Pattern Recognition Letters 52 (2015) 53–58

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

# Low-rank matrix approximations for Coherent point drift ☆

Ján Dupej [a,b,*], Václav Krajíček [a], Josef Pelikán [a]

[a] Department of Software and Computer Science Education, Faculty of Mathematics and Physics, Charles University in Prague, Malostranské Náměstí 25, 11800 Prague, Czech Republic

[b] Laboratory of 3D Visualization and Analytical Methods, Department of Anthropology and Human Genetics, Faculty of Science, Charles University in Prague, Viničná 7, 12843 Prague, Czech Republic

ARTICLE INFO

*Article history:*
Received 20 February 2014
Available online 18 October 2014

*Keywords:*
Point set registration
Coherent point drift
Low-rank approximation
Nyström method

ABSTRACT

Coherent point drift (CPD) is a powerful non-rigid point cloud registration algorithm. A speed-up technique that allows it to operate on large sets in reasonable time, however depends on efficient low-rank decomposition of a large affinity matrix. The originally used algorithm for finding eigenvectors in this case is based on Arnoldi's iteration which, though very precise, requires the calculation of numerous large matrix-vector products, which even with further speed-up techniques is computationally intensive. We use a different method of finding that approximation, based on Nyström sampling and design a modification that significantly accelerates the preprocessing stage of CPD. We test our modifications on a variety of situations, including different point counts, added Gaussian noise, outliers and deformation of the registered clouds. The results indicate that using our proposed approximation technique the desirable qualities of CPD such as robustness and precision are only minimally affected, while the preprocessing times are lowered considerably.



## 1. Introduction

There are numerous areas of pattern recognition and computer graphics that rely on robust and fast point set registration algorithms. These include, among others, image registration, medical data analysis, stereo matching and many others. Generally speaking, a registration algorithm receives two point sets as its input and returns a deformation that optimally transforms one shape into the other, be it a parametric function or directly positions of the aligned points.

Registration algorithms can be categorized according to the kind of deformation they take into account when superimposing the two shapes. First, rigid registration only allows translation, rotation and scaling. On the other hand, non-rigid methods allow local deformations, usually consisting of piecewise rigid or affine transformations, though there are methods that do not use explicit functions to model their deformations such as [12] or [8].

Non-rigid registration is a most challenging task, even more so on real-world data. Every such algorithm should be able to deal with noise, missing and spurious false data while maintaining reasonable computational complexity. Another non-trivial problem is that the registration method is not aware of the deformation that is required to align the two objects. Very complex deformations are not even feasible as the optimization would suffer from local minima and non-convexity of the problem. Therefore simpler models are typically utilized, such as piecewise rigid or affine transformations [16]. In contrast, some methods like Coherent point drift (CPD) use displacement fields and thus calculate motion vectors for each vertex individually within some constraints.

We present speed-up techniques for CPD that make use of faster albeit less precise low-rank matrix approximation methods based on the sampling of the matrix's columns, otherwise known as the Nyström method.

The remainder of this paper is structured the following way: Section 2 discusses CPD and related research. Section 3 details our acceleration methods. Section 4 presents the results and finally, Section 5 concludes with a brief discussion.

## 2. Related work

### 2.1. Non-rigid registration algorithms

In this subsection we list some non-rigid registration algorithms related to CPD. An excellent survey of these methods is presented in [16].

Many modern registration methods use Gaussian mixture models (GMM) to model their source point clouds, such as one developed by [4] who represent the vertices in the target cloud as radially symmetric Gaussian kernels that compose a probability density estimate, though the authors state that other kernel shapes can be used.

☆ This paper has been recommended for acceptance by A. Marcelli.

* Corresponding author. Tel.: +420 221 914 240.
*E-mail address:* jdupej@cgg.mff.cuni.cz (J. Dupej).

http://dx.doi.org/10.1016/j.patrec.2014.10.005
0167-8655/

Myronenko et al. [13] presented the Coherent point drift method. They treat the source cloud as GMM centroids that are being fitted onto the target point cloud by maximizing a probability density function. The motion of the GMM centroids is smoothed and thus forced to be group-wise coherent in the maximization step of the EM algorithm [2].

A similar GMM-based algorithm has been proposed by Jian & Vemuri [8]. They, however use a closed-form expression of L2 distance between two Gaussian mixtures, which decreases the computational complexity.

Myronenko and Song [12] presented enhancements to their original algorithm consisting mostly of speed-up techniques. Partial eigenvector decomposition of a matrix is used to approximate a large matrix inversion and decrease the computational complexity of the iteration. Additionally, the width of GMM centroids is calculated without the use of simulated annealing.

Further improvements to CPD were proposed by [14] who alternate the roles of source and target point cloud in every iteration. Their approach shows some improvement in robustness on incomplete and noised data.

Recently another GMM-based approach was introduced by [15]. Unlike previous methods they use a bidirectional EM process and allow the GMM centroids to have different weights based on local features. Increased robustness to outliers is reported.

### 2.2. Coherent point drift

In this section we briefly reiterate the non-rigid version of CPD and identify the most obvious bottlenecks. A comprehensive description and analysis of this method is presented in [12]. CPD can perform rigid, affine and non-rigid registration. In the remainder of this paper we will only work with CPD in its non-rigid mode.

Before registration, the following parameters have to be set. Motion smoothing kernel width is controlled by $\beta$ ($\beta > 0$); small values will result in locally smooth transformations, while large values will average the individual vertex motions over large areas, turning the deformation into a pure global translation. The amount of regularization is set by $\lambda$ ($\lambda > 0$); small values favor data fitting while large values favor motion smoothing. A more detailed description of this parameter can be found in [19]. The parameter $w$ ($0 \leq w \leq 1$) provides an assumption concerning the expected amount of outliers and controls the sensitivity of correspondence probability to Euclidean distance. Large values enforce lower dependence of correspondence probability on point distance by making the distribution more uniform and thus the effect of outliers and noise is mitigated.

**Step 1:** Given the source point coordinates $\mathbf{Y} = (\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_m)^T$ and the target point coordinates $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n)^T$. Further initialize the matrix $\mathbf{T} = \mathbf{Y}$ and calculate the initial estimate of $\sigma^2$:

$$\sigma^2 = \frac{1}{dnm} \sum_{i,j=1}^{m,n} \|\mathbf{x}_j - \mathbf{y}_i\|^2. \tag{1}$$

**Step 2:** Construct the motion smoothing matrix $\mathbf{G}$.

$$g_{ij} = \exp\left(-\frac{1}{2\beta^2}\|\mathbf{y}_i - \mathbf{y}_j\|^2\right), \quad 1 \leq i, j \leq m. \tag{2}$$

**Step 3 (E-step):** Construct the correspondence matrix $\mathbf{P}$ according to target vertex coordinates $\mathbf{x}_i$ and the source vertex coordinates in the current iteration $\mathbf{t}_i$. Note the last term of the numerator that blends the match probabilities with a uniform distribution based on the value of $w$.

$$p_{ij} = \frac{\exp(-\frac{1}{2\beta^2}\|\mathbf{x}_j - \mathbf{t}_i\|^2)}{\sum_{k=1}^{m} \exp(-\frac{1}{2\beta^2}\|\mathbf{x}_j - \mathbf{t}_k\|^2) + \frac{wm(2\pi\sigma^2)^{d/2}}{(1-w)n}}. \tag{3}$$

**Step 4 (M-step):** Solve Equation (4) for $\mathbf{W}$ to find the motion of every target vertex in this iteration.

$$(\mathbf{G} + \lambda\sigma^2 d(\mathbf{P1})^{-1})\mathbf{W} = d(\mathbf{P1})^{-1}\mathbf{PX} - \mathbf{Y} \tag{4}$$

where $\mathbf{1}$ is a column vector of appropriate length whose all elements are 1. Recalculate the transformed point cloud for this iteration $\mathbf{T}$ by displacing the original source cloud with smoothed individual motions as follows:

$$\mathbf{T} = \mathbf{Y} + \mathbf{GW}. \tag{5}$$

Refine the estimate of $\sigma^2$ for the next iteration. If convergence has been reached, return the final transformed target point set $\mathbf{T}$ and the probabilities of correspondence of each pair of vertices from $\mathbf{X}$ and $\mathbf{Y}$ in $\mathbf{P}$. Otherwise proceed with Step 3.

There are two speed-up techniques proposed in [12]. One involves the use of fast Gaussian transform by [6] or the improved fast Gaussian transform by [18] for the calculation of the large matrix products $\mathbf{P1}$, $\mathbf{P}^T\mathbf{1}$ and $\mathbf{PX}$.

The other method uses low-rank approximation of $\mathbf{G}$ to create the matrices $\mathbf{Q}$ and $\mathbf{\Lambda}$. This way $\mathbf{G}$ can be approximated as $\hat{\mathbf{G}} = \mathbf{Q}^T\mathbf{\Lambda}\mathbf{Q}$. It should be noted that $\mathbf{Q}$ consists of the first $l$ eigenvectors with the largest corresponding eigenvalues and $\mathbf{\Lambda}$ is a diagonal matrix with $l$ largest eigenvalues. This simplifies the linear solve operation in Equation (4) to finding the inverse matrix to $\hat{\mathbf{G}} + \lambda\sigma^2 d(\mathbf{P1})^{-1}$ using the Woodbury identity [7] as expanded in Equation (6) and multiplying the right-hand term of Equation (4). The transformed set is also obtained using the approximated matrix $\mathbf{T} = \mathbf{Y} + \hat{\mathbf{G}}\mathbf{W}$

$$(\hat{\mathbf{G}} + \lambda\sigma^2 d(\mathbf{P1})^{-1})^{-1} = \frac{1}{\lambda\sigma^2} d(\mathbf{P1}) - \frac{1}{(\lambda\sigma^2)^2} d(\mathbf{P1})\mathbf{Q}(\mathbf{\Lambda}^{-1} + \frac{1}{(\lambda\sigma^2)}\mathbf{Q}^T d(\mathbf{P1})\mathbf{Q})^{-1}\mathbf{Q}^T d(\mathbf{P1}). \tag{6}$$

Using this expression considerably speeds up the linear solve and, consequently, the iteration as the right-hand side only consists of scaling, addition and multiplication by diagonal matrices in high dimension; matrix inversion is only done in low dimension.

The preprocessing step (partial decomposition of $\mathbf{G}$ into eigenvectors) now becomes the most time-consuming part and we will refer to the length of this operation as *init time*. On the other hand, *total time* will be the time required for CPD to initialize and converge. Myronenko reported that about 60% of total time is init time despite a fast method like Arnoldi's iteration is used. Our results support this finding and we will thus focus our acceleration endeavors to this area.

### 2.3. Low-rank approximations of matrices

Numerous ways of approximating matrices in low rank have been proposed both for general matrices and for matrices with some specific property. In this subsection we review some of those methods, however as we will want to approximate $\mathbf{G}$ from Equation (2) that is a real symmetric matrix, we will limit ourselves to approximations of those.

Possibly the most popular way is by using the eigenvector decomposition (or singular value decomposition in general) $\hat{\mathbf{G}} = \mathbf{Q}^T\mathbf{\Lambda}\mathbf{Q}$ where $\mathbf{\Lambda}$ is a square matrix with the $k$ largest eigenvalues on its diagonal and zeros as the remaining elements and $Q$ is the matrix containing the corresponding eigenvectors. This approximation is also optimal in terms of Frobenius norm, specifically $\|\hat{\mathbf{G}} - \mathbf{G}\|_F$ will be minimal; this is, of course, dependent on finding the precise eigenvectors. Proof of this interesting property can be found in many linear algebra textbooks, for instance [5].

The current algorithms that approximate eigenvectors of matrices differ in numerical stability, computational cost and expected error bounds. Among the most successful algorithms are those that operate by constructing Krylov subspaces of the matrix. These include among others the Arnoldi iteration which is applicable to general