A Novel Physics-Based Interaction Model for Free Document Layout

Ricardo Piccoli, Rodrigo Chamun, Nicole Cogo,
João Oliveira and Isabel Manssour

Centro de Pesquisa em Computação Aplicada
Faculdade de Informática — PUCRS
Porto Alegre — Brazil
{oliveira,manssour}@inf.pucrs.br

ABSTRACT

Marketing flyers, greeting cards, brochures and similar materials are expensive to produce, since these documents need to be personalized and typically require a graphic design professional to create. Either authoring tools are too complex to use or a predefined set of fixed templates is available, which can be restrictive and difficult to produce the desired results. Thus, simpler design tools are a compelling need for small businesses and consumers. This paper describes an interactive authoring method for creating free-form documents based on a force-directed approach, traditionally applied for graph layout problems. This is used for automatically distributing and manipulating images, text and decorative elements on a page, according to forces modeled after physical laws. Such approach can be used for enabling easy authoring of personalized brochures, photo albums, calendars, greeting cards and other free-form documents. A prototype has been developed for evaluation purposes, and is briefly described in this paper. Evaluation results are presented as well, showing that users enjoy the experience of designing a page by interacting with it, and that end results can be satisfactory.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous; I.3.6 [**Computer Graphics**]: Methodology and Techniques—*Interaction Techniques*; I.7.2 [**Document and Text Processing**]: Document Preparation—*Format and notation, Photocomposition/typesetting*

General Terms

Algorithms, Design, Experimentation

Keywords

automatic document layout, authoring, personalized documents, force-directed methods, physics-based simulation

1. INTRODUCTION

Authoring solutions for personalized material such as greeting cards, brochures, photo albums, and marketing collaterals are usually based on static templates produced by graphic designers, filled with content provided by a user. Good examples of such systems are Snapfish[1] and Shutterfly[2]. However, those templates can be expensive to create and maintain and are difficult to adapt to different purposes, media sizes or customers. Recent years have seen considerable advances on automatic document generation and adjustment, but providing non-specialists with a pleasant interface for producing simple documents has always been a challenge: editors have too many operations or are very complicated, turning the task of designing a simple document such as a greeting card into a laborious experience.

This paper describes a new interaction model for creating documents in a quick and intuitive way. A variety of content combinations (text, images, backgrounds and decorative items) and document form factors can be used to create different documents. The aim is to provide a better, easier experience at designing loosely-formatted documents. We do not intend to replace complex software systems to produce magazines, but assist small business owners in designing their own leaflets, and enable users with no experience in graphic design to produce their own personalized greeting cards, calendars, photo albums and others.

In our approach, items float freely on the page as new items are inserted or moved around, so that the user (designing a simple brochure for his shop, for example) always sees a page where items are well-spread, do not overlap and he may concentrate on placing a single item knowing that existing items will react to it and place themselves on better positions. Items float on the page and react to each other, trying to be as far away from each other as possible. This is performed through the use of a force-directed method, i. e., a physics model based on electrical repulsion. Items have different charges according to their sizes and the physics model was adapted for the sake of an easier manipulation of the objects. The use of force-directed methods seems to be quite new in the context of document layout. The approach used in this paper is briefly described in Section 3. Details of related approaches are discussed in Section 2.

The end result of our approach is an intuitive authoring method that is able to produce good-looking pages with

[1]`http://www.snapfish.com`
[2]`http://www.shutterfly.com`

small effort. We also provide extra facilities like anchoring an item to a specific position, or anchoring an item to another (so that a picture and a price tag may be moved together easily, for example). After the user is done, a printable document is generated as a PDF [3] file. Sections 4 and 5 describes the development of this interaction model and document generation in detail.

To assess the method described in this paper, a simple tool for authoring personalized calendars was developed. Additionally, a preliminary user evaluation was performed using this prototype, and the obtained results show that the interaction is in fact intuitive and produce results quickly. Some users reported that constructing a layout using the proposed method feels like playing a game, and they are usually less concerned about the design details (for instance, aligning exactly each item on the page on a grid). To show that the end results can be satisfactory, the prototype tool was used to generate sample results as well, which are presented in this paper. The user evaluation and sample results are presented and discussed in Section 6, and the final considerations and future works are discussed in Section 7.

2. RELATED WORKS

Several approaches for automatically constructing mixed-content documents have been proposed in the recent years [8]. Most approaches focused on producing mass-customized documents for Variable Data Printing (VDP) [23], or adapting some content for various media and sizes. In our approach, the creation of the document must be interactive, in order to allow the user to easily specify the desired layout. This is not the case with VDP, where most of document customization takes place in the latest stages of the printing process, thus still relying on templates and reducing the potential for personalization. Additionally, the generated documents must be free-form, as opposed to a grid-based document.

Typical document design systems such as Scribus [21] and InDesign[3]are very rigid in the sense that they require each item to be moved and aligned independently from others, since items cannot interact with each other. For example, items have to be spread on the page manually by the user, whereas in our approach they spread over the page automatically. The problem of automatically producing document layouts is best stated as a constrained optimization problem, where a number of geometric constraints guide an optimization method for finding a good placement and size for each document item [8]. In this sense, most works on document layout are based of constraint solving problems, which can be very expensive to solve. When the layout must be produced quickly, either the document layout model must be simplified or an approximate solution is used [8]. Our approach is based on force-directed methods [16], which have been extensively used for graph drawing and visualization [2, 11], but these approaches seem to never have been used for document layout.

TEX [20] is notable for its use of stretchable space (also known as *glue*) to join pieces of text in a page, which works similarly to a physical model of a spring. However, it only works on horizontal or vertical directions, at specific locations or explicitly defined between pairs of boxes [20]. This works well for low-level formatting, where boxes must be well-aligned (i. e. to form words, lines, paragraphs, and so forth), but in free-form documents items can be influenced by others in all possible directions. Moreover, in our scenario documents are assembled interactively, causing items to regularly find new positions in the page, and thus a different model is needed in our case.

Purvis et al. [18] proposed a genetic algorithm for finding a good placement of document items, using a combination of aesthetic measures [6] to guide the heuristic search. Unfortunately, this process is slow and aesthetics often interfere with each other, so the user may be required to tune aesthetic parameters and start over until a satisfactory layout is obtained. Jacobs et al. [10, 22] describe an approach for creating grid-based documents using templates that adapt for different sizes. However, as pointed out by the authors, producing an adaptive template is a complex task, which is not suitable for a non-expert user, and thus professional templates are still required. The approach by Lin [13] is similar, but appears to be slower and requires an initial document for adapting its layout to variable content. Oliveira [4] describes a method for producing newspaper-like layouts, based on repeated bisection. However, items on a page are not free-form as in brochure-like documents, but rather they are constrained to the hierarchical partition.

In all of these approaches, the end user has little or no involvement when producing the document. Either the task is fully automatic for mass-production or the user is forced to use only the previously-designed page templates.

The work from McCormack et al. [15] uses constraint solving to enable a user to create diagrams that are adjustable to different viewing conditions. It uses an interactive authoring tool as in our work, but as the authors pointed out, the use of constraint solving techniques is only viable when the geometric space on the page is discrete, as in a grid-based document. In our case, documents must be loosely-formatted.

A recent work from Ali et al. [1] appears to be similar to the work proposed in this paper in its use of physics. However, their system requires a template database and a constraint-solving system for initial layouts, which limits the use of physics only as a post-processing stage. Nevertheless, their model handles item repulsion similarly to our work (see Section 4.2), but requires a graph of adjacencies between items, obtained from the template. One interesting difference is the use of a gas pressure model for automatically resizing items during interaction, whereas in our case the resizing of items is performed by the user. Unfortunately, the paper lacks details regarding user interaction and experimental data, making it difficult to compare to the method from this paper.

3. PROPOSED APPROACH

The approach described in this paper is a semi-automatic method for document layout. In other words, the layout is constructed by a non-specialist user by means of an authoring tool, which automatically moves items on a page, to avoid item overlapping and bad distribution over the page [6].

User interaction involves selecting items from a set of pictures and textual content, dragging these to one ore more empty pages. Automatic distribution of content occurs in real-time, i. e., the page is reorganized as the user constructs it. For practical purposes, this work assumes some simplifications in the interaction model:

[3]`http://www.adobe.com/products/indesign`

- The page is loosely-formatted, i. e., there are no columns or grids;
- Pictures and text are rectangle-shaped;
- Items cannot be rotated, but may be scaled or deleted;
- Collision detection is not performed [16], i. e., items may overlap with each other, if necessary (see Section 4.2 for details).

The user may drag images and text boxes to an empty page and those items float around: they repel each other using an electrical repulsion model and as a consequence of that repulsion content is automatically spread on the page (see Figure 1). As some item is resized or moved other items recede to provide space for it. Moving items makes other items react to that movement by moving as well, and a chain-effect of several editing software is avoided: to move one item from A to B one has to move another item from B to C to make room for it, and to move the second one has to move a third from C to D and so on. In our approach items distribute themselves and react to the movement of the first item.

After completing the desired layout, the final document must be produced by a rendering engine in order to obtain a printable PDF [3] file (see Section 5).

4. INTERACTION MODEL

While traditional physics simulation models usually attempt to replicate real world physics [16], we found that these lead to non-intuitive user cues and a poor user experience. Thus, the simulation engine used in our method has been built from scratch using a simplified model for the physical forces, aimed at reducing user surprise.

The user is initially presented with an empty document, and the final document is constructed interactively, as the user inserts or manipulates content. Any number of pages can be used to create a document, but each page is independent, and no attempts are made to perform automatic pagination [17] or flowing items between pages.

Items already placed in the page continuously interact with each other. At the same time, the user can still interact with the system, performing any of the following actions:

- Inserting new items into the page;
- Deleting items from the page;
- Moving items around;
- Resizing items.

Items are either text chunks, color blocks or pictures, which are represented by rectangles in this paper's approach. An item i is defined by its central position (x_i, y_i), and size (w_i, h_i). For simplicity, we assume that content is already available from some source or created by the user prior to the interaction with the layout system.

The entire system works according to the user events mentioned above, as well as the physical simulation itself, which is carried out in real-time. Each simulation step computes new positions for all items present in a page, if distribution is necessary. For efficiency, the simulation may execute several steps for each time the document frame is drawn on the screen. This will be described in more detail in Sections 4.1 and 4.3.

4.1 Physics model

The problem of finding new positions of items (for a better distribution on a page) can be modeled as an energy minimization approach, similar to graph drawing methods [2].

Although energy minimization can be achieved efficiently with general optimization heuristics such as simulated annealing [5], a force-based simulation is used in this paper. The forces are computed for each time step and the items are moved accordingly. Although slower, this allows the user to interact with the system and follow any changes while it works in finding a good placement. Despite being a standard approach for graph layout, its use is new in the general document layout context. Thus, several modifications were necessary to adapt the force-directed method for this context.

From a usability perspective, items should move smoothly on the page, so that the user is able to easily follow the items and thus feel more familiarized when interacting with the system. Therefore, a *velocity* parameter (vx_i, vy_i) was also added for each item i, allowing items to move by inertia. Section 4.3 discusses this in more detail.

Given this description, an equilibrium state of this system is defined as the state that minimizes the total kinetic energy ε, i. e.,

$$\varepsilon = \sum_i w_i\, h_i\, \left(vx_i^2 + vy_i^2\right). \tag{1}$$

Item positions and velocities are computed using simple Euler integration [16]. This method updates positions and velocities explicitly in discrete time steps Δ_t, until the system reaches an energy minimum, i. e., $\varepsilon = 0$. To compute new velocities (and consequently, item placement), it is necessary to compute the forces generated between items first.

4.2 Repulsion forces

The basic working force is the electrical repulsion between items. Every item has an influence on every other item (i. e., visibility issues are not considered), based on the distance between each other and their sizes. These forces are computed pairwise. When a page is overfull with items, the repelling forces will not be enough to prevent items from intersecting. Although collision detection could be used to prevent intersection between items [16], it showed to cause some instability in earlier tests, because of the increased kinetic energy from repulsion. However, items will always try to be as far apart as possible from each other, so overlapping seldom occurs.

The repulsion force between two items uses the simplification from Di Battista et al. [2] for Coulomb's Law. Thus, the repulsion force vector $\overrightarrow{f_{u,v}}$ of item v over item u is defined as

$$\begin{aligned}\overrightarrow{f_{u,v}} &= \left(fx\left(u,v\right), fy\left(u,v\right)\right)\\ fx\left(u,v\right) &= \frac{q_v\,\alpha}{\sigma\left(u,v\right)^2}\,\frac{\Delta_x}{\sigma\left(u,v\right)}\\ fy\left(u,v\right) &= \frac{q_v\,\alpha}{\sigma\left(u,v\right)^2}\,\frac{\Delta_y}{\sigma\left(u,v\right)},\end{aligned} \tag{2}$$

where $q_v = w_v\, h_v$ is the *charge* of item v, $\Delta_{x,y}$ is the displacement in x or y, α is the repulsion constant that can be tuned according to user preferences, and $\sigma\left(u,v\right)$ measures

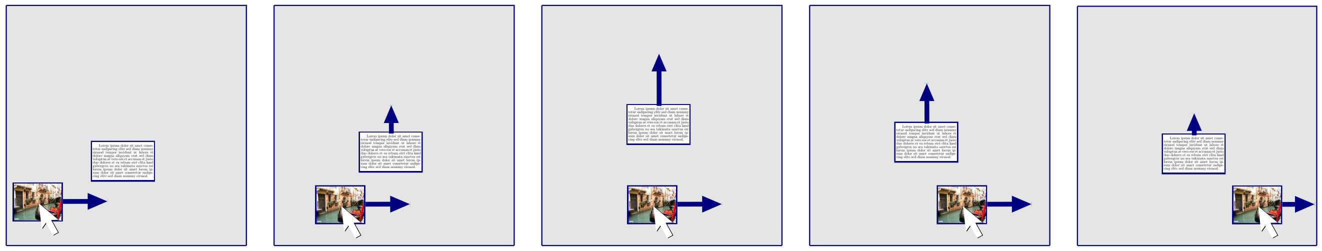


Figure 1: Moving items according to physics laws: the lower item moves from left to right and the upper item recedes as the lower one approaches.

the distance between u and v in the page. The distance between two items is usually measured between their central positions. Thus, the charge parameter q is set to be proportional to the item's area, since larger items need to be further away from others to avoid overlapping. Distance can also be measured in different ways, each achieving different results. Section 4.6 discusses two approaches for computing $\sigma(u, v)$.

Since multiple items will repel an item u, a resultant force must be computed so that a repelling direction and intensity is known in order to move item u. Thus, for every item u, the resultant force vector $\overrightarrow{F_u}$ is computed as

$$\overrightarrow{F_u} = \sum_{v \neq u} \overrightarrow{f_{u,v}} \tag{3}$$

From the resultant forces, the new velocities and positions can be computed for the next time step.

4.3 Making items move

After computing the resultant force vectors $\overrightarrow{F_i}$ for every item, they must be applied in order to move items for one time step. We developed a simple physical simulation consisting of force, velocity and position updates only, computed using the Euler integration method. Although Euler integration is usually unstable for real-time physical simulation [16, 7], we found it to be simple and stable enough for the purposes of this work. Moreover, it is efficient enough to be used in real-time in an interactive system, such as the one proposed in this paper.

To compute positions, assuming that an item i is positioned at $\left(x_i^t, y_i^t\right)$ at an instant t, we compute the positions for $t+1$ as

$$\begin{aligned} x_i^{t+1} &= x_i^t + \Delta_t\, vx_i \\ y_i^{t+1} &= y_i^t + \Delta_t\, vy_i. \end{aligned} \tag{4}$$

For velocities, adding a damping (i. e., friction) coefficient is required, as the inertia could cause the system to move forever and not settle. Thus, velocity for item i is computed as

$$\begin{aligned} vx_i^{t+1} &= \left(vx_i^t + \Delta_t\, fx_i\right)\,(1-\delta) \\ vy_i^{t+1} &= \left(vy_i^t + \Delta_t\, fy_i\right)\,(1-\delta)\,, \end{aligned} \tag{5}$$

where δ is a damping coefficient in the range $[0 \ldots 1]$, being $\delta = 0$ a frictionless system and $\delta = 1$ a static system.

4.4 Constraining items to the page

Repulsion forces are not enough to keep items distributed inside the page, since they can repel each other apart infinitely far. Therefore, it is necessary to incorporate more forces into our model, so that items are confined within the document page.

An approach that works well is to add virtual or "wall" forces in the page boundaries, so that escaping items are always pushed back inside.

Figure 2 shows a simple way to do this. In this case, forces are added for each side of the page (i. e., top, bottom, left and right), and follow the items in both axes, and using the same charge q as the item being pushed. This results in all items attempting to float to the center of the page. The dispute for this space results in items being spread evenly over the page.

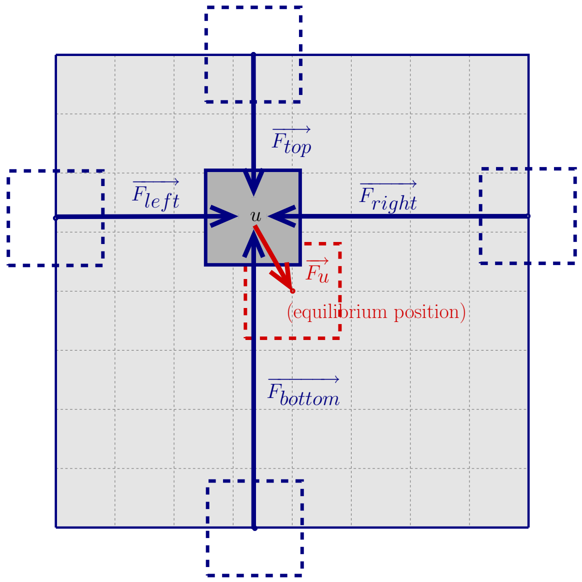

$\overrightarrow{F_{top}}$
$\overrightarrow{F_{left}}$
u
$\overrightarrow{F_{right}}$
$\overrightarrow{F_u}$
(equilibrium position)
$\overrightarrow{F_{bottom}}$

Figure 2: Constraining an item by adding virtual forces, forcing it to the center of the page.

4.5 Anchoring & grouping items

Sometimes, a user might wish to "glue" an item at a certain location in the page, and not allow it to move away from that location. This will be referred in this paper as *anchoring* an item from now on.

The force model can be adapted to support anchoring, by creating a new attribute (ax_i, ay_i) to store the anchoring location for item i, and a force to attract item i to the anchor point. This force is then added to the resultant force $\overrightarrow{F_i}$.

An intuitive force model that can be used for anchoring is a spring model. Hooke's law [2] can be used to keep (x_i, y_i) close to (ax_i, ay_i), while still allowing the item to move when under excessive pressure. The spring model was further simplified by assuming the spring rest length to be zero, since the desired position for an item is exactly over its anchor location. Thus, the spring force vector $\overrightarrow{g_i}$ of item i is described as

$$\begin{aligned} \overrightarrow{g_i} &= (gx(i), gy(i)) \\ gx(i) &= \gamma\,\beta\,\Delta_x \\ gy(i) &= \gamma\,\beta\,\Delta_y, \end{aligned} \tag{6}$$

where β is the spring stiffness constant, γ is a user adjustable parameter in the range $[0\ldots1]$, and $\Delta_{x,y}$ are the displacements in x and y. The γ parameter was introduced for flexibility: it can be used to allow the user to fine tune the forces of individual items, by making the distance constraint between an item and its anchor point more or less rigid.

Although intuitive, this method can make the physical system more unstable due to the amount of movement produced when stretching the spring too far from its anchor point. For more stability, the force model can be dropped and the item's position can be directly manipulated after the items are moved at the end of the current time step.

The anchoring idea can be extended for a finer control of the layout as well. For instance, items can be anchored to each other, so they always end up near the same location. This idea was implemented in a prototype but was not included in the user evaluation from Section 6.3.

4.6 Measuring distance between items

A measure of distance must be defined in order to compute repulsion forces. The initial attempt was to measure the euclidean distance σ between items u and v as

$$\sigma(u,v) = \sqrt{(x_u - x_v)^2 + (y_u - y_v)^2} \tag{7}$$

between the center (x, y) of items u and v.

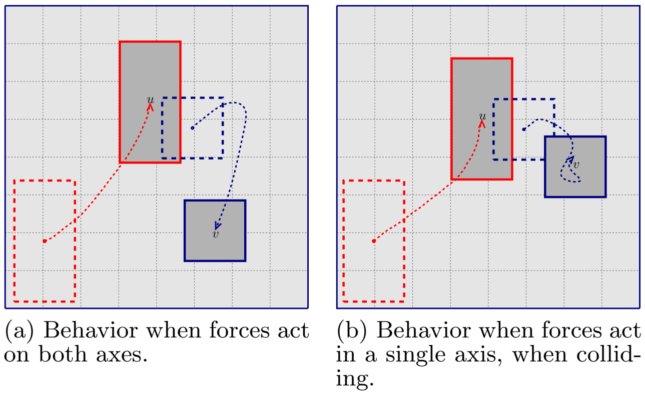
From early testing we found that, although this measure provided a realistic behavior physically, it caused the items to move too much on the page until they reached an equilibrium. In some circumstances this behavior may cause difficulties for a user to follow and interact with the system.

To circumvent this effect, an alternative distance measure was developed. It is based on the same euclidean distance from Equation (7). However, when two items u and v intersect in the x or y axes, the intersecting axis is cut off from the equation, i. e., only one direction component of the distance is used:

$$\sigma(u,v) = \begin{cases} |x_u - x_v| & y\text{-only intersection} \\ |y_u - y_v| & x\text{-only intersection} \\ \sqrt{(x_u - x_v)^2 + (y_u - y_v)^2} & \text{otherwise.} \end{cases} \tag{8}$$

From our tests, this approach seemed to be more stable than the one from Equation (7), because as items come closer to each other, less movement is generated, causing the physical system to stabilize more quickly.

Figure 3 illustrates the behavior of each different distance measure when dragging item u against item v.



(a) Behavior when forces act on both axes. (b) Behavior when forces act in a single axis, when colliding.

Figure 3: Repelling behavior using different approaches for measuring distance. Item u is being dragged by a user towards item v, which gets repelled in different ways as a consequence.

4.7 Stability

The final document can only be sent to a rendering engine when $\varepsilon = 0$, thus it is necessary to ensure that the physical system will be stable enough to converge. As shown in Section 4.3, a damping coefficient is necessary to ensure that the total sum of forces at each time step will eventually converge to zero. However, given the poor approximation obtained with the Euler equations when the time step is large, the force vectors may grow larger and cause a well-known phenomenon of *overshooting* [7, 16]. Moreover, the physical system may naturally end up in an oscillatory behavior.

For enhancing simulation stability, the time step Δ_t and damping δ parameters can be tuned for different values, depending on the number of items, page size, and others. Small time steps result in a more stable and precise simulation [16], but require more processing power.

To guarantee that the physical system will eventually stop without completely freezing it, i. e., $\delta = 1$, the repulsion constant α and spring stiffness β can be automatically damped over simulation steps by a small factor, e. g. 0.01, until the system stops or the user performs a new operation on the page, such as adding a new element. In the latter case, the constants are reset to their original values previous to damping. The damping factor also automatically increases as more items are added to the page, reducing instability.

5. GENERATING THE DOCUMENT

After deciding the final layout, the user can produce a PDF file through a rendering engine. The document description is given as a page rectangle (W, H) and a set of item rectangles (xc_i, yc_i, w_i, h_i) for each item i.

Content is placed over each rectangle independently by an external rendering engine. For this, LaTeX [12], Scribus [21] or the iText Java library [14] can be used to place pictures and textual content inside each rectangle. Other engines may be used, as long as they support some programming capabilities, such as font resizing in a text box. Both LaTeX and iText renderers were implemented in a prototype, achieving similar results.

To place pictures, the width and height of the picture are simply scaled to the rectangle size. We do not attempt to perform automatic cropping [24], thus the user interface is responsible for preserving the original picture's aspect ratio as the user resizes a picture.

For textual content, attributes such as font type and color may be defined for each different text. However, since we treat text as chunks that are completely contained in their enclosing rectangle, the font sizes are automatically computed so that the text completely fills the rectangle. Automatic justification and line breaking are also required. In this paper, a simple greedy line-break algorithm [9] was used along with left aligned text, but other methods may be used.

Example camera-ready documents created by the rendering process are presented in Section 6, produced by a prototype authoring tool that uses iText for rendering.

6. RESULTS AND EVALUATION

To produce results and evaluate the proposed method, a prototype was developed for authoring personalized calendars, which will be described briefly in this section. From this prototype, examples of different documents that can be produced are presented in Section 6.2.

A preliminary user evaluation was performed using this prototype tool as well, and will be presented in this section.

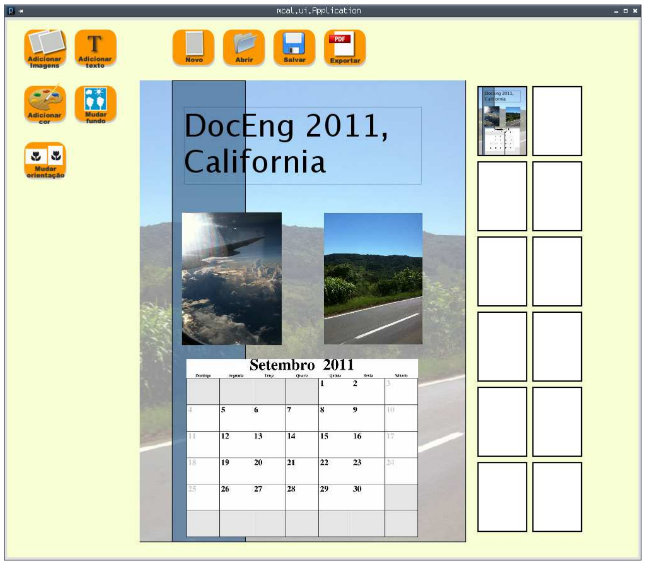
6.1 Case study: a personalized calendar authoring tool

To test the proposed interaction model, a simple authoring tool was created, intended for creating personalized calendars by a user. Calendars can be created, opened, saved and exported to a PDF file. Twelve day-month panels are already provided as pictures in the user interface, and the user can select pictures from his own collection and create text boxes as well. Other available items are colors boxes and page backgrounds. These decorative items are not affected by the physics engine, and can be freely placed and overlapped by the user. The main content is distributed automatically by the physics engine, but the possibility of anchoring (Section 4.5) was not included, for the purposes of the user evaluation (Section 6.3). In addition, the user has two different pages available for selection and editing. It is also possible to switch between portrait and landscape orientations. Figure 4 shows the main user interface of the authoring prototype.

After finishing the document, the user can produce a PDF file intended as a printing material, using a print option that renders the document into a PDF as described in Section 5.

The authoring tool has been implemented using the Processing API [19] in Java, and deployed as an applet in the Web[4].

It is important to note that the tool is still a prototype and has a limited set of capabilities regarding style: there is only one font style available for textual items, a limited range of colors, and a single page size (A4). Although it is possible to produce simple but aesthetically pleasing documents, the main purpose of the prototype is to evaluate the actual interaction with the physics model, not to produce professional-looking documents.

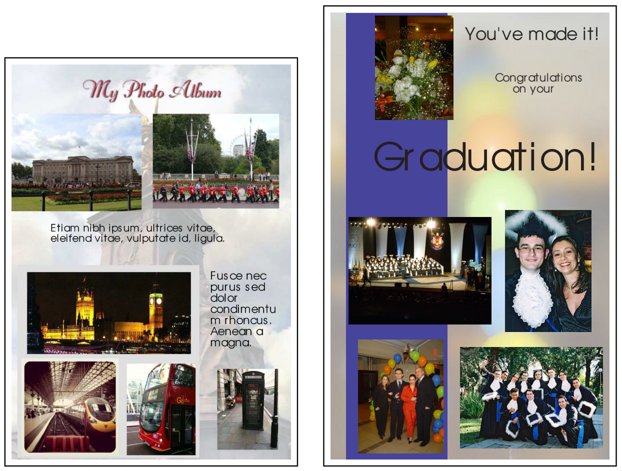


DocEng 2011, California

Figure 4: Screen shot of the prototype authoring tool.

6.2 Application examples

This section shows some example pages produced with our interactive approach. Because of the limitations of the prototype, the sample documents are not professional-looking, although they demonstrate what types of documents can be produced by the proposed interaction method. Moreover, the examples demonstrate how pictures, text and decorations can easily be combined into a document, and how the interaction model (Section 4) and the rendering process (Section 5) are integrated in the authoring tool. Figure 5 shows four examples of calendar pages, produced by the authoring tool described in Section 6. An example of a 5x8 personalized greeting card and photo album are shown in Figure 6 and were created with a more complete version of the tool that allows anchoring and different page sizes.


My Photo Album

Etiam nibh ipsum, ultrices vitae, eleifend vitae, vulputate id, ligula.

Fusce nec purus sed dolor condimentum rhoncus. Aenean a magna.

You've made it!

Congratulations on your

Graduation!

Figure 6: Examples of a photo album page and a 5x8 greeting card.

[4]Our prototype can be tested at `http://www.cpca.pucrs.br/BrochureLayout`

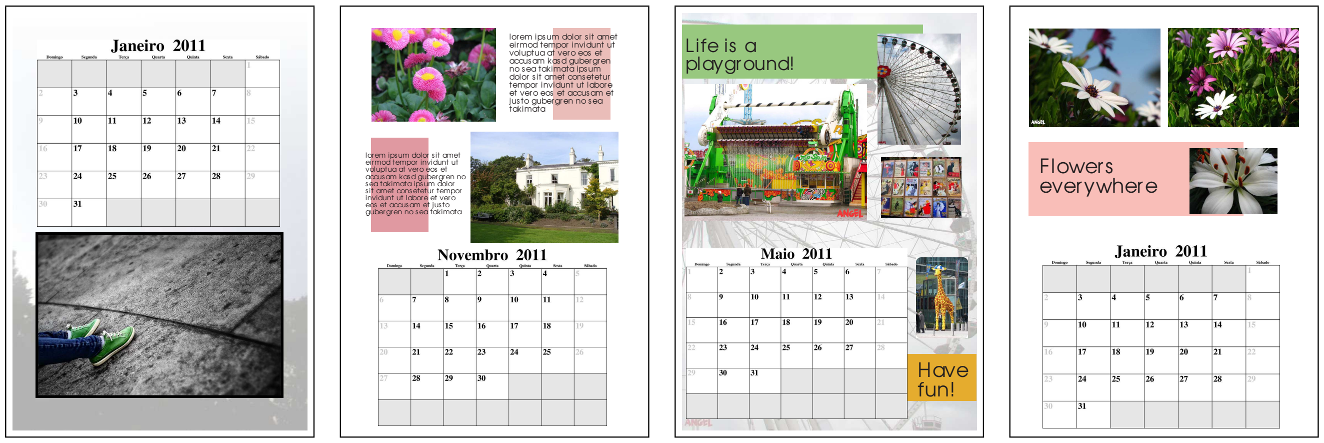


Figure 5: Sample personalized month calendars generated by the prototype.

6.3 User evaluation

To evaluate the proposed interaction method, a simple user study was performed. The experiment consisted in obtaining volunteers to test our prototype and produce a personalized calendar using pictures from two local soccer teams, according to the volunteers' personal tastes. There were no restrictions on the number of pictures or the time required to produce a document, but each volunteer could only produce at most two pages of a calendar. A minimum of instructions were given to the volunteers regarding the operation of the prototype tool and what were the goals of the evaluation. They were further instructed to stop and generate a final PDF of their documents whenever they felt the document looked as desired, or have given up using the tool. At the end of the process, the volunteers were asked to fill out a questionnaire, and had the option to send the produced calendar by e-mail.

In order to be as simple as possible, the version of the prototype used in the evaluation did not contain anchoring features. This enabled a better assessment of the interaction proposed in this paper, because the user might otherwise get distracted by the user interface features, and not pay attention to the interaction model. The physical parameters (described in Section 4) were tuned to the the following values:

- Electric repulsion was set to $\alpha = 150$;
- Velocity damping was set to $\delta = 0.01$;
- Time step was set to $\Delta_t = 0.005$;
- 500 simulation steps per frame;
- 33 frames per second (roughly);
- Repulsion forces were not damped.

For obtaining volunteers for this evaluation, a kiosk was assembled in the main hall of our university's computer science department, during rush hours. All volunteers were undergraduates from computer science, computer engineering and information systems courses.

In total, the evaluation spanned 3 week days, 2 hours per day, and a total of 27 people aged from 17 to 30 years old participated in this evaluation. These volunteers had no previous contact with the prototype.

After testing the authoring prototype, the volunteers were asked to answer a questionnaire consisting of five questions, regarding the interaction model implemented in the prototype:

1. Did the automatic distribution aid you for creating the page more quickly?
2. How did you feel about the automatic positioning of page items?
3. How much did you like the final result?
4. How did you feel about manipulating (moving, deleting, resizing) images on the page?
5. Would you use such a program?

The volunteers had the liberty of answering as many questions as they desired. Questions 1 and 5 were yes/no questions, while the answers for questions 3 and 4 were required to be given in a $1 - 5$ likert scale (i. e., very unsatisfied, unsatisfied, partially satisfied, satisfied and very satisfied). Question 2 had four levels, but encouraged the volunteer to answer the question subjectively. A space for comments was available for every question, allowing the volunteers to express their opinions better.

6.4 Final results and discussion

The evaluation results are presented in Figures 7, 8, 9, 10 and 11 and the comments from each volunteer are also discussed in this section.

As shown in Figure 7, the majority of the volunteers answered question 1 agreeing that the tool is responsive and quick to produce results.

The results from question 2 are shown in Figure 8. From the comments, most users found the interaction immediately intuitive or got used after a while. Few users however, reported feeling somewhat uneasy with the interaction, either because the automatic positioning interferes too much in the layout, or that a finer control is required to be more usable.

Regarding the final document layout and the generated PDF file (question 3), the volunteers were mostly partially

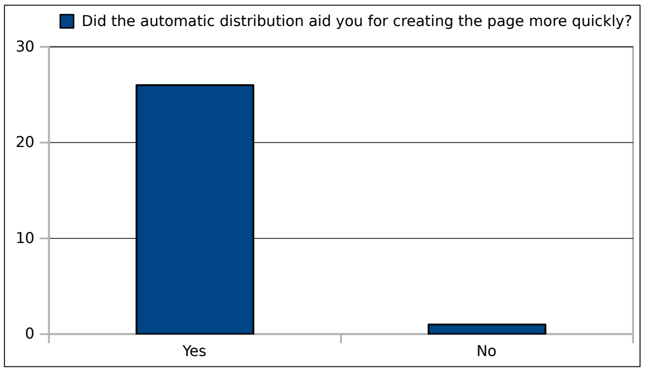


Figure 7: Evaluation results from question 1.

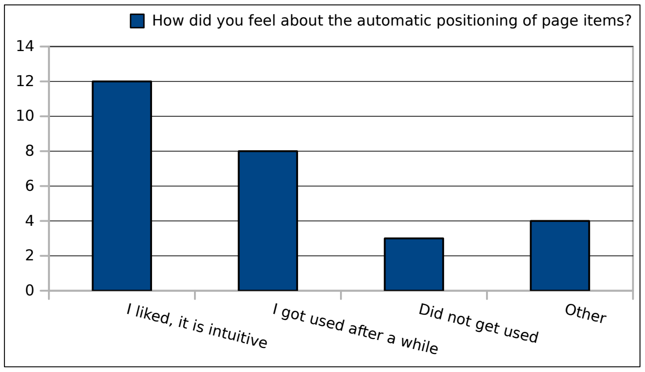


Figure 8: Evaluation results from question 2.

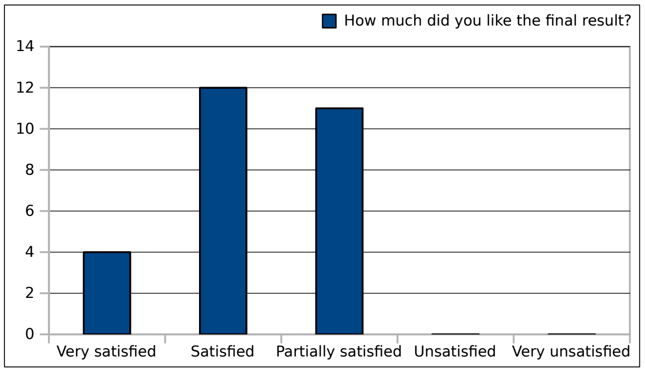


Figure 9: Evaluation results from question 3.

satisfied, as shown in Figure 9. Most likely this is due to the currently limited capabilities of the authoring tool.

As for the image manipulation (question 4), the final results are shown in Figure 10. Twelve users reported that the interaction is intuitive, provides reasonable results quickly and the available operations seem to be clear as well. Five users reported user interface issues unrelated to the interaction model, such as glitches in the prototype tool, the location for certain operations or lack of some facilities such as adding a large set of images to the page at once. Nine users did not get used to the interaction model, and reported a lack of control of the layout, but suggested the use of a fixing mechanism (such as the anchoring method described in Section 4.5), or the splitting of the document page into areas for grouping items together.

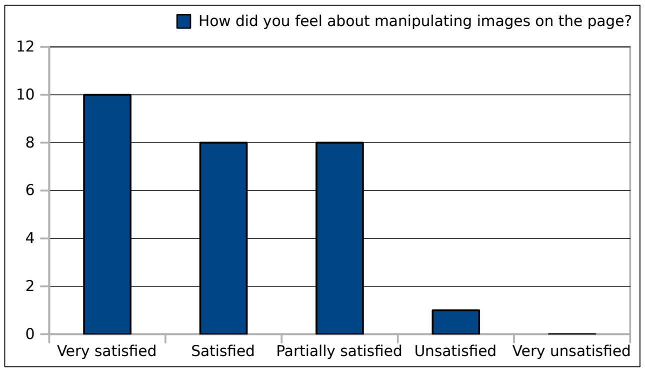


Figure 10: Evaluation results from question 4.

Finally, the results for question 5 are shown in Figure 11. Most of the volunteers approved the prototype tool and would use it for creating their personalized documents. The main reason reported by the volunteers was either because there are no immediately obvious alternatives available or that the tool is really pleasant to use and easy to interact. Few complained of a lack of more features to control positioning of items on the page and felt that, for their purposes, simpler tools such as Microsoft's PowerPoint or Paintbrush would be sufficient.

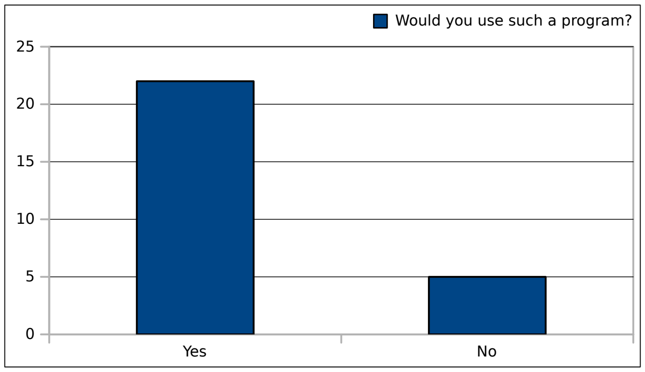


Figure 11: Evaluation results from question 5.

7. CONCLUSIONS AND FUTURE WORKS

This work presented a new interaction model for authoring personalized documents, such as photo albums, brochures, leaflets and others. It is based on a physics engine developed specifically for this purpose. A prototype authoring tool has been developed using the physics engine, and was deployed as a Web application[5] to create personalized photo albums, greeting cards and calendars. Although this tool is only a prototype and does not contain a wide range of artwork, shapes and design choices that would be needed in a whole product offering, the basic principles of the tool are functional, and usable to create small projects and experiment with the proposed model.

[5]`http://www.cpca.pucrs.br/BrochureLayout`

Marketing collaterals are usually expensive to create since it typically requires expert graphic design. Thus, we believe that such automated method will help lower the cost of generating layouts for small businesses (brochures, leaflets and others), allowing these documents to be individually customized. Moreover, a quick and intuitive method for creating documents will encourage the creation of personalized prints (photo albums, greeting cards and others) by non-designers, in an easier way than currently performed today by commercial applications, such as Snapfish[6] and Shutterfly[7].

For future works, there are several possible improvements to the current method. In particular, we are currently working on an extension to support arbitrarily shaped regions and cutouts (since these are common in brochures and greeting cards) and to support interactions among multiple polygonal items, since the current method can only handle rectangular items. A similar idea appears in the paper from Ali et al. [1], but few details are provided. Thus, we feel that further investigation on the use of polygons is still required, as there are implications for the physical model (e. g. performance and stability) and user interaction (e. g. designing and resizing arbitrary regions).

Other possibilities are the inclusion of grid capabilities on a page (for instance, dividing a page into rows or columns), and the addition of hierarchical groups of items for facilitating their placement together. It would also be interesting to experiment with more stable physics solvers, such as Verlet integration [16], and analyze speed/accuracy trade-offs.

Regarding the user study, we found that the obtained results – albeit preliminary – are encouraging, and most of the time the automatic distribution using a physical model is intuitive and pleasant to use. However, it might get in the way of the user at times, e. g. by occasionally moving items away from their previously set location. As the volunteers themselves suggested, an anchoring mechanism or the division of the layout into several areas would be able to solve this problem. Besides investigating the use of such features, we also intend to perform a comparative study with a commercial application for creating personalized documents (such as Snapfish or Shutterfly), to better understand how automatic placement helps users in assembling personalized documents.

8. ACKNOWLEDGMENTS

This paper was achieved in cooperation with Hewlett-Packard Brasil Ltda. using incentives of Brazilian Informatics Law (Law n°. 8.2.48 of 1991).

9. REFERENCES

[1] K. Ali, K. Hartmann, G. Fuchs, and H. Schumann. Adaptive layout for interactive documents. In *Proceedings of the 9th international symposium on Smart Graphics*, SG '08, pages 247–254, Berlin, Heidelberg, 2008. Springer-Verlag.

[2] G. D. Battista, P. Eades, R. Tamassia, and I. G. Tollis. *Graph Drawing: Algorithms for the Visualization of Graphs*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1998.

[3] R. Cohn. *Portable Document Format Reference Manual*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1993.

[4] J. B. S. de Oliveira. Two algorithms for automatic page layout and possible applications. *Multimedia Tools Appl.*, 43(3):275–301, 2009.

[5] M. Fleischer. Simulated annealing: past, present, and future. In *WSC '95: Proceedings of the 27th conference on Winter simulation*, pages 155–161, Washington, DC, USA, 1995. IEEE Computer Society.

[6] S. J. Harrington, J. F. Naveda, R. P. Jones, P. Roetling, and N. Thakkar. Aesthetic measures for automated document layout. In *DocEng '04: Proceedings of the 2004 ACM symposium on Document engineering*, pages 109–111, New York, NY, USA, 2004. ACM Press.

[7] L. Hilde, P. Meseure, and C. Chaillou. A fast implicit integration method for solving dynamic equations of movement. In *Proceedings of the ACM symposium on Virtual reality software and technology*, VRST '01, pages 71–76, New York, NY, USA, 2001. ACM.

[8] N. Hurst, W. Li, and K. Marriott. Review of automatic document formatting. In *DocEng '09: Proceedings of the 9th ACM symposium on Document engineering*, pages 99–108, New York, NY, USA, 2009. ACM.

[9] N. Hurst, K. Marriott, and P. Moulder. Minimum sized text containment shapes. In *DocEng '06: Proceedings of the 2006 ACM symposium on Document engineering*, pages 3–12, New York, NY, USA, 2006. ACM.

[10] C. Jacobs, W. Li, E. Schrier, D. Bargeron, and D. Salesin. Adaptive grid-based document layout. In *SIGGRAPH '03: ACM SIGGRAPH 2003*, pages 838–847, New York, NY, USA, 2003. ACM Press.

[11] M. Kaufmann and D. Wagner. *Drawing Graphs, Methods and Models*, volume 2025 of *LNCS*. Springer, 2001.

[12] L. Lamport. *LaTeX: A Document Preparation System*. Addison-Wesley, Reading, Massachusetts, USA, Boston, MA, USA, 1986.

[13] X. Lin. Active document layout synthesis. *Proceedings. Eighth International Conference on Document Analysis and Recognition, 2005.*, pages 86–90 Vol. 1, 29 Aug.-1 Sept. 2005.

[14] B. Lowagie. *iText in Action*. Manning Publications Co., Greenwich, CT, USA, 2010.

[15] C. McCormack, K. Marriott, and B. Meyer. Authoring adaptive diagrams. In *Proceeding of the eighth ACM symposium on Document engineering*, DocEng '08, pages 154–163, New York, NY, USA, 2008. ACM.

[16] M. Müller, J. Stam, D. James, and N. Thürey. Real time physics: class notes. In *ACM SIGGRAPH 2008 classes*, SIGGRAPH '08, pages 88:1–88:90, New York, NY, USA, 2008. ACM.

[17] M. F. Plass. *Optimal pagination techniques for automatic typesetting systems*. PhD thesis, Stanford, CA, USA, 1981.

[18] L. Purvis, S. Harrington, B. O'Sullivan, and E. C. Freuder. Creating personalized documents: an optimization approach. In *DocEng '03: Proceedings of*

[6]`http://www.snapfish.com`

[7]`http://www.shutterfly.com`

the 2003 ACM symposium on Document engineering, pages 68–77, New York, NY, USA, 2003. ACM.

[19] C. Reas and B. Fry. *Processing: A Programming Handbook for Visual Designers and Artists*. The MIT Press, Sept. 2007.

[20] D. Salomon. *The Advanced TEXbook*. Springer-Verlag, Berlin, 1995.

[21] C. Schäfer and G. Pittman. *Scribus: Open-Source Desktop Publishing*. FLES books Ltd., United Kingdom, 2009.

[22] E. Schrier, M. Dontcheva, C. Jacobs, G. Wade, and D. Salesin. Adaptive layout for dynamically aggregated documents. In *IUI '08: Proceedings of the 13th international conference on Intelligent user interfaces*, pages 99–108, New York, NY, USA, 2008. ACM.

[23] R. Sellman. Vdp templates with theme-driven layer variants. In *DocEng '07: Proceedings of the 2007 ACM symposium on Document engineering*, pages 53–55, New York, NY, USA, 2007. ACM.

[24] V. Setlur, S. Takagi, R. Raskar, M. Gleicher, and B. Gooch. Automatic image retargeting. In *MUM '05: Proceedings of the 4th international conference on Mobile and ubiquitous multimedia*, pages 59–68, New York, NY, USA, 2005. ACM.