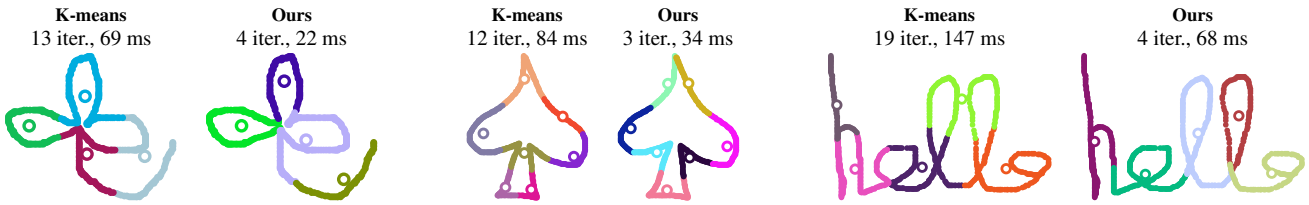
Simple, Fast, and Accurate Clustering of Data Sequences*

Luis A. Leiva and Enrique Vidal
Institut Tecnològic d'Informàtica, Universitat Politècnica de València
{luileito,evidal}@iti.upv.es



(a) Trifolium: 467 points, 4 classes **(b) Pica**: 484 points, 7 classes **(c) Hello**: 836 points, 5 classes

Figure 1: *Continuous* handwriting HTML5 canvas-based experiments performed on an i686 dual core CPU @ 1GHz running Firefox 3.6. (1a) Identifying drawing parts. (1b) Discovering symmetries. (1c) Isolating characters. As observed, our proposal provides better solutions in terms of performance (less computation time) and accuracy (well-formed segmentations).

ABSTRACT

Many devices generate large amounts of data that follow some sort of sequentiality, e.g., motion sensors, e-pens, or eye trackers, and therefore these data often need to be compressed for classification, storage, and/or retrieval purposes. This paper introduces a simple, accurate, and extremely fast technique inspired by the well-known K-means algorithm to properly cluster sequential data. We illustrate the feasibility of our algorithm on a web-based prototype that works with trajectories derived from mouse and touch input. As can be observed, our proposal outperforms the classical K-means algorithm in terms of accuracy (better, well-formed segmentations) and performance (less computation time).

Author Keywords

Sequential data, clustering, trajectory segmentation

ACM Classification Keywords

H.5.2 User Interfaces: Input devices and strategies; I.5.3 Clustering: Algorithms

INTRODUCTION

There is a notably increasing spectrum of devices that capture dense volumes of sequentially-generated information (e.g., active RFID tags, e-pens, eye trackers, etc.). Therefore, a clear interest exists in new compression techniques and methods to simplify the structure of this kind of data, so that original objects can be represented by more compact structures that are better tailored for classification, storage, and retrieval purposes. This can be approached as a clustering problem, i.e., organizing such an object sequences into subgroups whose members are similar in some way.

When clustering sequential data there exists a strong constraint, often related to time, that can be exploited to a great advantage. Nonetheless, by ignoring this constraint, classical clustering techniques fail to cope with the underlying sequential data structure, as illustrated in Figure 1. What is more, previous research on sequential data clustering considered the objects to cluster as whole data sequences or previously determined subsequences thereof, c.f. [3, 4]. Instead, we are interested in discovering *subtrajectories* within a single trajectory, so that we can obtain a simplified data structure preserving the underlying data sequentiality. From this point of view, approaches based on Hidden Markov Models (HMMs) have been proposed; e.g., [1]. The downside of HMMs, however, is that they require a complex training, and also can be prohibitive if processing power is a restriction, e.g., working on mobile devices. As such, we propose here a closed-form solution having a *low computational cost* in terms of performance, which translates to really fast convergence times, and providing *consistent results* in terms of accuracy: each run for a given number of classes always yields the same (well-formed) sequential clustering configuration.

SYSTEM OVERVIEW

In our prototype, the user can draw on a browser canvas with any pen-based device, including a traditional computer mouse or a touchscreen. The desired number of clusters can be either specified by the user or estimated by the system. The generalized workflow is shown in Algorithm 1.

*Work supported by MIPRCV (CSD2007-00018) and MITTRAL (TIN2009-14633-C03-01) projects, and grant Prometeo/2009/014.

IUI'12, February 14–17, 2012, Lisbon, Portugal.
ACM 978-1-4503-1048-2/12/02.

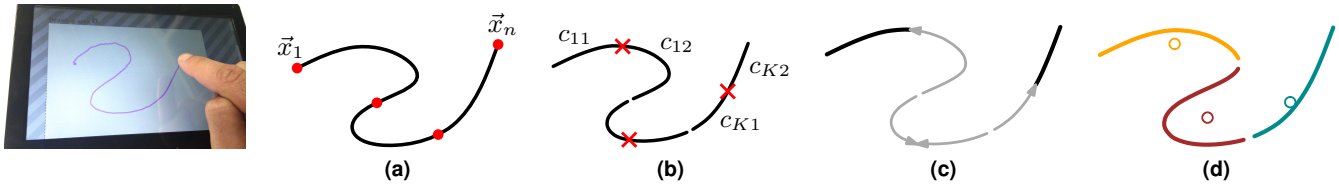


Figure 2: Graphical overview of Algorithm 1 for $K = 3$ clusters. (2a) Key points identification: the first and last key points always match the first and last data points ($\vec{x}_1$ and $\vec{x}_n$); (2b) Initial segmentation: crosses mark the K segments' middle points; (2c) Point visiting order for reallocation: notice that chunks c_{11} and c_{Kn} do not need to be inspected; (2d) Final clustering configuration: circles represent each segment's centroid.

Method Rationale

Given a data sequence (or *trajectory*) X of n points and the number of clusters (or *segments*) K, the system provides a *sequential* clustering configuration that minimizes the sum of quadratic errors (or *distortion*) criterion. To this end, the increment in the overall distortion that would be caused by reallocating a point $\vec{x}$ from cluster j to cluster k is incrementally computed as [2]:

$$\Delta J(\vec{x}, j, k) = \frac{n_k}{n_k + 1} \parallel \vec{x} - \vec{\mu}_k \parallel^2 - \frac{n_j}{n_j - 1} \parallel \vec{x} - \vec{\mu}_j \parallel^2 \quad (1)$$

where $\vec{\mu}_j$ and $\vec{\mu}_k$ are the means of the two clusters involved. Point $\vec{x}$ is reallocated when this increment is negative. This way of reallocating data points is inspired by Duda and Hart's K-means algorithm [2], which allows an incremental computation of the new cluster means, and has been shown to perform better than the popular K-means version [5]. As discussed below, we follow Duda and Hart's reallocation strategy but, in contrast to K-means, we capitalize on the data sequentiality to produce well-defined partitions.

Algorithm pseudo-code

Input: Trajectory $X = \vec{x}_1, \ldots, \vec{x}_n$; No. Clusters $K \geq 2$
Output: Minimum Error Segmentation of X

```
1: Interpolate K + 1 points from X
2: Initialize K segments
3: repeat
4:    for j = 1 to K do
5:       Inspect the first chunk of segment j
6:       Inspect the second chunk of segment j
7: until no reallocations
```

Algorithm 1: The steps of our sequential clustering algorithm.

Method Description

To begin with, a series of key points must be initialized in an ordered way (Figure 2a). This can be achieved by using any interpolation method (e.g., piecewise constant, linear, nonlinear, etc.) over the input data set. Such ordered points are then used to build the initial segments. Next, each segment is divided into two chunks around its middle point (Figure 2b), and both are sequentially inspected. Points belonging to the first chunk in segment j can only be reassigned to the second chunk of segment $j - 1$, and points in the second chunk of segment j can only be reassigned to the first chunk of segment $j + 1$ (Figure 2c). If and only if a point reassignment (Equation 1) contributes to decrease the overall clustering distortion, then the point is finally reallocated. Otherwise, the next chunk is inspected, in order to preserve the clustering sequentiality. In sum, three key features differentiate our approach from other K-means based algorithms: initialization, visiting order, and sequentiality constraints.

ENVISIONED APPLICATIONS

Besides the illustrative web-based system for freehand drawing described before[1], we depict here some usage scenarios where our approach suitably fits.

Online Handwriting. Our clustering technique can be used as a preprocessing step for online text recognition. As illustrated in Figure 1c, the obtained (well-formed) segments capture pen-stroke or even full-character regularities which can be advantageously exploited by existing handwritten recognition approaches to increase character recognition accuracy.

Eye/Mouse Tracking. This algorithm entails a reliable contribution to clustering eye movements on aggregated data; e.g., both heatmaps and areas of interets (AOIs) are computed by distance-based clusters, and therefore they do not distinguish between long-time fixations of a single person or short-time fixations of a group of people.

Motion Segmentation. The storage and transmission of motion tracking content is a problem due to their tremendous size and the noise caused by imperfections in the capture process. Thus, one could use our method for a more compact representation of these (large) data.

In general, any discipline that would handle ordered data sequences could benefit from our approach; e.g., human motion classification from surveillance cameras or automatic video key frame extraction.

REFERENCES

1. Bashir, F. I., Khokhar, A. A., and Schonfeld, D. Object trajectory-based activity classification and recognition using hidden markov models. *IEEE Trans. on Image Processing* (2007), 1912–1919.
2. Duda, R. O., Hart, P. E., and Stork, D. G. *Pattern Classification*, 2nd ed. John Wiley & Sons, 2001.
3. Guralnik, V., and Karypis, G. A scalable algorithm for clustering sequential data. In *Proc. ICDM* (2001), 179–186.
4. Lee, J.-G., Han, J., and Whang, K.-Y. Trajectory clustering: a partition-and-group framework. In *Proc. SIGMOD* (2007).
5. Leiva, L. A., and Vidal, E. Warped K-Means: An algorithm to cluster sequentially-distributed data. *Pattern Recognition Letters* (2012). To be published.

[1] Available at `http://personales.upv.es/luileito/wkm/`.