Graph and point cloud registration for tree-like structures: survey and evaluation

Miguel Amável PINHEIRO

Center for Machine Perception, Faculty of Electrical Engineering, Czech Technical University in Prague, Czech Republic

amavemig@cmp.felk.cvut.cz

Abstract. Graph and point cloud registration is a recurrent problem in computer vision related applications. In particular, we study the application of this problem for tree-like structures extracted from medical imaging.

Often the texture information present in vessel and vascular structures is insufficient in order to match content in two or more images. One approach to this problem is to segment the structure, and match its geometric properties between the images. This can be achieved using point cloud or graph matching techniques.

A number of algorithms was tested in order to compare their performance on synthetically generated data. In this manuscript, we present a detailed description of each tested method and of these experiments.

Keywords

Graph matching, point cloud matching, medical imaging, image registration

1. Introduction

Current methods for registering pairs of images featuring vessels, vascular structures or other tree-like structures use different types of information that is present in the images. One can match image information using texture or intensity information [1], optimizing the matching defining an energy criterion and finding its minimum. Further algorithms make use of image descriptors in order to find key features within images, enabling a more robust matching between two pictures, such as SIFT descriptors or salient feature regions [2]. This is possible whenever the images to be registered present useful common texture information that can be used on the image registration.

Images in the medical imaging field have generally a very low level of similarity in texture. Such is the case when registering images extracted with electron microscopy (EM) and light microscopy (LM). These images also present very different resolutions – the resolution of a good LM is about 300 nm and the one of an EM can fall below 1 Ångström(reaching a subatomic level) [3, 4] – in our data it is a few nm. Although the EM images give us a better resolution of the observed tissue, the LM images help us obtain a bigger picture and a better overall map, and therefore it is useful to match both. The lack of common texture information in images is also present in images extracted with different devices such as in images of the retina fundus [5, 6]. In order to surpass these difficulties, a common approach has been to segment the structures present on the images, followed by the matching of those structures based on their geometric properties.

These structures can be described by different properties. In some cases authors choose to describe them using landmarks [5], graphs [7] or trees. This decision is often connected to the quality of the images and the segmentation approach. Nonetheless, the matching approach will then take the form of a point, graph or tree matching. We are particularly interested in graph matching, since it covers most applications in medical imaging.

There are several other scenarios where graph matching is used, such as other medical imaging, cartography and character recognition. Particularly in medical imaging, we can apply these approaches to match images of retina, lung or heart vessels. Matching of retina and lung vessel images are particularly interesting in the case of building an entire map of the vessel structure, since one frame is not enough to observe it entirely, as in the case of blood or heart vessels.
one is interested to observe the changes in the structure over time.

In this manuscript, we present a detailed description of the state-of-the-art methods for graph and point cloud matching, as well as a evaluation of these methods, using synthetic data.

A variety of matching algorithms have been presented in the past years. Some were specifically developed towards the problem of medical image registration and others have been presented as general approaches for graph and point cloud matching. The tested methods included in this report were the following:

- Point matching
  - Coherent Point Drift (CPD) [8]
  - Iterative Closest Point (ICP) [9]
  - Iterative Closest Reciprocal Point (ICRP) [10]
  - Thin-plate spline - robust point matching (TPS) [11]
  - Random Sample Consensus (RANSAC) [12]
  - Shape Context (SHAPEC) [13]
- Graph matching
  - Spectral Matching with Affine Constraints (SMAC) [14]
  - A Path Following Algorithm (PATH) [15]

2. Point cloud matching

2.1. RANSAC

The RANSAC method was first presented in 1981 [12], and it is a method for fitting a model to experimental data. It is applicable not only for the point matching problematic, but it can be adapted to our purposes.

The approach selects $s$ sample points in each iteration $i$ from both point clouds $x_A$ and $x_B$, which we are going to refer to as $s_A^{(i)}$ and $s_B^{(i)}$. In order to avoid problems with different sampling values, and mainly to decrease the number of points in the problem, we take $x_A$ and $x_B$ to be the branching and leaf nodes of the graphs (see Fig. 2 in Sec. 5).

We then calculate a transformation $T^{(i)}$ based on these two set of points as

$$T^{(i)} = \left[ \begin{array}{c} s_A^{(i)} \\ 1 \end{array} \right] \left[ \begin{array}{c} s_B^{(i)} \\ 1 \end{array} \right]^{-1},$$

where $\mathbb{1}$ is a vector of ones with the size of each respective set. We fit this transformation to all the points $T^{(i)}(x_B) = T^{(i)}x_B$, and count how many inliers this transformation computes. We do this by counting how many points in $T^{(i)}(x_B)$ have at least one point of $x_A$ at a distance of at most $\alpha$, to which we refer as inliers. The transformation which produces the most inliers is the output of the approach.

The number of iterations that is required to have a probability of $p$ of finding the optimal solution is estimated as

$$N = \frac{\log (1 - p)}{\log (1 - p_i)},$$

where $p_i$ is the percentage of inliers obtained with transformation $T^{(i)}$ at iteration $i$. $N$ is updated when the method finds the highest number of inliers so far.

2.2. TPS

The TPS-RPM algorithm was introduced in 2003 [11], in a paper co-authored by one of the authors of the softassign algorithm [16]. The method presents many of the ideas of previous work, such as the correspondence matrix with an extra row and column to identify outliers and a very similar objective function to minimize. A new entropy term $T\sum_{j=1}^{m} \sum_{k=1}^{N} m_{jk} \log m_{jk}$ is introduced into the criterion, where $T$ is called a temperature parameter and the remaining variables are defined as in softassign. As the algorithm iterates, the temperature is reduced, controlling the level of convexity of the objective function. The $g(A)$ function is replaced by a new operator $L$, with $g(A) = -\|[LA]\|^2$, which is referred to as a smoothness measure. Furthermore, the method also analyses a non-rigid case represented by a function

$$A(Y_k, d, w) = Y_k \cdot d + \phi(Y_k) \cdot w,$$

where $d$ represents a $(D + 1) \times (D + 1)$ affine transformation matrix, where $D$ is the number of dimensions, $w$ is a $K \times (D + 1)$ warping coefficient matrix and each $\phi(Y_k)$ is a $1 \times K$ vector of thin plate splines basis functions, $\phi(Y_k) = ||Y_k - Y_k||^2 \log ||Y_k - Y_k||$ for $b = 1, ..., K$. With this notation it is possible to represent non-rigid transformations.

2.3. CPD

In 2010, Myronenko and Song presented an alignment technique for rigid, affine and non-rigid transformation cases, called Coherent Point Drift [8]. The authors look at the task as a probability density estimation problem, with one of the sets being data points $X = (x_1, ..., x_N)^T$, and the second one representing Gaussian Mixture Model (GMM) centroids $Y = (y_1, ..., y_M)^T$. The GMM probability density is therefore

$$p(x) = \sum_{m=1}^{M+1} P(m)p(x|m),$$
where

\[ P(m) = \begin{cases} 
(1 - \omega) \cdot 1/M & \text{if } m \neq M + 1 \\
\omega \cdot 1/N & \text{if } m = M + 1
\end{cases}, \]

\[ p(x|m) = \frac{1}{(2\pi\sigma^2)^{D/2}} \exp\left(-\frac{|x - y_m|^2}{2\sigma^2}\right), \]

and where \( \sigma^2 \) is the variance for all GMM components, \( D \) the number of dimensions and \( \omega \) the weight for uniform distribution for \( M + 1 \), with \( 0 \leq \omega \leq 1 \). The added \( M + 1 \) accounts for outliers and noise present in the task. The objective function is the negative log-likelihood, leading to

\[ Q(\theta, \sigma^2) = \frac{1}{2\sigma^2} \sum_{n=1}^{N} \sum_{m=1}^{M} P(m|x_n)||x_n - T(y_m, \theta)||^2 \]

\[ + \frac{D}{2} \sum_{n=1}^{N} \sum_{m=1}^{M} P(m|x_n) \log \sigma^2 \]  

(5)

where \( T(y_m, \theta) \) is the transformation of the point \( y_m \) using parameters \( \theta \).

An expectation maximization algorithm is used. In the E-step, the values for \( P(m|x) \) are reassigned and normalized. In the M-step, the transformation’s parameters and the variance of the density functions are updated, using the new value for \( P \).

The article goes on to describe different and more detailed approaches to calculate \( T(y_m, \theta) \) for the rigid and affine cases. In the rigid case, the transformation is represented by a rotation matrix \( R \) and a scaling parameter \( s \) together with a translation vector \( t \), i.e. \( T(y_m; R, t, s) = sRy_m + t \). For the affine case, the transformation is represented by a general transformation matrix \( B \) and a translation vector, i.e. \( T(y_m; B, t) = By_m + t \).

A non-rigid approach is also presented. Here, the transformation takes the shape \( T(Y, v) = Y + v(Y) \), where \( v \) is a displacement function. The objective function is changed and a regularization term is added, leading to

\[ Q(v, \sigma^2) = \frac{1}{2\sigma^2} \sum_{n=1}^{N} \sum_{m=1}^{M} P(m|x_n)||x_n - (y_m + v(y_m))||^2 \]

\[ + \frac{D}{2} \sum_{n=1}^{N} \sum_{m=1}^{M} P(m|x_n) \log \sigma^2 + \frac{\lambda}{2} ||Lv||^2 \]  

(6)

where \( \lambda \) is a trade-off parameter for regularization and \( L \) is a regularization operator [17]. Taking a derivative w.r.t. \( v \) and equaling it to zero, we get equations for the minimum. The non-linear transformation turns out to be \( T(Y, v(Y')) = T(Y, W) = Y + GW \), where \( G \) is an \( M \times M \) matrix with elements \( g_{ij} = e^{-\frac{1}{2}\|\frac{y_m - y_n}{\sigma\sqrt{2\pi}}\|^2} \) and \( W \) is an \( M \times D \) matrix with elements \( w_m = \frac{1}{\sigma^2}\lambda \sum_{n=1}^{N} P(m|x_n)(x_n - (y_m + v(y_m)))\).

The value for \( \sigma^2 \) is also trivially obtained deriving the \( Q \) function. The algorithm is similar as in the affine and rigid cases.

2.4. ICP

The Iterative Closest Point is a very popular method and was first presented in 1992 by Besl and McKay [9]. In the method’s own notation, the task is to find a transformation with which the data points \( X \) can fit the model points \( P \) in a three-dimensional space (3D) – which can also be adapted to a 2D case. The method finds the closest point in \( P \) for each point of \( X \), denoted \( Y \) in an operation denominated \( C(P_k, X) \), i.e.

\[ Y_k = C(P_k, X). \]  

(7)

For each iteration \( k \), a set of points \( Y_k \) is computed as \( C(P_k, X) \). The algorithm then uses quaternion representation (although it can be extended to allow other types of transformations) to obtain the transformation between \( P_0 \) and \( Y_k \) described by a vector \( \hat{q}_k \). This transformation is applied to \( P_0 \) to obtain a new set of points, i.e. \( P_{k+1} = \hat{q}_k(P_0) \). These steps are performed repeatedly until the difference between the mean squared points matching errors (between \( X \) and \( P_k \)) for subsequent iterations \( d_k \) and \( d_{k+1} \) falls below a threshold \( \tau \).

2.5. ICRP

The Iterative Closest Reciprocal Point [10] algorithm is similar ICP, as the name implies. To improve robustness, the method only considers points which are reciprocally the closest points between the two sets.

2.6. SHAPEC

Shape Matching using Shape Context (SHAPEC) was proposed by Belongie et al. [13] and builds a local description for each point of each cloud. This description is composed of a histogram of neighbor points, where the bins are distributed depending on the angle and Euclidean distance between points. The similarity between two points is the absolute difference between the number of elements in each bin of each histogram.

This formulation allows for some nonlinearity, however it is variant to rotation, scale and shearing.

The assignment between points is obtained by simply applying the Hungarian algorithm to the similarities between the points on both sets.
3. Graph matching

3.1. SMAC

The paper presented in 2006 by Cour et al. [14], introduces a technique named SMAC where two graphs \( G = (V, E, A) \) and \( G' = (V', E', A') \) are matched. The method tries to obtain the mapping between the vertices \( V \) and \( V' \) and consequently also \( E \) and \( E' \), based on edge attributes \( A, A' \) gathered from the edges \( E \) and \( E' \).

The method builds a compatibility matrix \( W \) with the size \( |V||V'| \times |V||V'| \) where each entry describes the compatibility between the edges of the two graphs \(- W_{ij,j'j'} = f(A_{ij}, A_{i'j'}) \) where \( ij \in E \) and \( i'j' \in E' \).

Having built this matrix \( W \), the method needs to correctly assign the correspondences between edges of opposite graphs. Since the task is NP-complete, the proposed algorithm finds an approximate solution by Spectral Matching [18]. In summary, this method considers solely the edge information to match graphs, which is positive for situations where the Euclidean distance between vertices has not been significantly changed, i.e. the length of the edges remains similar – rotation and translation cases.

3.2. PATH

The path following algorithm (PATH) presented by Zaslavskiy et al. [15] uses a convex relaxation of quadratic programming, to find similarities between graph nodes. It relaxes the classic quadratic programming formulation \( x^T W x, x \in D \) where \( D \) is the set of all doubly stochastic matrices to a concave formulation with \( x \in \mathcal{P} \), where \( \mathcal{P} \) is the set of all permutation matrices. The relaxation is done iteratively and using Frank-Wolfe’s algorithm.

In order to build a graph, the paper proposes taking all nodes or points from a point cloud and connecting all neighbor nodes which are within a radius \( r \). The weight proposed for retina fundus graph matching for each edge is \( w_{ij} = \exp(-|i - j|) \) for \( i, j \in V \), where \( |i - j| \) is the Euclidean distance between the nodes. Although this formulation is clearly invariant to rotation and translation, it will not be invariant to other kinds of transformation.

4. Notation

Let \( G^A = (V^A, E^A) \) be an undirected acyclic graph without loops, where \( V^A \) represents the graph’s vertices and \( E^A \) its edges. A geometrical position in \( \mathbb{R}^D \) is associated with each vertex \( i \in V^A \), where \( D \) is the dimension of the space. Let us also define a subset \( K^A \subseteq V^A \) as the group of branching and leaf nodes (key nodes) of \( G^A \), i.e.

\[
K^A = \{ i \in V^A \mid \deg(i) > 2 || \deg(i) = 1 \},
\]

where \( \deg(i) \) is the degree of the vertex \( i \).

Let us also consider a second graph \( G^B = (V^B, E^B) \). This graph is obtained through a geometrical transformation \( T \) of the first graph, and therefore \( G^B = T(G^A) \). This transformation may be rigid, affine or nonlinear. Typically in medical imaging applications, this transformation is composed of a rigid component plus a nonlinear deformation. Furthermore, the transformation may not be isomorphic and therefore, vertices may be missing or added in \( G^B \). We keep the mapping \( C : V^A \rightarrow V^B \) between each node. The task is therefore to match the two graphs \( G^A \) and \( G^B \). Note that this can be achieved by solely matching the positions associated with \( V^A \) and \( V^B \), which is therefore a point cloud matching formulation – however it does not use all the information available.

5. Datasets

In order to thoroughly test and compare the performance of each method, several sets of graphs were synthetically generated. Each set has 200 pairs of graphs, where each pair has one graph \( G^A = (V^A, E^A) \) generated using randomly uniformly distributed points and the minimum spanning tree (MST) [19] of the points. The distribution of vertices that the MST generates is similar to the structure of the real data, hence the decision of using this approach. The other graph \( G^B = (V^B, E^B) \) of the pair is obtained through a transformation \( T(V^A) = V^A.T^T \) of the vertices \( V^A \), i.e. \( G^B = (T(V^A), E^A) \). For each of the sets, \( T \) takes different values, in order to test each method’s robustness towards
Fig. 3: Three examples of pairs of graphs transformed by rigid components and deformed using B-spline deformation. The deformation increases from left to right.

each type of transformation. The names and values for each set are as follows:

- **Rotation set** – \( T = R(\theta) = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \)

- **Scale set** – \( T = S(a,b) = \begin{bmatrix} e^a & 0 & 0 \\ 0 & e^b & 0 \\ 0 & 0 & 1 \end{bmatrix} \)

- **Shear set** – \( T = H(k_a,k_b) = \begin{bmatrix} 1 & k_a & 0 \\ k_b & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \)

- **Noisy set** – \( T = R + \phi(V_A) \)

- **Crop set**

where \( R \) is a rigid transformation constant for all pairs, and \( \phi(V_A) \) is an additional B-spline deformation – a few coefficients are randomly picked and increased with the number of graphs considered. For the crop set, the transformation is not only composed by a constant rigid transformation \( R \) constant for all pairs, but also branches of the graphs were cut in \( G^A \) and \( G^B \). An example taken from the noisy set is presented in Fig. 2.

For every pair of graphs, there is a mapping \( C : V_A \rightarrow V_B \), which is in fact not bijective in all sets since there are unpaired elements in the Crop set.

### 6. Results

#### 6.1. Success rate

In Fig. 4, we depict the correct correspondence rate extracted from the experiments for each method done over the synthetic datasets. Each line presents a smoothing average of the results of matching each pair of graphs. Together with these results, we present the percentage of experiments with perfect accuracy (where 100% correct correspondences were obtained) for each method and dataset in Table 1.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Rotation</th>
<th>Scale</th>
<th>Shear</th>
<th>Noisy</th>
<th>Crop</th>
</tr>
</thead>
<tbody>
<tr>
<td>RANSAC</td>
<td>100.0</td>
<td>100.0</td>
<td>100.0</td>
<td>3.5</td>
<td>95.0</td>
</tr>
<tr>
<td>CPD</td>
<td>16.0</td>
<td>98.5</td>
<td>31.5</td>
<td>2.0</td>
<td>9.0</td>
</tr>
<tr>
<td>ICP</td>
<td>15.5</td>
<td>2.5</td>
<td>1.0</td>
<td>0.0</td>
<td>10.0</td>
</tr>
<tr>
<td>ICRP</td>
<td>16.0</td>
<td>2.5</td>
<td>1.0</td>
<td>0.5</td>
<td>10.0</td>
</tr>
<tr>
<td>PATH</td>
<td>97.0</td>
<td>0.5</td>
<td>0.5</td>
<td>0.0</td>
<td>1.0</td>
</tr>
<tr>
<td>SHAPEC</td>
<td>0.0</td>
<td>19.5</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>SMAC</td>
<td>1.0</td>
<td>40.5</td>
<td>21.5</td>
<td>21.0</td>
<td>2.5</td>
</tr>
<tr>
<td>TPS</td>
<td>53.5</td>
<td>89.5</td>
<td>28.5</td>
<td>3.0</td>
<td>9.0</td>
</tr>
</tbody>
</table>

Tab. 1: Percentage of experiments with perfect accuracy

It is clear that RANSAC performs well under strictly affine component variation (rotation, scale, shear), however under more realistic constraints, it is not able to find an affine transformation that can compensate for the deformation. It also shows the best performance on the Crop dataset. TPS and CPD show a rather similar performance. The methods however strongly depend on the initial position of the graphs, showing poor performances at high rotation and shearing. ICP and ICRP also show similar performances between themselves, however well below TPS and CPD.

The SHAPEC approach shows a poor performance on the presence of rotation, which is the reason for the poor performance in the noisy and crop datasets, where we also impose a small rigid transformation. The method show indeed a good performance on sets with no rotation and a small deformation, however this is not the situation we are in many medical imaging applications.

The SMAC algorithm’s results show a very irregular variation for every dataset. In particular, for the rotation set, we expected all results to be at 100%. However, analyzing the output of the quadratic programming solved using spectral matching, we conclude that in most cases in which the
approach fails, the algorithm falls into some local maximum, and that the correct solution has a higher function value that the one obtained. In other cases, local symmetry between branches can allow for correspondences to be swapped, since the function value is the same for both cases.

The PATH approach presents as expected an invariance towards rotation, however the way it builds the graph based in Euclidean distances makes it variant to small differences in scale, shear, missing nodes and also deformation.

6.2. Time complexity

In Fig. 5, the dependence of time on the number of points in the graphs is shown. Note that some program implementations are more efficient than others, therefore the focus of this study should lie on the time complexity of the methods, rather than its absolute value.

It is clear that RANSAC shows an exponential increase with respect to the number of points. The fact that we take the key points (branching and leaf nodes) means that in the Figure we see for RANSAC a different range of number of points from the rest of the methods. It also shows a quite irregular increase, which is a direct consequence of the randomness of finding the true correspondence.

Analyzing the time complexity of the remaining approaches, we can conclude that CPD, TPS and SHAPEC have time complexities between quadratic and cubic times. As for the PATH and SMAC methods, we obtain time increases closer to cubic complexity. Both ICP and ICRP have close to linear complexity.

7. Conclusions

We can state that RANSAC shows a good approach for these problems. However, the time complexity that it presents, makes it impracticable for normal sized sets. For most cases with small amplitude deformation, CPD is a fast approach with good results. However, in cases with high rotation or where some nodes are missing, the method fails.
Analyzing the results, we can conclude that there is a need for a new method, which can solve cases with unknown rotation, high deformation and with missing nodes – this is the case of applications such as registering EM and LM images.

Acknowledgements

The research described in the paper was supervised by doc. Jan Kybic and supported by the Czech Science Foundation under the project P202/11/0111 and by the Grant Agency of the Czech Technical University in Prague under the grant SGS12/190/OHK3/3T/13. The author would also like to acknowledge the Fundação para a Ciência e Tecnologia (FCT) for the Ph.D. grant SFRH/BD/77134/2011.

References


About Authors...

Miguel Amável Pinheiro received a BSc and MSc degree from Faculdade de Engenharia da Universidade do Porto, Portugal. He is currently pursuing a PhD degree at Faculty of Electrical Engineering of the Czech Technical University in Prague. His research interests include graph and point cloud matching, image registration and medical imaging.