Graphical Models and Point Pattern Matching

Tibério S. Caetano, Terry Caelli, *Fellow, IEEE*, Dale Schuurmans
and Dante A. C. Barone

Abstract

This paper describes a novel solution to the rigid point pattern matching problem in Euclidean spaces of any dimension. Although we assume rigid motion, jitter is allowed. We present a non-iterative, polynomial time algorithm that is guaranteed to find an optimal solution for the noiseless case. First we model point pattern matching as a weighted graph matching problem, where weights correspond to Euclidean distances between nodes. We then formulate graph matching as a problem of finding a maximum probability configuration in a Graphical Model. By using graph rigidity arguments, we prove that a sparse Graphical Model yields equivalent results to the fully connected model in the noiseless case. This allows us to obtain an efficient Junction Tree algorithm that runs in polynomial time and is provably optimal for exact matching between noiseless point sets. For inexact matching, we can still apply the same algorithm to find approximately optimal solutions. Experimental results obtained by our approach show improvements in accuracy over current methods, even when matching noisy point patterns.

Index Terms

Structural pattern recognition, graph-theoretic methods, Markov random fields, pattern matching

I. Introduction

Point pattern matching (or point set matching) is a basic problem in Pattern Recognition that is fundamental to Computer Vision (stereo correspondence, image registration and model-based object recognition [1]–[4]), Astronautics [5], [6], Computational Chemistry [7], [8] and

T. S. Caetano is with National ICT Australia

T. Caelli is with National ICT Australia

D. Schuurmans is with the Department of Computing Science, University of Alberta

D. A. C. Barone is with the Instituto de Informática, Universidade Federal do Rio Grande do Sul

Computational Biology [9], [10]. Here we consider the (possibly noisy) rigid body case: when one pattern differs from a subset of the other by an isometry, but where position jitter may be present.

A. Problem Description and Related Problems

In general terms, the problem consists of finding a correspondence between elements of two point sets in $\mathbb{R}^2$ or $\mathbb{R}^3$ (or in $\mathbb{R}^n, n \in \mathbb{N}$, for general—not necessarily visual—patterns). In the case of *exact* matching, one point set differs from a subset of the other by an isometric transformation. In the *inexact* case, there is position jitter in one point set with respect to the other. This always occurs in practical application domains like those cited above, thus we face an *inexact* matching problem in practice, and matching algorithms need to take this into account.

A related, but more general problem, is that of *graph* matching, which consists of finding correspondences (one-to-one [11], many-to-one [12] or many-to-many [13], [14]) between the nodes of two graphs so as to achieve some form of global consistency. In this case, nodes and edges may have vector attributes or labels. There is a vast literature addressing the graph matching problem in pattern recognition, which can be divided generally into work on search methods [12], [15]–[21], and work on non-search methods, such as probabilistic relaxation [22]–[33], spectral and least-squares methods [3], [34]–[38], graduated assignment [11], genetic optimization [39] and other principles [13], [14], [40], [41]. For a recent comprehensive review on graph matching for pattern recognition, see [42]. We have shown how ideas similar to those presented in this paper can be applied to the graph matching problem in [12] and [43], however in this paper we focus specifically on the point pattern matching problem.

B. Potential Applications

Isometric point pattern matching (with jitter) is encountered in many application domains.

In Computer Vision, two sets of interest points extracted from two stereo images are approximately related by an isometry when the stereo pair has a narrow baseline. An accurate correspondence between the features results in an accurate depth map or the recovery of the 3D geometry of the scene [44]. This form of stereo correspondence constitutes one of the fundamental point pattern matching problems of Computer Vision.

In Astronautics, the attitude of sounding rockets or satellites can be estimated by matching stellar images acquired from the onboard star sensor (a CCD camera) to those in an empirical star catalog [45]. Images acquired from the same region of the sky but from different viewpoints reveal sets of stars whose coordinates are related by an isometric transformation [5]. In this way, the star matching problem can be posed as a rigid point pattern matching problem.

In Computational Chemistry, rigid point pattern matching is a recurrent problem in drug design, specifically in the identification of pharmacophores—common subsets of molecules that systematically interact with some receptor (i.e. that perform some specific task). By matching a set of molecules (called ligands) that activate ("bind") a given receptor, one can identify whether there is a common sub-conformation among the ligands. If this is the case, the structure encountered becomes a candidate pharmacophore, which is a distillation of the functional attributes of ligands that accomplish some specific task. The pharmacophore can then be used in the design of a new drug which is expected to systematically interact with the given receptor [8].

Finally, a similar problem arises in Computational Biology, when the interest is to detect specific structural motifs within a family of proteins (or DNA sequences). Identification of these motifs contributes to uncovering the mechanism of the proteins' operation [10].

In all these problems, rigid point pattern matching is a reasonable assumption, but small stochastic deviations in the point positions must be accommodated (jitter). This latter condition excludes methods that only apply to exact point pattern matching problems (like [46]). The technique proposed in this paper is precisely designed for this case: we make the rigid body assumption (isometric assumption) *but* jitter is allowed.

C. Related Literature

Several approaches have been proposed to solve the inexact point pattern matching problem. Major classes of solutions are based on *spectral* methods [2], [3], [35], *relaxation labeling* [26]–[28], [31]–[33], and *graduated assignment* [11], [47]. The first compares the eigen-structure of proximity matrices of the point sets. The second defines a probability distribution over mappings and optimizes using a discrete relaxation algorithm. The third combines the "softassign" method [48] with Sinkhorn's method [49] to optimize the mapping. All these approaches can be seen as using optimal representations (complete data models) and approximate inference procedures. Spectral methods use the spectrum of the adjacency matrix, but it is well-known that different

graphs can be co-spectral [50]; probabilistic relaxation labeling typically uses compatibility functions defined over all points, but the optimization procedure is iterative and known to be convergent to local minima [27]; graduated assignment also uses the entire set of pairwise compatibilities, being a continuous relaxation of the original combinatorial problem which aims at tractability, but is also only convergent to local minima [11]. These sources of approximation impact on performance in various ways. For example, it has been frequently reported that spectral methods are not robust to structural corruption nor to matching patterns of very different sizes [2], [3]. Relaxation methods degrade with significant increases in point set sizes [11]. Graduated assignment, although extremely robust with respect to jitter, has a number of heuristic parameters that need to be tuned and, more importantly, is very sensitive to matching sets of significantly different sizes [11], [51]. All these methods are polynomial time approximations that do not guarantee global optimization.

D. The Proposed Technique

In this paper, we propose a conceptually different approach that overcomes many of the limitations of previous techniques. Rather than using a complete data model and an approximate inference algorithm, we do the opposite: we approximate the representation but show how optimal polynomial time algorithms can be applied to the approximated data model. However, the hallmark of this approach is that the "approximated" data model can be proven to be *equivalent* to the complete data model in the limit case of exact matching. In other words, the result is an optimal algorithm that runs over an optimal representation. This allows us to obtain guaranteed optimal solutions in the noiseless case, and excellent approximate solutions for moderate noise, as will be shown. The resulting technique is robust with respect to size increases in the point patterns, as well as with respect to extreme differences in their sizes. It is also robust to moderate point jitter. Moreover, contrary to heuristic formulations, it is derived from first principles using Markov random field theory: the technique is non-iterative and has no intrinsic parameters to be tuned (the only parameter involved being inherent to all techniques that aim to cope with jitter).

Our formulation is based on posing the problem of deriving the best assignment as one of finding the maximum a posteriori (MAP) configuration of random variables in a probabilistic Graphical Model. We draw a fundamental connection between exact inference in Graphical Models and the "rigidity of graphs": by formalizing the redundancies present in point sets

embedded in Euclidean Spaces, we prove that there is a sparse Graphical Model topology that has the same MAP solutions as the fully connected model in the limit of exact matching. Remarkably, the sparse model has sufficient structure to allow us to perform exact MAP computation in polynomial time—a computation that is intractable in the fully connected model. To the best of our knowledge, this constitutes the first provably optimal polynomial time algorithm for exact point set matching in $\mathbb{R}^n$ that is also applicable to inexact matching (optimal algorithms which are exclusive to the unrealistic exact case do exist [46]).

For the realistic problem of matching noisy point patterns, we present experimental results comparing the proposed algorithm with well-known alternative methods. Our results show that the proposed technique offers accuracy improvements, particularly when matching patterns of different sizes.

Some of the basic ideas in this paper were first presented in [52].

II. Point Matching as a Weighted Graph Matching Problem

We start by showing how point pattern matching can be formulated as a weighted graph matching problem. Assume we have two point sets in $\mathbb{R}^n$ ($n \in \mathbb{N}$), named $\mathcal{T}$ for "template" and $\mathcal{S}$ for "scene", with cardinality T and S, respectively. The idea is that some noisy instance of $\mathcal{T}$ (denoted $\mathcal{T}'$) is present in $\mathcal{S}$, up to an isometric transformation. Our goal is to find this instance $\mathcal{T}'$ in $\mathcal{S}$ and, moreover, determine a map $f : \mathcal{T} \mapsto \mathcal{S}$ that maximizes some "global similarity measure" between $\mathcal{T}$ and $\mathcal{T}'$. The *only* restriction we impose on f is that it must be a function: every point in $\mathcal{T}$ must map to some point in $\mathcal{S}$. This is in contrast to the one-to-one mapping [11], which considers a smaller class of solutions. It is natural to understand $\mathcal{T}$ as the point set corresponding to a "model" and $\mathcal{S}$ as the point set obtained from a "scene" wherein we want to find some instance of the model.

Here we will refer to the template pattern as a "domain pattern" and the scene pattern as a "codomain pattern", in analogy to their role played with respect to the mapping function f. The i^{th} point in the domain pattern is denoted d_i, whereas the k^{th} point in the codomain pattern is denoted c_k. The Euclidean distance between d_i and d_j is denoted y^d_{ij}, and between c_k and c_l is denoted y^c_{kl}.

The key idea for modeling point pattern matching as a weighted graph matching problem is as follows. Recall that an isometry exists between two point sets if and only if they have

the same Euclidean Distance Matrix [53](EDM) under some permutation [54]. Consequently, an isometry can be tested by comparing all the permutations of two EDMs entry-wise. In our case, we would like to handle inexact matching, which means we must also accommodate noisy situations and sets of different sizes. Thus, we define the matching problem as finding the map f that minimizes the cost

$$U_T(f) = \sum_{i=1}^{T} \sum_{j=1}^{T} \mathcal{D}(y_{ij}^d, y_{f(i)f(j)}^c) \tag{1}$$

under the constraint that the map is a function (many-to-one mapping). Here $U_T(f)$ is the "total" cost to be minimized (the reason for calling it "total" will be clear later), and $\mathcal{D}(\cdot,\cdot)$ is some *dissimilarity measure* between distances. Note that the arguments of $\mathcal{D}(\cdot,\cdot)$ represent the entries in the EDMs under the permutation induced by f.

This definition is equivalent (apart from f being many-to-one instead of one-to-one) to that of the weighted graph matching problem of [11], where edge weights are restricted to be relative Euclidean distances between points corresponding to the respective vertices embedded in $\mathbb{R}^n$. (Note that since all distances are taken into account, the graphs are *fully connected*.) Eq. (1) actually represents an instance of the quadratic assignment problem which, in general, is known to be NP-complete [11]. Due to this graph matching formulation, we will refer to the "domain graph" G_d and the "codomain graph" G_c as the graph abstractions of the point sets. This gives the formulation of our problem as a "Euclidean" weighted graph matching problem.

III. Weighted graph matching as a MAP problem in a Graphical Model

This problem can be further reformulated as finding a maximum probability (MAP) configuration in a probabilistic Graphical Model [55]–[58]. Before presenting our formulation, we briefly review the main ideas about Graphical Models that will be required in our exposition.

A. Graphical Models

Graphical Models are graphical representations for families of factored joint probability distributions [55], [57], [59]. We will be considering exclusively *undirected* Graphical Models, sometimes referred to as *Markov random fields* in certain application domains. (In this paper, "Graphical Models" and "Markov random fields" are complete synonyms.) A Graphical Model is essentially a graph where nodes represent random variables and the edges represent a set of

conditional independence assumptions made among the random variables[1]. If a subset of nodes B separates (in the graph-theoretic sense) the set of nodes A from the set of nodes C, then this means, in the Graphical Model formalism, that A and C are conditionally independent on B; that is $p(AC|B) = p(A|B)p(C|B)$. For examples of Graphical Models that induce different sets of conditional independence assumptions among their variables, see Figure 1.

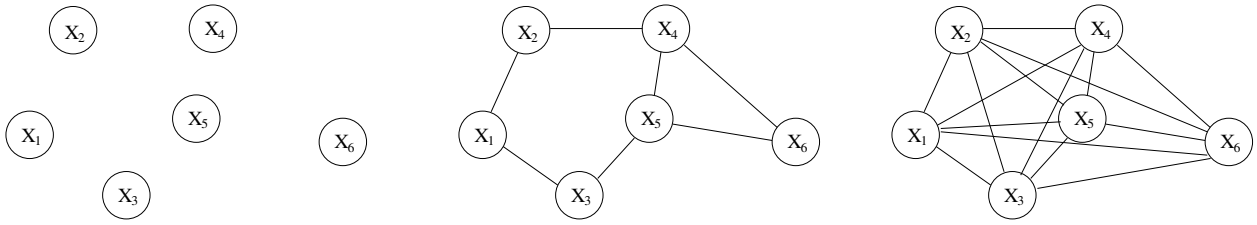

Fig. 1. Example of three undirected Graphical Models. **Left:** every imaginable conditional independence assumption holds. **Middle:** some conditional independence assumptions hold, some do not. **Right:** there are no conditional independence assumptions.

Figure 1 shows three Graphical Models. Each node, X_i, in a model corresponds to a random variable, which can assume a set of different realizations (in our context this set will be discrete). A fundamental result about Graphical Models is the Hammersley-Clifford (HC) theorem, which states that any strictly positive probability distribution that respects the set of conditional independencies implied by a graph can be written in a factored form, namely as a product of functions over the maximal cliques[2] [55], [57]:

$$p(x) = \prod_{c \in \mathcal{C}} \psi_c(x_c)/Z, \tag{2}$$

where c is a maximal clique, $\mathcal{C}$ is the set of all maximal cliques and x_c is the restriction of x to the clique c. Z is the normalization constant that renders $\sum_{\mathcal{X}} p(x) = 1$. The non-negative function $\psi_c(x_c)$ is called the *potential function* which, in our case, will be a table with the dimensionality of x_c. From this theorem, it is clear that all we need to specify a probability distribution is a connectivity pattern for the Graphical Model and a set of potential functions.

The basic "query" that we will be then interested in answering about a Graphical Model is the following: what is the most likely joint realization of all the random variables? In other words,

[1]All our statements about Graphical Models in this paper will be restricted to *discrete* random variables.

[2]Recall that a clique is a complete subgraph and a maximal clique is a clique which is not a proper subset of another clique.

what is the mode of the joint probability distribution defined by a Graphical Model and its potential functions? This is known as the MAP (maximum a posteriori) problem in a Graphical Model. For fully connected models, like the one in Figure 1-Right, this problem is intractable (for discrete random variables). For completely independent models, like that in Figure 1-Left, this problem is trivial: the joint mode can be obtained by computing each of the individual modes independently. Models that lie between these two extremes, of which the one in Figure 1-Middle is an example, can be either tractable or not.

At this point, it is important to state what determines the tractability of the model. The fundamental algorithm for exact inference in Graphical Models is the Junction Tree algorithm [55]–[57], [59]. It works by creating a hypergraph (a "Junction Tree") from the original graph and then running a dynamic programming algorithm on this hypergraph. However, Junction Trees can only be created for *triangulated*[3] (i.e. chordal) graphs [55], [57], so the effective computational complexity depends on triangulated versions of the original graph.[4] In general, there are many possible triangulations for a given graph. The exponential complexity of the MAP computation for a given Graphical Model will be determined by the minimum size, taken over all possible triangulations, of the maximal clique in the triangulation. If this exponent grows with the size of the graph, then the model is intractable, otherwise it is tractable. For example, a fully connected graph is triangulated with maximal clique size equal to the size of the graph itself, which immediately implies intractability. Naturally, in practice one requires the exponent to be not only fixed but also small. Notice also that, if a graph is *already* triangulated, other triangulations will only potentially *increase* the size of the maximal clique, so the exponential complexity will be given directly by the size of the maximal clique of the graph, without any need for triangulation. Since the problem of finding a triangulation that has minimal maximal clique size is NP-complete [57] (one calls it an "optimal triangulation"), the "ideal" scenario would be one in which the graph is already triangulated. We exploit this fact below by identifying a triangulated Graphical Model structure for our problem that has a small maximum clique size.

Next we show how the point pattern matching problem can be formulated as a MAP problem

[3]A graph becomes triangulated (or, equivalently, chordal) by adding edges in such a way that all cycles of length greater than three have a chord. A chord is an edge between two non-consecutive nodes in the cycle.

[4]Note that "transforming" a graph by triangulating it is not restrictive, since triangulation can only add edges and therefore only reduces the set of conditional independence assumptions implied by the original graph.

in a Graphical Model. Although in the initial formulation the Graphical Model will be fully connected (and thus intractable) we will show afterwards how we can obtain *the same* MAP solutions with a sparse, tractable model.

B. Formulation

The key idea for modeling weighted graph matching as a MAP problem in a Graphical Model is as follows. Assume that each vertex in the domain graph is a random variable X_i, and that each such random variable has a finite set of possible realizations coinciding with the set of vertices in the codomain graph. This means that a particular realization x_k of a random variable X_i corresponds to a particular map between the point d_i in the domain pattern and a point c_k in the codomain pattern. Thus, a joint realization $x = \{x_k\}$ of the set of variables $X = \{X_i, \forall i | d_i \in \mathcal{T}\}$ corresponds to a particular match between the point sets $\mathcal{T}$ and $\mathcal{S}$. In this spirit, one can define a probability distribution such that the most likely joint realization of the variables (the MAP configuration) corresponds to the minimum of Eq. (1).

In order to accomplish this, we specify a Markov random field based on edge-wise potentials over the *fully connected graph*. Let ψ_{ij} denote the local potential function for edge (i, j). Then, the joint probability distribution over the pairwise Markov random field is

$$\begin{aligned} p(X = x) &= \frac{1}{Z} \prod_{(i,j)} \psi_{ij}(X_i = x_i, X_j = x_j) \\ &= \frac{1}{Z} \exp\left(- \sum_{(i,j)} V_{ij}(X_i = x_i, X_j = x_j) \right) \end{aligned} \tag{3}$$

where $V_{ij}(X_i, X_j) = -\log(\psi_{ij}(X_i, X_j))$, and Z is a global normalization constant determined by summing the product of potentials over all possible joint realizations x. For clarity, in Eq. (3), we have used standard notation where x_i denotes a *generic* realization of X_i (i.e., *any* realization, not one in particular indexed by i). In the context of this paper, we find it more convenient to modify this notation such that X_i is still the random variable, but $x_{f(i)}$ is now the *specific* realization indexed by $f(i)$.

To relate this problem to Eq. (1) (and here we use the new notation), all we have to do is

specify appropriate potentials. In particular, define

$$\begin{aligned} V_{ij}(X_i = x_i, X_j = x_j) &= V_{ij}(d_i \mapsto c_{f(i)}, d_j \mapsto c_{f(j)}) \\ &\equiv \mathcal{D}(y^d_{ij}, y^c_{f(i)f(j)}). \end{aligned}$$

The resulting model becomes

$$\begin{aligned} p(f) &= \frac{1}{Z} \exp\left(-\sum_{i=1}^{T} \sum_{j=1}^{T} \mathcal{D}(y^d_{ij}, y^c_{f(i)f(j)}) \right) \\ &\propto \exp\left(-U_T(f)\right), \end{aligned} \tag{4}$$

thus maximizing $p(f)$ is equivalent to minimizing $U_T(f)$. Note that we now write the realization $X = x$ in the form of a map f: each random variable X_i, which corresponds to a point d_i, will "map" to a realization $x_{f(i)}$, which corresponds to point $c_{f(i)}$ (note the new notation).

Although this observation is interesting, it does not immediately yield a useful approach to solving the problem because MAP computations over a fully connected Markov random field are intractable. The key idea in this paper is to *approximate* $U_T(f)$ in such a way that only a subset of all the pairwise cliques in the fully connected model is taken into account. This will eventually lead us to a Graphical Model that is tractable. However, the hallmark of the particular model that we will obtain is that its MAP solutions can be proven to be *the same* as those of the fully connected model in the noiseless case. This makes the "approximation" exact.

IV. The Model

To construct a sparse alternative to the fully connected Graphical Model given in Eq. (4) we need to specify: (i) a set of potential functions that will define the function $\mathcal{D}$; and (ii) a connectivity pattern that will define the subset $\mathcal{C}_2$ of edges on which we will define potentials.

Fig. 2. Local "kernel" structure of the Graphical Model. Each random variable can assume S possible realizations, so that the sample space for two connected random variables has S^2 elements.

First, to specify the potentials, consider the local "kernel" structure of our model shown in Figure 2. Generally speaking, a potential function associates to each element of the sample space a non-negative real number [55], [57]. In our model, potentials will be defined on edges, where each node contained in an edge (a random variable) represents one of the T vertices in G_d, which, in turn, can assume a set of S possible realizations (which themselves correspond to vertices in G_c). Thus the sample space for each edge has S^2 elements, and we can specify the potential function for an edge (i.e., a pair $\{X_i, X_j\}$ in G_d) by an $S \times S$ matrix

$$\psi_{ij}(X_i, X_j) = \begin{pmatrix} \mathcal{S}(y_{ij}^d, y_{11}^c) & \dots & \mathcal{S}(y_{ij}^d, y_{1S}^c) \\ \vdots & \ddots & \vdots \\ \mathcal{S}(y_{ij}^d, y_{S1}^c) & \dots & \mathcal{S}(y_{ij}^d, y_{SS}^c) \end{pmatrix}, \tag{5}$$

where y_{ij}^d denotes the edge weight between vertices with indexes i and j in graph G_d (which corresponds to the Euclidean distance between points d_i and d_j). An analogous notation holds for y_{kl}^c. $\mathcal{S}$ is a function that measures the compatibility of the two arguments. Note that the compatibilities are a function of the *distances* between a pair of points in the domain pattern and corresponding points in the codomain, not similarities between *points* themselves.

To measure compatibility in the exact matching case (no noise) we can simply use the indicator function

$$\mathcal{S}(y_{ij}^d, y_{kl}^c) = \mathbf{1}(y_{ij}^d = y_{kl}^c) \equiv \begin{cases} 1, & \text{if } y_{ij}^d = y_{kl}^c \\ 0, & \text{if } y_{ij}^d \neq y_{kl}^c \end{cases} \tag{6}$$

For inexact matching, where we assume jitter in the point positions (typical in practice), we need a more general "proximity measure" to cope with uncertainty. Thus, in these cases we measure compatibility using the Gaussian kernel[5]

$$\mathcal{S}(y_{ij}^d, y_{kl}^c) = \exp\left(-\frac{1}{2\sigma^2}|y_{ij}^d - y_{kl}^c|^2\right). \tag{7}$$

Other similarity measures could be chosen, but we do not focus on this choice in this paper[6]. Note that *any* technique for matching noisy patterns requires some soft similarity measure, including the methods we compare to in this paper (where we use the same kernel). For example, relaxation

[5]In the exact matching case, the Gaussian kernel actually gives identical results to the indicator, since its maximum is attained uniquely at an exact match. However, the indicator makes the upcoming theoretical results clearer.

[6]An early attempt do evaluate different measures is reported in [60].

labeling [27] and graduated assignment [11] both use a *compatibility measure* between pairs of assignments to score any putative matching. These scores use a parameter to adjust for the level of position jitter in the data. Thus, the single parameter in Eq. (7) is not, itself, an artifact of our method, but a necessary element in any matching model that aims to cope with noise.[7]

Having specified the potential functions, it remains to determine the connectivity of the Graphical Model. Here we will simply propose the Graphical Model structure shown in Figure 3, and assert that this Graphical Model structure *preserves the MAP solutions of the fully connected model*—a fact we will verify in Section V below.

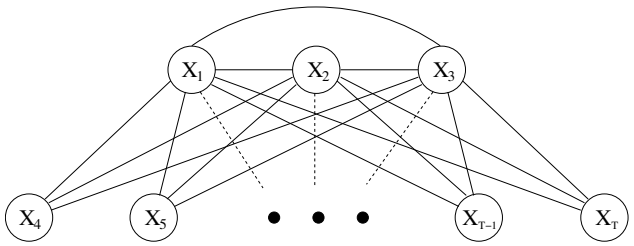

Fig. 3. A model for matching in $\mathbb{R}^2$. The topology of the model corresponds to that of a 3-tree graph, whose maximal clique size is 4 (thus independent on T, the number of nodes, and S, the number of possible realizations for each random variable).

Before proceeding with the proof of its optimality, we make a few remarks about this model. First, Figure 3 illustrates a model that is specifically constructed for matching in $\mathbb{R}^2$. For matching in $\mathbb{R}^{k-1}$, an analogous topology can be used: instead of a 3-clique in the upper layer, one simply uses a k-clique, and each of the other $T-k$ nodes is then connected to each of these k nodes. For any k (and $T > k$), this generic model topology has two important features: (i) it is *already* triangulated, and (ii) the size of the maximal clique is $k+1$, *independent* of both the number of nodes T and of the number of possible realizations S. As explained in Section III, because it is triangulated, we know that this model has a Junction Tree, and because it has a bounded maximal clique size, the "Junction Tree algorithm" has polynomial complexity in this model. That is, for models like the one in Figure 3 the *exact* MAP solutions can be computed in polynomial time.

It might sound artificial to define the "candidate" topology as a triangulated Graphical Model with a fixed maximal clique size (which together form sufficient conditions for polynomial

[7]We might then claim that our proposed method has no "intrinsic" parameter, in the sense that it does not introduce any parameter other than the one required to noise modeling.

time complexity). However, in the next section we show that this topology is not postulated, but derived from first principles, which reveals a subtle connection between exact inference in Graphical Models and the "global rigidity of graphs".

V. Optimality of the Model

In this section, we present theoretical results that lead to a special kind of graph: a "k-tree". The properties of this graph will allow us to draw a connection to the problem of exact inference in Graphical Models, and will ultimately lead us to prove that the model shown in Figure 3, although sparse and computationally tractable, yields *equivalent* results to the fully connected model in the limit case of exact matching.

A. A relevant lemma

We start by presenting a lemma that will be necessary to obtain the subsequent results.

Lemma 1: Let $S_1, S_2, \ldots, S_{n+1}$ be $(n+1)$ spheres in $\mathbb{R}^n$ whose centers are in general position (do not lie in a $(n-1)$-dimensional vector subspace). Then the intersection set $\cap_{i=1}^{n+1} S_i$ is either a single point or the empty set.

Proof: We use induction over n. Recall that a sphere in a vector space is the set of points equidistant to a fixed point.

The Lemma obviously holds for the base case when $n = 1$. See Figure 4.

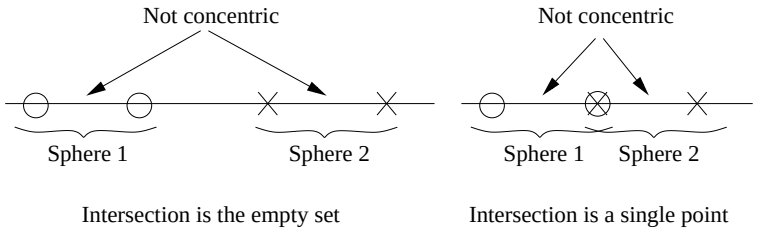

Fig. 4. Illustration for $n = 1$: 2 spheres in $\mathbb{R}^1$ whose centers do not lie in a 0-dimensional vector space (i.e. they are not concentric). **Left**: empty intersection. **Right**: intersection is a single point.

Now let $S_1 \cap S_2 = I_1$—see Figure 5(a). Then I_1 is a $(n-2)$-sphere lying in a $(n-1)$ vector subspace Q. (We use the convention of topology, which states that an $(n-2)$-sphere

is spanned necessarily by a *complete* basis in $\mathbb{R}^{n-1}$. For example, the 3D sphere in $\mathbb{R}^3$ is a 2-sphere, not a 3-sphere.) Let $I_i = S_{i+1} \cap Q$ for $i = 2, 3, \ldots, n$. Then $I_1, I_2, \ldots, I_n$ are n spheres in $Q \cong \mathbb{R}^{n-1}$ ($\cong$ denotes congruency) and, obviously, $\cap_{j=1}^{n} I_j = \cap_{i=1}^{n+1} S_i$—see Figure 5(b-d). Given the above definitions, the natural induction hypothesis that arises is: if the centers of the spheres $I_1, I_2, \ldots, I_n$ do not lie in a $(n-2)$ vector subspace, then the intersection of these spheres consists of at most a single point. Since $\cap_{j=1}^{n} I_j = \cap_{i=1}^{n+1} S_i$, we have from the hypothesis that $\cap_{i=1}^{n+1} S_i$ consists of at most a single point. So, what is left to prove is that the centers of the spheres $S_1, S_2, \ldots, S_{n+1}$ do not lie in a $(n-1)$ dimensional vector space (i.e. are in general position). Let $(x_1, x_2, \ldots, x_n)$ be the coordinates of $\mathbb{R}^n$. Let $(a_{i1}, a_{i2}, \ldots, a_{in})$ be the center of S_i. Without loss of generality, we may assume that Q is given by $x_1 = 0$. Then $Q \cong \mathbb{R}^{n-1}$ is parameterized by $(x_2, x_3, \ldots, x_n)$. The center of I_{j-1} has coordinate $(a_{j2}, a_{j3}, \ldots, a_{jn})$, $j \geq 2$. The centers of $I_1, I_2, \cdots, I_n$ are in general position if and only if the matrix

$$\begin{bmatrix} a_{22} & a_{23} & \ldots & a_{2n} & 1 \\ a_{32} & a_{33} & \ldots & a_{3n} & 1 \\ \vdots & \ddots & \vdots & \vdots & \vdots \\ a_{n+1,2} & a_{n+1,3} & \ldots & a_{n+1,n} & 1 \end{bmatrix} \tag{8}$$

is invertible, i.e. has maximal rank.

But this matrix is precisely the $n \times n$ lower-right submatrix of the following matrix

$$\begin{bmatrix} a_{11} & a_{12} & \ldots & a_{1n} & 1 \\ a_{21} & a_{22} & \ldots & a_{2n} & 1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{n+1,1} & a_{n+1,2} & \ldots & a_{n+1,n} & 1 \end{bmatrix} \tag{9}$$

which is the analogous matrix for the centers of $S_1, S_2, \ldots, S_{n+1}$. By subtracting the second row from the first row of matrix (9), we obtain

$$\begin{bmatrix} a_{11} - a_{21} & 0 & \ldots & 0 & 0 \\ a_{21} & a_{22} & \ldots & a_{2n} & 1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{n+1,1} & a_{n+1,2} & \ldots & a_{n+1,n} & 1 \end{bmatrix} \tag{10}$$

Note that $Q = \{x_1 = 0\}$ implies that $(a_{12}, a_{13}, \ldots, a_{1n}) = (a_{22}, a_{23}, \ldots, a_{2n})$, which creates the zeros in the first row. It is evident that matrix (10) is invertible if and only if $a_{11} \neq a_{21}$ and matrix (8) is invertible, which is the induction hypothesis. This implies that the centers of S_i do not lie in a $(n-1)$ vector subspace in R^n, which completes the proof. □

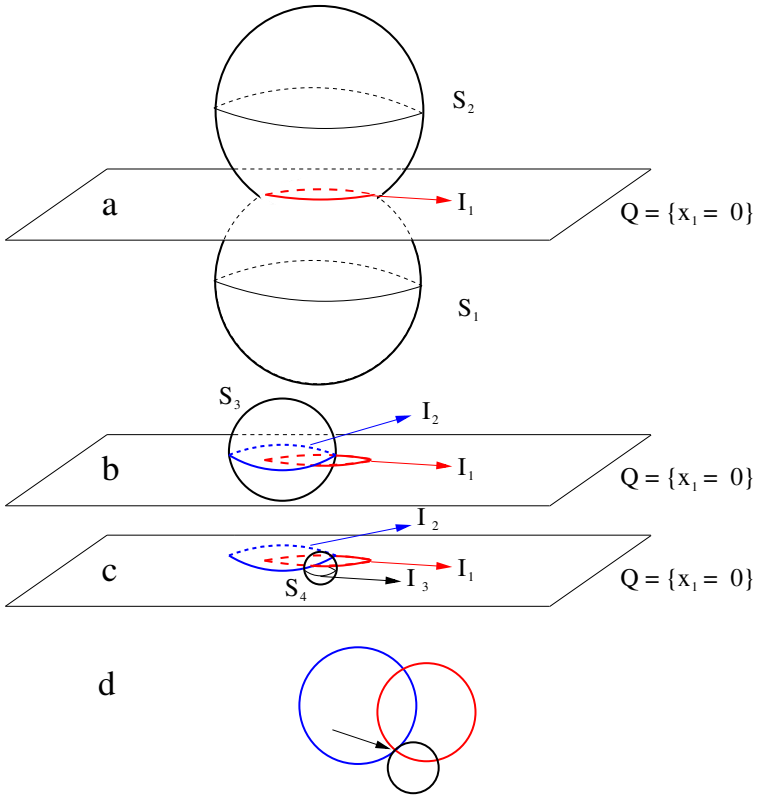

Fig. 5. Illustration of the construction used to prove Lemma 1: **(a)** The intersection of spheres S_1 and S_2 in $\mathbb{R}^n$ is *another sphere*, I_1, which lies in $Q \cong \mathbb{R}^{n-1}$; **(b) and (c)** (spheres S_1 and S_2 are omitted for clarity), I_j's, $j > 1$, are obtained by intersecting S_i's, $i > 2$, with Q; **(d)** The intersection of the I_j's is a single point (and so is that of the S_i's, due to $\cap_{j=1}^{n} I_j = \cap_{i=1}^{n+1} S_i$). In this example in $\mathbb{R}^3$, the 4 spheres S_1, S_2, S_3, S_4 have their centers in general position. The key equation is $\cap_{j=1}^{n} I_j = \cap_{i=1}^{n+1} S_i$, which allows us to construct an appropriate induction hypothesis relating n spheres (I_j's) in $\mathbb{R}^{n-1}$ with $n+1$ spheres (S_i's) in $\mathbb{R}^n$.

Another way to see this result is the following: if the distances from an unknown point to $n+1$ known points in $\mathbb{R}^n$ are determined, then this point is unique—provided the $n+1$ points are in general position. In order to see this fact, note first that the unknown point is clearly in the intersection of the spheres whose centers are the $n+1$ fixed points and the radii are the respective distances between their centers and the unknown point. Second, note that Lemma 1

states that the intersection is either a single point or empty (which is not the case because we have assumed the existence of this unknown point). This implies that the point is unique. This result will be used in the following in order to obtain another result concerning the "global rigidity of graphs".

B. Global rigidity of k-trees

Here we use Lemma 1 to infer a second result that will ultimately lead us to obtain the main theorem about the topology of the Graphical Model. The theory of graph rigidity, although mathematically rich and sophisticated [61], involves concepts that are easy to understand. Strictly speaking, we talk about the rigidity of *graph embeddings* in $\mathbb{R}^n$ where the edges are straight lines (these embeddings are called *frameworks*). Simply put, we say that a framework is globally rigid if the lengths of the edges uniquely determine the lengths of the "absent edges" (the edges of the complement graph).

To present the key result about the global rigidity of a special kind of framework—a k-tree—we start by reviewing some basic definitions from graph theory [62]. In what follows a complete graph with n vertices is denoted as K_n, and a k-clique is a clique with k vertices. Also recall that a *framework* is a straight line embedding of a graph.

Definition 1 (k-tree, base k-clique): A *k-tree* is a graph that arises from K_k by zero or more iterations of adding a new vertex to the graph and connecting it with k edges to an existing k-clique in the previous graph. The k-cliques defined by the new vertices are called *base k-cliques*.

Figure 6 shows the process of creating a k-tree, in the particular case where $k = 3$. We start with a K_3 graph. Then we add a vertex (4) and connect it to every vertex of the (so far unique) base 3-clique. Vertex (5) is then added and is connected, in this example, to the same base 3-clique. Vertex (6) is then added and connected to another base 3-clique, formed by vertices (2), (3) and (4). Note that all intermediate graphs generated in this way are themselves legitimate 3-trees. Also note that, in general, the resulting graph is sparse (the graph with 5 nodes is the first to present sparseness, since the edge (4-5) is absent).

A careful examination reveals that the size of the maximal clique of a k-tree with n vertices is precisely k if $n = k$ and precisely $k + 1$ if $n > k$. (This is easy to see because every time a

new vertex is added it is connected to exactly k vertices of a k-clique, forming a $(k+1)$-clique.)

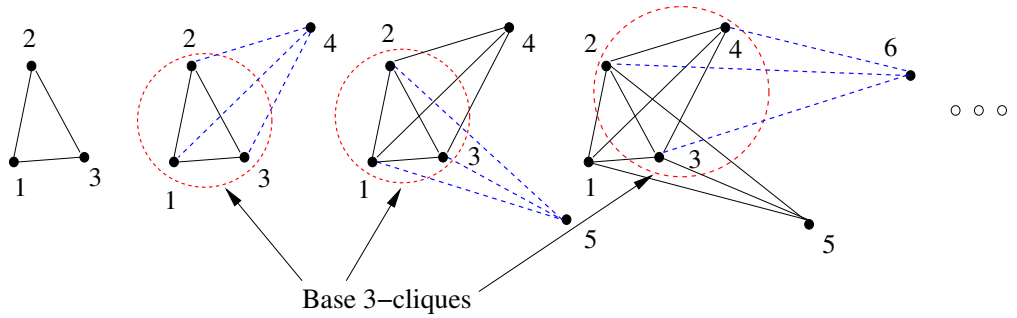

Fig. 6. The process of constructing 3-trees. At each step, a new node is added and connected to all nodes of an existent 3-clique (which is then called a "base 3-clique").

We are now equipped to present the second result:

Lemma 2: A k-tree framework with all base k-cliques in general position in $\mathbb{R}^{k-1}$ is globally rigid in $\mathbb{R}^{k-1}$.

Proof: We use induction on the number of vertices n in the k-tree framework. For $n = k$ the result is obvious because the graph is simply a k-clique, which is fully connected and by definition is globally rigid. Now assume the lemma is true for some $n > k$. First, choose a fixed (but arbitrary) coordinate system S. If the lemma holds for some $n > k$, then all the points in the framework are determined in S. Now include a new vertex with given distances from all the k vertices of any existent base k-clique in general position. By drawing edges corresponding to these known distances, we generate a new framework with $n+1$ vertices. But since the inserted vertex has determined distances from all vertices of a base k-clique which is in general position, its position is determined in S by virtue of Lemma 1. If its position is determined in S, then the positions of all vertices in the new framework are determined in S (because the previous framework was globally rigid by the induction hypothesis). If all the positions are determined in S, all pairwise distances are determined (irrespectively of S). This guarantees that the new framework is globally rigid in $\mathbb{R}^{k-1}$, which completes the proof. □

The direct implication of this result is that the k-tree framework, from the perspective of pairwise distances, has *exactly* the same information content as a fully connected framework. We

now show, using this fact, that our sparse model yields equivalent results to the fully connected model in the noiseless case.

C. Equivalence of k-tree versus full model

To present the main theoretical result, we introduce some new terminology to that established in Section II. We specifically analyze the noiseless case, where $\mathcal{T}$ and $\mathcal{T}'$ are related by an isometry. Consider the domain and codomain graphs G_d and G_c defined in Section II. Define $G_d^{kt} = (\mathcal{V}_d, \mathcal{E}_d^{kt})$ as a graph with the same nodes as G_d but with edge connectivity given by a k-tree whose base k-cliques are in general position in $\mathbb{R}^{k-1}$. Let $G_c^{kt} = (\mathcal{V}_c^{kt}, \mathcal{E}_c^{kt})$ be the subgraph of G_c whose nodes are those to which the nodes $\mathcal{V}_d$ map under an optimal map f. We define as $\bar{G}_d^{kt} = (\mathcal{V}_d, \bar{\mathcal{E}}_d^{kt})$ the complement graph of G_d^{kt}, while $\bar{G}_c^{kt} = (\mathcal{V}_c^{kt}, \bar{\mathcal{E}}_c^{kt})$ is the complement graph of G_c^{kt}.

Now, if we choose the edge set of the model (the set of pairwise cliques $\mathcal{C}_2$) to be a k-tree, the "approximated" optimization problem over this k-tree graph G_d^{kt} can be defined as one of minimizing the following "partial" cost function over f (as opposed to the "total" cost U_T from Eq. (1)):

$$U_{G_d^{kt}}(f) = \sum_{i,j|d_{ij} \in \mathcal{E}_d^{kt}} \mathcal{D}(y_{ij}^d, y_{f(i)f(j)}^c), \tag{11}$$

where d_{ij} is the edge between vertices d_i and d_j in G_d and $\mathcal{E}_d^{kt}$ is the edge set of graph G_d^{kt}.

We can now state our main result.

Theorem 1: In the exact matching case, a mapping function f which minimizes $U_{G_d^{kt}}(f)$ also minimizes $U_T(f)$.

Proof: If we define a cost function over the complement graph of G_d^{kt} ($\bar{G}_d^{kt}$) :

$$U_{\bar{G}_d^{kt}}(f) = \sum_{i,j|d_{ij} \in \bar{\mathcal{E}}_d^{kt}} \mathcal{D}(y_{ij}^d, y_{f(i)f(j)}^c), \tag{12}$$

we have

$$U_T(f) = U_{G_d^{kt}}(f) + U_{\bar{G}_d^{kt}}(f). \tag{13}$$

In the noiseless case, the dissimilarity function $\mathcal{D}(\cdot,\cdot)$ associated to a particular match is described simply in terms of an indicator function (Eq. (6)):

$$\mathcal{D}(y^d_{ij}, y^c_{f(i)f(j)}) \;\; = \;\; 1 - \mathbf{1}(y^d_{ij} = y^c_{f(i)f(j)}). \tag{14}$$

The optimal matching function f is such that $U_T(f) = 0$. Obviously, from Eq. (13) it holds that $U_T(f) = 0 \Rightarrow U_{G^{kt}_d}(f) = 0$, since $U_{G^{kt}_d}(f)$ and $U_{\bar{G}^{kt}_d}(f)$ are non-negative because $\mathcal{D}(\cdot,\cdot)$ is non-negative (Eqs. (11) and (12)). Our purpose is to prove the converse, i.e. that $U_{G^{kt}_d}(f) = 0 \Rightarrow U_T(f) = 0$. According to Eq. (13), in order to do so, it suffices to prove that $U_{G^{kt}_d}(f) = 0 \Rightarrow U_{\bar{G}^{kt}_d}(f) = 0$.

Here we use Lemma 2, which asserts that if the distances corresponding to the edges of a k-tree framework whose base k-cliques are in general position are determined, then all the remaining distances between vertices not connected by an edge are determined.

Let us write this result symbolically, for a k-tree in the domain graph, as

$$\{y^d_{ij} = const^d_{ij}, \forall i,j | d_{ij} \in \mathcal{E}^{kt}_d\} \;\; \Rightarrow \;\; \{y^d_{ij} = const^d_{ij}, \forall i,j | d_{ij} \in \bar{\mathcal{E}}^{kt}_d\} \tag{15}$$

where $const^d_{ij}$ is a constant for fixed i and j.

Since $U_{G^{kt}_d}(f) = 0$, every term of the sum in Eq. (11) must be zero, since they are non-negative:

$$\mathcal{D}(y^d_{ij}, y^c_{f(i)f(j)}) = 0, \forall i,j | d_{ij} \in \mathcal{E}^{kt}_d. \tag{16}$$

However, from the definition of $\mathcal{D}(\cdot,\cdot)$ for exact matching (Eq. (14)), this means that

$$y^d_{ij} = y^c_{f(i)f(j)}, \forall i,j | d_{ij} \in \mathcal{E}^{kt}_d. \tag{17}$$

Notice that the statement (Eq. (15)) holds for any k-tree whose base k-cliques are in general position, so it holds for G^{kt}_c in particular. (Recall that G^{kt}_c has its base k-cliques in general position in $\mathbb{R}^{k-1}$ because it is assumed isometric to G^{kt}_d, which by assumption has its base k-cliques in general position and so is globally rigid.) Therefore we conclude:

$$\begin{aligned} &\{y^c_{f(i)f(j)} = const^c_{f(i)f(j)}, \forall i,j | d_{f(i)f(j)} \in \mathcal{E}^{kt}_c\} \\ &\quad \Rightarrow \;\; \{y^c_{f(i)f(j)} = const^c_{f(i)f(j)}, \forall i,j | c_{f(i)f(j)} \in \bar{\mathcal{E}}^{kt}_c\} \end{aligned} \tag{18}$$

Notice that Eq. (17) implies that the left hand sides of implications (15) and (18) are equivalent. As a result, their right hand sides are equivalent and we obtain:

$$y^d_{ij} = y^c_{f(i)f(j)}, \forall i,j | d_{ij} \in \bar{\mathcal{E}}^{kt}_d. \tag{19}$$

Substituting this into Eq. (12), yields

$$U_{\bar{G}_d^{kt}}(f) = 0, \tag{20}$$

which was what we wanted to prove. □

Note now that the model shown in Figure 3 has the topology of a k-tree (a 3-tree). As a result, the solution obtained by the Junction Tree algorithm over this model will not only minimize the cost function $U_{G_d^{kt}}(f)$, but also the cost function of a complete model, $U_T(f)$ (Eq. (1)). This is our main theoretical result.

Actually, other models can be used, as long as they have the topology of a k-tree. The specific choice of 3-tree for Figure 3 was made simply because it has a single base 3-clique, and therefore only requires these 3 points to be non-collinear (the points corresponding to random variables X_1, X_2 and X_3). In the case of exact matching, as long as these points are not collinear, *any* choice can be made and Theorem 1 will still hold. However, when there is position jitter, different choices can give different results, and the variance of the results over different selections of the reference points will increase with jitter (experimental evidence of this fact will be provided). A principled way of selecting the reference points in this case is still an open problem which we are currently investigating, and for the purposes of the experiments presented in this paper the selection of the reference points is made randomly.

VI. INFERENCE

Given the k-tree model, we must solve the MAP problem, i.e. determine the most likely joint realization of the random variables in the model. This is done with the Junction Tree algorithm. In this section we describe how the Junction Tree algorithm is applied to our particular case (for details of the general case, see [57] and [55], [56]).

The Junction Tree for the model shown in Figure 3 is given in Figure 7.

The tree in this case is actually just a chain, thus we have a "Junction Chain". The maximal cliques in a Junction Tree are denoted by circles, called "clique nodes", whereas the set of variables common to adjacent clique nodes are represented by rectangles, called "separator nodes". The Junction Tree algorithm is a dynamic programming procedure that systematically changes the potentials in the clique nodes and separator nodes in a two-way "message-passing" scheme, similar to the Viterbi algorithm for MAP computation in Hidden Markov Chain models

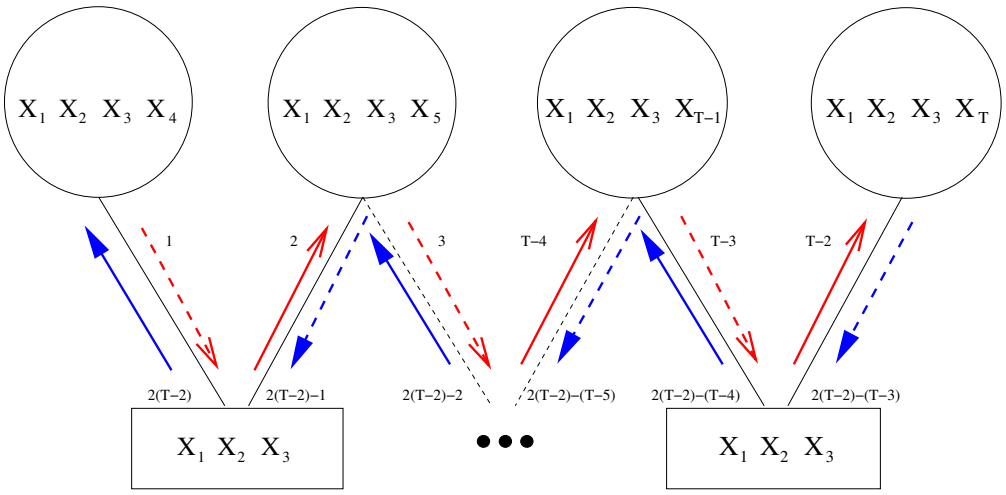

Fig. 7. The Junction Tree for the model in Figure 3. Circles ("clique nodes") correspond to the maximal cliques of the original graph, whereas rectangles ("separator nodes") correspond to the intersection between adjacent clique nodes. Non-filled arrows correspond to the first message-passing, whereas filled arrows correspond to the second. The value adjacent to an arrow denotes the order in which the corresponding message is passed. Dashed arrows correspond to Eq. (21), whereas solid arrows correspond to Eq. (22).

[63]. These updates preserve the joint distribution, but achieve a useful form of local consistency. Just like the Viterbi algorithm, which after the forward and backward operations delivers the individual MAP distributions for each node, the Junction Tree algorithm delivers the MAP distribution for each clique node (which, in this case, possibly involves multiple nodes of the original graph—four in the case of Figure 7). The final MAP distribution for each individual node X_i can then be computed by "maximizing out" the remaining individual nodes within the clique node [55], [57]. For example, the final MAP distribution for node X_1 in Figure 7 can be computed by $p(x_1) = \max_{x_2} \max_{x_3} \max_{x_4} p(x_1, x_2, x_3, x_4)$. This operation is clearly exponential on the number of variables in the clique node, and that is one of the reasons why the Junction Tree algorithm is only efficient for graphs with small maximal clique size.

The message-passing scheme works as follows. First we initialize the potential functions for each of the clique nodes by combining the pairwise potential functions (which are obtained from the compatibilities described in Section IV): $\Psi(x_i, x_j, x_k, x_l) = \psi(x_i, x_j)\psi(x_j, x_k)\psi(x_k, x_l)$. The separator nodes are all then initialized to 1 [57]. Then we perform message-passing: starting

with a clique node V that is a leaf of the chain, we compute

$$\Phi_S^* = \max_{V \setminus S} \Psi_V \tag{21}$$

$$\Psi_W^* = \frac{\Phi_S^*}{\Phi_S} \Psi_W, \tag{22}$$

where W is the clique node to which V is "sending a message". This "message" actually consists of two updates: (i) substituting the potential in the separator S by computing the MAP of clique node V with respect to the nodes that are common with the separator; and (ii) re-weighting the potential in clique node W by the ratio between the new and the previous separator potentials. This local operation is then propagated until the other leaf is reached, when it is repeated in the reverse direction. Once complete, the joint distribution has been preserved, but the marginalization property Eq. (21) has now been established between every clique node and its separators. This ensures that we have obtained the desired MAP distribution at each clique node [57], as mentioned above.

To compute the messages, note that each potential Ψ is a 4D table, with S bins per dimension; see Figure 7. Thus, the maximization operation in Eq. (21) runs over the dimension of the 4D table, Ψ_V, that is not common to the 3D table, Φ_S. Similarly, the division and multiplication operations of Eq. (22) are performed entry-wise in the tables. Figure 7 shows details of how the overall dynamic programming procedure works. As mentioned above, after the two-way message-passing is finished, local maximization yields the final MAP distributions of the singleton nodes X_i, from which the mode indicates the point in the codomain pattern that matches the point in the domain corresponding to X_i.

The complexity of computing each of the messages (Eq. (21) and Eq. (22)) is $O(S^4)$, since the largest tables (Ψ's) are 4-dimensional with S bins per dimension. There are in total $2(T-2)$ messages, so that the overall computational complexity for this model is $O(TS^4)$, which is polynomial in the size of the domain pattern (T) and in the size of the codomain pattern (S). Note that there is no iterative procedure involved, and no concept of "initialization" is present. The algorithm runs is precisely $2(T-2)$ steps and will always deliver the same result for the same input. This is because the algorithm is strictly deterministic, based solely on the dynamic programming principle [55], [57], [59]. Since the dynamic programming finds the global optimum for the given model *and* the model itself is optimal in the noiseless case, we have an algorithm

for point pattern matching that has polynomial complexity and is provably optimal in the limit case of exact matching.

VII. Experiments and Results

One obvious shortcoming of this theory is that it only addresses the exact matching case. For inexact matching, the theoretical guarantee that the minimum of Eq. (11) equals the minimum of Eq. (1) no longer holds. However, there remains value to the framework: in the noisy case, one can still run the Junction Tree algorithm with the compatibility measure of Eq. (7) to cope with approximate matches, requiring the same polynomial time. The only question is: how significantly does the quality of the approximate match degrade?

To evaluate this question, we conducted a number of experiments to compare our method (denoted simply as JT) to standard techniques in the literature, including probabilistic relaxation labeling (PRL), as described in [26], the spectral method (SB) presented in [35], and Graduated Assignment (GA) [47]. Note that these methods encode all pairwise distances in their objectives, whereas our method only encodes those distances that correspond to the k-tree topology. On the other hand, our approach uses an optimal non-iterative algorithm, whereas the others are based on approximate heuristic algorithms. None of the standard approaches—PRL, SB or GA—have any optimality guarantees, even in the noiseless case. The experiments involve matching tasks in $\mathbb{R}^2$, so we use the 3-tree model of Figure 3.

A. Synthetic data

To compare techniques across a range of problem conditions, we generated random points according to a bivariate uniform distribution in the interval $x = [0, 1]$, $y = [0, 1]$. We conducted two sets of synthetic experiments: (i) point sets $\mathcal{S}$ and $\mathcal{T}$ of equal sizes comparing JT, GA, PRL and SB, (ii) point sets $\mathcal{S}$ and $\mathcal{T}$ of different sizes comparing JT, GA and PRL (SB is not suited for different graph sizes). In order to compute the similarity measure between pairwise assignments, we use the same Gaussian kernel with $\sigma = 0.4$ for all methods (see Section (IV)). This is the only parameter involved in our method, but is also necessary in the other ones. The problem of selecting σ is far from trivial [3], and in this paper we do not aim at optimizing over this parameter. See [60] for tentative experiments in this sense. We basically choose a value that we know will not underflow the kernel computation in the case of extremal differences between

the argument and the mean of the Gaussian function. For the construction of the 3-tree, the 3 reference points were selected randomly.

In the first experiment, we used patterns of size (10,10), (20,20), (30,30) and (40,40) points. For each of these 4 instances, we perturbed the codomain pattern with progressive levels of noise: from small levels (std = 0 to 1), to moderate levels (std = 2), to high levels (std = 4). The value 'std' is 256 times the real standard deviation used (i.i.d. Gaussian noise). Typical instances of patterns perturbed with jitter of std = 1 and std = 4 are shown in Figure 8. Figure 9 shows the obtained curves under these experimental conditions. Each point in a curve is the average over 300 trials.

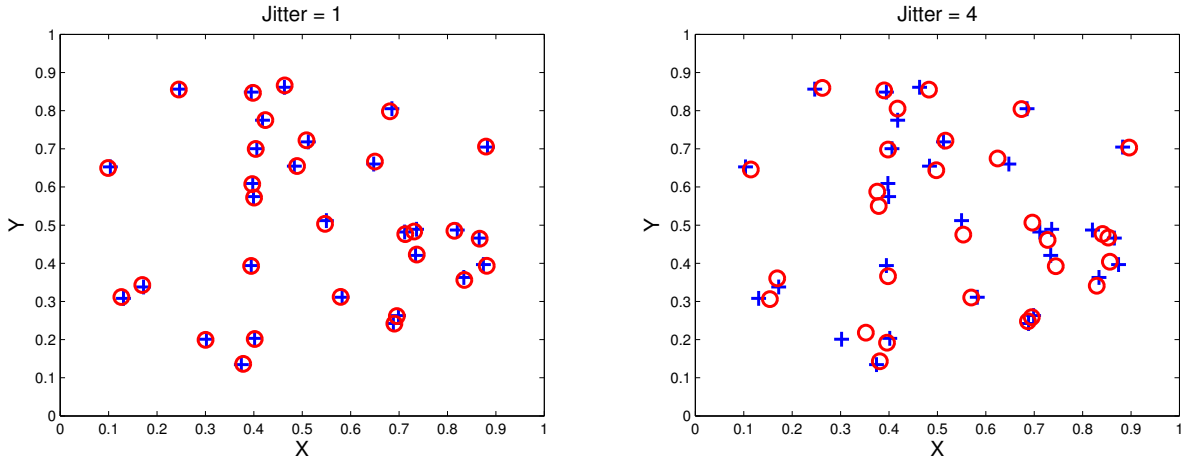

Fig. 8. Instances of patterns when different levels of jitter are introduced. Circles correspond to the jittered pattern whereas "plus" corresponds to the original pattern (the patterns were superimposed for the purpose of visual comparison; in practice, as should be clear from the text, they may be translated/rotated/reflected with respect to each other and may also have different cardinalities).

The graphs show that JT, GA and PRL are much more robust under jitter than SB, confirming the known fact that spectral methods are very sensitive to structural corruption (which is one of the reasons why significant research effort has been recently dedicated to alleviating this problem with spectral techniques [2], [3], [38], [51]). The graphs also show that, when the pattern sizes are increased, GA and PRL are still very robust across the whole range (the curves are almost horizontal), whereas JT is more sensitive to high jitter. However, it is clear that JT is competitive for small to moderate jitter (std = 0-2). The curves for GA and PRL essentially just undergo a change in offset for different pattern sizes, which reveals decreasing performance in the low jitter

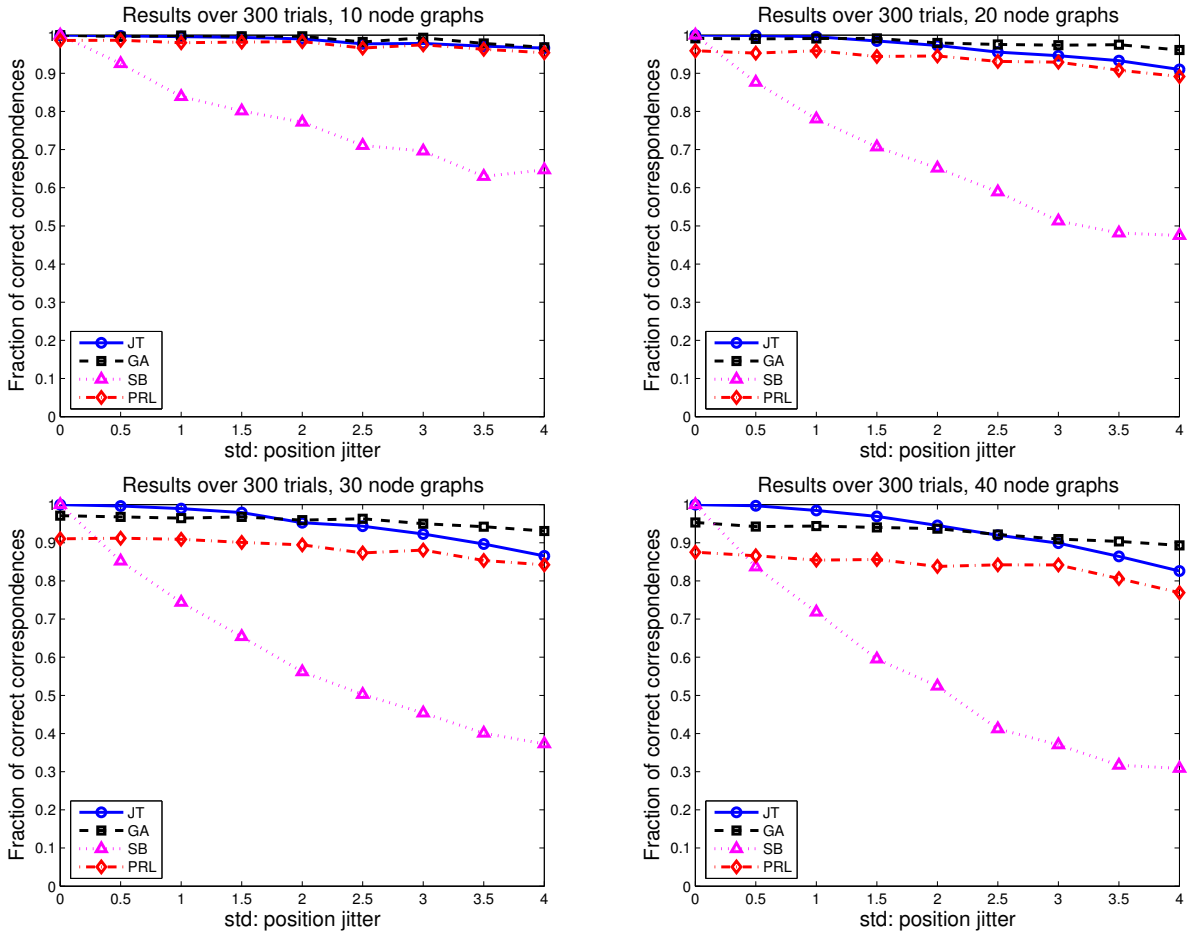

Fig. 9. Comparison of JT, GA, PRL and SB in matching equal-sized point sets under varying jitter. Results shown for 10, 20, 30 and 40 node graphs.

region. PRL is particularly more sensitive than GA for large matching problems, as reported in [11].

In the second experiment we held the size of the domain pattern $\mathcal{T}$ constant (10 nodes) and varied the size of the codomain pattern $\mathcal{S}$ (from 10 to 35 nodes in steps of 5), for various jitter levels (std = 1,2,3,4). In this experiment, we compared JT, GA and PRL only, since SB is not suited for graphs with different sizes. Figure 10 shows the results of this experiment. Each point in a graph corresponds to the average over 300 trials. Clearly the accuracy of JT does not degrade significantly for larger codomain patterns, even under high jitter, whereas the performances of GA and PRL begin to fail dramatically.

Overall, it is possible to notice that, whereas GA and PRL are more robust with respect to

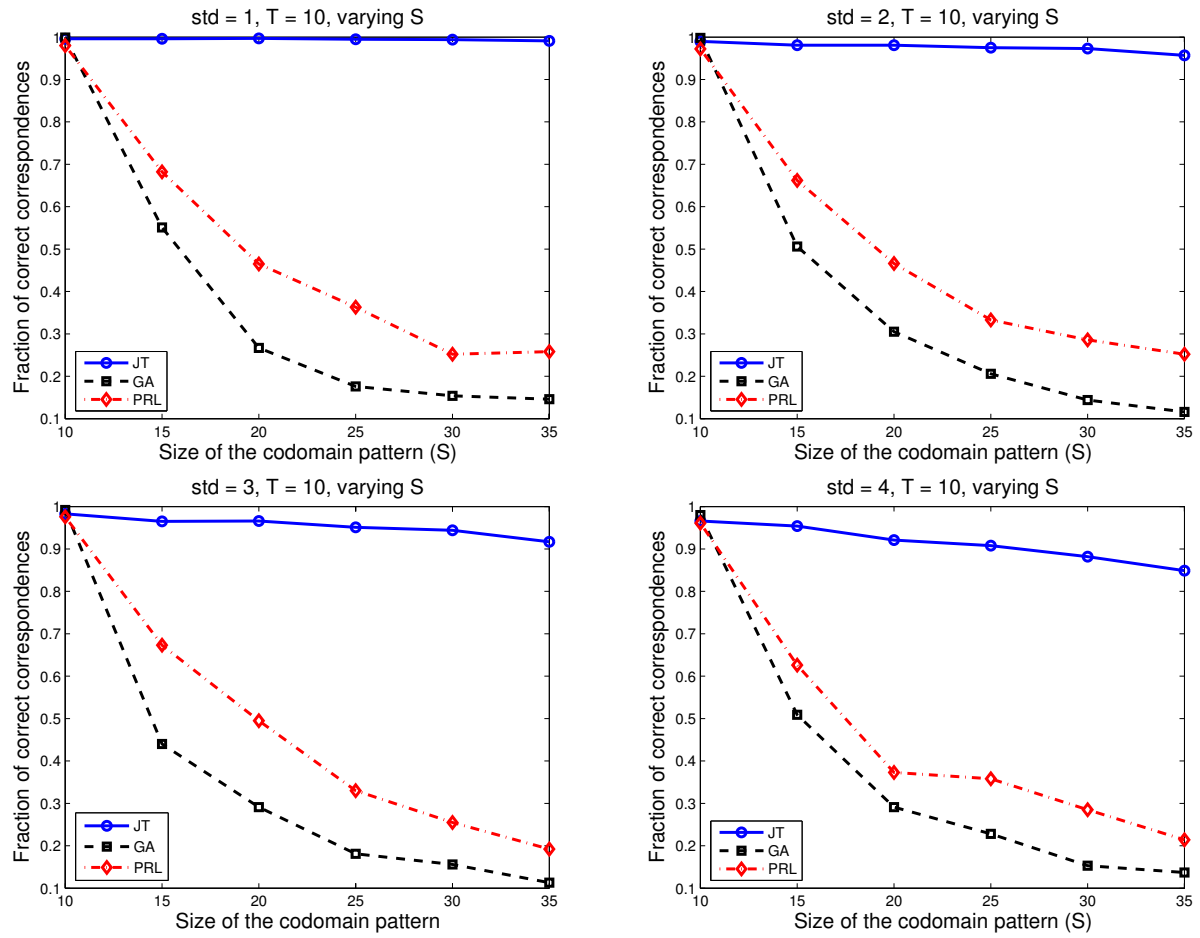

Fig. 10. Comparison of JT, GA and PRL for matching under varying relative sizes. Results for various levels of jitter (std = 1, 2, 3 and 4).

noise, JT is more robust with respect to increases in the sizes of the patterns, across a wide range of jitter. SB is only satisfactory in the uninteresting case of equal pattern sizes *and* virtually no jitter. Despite the fact that it only gives theoretical guarantee of optimality in the noiseless limit (which was confirmed in the experiments by noting the perfect performance in all cases where std = 0), the proposed technique, JT, presents excellent performance for small jitter, regardless of the pattern sizes. Even for high jitter, it significantly outperforms the alternative techniques if the pattern sizes are significantly different. This is a very important result: in real applications where either (i) we have equal pattern sizes and the jitter is small, or (ii) we have different pattern sizes (regardless of jitter), this approach finds its best applicability.

B. Real-world data

We also conducted experiments on real image data to evaluate the techniques on a realistic Computer Vision problem. In the real-world experiments, we used the CMU house sequence available at http://vasc.ri.cmu.edu/idb/html/motion/house/index.html. This database consists of 111 frames of a moving sequence of a toy house. We matched all images spaced by 10, 20, 30, 40, 50, 60, 70, 80, 90 and 100 frames and computed the average correct correspondence. Since there are 111 frames, note that the number of image pairs spaced by these amount of frames are, respectively, 101, 91, ..., 11.

Figure 11 shows typical images separated by these quantities of frames.

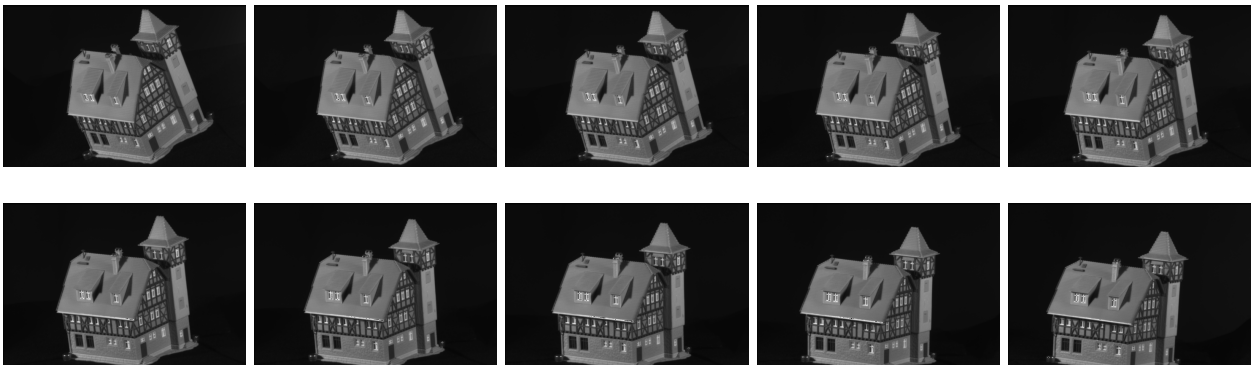

Fig. 11. Images from the CMU house sequence (top row: frames 1, 11, ..., 41; bottom row: frames 51, 61, ..., 91.)

Since all the images have size 384 $\times$ 576 (contrary to the synthetic experiments, where the point sets lied on $x, y = [0, 1]$), we need to use, accordingly, a large value for σ (see Eq. 7) that does not underflow the kernel computation for very large deviations in the pairwise assignments. Here we used the value $\sigma = 150$. Also, as in the synthetic experiments, the 3 reference points for the 3-tree were selected randomly.

Figure 12 shows examples of correspondences between different views of the same object obtained with the proposed technique for (i), narrow baseline (top) and (ii), wide baseline (bottom). For the narrow baseline case, all points are matched correctly in this example, whereas for the wide baseline case there are mistakes induced by the fact that rigid motion in the plane is no longer a reasonable assumption.

In total, 30 landmark points were manually marked in each of the images. We then conducted

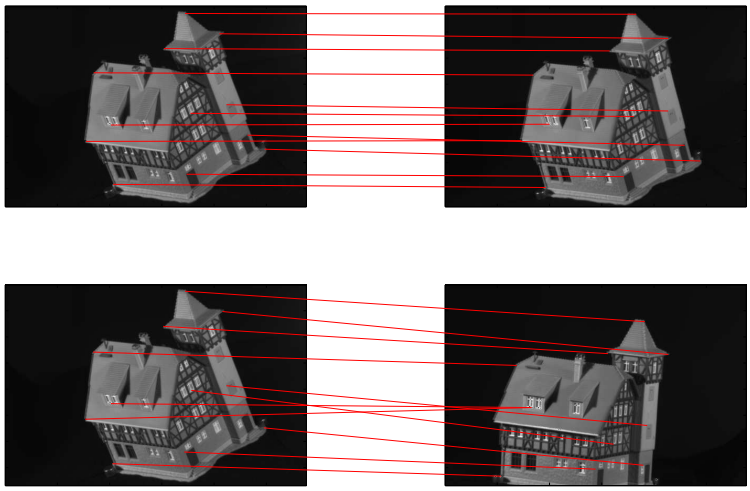

Fig. 12. Correspondences obtained with the proposed algorithm. Top: narrow baseline. Bottom: wide baseline. Note that several mistakes occur for wide baseline, since the isometric assumption is no longer appropriate.

four different experiments. We matched 15 against 30, 20 against 30, 25 against 30 and finally 30 against 30 points for every image pair in the experimental setting defined above. This allows us to evaluate how the techniques perform in a real problem with different point set sizes. For the 30×30 case, we run all 4 techniques (JT, GA, PRL and SB), whereas for the remaining cases SB was not used since it is not suitable for patterns of different sizes, as already mentioned.

Figure 13 shows the results for these experiments. The average value is taken over different spacings between image pairs in the frame sequence. Here we can observe that, as the relative sizes of the patterns become progressively different, the advantage of JT over the other techniques increases, which agrees with the synthetic experiments. For the reported values of different pattern sizes, JT performs significantly better than the competing methods, even for a wide baseline. Although the isometric assumption clearly does not hold for a wide baseline, we note that the same pairwise distances were used as features in *every* algorithm, so we expect this to be a fair comparison.[8] For the experiment with patterns of the same size (30×30), JT has similar performance to GA and PRL when the baseline is increased. For the narrow baseline case, JT

[8]It should be clear that in real problems, any of these approaches, since they rely exclusively on distance features, will only be competitive in the narrow baseline case.

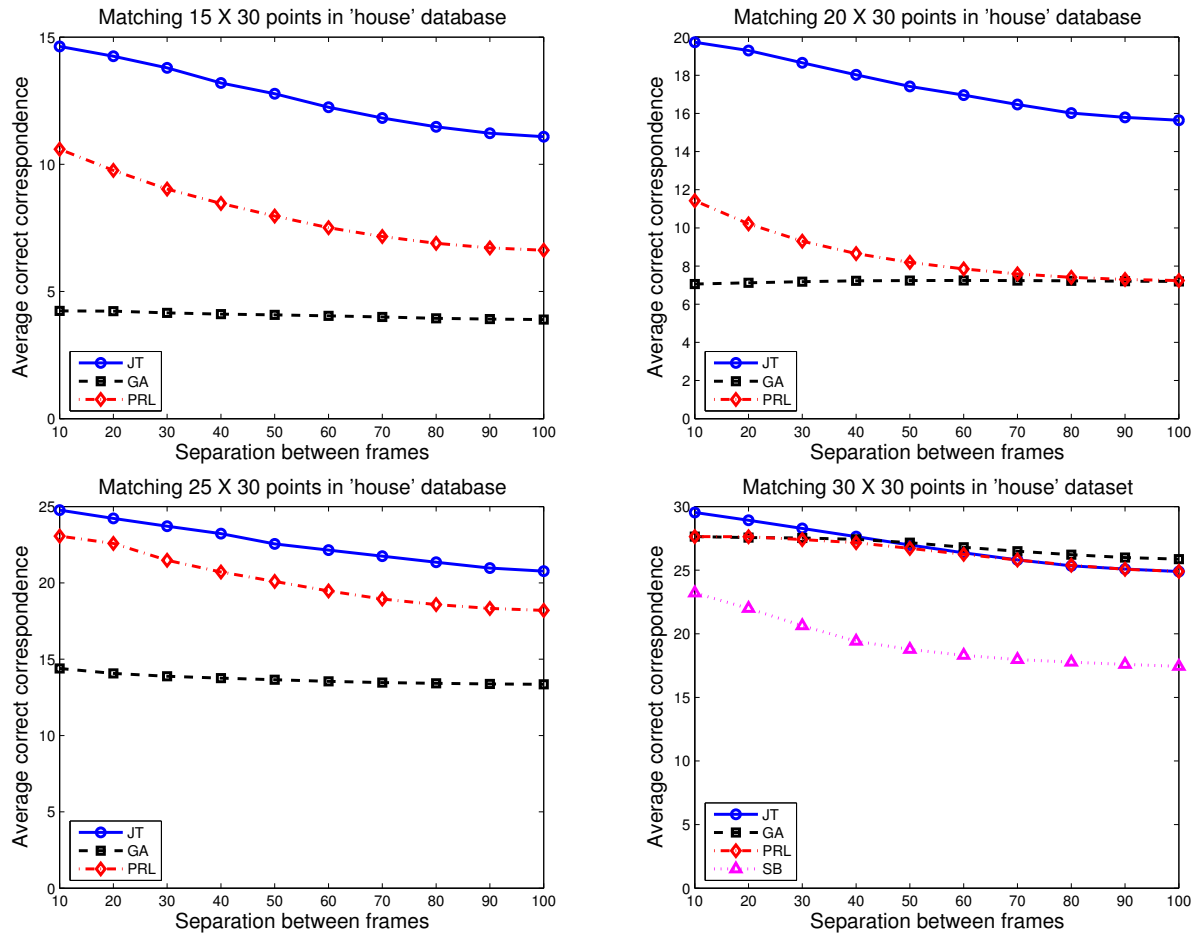

Fig. 13. Performances of JT, GA, PRL and SB in the CMU houses sequence for increasing baselines (10 to 100 frame separation) and different domain/codomain sizes (15/30, 20/30, 25/30, 30/30).

slightly outperforms the competing methods.

Our results in the real-world experiments lead us to conclude that, in the particular real-world problem of stereo correspondence, JT finds its best applicability in the narrow baseline case for patterns of different sizes where the isometric assumption is guaranteed to hold (apart from jitter of course). This finding agrees with the synthetic experiments.

Given the positive results on both synthetic and real-world experiments where there is a narrow baseline, we expect that in other applications, like those involving matching of star constellations, flexible ligands and protein motifs (where the isometric assumption also holds with good approximation) the proposed technique should perform similarly well.

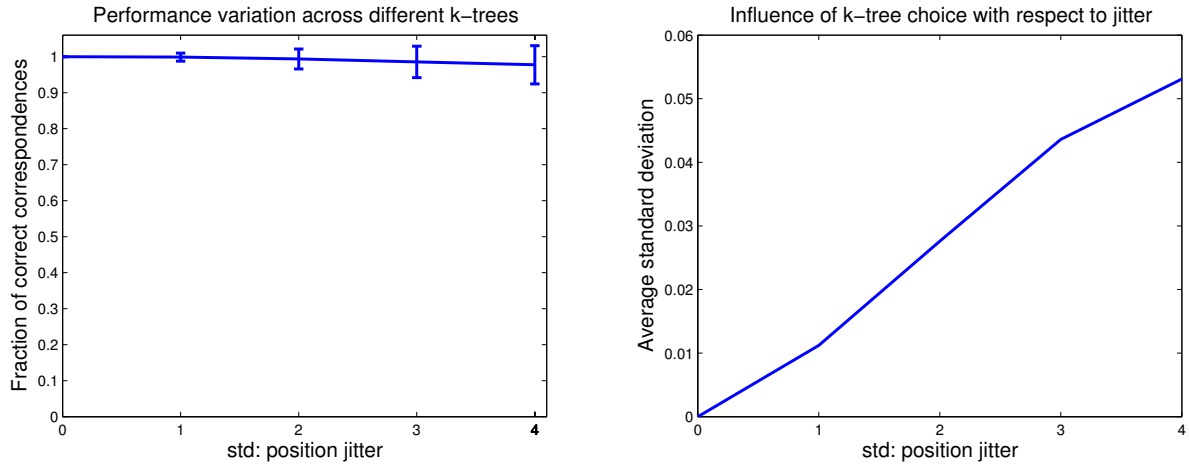

Fig. 14. Variation in performance of JT when different k-trees are used: matching two patterns of 20 nodes each. For each jitter level, 100 experiments were carried out (100 different domain-codomain point sets). For each of these, 100 random k-trees were generated and for each of these the JT algorithm was run. **Left:** each error bar corresponds to the average, over the 100 experiments, of the standard deviation computed over the 100 different k-trees of each experiment; **Right:** the growth of the variance is plotted explicitly. Note the linear behavior.

C. Empirical performance evaluation for varying k-trees

We mentioned previously that there is a theoretical problem that remains unsolved in this framework: how to select the k-tree when there is position jitter. When there is no jitter, any k-tree will find the same—optimal—solution. However, when jitter is present, different k-trees can result in different accuracy. Here we provide empirical evidence of this fact by measuring the variance of the performance of a matching task over a range of possible choices of k-trees.

Figure 14 shows the results of our experiments. We used the same setting of the previous synthetic experiments: random point patterns in $[0, 1]^2$. We randomly generated 100 domain-codomain pairs in this range, each with 20 points. For each of these 100 configurations, we randomly selected 100 3-trees for the domain pattern. Finally, the JT algorithm was run in each of these 100 3-trees. This allows us to measure the performance variability within the same pair domain-codomain over a wide selection of 3-trees. We simply computed the standard deviation over the 100 runs in each pair and averaged the results over the 100 configurations. This "average standard deviation" is plotted in the error bars in Figure 14-Left.

We observe from Figure 14-Right something that was already expected: the performance variance increases with jitter. In particular, it is zero for zero jitter, confirming the theoretical

results. Note that the average standard deviation grows only linearly with respect to the jitter.

D. Processing Times

The computational complexity of the proposed method is higher than in the other approaches. Spectral, graduated assignment and relaxation methods are, respectively, $O(T^3)$ (S=T), $O(T^2S^2)$ and $O(T^2S^3)$, while the proposed Junction Tree approach is $O(TS^4)$ (for matching in $\mathbb{R}^2$). However, the graphs showing real processing times (see Figure 15) indicate that even for a reasonable size, like 40, the actual running time is just twice that of PRL and 4 times that of GA. For graphs with about 30 nodes, the technique is about as fast as relaxation labelling.

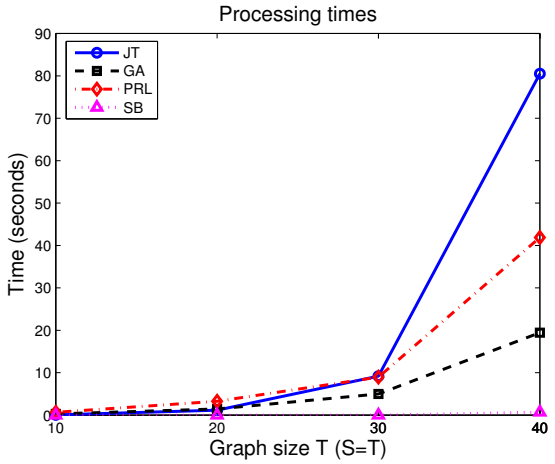

Fig. 15. Processing times for JT, GA, PRL and SB all in MATLAB implementations running on a Pentium 4, 3.2 GHz and 1GB of RAM.

VIII. Discussion and Future Work

A matching algorithm should, ideally, present high robustness with respect to jitter as well as with respect to size increases of the patterns. Our experiments revealed, essentially, two things. First, for matching patterns of the same size, the proposed method is very robust to small and moderate position jitter and reasonably robust to high position jitter. Second, and more important, the method is extremely robust to increasing differences in the pattern sizes. In experiments where the sizes of the two patterns are significantly different, the performance of JT is by far superior to that of the alternative methods. This is an important result, because in many

relevant application domains the problem of finding a "small" model within a "large" scene is of primary concern. (A typical such scenario that arises in Computer Vision is model-based object recognition in cluttered scenes.) We believe that the results presented in this paper indicate a possible direction in the search for robust algorithms for subgraph matching, where the sizes of the graphs can differ significantly.

This research has opened a wide field of investigation. There are several ways in which the current work can be extended. First, by considering higher-order potentials one might be able to cope with more complex invariances, such as invariance to affine transformations. Second, non-rigid matching might be attainable by augmenting the clique potentials with terms that allow for some kind of nonlinear transformation. Third, theoretical results on the accuracy of the method for the noisy case can be investigated. Fourth, the framework should be extended to deal robustly with outliers. Also, a deeper understanding of the noisy case might lead to a principled technique for the k-tree selection problem.

IX. Conclusion

This paper proposed a new solution to the rigid point pattern matching problem where jitter is allowed. The approach consisted of modelling the point matching task as a weighted graph matching problem and solving it using exact probabilistic inference in an appropriately designed Graphical Model. By using graph rigidity arguments, we showed that this Graphical Model, while allowing for exact MAP computation in polynomial time, still remains equivalent to the fully connected model in the noiseless limit. Contrary to many alternative heuristic approaches, the method we obtain is built from first principles (it has no intrinsic parameters), is non-iterative, obtains results independent of initialization, and provably finds a global optimum in polynomial time in the exact matching case. For inexact matching, our experiments indicate that the proposed technique is more accurate than standard methods when matching patterns of different sizes.

Acknowledgements

National ICT Australia is funded by the Australian Government's *Backing Australia's Ability* initiative, in part through the Australian Research Council. This work was initiated when the first three authors were with the Department of Computing Science, University of Alberta. We would like to thank NSERC, CFI and the Alberta Ingenuity Centre for Machine Learning for their support. The first author thanks CAPES for financial support. Also, we kindly thank X. Chen for a discussion that lead to the idea behind the proof of Lemma 1.

REFERENCES

[1] R. Myers, R. C. Wilson, and E. R. Hancock, "Bayesian graph edit distance," *IEEE Trans. on PAMI*, vol. 22, no. 6, pp. 628–635, 1999.

[2] M. Carcassoni and E. R. Hancock, "Spectral correspondence for point pattern matching," *Pattern Recognition*, vol. 36, pp. 193–204, 2003.

[3] H. Wang and E. R. Hancock, "A kernel view of spectral point pattern matching," in *International Workshops SSPR & SPR*, 2004, pp. 361–369.

[4] M. T. Goodrich and J. S. B. Mitchell, "Approximate geometric pattern matching under rigid motions," *IEEE Trans. on PAMI*, vol. 21, no. 4, pp. 371–379, 1999.

[5] F. Murtagh, "A new approach to point-pattern matching," *Astronomical Society of the Pacific*, vol. 104, no. 674, pp. 301–307, 1992.

[6] G. Weber, L. Knipping, and H. Alt, "An application of point pattern matching in astronautics," *Journal of Symbolic Computation*, no. 11, pp. 1–20, 1994.

[7] Y. Martin, M. Bures, E. Danaher, J. DeLazzer, and I. Lico, "A fast new approach to pharmacophore mapping and its application to dopaminergic and bezodiazepine agonists," *J. of Computer-Aided Molecular Design*, vol. 7, pp. 83–102, 1993.

[8] P. W. Finn, L. E. Kavraki, J.-C. Latombe, R. Motwani, C. Shelton, S. Venkatasubramanian, and A. Yao, "Rapid: Randomized pharmacophore identification for drug design," *Computational Geometry*, vol. 10, pp. 263–272, 1998.

[9] T. Akutsu, K. Kanaya, A. Ohyama, and A. Fujiyama, "Point matching under non-uniform distortions," *Discrete Applied Mathematics. Special Issue: Computational biology series issue IV*, pp. 5–21, 2003.

[10] R. Nussinov and H. J. Wolfson, "Efficient detection of three-dimensional structural motifs in biological macromolecules by computer vision techniques," *Proc. Natl. Acad. Sci.*, vol. 88, pp. 10 495–10 499, 1991.

[11] S. Gold and A. Rangarajan, "A graduated assignment algorithm for graph matching," *IEEE Trans. PAMI*, vol. 18, no. 4, pp. 377–388, 1996.

[12] T. S. Caetano, T. Caelli, and D. A. C. Barone, "Graphical models for graph matching," in *IEEE International Conference on Computer Vision and Pattern Recognition*, Washington, DC, 2004, pp. 466–473.

[13] Y. Keselman, A. Shokoufandeh, M. F. Demirci, and S. Dickinson, "Many-to-many graph matching via metric embedding," in *International Conference on Computer Vision and Pattern Recognition*, vol. 1, 2003, pp. 18–20.

[14] M. Demirci, A. Shokoufandeh, S. Dickinson, Y. Keselman, and L. Bretzner, "Many-to-many feature matching using spherical coding of directed graphs," in *European Conference on Computer Vision*, 2004, pp. 322–335.

[15] J. Ullman, "An algorithm for subgraph isomorphism," *Journal of the ACM*, vol. 23, no. 1, pp. 31–42, 1976.

[16] B. T. Messmer and H. Bunke, "A new algorithm for error-tolerant subgraph isomorphism detection," *IEEE Trans. PAMI*, vol. 20, no. 5, pp. 493–503, 1998.

[17] S. Berreti, A. D. Bimbo, and E. Vicario, "Efficient matching and indexing of graph models in content-based retrieval," *IEEE Trans. PAMI*, vol. 23, no. 10, pp. 1089–1105, 2001.

[18] K. S. Fu, "A step towards unification of syntatic and statistical pattern recognition," *IEEE Trans. PAMI*, vol. 5, no. 2, pp. 200–205, 1983.

[19] M. Eshera and K. Fu, "A graph distance measure for image analysis," *IEEE Transactions on Systems Man and Cybernetics*, vol. 14, no. 3, pp. 353–363, 1984.

[20] W. H. Tsai and K. S. Fu, "Subgraph error-correcting isomorphisms for syntactic pattern recognition," *IEEE Trans. on Systems, Man and Cybernetics*, vol. 13, no. 1, pp. 48–62, 1983.

[21] K. L. Boyer and A. C. Kak, "Structural stereopsis for 3-d vision," *IEEE Trans. on PAMI*, vol. 10, no. 2, pp. 144–166, 1988.

[22] B. Bhanu and O. D. Faugeras, "Shape matching of two-dimensional objects," *IEEE Trans. PAMI*, vol. 6, no. 2, pp. 137–156, 1984.

[23] L. S. Davis, "Shape matching using relaxation techniques," *IEEE Trans. PAMI*, vol. 1, no. 1, pp. 60–72, 1979.

[24] O. D. Faugeras and M. Berthod, "Improving consistency and reducing ambiguity in stochastic labeling: an optimization approach," *IEEE Trans. PAMI*, vol. 3, pp. 412–423, 1981.

[25] S. Ulmann, "Relaxation and constraint optimization by local process," *Computer Graphics and Image Processing*, vol. 10, pp. 115–195, 1979.

[26] A. Rosenfeld and A. C. Kak, *Digital Picture Processing*. New York, NY: Academic Press, 1982.

[27] W. J. Christmas, J. Kittler, and M. Petrou, "Structural matching in computer vision using probabilistic relaxation," *IEEE Trans. PAMI*, vol. 17, no. 8, pp. 749–764, 1994.

[28] S. Z. Li, "A markov random field model for object matching under contextual constraints," in *International Conference on Computer Vision and Pattern Recognition*, 1994, pp. 866–869.

[29] R. A. Hummel and S. W. Zucker, "On the foundations of relaxations labeling process," *IEEE Trans. on PAMI*, vol. 5, no. 3, pp. 267–286, 1983.

[30] A. Rosenfeld, R. Hummel, and S. Zucker, "Scene labeling by relaxation operations," *IEEE Trans. on Systems, Man and Cybernetics*, vol. 6, pp. 420–433, 1976.

[31] R. C. Wilson and E. R. Hancock, "Structural matching by discrete relaxation," *IEEE Trans. PAMI*, vol. 19, no. 6, pp. 634–648, 1997.

[32] E. Hancock and R. C. Wilson, "Graph-based methods for vision: A yorkist manifesto," *SSPR & SPR 2002, LNCS*, vol. 2396, pp. 31–46, 2002.

[33] J. V. Kittler and E. R. Hancock, "Combining evidence in probabilistic relaxation," *Int. Journal of Pattern Recognition and Artificial Intelligence*, vol. 3, pp. 29–51, 1989.

[34] S. Umeyama, "An eigen decomposition approach to weighted graph matching problems," *IEEE Trans. PAMI*, vol. 10, pp. 695–703, 1998.

[35] L. Shapiro and J. Brady, "Feature-based correspondence - an eigenvector approach," *Image and Vision Computing*, vol. 10, pp. 283–288, 1992.

[36] B. J. van Wyk and M. A. van Wyk, "Kronecker product graph matching," *Pattern Recognition*, vol. 36, no. 9, pp. 2019–2030, 2003.

[37] M. A. van Wyk, T. S. Durrani, and B. J. van Wyk, "A rkhs interpolator-based graph matching algorithm," *IEEE:PAMI*, vol. 24, no. 7, pp. 988–995, 2002.

[38] A. Robles-Kelly and E. R. Hancock, "Graph edit distance from spectral seriation," *IEEE Trans. on PAMI*, vol. 27, no. 3, pp. 365–378, 2005.

[39] P. N. Suganthan, "Structural pattern recognition using genetic algorithms," *Pattern Recognition*, vol. 35, pp. 1883–1893, 2002.

[40] M. Pelillo, "Replicator equations, maximal cliques, and graph isomorphism," *Neural Comput.*, vol. 11, pp. 1933–1955, 1999.

[41] M. Pelillo, K. Siddiqi, and S. Zucker, "Matching hierarchical structures using association graphs," *IEEE Trans. PAMI*, vol. 21, no. 11, pp. 1105–1120, 1999.

[42] D. Conte, P. Foggia, C. Sansone, and M. Vento, "Thirty years of graph matching in pattern recognition," *Special Edition of the International Journal of Pattern Recognition and Artificial Intelligence on Graph Theory in Vision*, vol. 18, no. 3, pp. 265–298, 2004.

[43] T. Caelli and T. Caetano, "Graphical models for graph matching: approximate models and optimal algorithms," *Pattern Recognition Letters*, vol. 26, pp. 339–346, 2004.

[44] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge, UK: Cambridge Univesity Press, 2000.

[45] J. Mejía, T. Villela, and J. Braga, "The ccd stellar sensor of the masco telescope pointing system," *Advances in Space Research*, vol. 26, no. 9, pp. 1407–1410, 2000.

[46] P. J. de Rezende and D. T. Lee, "Point set pattern matching in d-dimensions," *Algorithmica*, vol. 13, pp. 387–404, 1995.

[47] H. Chui and A. Rangarajan, "A new algorithm for non-rigid point matching," in *CVPR*, vol. 2, 2000, 44–51.

[48] A. Rangarajan, A. Yuille, and E. Mjolsness, "Convergence properties of the softassign quadratic assignment algorithm," *Neural Computation*, vol. 11, pp. 1455–1474, 1999.

[49] R. Sinkhorn, "A relationship between arbitrary positive matrices and doubly stochastic matrices," *Ann. Math. Statis.*, vol. 35, pp. 876–879, 1964.

[50] T. Caelli and S. Kosinov, "An eigenspace projection clustering method for inexact graph matching," *IEEE Trans. PAMI*, vol. 26, no. 4, pp. 515–519, 2004.

[51] A. Robles-Kelly and E. R. Hancock, "String edit distance, random walks and graph matching," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 18, no. 3, pp. 315–327, 2004.

[52] T. S. Caetano, T. Caelli, and D. A. C. Barone, "An optimal probabilistic graphical model for point set matching," in *International Workshops SSPR & SPR*, Lisbon, 2004, pp. 162–170.

[53] J. Dattorro, "Euclidean distance matrices," Ph.D. dissertation, Stanford University, 2005.

[54] H.-X. Huang, Z.-A. Liang, and P. M. Pardalos, "Some properties of the Euclidean distance matrix and positive semidefinite matrix completion problems," *Journal of Global Optimization*, vol. 25, pp. 3–21, 2003.

[55] M. J. Wainwright, "Stochastic processes on graphs with cycles: Geometric and variational approaches," Ph.D. dissertation, Department of EECS, Massachusetts Institute of Technology, 2002.

[56] M. J. Wainwright and M. I. Jordan, "A variational principle for graphical models," in *New directions in statistical signal processing*, to appear in 2005.

[57] M. I. Jordan, *An Introduction to Probabilistic Graphical Models*, in preparation.

[58] ——, "Graphical models," *Statist. Sci.*, vol. 19, no. 1, pp. 140–155, 2004.

[59] S. L. Lauritzen, *Graphical Models*. New York, NY: Oxford University Press, 1996.

[60] T. S. Caetano, T. Caelli, and D. A. C. Barone, "A comparison of junction tree and relaxation algorithms for point matching using different distance metrics," in *IEEE International Conference on Pattern Recognition*, vol. 2, Cambridge, UK, 2004, pp. 124–127.

[61] R. Connelly, "Rigidity and energy," *Invent. Math.*, vol. 66, no. 1, pp. 11–33, 1982.

[62] D. B. West, *Introduction to Graph Theory*. Upper Saddle River, NJ: Prentice Hall, 2001.

[63] L. R. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.