First publ. in: Proceedings / Third IEEE International Conference on Data Mining, ICDM 2003 : 19 - 22 November 2003, Melbourne, Florida, pp. 565-568

PixelMaps: A New Visual Data Mining Approach for Analyzing Large Spatial Data Sets

Daniel A. Keim, Christian Panse, Mike Sips
University of Konstanz, Germany
{keim, panse, sips}@informatik.uni-konstanz.de

Stephen C. North
AT&T Shannon Laboratory, Florham Park, NJ, USA
north@research.att.com

Abstract

PixelMaps *are a new pixel-oriented visual data mining technique for large spatial datasets. They combine kernel-density-based clustering with pixel-oriented displays to emphasize clusters while avoiding overlap in locally dense point sets on maps. Because a full evaluation of density functions is prohibitively expensive, we also propose an efficient approximation,* **Fast-PixelMap**, *based on a synthesis of the quadtree and gridfile data structures.*

1 Introduction

Progress in technology now allows computer systems to store and exchange datasets that were, until recently, considered extraordinarily vast. Almost all transactions of everyday life (purchases made with credit cards, web pages visited, and telephone calls made) are recorded by computers. This data is collected because of its potential to provide a competitive advantage to its holders. Finding valuable details that reveal fine structures hidden in the data, however, is difficult.

In many application domains, data is collected and referenced by its geo-spatial location. Consider, for example, a credit card purchase transaction record that describes products, quantities, time, and addresses of both the customer and merchant. There are many ways of approaching analysis of this data, including creating statistical models, clustering, and finding association rules, but often it is just as important to find relationships involving geographic location.

Automated data mining algorithms are indispensable for analyzing large geo-spatial data sets, but often fall short of completely satisfactory results. Although automatic approaches have been developed for mining geo-spatial data [3], they are often no better than simple visualizations of the data on a map. Interactive data mining based on a synthesis of automatic and visual data mining may not only yield better results, but offer a higher degree of user satisfaction and confidence in the findings [3]. Presenting data in an interactive, graphical form often fosters new insights, encouraging the formation and validation of new hypotheses to the end of better problem-solving and gaining deeper domain knowledge. Analysis may involve multiple parameters, shown on multiple maps. If all maps in such a collection show the data in the same way, it may be easier to relate the parameters and to detect local correlations, dependencies, and other interesting patterns. On the other hand, when large data sets are drawn on maps, the problem of identifying local patterns is greatly confounded by undesired overlap of data points in densely populated areas, while lightly populated areas are almost empty.

Previous Approaches There are several approaches to coping with dense geographic data already in common use. One popular method is a 2.5D visualization showing data points aggregated up to map regions. This technique is commercially available in systems such as VisualInsight's In3D [1] and ESRI's ArcView [2]. Another approach, showing more detail, is the visualization of individual data points as bars on a map. This technique is embodied in systems such as SGI's MineSet [5] and AT&T's Swift 3D [6]. An alternative that does not aggregate data, and still avoids overlap in the two-dimensional display, is the Gridfit method [7]. The idea of Gridfit is to automatically reposition pixels that would overlap, an idea we also adopt in this contribution.

Our Approach In this paper we describe PixelMaps, a new approach to the display of dense point sets on maps, which combines clustering and visualization. PixelMaps are novel in several ways: First, they provide a new tool for exploratory data analysis with large point sets on maps, and thus augment the flexibility, creativity, and domain knowledge of human data analysts. Second, they combine advanced clustering algorithms with pixel-oriented visualization, and thus exploit the computational and graphics capabilities of current computer systems.

2 Problem Definition

The problem of visualizing geo-spatial data can be described as a mapping of input data points, with their original positions and associated statistical data values, to unique positions on an output map. Let A be the set of original data points $A = \{a_0, \ldots, a_{N-1}\}$, where $a_i = (a_i^x, a_i^y)$ is the original position of a data point, and $S_1(a_i), \ldots, S_k(a_i)$ are statistical parameters associated with a point. Since A is assumed to be large, it is likely that many data points i and j have the same original positions, *i.e.* $a_i = a_j$. Let the data display space DS be defined as $DS = \{0, \ldots, x_{max}-1\} \times \{0, \ldots, y_{max}-1\}$, where x_{max} and y_{max} are the extents of the display region. Our goal is to determine a mapping function f from the original data set to a solution set $B = \{b_0, \ldots, b_{N-1}\},\ 0 \leq b_i^x \leq x_{max}-1,\ 0 \leq b_i^y \leq y_{max}-1$ such that $f : A \rightarrow B, \quad f(a_i) = b_i \quad \forall i = \{0, \ldots, N-1\}$, i.e. f determines the new position b_i of a_i. The mapping function must satisfy three constraints:

1. **No overlap Constraint**
   The first and most important constraint is that all data points must be visible, which means that each one must be assigned to a unique position. Formally, this means $i \neq j \quad \Rightarrow \quad b_i \neq b_j \quad \forall i, j \in \{1, \ldots, N-1\}$

2. **Position Preservation Constraint**
   The second constraint is that the new positions should be as close as possible to the original ones. We measure this objective by summing the absolute distances of the data points from their original positions $\sum_{i=0}^{N-1} d(a_i, b_i) \longrightarrow min$ or the relative distances between the data points $\sum_{i=0}^{N-1} \sum_{j=0, i\neq j}^{N-1} (d(b_i, b_j) - d(a_i, a_j))^2 \longrightarrow min.$ The distance function d can be defined by a L^m-norm ($m = 1$ or $m = 2$). This constraint ensures that the display closely represents the original data. The specific data analysis task at hand probably determines whether an absolute or relative metric is more suitable.

3. **Clustering Constraint**
   The third constraint involves clustering on one of the statistical attributes $S_i, i \in \{0, \ldots, k\}$. The idea is to present the data points such that those with high similarity in S_i are close to each other[1]. In other words, points in a neighborhood of any given data point should have similar values, so the output has pixel coherence. This can be expressed as: $\sum_{i=0}^{N-1} \sum_{b_j \in \mathcal{NH}(b_i)} d_S(S(b_i), S(b_j)) \longrightarrow min.$ Note that this depends on the definition of the neighborhood $\mathcal{NH}$ of data points a_i, and the distance function d_S on the statistical attribute S.

[1]We assume that the clustering depends on the statistical attribute $S \in \{S_0, \ldots, S_k\}$.

Trade-Offs and Complexity While it is not too hard to find a good solution for any of these three constraints taken individually, they are difficult to optimize simultaneously. Since we give priority to constraint 1 (no overlap), the other two constraints often conflict. If constraint 2 is optimized, the location information is retained as much as possible but there may be little pixel coherence in the display. If constraint 3 is satisfied, the data is clustered according to S but the location information may be destroyed. Therefore, our goal is to find a good trade-off between constraints 2 and 3. This is a complex optimization problem that is likely to be NP-hard.

3 The PixelMap Algorithm

In this section, we describe an algorithm for making PixelMaps by optimizing the objectives described previously. The PixelMap algorithm solves the optimization problem by kernel density estimation and an iterative local repositioning scheme. It starts by computing a kernel-density-estimation-based clustering in the three dimensions $(a_i^x, a_i^y, S(a_i))$. Kernel density is a way of estimating the density of a statistical value $(S(a_i))$ at all locations in a region based on (a_i^x, a_i^y). The clustering defines sets of related pixels determined by the two spatial dimensions and the additional statistical parameter. The idea is to place all data points belonging to the same cluster in proximate display pixels. The next step is a second kernel density estimation based clustering on the two geographical dimensions (a_i^x, a_i^y). The information obtained in the two clustering steps is used for iterative positioning of the data points. Starting with the densest region, all data points belonging to one cluster are placed at neighboring pixels without overwriting previously placed ones. If multiple clusters are in the same area, the smallest cluster is positioned first. After all pixels in an area are positioned, the algorithm applies the same procedure to the clusters of the next densest region, until all the data points are positioned. Outliers and very small clusters, which would otherwise be treated as noise, are at last positioned at the remaining free pixels.

Complexity of the PixelMap Algorithm. Since our goal is to cluster many points locally according to a statistical parameter, we must anticipate a large number $(O(n))$ of relatively small clusters. This requires the kernel density estimation to be computed at a fine grain, with many peaks that must be discovered (such as by hill-climbing). In addition, the smoothness (σ) of the kernel function needs to vary with spatial density, and different kernel functions are needed for the spatial and statistical dimensions. These problems make it computationally prohibitive to directly implement the PixelMap algorithm for large data sets.

4 Fast-PixelMap - An Efficient Solution of the PixelMap Problem

The basic idea of Fast-PixelMap is to rescale certain map regions to better fit dense point clouds to unique positions on the output map. The *Fast-PixelMap*-algorithm is an efficient heuristic approximation to the *PixelMap*-algorithm, combining some of the advantages of grid-files and quadtrees in a new data structure to approximate the kernel density functions and enable placement of data points at unique output map positions. This data structure supports, first, the recursive partitioning of both the geo-spatial data set and the Euclidean 2D display space to enable an efficient distortion of the map regions, second, an automatic smoothing depending on the x-y density, and third, an array-based 3D density estimation.

The above mentioned recursive partitioning can be efficiently stored as a binary tree in each case, and the combination of both binary trees within a single multidimensional array. This combination is realized through the storage of the coordinates of the two different arising split points (in the data and in the display space) in each top-down construction step. Note, that our data structure uses midden split-operations according to different parameters. In case of the geo-spatial data set, a gridfile-like midden-split, and in case of the display space, a quadtree-like midden split operation is performed. The gridfile-like partitioning of geo-spatial data sets applies split operations within the 10% surrounding neighborhood of the middle point (left+right)/2 of the arising geo-spatial partition. The recursion terminates if the maximal split level is reached, or if a partition contains fewer than four data points. The goal is to find dense areas in the spatial dimensions (a_i^x, a_i^y) and to allocate enough pixels to place all the points of these dense regions at unique positions. The Fast-PixelMap data structure enables, in a second step, the efficient distortion of certain map regions in the 2D display space, by relocating all data points within the old boundaries of the quadtree partition to new positions within new boundaries of the quadtree partition. After rescaling all data points to the new boundaries, the iterative positioning of data points starts with the densest region. Within a region, the smallest cluster is chosen first. The iterative pixel position heuristic places all data points belonging to one cluster at adjacent pixels without overwriting existing ones.

Complexity The time complexity of the proposed approach is $O(n\log^2 n)$. The additional space overhead, $0(n + \log n)$, is negligible. This additional space is needed by the Fast-PixelMap data structure to store the original data points with a constant number of split-operations (which depends on the maximal split-level).

5 Application and Evaluation

We experimentally compared the Fast-PixelMap algorithm with a genetic algorithm for multi-objective optimization [8], and with PixelMap (based on the DenClue clustering algorithm [4]). We evaluated them with respect to time efficiency and the objectives presented in section 2. The experiments were run using a sample of 30000 points from the U.S. Year 2000 Census Household Income Database, on a 700 MHz Pentium computer with 1GByte of main memory.

Efficiency and Effectiveness Figure 2 shows time-performance curves of all three methods, with varying degrees of input point overlap. The efficiency results show that the average number of data points assigned to the same position plays an important role in the performance of all three methods. The results indicate that the Fast-PixelMap algorithm outperforms the other two methods for all degrees of overlap, and is computationally practical for large spatial data sets. Effectiveness can be measured with respect to the three optimization goals defined in section 2. Figure 3 shows measured error curves for the three optimization goals. In summary, the results show that Fast-PixelMap is an effective approximation for the pixel placement problem, and is practical for visually exploring large geo-spatial statistical data sets in search of local correlations.

Visual Evaluation and Applications Formal measures of effectiveness are only meaningful if they lead to useful visualizations. Figure 1 shows a sample from the U.S. Year 2000 Census Median Household Income Database for the State New York, which in general validates the mathematically defined effectiveness criteria.

6 Conclusions

We presented the PixelMap algorithm, which combines kernel-density-based- clustering with a novel pixel-based visualization technique. It avoids loss of information due to overplotting of data points. It assigns each input data point to a unique pixel in 2D screen space, and balances the trade-off of spatial locality (absolute and relative position preservation) with clustering to achieve pixel coherence. We also described the Fast-PixelMap heuristic that provides efficient approximate solutions to the PixelMap optimization problem, and is of practical value for exploring geo-spatial statistical data.

Acknowledgments We thank Carmen Sanz Merino and Hartmut Ziegler for their great support. We thank Dave Belanger and Mike Wish for encouraging this investigation.

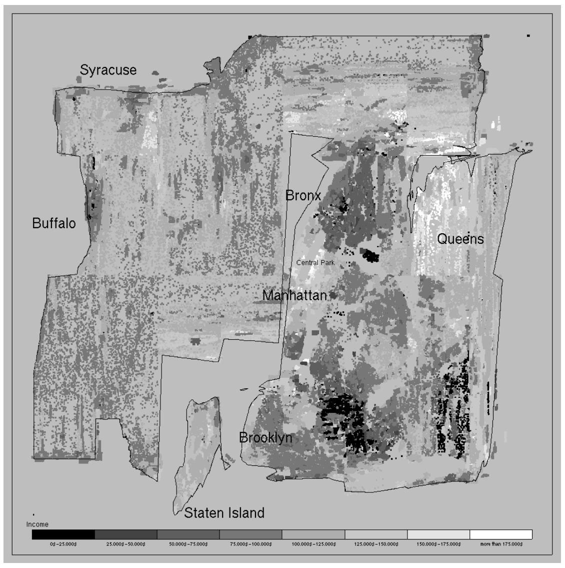

Figure 1. New York State Year 1999 Median Household Personal Income - PixelMap displays cluster regions. Note high-income clusters on the East side of Manhattan's Central Park, and low-income clusters on the West end of Brooklyn.

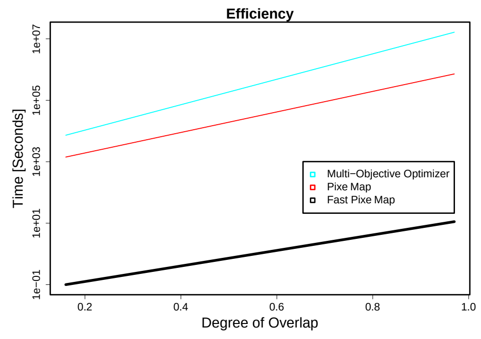

Figure 2. Comparison of the efficiency of Fast-PixelMap, PixelMap, and a multi-objective genetic optimization algorithm (log-scale)

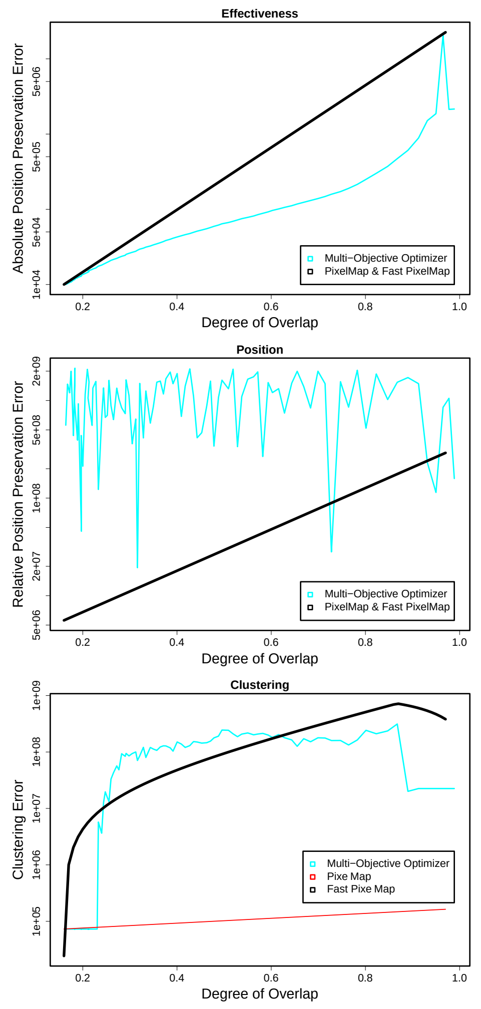

Figure 3. Effectiveness Measurement of the defined optimization constraints 1, 2, and 3 in section 2 (log-scale)

References

[1] I. Advizor Solutions. Visual insight in3d. `http://www.advizorsolutions.com/`, Aug 26 15:19 2003.

[2] ESRI. An esri white paper: Arcview 3d analyst features, 1998. `http://www.esri.com/library/whitepapers/pdfs/3danalys.pdf`, Aug 26 15:13 2003.

[3] J. Han and M. Kamber. *Data Mining: Concepts and Techniques*. Morgan Kaufmann Publishers, 2001.

[4] A. Hinneburg and D. A. Keim. An efficient approach to clustering in large multimedia databases with noise. In *Knowledge Discovery and Data Mining*, pages 58–65, 1998.

[5] S. M. Homepage. Sgi mineset. `http://www.sgi.com/software/mineset.html`, Aug 26 15:13 2003.

[6] D. Keim, E. Koutsofios, and S. C. North. Visual exploration of large telecommunication data sets. In *Proc. Workshop on User Interfaces In Data Intensive Systems (Invited Talk), Edinburgh, UK*, pages 12–20, 1999.

[7] D. A. Keim and A. Herrmann. The gridfit algorithm: An efficient and effective approach to visualizing large amounts of spatial data. pages 181–188, 1998.

[8] E. Zitzler and L. Thiele. Multiobjective optimization using evolutionary algorithms - a comparative case study. parallel problem solving from nature. In *PPSN-V*, pages 292–301, September 1998.