Mixed Integer Optimization for Layout Arrangement

Erick Gomez-Nieto*, Wallace Casaca†, Luis Gustavo Nonato†, Gabriel Taubin‡

* San Pablo Catholic University, Peru
† ICMC, University of São Paulo, São Carlos, Brazil
‡ School of Engineering, Brown University, USA

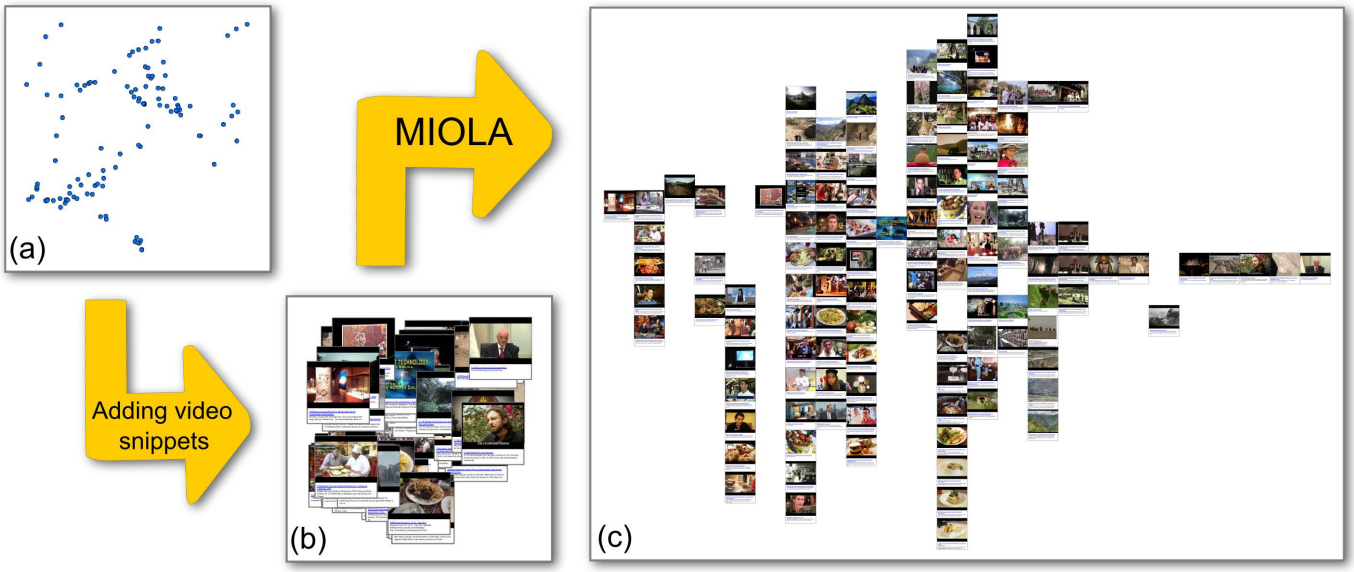


Fig. 1. Practical application in the context of video snippet visualization. (a) High-dimensional data is generated taking as input the Youtube's query "Peru" (120 instances) and, then, mapped to the visual space using a multidimensional projection technique. (b) Textual video snippets are embedded into rectangular boxes positioned according to the projected points. Boxes containing snippets overlap considerably, hampering visualization. (c) Our technique (MIOLA) is, then, applied in order to rearrange the boxes while still preserving neighborhood structures.

***Abstract*—Arranging geometric entities in a two-dimensional layout is a common task for most information visualization applications, where existing algorithms typically rely on heuristics to position shapes such as boxes or discs in a visual space. Geometric entities are used as a visual resource to convey information contained in data such as textual documents or videos and the challenge is to place objects with similar content close to each other while still avoiding overlap. In this work we present a novel mechanism to arrange rectangular boxes in a two-dimensional layout which copes with the two properties above, that is, it keeps similar object close and prevents overlap. In contrast to heuristic techniques, our approach relies on mixed integer quadratic programming, resulting in well structured arrangements which can easily be tuned to take different forms. We show the effectiveness of our methodology through a comprehensive set of comparisons against state-of-art methods. Moreover, we employ the proposed technique in video data visualization, attesting its usefulness in a practical application.**

***Keywords*-Overlap removal; Neighborhood preservation.**

I. Introduction

Arranging geometric entities in a two-dimensional space while avoiding overlaps is an NP-Hard problem investigated in many different contexts. The optimization community, for instance, faces the problem when solving the bin packing problem [1], [2], typically through constraint optimization. Overlap removal is also investigated in the context of graph drawing, where physical models such as force-based schemes [3], [4], [5] and spring systems [6], [7] are the methods used to position geometric entities representing graph nodes in the visual space. Overlap removal algorithms for bin packing optimization and graph drawing differ as to the mathematical formulation as well as the nature of overlap removal problem. More precisely, graph drawing has an additional constraint related to the closeness (and distance) of nodes, that is, nodes have to be placed in the visual space such that the neighborhood of the nodes are preserved as much as possible. Another issue with the physical-inspired overlap removal techniques is the lack of guarantees as to convergence and quality of the resulting layout. Such issue is addressed by optimal graph drawing techniques such as the ones proposed by Dwyer et al. [8] and Marriott et al. [9], both of which relying on quadratic optimization. The main problem with these optimal approaches is that the energy to be minimized depends on intersection tests, thus introducing several cases to be handled by the algorithm. An interesting discussion about graph drawing literature can be found in [10].

The overlap removal problem also appears in the context of information visualization as a mechanism to generate data classification [11], word clouds [12], [13] and visual boards [14]. In contrast to optimal and physical-based methods, most of the techniques used in information visualization rely on heuristic algorithms to build the layout. Heuristics such as Rolled-out Wordles [15] can even take into account information of similarity to place similar objects close to each other while avoiding overlaps. The main advantage of heuristic techniques is their low computational cost, which enable to build large layouts quite efficiently but there is no guarantee in always finding a unique solution.

In this work we propose a novel technique to tackle the problem of arranging rectangular boxes in the visual space so as to place objects representing similar content close to each other while avoiding overlaps. We formulate the problem as a *Mixed Integer Quadratic Programming Problem* (MIQP), which enables well structured layouts. In contrast to other optimal methods that take into account the similarity between instances, our approach does not rely on intersection tests, making the algorithm simpler to implement. Moreover, our technique is quite flexible, being able to generate different layouts by just handling optimization constraints.

The practical usefulness of our method is demonstrated in a video data set visualization application. Textual information associated to each video is used to measure the similarity between them, that is, the similarity between videos is computed using bag-of-words derived from textural information and the cosine metric. The similarity measure is, then, used by a multidimensional projection technique that maps the data to the visual space, where the boxes representing the videos are arranged by the proposed optimization mechanism, generating the final layout.

Contributions In summary, the main contributions of this work are:

- A new mathematical formulation, which we call *MIOLA (Mixed Integer Optimization for Layout Arrangement)*, for the problem of arranging rectangular boxes in the visual space so as to place similar entities close while avoiding overlaps.
- *MIOLA* combines flexibility and capability in generating different layouts while still solving the box-overlap removal problem.
- A video visualization application using *MIOLA* that allows for handling and exploring video data sets.

II. Overlap Removal by Optimization

Let $B = \{B_1, B_2, ..., B_n\}$ be a set of n rectangular boxes arranged in the visual space such that the neighborhood structure of the boxes reflects a property of interest. For instance, if a data set is mapped to the visual space using a multidimensional projection technique and a box is centered on each projected data, the resulting arrangement makes neighbor boxes correspond to similar data. Boxes in this arrangement, however, should overlap considerably, impairing the visualization of individual boxes. In order to make each box visible, one has to displace the boxes in the two-dimensional space so as to remove overlaps, but preserving the initial neighborhood structures to keep similar objects close to each other. As described next, we formulate the problem above as a mixed integer quadratic programming optimization.

A. Problem Statement

Let $B = \{B_1, B_2, ..., B_n\}$ be a set of n rectangular boxes initially positioned in a two-dimensional space. Each box B_i is specified by a four dimensional vector $B_i = (x_i, y_i, w_i, h_i) \in \mathbb{R}^4$, where (x_i, y_i), $w_i > 0$, $h_i > 0$ are the centroid, width and height of B_i, respectively (see Fig. 2(a)). Two boxes B_i and B_j do not overlap if and only if one of the following inequalities holds:

$$|x_j - x_i| \geq \frac{w_i + w_j}{2} \quad \text{or} \quad |y_j - y_i| \geq \frac{h_i + h_j}{2} \tag{1}$$

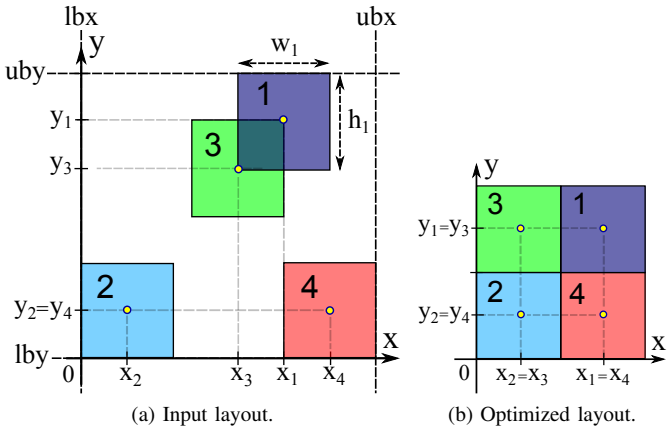
We refer to the inequalities in (1) as *non-overlap constraints*.

Moreover, the boxes must respect the bounds of the visualization window during displacement, that is,

$$\begin{aligned} &\frac{w_i}{2} + lbx \leq x_i \leq ubx - \frac{w_i}{2} \quad \textit{(x lower/upper bounds)} \\ &\frac{h_i}{2} + lby \leq y_i \leq uby - \frac{h_i}{2} \quad \textit{(y lower/upper bounds)} \end{aligned} \tag{2}$$

where lbx, lby, ubx, uby are lower and upper visualization window bounds (constants) in x and y directions. If needed, visualization window bounds can also assume independent values for each box.

Equations (1) and (2) provide the conditions to be held so as to guarantee that boxes do not overlap and are inside a visualization window. However, those equations do not take into account neighborhood structures, thus neighbor boxes can be placed far apart from each other after the overlap removal process. One useful way to keep up the neighborhood



(a) Input layout. (b) Optimized layout.

Fig. 2. Inequalities (2) ensure that no boxes will not displayed offscreen while inequalities (3) preserve the order of the coordinates w.r.t. initial layout. (a) Input layout and (b) result after layout arrangement (orthogonal ordering: $x_{p_i} \leq x_{p_{i+1}}, p_1 = 2, p_2 = 3, p_3 = 1, p_4 = 4$; similarity for y case).

relationships is to preserve the relative order of the centroids of boxes, that is,

$$\begin{aligned} x_{p_1} \leq x_{p_2} \leq ... \leq x_{p_n} \ \textit{(x orthogonal order)} \\ y_{q_1} \leq y_{q_2} \leq ... \leq y_{q_n} \ \textit{(y orthogonal order)} \end{aligned} \tag{3}$$

where p, q: $\{1, 2, ..., n\} \rightarrow \{1, 2, ..., n\}$ are permutations of indices obtained by sorting the coordinates x and y of the centroids of boxes in the visual space (see Fig. 2).

Therefore, by moving the centroid of the boxes while ensuring Equations (1), (2), and (3) can generate an overlap free layout that preserves the initial neighborhood structures.

B. The MIQP Formulation

The problem of positioning the boxes B_i in the visual space so as to ensure that Equations (1), (2), and (3) hold can be formulated as a *Mixed Integer Quadratic Programming Problem* [16] as follows:

$$\begin{aligned} &\min_{z} \quad f(\mathbf{z}) = \frac{1}{2}\mathbf{z}^T Q \mathbf{z} = \sum_{i=1}^{n}\sum_{j=i+1}^{n} Dist^2(B_i, B_j) \\ &\text{subject to} \quad \begin{cases} A\mathbf{z} \leq \mathbf{b} \\ \mathbf{lb} \leq \mathbf{z} \leq \mathbf{ub} \end{cases} \\ &\mathbf{z} = \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \\ \mathbf{r} \end{bmatrix}, \quad \begin{matrix} \mathbf{x} = (x_1, x_2, ..., x_n)^\top \in \mathbf{R}^n \\ \mathbf{y} = (y_1, y_2, ..., y_n)^\top \in \mathbf{R}^n \end{matrix} \\ &\mathbf{r} = (r_{12}, \ldots, r_{1n}, r_{23}, \ldots, r_{2n}, \ldots, r_{n-1n})^\top, \ r_{ij} \in \{0, 1\} \end{aligned} \tag{4}$$

where $\mathbf{z}$ is the sought solution, $Dist(B_i, B_j)$ denotes the euclidian distance between (x_i, y_i) and (x_j, y_j), and vectors $\mathbf{lb}$ (lower bounds) and $\mathbf{ub}$ (upper bounds) are the visualization window bounds as defined in the inequalities (2). Q is the positive semi-definite matrix composed by blocks L given by:

$$Q = \begin{pmatrix} \left[\begin{array}{c|c} L & 0 \\ \hline 0 & L \end{array}\right] & \begin{matrix} 0 \\ 0 \end{matrix} \\ \begin{matrix} 0 & 0 \end{matrix} & 0 \end{pmatrix}, \qquad L = nI_d - ones(n, n),$$

where I_d is the identity matrix and $ones(n, n)$ is the $n \times n$ matrix with all entries equal one.

Matrix A and vector $\mathbf{b}$ are defined so as to incorporate the constraints (1) and (3). Precisely, the ordering given by (3) allows us to write:

$$x_{p_1} \leq x_{p_2} \leq ... \leq x_{p_n} \Rightarrow x_{p_i} - x_{p_{i+1}} \leq 0, \ i = 1...n-1, \tag{5}$$

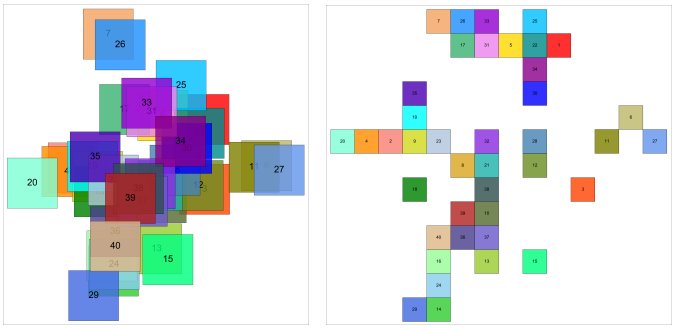
with a similar expression holding for y. Additionally, the sorting also allows us to get rid of the absolute value function in (1), that is,

$$x_{p_i} \leq x_{p_j} \Rightarrow |x_{p_j} - x_{p_i}| = x_{p_j} - x_{p_i}. \text{ Thus,}$$

$$x_{p_i} - x_{p_j} \leq -\frac{(w_{p_i} + w_{p_j})}{2}, \quad i < j, \tag{6}$$

with a similar expression holding for y. The variables r_{ij} allows for incorporating the *OR* condition defined in (1) into the optimization problem as follows:

$$\begin{aligned} &x_{p_i} - x_{p_j} \leq w_{ij} + M r_{ij} \Leftrightarrow y_{q_i} - y_{q_j} \leq h_{ij} + M(1 - r_{ij}), \\ &i < j, \quad r_{ij} \in \{0, 1\}, \end{aligned} \tag{7}$$



(a) Original layout. (b) Standard optimized layout.

Fig. 3. Result of our method to rearrange boxes with overlaps.

where $w_{ij} = -\frac{(w_{p_i} + w_{p_j})}{2}$, $h_{ij} = -\frac{(h_{q_i} + h_{q_j})}{2}$ and M is a very large constant. The rationale behind the construction in Equation (7) is that if $r_{ij} = 0$, the constraint $x_{p_i} - x_{p_j} \leq w_{ij}$ becomes mandatory while $y_{q_i} - y_{q_j} \leq h_{ij} + M$ is naturally satisfied as M is a large number (if $r_{ij} = 1$, we have the opposite situation, instead). Therefore, optimizing the centroid positions and the decision variables r_{ij} simultaneously allows us to find the optimal position for the boxes while respecting all the constraints as stated in Equations (1), (2), and (3).

Equations (5) and (7) are incorporated into matrix A using the auxiliar matrices C_x, C_y, D_x, D_y as follows:

$$A = \left(\begin{array}{c|c|c} C_x & 0 & 0 \\ 0 & C_y & 0 \\ \hline D_x & 0 & -MI_d \\ 0 & D_y & MI_d \end{array}\right) \qquad \mathbf{b} = \begin{pmatrix} \mathbf{c} \\ \mathbf{d} \end{pmatrix} \tag{8}$$

where $[C_x|C_y|\mathbf{c}]$ and $[D_x|D_y|\mathbf{d}]$ are built from constraints (5) and (7), respectively. Notice that vector $\mathbf{c}$ is null as stated in (5). However, this constraint can be relaxed so as to introduce more flexibility into the layout, as explained in subsection II-C. Fig. 3 illustrates the resulting layout from solving the standard MIQP (4). Notice that the orthogonal order is accurately preserved while the boxes are thoroughly spread in the visualization window.

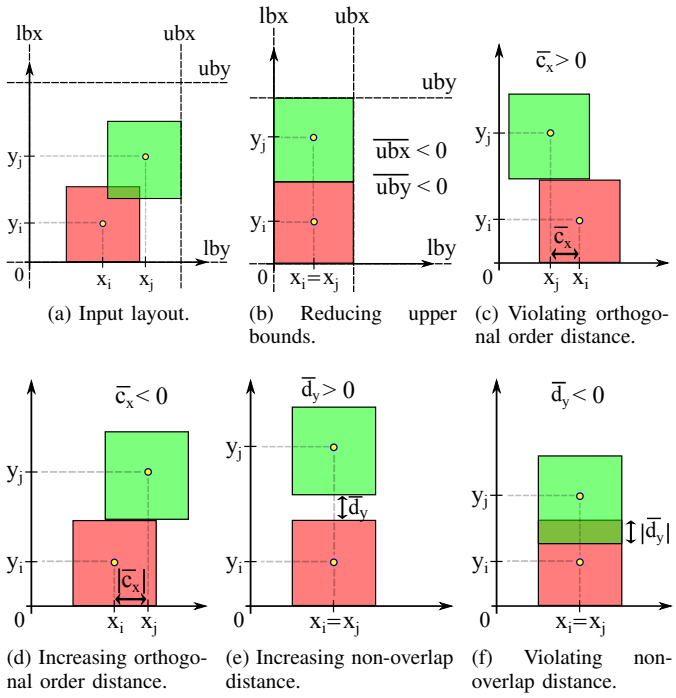
C. Relaxation of constraints

Besides providing accuracy and robustness when dealing with complex layouts, formulation (4) enables great flexibility in generating a variety of non-overlap arrangements. Different layouts can be easily obtained by introducing relaxation constants within lower $\mathbf{lb}$ and upper $\mathbf{ub}$ bounds in Equations (4) and/or vector $\mathbf{b} = [\mathbf{c}|\mathbf{d}]^T$ in (8) as follows:

$$\mathbf{lb}_{relax} = \mathbf{lb} + \overline{\mathbf{lb}}, \quad \mathbf{ub}_{relax} = \mathbf{ub} + \overline{\mathbf{ub}} \tag{9}$$

$$\mathbf{c}_{relax} = \mathbf{c} + \overline{\mathbf{c}}, \quad \mathbf{d}_{relax} = \mathbf{d} + \overline{\mathbf{d}}, \tag{10}$$

where $\overline{\mathbf{lb}} = (\overline{lb_i})$, $\overline{\mathbf{ub}} = (\overline{ub_i})$, $\overline{\mathbf{c}} = (\overline{c_k})$ and $\overline{\mathbf{d}} = (\overline{d_l})$ are the relaxation constants. Notice that relaxation constants can be individually chosen for each box in (9) or for each pair of boxes defined in (10). Moreover, each coordinate of $\overline{\mathbf{lb}}$, $\overline{\mathbf{ub}}$, $\overline{\mathbf{c}}$ and $\overline{\mathbf{d}}$ can assume both positive and negative values.



(a) Input layout. (b) Reducing upper bounds. (c) Violating orthogonal order distance.

(d) Increasing orthogonal order distance. (e) Increasing non-overlap distance. (f) Violating non-overlap distance.

Fig. 4. Geometric interpretation of relaxation constants. Fig.(a) shows the input layout while Fig.(b) depicts the optimization result after setting upper bound relaxation (< 0) in both directions. Fig.(c)-(d) show the maximum distance tolerance when violating the orthogonal order constraints and imposing a displacement between boxes, respectively (in x-direction). Fig.(e)-(f) yield the same previous conclusion w.r.t. non-overlap constraints but obtained by handling y-direction relaxation.

While $\overline{\mathbf{lb}}$ and $\overline{\mathbf{ub}}$ regulate the effective visual space of the boxes, $\overline{\mathbf{b}}$ and $\overline{\mathbf{c}}$ control the relaxation distances of orthogonal ordering and non-overlap constraints. Fig. 4 depicts the geometric interpretation when handling Equations (9)-(10).

III. Results, Comparisons and Evaluation

We start this section showing the usefulness of our methodology in a practical application devoted to video data visualization. The flexibility to generate different layouts by just tuning the constraints in the optimization is illustrated in several examples. Finally, we show the effectiveness of MIOLA by comparing it against state-of-art overlap removal techniques typically employed in information visualization applications. We denote the pair of numbers $VS = ([-S_x, S_y], [-S_y, S_y]) := (S_x, S_y)$ as the lower and upper bound constants in (9). Orthogonal order as well as non-overlap relaxation constants in (10) are denoted by the numbers *OO* and *NOL*. The minimization problem (4) is solved by an extension of the *Linear-Programming based Branch-and-Bound Algorithm* [17], available from the Gurobi library at http://www.gurobi.com/. Our code was implemented in MATLAB with support to C++ MEX routines and free optimization interface provided by [18].

A. Applications

Video Snippet Visualization Fig. 5 shows the layout resulting from MIOLA applied to video snippet visualization. Fig. 5(a) depicts the initial layout just after projecting video data obtained from Youtube, using the word "SIBGRAPI" as query. The projection is generated by applying the LSP multidimensional projection technique [19], measuring the similarity between video snippets through the cosine metric applied to feature vectors extracted from the textual data associated to the videos. More precisely, feature vectors are generated by extracting a bag-of-words [20] from the text associated to each video. Fig. 5(b) shows the resulting layout after optimization. Notice that *SIBGRAPI's Fast Forward* videos are nicely grouped on top left while videos related to *volume rendering, modeling and graphics* are arranged on middle/bottom left. Top right snippets in Fig. 5(b) are mainly related to videos presented in *SIBGRAPI's Video Festival* while bottom right videos are related to "dancing animation". The video visualization application clearly shows that MIOLA is able to arrange the video snippets in the visual space avoiding overlaps while preserving the proximity relation among videos.

B. Flexibility to generate different layouts

Flexibility and Adaptability Fig. 6(a)-(e) show the flexibility of MIOLA in producing different layouts by just modifying constraints in the optimization. Fig. 6(a) shows the original layout while Fig. 6(b) displays the resulting layout when no relaxation is used during optimization. Changing the constraints that define the lower and upper bounds of the visualization window give rise to a more compact layout, as shown in Fig. 6(c). We use colors and numbers to label the boxes, from which one can clearly see that MIOLA preserves neighborhoods quite nicely. An even more compact layout can be obtained by further squishing the visualization window and relaxing the orthogonality constraints, as depicted in Fig. 6(d). Fig. 6(e) shows an extreme case where the visualization window is set to be one-dimensional, that is, the centroid of each box must be place in a horizontal line. Notice that even in this drastic situation the algorithm was able to arrange the square respecting neighborhoods.

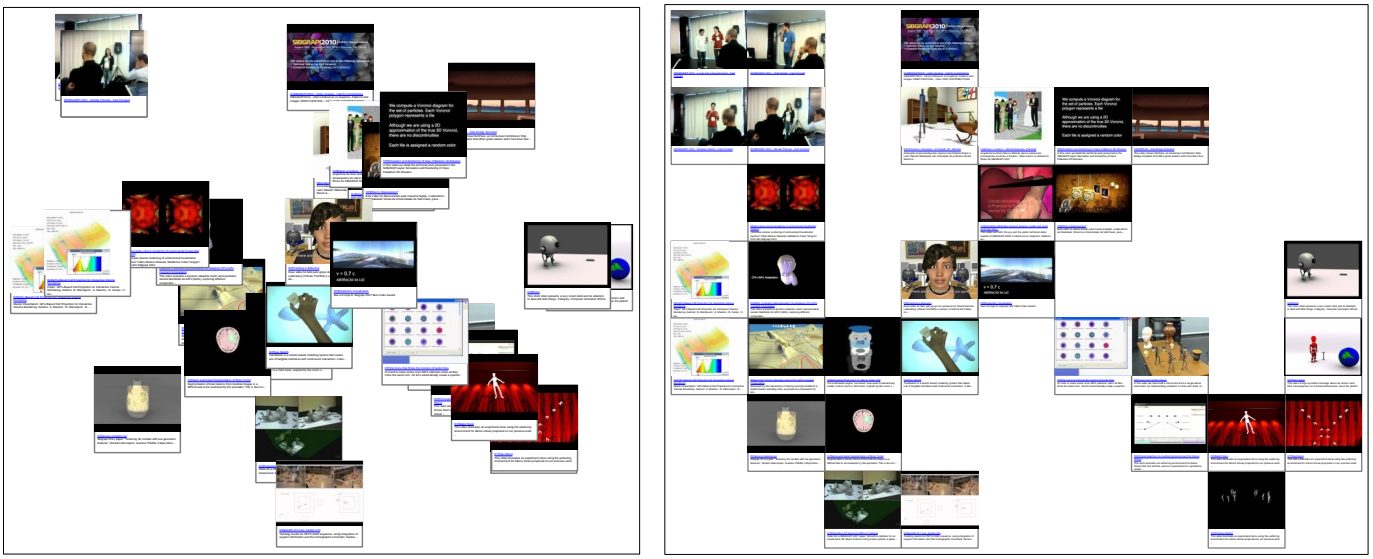
Shrinking scattered layouts MIOLA can also be used for shrinking sparse layouts. Fig. 7(b) shows the result of optimizing the spread layout depicted in Fig. 7(a). One can see that the optimization produces a pleasant layout in terms of compactness and organization. Moreover, neighborhood structures have also been nicely preserved.

C. Comparisons against state-of-art methods

The effectiveness of MIOLA is assessed by quantitatively measuring its accuracy as to layout organization, neighborhood preserving and compactness against state-of-art methods. The quantitative measures are computed as follows:

Euclidean distance (E): let p_i^o and p_i be the original and final position of the center of each box. The Euclidean distance metric is defined as

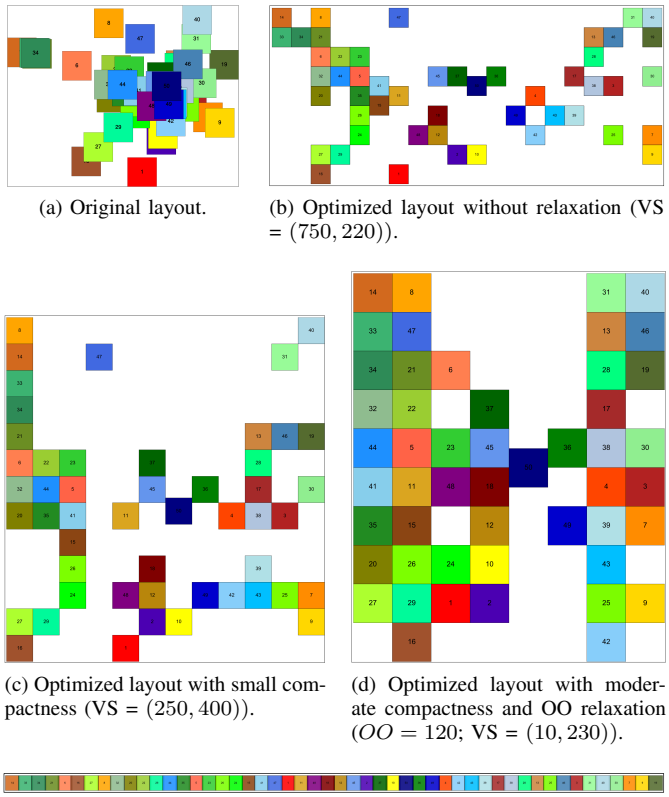
$$E = \frac{1}{n} \sum_i d(p_i^o, p_i), \tag{11}$$



(a) Input layout.

(b) Optimized layout after applying MIOLA without relaxation constants.

Fig. 5. Result of our method when applying on a content layout taken from Youtube's search engine.



(a) Original layout.

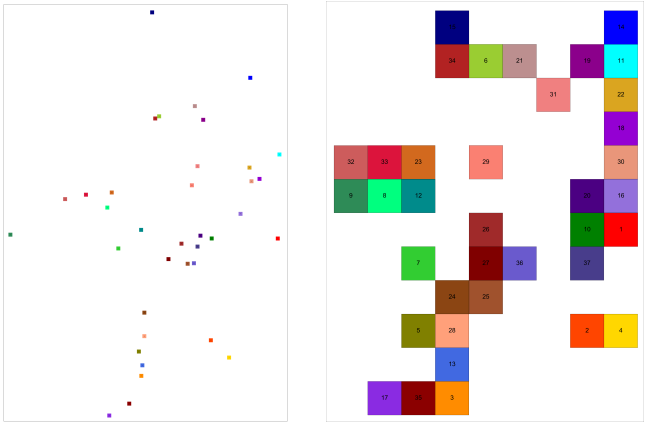
(b) Optimized layout without relaxation (VS = $(750, 220)$).

(c) Optimized layout with small compactness (VS = $(250, 400)$).

(d) Optimized layout with moderate compactness and OO relaxation ($OO = 120$; VS = $(10, 230)$).

(e) Optimized layout with hard compactness and NOL relaxation ($NOL = 10$; VS = $(3800, 5)$).

Fig. 6. Flexibility and adaptability of the proposed formulation.



(a) Layout with scattered boxes.

(b) Result after layout adjustment.

Fig. 7. Optimizing sparse layouts with MIOLA.

where n is the number of boxes and d is the Euclidean distance. This metric gives an estimative of how much the boxes displace during the overlap removal process. Less movement is preferred, since the original configuration is better preserved.

Layout similarity (σ): quantifies how much neighborhoods are affected by the overlap removal mechanisms. The idea is to measure changes in the length of Delaunay edges, computed with respect to original layout. In mathematical terms, let l^o_{ij} and l_{ij} denote the lengths of the Delaunay edges before and after overlap removal. The layout similarity is then given by

$$\sigma = \frac{\sqrt{\left(\sum (r_{ij} - \overline{r})^2\right)/m}}{\overline{r}}, \quad \overline{r} = \frac{\sum r_{ij}}{m}, \tag{12}$$

where $r_{ij} = l_{ij}/l^o_{ij}$ and m is the number of edges in the Delaunay triangulation. The closer to zero the better.

Orthogonal ordering (O): quantifies how much the constraints (3) are violated. In numerical terms, this metric measures the number of changes in the relative order of the boxes, in the horizontal as well as vertical directions. More precisely, positions are sorted in increasing order for x and y coordinates, and the number of inversions in the lists gives us the ordering measure. In mathematical terms:

$$O = \sum_{s \in \{x,y\}} \sum_{i<j} inv_s^{(t)}(i,j), \tag{13}$$

$$inv_x^{(t)}(i,j) = \begin{cases} 1, & \text{if } (x_i^{(t)} - x_j^{(t)}).(x_i^{(t-1)} - x_j^{(t-1)}) < 0 \\ 0, & \text{otherwise} \end{cases} \tag{14}$$

where i, j are indexes of the sorted lists and $inv_y^{(t)}(i,j)$ is defined analogously to $inv_x^{(t)}(i,j)$. Small values are better.

Size increase (S): given the convex hulls C^o and C of the original and final layouts, the size increase is measured as:

$$S = \frac{area(C)}{area(C^o)}, \tag{15}$$

determining the relative changes in size as well as the compactness of the representation. Values closer to 1 are better.

Neighborhood preservation (K): computes the average percentage of the k-nearest neighbors of each box that are preserved in the final layout. The higher the curve the better.

Measures above allow us to quantitatively compare our technique against the following well-established overlap removal methods: VPSC [8], PRISM [10], Voronoi-based [11] and RWordle-C [15].

Fig. 8 shows the resulting layout after applying MIOLA and the four techniques we compare against on five different data sets containing textual information gotten from web search engines. The first three data sets, namely, "Scientific Visualization" (50 instances), "Sibgrapi" (32 instances) and "Purple"(100 instances), were generated using fixed-size square boxes while the last two datasets - "Wave" (64 instances) and "Batman" (50 instances) - were created taking rectangles with sizes given by the rank of the document (the rank is provided by the search engine). Larger rectangles have higher rank. One can note that the Voronoi-based method results in unstructured overlap-free layouts when compared to the other methods. PRISM preserves relevant parts of the initial shape, but with the disadvantage of excessive use of space, hampering data exploration. The layouts produced by VPSC are quite structured, helping readability, however, it is prone to stretch the layout vertically (see the "Wave" and "Batman" results). RWordle-C produced satisfactory layouts preserving, in some cases, clusters visually identified by the user (see the "Wave" result), but still leaving some expressive empty spaces when compared with MIOLA. Moreover, the orthogonal ordering is considerably affected by RWordle-C. In contrast, MIOLA produces very well-organized and compact layouts while still respecting the orthogonal order in a satisfactory way.

Fig. 9 summarizes the quantitative results of the metrics described above. Notice that MIOLA produces more compact layouts measured by Euclidian distance and Size increase, being fairly stable as to other metrics.

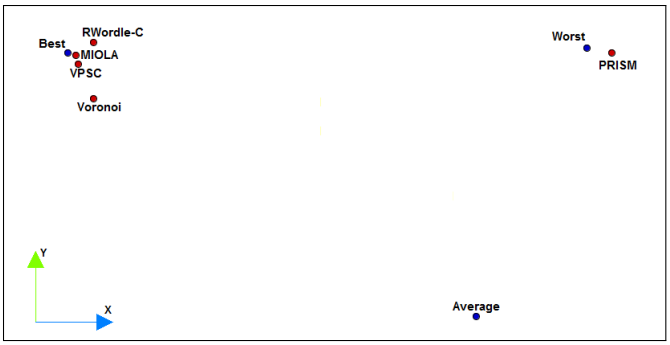


Fig. 10. Comparison of overlap removal techniques respecting the five metrics simultaneously. Red points represent the evaluated techniques while the blue points show the worst, average and best points artificially produced.

Fig. 10 plots a map that considers all metrics simultaneously. Each overlap removal technique has been represented as a five-dimensional vector $(E_a, \sigma_a, O_a, S_a, k_a)$, where $E_a, \sigma_a, O_a, S_a, k_a$ are the average values of the metrics (15)-(11) and k-nearest neighbors computed for each technique, over all data sets from Fig.8. The blue points denoted by "Best", "Average", and "Worst" in 10 were created artificially as five-dimensional vectors containing the better, average and worst values taken from the results of all methods. We use the multidimensional projection method LAMP [21] to project the five dimensional data to the visual space. The technique mapped closest to "Best" is the one that, on average, has shown the best performance, relative to all metrics. Clearly, one can observe that MIOLA is the technique closer to the best result.

IV. Discussion and Limitations

Comparisons presented in Section III clearly show the effectiveness of the proposed optimization method, surpassing, in terms of accuracy, existing methods. As shown in Fig. 6, MIOLA turns out to be flexible and capable of producing a variety of layouts, a characteristic not present in any other state-of-art method. The snippet video application depicted in Fig. 5 indicates that MIOLA can be used to remove overlaps while keeping similar entities close to each other, thus preserving the similarity relations introduced during the multidimensional projection.

Despite good results and solid mathematical foundation, there are two aspects to be observed when using MIOLA. First, the visualization window as well as the number of boxes to be displayed simultaneously can affect the readability of the visualization. Easy to read layouts depend on the screen size. Finding out a good trade off between clean/pleasant layouts and the relaxation parameters is another issue that deserves further investigation.

V. Conclusion and Future Work

In this work we proposed a new technique for solving the box overlap removal problem. The evaluation we provided shows that the propose method, called MIOLA, outperforms

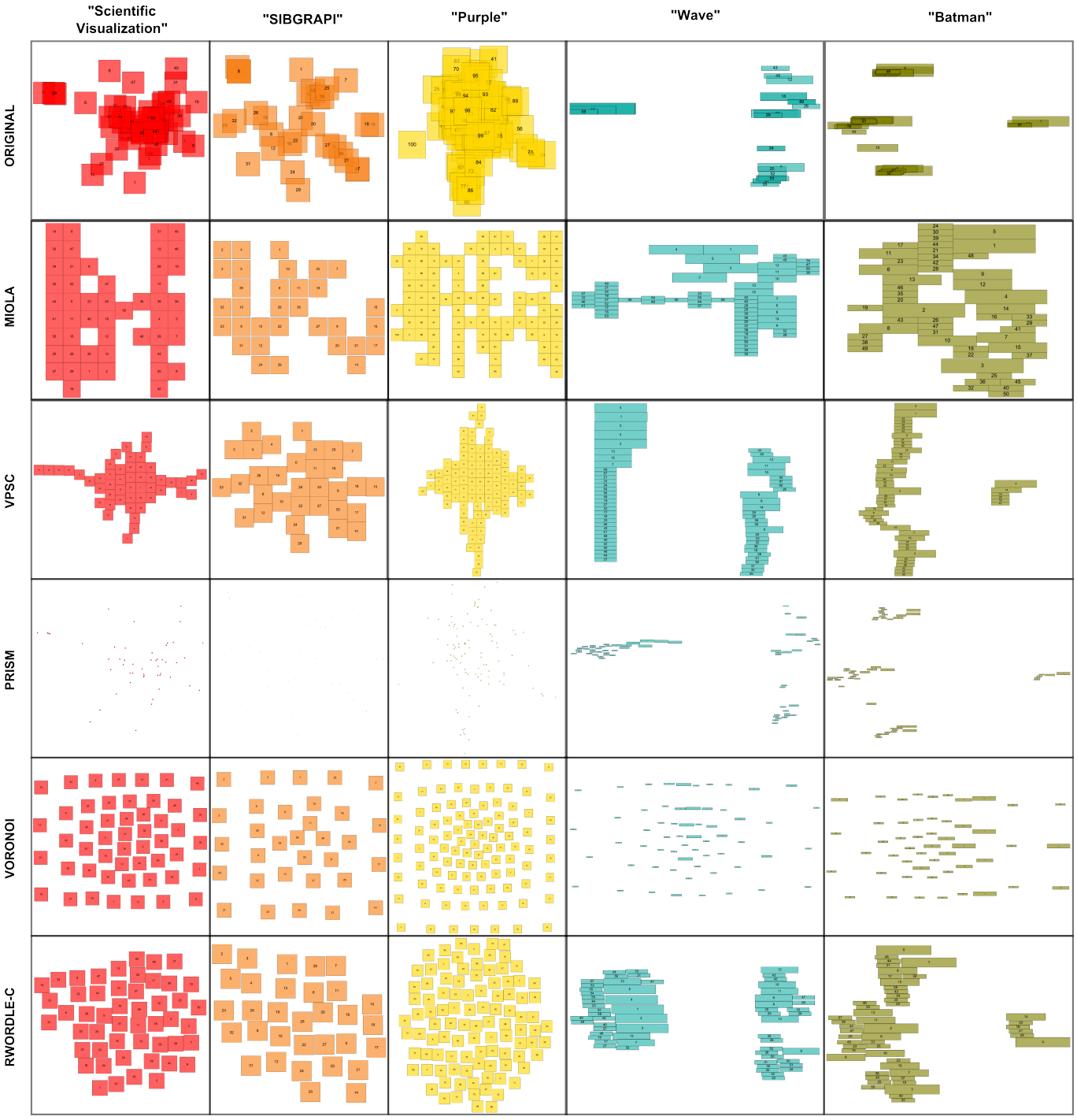
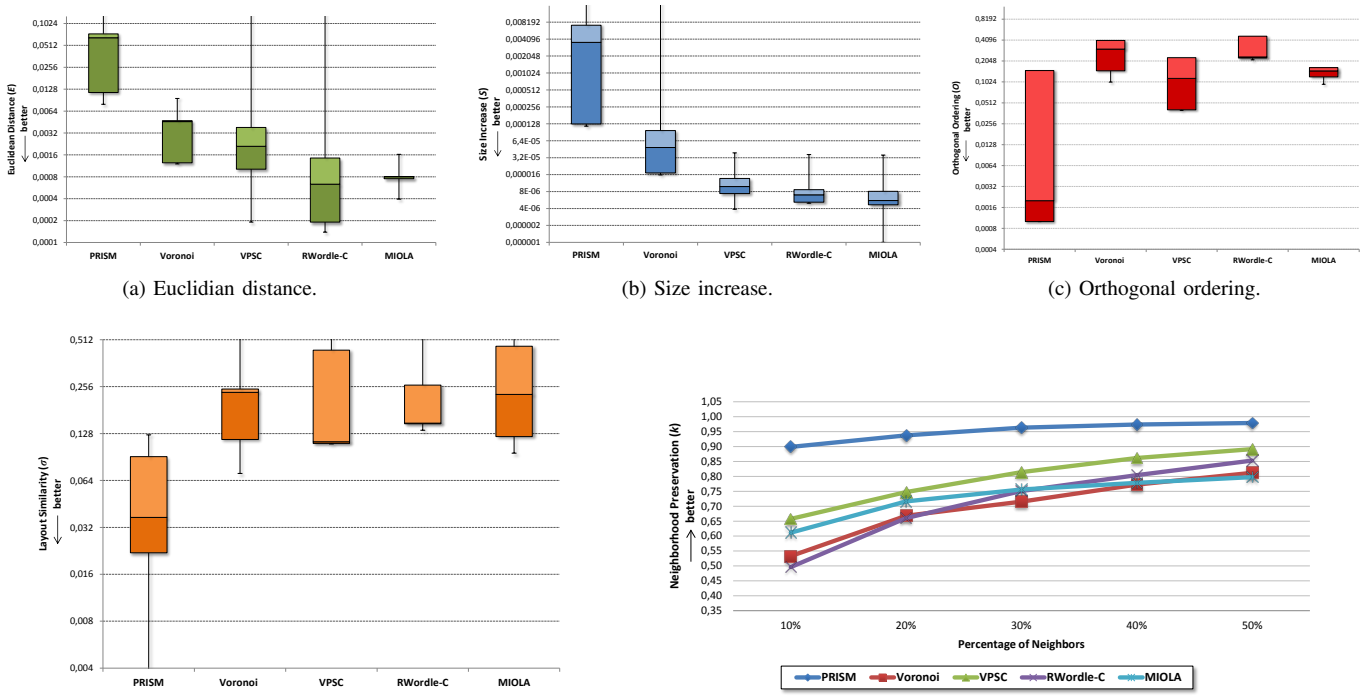


Fig. 8. Layouts produced by our approach (MIOLA), VPSC, PRISM, Voronoi-based, and RWordle-C for five distinct data sets.



(a) Euclidian distance. (b) Size increase. (c) Orthogonal ordering.

(d) Layout Similarity. (e) k-neighborhood preservation.

Fig. 9. Comparing our approach, VPSC, PRISM, Voronoi-based, and RWordle-C considering the metrics E (a), σ (b), O (c), S (d), and k-nearest neighborhoods (e) (in log scale after normalization) regarding to Fig.8.

existing methods in terms of producing well structured layout. MIOLA turns out to be effective in practical application and quite flexible to generate different layouts by just tuning parameters, rending it a very attractive overlap removal approach.

We are currently extending MIOLA to work with geometric entities other than boxes. in addition to combine the relaxation parameters with strategies for clustering snippets.

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for their useful and constructive comments. This research has been supported by Computer Science Research Center of San Pablo Catholic University, FAPESP-Brazil, NSF (grants IIS-0808718 and CCF-0915661), CAPES and CNPq-Brazil.

REFERENCES

[1] M. A. Boschetti and A. Mingozzi, "The two-dimensional finite bin packing problem. part i: New lower bounds for the oriented case," *4OR*, vol. 1, no. 1, pp. 27–42, 2003.

[2] R. Andrade and E. Birgin, "Symmetry-breaking constraints for packing identical orthogonal rectangles within polyhedra," *Optimization Letters*, vol. 7, pp. 375–405, 2011.

[3] K. Misue, P. Eades, W. Lai, and K. Sugiyama, "Layout adjustment and the mental map," *Journal of visual languages and computing*, vol. 6, no. 2, pp. 183–210, 1995.

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

[5] A. S. Spritzer and C. M. Dal Sasso Freitas, "Design and evaluation of magnetviza graph visualization tool," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 18, no. 5, pp. 822–835, 2012.

[6] J.-H. Chuang, C.-C. Lin, and H.-C. Yen, "Drawing graphs with nonuniform nodes using potential fields," in *Graph Drawing*. Springer, 2004.

[7] D. Harel and Y. Koren, "Drawing graphs with non-uniform vertices," in *Proceedings of the Working Conference on Advanced Visual Interfaces*. ACM, 2002, pp. 157–166.

[8] T. Dwyer, K. Marriott, and P. J. Stuckey, "Fast node overlap removal," in *Graph Drawing*. Springer, 2006, pp. 153–164.

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

[10] E. R. Gansner and Y. Hu, "Graph drawing," 2009, ch. Efficient Node Overlap Removal Using a Proximity Stress Model, pp. 206–217.

[11] Q. Du, V. Faber, and M. Gunzburger, "Centroidal voronoi tessellations: Applications and algorithms," *SIAM Review*, vol. 41, no. 4, p. 637, 1999.

[12] F. B. Viegas, M. Wattenberg, and J. Feinberg, "Participatory visualization with wordle," *IEEE TVCG*, pp. 1137 –1144, 2009.

[13] K. Koh, B. Lee, B. Kim, and J. Seo, "Maniwordle: Providing flexible control over wordle," *IEEE Transactions on Visualization and Computer Graphics*, vol. 16, no. 6, pp. 1190 –1197, 2010.

[14] R. Pinho, M. Oliveira, and A. Andrade Lopes, "An incremental space to visualize dynamic data sets," *Multimedia Tools and Applications*, vol. 50, no. 3, pp. 533–562, 2010.

[15] H. Strobelt, M. Spicker, A. Stoffel, D. Keim, and O. Deussen, "Rolled-out wordles: A heuristic method for overlap removal of 2d data representatives," in *Computer Graphics Forum*, vol. 31, 2012, pp. 1135–1144.

[16] R. Lazimy, "Mixed-integer quadratic programming," *Maht. Programming*, vol. 22, pp. 332–349, 1982.

[17] J. Lee and S. Leyffer, *Mixed Integer Nonlinear Programming (The IMA Volumes in Mathematics and its Applications)*. Springer, 2012.

[18] W. Yin, "Gurobi mex: A matlab interface for gurobi (2011)," in *convexoptimization.com/wikimization/index.php/gurobi_mex*.

[19] F. Paulovich, L. Nonato, R. Minghim, and H. Levkowitz, "Least square projection: A fast high-precision multidimensional projection technique and its application to document mapping," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 14, no. 3, pp. 564–575, 2008.

[20] G. Salton, A. Wong, and C. S. Yang, "A vector space model for automatic indexing," *Commun. ACM*, vol. 18, no. 11, pp. 613–620, 1975.

[21] P. Joia, D. Coimbra, J. Cuminato, F. Paulovich, and L. Nonato, "Local affine multidimensional projection," *IEEE TVCG*, pp. 2563–2571, 2011.