Dealing with Multiple Requirements in Geometric Arrangements

Erick Gomez-Nieto, *Student Member, IEEE*, Wallace Casaca, Danilo Motta, Ivar Hartmann, Gabriel Taubin, *Fellow, IEEE*, and Luis Gustavo Nonato, *Member, IEEE*

Abstract—Existing algorithms for building layouts from geometric primitives are typically designed to cope with requirements such as orthogonal alignment, overlap removal, optimal area usage, hierarchical organization, among others. However, most techniques are able to tackle just a few of those requirements simultaneously, impairing their use and flexibility. In this work we propose a novel methodology for building layouts from geometric primitives that concurrently addresses a wider range of requirements. Relying on multidimensional projection and mixed integer optimization, our approach arranges geometric objects in the visual space so as to generate well structured layouts that preserve the semantic relation among objects while still making an efficient use of display area. Moreover, scalability is handled through a hierarchical representation scheme combined with navigation tools. A comprehensive set of quantitative comparisons against existing geometry-based layouts and applications on text, image, and video data set visualization prove the effectiveness of our approach.

Index Terms—Overlap removal, similarity preserving, structured layouts, area optimization

1 Introduction

Arranging geometric primitives such as boxes and discs in a two-dimensional layout is a nontrivial task that inherently appears in important visualization applications, as for example, word cloud construction [1], [11], small-multiples arrangements [2], [3], [4], and visual boards [5], [6], [7]. The difficulty in building layouts made up of dozens of geometric objects rests in the set of requirements to be handled simultaneously, e.g., readability, overlaps, object size, semantic proximity and area usage. Moreover, the number of data instances represented as geometric entities is typically much larger than the visualization area, demanding the use of clustering, hierarchies, and navigation resources to assist the visualization.

Although significant advances have been made towards building meaningful layouts from geometric primitives, existing techniques are formulated to deal with a limited number of requirements simultaneously, restricting their use to specific applications. For instance, techniques such as visual boards and small multiples provide well structured layouts which are easily readable, but they pay the price of scalability. Hierarchical methods such as Treemaps [8] mitigate the issue of scalability while making an efficient use of display area. However, readability and semantic organization of data are aspects not so easily handled by those methods. Overlap-free semantic preserving techniques such as RWordles [9] and ProjCloud [10] generate somewhat structured layouts and keep instances with similar content close to each other. Nevertheless, they are not designed to make an efficient use of display area and also suffer from scalability.

Handling many requirements is not straightforward because distinct requirements can compete with each other during layout construction. For instance, to facilitate readability, layouts should be built with as large as possible geometric entities. However, large objects easily fill up the display area, thus limiting the number of instances that can be visualized. Therefore, finding an optimal balance among multiple concurrent requirements is a challenging task, which has not been properly tackled by existing methods.

In this work we present a novel methodology for building layouts from geometric primitives which is able to deal with a wide range of requirements simultaneously. Relying on multidimensional projection, density-based adaptive grids, and mixed integer optimization, our approach is semantically aware, makes an efficient use of display area, and generates well structured grid-like layouts. Moreover, the formulation intrinsically imposes a hierarchy on the data, enabling alternatives for the scalability issue.

The proposed optimization scheme arranges geometric entities (boxes) with varying sizes so as to avoid overlaps while preserving the neighborhood structure of the underlying data (semantics). The area of each geometric primitive is also included in the optimization process to ensure that the display area will be efficiently occupied. In fact, supported by the adaptive grid, our formulation is able to scale elements with different sizes using only one variable, thus rendering the optimization procedure as simple as possible. A comprehensive set of quantitative comparisons against existing geometry-based layouts shows the effectiveness of

- *E. Gomez-Nieto, W. Casaca, D. Motta, and L.G. Nonato are with the Instituto de Ciências Matemáticas e de Computação (ICMC), Universidade de São Paulo, São Carlos, Brazil. E-mail: erick.gomez@usp.br, {wallace, daniloam, gnonato}@icmc.usp.br.*
- *I. Hartmann is with the Universidade Estadual de Rio de Janeiro, Rio de Janeiro, Brazil. E-mail: ivarhartmann@gmail.com.*
- *G. Taubin is with the Brown University, School of Engineering, Providence. E-mail: taubin@brown.edu.*

Manuscript received 10 Dec. 2014; revised 14 July 2015; accepted 28 Sept. 2015. Date of publication 11 Oct. 2015; date of current version 3 Feb. 2016. Recommended for acceptance by A. Tal.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TVCG.2015.2489660

our approach. The usefulness of our methodology in visualizing text, image, and video data sets is also confirmed in several practical applications.

The main contributions of this work can be summarized as follows:

- A novel methodology to generate layouts made up of geometric primitives which deals with multiple requirements such as grid-like structure, semantic relation among objects, display area usage, and overlap removal. The methodology naturally imposes an hierarchy on the visual representation in order to handle a large number of data instances.
- A new optimization scheme that relies on a reduced number of unknowns to generate well structured layouts while making a good use of display area.
- A comprehensive set of comparisons against existing methods and several practical applications to confirm the effectiveness and usefulness of our methodology.

One of the positive aspects in managing multiple requirements during layout construction is that we can play with properties such as object size and neighborhood to highlight relevant portions of the layout without losing the semantic relation among objects. By handling additional requisites such as overlap-free and optimal usage of display area we can reduce visual clutter while improving readability.

As far as we known, no other technique devoted to build layouts from geometric primitives is able to deal with so many concurrent requirements to generate meaningful layouts.

2 Related Work

We focus the following discussion on techniques that rely on the arrangement of simple geometric objects such as boxes, discs, and polygons to visualize data information. We group existing methods in three main categories: structured arrangements, hierarchy-based, and node-displacement techniques. It is worth mentioning that our goal is just to provide an overview of the main classes of methods that make use of simple geometric objects to build visualizations, pointing out their strengths and weaknesses in order to better contextualize our contribution. There are a huge amount of work in the literature and a comprehensive survey about all those methods is beyond the scope of this paper.

Structured arrangements comprise the set of techniques that tile the visual space with geometric primitives so as to produce a well structured layout. IncBoard [11], for instance, uses boxes or hexagons to represent high-dimensional instances which are arranged as a regular grid in the visual space. The arrangement is done such that proximity among objects reflects similarity, that is, neighbor cells in the grid tend to represent similar instances similarly to a chessboard where rows and columns determines cells to represent high dimensional data on 2 D space. Self-Sorting Map [6], [7] is another approach that relies on regular grids to perform visualization. Items are arranged based on a sorting mechanism that keeps similar instances close to each other by using a hierarchical swapping process which aims to maximize normalized cross correlation between the input data and the positions of a structured grid. Small multiples [2], [3] is another class of techniques that rely on structured arrangement of geometric primitives to build visualization layouts. Small multiples is typically used to create different views of a data set [4] or to enable the visual analysis of multiple complex data such as time series [11], [12].

One of the main advantages of structured layouts is their ease interpretation. In fact, van den Elzen and van Wijk [4] showed in a user study that visualizations made up of small multiples are more effective than other methods to visualize different views of the same data. A similar conclusion has been reached by Javed et al. [11] in the context of time series visual analytics. Despite the advantages, structured layouts do not scale easily, which impairs their use in applications involving large data sets.

Hierarchy-Based methods are designed to make an efficient use of display area while enabling dynamic navigation throughout different levels of a hierarchical representation [13]. Treemap [14] and its variants [8], [15], [16], [17] are examples of hierarchy-based techniques tailored to visualize data organized as a tree structure. The visualization is performed by recursively splitting the space in rectangular boxes whose size and orientation reflect the extent of nodes in the hierarchy. Another example is Pedvis [18], a technique that builds upon H-tree layout and rectangular boxes to depict pedigree information structured as deep hierarchies. In contrast to the hierarchical methods described above, Voronoi Treemap [19] uses polygonal objects rather than rectangular boxes towards better representing the importance of nodes in the hierarchy and further differentiate sibling and non-sibling nodes. Despite the efficiency in space occupation, most tree-based layouts are not devised to place similar instances close to each other, making them unsuitable for applications involving similarity-based data exploration. Aiming at addressing this issue, some authors [20], [21] have proposed ordering mechanisms that consistently arranges tree nodes according to some similarity measure. Nocaj and Brandes [22] combined multidimensional scaling, Voronoi Treemap, and a set of visualization resources to highlight the similarity among data instances while enabling navigation throughout the hierarchy.

Hierarchy-based methods scale well and they make good use of display area. However, in contrast to structured arrangements, the layout resulting from those methods are not so easy to read as similar instances are not necessarily placed close to each other.

Node-Displacement techniques arrange geometric objects in the visual space so as to preserve similarity relations while avoiding overlaps. Overlap-free techniques vary greatly as to mathematical formulation and they can be classified as physical-based, heuristic-based, and optimal-based models.

Physical-based models rely on force schemes [23], [24] or spring systems [25], [26] to position geometric entities in the visual space. The lack of guarantees in terms of convergence and the untidiness of the resulting layout are the main weaknesses of physical-based approaches. Heuristic methods typically scan the visual space looking for the best - location for each instance. Recent approaches such as RWordles [9] take into account similarity information to

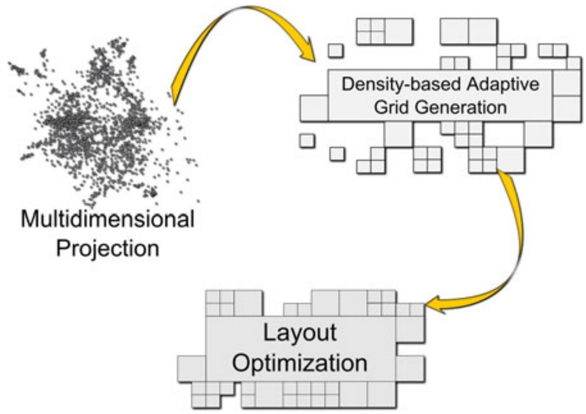


Fig. 1. The proposed pipeline comprises three main steps: (i) data is mapped into visual space using a multidimensional projection technique, (ii) adaptive grid is built from projected points and (iii) an optimization is performed to reach an optimal layout.

build semantically aware layouts. The main advantage of heuristic techniques is their low computational cost that makes possible to build large layouts quickly. Uneconomical use of display area is, however, a major drawback. Optimal approaches arrange objects by minimizing an energy functional subject to a set of constraints. Dwyer et al. [27] and Marriott et al. [28], proposed energy functionals derived from intersection tests, using quadratic optimization to find an optimal solution. In order to improve readability some techniques impose constraints to the energy functional towards generating grid-like arrangements [29], [30]. ProjSnippet [31] is a two-step approach that first projects data onto the visual space and then builds an energy functional that accounts for object overlap as well as neighborhood preservation. The technique called MIOLA [32] also makes use of neighborhood information provided by multidimensional projection to formulate a mixed integer optimization that results in almost orthogonal layouts.

Heuristic and optimal methods perform well in terms of overlap removal and similarity preservation. However, those methods do not make an efficient use of display area, mainly when dealing with geometric objects with different sizes. Scalability is another hurdle for those techniques, since, with a few exceptions [33], node-displacement methods typically do not operate hierarchically.

The method proposed in this work gathers a set of traits not presented in other existing approaches, as it has been designed to account for multiple concurrent requirements such as object scale, neighborhood and aspect ratio preservation among cells, optimal use of display area, and overlap-free grid-like cell arrangement (see Fig. 1(bottom)). Moreover, the proposed technique naturally imposes a hierarchy on the data, thus being able to handle large data sets. Proximity between similar entities is also enforced during layout construction and navigation. In summary, the proposed method bears a set of properties not present together in any other geometry-based layout technique.

3 Proposed Methodology

As illustrated in Fig. 1, the proposed methodology comprises three main steps: *multidimensional projection*, *density-based adaptive grid generation*, and *layout optimization*.

In the first step, high-dimensional data is mapped to the visual space so as to preserve distance among data instances. In our implementation we use the Least Square Projection (LSP) method [34], which preserves distances nicely during the mapping process, thus enforcing that neighbor points in the visual space correspond to similar/close instances in the original space. The use of a distance preserving mapping to place instances in the visual space provides the neighborhood structure that must be "mimicked" by the final layout.

An adaptive grid is then constructed from projected points. In contrast to traditional adaptive grid generation schemes, our approach refines the grid in less dense (but not empty) regions of the visual space, thus placing larger grid cells in denser regions (see Fig. 1(top right)). The rationale is to allow users easily identify dense regions by visually recognizing large geometric objects in the layout. The number of refinement levels is a user defined parameter.

The adaptive grid produces a size varying tiling of the visual space. Moreover, grid cells inherit the semantic relation of the project instances, that is, neighbor grid cells tend to encompass similar instances. Although well structured, the cell arrangement resulting from the refinement is typically spread, making an inefficient use of display area. Therefore, in the third step of the proposed pipeline, cells are rearranged in the visual space to optimize the area usage. The optimization is formulated to account for object scale, overlapping, and grid-like arrangement while preserving neighborhood relationships and the aspect ratio among cells (see Fig. 1(bottom)).

The rationale here is to build layouts as readable as possible while still dragging users attention to denser regions that should be further explored.

The following subsections detail the second and third step of our pipeline, which correspond to the major technical contributions of our approach.

3.1 Adaptive Grid Generation

Let $\mathcal{P} = \{p_1, p_2, \ldots, p_q\} \subset \mathbb{R}^2$ be the projection of a set of instances into the visual space and G be an $n \times m$ regular grid which discretizes the bounding box of $\mathcal{P}$. The number of cells in each orthogonal direction is defined such that each grid cell g_{ij} has a square shape, that is, given the number of subdivision in one direction, for instance m, the number of subdivision in the other direction is $n = \lceil \frac{H}{W} m \rceil$ (the boundary of the bounding box can be displaced to accommodate the number of cells), where H and W account for the height and width of the bounding box of $\mathcal{P}$, respectively.

The regular mesh G is the coarser grid level for the adaptive process. The refinement is driven by density information computed from $\mathcal{P}$ in each grid cell. The density can be estimated in different ways, for instance by integrating a kernel density estimator over each grid cell. As our application does not demand highly accurate density estimation, we opt to approximate the density in each cell by simply counting the number of points p_i contained in the cell.

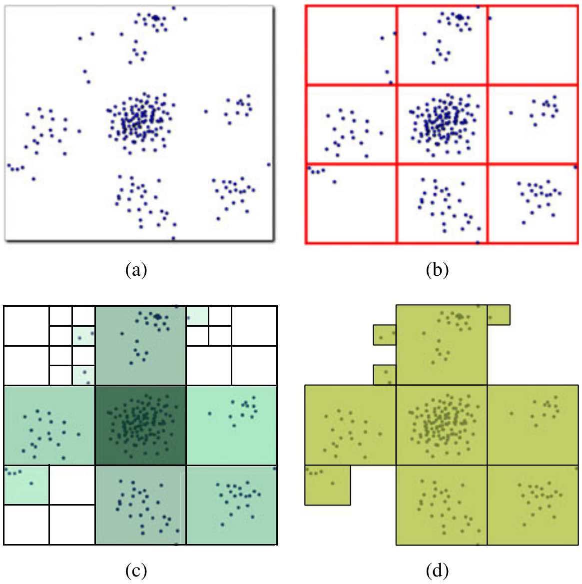


Fig. 2. Density based refinement process. (a) An initial regular grid is defined in the visual space providing the coarser refinement level (first level, as depicted in (b)). (c) An adaptive grid is built from density information. Denser areas are represented by larger cells. (d) Active (not empty) cells are then ready to be optimized in the next step of the proposed framework.

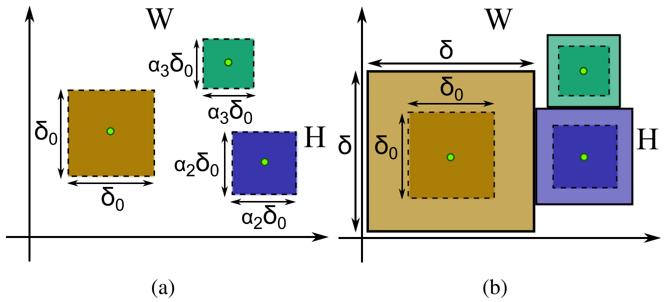


Fig. 3. (a) The input layout with δ_0 being the length of the coarser cell. The size of the remaining cells are defined according to the initial parameter δ_0 and their scale parameters α_i. (b) The result of the optimization when $\delta > \delta_0$.

Given the density information, the refinement is carried out as follows: let d_{ij} be the density of the grid cell g_{ij} and $d_{max} = \max\{d_{ij}\}$ be the largest density value in G. A grid cell g_{ij} undergoes one level of refinement if $d_{ij} \leq 0.5d_{max}$, two levels of refinement if $d_{ij} \leq 0.25d_{max}$, three levels if $d_{ij} \leq 0.125d_{max}$, and so on. Fig. 2 illustrates the refinement process with three levels.

Although simple, the grid generation scheme described above has two helpful properties. First, dense regions in the visual space can easily be identified from the larger cells, pointing out where users should nail down their exploration. Second, the proposed refinement scheme allows for resizing all grid cells by controlling only one parameter. In other words, if one wants to scale all cells preserving their relative size, the only parameter to be tuned is the length δ of coarser cells. In fact, the length of the cells in each refinement level is given by $(1/2^k)\delta$, where $k = 0, 1, \dots$ is the level of refinement. This last property will be exploited during optimization to find the value of δ that results in the best use of display area, as detailed below.

3.2 Optimization

Let $\tilde{G} = \{g_1, g_2, \dots, g_N\}$ be the set of non-empty cells in G, that is, $\tilde{G}$ comprises the cells in G with projected points in their interior. Each cell g_i is a square box described by the vector $g_i = (x_i, y_i, w_i) \in \mathbb{R}^3$, where (x_i, y_i) accounts for the center of the box and $w_i > 0$ is the edge length of g_i. The cells g_i should be rearranged and resized inside a $W \times H$ display area so as to make an efficient use of display area while preserving grid alignment and neighborhood structures. As described in Section 3.1, $w_i = \alpha_i\delta$, where $\alpha_i = 1/2^k$, k corresponding to the level of refinement of g_i. Therefore, δ is a parameter to be optimized (initially set as the length δ_0 of the coarser cell) from which the size of each cell is derived, as illustrated in Fig. 3a.

3.2.1 Rearranging Cells with Area Resizing

The problem of rearranging cells g_i so as to generate layouts that optimize the use of area while presenting a grid-like structure is an NP-hard problem [35] that appears in contexts such as *cutting optimization* and *rectangle packing* [36]. In order to get an approximate solution, we formulate the problem as a computationally tractable quadratic optimization problem. The optimization is formulated as in Equation (1) below:

$$\begin{aligned}
&\text{minimize } E(\mathbf{z}) = E_{comp}(\mathbf{z}) + E_{rezise}(\mathbf{z}),\\
&\text{subject to} \quad A\mathbf{z} \leq \mathbf{b}, \quad \mathbf{z} = [\mathbf{x}\ \ \mathbf{y}\ \ \mathbf{r}\ \ \delta]^\top,\\
&\mathbf{x} = (x_1, x_2, \dots, x_N)^\top \in \mathbf{R}^N\\
&\mathbf{y} = (y_1, y_2, \dots, y_N)^\top \in \mathbf{R}^N\\
&\mathbf{r} = (r_{12}, \dots, r_{1N}, r_{23}, \dots, r_{2N}, \dots, r_{N-1N})^\top, r_{ij} \in \{0, 1\}\\
&\delta_0 \leq \delta \leq \min(W, H),
\end{aligned} \tag{1}$$

where $\mathbf{z}$ is the sought solution; $\mathbf{y}$ and $\mathbf{x}$ correspond to the coordinates of the centroids of the cells; δ is the scaling factor; A and $\mathbf{b}$ hold the constraints imposed on the optimization problem. The unknowns r_{ij} are control variables used to properly avoid overlaps. The energy components $E_{comp}(\mathbf{z})$ and $E_{resize}(\mathbf{z})$ control the proximity between cells and the area increase, respectively. The former term accounts for overlaps and neighborhood preservation and the second term is designed to scale the box to fill up as much area as possible (see illustration in Fig. 4).

Compactness energy term. The first term in the energy functional, E_{comp}, controls the relative position among cells and its goal is to keep the layout compact. The energy E_{comp} is a quadratic function that simply takes into account the centers of the boxes, as shown in Equation (2):

$$E_{comp}(\mathbf{z}) = C \sum_{(i,j)} (x_i - x_j)^2 + (y_i - y_j)^2, \tag{2}$$

where $(i, j) = (j, i)$ represents the $\frac{N(N-1)}{2}$ pairs $(g_i, g_j), i, j, \in \{1, \dots, N\}$ and $C = 1/(\min(W, H) \cdot \frac{N(N-1)}{2})$ is a normalization factor. In less mathematical terms, E_{comp} forces boxes to be close to each other and the normalization factor C ensures that the term E_{comp} contributes in the same amount as E_{resize} in the total energy E, thus enabling a good balance between compactness and area usage.

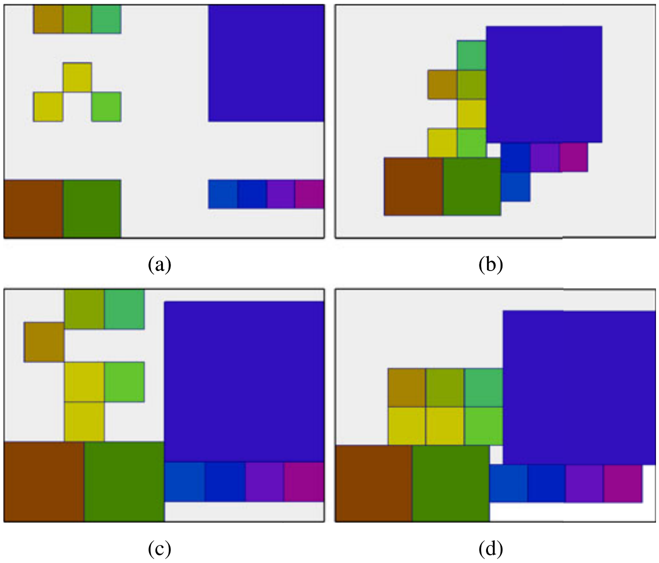


Fig. 4. Minimizing each term of the energy functional E in an illustrative layout. (a) Original layout, (b) E_{comp} only, (c) E_{resize} only, (d) the proposed energy functional $E = E_{comp} + E_{resize}$.

Area usage energy term. The energy term E_{resize} controls the amount boxes should be scaled to optimize the use of display area. Since the area of any cell depends only on the length of the largest cells, that is, the area of each cell g_i is given by $A_i = (\alpha_i \delta)^2$, $\alpha_i = 1/2^k$, where k is the refinement level of g_i, we can define the resize energy as a quadratic function as follows:

$$E_{resize}(\mathbf{z}) = (\delta - \min(W, H))^2, \tag{3}$$

where $\min(W, H)$ is the minimum between the width and height of the display area (see Fig. 4c).

It is easy to see that if no constraints are imposed to the unknowns, the minimum of E_{comp} is reached when all cells have the same center and the minimum of E_{resize} takes place when the length of the larger cell is equal to $\min(W, H)$. Therefore, without constraints the optimization process will stack cells on over the other. To avoid such unsuitable output, we must settle constraints as discussed next.

3.2.2 Optimization Constraints

As described in Equation (1), constraints are gathered in matrix A and vector $\mathbf{b}$ and they are imposed to properly control overlaps and the relative position of cells. Given the initial position (x_i, y_i) and length $\alpha_i \delta$ of each cell, the *relative order* (also called *orthogonal order*) of the boxes is given by:

$$\begin{aligned} x_{p_1} \leq x_{p_2} \leq \cdots \leq x_{p_n} &\Rightarrow x_{p_i} - x_{p_{i+1}} \leq 0 \\ y_{q_1} \leq y_{q_2} \leq \cdots \leq y_{q_n} &\Rightarrow y_{q_i} - y_{q_{i+1}} \leq 0, \end{aligned} \tag{4}$$

where $p, q : \{1, 2, \ldots, N\} \rightarrow \{1, 2, \ldots, N\}$ are permutations of indices generated by sorting the coordinates x_i and y_i.

Inequalities (4) allows for preserving the relative order of the cells, but it does not account for overlap. Overlaps can be handled by forcing *non-overlap conditions* as follows:

$$|x_j - x_i| \geq \frac{(\alpha_i + \alpha_j)}{2}\delta \quad \text{or} \quad |y_j - y_i| \geq \frac{(\alpha_i + \alpha_j)}{2}\delta, \tag{5}$$

From Equation (4) we have that if $x_i \leq x_j$ and $y_i \leq y_j$ then $|x_j - x_i| = x_j - x_i$ and $|y_j - y_i| = y_j - y_i$, which give rise to the following linear system of inequality:

$$\begin{aligned} x_i - x_j &\leq -\frac{(\alpha_i + \alpha_j)}{2}\delta, \\ &\text{or} \\ y_i - y_j &\leq -\frac{(\alpha_i + \alpha_j)}{2}\delta, \end{aligned} \tag{6}$$

As proposed in [32], the *or* condition in (6) can be handled by binary variables $r_{ij} \in \{0, 1\}$ such that:

$$x_i - x_j \leq \alpha_{ij}\delta + M r_{ij} \Leftrightarrow y_i - y_j \leq \alpha_{ij}\delta + M(1 - r_{ij}), \tag{7}$$

where $\alpha_{ij} = -\frac{(\alpha_i + \alpha_j)}{2}$ and M is a very large constant. The role of r_{ij} is to ensure that if one of the inequalities in Equation (6) holds, its counterpart is set aside. For instance, if $x_i - x_j \leq \alpha_{ij}\delta$, then r_{ij} is equal to zero, thus $y_i - y_j \leq \alpha_{ij}\delta + M(1 - r_{ij})$ is naturally satisfied for any values of y. The linear inequalities, r_{ij} unknowns, and δ make up the linear system $[A|\mathbf{b}]$. The left and right bounds for the x coordinate of the center of the cells must also be imposed in $[A|\mathbf{b}]$, that is,

$$0 \leq x_i - \frac{\alpha_i}{2}\delta \quad \text{and} \quad x_i + \frac{\alpha_i}{2}\delta < W, \; i = 1, 2, \ldots, N, \tag{8}$$

Similar inequalities are set to limit the upper and lower bounds of the y coordinates.

Constraints relaxation. The optimization problem described in Equation (1) provides a flexible and simple mechanism to relax constraints and generate distinct layout arrangements. This is performed by adding relaxation parameters (scalars) in the inequalities to allow for some degree of overlap, orthogonal disorder and so on.

3.2.3 Computational Aspects

The formulation (1) is a *Mixed Integer Quadratic Programming (MIQP) Problem*, which is minimized by computing an extension of the so-called *Branch-and-Bound optimization scheme*, where the original MIQP problem is converted to a *Linear Programming Problem* in order to be properly handled and solved [37].

In terms of usage, our code was implemented using the solvers and optimization routines provided by *Gurobi Optimization Package*, which is available at http://www.gurobi.com/.

4 Results and Comparisons

The performance of the proposed optimization scheme is assessed through a set of comparisons against well established geometry based layout construction techniques. More precisely, we employ four distinct metrics to quantitatively measure the quality of layouts produced by two distinct classes of techniques, namely: *overlap removal methods* and *visual board techniques*. Before presenting the comparison, we briefly describe the metrics as well as the data sets used in our tests.

4.1 Metrics and Data Sets

4.1.1 Quantitative Metrics

Orthogonal Ordering (O): measures the number of changes in both horizontal and vertical order of geometric entities [38]. More precisely, positions are sorted in increasing horizontal and vertical order and the number

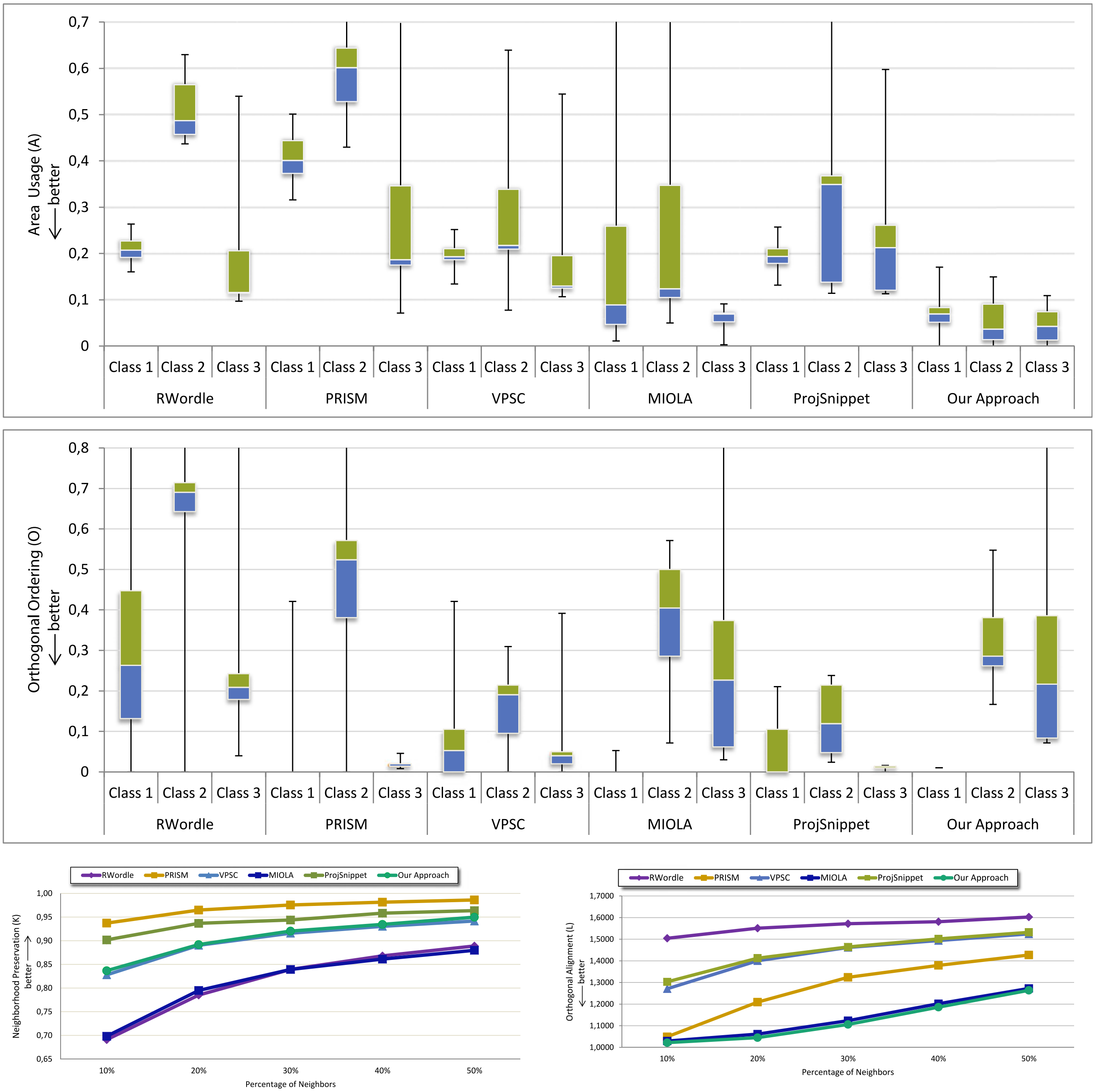


Fig. 5. Comparison against overlap removal techniques for the three initial classes of experiments.

of inversions in the lists provides the quality measure. In mathematical terms:

$$O = \sum_{s\in\{x,y\}} \sum_{i<j} inv_s^{(t)}(i,j), \tag{9}$$

$$inv_x^{(t)}(i,j) = \begin{cases} 1, & \text{if } (x_i^{(t)} - x_j^{(t)})(x_i^{(t-1)} - x_j^{(t-1)}) < 0 \\ 0, & \text{otherwise,} \end{cases} \tag{10}$$

where i, j are indices of the sorted lists before $(t-1)$ and after (t) layout arrangement. $inv_y^{(t)}(i,j)$ is defined analogously to $inv_x^{(t)}(i,j)$. Small values indicate better behavior.

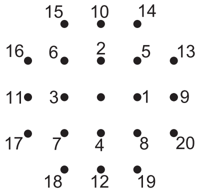
Neighborhood Preservation (K): computes the average percentage of the k-nearest neighbors of the initial configuration that are preserved in the final layouts [34]. More specifically, for each geometric entity we verify the percentage of its neighbors in the final layout that were also neighbors in the initial layout before optimization. Values close to 1 indicates 100 percent of preservation.

Area Usage (A): let r be the longer diagonal of the bounding box of the layout produced by a given method and A_i the area of the ith box in the layout. The area usage metric is defined similarly as [39]:

$$A = \frac{2\pi r^2}{\sum_i A_i}. \tag{11}$$

This metric gauges layout compactness and values close to $\pi/2$ are better.

Orthogonal Alignment (L)**:** Let $\{z_1, z_2, \ldots, z_r\}$ be points in $\mathbb{Z} \times \mathbb{Z}$ discretizing the interior of a disk, as illustrated in the inline figure. Moreover, let b_i be the center of a square in the layout and $b_{ij}, j = 1, \ldots, k$ be the center of the squares in the k-nearest neighbor of b_i. The orthogonal alignment metric is given by:



$$L = \frac{1}{nk} \sum_{i=1}^{n} \sum_{b_{ij}} \max_{s} \left\{ \left| \frac{< z_s, b_{ij} - b_i >}{\|z_s\| \|b_{ij} - b_i\|} \right| \right\}, \tag{12}$$

where n is the number of geometric objects in the layout and $< \cdot, \cdot >$ accounts for the dot product. This metric gauges how much a layout deviates from a regular grid, the closer to 1 the better.

Metrics above have been chosen because they measure important properties a layout must hold, namely, similarity preserving, compactness, and orthogonal alignment.

4.1.2 Data Sets and Test Settings

We divided the experiments in four groups: the first three groups of tests assess *overlap removal methods* while the last group evaluates the performance of our optimization scheme against *visual board techniques*.

In the first test we randomly draw points inside a rectangular two-dimensional domain, which represents the display area. The rectangle is discretized as a regular grid with dimensions $1{,}600 \times 1{,}200$. Cells of the grid that do not contain any point are discarded. The remaining cells define the boxes to be arranged in the layout. As no overlap exists at this point, we enforce the cells to overlap by uniformly increasing their area in 10 percent. Forcing overlap is mandatory since most overlap removal methods cannot start up from an overlap free arrangement. Notice that this is not the case for our approach, which also operates on overlap free layouts in order to optimize area usage and structure. Fifteen configurations, with 50 random points each, make up the first group of tests (called *class 1* in Fig. 5). The second experiment is similar to the first one, except we use a three level adaptive grid to generate the geometric primitives (*class 2* in Fig. 5).

In the third experiment (*class 3* in Fig. 5) we use real data set containing high-dimensional instances which are mapped to the visual space using the LSP multidimensional projection scheme. The layout is built from the projected points using a three level adaptive grid. Five data sets, taken from a public repository (available at http://infoserver.lcad.icmc.usp.br/infovis2/DataSets), have been employed in the third experiment. One data set contains a collection of snippets retrieved from a web search using "Scientific Visualization" as query expression; two data sets contain news from BBC, CNN, and REUTERS; the fourth data set corresponds to abstracts from IEEE InfoVis 2004 conference; and the last data set contains metadata from a scientific paper collection. Bag-of-words were extracted from textual content and used to define the similarity between instances. Those distances are used as input to the LSP projection method [34], which maps instances to the visual space.

Finally, in the fourth group of tests we have randomly generated squares in a $2D$ square domain, as depicted in Fig. 8. Those layouts are used as input to existing visual board techniques.

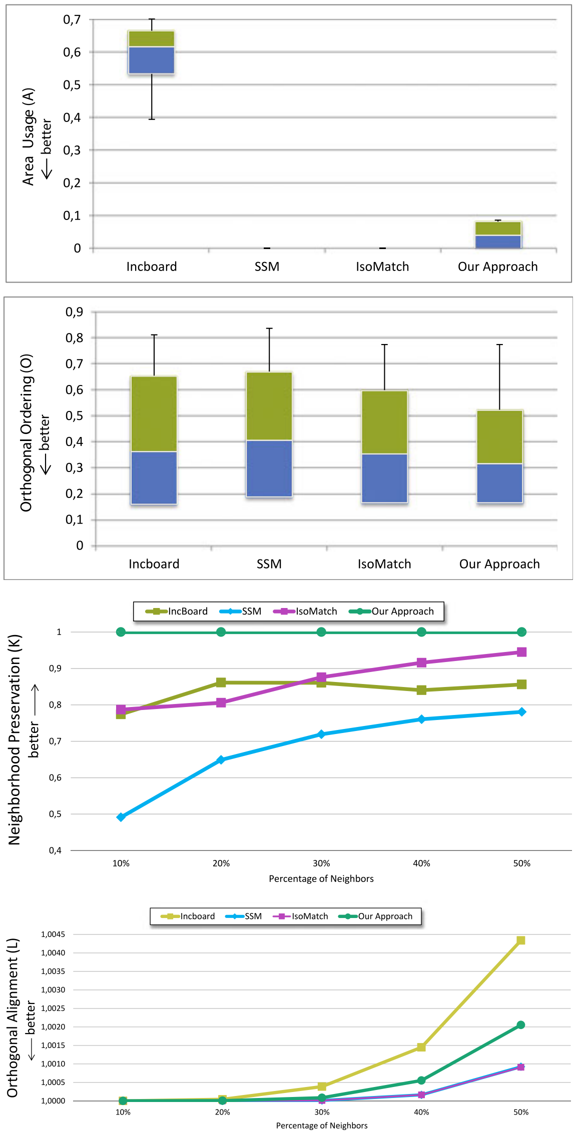


Fig. 6. Comparison with visual board techniques for the last class (taken from Fig. 8).

4.2 Comparisons

The metrics previously described are used to assess the effectiveness of our approach when compared against well known overlap removal techniques. Precisely, VPSC [27], PRISM [23], RWordle-C [9], MIOLA [32], and ProjSnippet [31] are used as basis for comparisons. These methods have been chosen either because they are widely used or due to their good performance reported in the literature. We also compare our approach against three visual board techniques, namely, Incboard [5], SSM [7] and IsoMatch [40].

Fig. 5 shows quantitative results obtained from the metrics described above. Notice that our approach clearly outperforms most techniques as to area usage (top row) and orthogonal alignment (bottom right). Regarding orthogonal ordering (middle row), our approach turns out to be competitive, performing better for class 1. In terms of neighborhood preservation (bottom row left), our approach also performed well, being surpassed only by ProjSnippet and

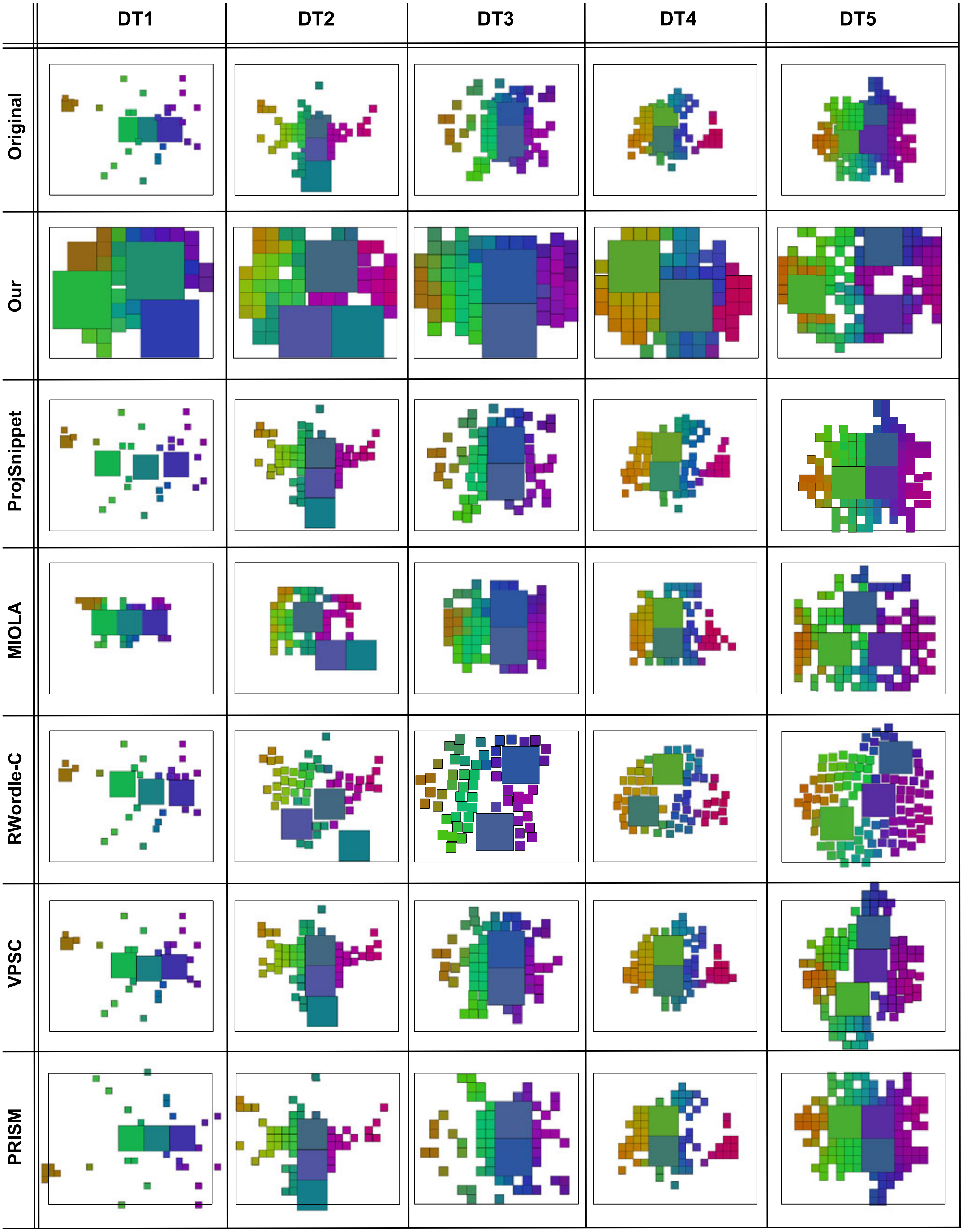


Fig. 7. Layouts produced by our approach, ProjSnippet, MIOLA, RWordle-C, VPSC and PRISM in five datasets.

PRISM, which are known to preserve neighborhoods well. Fig. 6 brings quantitative comparison against visual board techniques. Notice that the proposed method outperforms IncBoard and behaves nicely even when compared to SSM and IsoMatch, which have optimal area usage (notice the vanish box plots) and cell alignment.

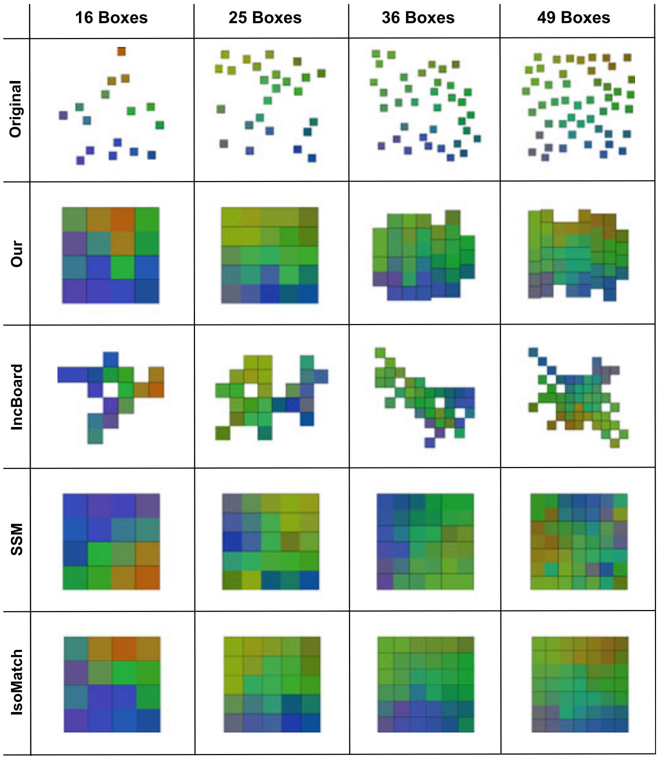


Fig. 8. Qualitative comparison of the proposed optimization scheme against visual board techniques Incboard [6], SSM [8] (when taking the best result from SSM w.r.t. their objective function after $1{,}000$ executions) and IsoMatch [41].

Figs. 7 and 8 depict qualitative results comparing our approach with overlap removal and visual board techniques, respectively. Notice that the proposed method gives rise to well structured layouts where neighborhoods (indicated by color map) are nicely preserved. Moreover, our approach makes a better use of display area, thus improving readability and content analysis.

5 Applications

In this section we present three different applications of our technique, namely, image gallery construction, textual documents analysis, and video data set visualization. In those applications, effective data exploration is enabled if requirements such as object size, semantic and overlap-free arrangements, scalability, and optimal area usage are combined in an optimal manner, showing the relevance of our methodology to applications as the ones presented in this section.

The image gallery application, depicted in Fig. 9, aims at visualizing image data sets such that highly similar images, which tend to be projected close to each other in the visual space, are summarized in large icons, while images with discriminative feature are represented in small icons. To build the image gallery we extract 96 features from color attribute. More specifically, each image is split in 16 non-overlapping regular regions covering the image from which the first and second statistical moments for each R,G,B channel are extracted. Feature vectors representing the images are projected in the visual space using the Least Square Projection [34] method. A three level adaptive grid is constructed and the proposed layout optimization scheme is triggered in order to arrange the cells in a structured overlap free layout. Large grid cells containing several similar images give rise to iconic representations that summarizes the underlying images. Such iconic representation is built by blending a randomly chosen subset of the underlying images using five levels of Laplacian pyramid [41]. In the

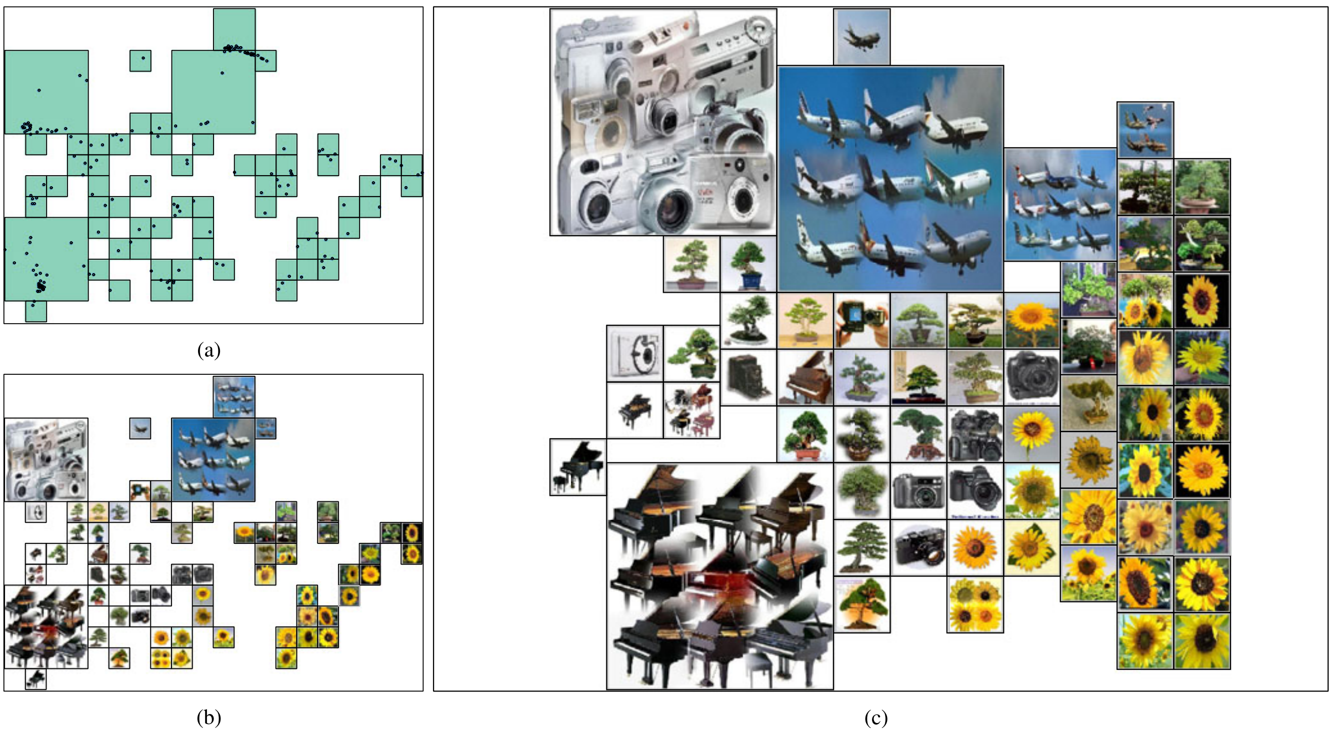
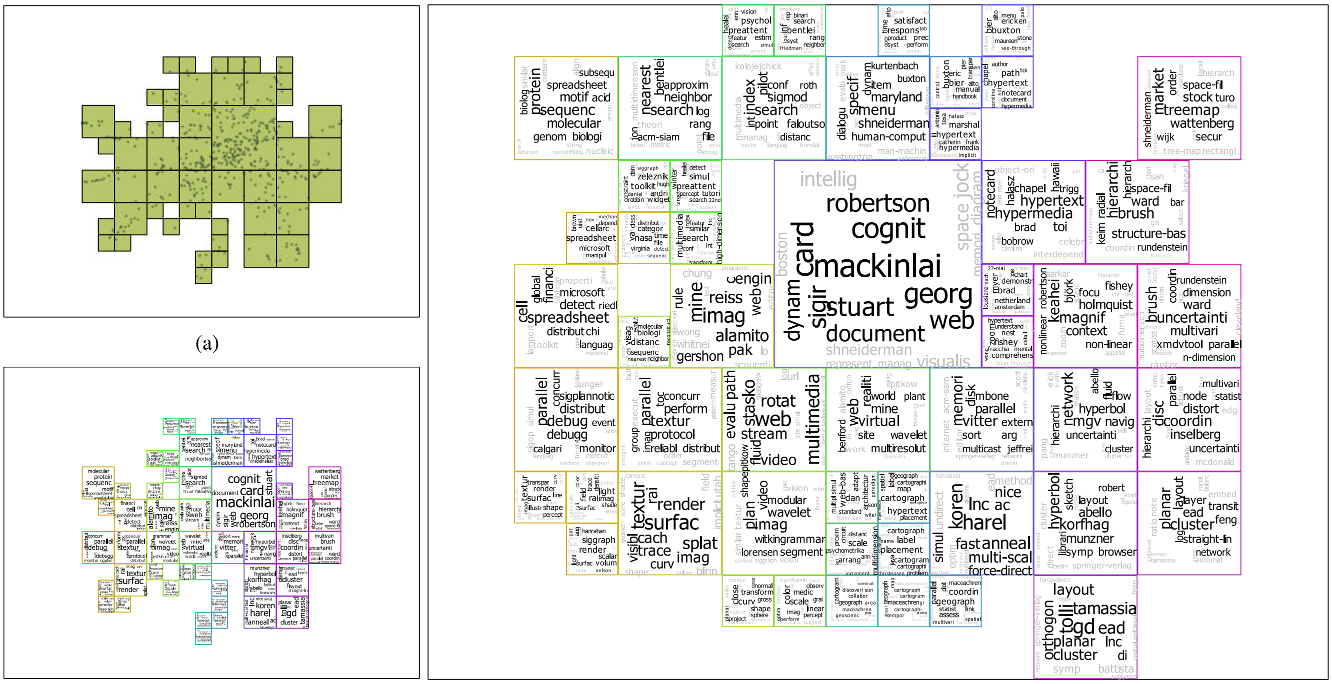


Fig. 9. Image gallery application: (a) an adaptive grid is built from projected images; (b) an initial iconic representation is generated; (c) the layout is optimized to make an optimal use of area while preserving the neighborhood of the grid cells.



(a)

(b)

(c)

Fig. 10. Visualizing a dataset containing 515 documents using word clouds.(a) each document is represented by a point embedded in 2D space where adaptive grid is imposed, (b) a word cloud based on term-frequency of documents contained in each cell of grid is built.(c) shows the optimized layout using the proposed method. Notice the size increasing that allows us to increase the number of words by each cell (colored in lightgray).

example shown in Fig. 9, cells containing less then four images are represented by a single representative image, cells containing four to eight images gives rise to a four blended image icon, and icon with nine blended images is used to represent cells with nine or more images. Images used in this application were obtained from [42].

The second application concerns textual document visualization. Following the conventional tf-idf bag-of-words construction [43] and stemming [44], one can yield feature vectors from which our methodology can straightly be applied. In the application illustrated in Fig. 10, we visualize a collection of 515 related papers published in the IEEE VisWeek Conference 2004. Each cell is textured with a word cloud built from keywords extracted from the documents contained in the cell while font size and opacity is used to highlight the most relevant keywords whose ranking is computed using the method described in [10]. Notice that the layout is easy to read, mainly due to the effective use of display area. Moreover, it is not difficult to note that the grid-like structure makes the visual identification of similar documents an easier task.

Our last application regards video visualization, as illustrated in Fig. 11. We build a collection of 300 videos from Youtube querying six distinct topics, namely, linux, civil war, fifa world cup, hawk, guitar and information visualization. Textual information associated to each video is processed to generate a feature vector that represents the video in a high-dimensional space. The proposed methodology with three level grid refinement is employed to build a layout where larger cells are textured as word clouds while the cells in the lowest refinement level (smaller cells) are textured with a snippet build from a screenshot of a randomly chosen video contained in the cell as well as textual information containing title, description and URL of the video. Word clouds are generated using keywords extracted from the text associated to the video. As in the previous application, font sizes and priority-based opacity are used to highlight the importance of each keyword, which are ranked as in [10].

Notice how our approach was able to semantically organize the layout according to the distinct topics, keeping videos related to "guitar" grouped on the left part of the layout, "linux" and "information visualization" on the top right, "hawk" in the center, "fifa world cup" on the bottom right, and "civil war" centered on the bottom. Interestingly, the smaller cells in the layout contain mainly videos about civil war and music videos related to the song "Civil War"

Fig. 11. Visualizing 300 videos extracted from Youtube querying six different topics.

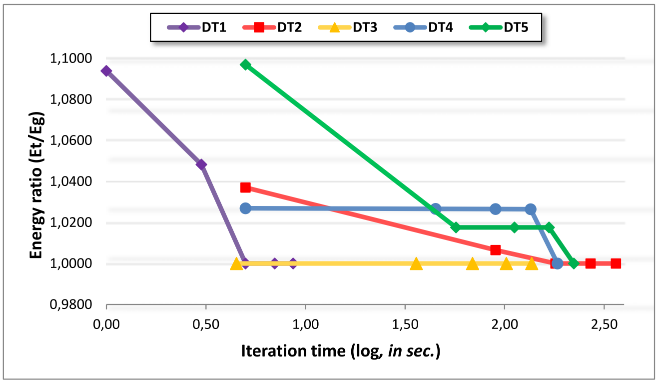


Fig. 12. Computational time versus optimization convergence rate for the layouts from Fig. 7.

by Guns N'Roses. Moreover, besides being well structured and semantically organized, the layout resulting from our methodology is compact and easy to read due to its effective use of display area. One interesting fact of all generated layouts is the high level of interactivity for the exploration task, i.e., users are allowed to choose a cell summarizing some content (from all levels except the last one) and then further explore that cell in more detail by recursively applying the whole layout construction process only within that cell.

6 Discussion and Limitations

Quantitative and qualitative comparisons presented in Section 4 attest the quality of our layout arrangement mechanism. In fact, our approach clearly outperforms most existing techniques with respect to properties such as area usage, orthogonal ordering and layout alignment, in addition to perform well as to neighborhood preservation. Therefore, our approach is able to handle several requirement simultaneously, making it an attractive alternative for several applications. The applications described in Section 5 show the versatility of our approach in scenarios ranging from image gallery construction to word cloud based text visualization. Moreover, the layouts resulting from our methodology turned out to be clean, well structured, and easy to read.

Another interesting aspect of our approach is that the only parameter to be set is the number of refinement levels in the adaptive grid. In our tests we use three levels of refinement and we notice that the layout is not so easy to read when more than five levels are used. However, the appropriate number of refinement levels depends on the underlying application and the size of screen. A weakness of our methodology is that dense and tight data cluster may be cut into multiple cells during the grid construction. This problem can be mitigated through density-aware grid generation, an improvement to be incorporated in our methodology.

The quality of the semantic relation among grid cells also depends on the effectiveness of the multidimensional projection scheme in preserving neighborhoods during the mapping process. Visually encoding uncertainties as to neighborhood relation can help users during data exploration. However, this is an aspect that has not been properly addressed even in the context of multidimensional projection methods.

From our experimental analysis we also notice that it is not necessary to run the optimization procedure until convergence. In other words, good layouts are obtained after a few hundred iteration steps, being unnecessary to wait until a local minimum is reached. This fact makes the proposed methodology also attractive in terms of computational times. For instance, the layouts presented in Section 5 took less than six seconds to be produced. However, the algorithm can take a few minutes to fully converge if a local minimum is sought. To numerically illustrate this fact, we show in Fig. 12 the computational time spent by our methodology to reach a certain number of iterations versus the ratio between the resulting and global minimization energies E_t/E_g for each layout from Fig. 7. Note from the y-axis of the graph that how close to 1 the energy rates are, even for the layouts produced with a reduced number of iteration steps.

7 Conclusion

We introduced a novel methodology to build geometric layouts to visualize data. In contrast to existing techniques, the proposed method makes use of a novel optimization procedure that is able to handle several requirements simultaneously, yielding structured overlap-free arrangements while still ensuring a semantic relation among neighbor entities. Furthermore, the method makes optimal use of display area, rendering it an effective and flexible tool for applications varying from image gallery construction to video data set visualization.

We are currently investigating interactive mechanisms to enable a free navigation throughout the layout as well as dynamic user-driven layout updates.

Acknowledgments

The authors acknowledge the financial support from FAPESP (#2011/22749-8 #2013/00191-0 and #2014/16857-0) and CNPq (#302643/2013-3).

References

[1] K. Koh, B. Lee, B. Kim, and J. Seo, "Maniwordle: Providing flexible control over wordle," *IEEE Trans. Vis. Comput. Graph.*, vol. 16, no. 6, pp. 1190–1197, Nov. 2010.
[2] J. Bertin, *Graphics and Graphic Information-Processing*, de Gruyter. Ann Arbor, MI, USA: The University of Michigan, 2007.
[3] E. Tufte, *Envisioning Information*, series Envisioning Information, no. 914. Graphics Press, Cheshire, CT, USA, 1990.
[4] S. V. d. Elzen and J. J. van Wijk, "Small multiples, large singles: A new approach for visual data exploration," *Comput. Graph. Forum*, vol. 32, no. 3pt2, pp. 191–200, 2013.
[5] R. D. Pinho, M. C. Oliveira, and A. Andrade Lopes, "An incremental space to visualize dynamic data sets," *Multimedia Tools Appl.*, vol. 50, no. 3, pp. 533–562, 2010.
[6] G. Strong and M. Gong, "Data organization and visualization using Self-sorting map," in *Proc. Graph. Interface*, 2011, pp. 199–206.
[7] G. Strong and M. Gong, "Self-sorting map: An efficient algorithm for presenting multimedia data in structured layouts," *Multimedia, IEEE Trans.*, vol. 16, no. 4, pp. 1045–1058, Jun. 2014.
[8] J.-D. Fekete and C. Plaisant, "Interactive information visualization of a million items," in *Proc. IEEE Symp. Inf. Vis.*, 2002, pp. 117–124.
[9] H. Strobelt, M. Spicker, A. Stoffel, D. Keim, and O. Deussen, "Rolled-out wordles: A heuristic method for overlap removal of 2d data representatives," *Comput. Graph. Forum*, vol. 31, no. 3pt3, pp. 1135–1144, 2012.
[10] F. V. Paulovich, F. Toledo, G. P. Telles, R. Minghim, and L. G. Nonato, "Semantic wordification of document collections," *Comput. Graph. Forum*, vol. 31, no. 3pt3, pp. 1145–1153, 2012.

[11] W. Javed, B. McDonnel, and N. Elmqvist, "Graphical perception of multiple time series," *IEEE Trans. Vis. Comput. Graph.*, vol. 16, no. 6, pp. 927–934, Nov./Dec. 2010.

[12] J. Kehrer, H. Piringer, W. Berger, and M. E. Groller, "A model for Structure-based comparison of many categories in small-multiple displays," *IEEE Trans. Vis. Comput. Graph.*, vol. 19, no. 12, pp. 2287–2296, Dec. 2013.

[13] R. Blanch and E. Lecolinet, "Browsing zoomable treemaps: Structure-aware multi-scale navigation techniques," *IEEE Trans. Vis. Comput. Graph.*, vol. 13, no. 6, pp. 1248–1253, Nov./Dec. 2007.

[14] B. Johnson and B. Shneiderman, "Tree-maps: A space-filling approach to the visualization of hierarchical information structures," in *Proc. 2nd Conf. Vis.*, 1991, pp. 284–291.

[15] M. Bruls, K. Huizing, and J. J. Van Wijk, "Squarified treemaps," in *Proc. Data Vis.*, 2000, pp. 33–42.

[16] B. Shneiderman and M. Wattenberg, "Ordered treemap layouts," in *Proc. IEEE Symp. Inf. Vis.*, 2001, pp. 73–73.

[17] B. Shneiderman. (1998–2013). Treemaps for Space-constrained visualization of hierarchies [Online]. Available: http://www.cs.umd.edu/hcil/treemap-history/

[18] C. Tuttle, L. G. Nonato, and C. Silva, "Pedvis: A structured, space-efficient technique for pedigree visualization," *IEEE Trans. Vis. Comput. Graph.*, vol. 16, no. 6, pp. 1063–1072, Nov./Dec. 2010.

[19] M. Balzer, O. Deussen, and C. Lewerentz, "Voronoi treemaps for the visualization of software metrics," in *Proc. ACM Symp. Softw. Vis.*, 2005, pp. 165–172.

[20] B. B. Bederson, B. Shneiderman, and M. Wattenberg, "Ordered and quantum treemaps: Making effective use of 2d space to display hierarchies," *ACM Trans. Graph.*, vol. 21, no. 4, pp. 833–854, 2002.

[21] J. Wood and J. Dykes, "Spatially ordered treemaps," *IEEE Trans. Vis. Comput. Graph.*, vol. 14, no. 6, pp. 1348–1355, Nov./Dec. 2008.

[22] A. Nocaj and U. Brandes, "Organizing search results with a reference map," *IEEE Trans. Vis. Comput. Graph.*, vol. 18, no. 12, pp. 2546–2555, Dec. 2012.

[23] E. R. Gansner and Y. Hu, "Efficient node overlap removal using a proximity stress model," in *Proc. 16th Int. Symp. Graph Drawing*, 2009, pp. 206–217.

[24] X. Huang, W. Lai, A. Sajeev, and J. Gao, "A new algorithm for removing node overlapping in graph visualization," *Inf. Sci.*, vol. 177, no. 14, pp. 2821–2844, 2007.

[25] C.-C. Lin, H.-C. Yen, and J.-H. Chuang, "Drawing graphs with nonuniform nodes using potential fields," *J. Vis. Languages Comput.*, vol. 20, no. 6, 2009, pp. 385–402.

[26] D. Harel and Y. Koren, "Drawing graphs with Non-uniform vertices," in *Proc. Work. Conf. Adv. Vis. Interfaces.*, 2002, pp. 157–166.

[27] T. Dwyer, K. Marriott, and P. J. Stuckey, "Fast node overlap removal," in *Proc. 13th Int. Symp. Graph. Drawing.*, 2006, pp. 153–164.

[28] K. Marriott, P. Stuckey, V. Tam, and W. He, "Removing node overlapping in graph layout using constrained optimization," *Constraints J.*, vol. 8, no. 2, pp. 143–171, 2003.

[29] S. Kieffer, T. Dwyer, K. Marriott, and M. Wybrow, "Incremental grid-like layout using soft and hard constraints," in *Proc. 21st Int. Symp. Graph Drawing*, 2013, pp. 448–459.

[30] U. Ruegg, S. Kieffer, T. Dwyer, K. Marriott, and M. Wybrow, "Stress-minimizing orthogonal layout of data flow diagrams with ports," in *Proc. 22nd Int. Symp. Graph. Drawing*, 2014, pp. 1–16.

[31] E. Gomez-Nieto, F. S. Roman, P. Pagliosa, W. Casaca, E. S. Helou, M. C. F. de Oliveira, and L. G. Nonato, "Similarity preserving Snippet-based visualization of web search results," *IEEE Trans. Vis. Comput. Graph.*, vol. 20, no. 3, pp. 457–470, Mar. 2014.

[32] E. Gomez-Nieto, W. Casaca, L. G. Nonato, and G. Taubin, "Mixed integer optimization for layout arrangement," in *Proc. 26th Conf. Graph., Patterns Images*, 2013, pp. 115–122.

[33] T. Dwyer and L. Nachmanson, "Fast Edge-routing for large graphs," in *Proc. 17th Int. Symp. Graph Drawing*, 2010, pp. 147–158.

[34] F. V. Paulovich, L. G. Nonato, R. Minghim, and H. Levkowitz, "Least square projection: A fast High-precision multidimensional projection technique and its application to document mapping," *IEEE Trans. Vis. Comput. Graph.*, vol. 14, no. 3, pp. 564–575, May/Jun. 2008.

[35] J. Leung, T. Tam, C. Wong, G. Young, and F. Chin, "Packing squares into a square," *J. Parallel Distrib. Comput.*, vol. 10, no. 3, pp. 271–275, 1990.

[36] K. Jansen and L. Pradel, *How to Maximize the Total Area of Rectangles Packed into a Rectrangle?* Inst. fur informatik, Cristian-albrechts-univ., kiev, 2009.

[37] J. Lee and S. Leyffer, *Mixed Integer Nonlinear Programming (The IMA Volumes in Mathematics and its Applications)*. New York, NY, USA: Springer, 2012.

[38] K. Misue, P. Eades, W. Lai, and K. Sugiyama, "Layout adjustment and the mental map," *J. Vis. Languages Comput.*, vol. 6, no. 2, pp. 183–210, 1995.

[39] H. Samet, *Foundations of Multidimensional and Metric Data Structures*. San Francisco, CA, USA: Morgan Kaufmann, 2005.

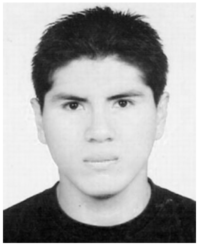
[40] O. Fried, S. DiVerdi, M. Halber, E. Sizikova, and A. Finkelstein, "IsoMatch: Creating informative grid layouts," *Comput. Graph. Forum*, vol. 34, no. 2, pp. 155–166, May 2015.

[41] M. N. Do and M. Vetterli, "Framing pyramids," *IEEE Trans. Signal Process.*, vol. 51, no. 9, pp. 2329–2342, Sep. 2003.

[42] L. Fei-Fei, R. Fergus, and P. Perona, "Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories," in *Proc. Comput. Vis. Pattern Recog. Workshop* 2004, p. 178.

[43] G. Salton, "Developments in automatic text retrieval," *Science*, vol. 253, no. 5023, pp. 974–980, 1991.

[44] H. P. Luhn, "The automatic creation of literature abstracts," *IBM J. Res. Develop.*, vol. 2, no. 2, pp. 159–165, Apr. 1958.



Erick Gomez-Nieto received the BSc degree in 2009 in informatics engineering from San Pablo Catholic University—Arequipa, Peru. He received the MSc degree in 2012 in computer science from University of São Paulo—SP, Brazil. He was a lecturer/researcher at Computer Science School, San Pablo Catholic University, where he was awarded by his notable scientific research, in 2012/2013. He is currently working toward the PhD degree at the University of São Paulo. His research interests include in interactive visualization, visual analytics, and geometry processing. He is a student member of the IEEE.

Wallace Casaca received the BSc and master's degrees in pure and applied mathematics from São Paulo State University (UNESP), Brazil, in 2008 and 2010, respectively. During 2010-2014, he worked toward the PhD degree in computer sciences and applied mathematics at the University of São Paulo (USP), Brazil. As part of his doctoral studies, he was also a visiting researcher at Brown University, School of Engineering, and in cooperation with MIT, Media Lab. His research interests include image segmentation, image restoration, image retrieval, information visualization, optimization techniques, partial differential equations, and numerical analysis.

Danilo Motta received the BSc degree in computer science by the Federal University of Mato Grosso do Sul, in 2010, and the MSc degree in computer science and computational mathematics from the University of São Paulo—SP, Brazil, in 2013. He is currently working toward the PhD degree and a member of the Visual and Geometry Processing Group—VGPG at ICMC-USP. His current interests include data processing and visualization, optimization and applied machine learning.

Ivar Hartmann received the MSc degree from the Catholic University of Rio Grande do Sul, Brazil, and the LLM degree from Harvard Law School. He is an S.J.D. candidate at the State University of Rio de Janeiro. He is a professor at FGV Law School in Rio de Janeiro, where he has taught cyberlaw, programming for lawyers and theory of justice. He coordinates the "Supreme Court in Numbers" project, which employs computer science to empirical legal research in order to produce data on tens of millions of lawsuits from different Brazilian courts. The project team is composed of nearly a dozen researchers with diverse backgrounds such as Law, Computer Engineering and Statistics. His own research and publications cover topics of Constitutional, Private, Enviromental, Procedural and Cyberlaw, as well as Empirical Legal Research.

Gabriel Taubin received the Licenciado en Ciencias matematicas degree from the University of Buenos Aires, Argentina, and the PhD degree in electrical engineering from Brown University. In 1990, he joined the IBM Research Division, where he held various positions, including a research staff member and research manager. In 2003, he joined the Brown University School of Engineering as an associate professor of engineering and computer science. During the 2000-2001 academic year, on sabbatical from IBM, he was a visiting professor of electrical engineering at the California Institute of Technology. During the Spring semester of 2010, on sabbatical from Brown, he was a visiting associate professor of media arts and sciences at MIT. He was the editor-in-chief of the *IEEE Computer Graphics and Applications Magazine* from 2010 to 2013, and has served as a member of the Editorial Board of the *Geometric Models Journal*, and as associate editor of the *IEEE Transactions of Visualization and Computer Graphics*. He has contributed to the field called Digital Geometry Processing with methods to capture 3D shape, for surface reconstruction, geometric modeling, geometry compression, progressive transmission, signal processing, and display of discrete surfaces. The 3D geometry compression technology that he have developed at IBM was incorporated into the MPEG-4 standard, and became an integral part of IBM products. He was named IEEE fellow for his contributions to the development of three-dimensional geometry compression technology and multimedia standards, won the Eurographics 2002 Günter Enderle Best Paper Award, and was named IBM Master Inventor.

Luis Gustavo Nonato received the PhD degree in applied mathematics from the Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro—Brazil, in 1998. He is currently a full professor at the Instituto de Ciências Matemáticas e de Computação (ICMC)—Universidade de São Paulo (USP)—Brazil. He spent a sabbatical leave in the Scientific Computing and Imaging Institute at the University of Utah from 2008 to 2010. Besides having served in several program committees, including IEEE Visualization and Eurovis, he was a member of the editorial board of Computer Graphics Forum and the president in the Special Committee on Computer Graphics and Image Processing of Brazilian Computer Society. Currently he leads the Visual and Geometry Processing Group at ICMC-USP. He is a member of the IEEE.