

2 Meaningful subgraphs and rigidity

The study of some subgraphs of G can provide some important information that may allow for verifying the relative dependence of the objects v during the animation. We will denote with $G[\cdot]$ the subgraph induced by the subset of vertices given as an argument.

The subgraph $G_t = G[V \times \{t\}]$, induced by the set product between V and only one temporal value t, corresponds to one *frame* of the animation at a fixed time. Let E_t be its edge set. In some dynDGP applications, this subgraph may contain a graph S, to which a skeletal structure [14] may be associated.

Definition 1 Given a connected graph $S = (V_S, E_S)$, we say that the pair (S, χ) is a *skeletal structure* if

$$\chi : V \longrightarrow \mathbb{R}^K$$

is a realization of the graph S such that $\chi(u) \neq \chi(v)$ for each $u, v \in V_S$ for which $u \neq v$.

Two skeletal structures (S, χ_1) and (S, χ_2) are said *isometric* if

$$\forall \{u, v\} \in E_S, \quad ||\chi_1(u) - \chi_1(v)|| = ||\chi_2(u) - \chi_2(v)||.$$

Moreover, we say that two skeletal structures (S,χ_1) and (S,χ_2) are *congruent* if

$$\forall\{u,v\} \in V_S \times V_S, \quad ||\chi_1(u) - \chi_1(v)|| = ||\chi_2(u) - \chi_2(v)||.$$

The two definitions above show that congruency implies isometry, because the set of constraints to be satisfy to verify the isometry case are also included in the set of constraints related to congruency (see equations above). These preliminary definitions allow us to define "graph rigidity" [11].

Definition 2 Given a connected graph $S = (V_S, E_S)$, if every pair of isometric skeletal structures (S,χ_1) and (S,χ_2) are also congruent, then we say that the graph S is rigid.

Poorly speaking, a graph is rigid when it is not possible to apply a continuous deformation to a realization of the graph while preserving the distances related to E_s.

Definition 3 Given a graph G representing a dynDGP instance, we say that G admits *skeletal structure* (S,χ) if, for every $t \in T$, S is subgraph of G_t and

$$\forall\{u,v\} \in E_S, \quad ||\chi(u) - \chi(v)|| = \delta(u_t, v_t).$$

It is important to remark that the realization χ is *not* the solution, for any time t, of the dynDGP instance. This realization ensures that the distances related to the edges in E_S are fixed to the value $||\chi(u) - \chi(v)||$ in the animations x which are solutions to the dynDGP. In the applications, such distances can represent stiff physical components of an object, such as a car, or of the human skeleton (e.g. its bones are stiff). Moreover, the existence of χ ensures that (S,χ) is related to a DGP instance that is realizable.

Similarly, the subgraph $G_v = G[\{v\} \times T]$ represents a sub-instance of the dynDGP instance where only one object of V is concerned. Let E_v be its edge set. We also consider the subgraphs $G_v^{(\underline{t},\bar{t})}$ corresponding to $G_v = G[\{v\} \times \{\underline{t},\ldots,\bar{t}\}]$, where $\underline{t}, \bar{t} \in T$ with $\underline{t} < \bar{t}$. Let $E_v^{(\underline{t},\bar{t})}$ be the edge set of $G_v^{(\underline{t},\bar{t})}$. In terms of dynDGP, a realization of the subgraphs G_v represents a possible *trajectory* of a fixed object v over time.

3 dynDGPs with no skeletal structure

We consider in this section a set of applications where the graph G, representing an instance of the dynDGP, admits no skeletal structure (see Def. 3). This is the case when working with animations of independent (or almost independent) objects, such as pedestrians in a crowd [9,27], "aircraft" [28,29], and multi-robot systems [5,20, 31]. In the first two applications, given the initial configuration of a set of moving objects, with known directions and speeds, our aim is to predict their future configurations while avoiding, for example, any kind of collisions. We will not enter in the details of every mentioned application, but we will rather consider a simplified problem where animations of moving objects in a plane are manipulated by formulating a dynDGP. However, we will discuss in more details the multi-robot system application in Section 4.3.

Given a known animation of a set of objects, we can obtain a representation of the animation that is based on inter-object distances. This allows us to construct an initial

graph G, which is complete, because all distances can be computed from a given animation. However, these distances are not all strictly necessary for the representation of the original animation.

In fact, the most important distances are those allowing to represent the movements of the considered objects, i.e. the inter-frame distances between the objects at different times t. Moreover, when there is no skeletal structure, it is not necessary to keep distances between different objects in a common frame. We can consider this kind of distances only in a second step, when including new desired distance constraints in G. For example, in order to avoid collisions that may occur in the original animation, we can impose that all distances between objects at the same frame are greater than a predefined positive threshold Δ.

More formally, instances of the dynDGP related to these applications can be modeled by a graph having the following edge set:

$$E^* = \bigcup_{t \geq \tau+1} \bigcup_{v \in V} E_v^{(t-\tau,t)} \subset E, \tag{1}$$

where $E_v^{(t-\tau,t)}$ is the edge set of the subgraph $G[\{v\} \times \{t-\tau, \ldots, t\}]$ (see Section 2), and the depth parameter $\tau > 0$ is the number of previous positions (in previous frames) for an object of V that are used to represent its movement. The corresponding distances δ can be either considered as is, or modified in order to imposing some desired effect. Different priority levels π can be associated to such distances, depending on which distances are considered to be more important in the animation (for example, the distances between two vertices concerning the same object v at two consecutive times t_1 and t_2 represent the movement speed, which will be preserved if a high priority is associated to this distance).

We will briefly review a well-known method for local continuous optimization in Section 3.1, and some computational experiments, where this method will be employed, will be presented in Section 3.2.

3.1 A non-monotone spectral gradient method

Gradient descent methods are typically used for local optimization, where the direction given by the vector opposite to the function gradient is explored for minimizing the function values. A crucial point is the identification of the *step* along the opposite gradient direction: when too short the function may be further optimized; when too long a local minima may be missed.

In [7], the gradient descent was coupled with a closed-form formula capable of providing the step α that improves the convergence properties of the gradient method, which depend upon the Hessian matrix of σ. For this reason, this class of methods is known as *spectral* gradient methods, for which proof of convergence was given for strictly convex functions (the function σ in Section 1 does not belong to this category). It was noticed that the function values do not decrease monotonically during the iterations of the spectral gradient. Therefore, as in [22], we use the closed-form formula for α proposed in [7] only to define an initial value that we subsequently

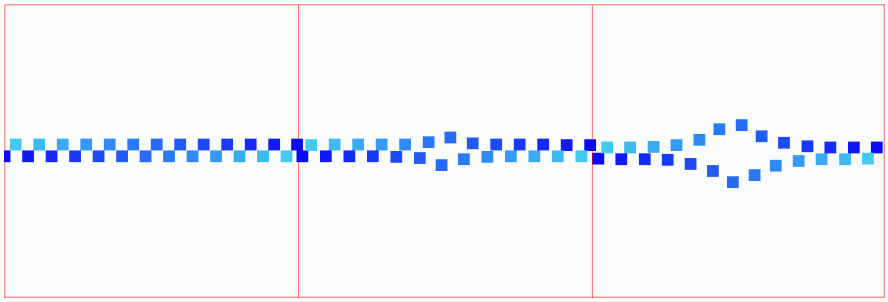


Fig. 1 Two trajectories of objects moving one towards the other. Different degrees of blue are used for representing different time steps. **(a)** the original animation; **(b)** the dynDGP solution found by imposing $\Delta = 0.1$; **(c)** the dynDGP solution found by imposing $\Delta = 0.2$.

refine with a non-monotone line search [3,33]. We implemented a spectral gradient algorithm with box projection [2], which allows to bound every distance in the corresponding intervals (the interval is degenerate in case of exact distances).

3.2 Some computational experiments

Fig. 1 shows a simple animation, that we intend to manipulated by our distance-based approach. In the original animation, two objects are initially positioned at the opposite sides of a 2D box of size 1.0×1.0, and they subsequently move towards each other. At the central frame, both objects are positioned near the center of the box, where they come very close to each other. Different degrees of blue (gray scale in black-and-white) are used for representing different time steps.

By representing this animation by distances, we can construct the graph G with the edge set E^* as in equ. (1), where our depth parameter τ is set to 3. Moreover, we added the following set of edges:

$$E^+ = \left\{ \{u_t, v_t\} : u, v \in V, u \neq v \text{ and } ||x^t_u - x^t_v|| \leq \Delta \right\},$$

where Δ is a strictly positive real number, and we added the following distance constraints:

$$\forall \{u_t, v_t\} \in E^+, \quad \delta(u_t, v_t) \in [\Delta, +\infty]. \tag{2}$$

In order to employ the spectral gradient method (see Section 3.1), $+\infty$ can be replaced by a sufficiently large real value. Notice that, differently from the initial paper [22] where the dynDGP was firstly introduced, we can currently work with distances represented by interval distances, which allows us to generate animations that look more natural.

The starting point for the spectral gradient method is the original animation, because not "too far" from the solution that is searched. Fig. 1(b) shows a solution where Δ was set to 0.1; Fig. 1(c) shows the equivalent result for $\Delta = 0.2$. The simplicity of the animation allows us to find good solutions by employing a local optimization

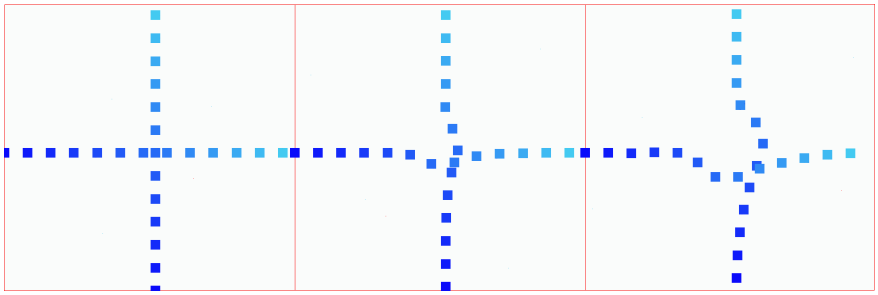


Fig. 2 An example similar to the one in Fig. 1, where the two objects have orthogonal trajectories. **(a)** the original animation; **(b)** dynDGP solution with $\Delta = 0.1$; **(c)** dynDGP solution with $\Delta = 0.2$.

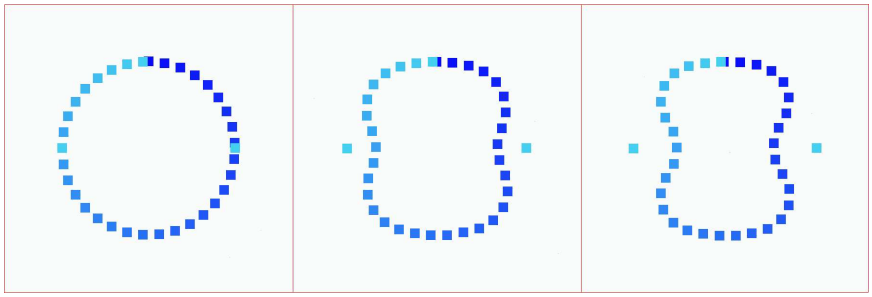


Fig. 3 A clock-like trajectory that initially finds two obstacles on its way (in light blue). **(a)** the original animation; **(b)** dynDGP solution with $\Delta = 0.1$; **(c)** dynDGP solution with $\Delta = 0.2$.

method. The animation in Fig. 2(a) is very similar to the previous one, but the two original trajectories are here orthogonal.

The experiment in Fig. 3 shows an initial animation where one object performs a clock-like motion. On its way, however, it finds two obstacles. Our dynDGP instance is therefore conceived in order to preserve this motion as much as possible, but at the same time to avoid any collisions with the obstacles. Two new animations are obtained, which resemble the original one, and where the imposed distance constraints are satisfied (with two different Δ values).

Notice that the use of the initial animation as a starting point in dynDGP solution methods such as the one in Section 3.1 allows us to ensure that the first and the last frame of the animation are preserved (if no new added constraints are violated by these two configurations), as it is generally required in these applications. Moreover, as far as not incompatible with newly added distance constraints, the optimization process tends to preserve as much as possible the original animation. This is the reason why it is unlikely to simulate, by means of a dynDGP, the trajectory of an object (such as an aircraft) that could go back to its initial position to avoid a collision, unless it is constrained to do so.

4 dynDGPs with skeletal structure

We consider now a dynDGP class where the graph G admits a skeletal structure (S,χ) (see Section 2). Given the graph S, different realizations χ_1 and χ_2 can be associated to S, where the relative distances between objects connected by the edges in S can be different.

In character animation, it is typical to create new animations for a character having skeletal structure (S,χ_2) by using a previously recorded animation of a character having skeletal structure (S,χ_1). The skeletal structure can either consist in a simplification of the human skeleton (in this case, distances represent bone lengths, see the representations in Fig 5); or it can represent more accurately the character morphology, by employing a more complex skeletal structure that is generally referred to as *mesh structure*. We will focus our attention on skeletal structures such as the one used for the animations in Fig 5. A discussion about the possible use of mesh structures is in Section 5.

One main challenge in character animation is the one of reproducing the desired animations without causing undesired body contacts [6,8,26]. The simplest approach to this problem consists in transferring all edge angles of the character having skeletal structure (S,χ_1) to the one having skeletal structure (S,χ_2). Consider, for example, the character representation in Fig 5, and consider that the character stands with his hands on his ears at frame t. Increasing the lengths of the arms (i.e the arms of (S,χ_1) are shorter than the arms of (S,χ_2)), and transferring the angles to the new skeletal structure would result in an animation where the character penetrates its brain with the hands. These particular problems are part of a general application that is known in the specialized literature as *motion retargeting*.

4.1 Non-rigid skeletal structures

Techniques that are more sophisticated than the simple angle-transfer approach have been proposed over the last years, yet without providing a general and efficient solution to the problem. These techniques mainly belong to two classes: the one where existing captured motions are manipulated [25], and the one where they are rather simulated by exploiting physical equations [10].

Our dynDGP approach to this application belongs to the former class of methods listed above. The main idea is to represent the original animation by distances and, with this distance information, to construct the graph G associated to the dynDGP. Since a character animation can be seen as a sequence of character postures, there are two particular ways to construct the dynDGP instances associated with a human animation. First, we can define the edge set of the graph G by employing the same procedure detailed in Section 3, where the information about the skeletal structure is included in the set E^+, and the highest importance is given to the corresponding distances. The experiments presented in Section 4.2 will be based on this model.

Alternatively, another way to generate a dynDGP instance from a given motion is to consider only inter-frame distances. In this case, we exploit the fact that every frame of the animation is a posture, which can be reconstructed independently, pro-

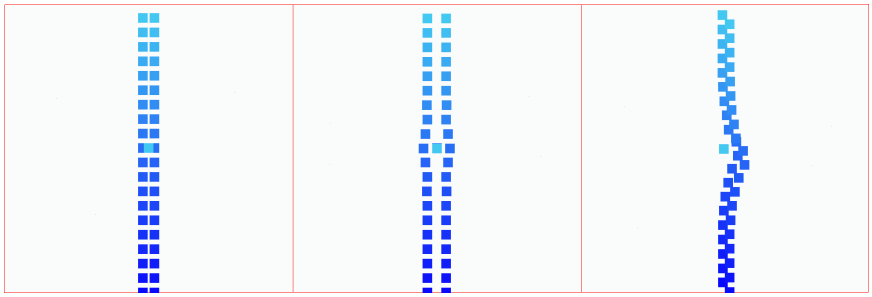


Fig. 4 Two "friendly" pedestrians that would be separated when avoiding an obstacle with the smallest deviation. **(a)** the original animation; **(b)** dynDGP solution without skeletal structure; **(c)** dynDGP solution with skeletal structure.

vided that enough distance information is given to obtained a final animation that is smooth and natural. The edge set $E^*$is defined as:

$$E^* = \bigcup_{t \in T} E_t,$$

and it contains all edges that relate vertices at same times t. Such edges present either bone lengths or relative movements. This alternative approach has the advantage of generating a sequence of smaller (and static) DGPs, which is convenient when dealing with very long animations. On the other hand, it requires an adaptation of the overall set of distances to the target skeletal structure, as explained below.

The process of replacing distances of (S,χ_1) with the corresponding distances of (S,χ_2) needs to take into consideration the overall set of distances. This is a very delicate step, because the inclusion of large changes in the modified distances may cause large distance incompatibilities, and spoil the simulations. One possible technique for modifying the distances related to pairs $\{u,v\} \notin E_S$ in accordance with new skeletal structure was recently presented in [1,23]. Some preliminary results can be viewed in a video clip associated to the paper [1] (downloadable from the conference website).

4.2 Some computational experiments

Two pedestrians that walk together, while chatting for example, tend to continue their trajectories in a way to keep interacting. In Fig. 4(a), our simulated pedestrians walk on a linear trajectory where they meet an obstacle at the central point of their environment (the same 1.0×1.0 used in the previous experiments). When a dynDGP instance is created from this animation, and distance constraints are added to the instance for avoiding collisions between the two pedestrians, as well as between pedestrians and the obstacle, then the solution reported in Fig. 4(b) is obtained. In order to preserve the original animation as much as possible, both pedestrians find their own way around the obstacle.

In order to simulate a joint walk, we added a *virtual* skeletal structure in our dynDGP instance, which simply connects the two pedestrians with one edge. Distances that are related to this skeletal structure naturally have a higher importance π. The solution to the so-modified dynDGP instance is given in Fig. 4(c). This simple example shows how important can be the impact of a skeletal structure in the found animations.

Finally, we present an illustrative experiment with a dynDGP instance having a skeletal structure resembling the human skeleton. The original animation is in dimension 2, and it simply shows a character slowly raising his hands until they come together and very close to his head (see Fig. 5(a)). In this setting, null distances between some pairs of vertices need to be avoided: in fact, while two hands can touch each other, a hand should not overlap with the vertex representing the geometrical center of the character head. The animation is recorded in terms of angle variations between pairs of consecutive edges, so that it can be played with different skeletal structures. This animation is composed by 100 frames. The frames 1, 25, 50, 75 and 100 are shown in the first column of Fig. 5.

When changing the skeletal structure by reducing the upper arms of the character by 20%, we obtain the animation shown in Fig. 5(b), where, at the end of the animation, the vertex head and the two hands are in collision, because placed almost at the same position in the 2-dimensional space. This is a typical retargeting problem, as mentioned in Section 4.

In order to correct this retargeting problem, we define a dynDGP instance as detailed in Section 4.1 (first method). We impose the modifications on the skeletal structure by high-priority distance constraints, and solve the instance by the spectral gradient method (see Section 3.1). The obtained result is in Fig. 5(c): the new animation satisfies the distance thresholds imposed on the skeletal structure, which avoids the overlaps between head and hands. Notice moreover that all frames of the animations are generated at the same time, so that frames where there are no collision problems (see for example the initial frames of our animation) are also modified. This helps obtaining a smoother animation because the changes are not only performed on the frames affected by retargeting problems.

4.3 The particular case of multi-robot systems

The application on multi-robot systems has some particular additional properties. We focus our attention on local on-board sensors that are capable to measure the distance between mobile robots. The main problem is to implement decentralized group localization and formation control algorithms, with the aim of deploying a highly autonomous robot teams in "non-trivial" environments (e.g. inside buildings, underwater, underground, or even in deep space). In recent works, the formation control is performed by ensuring that the robots are able to define, at every time $t \in T$, a skeletal structure (S, χ) that is rigid [32]. The existence or absence of an edge in S is generally associated to the distance range between two robots (if too large, the robots cannot communicate), and to the presence of obstacles (that does not allow the robots to obtain the necessary relative measurements).

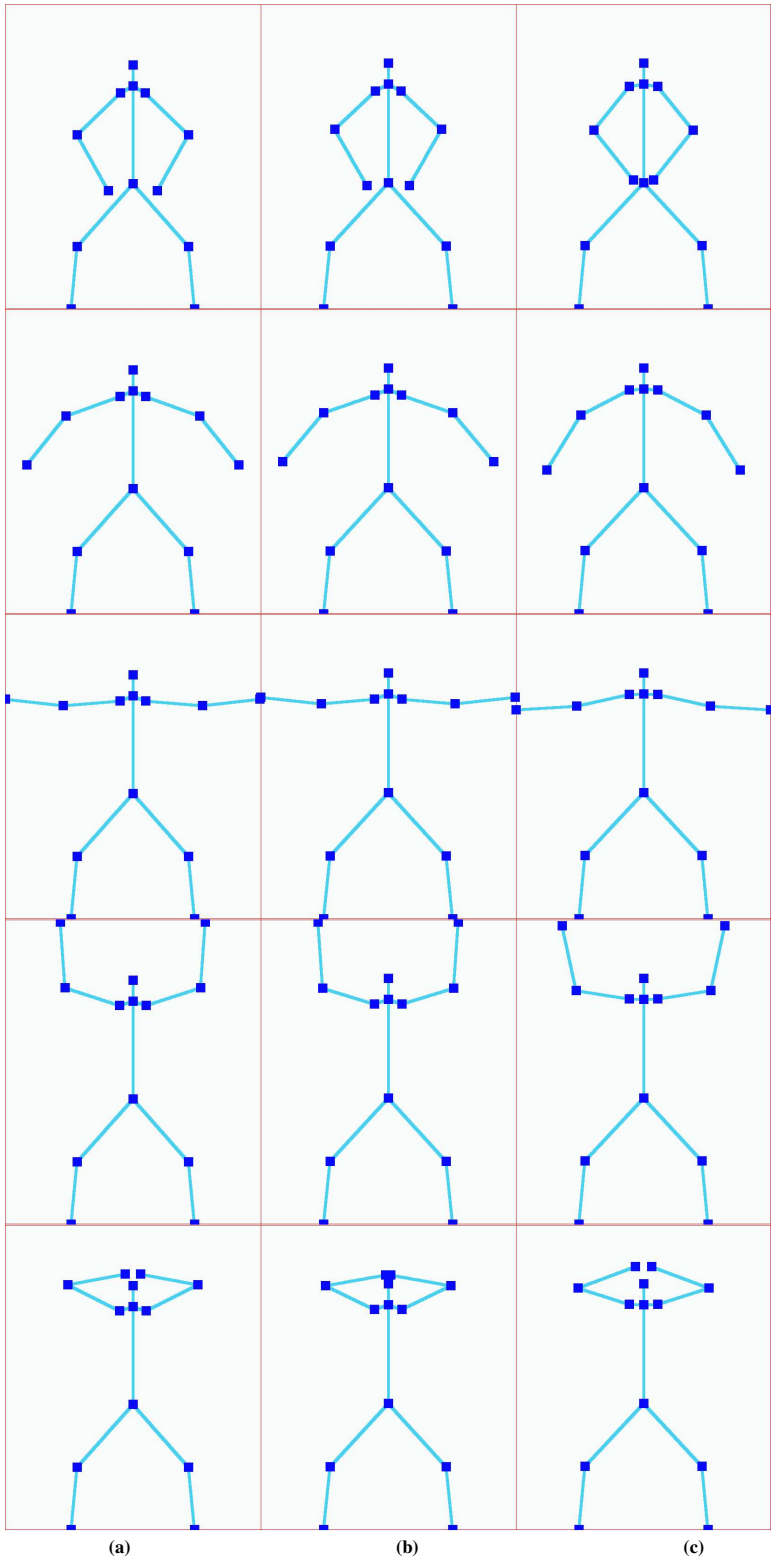


Fig. 5 A human skeleton in the 2-dimensional space. **(a)** the original animation; **(b)** a classical retargeting problem occurring when the animations are given in terms of angle variations; **(c)** the solution to our dynDGP instance.

Given a pre-recorded animation of a multi-robot system, we can represent it by relative distances for creating a graph G representing such an animation (see above). As for the other applications, by Def. 3, there is no skeletal structure in G. However, since it is imposed to the robots to form a rigid structure at every frame, the pair (G_t, x^t) is in fact a rigid skeletal structure for every $t \in T$.

As above, new distance constraints can be included in the dynDGP instance, and a new manipulated animation, describing the new trajectories for the robots, can be obtained by solving such an instance. However, in the particular application concerning multi-robots formations, we have an additional constraint imposing that (G_t, x^t) keeps defining a rigid skeletal structure. This is done to ensure a good level of communication among the robots. Even if this constraint is satisfied by the pre-recorded animation, the introduction of new distances in G can define new animations where this constraint may not be satisfied (when two robots are constrained to get further apart, for example, they may not be able to communicate anymore). The rigidity of (G_t, x^t), for every $t \in T$, is therefore an additional constraint that we will consider in future works in the context of the dynDGP.

4.4 Discretizable dynDGPs with rigid skeletal structure

The discretization of a dynDGP instance and the rigidity of the graph G (see Section 2) representing such an instance are two very close concepts. We say that a dynDGP instance is discretizable if it satisfies some special assumptions that allow for reducing its search space to a discrete domain having the structure of a tree [18]. It is easy to verify that, given a dimension K, every discretizable instance has an underlying graph G that is rigid in dimension K [15]. The inverse implication is not true in general. However, a dynDGP admitting a rigid skeletal structure includes such a rigid structure at every frame, and G is a graph that contains all such rigid subgraphs, together with several other distances. Therefore, even if this is not true in general, we can say that there are "chances" that a so-constructed graph G admits the discretization.

An example is given by the kind of mesh structure that is basically formed by a closed triangular grid. Such structures generally work in spaces having dimension $K = 3$. In terms of graph, every triangle of the mesh is a 3-clique, while two overlapping triangles induce a subgraph which misses only one edge to form a 4-clique. We will refer to this missing distances as the 6^{th} distance (six distances are necessary for a subgraph consisting of four vertices to form a clique). Since the 6^{th} distance can be estimated in practice, the discretization assumptions can be satisfied by dynDGP instances that rely on a triangular mesh skeletal structure.

In the representation of a human hand, for example, where several possible movements are possible, the pairs of overlapping triangles where the 6^{th} distance can be estimated are quite limited. Inversely, in the representation of a human skull, this distance can be fixed in most pairs of overlapping triangles.

The discretization allows to employ a branch-and-prune (BP) algorithm for the solution of dynDGP instances [17]. This algorithm performs a systematic exploration of the search tree, but it uses pruning devices to discover infeasible branches, so that

it is able to focus the search around the branches of the tree which contain solutions. In the context of motion retargeting, apart from reducing the dynDGP search space to a tree, the discretizability has the great advantage to fix all *exact* distances, such as the ones related to edges forming the stiff mesh triangles. This can avoid undesired vibrations of the skeletal structures in the animations obtained as dynDGP solutions.

5 Conclusions

This paper sets the ground for future research on specialized solution methods for the dynDGP. The dynDGP is an extension of the classical DGP, which turns out to have several interesting applications in different application domains. Our contribution is to be considered as an initial step for a more complete classification of these applications on the basis of their features. Even if it does not propose any new method from an optimization point of view, this work shows how different dynamical applications in the context of the DGP can be in fact tackled by existing optimization tools. Future research will consist in exploring in more details all different situations that are briefly discussed in this paper.

Acknowledgments

We wish to thank Douglas S. Gonçalves for the fruitful discussions. This work was partially supported by an INS2I-CNRS 2016 “PEPS” Project, and by the ANR Project ANR-14-CE27-0007 SenseFly.

References

1. A. Bernardin, L. Hoyet, A. Mucherino, D.S. Gonçalves, F. Multon, *Normalized Euclidean Distance Matrices for Human Motion Retargeting*, ACM Conference Proceedings, Motion in Games 2017 (MIG17), Barcelona, Spain, November 2017.
2. E.G. Birgin, J.M. Martínez, *Large-Scale Active-Set Box-Constrained Optimization Method with Spectral Projected Gradients*, Computational Optimization and Applications **23**, 101–125, 2002.
3. E.G. Birgin, J.M. Martínez, M. Raydan, *Nonmonotone Spectral Projected Gradient Methods on Convex Sets*, SIAM Journal on Optimization **10**, 1196–1211, 2000.
4. J. de Leeuw, *Differetiability of Kruskal's Stress at a Local Minimum*, Psychometrika **49**, 111–113, 1984.
5. D. Fox, J. Ko, K. Konolige, B. Limketkai, D. Schulz, B. Stewart, *Distributed Multirobot Exploration and Mapping*, Proceedings of the IEEE **94**(7), 1325–1339, 2006.
6. M. Gleicher, *Retargetting Motion to New Characters*. ACM Proceedings of the 25^{th} annual conference on Computer Graphics and Interactive Techniques, 33–42, 1998.
7. W. Glunt, T.L. Hayden, M. Raydan, *Molecular Conformations from Distance Matrices*, Journal of Computational Chemistry **14**(1), 114–120, 1993.
8. Ch. Hecker, B. Raabe, R.W. Enslow, J. DeWeese, J. Maynard, K. van Prooijen, *Real-Time Motion Retargeting to Highly Varied User-Created Morphologies*, Proceedings of ACM SIGGRAPH 2008, ACM Transactions on Graphics **27**(3), 2008.
9. D. Helbing, I. Farkas, T. Vicsek, *Simulating Dynamical Features of Escape Panic*, Nature **407**, 487–490, 2000.
10. J.K. Hodgins, W.L. Wooten, D.C. Brogan, J.F. O'Brien, *Animating Human Athletics*, Proceedings of the 22^{nd} annual conference on Computer Graphics and Interactive Techniques (SIGGRAPH95), 71–78, 1995.

11. B. Jackson, T. Jordán, *Connected Rigidity Matroids and Unique Realizations of Graphs*, Journal of Combinatorial Theory, Series B **94**, 1–29, 2005.
12. N. Krislock, H. Wolkowicz, *Explicit Sensor Network Localization using Semidefinite Representations and Facial Reductions*, SIAM Journal on Optimization **20**, 2679–2708, 2010.
13. J.B. Kruskal, *Multidimensional Scaling by Optimizing Goodness of Fit to a Nonmetric Hypothesis*, Psychometrika **29**, 1–27, 1964.
14. G. Laman, *On Graphs and Rigidity of Plane Skeletal Structures*, Journal of Engineering Mathematics **4**(4), 331–340, 1970.
15. C. Lavor, L. Liberti, N. Maculan, A. Mucherino, *The Discretizable Molecular Distance Geometry Problem*, Computational Optimization and Applications **52**, 115–146, 2012.
16. C. Lavor, L. Liberti, A. Mucherino, *The interval Branch-and-Prune Algorithm for the Discretizable Molecular Distance Geometry Problem with Inexact Distances*, Journal of Global Optimization **56**(3), 855–871, 2013.
17. L. Liberti, C. Lavor, N. Maculan, *A Branch-and-Prune Algorithm for the Molecular Distance Geometry Problem*, International Transactions in Operational Research **15**, 1–17, 2008.
18. L. Liberti, C. Lavor, N. Maculan, A. Mucherino, *Euclidean Distance Geometry and Applications*, SIAM Review **56**(1), 3–69, 2014.
19. L. Liberti, C. Lavor, A. Mucherino, N. Maculan, *Molecular Distance Geometry Methods: from Continuous to Discrete*, International Transactions in Operational Research **18**(1), 33–51, 2011.
20. E. Montijano, E. Cristofalo, D. Zhou, M. Schwager, C. Sagues, *Vision-based Distributed Formation Control without an External Positioning System*, IEEE Transactions on Robotics **32**(2), 339–351, 2016.
21. A. Mucherino, R. de Freitas, C. Lavor, *Distance Geometry and Applications*, special issue of Discrete Applied Mathematics **197**, 1–144, 2015.
22. A. Mucherino, D.S. Gonçalves, *An Approach to Dynamical Distance Geometry*, Lecture Notes in Computer Science **10589**, F. Nielsen, F. Barbaresco (Eds.), Proceedings of Geometric Science of Information (GSI17), Paris, France, 821–829, 2017.
23. A. Mucherino, D.S. Gonçalves, A. Bernardin, L. Hoyet, F. Multon, *A Distance-Based Approach for Human Posture Simulations*, IEEE Conference Proceedings, Federated Conference on Computer Science and Information Systems (FedCSIS17), Workshop on Computational Optimization (WCO17), Prague, Czech Republic, 441–444, 2017.
24. A. Mucherino, C. Lavor, L. Liberti, N. Maculan (Eds.), *Distance Geometry: Theory, Methods and Applications*, 410 pages, Springer, 2013.
25. F. Multon, L. France, M.P. Cani-Gascuel, G. Debunne, *Computer Animation of Human Walking: a Survey*, The Journal of Visualization and Computer Animation **10**(1), 39–54, 1999.
26. F. Multon, R. Kulpa, B. Bideau, *MKM: a Global Framework for Animating Humans in Virtual Reality Applications*. Presence: Teleoperators and Virtual Environments **17**(1), 17–28, 2008.
27. A.H. Olivier, A. Marin, A. Crétual, J. Pettré, *Minimal Predicted Distance: a Common Metric for Collision Avoidance During Pairwise Interactions between Walkers*, Gait Posture **36**(3), 399–404, 2012.
28. J. Omer, *A Space-Discretized Mixed-Integer Linear Model for Air-Conflict Resolution with Speed and Heading Maneuvers*, Computers and Operations Research **58**, 75–86, 2015.
29. A. Richards, J.P. How, *Aircraft Trajectory Planning with Collision Avoidance using Mixed Integer Linear Programming*, IEEE Conference Proceedings, American Control Conference 2002, Anchorage, AK, USA, 2002.
30. J. Saxe, *Embeddability of Weighted Graphs in k-Space is Strongly NP-hard*, Proceedings of 17^{th} Allerton Conference in Communications, Control and Computing, 480–489, 1979.
31. F. Schiano, A. Franchi, D. Zelazo, P. Robuffo Giordano, *A Rigidity-Based Decentralized Bearing Formation Controller for Groups of Quadrotor UAVs*, Proceedings of the 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS16), 5099–5106, 2016.
32. D. Zelazo, A. Franchi, H.-H. Bülthoff, P. Robuffo Giordano, *Decentralized Rigidity Maintenance Control with Range Measurements for Multi-Robot Systems*, The International Journal of Robotics Research **34**(1), 105–128, 2015.
33. H. Zhang, W.W. Hager, *A Nonmonotone Line Search Technique and its Applications to Unconstrained Optimization*, SIAM Journal of Optimization **14**(4), 1043–1056, 2004.