EdWordle: Consistency-preserving Word Cloud Editing

Yunhai Wang, Xiaowei Chu, Chen Bao, Lifeng Zhu
Oliver Deussen, Baoquan Chen and Michael Sedlmair

Fig. 1. Result of a case study with a professional writer who sought to visualize a BBC news feed: the left image shows the input Wordle layout; the right image shows the layout that was created using EdWordle. The writer ordered related words into semantically meaningful groups, one group per story. Each group was organized spatially together and color-coded, creating a layout that the user referred to as a *"storytelling cloud"*.

Abstract—We present EdWordle, a method for consistently editing word clouds. At its heart, EdWordle allows users to move and edit words while preserving the neighborhoods of other words. To do so, we combine a constrained rigid body simulation with a neighborhood-aware local Wordle algorithm to update the cloud and to create very compact layouts. The consistent and stable behavior of EdWordle enables users to create new forms of word clouds such as storytelling clouds in which the position of words is carefully edited. We compare our approach with state-of-the-art methods and show that we can improve user performance, user satisfaction, as well as the layout itself.

Index Terms—Wordle, consistency, text visualization

1 Introduction

Wordle [17] is a popular visualization tool that converts a piece of text into a word cloud, in which each word is sized according to its number of occurrences. Despite several concerns that have been raised by the visualization community [1], Wordle has gained great popularity and has been adopted by many non-expert users since its introduction in 2008. One of the major reasons for this success is the aesthetic and participatory output that Wordle creates. In other words, Wordle is mainly used as an authoring tool to produce visually pleasing word clouds, which can be customized for appearance and shared with others. It is very rarely used as a data analysis tool with the need of accurately representing the underlying data [37].

To fine-tune the output so that it meets their aesthetic goals, users often wish to further manipulate Wordles by adding, deleting, or modifying words [17]. The original method, however, only allows users to change the attributes of the whole word cloud. To fill this gap, Koh et al. [26] introduced ManiWordle, a technique that enables users to directly manipulate individual words with different operations such as selection, movement or rotation. Jo et al. [24] extended this work by developing WordlePlus for pen- and touch-enabled tablets. This version of the method provides full control over a wordle that includes resizing, adding, and deleting elements.

The manipulation of size, orientation, or position of words inevitably involves a re-organization of the whole layout. For example, when a user deletes a word, there will be empty space that needs to be filled. If a word is moved to a new position, others need to be moved away. Current state-of-the-art re-organization strategies [24, 26] relocate words that no longer fit anymore by simply moving them to empty spaces. While this approach warrants a compact overall layout, it also results in substantial global reordering with words being moved to completely different positions. This approach hence contradicts the idea of *consistency*, a core principle in many design-related areas [33]. A proper, consistency-preserving editing approach will be even more important in semantically-ordered word clouds [3, 41]. Here, the neighborhood of words is meaningful, so that words like "BBC" and "News", or "presidential" and "candidate" appear spatially close together, as shown in Figure 1. In such a case, inconsistent and unpredictable layout changes during editing would likely result in poor user experience.

To fill this gap, we designed EdWordle, a context-aware interaction technique that seeks to preserve the local and global order of words in a word cloud. Our technique is based on the coherent combination of two components: a customized rigid body dynamics simulation and a neighborhood-aware re-layout algorithm. Similar to morphable word clouds [8], each word is viewed as a rigid body with a mass and the

- *Y. Wang, B. Chen, X. Chu, and C. Bao are with Shandong University. Email: {wang.yh, baoquan}@sdu.edu.cn, {cuxiaoxie, baochen95 }@gmail.com.*
- *L. Zhu is with Southeast University. E-mail: lfzhulf@gmail.com.*
- *O. Deussen is with Konstanz University and VCC SIAT, China. E-mail: oliver.deussen@uni-konstanz.de.*
- *M. Sedlmair is with University of Vienna, Austria. E-mail: michael.sedlmair@univie.ac.at.*
- *Y. Wang and X. Chu are joint first authors.*

Manuscript received xx xxx. 201x; accepted xx xxx. 201x. Date of Publication xx xxx. 201x; date of current version xx xxx. 201x. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org. Digital Object Identifier: xx.xxxx/TVCG.201x.xxxxxxx

dynamics simulation arranges words by applying forces. Representing each body solely as a box might be too loose for producing compact word clouds. Hence, we use a two-level box representation for each word, where each letter is viewed as an individual box and the common part of all letters as the other box. Based on this representation, we construct two kinds of forces: neighboring forces between words pull them to stay as close as possible, and a central force pulls each word towards the center of the word cloud. Such a dynamic system is able to generate compact and non-overlapping word clouds, but it cannot guarantee that all empty spaces will be filled, especially at the boundary. To alleviate this problem, we additionally propose a neighborhood-aware local Wordle algorithm, which moves words that are far from the center to fill nearby empty spaces.

Although EdWordle's primary function is to edit word clouds, the method can also be used to improve any existing word cloud. Thus, we compare it to state-of-the-art word cloud creation algorithms [3, 41] by taking their outputs as our inputs and then investigating how much our approach improves these layouts under experimental settings as proposed by Barth et al. [4]. A quantitative analysis shows that our approach is i) consistently better in preserving word neighborhoods, ii) successfully avoids global changes to the layout, and iii) has similar or even better results in avoiding empty spaces. In addition, we compare EdWordle to ManiWordle to investigate its usability for Wordle editing by conducting a laboratory study. The results show that people were not only faster and more accurate with EdWordle, but also felt they achieved better results. Finally, we invited some designers and writers to investigate the value of EdWordle as an authoring tool. The results demonstrate the advantages of our approach over Wordle and ManiWordle in creating storytelling visualizations, for example laying out headlines in a semantically meaningful way as shown in Figure 1.

2 Related Work

We review previous work related to word cloud visualization and work related to authoring tools for visual representations more generally.

2.1 Word Cloud Visualization

A word cloud, also known as a tag cloud, is a visual representation of text data that has been used on the web since 1997 [36]. A word cloud encodes the frequency of words of a given text into font size and color [31], and spatially arranges the words on the canvas. Over the years, a number of different spatial arrangements have been proposed. Standard word cloud visualizations use a rectangular line-by-line layout, where the words may be sorted alphabetically or by their importance. To produce more compact and aesthetic visualizations, a large family of alternative layout methods have been proposed [20, 25, 32, 35, 37]. Among them, the most well-known algorithm is Wordle [37], which is the focus of the present study. Wordle uses a greedy approach to produce compact layouts, where words are placed in different orientations, not just in a single direction. To improve the orthogonal ordering of Wordle, Strobelt et al. [35] suggest to combine scan-line based techniques [29] with the greedy layout strategy of Wordle. To enable the user to easily create visually appealing word clouds, Tagxedo [28] and WordArt [40] both allow the user to put word clouds into specific shapes. However, such improvements still do not capture the relationship between words, let alone the temporal coherence of time-varying text data. Therefore, a variety of semantic and temporal word cloud generation methods and editing tools have been proposed in recent years.

Semantic Word Clouds While Wordle and its variants produce aesthetic visualizations, their layout algorithms do not incorporate the neighborhood relationships between words and thus they do not place semantically related words close to each other. This was mentioned by Hearst [22] as one of the critical limitations of traditional word cloud visualizations. To overcome this limitation, Wu et al. [41] proposed an approach that places similar words close to each other. To do so, they first compute a distance matrix between words and then use multidimensional scaling to place words onto a 2D canvas, while removing blank spaces through a carving scheme. Paulovich et al. [30] extended this idea to document collections by using multidimensional projection techniques to compute neighborhood relationship and arrange the words accordingly.

For pre-specified neighborhood relationships, Barth et al. [3] show that creating a semantic word cloud that strictly respects the relationship between words is a NP-hard problem. They consequently present approximation algorithms and conduct a comparison [4] between Wordle and the carving method [41]. Their findings show that semantic word clouds were not as compact as Wordle. Recently, Buchin et al. [6] pushed this line further, proposing geo word clouds that respect not only the neighborhood relationships but also the relative (geo-spatial) position associated with each word. All these algorithms, however, do not allow the user to re-position words, which means that semantics cannot be adjusted once the layout is generated. Our neighborhood-preserving editing approach is specifically important for semantic word clouds, and plays a crucial role for keeping the neighborhood between words. Globally re-positing words upon edits, as pursued by current state-of-the-art editing approaches (see below), would naturally interfere with the goal of keeping a semantic order. With our approach users are able to order words in a way that even tells a story within the word cloud, a feature that has not yet been possible.

Temporal Word Clouds Given a time-varying set of words, temporal word clouds attempt to visualize temporal trends while preserving temporal coherence. Collins et al. [10] introduce Parallel TagClouds that combine parallel coordinates and traditional word clouds, where the words of each document are distributed along each coordinate axis. Lee et al. [27] present Sparkclouds that visualize trends between multiple word clouds by integrating sparklines into a word cloud. Both methods perform well in the visualization of trends, with Sparkclouds being the better one in terms of scalability. Cui et al. [11] combine a trend chart and multiple word clouds together to illustrate temporal changes of the underlying data. By combining multidimensional scaling and force-directed layout, this method can create semantic and stable word clouds over time. Recently, Chi et al. [8] propose morphable word clouds where a sequence of spatial shapes is specified as a boundary for a set of time-varying word clouds. By using rigid body dynamics, they arrange words within the given shape sequence so that temporal changes are encoded by both the shapes and the content of the word clouds. In this paper, we also use rigid body dynamics to rearrange words after editing. Representing each word as a rigid body, however, might result in large empty spaces between words. To address this issue, we propose a two-level rigid body representation for each word and combine rigid body dynamics with a local version of the Wordle placement algorithm.

Word Cloud Editing Almost none of the existing word cloud visualizations allows users to edit typographical properties of individual words. While this might be plausible for pure data analysis settings, Viegas et al. found that Wordles are mainly used as an authoring tool for participatory visualizations [37]. In this scenario, users often want to manually customize and edit the visual output. To address this need, Koh et al. proposed Maniwordle [26], which allows the user to move, rotate and remove a word with a mouse. Wordleplus [24] by Jo et al., extended Maniwordle to multi-touch settings and enriched it with natural interactions. Since user interaction might result in empty space between words, ManiWordle re-runs the Wordle layout algorithm for the un-edited words and Wordleplus repeatedly uses the words at the boundary of Wordle to fill empty space within the cloud. Both approaches result in global and unpredictable layout changes, as can be seen in Figure 7 on page 6. Our approach, EdWordle, overcomes these limitations by striving to preserve a consistent neighbor-relationship before and after editing. It can also be used for other applications such as semantic word cloud generation and temporal cloud generation.

2.2 Visualization Construction and Authoring Tools

For the last few decades, there has been considerable effort to create easy-to-use visualization construction tools. Grammel et al. [21] provide a survey of various types of such visualization construction tools. Among them, visual builders provide large flexibility by allowing the

Fig. 2. Overview of our method: (a) given a Wordle, (b) we use our customized rigid body dynamics to move words close to each other; (c) if a word is moved, the forces update the words accordingly; (d) empty spaces are removed by using a local version of the Wordle algorithm.

user to move and resize visual elements in order to create custom visualizations. Two typical examples are Dust & Magnet [34] and flexible linked axes [9]. EdWordle also belongs to this type of interfaces that allows the user to freely control elements.

Today, designers can create visualizations with a large variety of tools [23, 38], and further edit visualization with drawing tools like Adobe Illustrator, if needed. Very recently, Bigelow et al. [5] pointed out that current tools leave a gap between producing and later editing of visualizations, which can considerably hinder designers in their creative process. Towards bridging this gap, they proposed a model which allows for a much easier iteration between visualization creation and editing. Along similar lines, Fulda et al. [19] proposed an authoring tool that allows designers and journalists to create and edit timeline visualizations for temporal story telling. Our work was inspired by this recent trend towards authoring tools in visualization, and eases the transition from an automatically generated word cloud to a manually adapted visual output.

3 EdWordle

Figure 2 illustrates the overall approach behind EdWordle. The process starts by simply loading some text that can be copied and pasted into the tool. Alternatively, we also can start from an existing word cloud as input. For illustrative purposes we assume the case that an initial word cloud has already been created, as illustrated in Figure 2(a). After the wordle is loaded, we first apply a *customized rigid body dynamics* approach, which helps us to make the layout more compact while preserving the neighborhood relationships. This process results in a compact representation of the input.

The user can then freely edit the cloud until the desired result is obtained. In our example, the user would like to move the word "dedicate" closer to the word "nation". For each interaction, EdWordle's main goal is to allow for predictable changes, as well as to produce a compact and aesthetic result after each editing step. More specifically, we seek to preserve neighborhoods of words to allow for consistency when changing the layout (instead of words that might jump around unpredictably). To do so, after each step the rigid body dynamics step is automatically invoked again (Figure 2(b)-(c)). At any time, the result can be further improved by performing a *Re-Wordle*, a local Wordle layout process, in which empty spaces at the boundary are filled by nearby words. All steps are based on our *two-level box representation* for the words, which allows us to create compact representations without words squeezing in between characters of other larger words.

Using this approach, EdWordle gives the user full creative control including the ability to drag, rotate, add, delete, or resize a word or multiple words at the same time. Moreover, it allows previewing of the intermediate outcome actions by continuously updating the layout while words are moved around by the user [33]. In the following, we describe each of the core components of EdWordle in more detail. We first briefly introduce the general rigid body dynamics, then explain our *two-level box representation*, as well as our *customized external forces* approach (Fig. 2(a)-(c)), and finally explain the *local Wordle layout* algorithm that we designed (Fig. 2(d)).

3.1 Rigid Body Dynamics Based Layout

By representing each word as a rigid object, rigid body dynamics allow us to avoid word overlapping by enforcing non-penetration constraints. We provide a brief review of rigid body dynamics, for more details please refer to Witkin [39]. Rigid body dynamics systems deal with the motion of bodies that are subject to external forces. Rigid bodies cannot penetrate each other, so their motion is simulated using two major components: unconstrained and constrained dynamics. The former updates position and velocity in response to (outer) forces, while the latter detects collisions between bodies and creates corresponding responses, please refer to Appendix A for unconstrained rigid body dynamics. In general, however, the state of a rigid body is described by the vector $\mathbf{Y}(t) = (x(t), \mathbf{R}(t), \mathbf{P}(t), \mathbf{L}(t))$, which includes its position x, orientation $\mathbf{R}$, linear momentum $\mathbf{P}$ and angular momentum $\mathbf{L}$.

A constraint is a restriction to the position or motion of a rigid body [16]. To satisfy a non-penetration constraint for instance, an appropriate collision detection and response between the rigid bodies is required. By representing each body as a convex hull with corresponding bounding box hierarchy, collisions can be detected by any efficient algorithm, such as the separating axis method [15]. After a collision is detected, the non-penetration constraint is enforced by impulse-based dynamics, which solve the imposed constraints using linear equations [7].

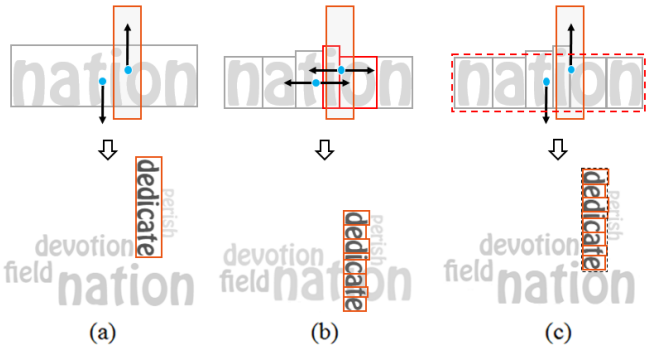

Fig. 3. Comparison of three different body representations for a word with collision responses shown by black arrows (upper row) and resulting layouts (lower row). (a) the word-level box representation introduces large empty spaces between words; (b) the letter-level boxes result in a compact layout but introduces overlapping words; (c) The two-level box representation combines the advantages of the representations in (a,b).

3.1.1 Adaptive Two-level Word Representation

Previous approaches [3, 8, 11, 35, 41] typically represent words with a single bounding box, which is simple but results in non-compact layouts (see Figure 3(a)). An alternative way is to represent each word as a combination of multiple letter-level bounding boxes, which allows compact layouts to be generated but might result in word overlaps. Figure 3(b) illustrates this issue: the collision response from the letter 'i' and 'o' of the word "dedicate" cancel out each other so that "dedicate" and "nation" are getting stuck due to their letter-level boxes. Since the collision detection is done for all boxes attached to the body, the

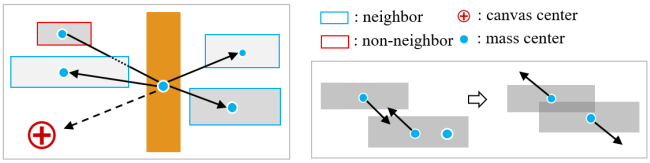

Fig. 4. Illustration of our rigid body dynamics: (a) Neighboring forces (black solid lines) and central forces (black dashed lines) act on the central box (neighbours are shown with blue outline); (b) Motion before and after collision between two words.

letter-level box representation also incurs a substantial computational overhead.

Thus, we propose to combine the two representations in an adaptive way. Directly using the bounding box of the entire word will make the letter-level boxes useless. As a consequence, we compute the bounding box of the common part of all letters to form our word-level box, shown as a red dashed box in Figure 3(c). The width of our word-level box is the width of the bounding box of the whole word and the height is the minimal height of all letters. To reduce the computational cost, we use this two-level box representation only for words whose relative sizes compared to the largest word are larger than a fixed threshold. In this way we do not introduce large empty spaces between words, since gaps between letters in small words are typically too small for placing other words. In our experiments, the threshold was set to 0.5. This means all words with at minimum half the font size of the largest word will be subject to our two-level box representation. For all other words we use normal word-level boxes. Our collision detection is based on this two-level box representation. In this study we primarily work on compact wordle layouts, but users are allowed to offset the proposed two-level bounding box for creating more whitespace if they want to make words more outstanding.

3.1.2 Customized External Forces

To produce a compact layout while preserving the neighborhood, we apply two external forces to the objects: neighboring forces and central forces. While the former pushes neighboring words close to each other, the latter drags all words to the canvas center. For computing neighboring forces, we first determine the neighborhood relation among the words in the given word cloud and then apply the external forces accordingly to move them (see Figure 4(a)). If two words collide, they will bounce off each other, see Figure 4(b). Note that forces are enforced to the center of the mass of each rigid body represented by our adaptive two-level boxes.

Neighborhood Search: The body center of each word is connected to the centers of all other words. If the line segment connecting two words does not intersect a third word, these two words are considered to be neighbors. As shown in Figure 4(a), the word with the red border is thus not taken as a neighbor of the selected word.

Neighboring Forces: Suppose the selected word has n_f neighbor words, then the exerted neighboring force is:

$$F_i^{neigh} = \sum_{j=1}^{n_f} m_i \times m_j / r_{i,j}^2 \tag{1}$$

where $r_{i,j}$ is the Euclidean distance between the body centers of the selected word i and the n neighbor words j. The mass m is a given weighting factor for each word. Since this force is inverse proportional to the distance (we divide by $r_{i,j}^2$), it will pull neighboring words more closely together. Figure 5(a,b) shows an example where the neighboring force pulls "struggled" and "testing" as well as "resolve" and "liberty" together after deleting the word "conceived".

Central Forces: Since the magnitude of neighboring forces decreases with their distance, neighboring words with large distances cannot be pulled together. This could result in considerable gaps. For example, there is a large gap between the words "struggled" and "score" in Figure 5(b). To address this issue, we introduce a central attraction source. That is, we place a virtual body with a large mass M at the center of the canvas to attract all words toward it using the following formula:

$$F_i^{cent} = m_i \times M \times r_{i,c}^2 \tag{2}$$

where M is a unit mass, and $r_{i,c}$ the distance between word i and the canvas center. Central forces are proportional to the distance between the words and the virtual object in the center (we multiply by $r_{i,c}^2$). Thus, they will especially attract those words that are farthest from the center. Figure 5(c) shows an example where the central force pulls the words "resolve" and "fought" more closely to the center.

Joining the Forces We now can simply join the forces by computing a weighted sum for each word body i, at each iteration step t:

$$F_i(t) = F_i^{neigh}(t) + \alpha F_i^{cent}(t) \tag{3}$$

where α is the weight. Large central forces might destroy the neighborhood relationship, for example, during movement from Figure 5(a) to (c), the neighboring words "vain" and "resolve" are not neighbors anymore. We heuristically found $\alpha = 0.1$ to be a good compromise between the forces.

Damping Strategy: Directly applying this force F_i to the words will, however, cause the system to halt in a non-equilibrium state since (i) the forces move words so that collisions occur and (ii) the collision forces pull them back into the opposite direction. This results in an unstable, oscillating system. The words would jitter and shake around at each iteration step. To address this issue, we apply an attenuation function $g(t)$ to the force. In combination with the the force F_i for each word body, we receive the following final equation at each iteration t:

$$F_i^{damp}(t) = F_i(t) \times g(t) \tag{4}$$

Since neighboring forces become much larger when words are close and the system is getting compact, the attenuation function should become smaller as the number of iteration increases. Thus, we set $g(t) = \beta/(t+1)$ with β an attenuation constant.

Nonetheless, only applying an attenuation function to the forces is not enough to avoid jittering. Words keep moving as long as their velocity is not zero. Thus, we also decay the velocity with a damping coefficient λ: $v_i(t) = v_i(t) \times \lambda$, where we set λ to 0.8. This damping strategy reduces the movement of the rigid bodies until they reach their resting positions.

Driven by these two strategies, any change to the position of a word has an effect on the forces being exerted on it. Such effect in turn changes the moving velocity of the word and subsequently its updated position, leading to an iteratively updating framework. Since different words have different velocities, we specify a maximum number of iterations to stop the simulation. In our experiment, we found that 80 iterations are enough for achieving convergence for all of our word clouds, where the running time is typically around 0.6s with our not yet optimized implementation.

Algorithm 1 Local Wordle Layout

```
identify the boundary word list
for each word w in the sorted boundary word list do
    compute a initial position for w
    search k candidate positions for w
    pick the position that best preserves the neighborhood
end for
```

3.2 Local Wordle Layout

When the rigid body simulation stops, most word body motions are blocked by neighboring words so that words are packed compactly. Since the spatial distribution according to the neighbourhood relation

Fig. 5. The resulting word clouds after (a) deleting the word "conceived" in the input word cloud. (b) The results generated by applying the neighboring forces; (c) central forces in combination with neighboring forces, and (d) central forces with a weight $\alpha = 0.1$.

might not be uniform, the arrangement of words might be biased towards some direction, which results in gaps among the words in the obtained layout. As shown in Figure 2(c), such gaps often appear at the boundary of a cloud. To fill them, we propose to re-arrange the related words using the original Wordle layout algorithm. However, this layout algorithm will, according to its greedy strategy, potentially destroy the neighborhood relationship among words. This would produce inconsistencies with the rigid body system. In other words, the re-layout has to take the neighborhood relationship into account. Thus, we propose a local version of the Wordle layout algorithm as outlined in Algorithm 1. It has two key components: i) identification of boundary words; and ii) searching a new position. Note that this algorithm might destroy the original neighborhood relation a little to improve the compactness and thus we leave it as an option for the user.

Identification of boundary words (line 1). We compute the width and height of an axis-aligned bounding box b of the current word cloud and then construct a circle centered at the center of the layout and with a radius $\beta * \min(width_b, height_b)$. All words that lie outside of this circle need to be re-placed, indicated as red boxes in Figure 6(a). After identifying such words, we sort them in terms of their font sizes.

Picking the best position (line 3-5). Before searching for a new position using the spiral scheme of Worlde (cf. [17]), we define its initial position as the midpoint of the line segment that connects the mass centers of the current word and the center of the entire word cloud. Then we find k candidate positions along the spiral and select the one that preserves the largest number of neighborhood relations on its new position. If more than one position preserves the same number of neighborhoods, we pick the one that is found first, because it is closer to the word cloud center. Figure 6(b) illustrates the procedure for the orange box highlighted in Figure 6(a).

In our experiment, β and k are set to 0.8 and 20, which works well for most of our data. In this way, our local Wordle layout algorithm not only fills gaps but also preserves the original neighboring relationship as much as possible. Figures 2(c,d) show a comparison of the word clouds without and with adaptation by the local Wordle layout.

3.3 Interactions

The above approach allows us to provide a set of new or refined types of interactions that enables users to create visually pleasing word clouds:

Neighborhood-preserving editing. After importing a word cloud, EdWordle allows the user to move, rotate, resize, add and delete words while preserving the neighborhood relationship. Although ManiWordle [26] and WordlePlus [24] both also support most of these operations, neither of them respects the original neighborhood relationships

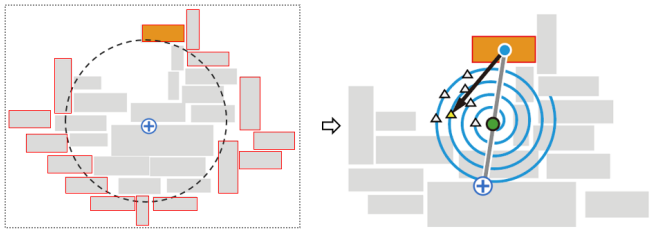

Fig. 6. Illustration of the local Wordle layout method, the point shown with the cross is the center of the entire word cloud. (a) The words outside of the circle are taken as boundary words and are shown with red borders; (b) k candidates (shown as triangles) are found for the word highlighted in orange, where the green circle indicates the starting position and the yellow triangle is the position finally selected.

of words. Figure 7 compares the re-layouted results generated by ManiWordle and EdWordle after moving and rotating the word "dedicate". In ManiWordle (b), this editing step evokes global repositioning of the words marked in orange, and results in a non-compact layout. Our approach overcomes these limitations, retaining the word neighborhoods and compactness of the layout.

Note that resizing words does interfere with the validity of word clouds, but only if used as a pure data/text analysis tool. As ManiWordle [26] and WordlePlus [24] , we take Wordle more as a communication and artistic tool, where users start with an initial layout and then want to adjust it to their needs. In this case, a precise representation of word counts through word size is not of primary concern, and users even request the ability to resize words as shown in the FAQ and discussion forums of Wordle [17].

Multi-word editing. To simultaneously manipulate multiple words, EdWordle allows the user to select multiple words by pressing the Ctrl button or with the help of rectangle selection. Once multiple words are selected, the user can move, rotate, delete, re-color, re-font and re-wordle them. With traditional editing approaches, such multi-word interactions would be impracticable as manipulations would result in substantial global changes and would introduce large gaps between words.

Re-Wordle. As mentioned in Section 3.2, our local Wordle layout algorithm can re-arrange the words so that gaps at the borders can be closed. We refer to this operation as Re-Wordle. While the standard Re-Wordle process has the default parameters (β and k), we also allow the user to play around with the parameters, or only use it on specific subsets of words. To select specific words, EdWordle allows the user to interactively adjust the circle center, circle radius and de-select some boundary words if they want to keep their positions.

Other interactions. To make EdWordle a full and usable creative editing environment, we also provide other common interactions, such as undo, redo, save and load, or edit data. We furthermore provide an additional multi-touch version of EdWordle, allowing users to select, drag, rotate, and resize words with multi-touch interactions similar to WordlePlus [24]. Such an approach is interesting specifically for collaborative Wordle creation where concurrent interaction of multiple users is now possible because the user does not get interrupted by words jumping around from another user editing the word cloud simultaneously.

3.4 Implementation and Tool

EdWordle is written in JavaScript and runs in the web-based environment, which is is available as an online tool[1]. The Hammer.js library (http://hammerjs.github.io/) is used for touch interaction. The accompanying video shows examples of the above described interactions.

[1] http://www.edwordle.net/

Fig. 7. Comparison between Maniwordle [26] and our method for movement and rotation. (a) Input word cloud where the word "dedicate" is moved down and rotated about a small angle; (b,c) Results generated by Maniwordle and our method, respectively.

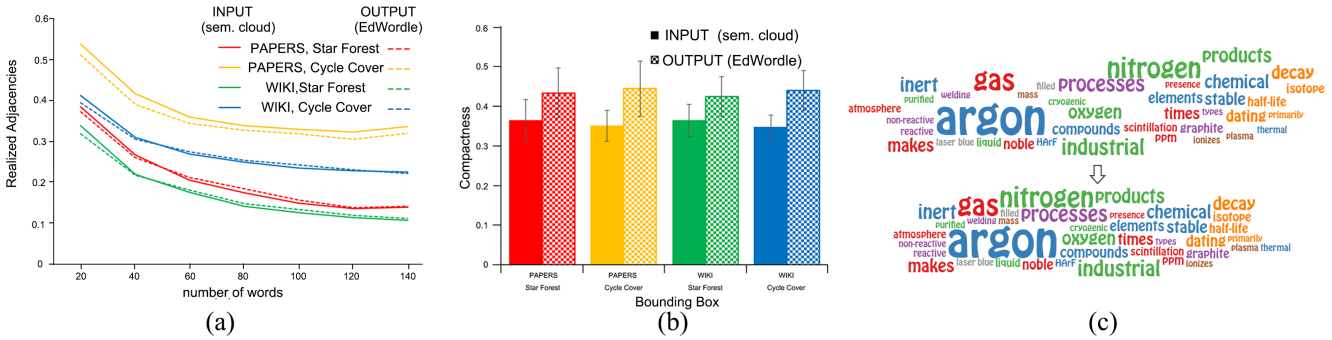

Fig. 8. (a) *Realized adjacencies* for various word clouds; higher is better. The dashed and continuous lines match up well, indicating that EdWordle is able to preserve neighborhoods well. (b) Mean and standard deviation of *compactness* for various word clouds; higher is better. EdWordle (dotted) produces substantially more compact results. (c) An example for refining a semantic word cloud (top) with EdWordle (bottom).

4 Evaluation

We evaluate our approach in three different ways. First, we illustrate how EdWordle allows us to refine and improve existing word clouds. Second, we present a small-scale quantitative lab study with 16 participants. And third, we discuss the results of a qualitative case study with 10 designers and writers. The purpose of these studies is to compare our approach to current state-of-the-art approaches for generating and editing word clouds. Further details on experimental designs and results can be found in the supplemental materials.

4.1 Quantitative Comparison

In this section, we validate that EdWordle can improve the quality of existing word clouds by enhancing their compactness, while preserving most of the neighborhood relations. To do so, we use Barth et al.'s semantic word clouds layout [4] as input for Edwordle for further processing. The resulting layouts produced by Edwordle are expected to be more compact, while at the same time we expect them to largely retain word neighborhoods. Such a quality refinement is specifically interesting for semantic word clouds, because under this condition the semantic neighborhood of words does not get destroyed by the quality optimization process. We illustrate the quality improvement by comparing the initial layout with the processed layout. For most word clouds with 50 words, the improvement procedure can be done in less than 0.5s.

4.1.1 Study Design

Metrics. According to Barth et al. [4], there are six common metrics for evaluating word cloud layouts: realized adjacencies, distortion, compactness, uniform area utilization, aspect ratio, and running time. Here, we focus on the *realized adjacencies*, and *compactness*, which are especially relevant for our goals. The metric of *realized adjacencies* is defined as the sum of edge weights of all pairs of boxes that share a common boundary. This metric mainly reflects how well the semantic relatedness between pairs of words is "realized" in the word cloud layout. *Compactness* indicates how much area is used to render the actual words in relation to the entire area the cloud covers, that is, *used area*/*all area*. Compactness therefore indicates how efficiently the drawing area is used. To define *all area*, we simply use the bounding box of the whole word cloud, while we set the *used area* as the sum of bounding box areas for all words.

Existing Algorithms. We use two different word cloud layout algorithms: Star Forest, and Cycle Cover, which have been found to outperform the other methods in realized adjacencies, and are competitive with respect to the other metrics [4].

Data. We use two datasets taken from Brath et al. [4]. For each algorithm, we tested both datasets.

1. *WIKI*, consisting of 112 plain-text articles from the English Wikipedia, each has 200 distinct words or more; and
2. *PAPERS*, consisting of 56 scientific papers from two experimental algorithms conferences (SEA and ALENEX, 2011-2012).

4.1.2 Results

Fig. 8 shows the results of our experiments. Fig. 8(a) demonstrates that the neighborhood relations in all word clouds created by EdWordle are well-preserved. The metrics of *realized adjacencies* for the output layouts created by EdWordle (dotted lines) are almost equal to the input lines by the original layouts (continuous lines). Only for the *PAPERS, Cycle Cover* case, we see that the EdWordle output is slightly worse than the original input. The loss, however, is very minor, and we deem it acceptable.

Fig. 8(b) shows the results for compactness. After applying EdWordle to the initial layout, the compactness of these word clouds improved substantially. This suggests that EdWordle indeed is able to further improve given semantic word clouds, by making them more compact while at the same time preserving their word neighborhoods. Fig. 8(c) illustrates this effect with an example.

4.2 Lab study

In this study, our goal was to evaluate how well different editing approaches perform in tasks related to keeping word neighborhoods. For instance, a user might like to move a word next to another word or to create a semantic layout where the position of words is meaningful [3,4,30,41]. He might even like to create a storytelling cloud, which shows complete headlines as a sequence of words inside the cloud.

4.2.1 Experimental factors

We had two experimental factors (independent variables): editing approach and tasks.

Editing tools. We compare our EdWordle approach with ManiWordle, the current state-of-the-art approach for editing Wordles. To ensure a fair comparison, we only test the basic functionality of both approaches, that is, moving and rotating objects. For EdWordle, we initially considered three variants to be tested: (1) without preview, (2) with preview, and (3) a version that "pushes" words away. We thought that a version with preview will be easier to use as it allows users to see how the overall-layout will change, even though necessarily adds response time to the system. The variant without preview is fast but not as user-friendly since users cannot foresee the impact of their actions. The third version was inspired by how moving physical objects would push away other objects while moving, such as stones that lie on a table. To determine which of the three versions to use in our study, we first conducted a pilot study with 6 participants. We found that users consistently preferred version (2) with preview, deeming the slight increase of response time as acceptable. We also considered including WordlePlus into the study, but refrained from it as WordlePlus uses very similar basic functions as implemented in ManiWordle.

Tasks. We carefully explored different test tasks and evaluated them based on how they can be measured, and how realistic they are. This process led us to the following two tasks for our study:

Task 1—Pairwise placement: This task is meant to test simple and common pairwise interactions of word placements. It encompasses three subtasks: put word x on top of word y; put x on left of y; put x on top of y and rotate them to a certain angle. For different users, x and y were randomly sampled. We visually illustrated the task in an abstract spatial way as shown in Fig. 9(a). Our goal with this approach was to allow participants to easily infer spatial target positions, while keeping it abstract enough to avoid a focus on superficial details.

Task 2—Semantic misplacement: The second task relates to the ability to form groups of related words, for instance, as common in Semantic Wordles [3,41]. Instead of testing the entire ordering process, we opted for simply correcting four randomly sampled misplaced words into their correct groupings, as shown in Fig. 9(b). We also opted to let users group words by color, instead of the actual semantic meaning of words. On the one hand, color can be perceived much quicker than semantic meaning, which avoids potential learning effects between the two conditions. On the other hand, semantic interpretation would leave too much room for subjective interpretation, making it hard to measure.

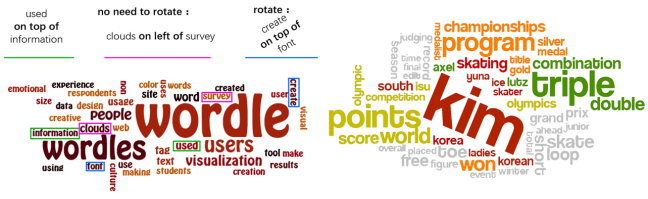

Fig. 9. (a) Example picture for Task 1. (b) Example picture for Task 2.

4.2.2 Hypotheses

Based on EdWordle's ability to preserve local and global neighborhoods, we had the following hypotheses before conducting the study:

- *H1*: For both tasks, users will be more efficient with EdWordle than with ManiWordle.
- *H2*: For Task 1 (pairwise placement), EdWordle will result in more accurate results, that is, with fewer errors.
- *H3*: The layouts created with EdWordle will be more compact.
- *H4*: Subjective ratings will be slightly better for EdWordle than for ManiWordle, as ManiWordle can lead to unpredictable word changes.

4.2.3 Experiment Design

We used a *within-subjects* design: each participant conducted both tasks with both editing tools (*independent variables*). The following *dependent variables* were tested:

- *Efficiency*: We used three efficiency measures: the *time* (seconds) a user takes to perform a task, the *number of clicks* needed, and the *distance* of the mouse movement.
- *Accuracy*: We counted the number of misplacements in each resulting Wordle layout, that is, words that were not correctly placed according to the subtask/task description.
- *Compactness*: As in the experiment above, we also computed the compactness c of the resulting layouts by $c = \textit{used area}/\textit{all area}$, using the bounding box approach to specify *all area*. Since we do not have word box information of ManiWordle, we counted the number of pixels with non-background color to define *used area*.
- *Subjective ratings*: We used the ManiWordle questionnaire [26] to gather subjective ratings of users on six questions, ranging from strong disagreement to strong agreement (7-point Likert scale). We attempted to avoid biasing effects by the interviewer in our study.

Data. We used two different datasets in our study:

- *D1*: the paper "Participatory Visualization with Wordle" by Viegas et al. [37]
- *D2*: the Wikipedia entry about the famous South Korean skater Yu-Na Kim.

To limit the complexity of the study design, D1 was used only in task 1, and D2 in task 2.

Participants. We recruited 16 participants (8 male, 8 female), most of whom are Computer Science students from the local university. The age ranged from 20 to 29 years (median age 22). All participants reported normal or corrected-to-normal vision, and had no color vision deficiencies. None of them had prior experience with word cloud generating tools.

Procedure. We defined the following procedure for the study:

1. introduction video, familiarization with both tools (2 min. each), and explanation of tasks by the interviewer;
2. Task 1, with both editing tools;
3. Task 2, with both editing tools; and
4. questionnaire and short interview.

The orders within (2)-(4) were counterbalanced to avoid systematic bias through potential learning effects. During (4), the participants were allowed to play around with the respective editing tools again, and were shown the wordles they created with this tool before. The study was run on a quad-core PC with a 23" LCD widescreen, using a mouse as input and a 1920 $\times$ 1080 px display as output. The system was equipped with a AMD FirePro W2100 GPU with 2048 MB of memory.

4.2.4 Results

We followed recent recommendations of statistical analysis practices and used an estimation-based approach with effect sizes and confidence intervals [2]. This approach overcomes several biases and limitations of classical null hypothesis testing with p-values (NHST) [12, 14]. Cumming and Finch offer guidance on how to interpret statistical tests using confidence intervals (CI) [13]. Specifically, if there is no overlap between CI error bars we can assume a statically significant result with a p-level of $p \approx .01$ (if the bars touch each other) or smaller (if there is a gap between them) [13].

The results of our quantitative measures are summarized in Figure 10. For *time*, *clicks*, and *distance*, lower values are better. For *compactness*,

higher values are better. The results are consistent with the trends we initially predicted with our hypotheses. All three efficiency measures, (a) *time*, (b) *clicks*, and (c) *distance* show a clear and strong effect of EdWordle being more efficient than ManiWordle.

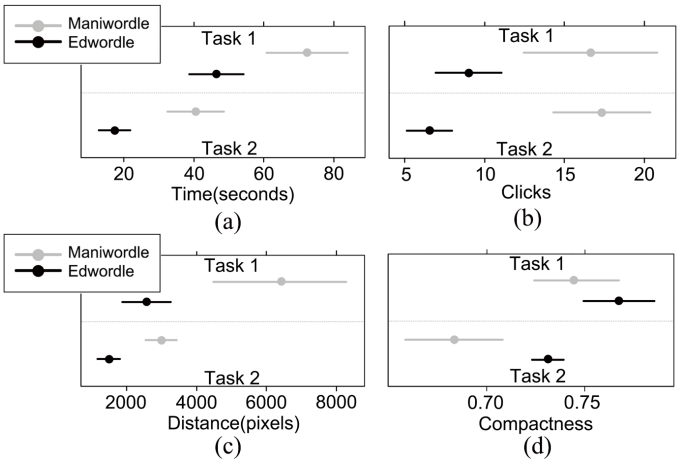

Fig. 10. Mean values and errors as 95% CIs of (a) time, (b) clicks, (c) distance, and (d) compactness. For (a)-(c), lower values are better; for (d) higher values are better

In terms of accuracy, we counted the *errors* that were made under each condition. For Task 1, an error is defined as an incorrect neighborhood between an indicated pair of words (e.g., because a word had jumped unpredictably around). For ManiWordle we counted 10 errors across all users, for EdWordle only 2. Screenshots of all errors are shown in the supplemental material. For Task 2, there were no errors under either condition. We expected this result, as Task 2 is based on pre-attentive color sorting that allows users to easily spot and correct for errors.

The results of the *compactness* scores, which we computed for each resulting word cloud layout, are shown in Figure 10(d). For Task 1, EdWordle is only slightly better than ManiWordle. For Task 2, the effect is much more pronounced and EdWordle created more compact results. When inspecting the resulting clouds of the participants, we noticed some EdWordle layouts in which words would purposely "stick out" from the bulk of other words, such as the word "culture" in Fig. 9(a). While this behavior is not negative per se, the bounding box-based compactness measure is very sensitive to it and explains the small effect on compactness values of EdWordle in Task 1. As a more sensitive alternative, Barth et al. [4] proposed basing the compactness measure on the convex hull of a word cloud. This hull-based approach would indeed avoid this issue, but was impractical in our case because we only have images available for ManiWordle layouts, and not the word-box information.

Finally, we looked at the *subjective ratings* of the ManiWordle questionnaire. The results are shown in Table 1, and they indicate that EdWordle was consistently better rated than ManiWordle by our participants. In particular, questions related to usability (Q2 & Q3) and to the generated results (Q6) show clear evidence that EdWordle subjectively outperformed ManiWordle.

Table 1. Subjective Responses to Six Questions(Average Ratings)

Questions	ManiWordle	EdWordle
Q1: It was easy to learn this visualization.	6	6.41
Q2: It was easy to use this visualization.	4.94	6.18
Q3: I liked to use this visualization.	4.56	6.09
Q4: It was fun to use this visualization.	5.12	5.74
Q5: I felt creative while using this visualization.	5.12	5.53
Q6: Overall, I am satisfied with the result layout.	4.69	6.08

4.3 Case Studies with Designers

In our last evaluation we wanted to determine the value of EdWordle in a more qualitative way. We thus conducted a set of case studies with designers and writers, two of our primary target audiences.

4.3.1 Study Design

Overall, we invited 10 participants to our study. All participants worked in some sort of creative profession, such as graphic design or journalism, and had different interests, such as international political affairs or technology. We asked them to use EdWordle to create a layout according to their own taste, desire, and needs. To do that, they were allowed to pick an article of their choice. The only prerequisite was that they had to be familiar with the selected article, in order to resemble a data presentation rather than a data exploration scenario. In contrast to the quantitative lab study, we offered the participants the full spectrum of EdWordle's functionality. The design process of each participant was closely observed and recorded. After each session, we collected their results and comments on EdWordle, as well as information about their background. Overall, every case study took 1–2 hours.

4.3.2 Results

We were happy to observe that all 10 participants engaged in a very creative design process, and came up with interesting and inspiring layouts. Fig. 11 as well as Fig. 1 show 6 of the resulting Wordle layouts. All of the created word clouds involved some sort of semantic layout, which is inherently fostered by our approach as it allows designers to manly organize words according to their semantics.

Fig. 1 shows an example done by a writer who is strongly interested in international affairs. Her EdWordle is based on a news feed from BBC, which included different threads of daily news. Deciding to reveal these different threads in her layout, she ordered the words that belong to a specific story spatially together, and also color-coded them with the same colors. With this layout it becomes very clear that the article is made up of five news pieces: (1) America supporting the Israeli-Palestinian solution; (2) President Trump's administration did fine-tune and publish the travel ban; (3) Marine Le Pen, a presidential candidate in France, was under investigation; (4) 70 people were killed in Pakistan because of a suicide attack, and (5) 50 people in Baghdad died of a car bomb by the IS.

Fig. 11(a) shows the result of another participant with a background in journalism, who sought to visualize an article about "deleting some of your apps will make you happier". As the article was heavily referring to iPhones and Apple's App Store, the journalist decided to design the overall shape of the Wordle as an Apple logo. While creating such a shape-oriented Wordle would also have been possible with previous approaches [8], the user went beyond that in that he carefully designed the leaf of the apple as a catchy headline "delete and you will thank me later". There are also other semantic groups in the layout, such as "2 million apps", and the two numbers that refer to a central message in the article: deleting "54%" of the apps freed up "24%" of the disk space.

Fig. 11(b)-(e) show the results of another four case studies. We see that the designers of these layouts also engaged in producing creative and semantically-inspired layouts, often with some sort of "story" behind it. Participants (b)-(d) made strong usage of different orientations, and sought to order words into meaningful groups. In (b), for instance, the participant visualized a speech of Obama by semantically ordering its four parts into separate corners, while having recurring themes in the center. Result (e) was the closest to a classical Wordle layout, with the difference that the participant sough to put the central message into the center: "All men are created equal". More details about the case studies, as well as the other studies can be found in the supplemental materials.

5 Conclusions and Future Work

In this work, we presented EdWordle, a novel approach for editing word clouds. EdWordle's main benefit is that it allows a neighborhood-preserving editing process, which keeps words at predictable and close locations during and after the editing process. In a set of quantitive

Fig. 11. Results of our case studies, visualizing (a) an article about how much time apps eat up, (b) a speech by Obama, (c) a transcript of an interview on a psychological topic, (d) an article about solar eclipse, and (e) a speech by Martin Luther King.

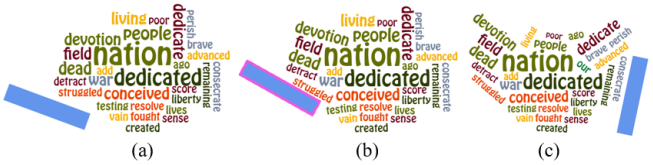

Fig. 12. Pushing bars can push words into specific directions. Once some words collide with the pushing bar, they will be moved along the pushing direction: (a) cloud before moving pushing bar; (b) intermediate result; (c) after using the pushing bar from two sides to customize the word cloud shape.

experiments with and without users, we found that this approach outperformed other state-of-the-art editing approaches. In our qualitative case study with 10 designers and writers, we furthermore found that, with its neighborhood-preserving character, EdWordle fosters new ways of creating and editing Wordles. Designers and writers would, for instance, automatically engage in creating some sort of semantically meaningful Wordles, which represented some aspects of the underlying stories, or simply *storytelling clouds*. This finding is in line with the increasing amount of recent research on semantic word clouds. So far, however, it has been very hard to edit such clouds without destroying their semantic layout. EdWordle fills this gap. We also believe that our neighborhood-preserving approach opens various doors for new forms of interactions. For instance, we experimented with *pushing bars* that allow users to interactively customize desired shapes by pushing the entire layout around. Figure 12 as well as the video illustrate this idea.

We believe that there are many other ideas that could be realized based on our consistency-preserving approach. Extending the pushing bar metaphor to multi-touch, one could, for instance, think about a solution that allows users to literally "shape" a word cloud with the hands. Another idea is to allow the user to draw a lasso or a shape, and pull at the ends of this "rope" to "tighten" the word cloud – an idea that is similar to shape word clouds [28, 40]. Or one could simply combine their shape constraints [8] with our approach of neighborhood-preserving editing. Such an approach would even better support editable shape Wordles as the one described in our "apple" case study (Fig. 11(a)).

A more general avenue for future work is to develop more sophisticated authoring tools for the generation of storytelling word clouds. Eight of our 10 designers/writers mentioned that they would have liked creative support through a more direct link to the underlying text. Our system still requires a considerable amount of user interaction to generate the final storytelling clouds as shown in Fig. 1. In the future, we thus plan to combine our consistency-preserving editing approach with advanced text analysis algorithms [18] to inform better initial layouts so that user interaction can be further reduced. Furthermore, our system is designed to preserve local word neighborhoods while reducing whitespace. The resulting word clouds might be too compact in some cases so that some distinct words might be hard to identify (e.g., when all words are horizontally aligned). For a word cloud consisting of multiple topics, separating topics by whitespace might also make storytelling clouds more understandable. It would be interesting to learn more about how compactness and whitespace affect the usability and readability of word clouds. These ideas indicate the potential of more sophisticated authoring tools that connect word cloud generation, editing, and text analysis tools.

Acknowledgments

The authors would like to thank Haifeng Zhang for making the evaluation. This work is supported by the grants of NSFC-Guangdong Joint Fund (U1501255), NSFC (61379091, 91630204), the National Key Research & Development Plan of China (2016YFB1001404), Shandong Provincial Natural Science Foundation (11150005201602), NSF of Jiangsu Province (BK20150634), National Foreign 1000 Talent Plan (WQ201344000169), Leading Talents of Guangdong Program (00201509), the Fundamental Research Funds of Shandong University, and the FFG project 845898 (VALID).

Appendix A: Unconstraint Dynamics

Given the center of mass $x(t)$ as the origin of the body space and its orientation represented by rotation matrix $\mathbf{R}(t)$ in world space at a time t, a position r_b in body space can be mapped to a position $r(t)$ in world space by:

$$r(t) = x(t) + \mathbf{R}(t)r_b. \tag{5}$$

To describe how position and orientation change over time, we define a linear velocity $v(t)$ and an angular velocity $\omega(t)$. The quantity $v(t) = \dot{x}(t) = \frac{d}{dt}x(t)$ gives the velocity of translation, while $\omega(t)$ specifies the rotation speed and the axis about which the body is rotating. By using the relation $\dot{\mathbf{R}}(t) = \omega(t) \times \mathbf{R}(t)$, the velocity of the body at the position $r(t)$ is:

$$\dot{r}(t) = v(t) + \omega(t) \times \mathbf{R}(t)r_b. \tag{6}$$

When external forces act on the body with the mass m, the position and velocity of the body will be changed. Suppose, an external force $\mathbf{F}(t)$ is given, then the torque $\tau(t)$ acting on the body is:

$$\tau(t) = (r(t) - x(t)) \times \mathbf{F}(t), \tag{7}$$

where $\tau(t)$ depends on the location $r(t)$ of the body relative to its center of mass $x(t)$.

Since momentums are conserved in nature, it is recommend to describe the state of a moving body with a linear momentum $\mathbf{P}(t)$ and an angular momentum $\mathbf{L}(t)$. They are computed from the linear and angular velocity, respectively:

$$\mathbf{P}(t) = mv(t), \quad \mathbf{L}(t) = \mathbf{I}(t)\omega(t) \tag{8}$$

where $\mathbf{I}(t)$ is an inertia tensor describing how the mass in a body is distributed relative to the center of mass. The change in linear momentum can be described as

$$\dot{\mathbf{P}}(t) = m\frac{d}{dt}v(t) = \mathbf{F}(t), \tag{9}$$

where the acceleration is $\alpha = \mathbf{F}(t)/m$. By analogy, the derivative of the angular momentum is formulated as $\dot{\mathbf{L}}(t) = \tau(t)$.

REFERENCES

[1] E. Alexander, C.-C. Chang, M. Shimabukuro, S. Franconeri, C. Collins, and M. Gleicher. The biasing effect of word length in font size encodings. In *Poster Compendium of the IEEE Conference on Information Visualization*, 2016.

[2] American Psychological Association. *Publication manual of the American psychological association (6th edition)*. American Psychological Association Washington, 2010.

[3] L. Barth, S. I. Fabrikant, S. G. Kobourov, A. Lubiw, M. Nöllenburg, Y. Okamoto, S. Pupyrev, C. Squarcella, T. Ueckerdt, and A. Wolff. Semantic word cloud representations: Hardness and approximation algorithms. In *Latin American Symposium on Theoretical Informatics*, pages 514–525, 2014.

[4] L. Barth, S. G. Kobourov, and S. Pupyrev. Experimental comparison of semantic word clouds. In *International Symposium on Experimental Algorithms*, pages 247–258, 2014.

[5] A. Bigelow, S. Drucker, D. Fisher, and M. Meyer. Iterating between tools to create and edit visualizations. *IEEE Trans. Vis. & Comp. Graphics*, 23(1):481–490, 2017.

[6] K. Buchin, D. Creemers, A. Lazzarotto, B. Speckmann, and J. Wulms. Geo word clouds. In *Proceedings of the IEEE Pacific Visualization Symposium*, pages 144–151, 2016.

[7] E. Catto. Iterative dynamics with temporal coherence. In *Proc. Game Developer Conference*, 2005.

[8] M.-T. Chi, S.-S. Lin, S.-Y. Chen, C.-H. Lin, and T.-Y. Lee. Morphable word clouds for time-varying text data visualization. *IEEE Trans. Vis. & Comp. Graphics*, 21(12):1415–1426, 2015.

[9] J. H. Claessen and J. J. Van Wijk. Flexible linked axes for multivariate data visualization. *IEEE Trans. Vis. & Comp. Graphics*, 17(12):2310–2316, 2011.

[10] C. Collins, F. B. Viegas, and M. Wattenberg. Parallel tag clouds to explore and analyze faceted text corpora. In *Proceedings of the IEEE Symposium on Visual Analytics Science and Technology*, pages 91–98, 2009.

[11] W. Cui, Y. Wu, S. Liu, F. Wei, M. X. Zhou, and H. Qu. Context preserving dynamic word cloud visualization. In *Proceedings of the IEEE Pacific Visualization Symposium*, pages 121–128, 2010.

[12] G. Cumming. *Understanding the new statistics: Effect sizes, confidence intervals, and meta-analysis*. Routledge, 2013.

[13] G. Cumming and S. Finch. Inference by eye: confidence intervals and how to read pictures of data. *American Psychologist*, 60(2):170, 2005.

[14] P. Dragicevic. Fair statistical communication in hci. In *Modern Statistical Methods for HCI*, pages 291–330. Springer, 2016.

[15] C. Ericson. *Real-time collision detection*. CRC Press, 2004.

[16] R. Featherstone. *Rigid body dynamics algorithms*. Springer, 2014.

[17] J. Feinberg. Wordle-beautiful word clouds. `http://www.wordle.net/`. 2009.

[18] R. Feldman and J. Sanger. *The text mining handbook: advanced approaches in analyzing unstructured data*. Cambridge university press, 2007.

[19] J. Fulda, M. Brehmer, and T. Munzner. Timelinecurator: Interactive authoring of visual timelines from unstructured text. *IEEE Trans. Vis. & Comp. Graphics*, 22(1):300–309, 2016.

[20] P. Gambette and J. Véronis. Visualising a text with a tree cloud. In *Proceedings of the International Federation of Classification Societies Conference*, pages 561–569, 2010.

[21] L. Grammel, C. Bennett, M. Tory, and M.-A. Storey. A survey of visualization construction user interfaces. In *EuroVis-Short Papers*, pages 19–23, 2013.

[22] M. A. Hearst. Whats up with tag clouds? *Visual Business Intelligence Newsletter*, 2008.

[23] S. Huron, Y. Jansen, and S. Carpendale. Constructing visual representations: Investigating the use of tangible tokens. *IEEE Trans. Vis. & Comp. Graphics*, 20(12):2102–2111, 2014.

[24] J. Jo, B. Lee, and J. Seo. Wordleplus: Expanding wordle's use through natural interaction and animation. *IEEE Computer Graphics and Applications*, 35(6):20–28, 2015.

[25] O. Kaser and D. Lemire. Tag-cloud drawing: Algorithms for cloud visualization. In *Proceedings of the Workshop onTagging and Metadata for Social Information Organization*, 2007.

[26] K. Koh, B. Lee, B. Kim, and J. Seo. Maniwordle: Providing flexible control over wordle. *IEEE Trans. Vis. & Comp. Graphics*, 16(6):1190–1197, 2010.

[27] B. Lee, N. H. Riche, A. K. Karlson, and S. Carpendale. Sparkclouds: Visualizing trends in tag clouds. *IEEE Trans. Vis. & Comp. Graphics*, 16(6):1182–1189, 2010.

[28] H. Leung. Tagxedo website. `http://www.tagxedo.com/`. last visited 06/2017.

[29] K. Misue, P. Eades, W. Lai, and K. Sugiyama. Layout adjustment and the mental map. *Journal of Visual Languages & Computing*, 6(2):183–210, 1995.

[30] F. V. Paulovich, F. Toledo, G. P. Telles, R. Minghim, and L. G. Nonato. Semantic wordification of document collections. *Computer Graphics Forum*, 31(3pt3):1145–1153, 2012.

[31] A. W. Rivadeneira, D. M. Gruen, M. J. Muller, and D. R. Millen. Getting our head in the clouds: toward evaluation studies of tagclouds. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, pages 995–998, 2007.

[32] C. Seifert, B. Kump, W. Kienreich, G. Granitzer, and M. Granitzer. On the beauty and usability of tag clouds. In *Proc. Int. Conf. on Information Visualisation*, pages 17–25, 2008.

[33] B. Shneiderman. *Designing the user interface: strategies for effective human-computer interaction*. Pearson Education India, 2010.

[34] J. Soo Yi, R. Melton, J. Stasko, and J. A. Jacko. Dust & magnet: multivariate information visualization using a magnet metaphor. *Information Visualization*, 4(4):239–256, 2005.

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

[36] F. B. Viégas and M. Wattenberg. Timelines tag clouds and the case for vernacular visualization. *Interactions*, 15(4):49–52, 2008.

[37] F. B. Viegas, M. Wattenberg, and J. Feinberg. Participatory visualization with wordle. *IEEE Trans. Vis. & Comp. Graphics*, 15(6), 2009.

[38] J. Walny, S. Huron, and S. Carpendale. An exploratory study of data sketching for visual representation. *Computer Graphics Forum*, 34(3):231–240, 2015.

[39] A. Witkin. Physically based modeling: principles and practice constrained dynamics. *Computer Graphics*, 1997.

[40] WordArt.com. Wordart website. `http://www.wordart.com/`. last visited 06/2017.

[41] Y. Wu, T. Provan, F. Wei, S. Liu, and K.-L. Ma. Semantic-preserving word clouds by seam carving. *Computer Graphics Forum*, 30(3):741–750, 2011.