Rek-means
A k-means Based Clustering Algorithm

Domenico Daniele Bloisi and Luca Iocchi

Sapienza University of Rome
Dipartimento di Informatica e Sistemistica
via Ariosto 25, 00185 Rome, Italy
{bloisi,iocchi}@dis.uniroma1.it
http://www.dis.uniroma1.it/~bloisi

Abstract. In this paper we present a new clustering method based on k-means that has been implemented on a video surveillance system. Rek-means does not require to specify in advance the number of clusters to search for and is more precise than k-means in clustering data coming from multiple Gaussian distributions with different co-variances, while maintaining real-time performance. Experiments on real and synthetic datasets are presented to measure the effectiveness and the performance of the proposed method.

Key words: data clustering, k-means, image segmentation

1 Introduction

Data clustering techniques are used to accomplish a multitude of tasks in many fields, such as artificial intelligence, data mining, computer vision, data compression, and others. There are a series of well known algorithms that perform the task of clustering data: k-means, Fuzzy C-means, Hierarchical clustering, EM, and many others [1]. It can be shown that there is no absolute best criterion which would be independent of the final aim of the clustering [2]. Consequently, the user must supply this criterion, in such a way that the result of the clustering will suit his/her needs. For instance, we could be interested in finding representatives for homogeneous groups (data reduction), in finding natural clusters and describe their unknown properties (natural data types), in finding useful and suitable groupings (useful data classes) or in finding unusual data objects (outlier detection). While the above discussed characteristic is common to all the clustering methods, it is possible to compare techniques with respect to time efficiency and solution quality. Among these methods, k-means is the most efficient in terms of execution time [1]. However, k-means requires to specify in advance the number of clusters k, it converges only to a locally optimal solution, and the solution depends from the initialization step.

The aim of this paper is to present an extension of k-means, called Rek-means which has the following features:

1. it provides better results in clustering data coming from different Gaussian distributions;
2. it does not require to specify k beforehand;
3. it maintains real-time performance.

In the rest of the paper, we will first present a brief overview of the ARGOS project, then we will describe the proposed algorithm, and finally we will illustrate implementation details and experimental results.

2 ARGOS project

The ARGOS project (Automatic Remote Grand Canal Observation System) [3] has been launched in early 2006 by the Municipal Administration of Venice and it is in full operation since September 2007.

The system provides for boat traffic monitoring, measurement and management along the Grand Canal of Venice based on automated vision techniques. The Venice Grand Canal (about 4 km length and 80 to 150 meters width) is monitored through 14 observation posts (Survey Cells), each composed by three perspective cameras and a PTZ camera. Boats are identified by background subtraction techniques and tracked with a set of multi-hypotheses Kalman Filters. All the information from the 14 survey cells are communicated to a central server that shows an integrated view of all the Grand Canal and of the traffic within it. The system allows for traffic statistics, event detection (e.g., speed limit and wrong ways) and is in use by Venice Administration in order to evaluate and monitor boat navigation rules.

While a more detailed presentation of the system is given in [3], in this paper we focus on data clustering step that is performed to refine segmentation of the images. Background subtraction is used to first distinguish foreground (i.e., boats) from background. However, with respect to other surveillance applications (like for example people tracking in indoor environment or car tracking on streets), our system has to deal with a significantly higher noise, mostly due to waves and sun reflections on the water, and with high number of boats in the scene. Therefore, we have implemented an optical flow step after background subtraction, in order to reduce the effects of under-segmentation due to boats being close each other and errors due to waves. Moreover, everything must be done in real-time and due to project constraints we need to process in real-time the image streams coming from three high-resolution cameras (1280x960) on a single PC with an Intel Core 2 Duo 2.4 GHz CPU.

The first approach to cluster data was based on k-means and gave us poor results, while more sophisticated approaches, like EM algorithm, were not compatible with real-time constraints we have in the project. The algorithm proposed in this paper has shown very good performance in clustering while maintaining real-time performance.

3 K-means limitations

K-means [4] is a very simple and fast algorithm for solving the clustering problem. Unfortunately k-means requires that the user knows the exact number of clusters (k) beforehand. When clustering a dataset, the right number k of clusters to use is often not obvious, and choosing k automatically is a hard algorithmic problem [12]. Furthermore k-means is not guaranteed to return a global optimum: The quality of the final solution depends largely on the initial set of centroids, and may, in practice, be much poorer than the global optimum.

Thus, in using k-means, one encounters a major difficulty, i.e., how to choose the k initial clusters. If we do it at random, k-means can in fact converge to the wrong answer (in the sense that a different and optimal solution to the minimization function above exists). As an example, Fig. 1 shows a bad result of k-means (shown by "plus" symbols) with random initialization, even in the case in which the number of cluster ($k = 5$) is correctly specified. In order to

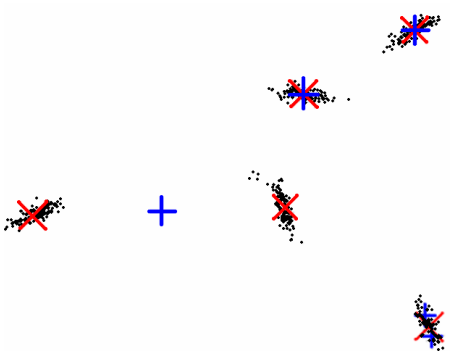

Fig. 1. Centroids found by k-means with random initialization (+) and true ones (x).

overcome the above mentioned limitation, our algorithm is designed to work without an *a priori* knowledge on the number of clusters (k) and to be robust with respect to a random initialization.

4 Rek-means

Rek-means been developed for clustering data coming from a supposed Gaussian distribution and extends k-means algorithm by using small rectangular areas first to over cluster the data, then to efficiently merge them in the final clusters. The prefix "Re" in Rek-means stands thus for rectangle.

4.1 Rek-means algorithm

The Rek-means algorithm (shown in Table 1) performs 5 steps. Suppose we have n 2D-points to cluster (see Fig. 2a). We may choose a distance d that is

the maximal distance between two points in order to consider them belonging to the same cluster. We may choose also a minimal dimension $dimC$ so that a cluster is considered of interest. As a first step, we compute k-means with

Table 1. The Rek-means algorithm.

Rek-means Algorithm
1. (*over-clustering step*) Compute k-means with $k = n/4$ where n is the number of points to cluster.
2. (*cutting step*) Discard every centroid c_i having $dim(c_i) \leq 2$.
3. (*discretizing step*) For each of the remaining centroids c_j, consider a rectangle RECT(c_j) containing all the points belonging to c_j.
4. (*associating step*) If $dist$(RECT(c_i), RECT(c_j)) $\leq d$ then merge clusters c_i and c_j.
5. (*validating step*) When the associating step is terminated, apply a validating test for each found cluster.

$k = n/4$ taking the n points we have to cluster as input (see Fig. 2b). The choice of this number k is a trade-off between speed of execution for this step and the number of clusters that will be associated in the next steps. Our choice ($k = n/4$) gave us good results in our application. In this step, we perform an over clustering of the data, thus we need to reduce the number of clusters.

The second step consists on discarding too small clusters, by eliminating centroids c_i having $dim(c_i) \leq 2$. Since we have initialized $k = n/4$, we expect that each cluster contains in average 4 points: if a cluster has a number of points < 4, then it is likely that such a cluster is far from the true centroid we are searching for. In fact an outlier or a point far from the center is isolated and k-means tends to assign it a centroid c_o with dimension $dim(c_o) = 1$. Furthermore, this discarding rule is useful in presence of noise between two distinct clusters (see Fig. 3a).

In the third step, Rek-means builds a rectangle around each remained centroid c_r (see Fig. 2c). One of the rectangle vertices has coordinates ($minX$, $minY$), while the opposite rectangle vertex has coordinates ($maxX$, $maxY$) where $minX$ and $maxX$ are the minimal and the maximal x values for the points belonging to c_r, $minY$ and $maxY$ are the minimal and the maximal y values for the points belonging to c_r.

In the forth step Rek-means associates clusters with respect to a distance d. In particular, if $dist(RECT(c_i), RECT(c_j)) \leq d$ then clusters c_i and c_j are associated. In Fig. 3b such associating step is depicted in an intuitive manner, while in Fig. 2c a line links the rectangles that are close enough. This step produces larger *macro*-clusters and subsequently a new set of centroids for these *macro*-clusters are computed, by merging the points contained in the associ-

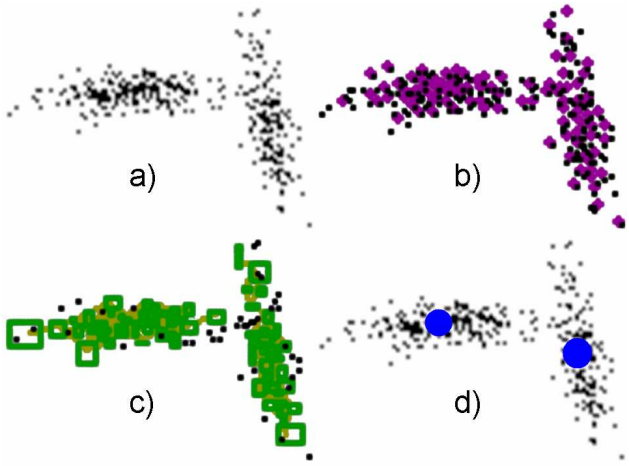

Fig. 2. Rek-means example.

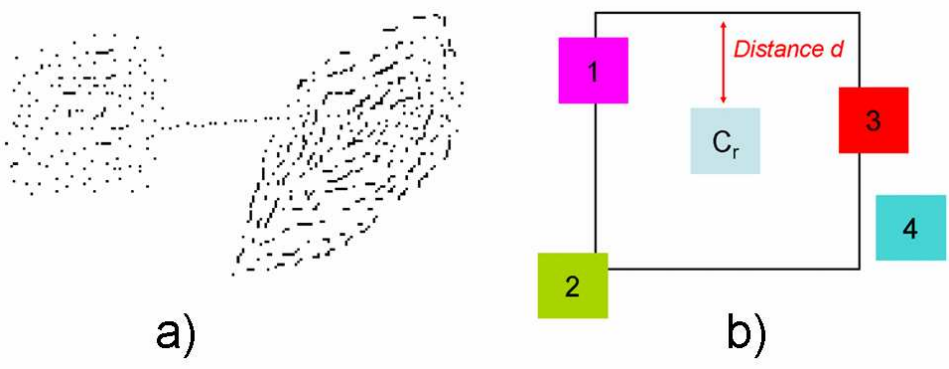

Fig. 3. a) Two clusters not well-separated; b) The associating rule: C_r is associated with 1, 2 and 3 but not with 4. When the algorithm will process 3, it will be associated with 4 and then C_r will be associated with 4 also.

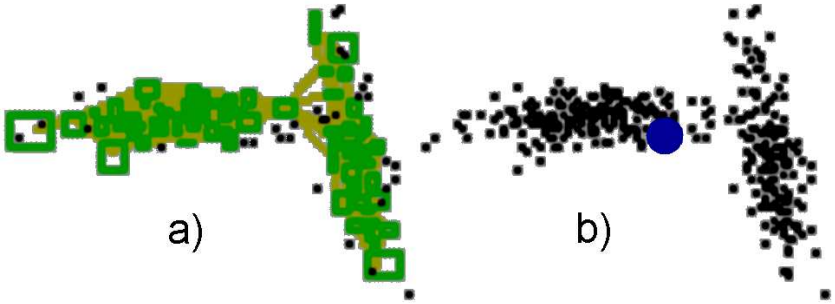

Fig. 4. Under clustering example.

ated rectangles (see Fig. 2d). Rek-means uses rectangles to discretize the cluster dimension since it is computationally efficient to compare the relative distance between rectangle as depicted in Fig. 3b.

The above presented algorithm works on 2D data, but it can be generalized for three or more dimensions: For instance, in 3D it is possible to use parallelepipeds instead of rectangles without modifying the algorithm core.

As one can note, the associating algorithm heavily depends on the value chosen for the distance d: If we choose a large value for d, then we may obtain an incorrect output (see Fig. 4). In order to avoid an under clustering caused by a wrong choice for d, we apply a validating test as the final step of the algorithm.

4.2 The validating step

The validating step is useful if the data to cluster are not well-separated, for example if we have to deal with overlapping clusters. A possible approach consists in applying a statistical test in order to discover if the points belonging to a found cluster follow a particular statistical distribution. There exist a series of one-dimensional statistical test that are effective and simple to apply (for example, the Kolmogorov-Smirnov test [5], the K-L divergence [6], etc.).

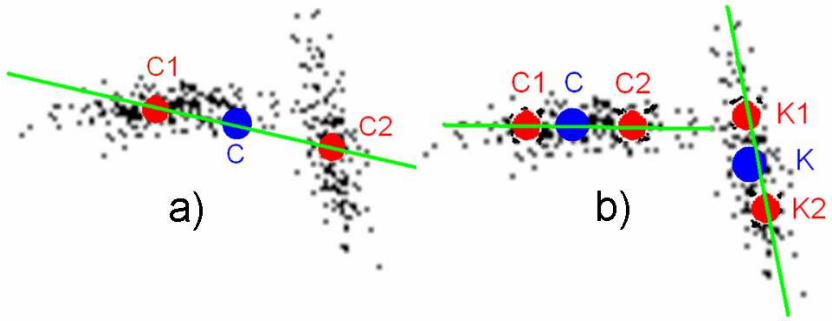

Fig. 5. Projection examples.

Since we are manipulating 2D data, we can either use a two dimensional statistical test (e.g., Peacock test [7], FF test [8], etc.) or reduce 2D-points to one dimension. The currently known 2D tests are not fast (the fastest has $O(nlogn)$ time complexity [9]) and easy to use as the 1D ones, so we have chosen to apply a mono-dimensional test through a projection. Rek-means assumes that the data to cluster follow a Gaussian distribution and uses the Anderson-Darling (AD) statistics [10, 11] to test the data with the confidence level α set to 0.0001. This one-dimensional test has been shown empirically to be the most powerful normality test that is based on the empirical cumulative distribution function (ECDF) [12]. As already mentioned, to apply this test we need to project 2D points over a line. The projection details are depicted in Fig. 5.

Table 2. The validating procedure.

The Validating Procedure
Let S be the set of centroids found by the associating algorithm. For every $c_i \in S$: 1. Find c_i^1 and c_i^2 (applying k-means with $k = 2$), project the points belonging to c_i onto the line connecting c_i^1 and c_i^2, translate, normalize and do the AD test. 2. If the test is successful then c_i is a true centroid and discard it from S, otherwise add c_i^1 and c_i^2 to S. Repeat until S is empty.

The simple idea [12] consists in taking the points belonging to a centroid C and then applying k-means with $k = 2$ with such points as input in order to find two new centroids *C1* and *C2* (see Fig. 5). If C is a correct centroid, the projected points over the line connecting *C1* and *C2* must have a Gaussian distribution in 1D and thus the projected points pass the statistical test (see Fig. 5b). If C is not a correct centroid, the points projected over the line connecting *C1* and *C2* are not sampled from a Gaussian distribution and so the statistical test rejects them (see Fig. 5a). In case of rejection, the Rek-means algorithm performs k-means over the subsets of data that do not pass the validating test with incrementing k , and then re-apply the AD test, until the test is satisfied. Fig. 5 can be viewed as a graphical explanation of two iterations made by the validating step: Fig. 5a depicts the first iteration with a negative test, while Fig. 5b depicts the second iteration with two positive tests. As a difference with [12], Rek-means applies the projection over a selected set of points (the one selected by the association step), minimizing in this way the possible errors due to the k-means initialization. Table 2 summarizes the operations made in the validating procedure. Note that it terminates always since the AD test for a single point is trivially positive.

4.3 Rek-means time complexity

The Rek-means time complexity can be analyzed considering the complexity of each single step:

1. over clustering step
2. cutting step
3. discretizing step
4. associating step
5. validating step

The first step has a complexity equal to the k-means complexity, i.e., $O(kn)$ (a particular exception for the k-means time complexity has been showed in [14]). The second and the third steps have $O(1)$ complexity. The forth step has $O(n^2)$ complexity if the rectangles to be compared are not sorted, but if we sort them

(for example, simply with respect to the x-coordinate), we can reduce such a complexity to $O(nlogn)$, i.e., the time needed to sort the data $O(nlogn)$ plus the time of a binary search $O(logn)$. The fifth step has $O(nlogn)$ time complexity, since we have to apply two geometrical transformations to project onto the x-axis and to compute the AD test with sorted samples. Summarizing, the overall complexity of the algorithm is $O(nlogn)$.

5 Implementation and Experiments

Rek-means has been designed for solving the problem of segmenting real images in real-time. The segmentation is made exploiting optical flow computation that yields a sparse map (see Fig. 6). Unfortunately, we cannot know in advance how many boats are in the image in a given moment and even if we suppose to know such number, k-means can produce poor results on data coming from the characteristic elliptic shape of boats (Fig. 6).

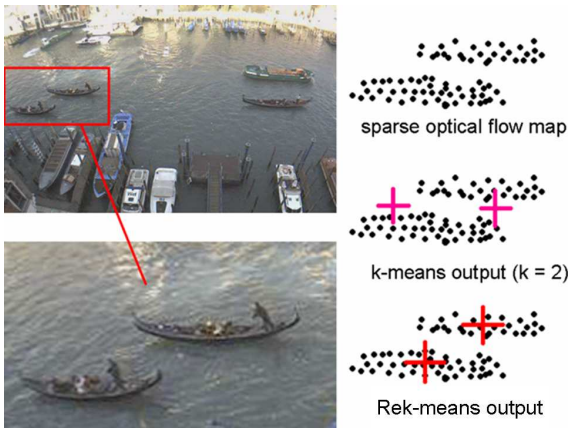

Fig. 6. Rek-means vs k-means in a real scenario.

The algorithm presented in this paper runs within the core of the ARGOS video-surveillance system in a very effective and efficient way. The overall system processes three high resolution (1280×960) image streams on a single PC with an Intel Core 2 Duo 2.4 GHz CPU, with an average frame rate of about 7 fps.

ARGOS is regularly evaluated by user inspection and several tests show the effectiveness of the system in image segmenting and tracking of all the boats in the channel. The main achievement, that is related to the Rek-means algorithm presented here, is the ability of the system of correctly distinguishing boats that are very close (see Fig. 7).

We also tested the algorithm on a synthetic dataset obtained by a Gaussian random number generator (developed by Elkan and Hamerly [12]). It is made of

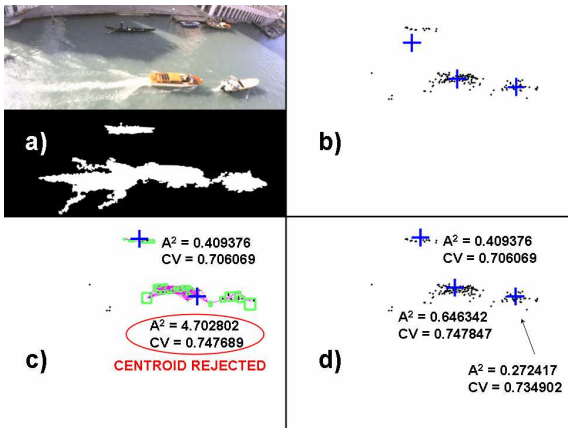

Fig. 7. a) a real frame and the corresponding foreground; b) k-means output with k = 3. c) Rek-means output before the validating step (A^2 is the AD value while CV is the critical value); d) Rek-means output after the validating step.

100 distinct images (with resolution 2000×2000), each containing data from 5 different Gaussian distributions with an eccentricity = 4, in order to simulate the typical shape of a boat. For every image Rek-means algorithm finds the correct number and position of the centroids. The experiment was repeated using k-means with k = 5 (using the OpenCV implementation [13]). The results of the comparison are showed in Table 3.

Table 3. A comparison between Rek-means and k-means.

Algorithm	**Percentage of Correct Output**	**CPU Time (ms)**
Rek-means	100	29
k-means	59	< 1

6 Conclusion

In this paper we have presented Rek-means, a new clustering method based on k-means that has been implemented on a video surveillance system. Rek-means aims at overcoming the main limitations of k-means, which consist in the dependence of the result to the initial configuration and in its need to guess the correct number of centroids (k) as input.

As shown by experimental results, Rek-means returns better results than k-means in clustering data coming from n-dimensional multiple Gaussian distributions with different co-variances, while maintaining real-time performance.

Acknowledgments. The project has been realized thanks to the view of the future and to the active participation of the City Council of Venice. In particular, special thanks to Lord Vice-Major of Venice, On. Michele Vianello, for his foresight in applying innovative technologies in the delicate and complex historical city of Venice. We are also grateful to the Responsible Manager Arch. Manuele Medoro and his staff for their constant support and commitment. We finally thank Ing. Luigi Tombolini and Ing. Luca Novelli for their valuable help on this research work.

References

1. Jain, A.K. and Murty, M.N. and Flynn, P.J.: Data Clustering: A Review. In: ACM Computing Surveys 31(3). (1999) 264–323.
2. Orlowska,M.E. and Sun, X. and Li, X.: Can exclusive clustering on streaming data be achieved? In: SIGKDD Explorations 8(2). (2006) 102–108.
3. Bloisi, D., Iocchi, L., Leone, G. R., Pigliacampo, R., Tombolini, L., and Novelli, L.: A Distributed Vision System for Boat Traffic Monitoring in the Venice Grand Canal. In: VISAPP07 (2007) 549–556.
4. MacQueen, J.B.: Some Methods for classification and Analysis of Multivariate Observations. In: Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability, Berkeley, University of California Press, 1. (1967) 281–297.
5. Chakravarti, Laha, and Roy: Handbook of Methods of Applied Statistics, Volume I, John Wiley and Sons. (1967) 392–394.
6. Kullback, S., and Leibler, R. A.: On information and sufficiency. In: Royal Astronomical Society, Monthly Notices, vol. 202. (1983) 79–86.
7. Peacock, J. A.: Two-dimensional goodness-of-fit testing in astronomy. In: Annals of Mathematical Statistics 22. (1951) 615–627.
8. Fasano, G. and Franceschini, A.: A Multidimensional Version Of The Kolmogorov-Smirnov Test. In: Monthly Notices of the Royal Astronomical Society, vol 225. (1987) 155–170.
9. Lopes, R. H., Reid, I., Hobson, P. R.: The two-dimensional Kolmogorov-Smirnov test. In: XI International Workshop on Advanced Computing and Analysis Techniques in Physics Research. (2007).
10. Anderson, T. W. and Darling, D. A.: Asymptotic theory of certain "goodness-of-fit" criteria based on stochastic processes. In: Annals of Mathematical Statistics 23. (1952) 193–212.
11. Stephens, M. A.: EDF Statistics for Goodness of Fit and Some Comparisons. In: Journal of the American Statistical Association 69 (1974) 730–737.
12. Hamerly, G. and Elkan, C.: Learning the k in k-means. In: proceedings of the seventeenth annual conference on neural information processing systems (NIPS). (2003) 281–288.
13. Open Computer Vision Library (OpenCV): http://opencvlibrary.sourceforge.net/
14. Arthur, D. and Vassilvitskii, S.: How slow is the k-means method? In: Symposium on Computational Geometry. (2006) 144–153.