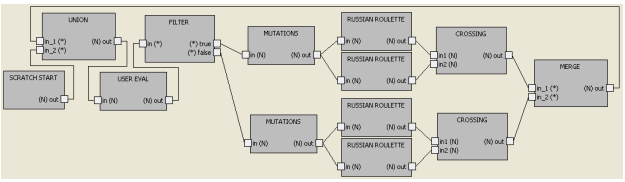
Magellan, an Evolutionary System to Foster User Interface Design Creativity

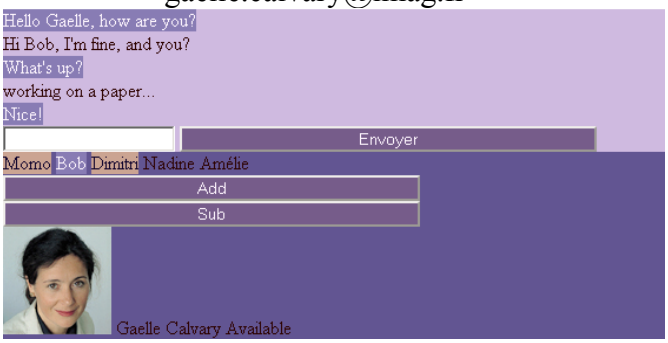
Dimitri Masson, Alexandre Demeure, Gaelle Calvary

Magellan, an Evolutionary System to Foster User Interface Design Creativity

Dimitri Masson, Alexandre Demeure
Laboratory of Informatics of Grenoble
655, Avenue de l'Europe
38330 Montbonnot-Saint-Martin, France
+33 (0)4 76 51 48 54
firstname.lastname@inrialpes.fr

Gaelle Calvary
Laboratory of Informatics of Grenoble
385, rue de la Bibliothèque - B.P. 53 - 38041
Grenoble Cedex 9, France
+33 (0)4 76 51 48 54
gaelle.calvary@imag.fr





ABSTRACT
Fostering creativity in User Interface (UI) design is challenging for innovation. This paper explores the combination of model-based approaches and interactive genetic algorithms to foster the exploration of the design space. A user task model is given in input. Magellan produces sketches of UIs that aim at inspiring the designer. Later on, appropriate tools may be used to tune the right design into the design right. Magellan is a proof of concept that deserves further exploration. Currently it is implemented using COMETs but it is not dependent of this technology.

Keywords
Model-based User Interface design, interactive genetic algorithm, creativity, Magellan.

ACM Classification Keywords
H.5.2 [**Information interfaces and presentation**]: User interfaces – *prototyping*.

INTRODUCTION
"Getting the right design" and "getting the design right" [16] are complementary challenges in Human Computer Interaction (HCI). Whilst most of research focuses on how "getting the design right", this paper aims at "getting the right design" by fostering creativity. The need of a support for creativity is crucial in model-based approaches. Transformations are complex to specify, as a result discouraging designers from defining new ones. This limits the exploration of the design space.

We explore how Interactive Genetic Algorithms (IGA) can support the generation of transformations for a given task model. Our goal is not to provide the final User Interface (UI) but to generate "sketches" for inspiring the designer. Appropriate tools have then to be used to "get the design right".

This paper presents related works first. It then describes the core principles of our approach. These principles have been implemented into Magellan, an early prototype.

RELATED WORKS
Model-based approaches support the generation of Concrete UIs (CUI) from a given task model and for a given context of use [2, 2, 6]. They ensure that the generated CUIs are compliant with the task model and make it possible to automatically maintain consistency between the two models (task and CUI). Some tools support the edition of CUIs by designers to explore alternatives [7], but they do not foster creativity.

Genetic Algorithms (GA) is a search technique used to find exact or approximate solutions. They were formalized in 1975 by Holland [8]. GA deal with a population of candidate solutions for a given problem. Each solution is encoded within a sequence of genes. GA are made of four main steps. First, an initial population is generated. Then, iteratively, each candidate solution gets a score that evaluates its fitness with regard to the problem (step 2). Then, some solutions are selected (step 3) and reused to generate a new population (step 4). The generation consists in applying operations on solutions (e.g., mutation,

EICS'10, June 19–23, 2010, Berlin, Germany.

inbreeding). Sims [14] applied GA to artistic creation and replaced automatic evaluation with human evaluation. This approach is referred to as Interactive GA (IGA) [15]. The main drawback of IGA is the user fatigue induced by the selection of the best individuals over generations. Usually, users can't go beyond 20 generations of 16 individuals [15].

In 2008, IGA provided outstanding results for image creation as demonstrated in Picbreeder [13]. The success of Picbreeder relies on two mainstays: storage and complexification. Keeping trace of generated individuals (i.e. automata that produce images) is an efficient means to tackle the user fatigue and to share individuals among a community. Individuals can be complexified along the IGA process: it is possible to obtain complex shapes (e.g., faces, car side views) from individuals that barely generate circles. This work motivates the exploration of IGA for UI generation.

In HCI, IGA have been explored by Monmarché to generate HTML web pages. In these works, individuals represent either Cascading Style Sheets (CSS) [10] or CSS plus webpage layout [11]. The population is composed of 12 individuals. Each of them is used to generate a web page. The set of web pages is scaled to fit on a single screen. Individuals can be edited by designers for saving time (e.g., colors customization). At each IGA iteration, designers select best candidates among the 12 web pages. These works suffer from three main limits. First the selectors that identify HTML elements on which to apply CSS transformations are predefined (e.g., titles of level 1, paragraphs, images). It is impossible to generate new groupings along the IGA process (e.g., even paragraphs and images). In addition, there is no high level description (e.g., no task model) which the IGA could rely on to generate more complex transformations (e.g., replace an entry text with a calendar). Last, individuals can not be complexified along the IGA process. Only parameters can be tuned, but no new parameter can be added (e.g., no insertion of texts or images to enhance existing elements).

Quiroz explores IGA to generate XUL UIs [12]. He uses wider populations (hundreds of individuals) but only presents a subset of the population to designers. The designers have to select the best and the worst individuals. Then, an interpolation/extrapolation algorithm is applied to the rest of the population to automatically evaluate individuals. Although at a first glance this approach seems to be promising with regard to the user fatigue, it raises a fundamental question that is not answered by the author so far: can a UI be automatically compared with "bad" and "good" samples? In [12], the evaluation is based on a basic criteria only: the UI blueness. As automatic evaluation is not solved so far, we prefer not to base our proposition on a so strong hypothesis. In addition, the layout is simplistic in [12] and it is not possible to replace interactors. Last, there is no complexification of individuals along the IGA process.

In conclusion, IGA seem to be powerful for improving designer creativity. We believe that IGA coupled with model-based approaches can be of a great interest in HCI for facilitating the design space exploration. Model-based approaches may also make it possible to support complexification (e.g., adding an image to improve an input field guidance).

CORE PRINCIPLES

The architecture of Magellan is roughly presented in Figure 1. Magellan takes a task model in input. It then processes it using a customizable flowchart of GA operations to create and make evolve a population of transformations. Transformations are applied to the task model to generate CUIs. We assume that the task model improves transformations. For instance, if the system knows that an input field corresponds to a "select date" task, it will be able to replace it with a calendar along a mutation. To support the process, we rely on an external database (Figure 1) that explicits the interactors that can be used to represent a task.

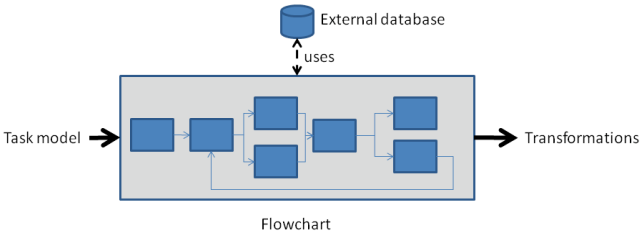


Figure 1: Overview of the Magellan architecture.

The individuals used in the IGA are tailored transformations that overcome the limitations identified in the state of the art. First, the selectors are not predefined. They can evolve. Second, thanks to the external database, it is possible to render a task using different interactors. Last, transformations can be complexified, resulting in more sophisticated UIs.

For flexibility, Magellan is based on a modular, fully customizable, flowchart architecture. We make it possible for designers to build their own IGA. We propose a toolbox of five flowchart component types: the four classical operations in GA (i.e., initialization of a population, genetic operation, evaluation of individuals and selection of individuals) plus the flowchart control. Control components support the creation of branches (e.g., if then else), union, merging, etc. Such a workflow architecture enables designers to program different evolution processes such as convergent/divergent thinking [9].

EARLY PROTOTYPE

This section describes the implementation choices we made in Magellan. A simplified instant messenger interactive system serves as illustration.

Technological choices

Magellan is based on the COMET toolkit [4]. COMETs are polymorphic task level interactors that cover both tasks and task operators. COMETs are multi-rendering multi-

technological interactors (WIMP and post-WIMP, Web and non Web as well as vocal). The COMET architecture ensures that the renderings are consistent. Figure 2 presents the graph of COMETs of the case study. A top level interleaving (Instant messenger) gives access to three COMETs. The first one (Chat Gaelle-Bob) is an instance of a "Chat" COMET. Its "Logs" child contains the messages Gaelle and Bob exchanged. The "Send new message" child contains two COMETs: a text entry and a "Send" command. The "Manage contacts" COMET gives access to Gaelle's contacts (Bob, Momo, etc.) and makes it possible to add or delete contacts. Last, the "Manage profile" COMET enables the management of Gaelle's photo, name and status.

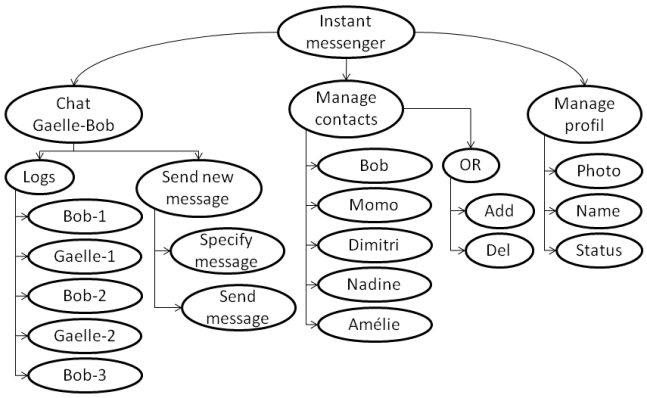


Figure 2: The graph of COMETs of the case study.

As presented in [5], we use a semantic network to implement the database that stores the possible presentations of a given task. The semantic network enables queries such as "Find a presentation for task *T* that is close to the *P* presentation". Such queries are useful in case of transformations mutations. For instance, the semantic network can inform Magellan that "*Choice of a date* tasks are represented with text entries" can be mutated into "*Choice of a date*" tasks are represented with a calendar. The transformation is updated accordingly.

The COMET toolkit comes with CSS++ as transformation language. A CSS++ transformation sheet is composed of a set of rules. Each rule is composed of two elements: a selector and a translator. The selector identifies the COMETs to be translated (e.g., all the top level interleavings). The translator specifies the way the selected COMETs are translated (e.g., set the background to white, display interactors as tabbed panes). The interactors choice for representing a given task can rely on the semantic network.

Magellan user interface

The UI of Magellan is divided into two pieces: a workflow editor (Figure 3) dedicated to users who need to tune the IGA, and a web based UI (Figure 4) for exploring the design space based on the current IGA. The workflow editor is composed of three parts: at the center, a canvas is dedicated to build the workflow. Components can be placed, removed and connected. The active components are highlighted. On the left, a list makes it possible to drag and drop evolution components onto the canvas. The three top buttons (Reset, Play/Pause and Step) control the flow.

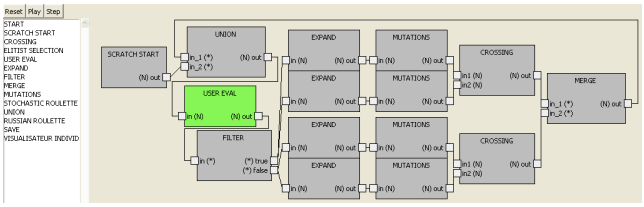


Figure 3: The Magellan workflow editor.

Figure 4 shows the web based UI of the user evaluator component. The three top left buttons (Reset, Play/Pause and Step) control the flow. The "Validate scores and continue" button enables to validate user choices and to go to the next step. The table of images represents the UIs produced by evaluated individuals (i.e., transformations). By clicking on images, the user selects the ones he/she prefers.

Figure 4: The web based UI of Magellan.

Magellan IGA

We first describe the representation we chose for individuals. Then we describe the Magellan components: initialization of the population, genetic operations, evaluation, selection and flow control.

Individuals

Individuals encode a set of transformations. Each transformation specifies a part of the UI. Transformations are encoded in a tree. Nodes (respectively edges) represent translators (respectively selectors). By default, the root node of each individual is associated with the root of the COMETs graph. To ensure visual consistency, we decorate individuals with a global color theme (i.e., a palette of five colors C1 to C5). For instance, in Figure 5, the background of the root is colored with C1 (the first color of the palette).

The selector of each edge applies to the set of elements selected in its source node. For instance, the edge between

the root and the node A selects the containers that are children of the root COMET. With respect to Figure 2 the COMET associated to the node A are "Manage contacts" and "Manage profile". They are translated into accordion containers.

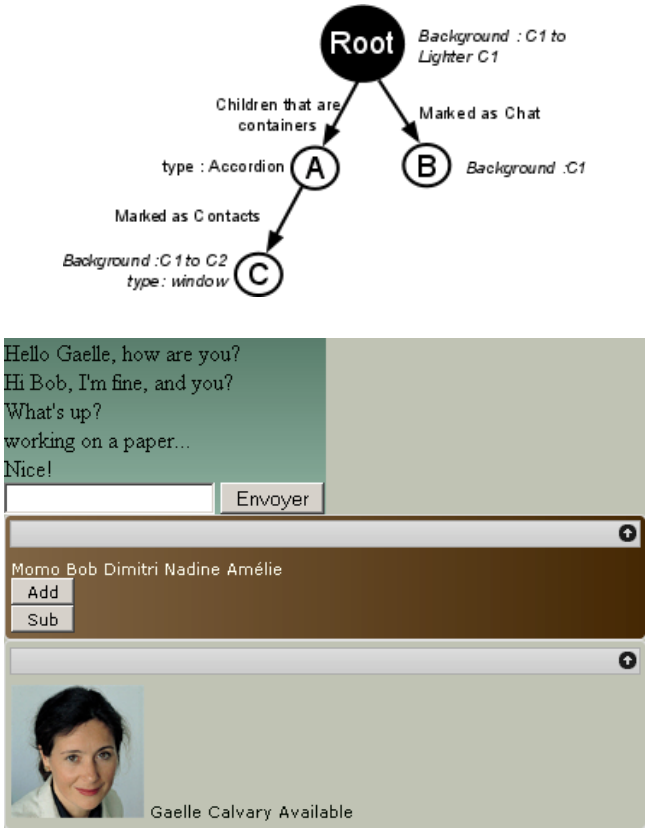


Figure 5: Example of an individual and the resulting UI.

Initialization has to deal with two concerns: the size of the population, and the generation mechanism. The size of the population is crucial: if too small, there is a risk to converge towards "bad" or undesired solutions; if too large, evaluation of solution becomes time consuming. Picbreeder [13] shows that it is possible to obtain good results with 16 elements. In Magellan, the size of the population is one parameter.

We envision several initialization mechanisms. We briefly describe them below and indicate whether they have been implemented or not.

- Random generation (implemented): Individuals are randomly generated by successive mutations of a default tree.
- Manual specification (implemented): Transformation trees can either be manually produced or imported from previous uses of Magellan. This is useful for supporting scenarios such as importing external transformations from other tools as well as collaborative work [1].
- Guidance based on existing UIs (planned): Designers could guide the system by providing UIs or UI sketches. This is useful to explore design options that are close to the given one in terms of mutations.

Genetic operations

Genetic operations are means to modify individuals. We describe the ones we implemented or planned to implement in Magellan.

- Mutation (implemented): Mutations can be applied at three levels: nodes, edges and the tree structure. The mutations are parameterized with the former value to control the evolution degree. Nodes mutations modify the COMET presentations. They can be applied to attributes (e.g., background color, layout, width, height, etc.). In the case of a color mutation, Magellan ensures that the contrast between the foreground and background colors remains high enough. Colors mutations can modify the color theme as well as a color of the theme by lightening or darkening it. Nodes mutations can also change the presentation of the COMET. For instance, in Figure 5, the "Type: Accordion" in node A can be mutated into a "Type: Frame". Mutations can also compose presentations to enhance the UI. In Figure 6 the sentences sent by Gaelle are composed of her photo and a label displaying the sentence. Composition of COMETs presentations is a powerful means to add complexity to individuals (Figure 6).
  Edges mutations modify the related selectors. The modification can lead to a broader selection (e.g., select children is generalized into select all descendants), a more narrow one (e.g., select all descendants is restricted to select children) or a completely different one (e.g., select children becomes select descendant containers). By doing so, we avoid the predefinition of selectors, unlike Monmarché [10] and Quiroz [12].
  Tree structure mutations add or remove nodes and related edges. By doing so, it is possible to complexify the individuals and thus the transformations to be applied to the COMETs graph.

Figure 6: Example of mutation: the photo stored in the user profile is added to each sentence.

- Inbreeding (implemented): There are several ways to achieve inbreeding between two (or even more)

individuals. So far we have implemented a mechanism which randomly selects branches of each parent to create a new individual. Another option for the future would be a guidance-based inbreeding. The idea is to promote interesting branches from each parent. The selection can be based on several criteria. For instance, the designer elicits the interesting parts of the resulting UI during the evaluation. The corresponding transformations are marked as interesting.

- Edition (planned): We plan to make it possible for the designer to directly edit transformations. This could be done in different ways: by editing transformations trees directly, or the resulting CUIs and then inferring the corresponding transformations. Being able to edit transformations would favor a quicker convergence. For instance, along the evolution process, the designer could discover which background color is the most suitable and fix it directly.
- Local optimization (planned): We plan to add some local optimization components to Magellan. At some point, it can be interesting to apply algorithms such as layout optimization [6]. The risk is to skip solutions that are locally not optimal but would be later on. In Magellan, the modular architecture puts the decision under the control of the designer.

Evaluation of individuals

In Magellan, the evaluation relies on the designer. The designer scores the UIs that are specified by the individuals. There are several ways to score candidates. The simplest scale is a binary notation, as done in Picbreeder [13]. An extension of this notation is to attribute scores on a wider range (e.g., 1 to 10, or a real between 0 and 1). These two solutions are modeled with evaluation components in Magellan. It is also possible to mark some parts of the UI as interesting. We plan to explore other evaluation approaches such as tournament evaluations. Tournaments consist in comparing series of two candidates, and building a partial order between solutions.

In addition to human evaluation, we believe it is useful to provide designers with some automatically computed scores. Computation can be based on ergonomic and performance criteria like it is done in SUPPLE [6].

Selection of individuals

Several selection methods exist in the literature. One can cite rank selection (best candidates are selected) or roulette wheel selection (selection probability proportional to the score). Elitist strategies may be applied. They consist in keeping the best individuals unchanged through generations in order to avoid regression. Resurrection enables the designer to recover an ancestor candidate and re-inject it into the current generation.

Control of the workflow

The workflow editor (Figure 3) makes it possible for designers to add and link components. Populations of individuals are conveyed along the links. We propose several components to control the workflow: conditional components such as filters and loop structures for building sophisticated IGA; shufflers and population controllers components to respectively randomize the order of individuals and to increase or decrease the population size; merge components to concatenate individuals from different populations into a single one. Shufflers and population controllers are particularly useful before inbreeding sources that are either identical or of a different size.

We propose two execution modes of an IGA: 1) a step-by-step mode for a precise control over the workflow ; 2) a run mode for going through all non interactive components, and stop only when user evaluation is needed.

We plan to support the control of the evolution process at a finer granularity. Parts of the UIs can be close to the designer expectations. As a result the designer may appreciate to stop or decrease the evolution process for these parts whilst increasing it in other parts. For instance, he/she would like to let a menu bar evolve whilst keeping the rest of the UI unchanged. In other words, the evolution process may be locally tuned. So far Magellan makes it possible to mark some parts of the UI as interesting. The corresponding branches of the individual are promoted during the inbreeding and are less subject to mutations.

CONCLUSION AND FUTURE WORK

This paper addresses the need of a support for stimulating creativity. Model-based approaches provide a powerful support for saving development and maintenance costs but they somehow kill creativity. Magellan explores the combination of model-based approaches and interactive genetic algorithms to tackle the problem while keeping benefit of models. Figure 7 and Figure 8 show UIs evolved from a task model describing an instant messenger. Figure 7 highlights the potential of the selector evolution whilst Figure 8 illustrates the interactors replacement. We believe that such mutations are key for future works.

Figure 7: Example of UI where chat sentences are differentiated by color depending on the speaker.

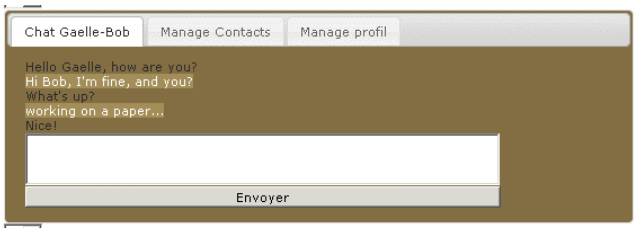


Figure 8: Example of UI where the top level interleaving has been represented using a tabbed panel.

The modular architecture of Magellan enables the designer to define his own IGA process. Up to now, additional components such as the manual edition and the local optimization are to be implemented. Magellan is an early prototype that serves as a technical proof of concept. We are yet setting up an evaluation study to validate the approach. In the future, we plan to use the approach to facilitate comparative evaluation [6] by generating mutants from existing designs.

REFERENCES

1. Banerjee, A., Quiroz, J., and Sushil, J.L. A Model of Creative Design Using Collaborative Interactive Genetic Algorithms. In *Design Computing and Cognition'08*, pp 397–416.
2. Berti, S., Correani, F., Mori, G., Paterno, F., and Santoro, C. Teresa: a transformation-based environment for designing and developing multi-device interfaces. In *CHI'04* (New York, NY, USA, 2004), ACM, pp. 793–794.
3. Calvary, G., Coutaz J., Thevenin, D., Limbourg, Q., Bouillon, L., and Vanderdonckt, J. A Unifying Reference Framework for Multi-Target User Interfaces. In *Interacting With Computers*, Vol. 15/3, pp 289-308, 2003.
4. Demeure, A., Calvary, G., and Coninx, K. COMET(s), A Software Architecture Style and an Interactors Toolkit for Plastic User Interfaces. In *Interactive Systems. Design, Specification, and Verification, 15th International Workshop, DSV-IS 2008*, T.C.N. Graham & P. Palanque (Eds), Lecture Notes in Computer Science 5136, Springer Berlin / Heidelberg, Kingston, Canada, July 16-18, 2008, pp 225-237.
5. Demeure, A., Calvary, G., Coutaz, J., and Vanderdonckt, J. The Comets Inspector: Towards Run Time Plasticity Control based on a Semantic Network. In *Fifth International Workshop on Task Models and Diagrams for UI design (TAMODIA'06)*, Hasselt, Belgium, October 23-24, 2006, pp 324-338.
6. Gajos, K., and Weld, D.S. Supple: automatically generating user interfaces. In *IUI '04: Proceedings of the 9th international conference on Intelligent user interface* (New York, NY, USA, 2004), ACM Press, pp. 93–100.
7. Gajos, K., and Weld, D.S. Preference elicitation for interface optimization. In *Proceedings of UIST 2005*, ACM Press, pp. 173–182.
8. Holland, J.H. Adaptation in natural and artificial systems, Ann Arbor, MI: University of Michigan Press, 1975.
9. Kelly, J., Papalambros, P.Y., and Seifert, C.M. Interactive genetic algorithms for use as creativity enhancement tools, *AAAI Spring Symposium on Creativity, Palo Alto, CA,* 2008.
10. Monmarché, N., Nocent, G., Slimane, M., Venturini, G., and Santini, P. Imagine: a tool for generating html style sheets with an interactive genetic algorithm based on genes frequencies. In *Proc. of IEEE Intl. Conf. on Systems, Man and Cybernetics* (1999), pp. 640–645.
11. Oliver, A., Monmarché, N., and Venturini, G. Interactive design of web sites with a genetic algorithm. In *Proceedings of the IADIS International Conference WWW/Internet* (Lisbon, Portugal, november 13-15 2002), pp. 355–362.
12. Quiroz, J.C., Dascalu, S.M., and Louis, S.J. Human guided evolution of xul user interfaces. In *CHI '07 extended abstracts on Human factors in computing systems* (New York, NY, USA, 2007), ACM, pp. 2621–2626.
13. Secretan, J., Beato, N., Ambrosio, D.B., Rodriguez, A., Cambell, A., and Stanley, K.O. Picbreeder: evolving pictures collaboratively online. In *CHI '08: Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems* (New York, NY, USA, 2008), ACM, pp. 1759–1768.
14. Sims, K. Artificial evolution for computer graphics. In *SIGGRAPH'91: Proceedings of the 18th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1991), ACM, pp. 319–328.
15. Takagi, H. Interactive evolutionary computation as humanized computational intelligence technology. In *Fuzzy Days* (2001), p. 1.
16. Tohidi, M., Buxton, W., Baecker, R., and Sellen, A. Getting the right design and the design right. In *CHI'06: Proceedings of the SIGCHI conference on Human Factors in computing systems* (New York, NY, USA, 2006), ACM, pp. 1243–1252.