Document Design with Interactive Evolution

Juan C. Quiroz, Amit Banerjee, Sushil J. Louis, Sergiu M. Dascalu

Abstract We present human guided evolution of brochure documents. The user interacts with a genetic algorithm, which evolves placeholders, each placeholder represented with one of three shapes: (1) ellipse, (2) rectangle, and (3) rounded rectangle. The user guides the evolutionary process by evaluating a small subset taken from a large population of documents. Along with the subjective user input, individuals in the population of the genetic algorithm are evaluated on a set of objective heuristics for document design. We present pretest results, including an evaluation of the tool and documents created.

1 Introduction

Design is a fundamental, purposeful, pervasive and ubiquitous activity and can be defined as the process of creating new structures characterized by new parameters, aimed at satisfying predefined technical requirements. It consists of several phases, which differ in details such as the depth of design, kind of input data, design strategy, procedures, methodology and results [12]. Usually the first stage of any design process is the preliminary or the conceptual design phase, followed by detailed design, evaluation and iterative redesign [3]. We are interested in supporting the creative conceptual design phase by allowing for the exploration and the ability to assess alternative design solutions. In the work presented in this paper we look at supporting a simple design task of creating a brochure document, where we specify a small set

Juan C. Quiroz, Sushil J. Louis, Sergiu M. Dascalu
University of Nevada, Reno,
Reno, NV 89557, USA e-mail: {quiroz, sushil, dascalus}@cse.unr.edu

Amit Banerjee
The Pennsylvania State University, Harrisburg
Middletown, PA 17057, USA e-mail: aub25@psu.edu

of requirements, but leave the rest of the problem to be defined by the user through exploration of designs.

We use genetic algorithms (GAs) to explore document designs. GAs are search algorithms based on the principles of genetics and natural selection [6]. GAs consist of a population of individuals, where each individual is a potential solution to the problem being solved. Assigning a fitness to individuals in the population plays a central role in the search success of a GA. However, there are times, such as when we are dealing with subjective criteria, when it is difficult if not impossible to determine the fitness of individuals in a GA population [13]. Interactive genetic algorithms (IGAs) differ from GAs in that the objective fitness evaluation is replaced with user evaluation, thus allowing for the user to guide the evolutionary process through subjective input [13]. By doing so, IGAs incorporate human knowledge, emotion, intuition, and preference into GAs. Figure 1 shows the process involved in an IGA.

Effective IGAs have to overcome several issues. GAs usually rely on large population sizes running for many generations, but asking a user to make hundreds or thousands of choices is likely an unrealistic task. A user would rapidly fatigue and/or lose interest. Furthermore, because of the subjective nature of human input, it can lead to users changing their goals through the IGA run, leading to noisy fitness landscapes – which coupled with user fatigue can result in suboptimal solutions [8].

The work presented here is specifically interested in the development of a tool based on human guided evolution, which would allow users to create documents with an IGA. Document templates can be found on various websites and some come as default installs in popular programs such as word processors. However, the support for users to modify an existing template to suit their particular needs is lacking. Many times the options given to users is to use an existing template or to start from scratch. GAs have been used for document design, but not with direct user feedback [5]. Our work is different in that not only is the evolution of documents driven by subjective criteria, but also by objective criteria taken from document style guidelines. We further address how the affordance of IGAs affects the user experience and the user's ability to create satisfactory document designs.

The remainder of this paper is organized as follows. Section 2 discusses background information on the use of GAs to create document layouts. Section 3 illustrates how the document design problem was mapped to IGAs. In Section 4 we

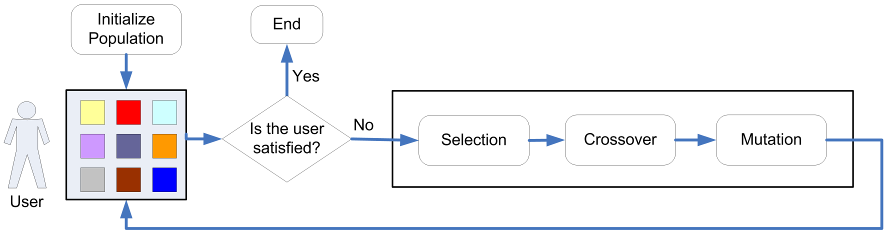


Fig. 1: Interactive genetic algorithm

present a discussion and evaluation of the system and created documents. Section 5 takes a look at affordance as a key challenge for IGAs. Finally, Section 6 presents our conclusions and directions for future work.

2 Background

Geigel and Loui present the use of a GA to evolve album page layouts [5]. A collection of pictures are loaded into the system, which then separates the pictures into various pages. Each image in a page is encoded using a 4-tuple: (1) x coordinate, (2) y coordinate, (3) scaling factor, and (4) rotation angle. These floating point values are arranged in an array, resulting in a chromosome of floating point values instead of the traditional binary string representation used in GAs. User interaction is limited to the specification of user preferences at the start of the evolutionary process, including attributes such as balance, spacing, emphasis, and unity [5]. The user specifies a value between 0 and 1 for each of the preference attributes, with 0 representing a criteria not considered important for the resulting album page layouts and 1 representing an important attribute. During fitness evaluation each of these attributes is computed by analyzing the images in a page, with the resulting values scaled by the preference values specified by the user. In our system the user does not specify such preference values for attributes. Instead, the user directly guides the evolutionary process by participating during each generation by selecting the best document in the population.

3 Interactive Evolution of Documents

In our previous work we applied interactive evolution to floorplanning [2, 10]. For floorplanning we used a simple graphic representation to represent rooms in an apartment. The boundary of the floorplan was depicted using a rectangular shape. We then recursively created a tree, by subdividing rooms either vertically or horizontally (starting with the entire floorplan as a single room and as the root node). In the end, the resulting floorplan (originally a single rectangle) ended up subdivided into smaller areas, each area representing a different room in the floorplan. Because of the way we originally coded the floorplan representation, only rectangular or square rooms could be created.

We wanted to build on the framework we had used for floorplanning. We figured that by taking the rectangular rooms created, and allowing for various transformations to be applied to each room while drawing each room respectively, we would be able to get some degree of overlap and interesting shape combinations. If we further allowed for various shapes to be drawn, such as drawing a circle instead of a rectangle for where a room should be, we would be able to achieve an even greater degree of variation and possibly creative and interesting documents.

We allowed for three types of shapes to be drawn: (1) rectangles, (2) ellipses, and (3) rounded rectangles. Each of these shapes could be scaled respectively along the x or y axis, by up to 10%, either scaling up or scaling down. These changes allow for the original floorplan representation to be transformed into a collection of shapes, where either each shape can represent a placeholder for content (such as text or an image) or where the collection of shapes could represent a background design.

We initialize the documents similar to the floorplans. Except that once we know the allocation of the rooms, we assign the shapes to one of four quadrants, based on the shapes' locations, using the shape's center as the point of reference. This results in a quad-tree of depth one.

3.1 Fitness Evaluation

The fitness of every individual in the population was assigned by combining objective and subjective heuristics. Subjective heuristics consisted of comparing attributes of individuals in the population with attributes of the document selected as the best by the user. Objective heuristics consisted of coded measures, which would help guide the evolutionary process towards documents with proper style and design guidelines, but which also reflected the user's preferences.

Harrington et al. present a set of aesthetic measures for automated document layout, which we use as criteria for the objective fitness evaluation [7]. We evaluate the use of white space, the degree of overlap in the shapes, and the spatial balance. To evaluate spatial balance, first we bisect the page vertically, computing how much area is covered in the left half versus the right half. This gives us a measure of vertical balance. Next we bisect the page horizontally, computing how much area is covered in the top half versus the bottom half, giving us a measure of horizontal balance. We combine horizontal and vertical balance to give us an objective measure of spatial balance of a document.

We evaluate white space by computing the area of every shape in a document, adding these areas together, and dividing this total by the area of the entire document. For example, if there are a lot of large shapes in a document, then the total sum of all shape areas will be close to the area of the entire document. The amount of overlap is computed by determining whether any two shapes overlap, and if so, figuring out the area of the overlap region. This is computed for every shape in the document. This objective measure enforces a low degree of overlap between shapes. Finally, the three objective criteria are averaged together.

What if a user likes documents which conflict with any of the objective criteria? While the objective criteria are meant to allow a user to create documents which adhere to proper style guidelines for documents, a user might want to create a document which does not follow all of the objective criteria. A user might want or need to create a document in which a high degree of overlap is desired. This is where the subjective criteria comes into play. The subjective and objective criteria, which might be contradictory at times, are optimized using Pareto optimality.

3.2 Generational Algorithm

The interaction between the user and the GA is key in IGAs. Instead of asking the user to evaluate all individuals in the IGA's large population (population size of 100), we select a subset of size nine from the large population, and display this to the user to be evaluated [11]. In case the user does not find any satisfactory individuals, the user has the option to scroll down on the user interface and view the rest of the population. We introduce a visible gap and a label of "Rest of Population" to make it clear to the user that viewing the rest of the population is optional, since having to view all individuals in the population over many generations can lead to user fatigue. We allow the user to guide the evolutionary process by selecting the best individual in the population, with the user having the choice of picking from the subset, or by selecting any other individual in the population.

The selection of the best individual is used to interpolate the fitness of every other individual in the population (for further details see [11, 2]). However, the manner in which offspring are generated from the population can have drastic effects in the behavior of the IGA, which can range from many diverse individuals to rapid convergence to individuals similar to the user's selected best. We explore three generational algorithms. The first one used was the Non-dominated Sorted multi-objective Genetic Algorithm, abbreviated as NSGA-II [4]. The NSGA-II creates fronts of non-dominated individuals, where within a front none of the individuals are any worse than any other individual across all optimization criteria. All individuals within a front are said to have the same rank. We select parents by using the crowded distance tournament operator. We pick two individuals to participate in the tournament, and we select the individual with the higher rank to be part of the mating pool. In case the two individuals have the same rank, and consequently belong to the same front, then the crowded distance of both individuals is computed, and we select the individual with the highest crowded distance to be part of the mating pool. This translates to the individual being in a less crowded region of the front and hence, the crowded distance selection favors the most diverse individuals within a front.

The second generational algorithm was the same as NSGA-II, except that every offspring was generated by selecting an individual from the population and mating it with the user selected best. The third generational algorithm generates offspring by applying mutations to the user selected best individual.

3.3 Customizing

In order to allow users to be able to create a document from the tool, we added support for customization of the evolved document designs. Any document can be edited by the user, however in the current version we do not allow changes made by the user to a document to be inserted back into the IGA population. Customizing opens a new window, where the document is displayed on a larger scale, allowing the user to appreciate detail and to facilitate editing. We support for the user to move any of the shapes around, to add text to one of the existing shapes, to overlay an image on a shape, to resize a shape, to add free floating textboxes and images (not attached to one of the shapes), and to delete any shape, text, or image. The user can also change either the color of any individual shape, or one of the predefined color schemes can be selected. If a color scheme is selected, the colors in the scheme are assigned sequentially to the shapes in the document. While this functionality is limited compared to the full breadth of options that come with tools such as Photoshop or a word processor, it is meant to allow users to be able to visualize how content would look on one of the evolved documents. After customization, the user also has the option to save the current document as an image.

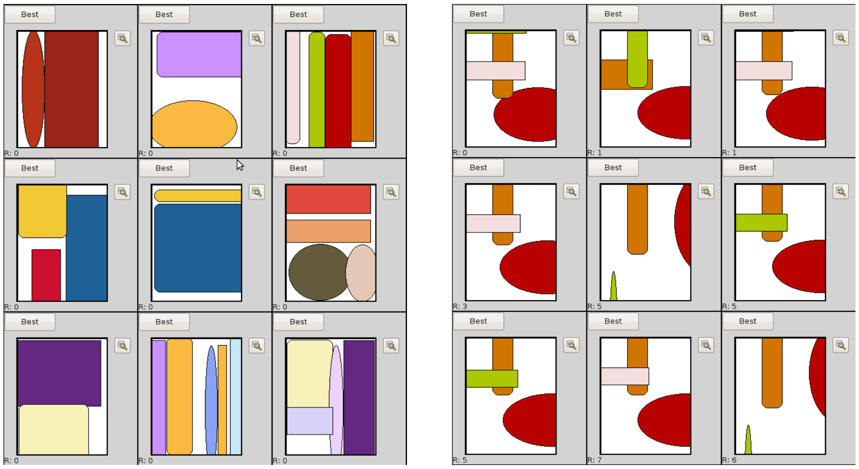
4 Results

Figure 2 shows a sample snapshot of the user interface at the beginning of the process (generation 0) and after 10 generations of user input. The IGA process begins with documents created randomly, as shown on the left side of the figure. Through feedback, the user can steer the evolutionary process in various ways, ranging from documents where all shapes (which stand for place-holders) are circular and overlapping, to documents with a small degree of overlap and lots of white space. As the IGA population converges (with the convergence rate depending on the generational algorithm) the focus of the search changes to fine-tunning, with documents varying by a small degree as seen on the right side of the figure. Some other examples of brochures created with our IGA tool are shown in Figure 3.

As an initial pretest, we had three participants test the tool. The participants were given the task of creating a brochure which advertised a new minor in interactive digital games to be offered by the computer science department. The requirements given to the participants were that the brochure was to include the following elements: (1) a header; (2) at least one paragraph, allowing for text to be distributed over the brochure as desired; and (3) at least two images. We interviewed the participants afterwards to get their thoughts on the usability of the system.

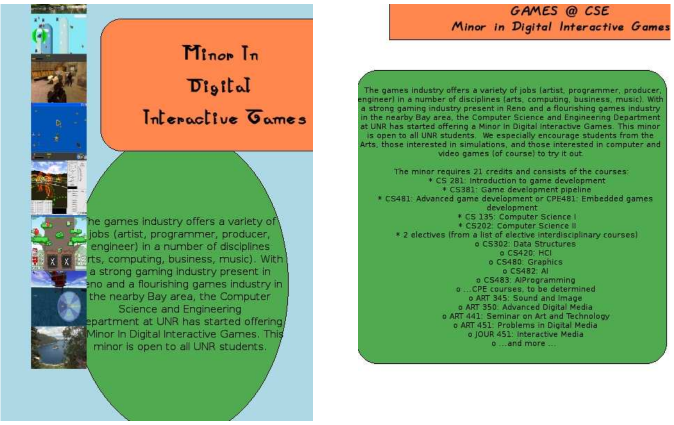
Due to the small sample, we cannot generalize the feedback given, but it allowed us to find areas that needed further work and had to be addressed before a full set of user studies. With regards to the generational algorithm, the participants preferred the two generational algorithms based on the NSGA-II. The participants found the generational algorithm which used mutations on the user selected best individual to be too sporadic, with too many random changes introduced from generation to generation. We attribute this to having used too high a mutation rate. However, the dilemma is finding the right mutation rate which would make this generational algorithm intuitive to users, since a too low of a mutation rate would not allow the user to effectively explore document designs.

All three users were able to create brochures which met the given requirements as shown in Figure 4. The users liked the ability to explore alternative designs, but found the interaction with the IGA to be too limiting at times. For example, the users could not modify a document and subsequently inject it back into the population. There were many times when a document design was close to the one desired by the participants, but through the picking the participants were not able to fine tune it to what they desired.



(a) Generation 0 (b) Generation 10

Fig. 2: Brochure templates as displayed to the user for evaluation.



Minor In
Digital
Interactive Games

he games industry offers a variety of jobs (artist, programmer, producer, engineer) in a number of disciplines rts, computing, business, music). With a strong gaming industry present in eno and a flourishing games industry in the nearby Bay area, the Computer Science and Engineering epartment at UNR has started offering Minor In Digital Interactive Games. This minor is open to all UNR students.

(a) Brochure 1

GAMES @ CSE
Minor in Digital Interactive Games

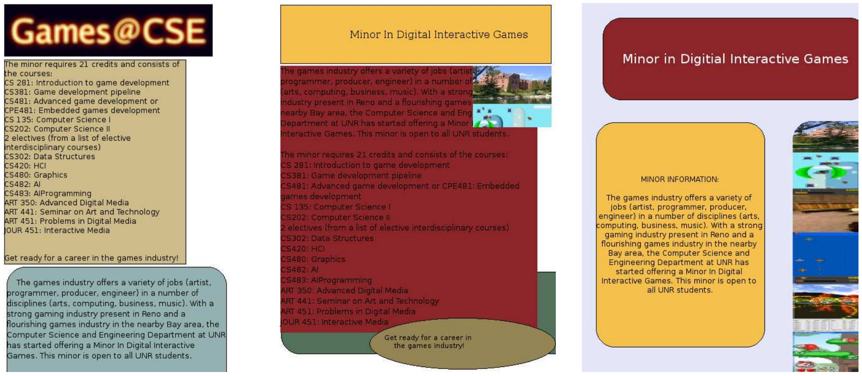
The games industry offers a variety of jobs (artist, programmer, producer, engineer) in a number of disciplines (arts, computing, business, music). With a strong gaming industry present in Reno and a flourishing games industry in the nearby Bay area, the Computer Science and Engineering Department at UNR has started offering a Minor In Digital Interactive Games. This minor is open to all UNR students. We especially encourage students from the Arts, those interested in simulations, and those interested in computer and video games (of course) to try it out.

The minor requires 21 credits and consists of the courses:
* CS 281: Introduction to game development
* CS381: Game development pipeline
* CS481: Advanced game development or CPE481: Embedded games development
* CS 135: Computer Science I
* CS202: Computer Science II
* 2 electives (from a list of elective interdisciplinary courses)
o CS302: Data Structures
o CS420: HCI
o CS480: Graphics
o CS482: AI
o CS483: AIProgramming
o ...CPE courses, to be determined
o ART 345: Sound and Image
o ART 350: Advanced Digital Media
o ART 441: Seminar on Art and Technology
o ART 451: Problems in Digital Media
o JOUR 451: Interactive Media
o ...and more ...

(b) Brochure 2

Fig. 3: Sample brochures.

All three participants were asked whether they believed it would have been faster for them to create the brochure from scratch instead of using the tool, and all three



Games@CSE

The minor requires 21 credits and consists of
the courses:
CS 281: Introduction to game development
CS381: Game development pipeline
CS481: Advanced game development or
CPE481: Embedded games development
CS 135: Computer Science I
CS202: Computer Science II
2 electives (from a list of elective
interdisciplinary courses)
CS302: Data Structures
CS420: HCI
CS480: Graphics
CS482: AI
CS483: AIProgramming
ART 350: Advanced Digital Media
ART 441: Seminar on Art and Technology
ART 451: Problems in Digital Media
JOUR 451: Interactive Media

Get ready for a career in the games industry!

The games industry offers a variety of jobs (artist, programmer, producer, engineer) in a number of disciplines (arts, computing, business, music). With a strong gaming industry present in Reno and a flourishing games industry in the nearby Bay area, the Computer Science and Engineering Department at UNR has started offering a Minor In Digital Interactive Games. This minor is open to all UNR students.

(a) User 1

Minor In Digital Interactive Games

The games industry offers a variety of jobs (artist
programmer, producer, engineer) in a number of
(arts, computing, business, music). With a strong
industry present in Reno and a flourishing games
nearby Bay area, the Computer Science and Eng
Department at UNR has started offering a Minor
Interactive Games. This minor is open to all UNR students.

The minor requires 21 credits and consists of the courses:
CS 281: Introduction to game development
CS381: Game development pipeline
CS481: Advanced game development or CPE481: Embedded
games development
CS 135: Computer Science I
CS202: Computer Science II
2 electives (from a list of elective interdisciplinary courses)
CS302: Data Structures
CS420: HCI
CS480: Graphics
CS482: AI
CS483: AIProgramming
ART 350: Advanced Digital Media
ART 441: Seminar on Art and Technology
ART 451: Problems in Digital Media
JOUR 451: Interactive Media

Get ready for a career in
the games industry!

(b) User 2

Minor in Digitial Interactive Games

MINOR INFORMATION:

The games industry offers a variety of jobs (artist, programmer, producer, engineer) in a number of disciplines (arts, computing, business, music). With a strong gaming industry present in Reno and a flourishing games industry in the nearby Bay area, the Computer Science and Engineering Department at UNR has started offering a Minor In Digital Interactive Games. This minor is open to all UNR students.

(c) User 3

Fig. 4: Brochures created by users.

agreed. Such a statement is understandable given the current limitations in the systems' functionality. However, the system is meant as more than a simple document creator, and all three participants recognized that and expressed positive feedback in the ability to view and assess many document designs in a matter of minutes, which helped them build a conceptual model of the brochure they wanted. A user with a clear set of goals, and who has a clear conceptual model of what he/she wants, might benefit from just using a standard design tool such as a word processor or a drawing system to create a document. However, when the requirements are open-ended, and if the user has to create a conceptual model for the given requirements, then the concept of this tool would be useful. The IGA would allow the user to explore many options, exposing the user to many possibilities which the user would not have come up with on his/her own, especially in a short period of time.

While evolutionary computation is a powerful technique, the interaction with a user presents many challenges and limitations. In the next section we propose that the main challenges faced by researchers in creating IGA tools for end-users stem from affordance issues.

5 Affordance of IGAs

One of the challenges faced when designing an IGA is affordance. Affordance, as discussed by Don Norman in his book The Design of Everyday Things, means that the designer must make "appropriate actions perceptible and inappropriate ones invisible" [9]. It is key for the designers to provide a conceptual model of the system, otherwise users end up making up their own conceptual model, which is bound to be erroneous. For example, a properly designed door knob lends its affordance to turn, just like a button presents the affordance of pushing.

We face such a dilemma when designing an IGA, especially an IGA not to be used by experts, but by end-users to create documents or brochures. If a user was asked to create a document, specifically a brochure for a new major or minor being offered by a department at a university, it would be unrealistic to expect the user to create such a brochure in a matter of minutes, even half an hour would be an unreasonable amount of time. However, spending 30 minutes in front of an IGA doing evaluation picks gets boring and frustrating quickly. Most IGA research places an implicit blame on the user, suffering from fatigue, where instead a tireless computer would be able to find satisfactory solutions by participating in evolution over many generations and evaluating hundreds of individuals. The problem is that most IGA applications suffer from conceptual models targeted to expert users, and in some cases the conceptual model makes sense only to the researchers who made the particular IGA application.

The participants in our pretest faced some of these issues while using our IGA. They found the interaction with the IGA (picking only the best document) to be limiting. However, this was done in order to mitigate user fatigue, since previous work has shown that too much feedback results in user fatigue [13]. Another challenge is conducting the IGA session in a manner which allows the user to build a correct conceptual model of the IGA. To an end-user, the concept and understanding of the workings of a GA should not be integral to understanding the system. However, the user must be able to develop an intuition of what is occurring from generation to generation. This introduces a final road-block, which is that if an IGA is failing to work properly, it is hard for the end-user to realize this. An error on a crossover or a mutation operator could introduce enough error in the overall process, resulting in either premature convergence or convergence to local optima. An end-user could be led to believe that he/she is not picking the right solutions, or that the problem is "too hard" and cannot be solved using evolution. IGA applications suffer from some or all of these challenges, and it is critical for the deployment of IGA based tools to end-users to address these issues, in order to fully embrace the computational power and benefits of IGAs.

6 Conclusions and Future Work

We have presented an approach to document design based on human guided evolution. Users guide the creation of brochure documents by picking the brochure they like the best from a subset displayed from a large population size. The IGA combines the user feedback with objective criteria taken from document design guidelines. We had three users successfully create brochure documents which met a set of given requirements, and with each brochure differing based on each of the users' preferences. We believe that a limiting factor of IGAs in general is affordance issues.

Interactive evolutionary design of documents is a promising direction of future work. First, we would like to conduct user studies in order to assess the utility of the tool. For example, we want to test whether participants are indeed capable of creating documents from scratch faster than by using our IGA tool. We also plan to determine whether the quality of the documents created with our IGA tool is greater than those created from scratch by the participants, using evaluation criteria from the Creative Product Semantic Scale [1].

Finally, we are also interested in supporting the exploration of alternative designs of an already created document by a user. Instead of starting the evolutionary process with documents created from scratch, the user would create a simple document, and then load this into the IGA. Thus, we can use this initial document as the seed from which to start the evolutionary process, allowing the user to search alternatives in fewer generations, and enabling him/her to visualize the desired content on the brochures, instead of evolving placeholders as in the current version. Another alternative would be to allow the user to specify the elements which must be included in the document, such as a set of images and text blocks. The IGA could then work on layouts and transformations on the elements specified by the user.

Acknowledgements We thank the study participants for their time. This work was supported in part by contract number N00014-0301-0104 from the Office of Naval Research and the National Science Foundation under Grant no. 0447416.

References

1. Andrews, J., Smith, D.C.: In search of the marketing imagination: Factors affecting the creativity of marketing programs for mature products. Journal of Marketing Research **33**, 174–187 (1996)
2. Banerjee, A., Quiroz, J.C., Louis, S.J.: A Model of Creative Design Using Collaborative Interactive Genetic Algorithms, pp. 397–416. Springer (2008)
3. Bentley, P.J., Wakefield, J.P.: Conceptual evolutionary design by genetic algorithms. Engineering Design and Automation Journal **3**, 119–131 (1997)
4. Deb, K.: Multi-Objective Optimization Using Evolutionary Algorithms. John Wiley and Sons (2001)
5. Geigel, J., Loui, A.: Using genetic algorithms for album page layouts. IEEE Multimedia **10**, 16–27 (2003)
6. Goldberg, D.E.: Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley Professional (1989)
7. Harrington, S.J., Naveda, J.F., Jones, R.P., Roetling, P., Thakkar, N.: Aesthetic measures for automated document layout. In: Proceedings of the 2004 ACM symposium on Document engineering, pp. 109–111. ACM, Milwaukee, Wisconsin, USA (2004)
8. Llorà, X., Sastry, K., Goldberg, D.E., Gupta, A., Lakshmi, L.: Combating user fatigue in iGAs: partial ordering, support vector machines, and synthetic fitness. pp. 1363–1370. ACM, Washington DC, USA (2005)
9. Norman, D.A.: The Design of Everyday Things. Basic Books (2002)
10. Quiroz, J.C., Banerjee, A., Louis, S.J.: IGAP: interactive genetic algorithm peer to peer. pp. 1719–1720. ACM, Atlanta, GA, USA (2008)
11. Quiroz, J.C., Louis, S.J., Dascalu, S.M.: Interactive evolution of XUL user interfaces. pp. 2151–2158. ACM Press, London, England (2007)
12. Renner, G.: Genetic algorithms in CAD. Computer-Aided Design **35**, 707–708 (2003)
13. Takagi, H.: Interactive evolutionary computation: fusion of the capabilities of EC optimization and human evaluation. Proceedings of the IEEE **89**, 1275–1296 (2001)