Visual Narratives of Parametric Design History

Aha! Now I see how you did it!

Halil I. Erhan¹, Rodolfo Sanchez¹, Robert F. Woodbury¹, Volker Mueller², Makai Smith²
¹Simon Fraser University, BC Canada, ²Bentley Systems Inc., PA USA
¹{herhan, rsanchez , rw}@sfu.ca, ²{Volker.Mueller, Makai.Smith}@bentley.com

Abstract. Histories are underdeveloped and underused features in parametric CAD (PCAD) systems. Designers employ backtracking and deferral strategies that both use and augment history. Using backtracking and deferral as a framework we present two classes of design process graph diagramming techniques for augmented history in PCAD. We compare the second version of these design process graphs across several designers who completed multiple tasks using two parametric systems (SolidWorks and GenerativeComponents). The two systems show similar types of action, with markedly more and deeper backtracking in GenerativeComponents. We present a third diagramming system as a design for a proof of concept prototype. This prototype aims to expand the capabilities of design histories beyond access to single prior states to visualize and enable direct interaction based on backtracking and deferral.

Keywords. Parametric CAD; design history; backtracking; deferral; design space.

INTRODUCTION

History, the record of what actually happened in design, helps designers manage, and to a lesser extent, reflect on and understand work. When edited, histories cease being logs and become stories; narratives. Through these narratives, designers often tell stories of how an idea develops from inception to a satisfying solution (Simon, 1956). In parametric computer-aided design (PCAD) systems in particular, the narrative of work done (or interpreted) is often the best means available to explain how and why complex parametric links came to be. However, in current systems, histories are mostly logs and designers are usually frustrated by the system providing inadequate help in organizing, recording and editing the process (Woodbury and Burrow, 2006). Throughout this paper we use the term history to refer to all tools that provide the ability to record and replay actions. Our stance is more akin to the view of history taken by the humanities (history is always partial and relative) than to that of computer scientists (history is a log).

In this study, we aim to understand design action when designers use PCAD tools and to suggest potential solutions to support design by using histories. We envision an interactive model of the parametric design process that designers use actively for both understanding and explanation. The study has two goals. The first focuses on a means to identify the design patterns and strategies unique to PCAD. The second is to provide insight for system developers to design tools to view, interact and explore using the parametric model’s design history. It aims
to take advantage of the systems’ parametric capacity not just to model, but also to explore design space and explain choices, both made and foregone. These goals are related but different from earlier research that mainly focuses on ‘interaction’ analysis (such as Bhavnani and Bonnie, 2000), visual undo/redo mechanisms (such as Grossman, Matejka and Fitzmaurice, 2010), or state-based version control using interactive design histories (such as Bueno et al., 2011).

We report two studies. We focused on the strategies designers adopt when they revisit the actions they perform as they search for alternatives and refine their design. In the first study, we observed designers using SolidWorks (SW) [1] to develop solutions to two different design problems under controlled conditions. In the second study, we analyzed design transaction records of actual work from another parametric tool, namely GenerativeComponents (GC) [2], to learn more about what history could reveal to designers. Together these studies provide insights towards a model for a parametric process graph and an interactive system adapting the model. Our hope is that, by identifying the opportunities, achievements and failures described in the design narrative, better design solutions can be reached.

Below we describe our motivation for this study followed by a brief description of the literature. We present our research apparatus and the studies we conducted. We present several visual design process graphs and use these to propose a design for a parametric design history graph. We conclude with a general discussion and ideas for future work.

MOTIVATION
Histories can provide us with insights into our accomplishments and failings that can be used to guide later decisions. These insights arise not only through individual points in time but also through a narrative of the process leading to and from any of the high or low points in design. The core idea is that designers, like ‘time-travelers,’ should not be bound to working only on single and present design states, but also on potential states that were visited earlier or missed. For this, new tools are needed for design histories that support not only editing history, but also simultaneous interaction with multiple points in history and the iterative nature of design search (Akin, 2008). There are several reasons for an augmented design history. First, a navigator may help designers access design history to re-explore past design states and discover new ones; Second, edited narratives of the parametric design process can build explanations for private (internal) or public (external) use. Third, histories may provide new tools for managing the complex relations networks created with PCAD, which is an essential feature for creativity support tools (Shneiderman, 2007). Fourth, design histories provide researchers with new tools for understanding PCAD and how it supports design.

Current design-support systems generally do not provide augmented design history. We identify some reasons for this as follows:

• Limited functions to navigate, edit, and explore past design states beyond simple undo and file versioning.
• Limited feedback on the development of design work. The information is model- and state-focused and does not reveal what happens to the model.
• Limited action records that can provide insight on design. Some tools only provide information on model structure and partial process records through construction trees or transaction logs. Deferrals and revisions are implicit.
• Limited retrieval of discarded design states that are valuable to understand and explain the design process. Such states may exist in a record but are easily lost in the noise of many minor changes and states.

PCAD tools and models allow users to improvise design history management partially. They enable rapid exploration through parametric changes. Through backtracking, the designer can change and explore the design space on the basis that greater exploration will ultimately provide better design solutions and should therefore be encouraged to do so. This
potential is currently untapped and relies on user driven strategies to cope with the absence of inbuilt tools. We propose that higher-order actions such as deferral and backtracking can structure more useful design histories.

**STRATEGIES IN PCAD: BACKTRACKING AND DEFERRAL**

The literature presents two high-level and interrelated strategies designers adopt when working with complex design models. The first is ‘backtracking’ (Akers, 2009). Designers backtrack to return to a previous design state. The second is ‘deferral’ (Woodbury, 2010). Designers model objects and relations approximately, knowing they will refine these later. Deferral is strongly assisted by PCAD as model changes propagate downstream – greatly lowering the cost of deferral strategies.

Akers provides a taxonomy of reasons for backtracking related to software usability studies (Akers, 2009). This includes error recovery, exploring the interface, exploring design alternatives, revising temporary actions, understanding action consequences and reversing undesirable system actions. Our particular interest is on backtracking when exploring design alternatives and when revising temporary actions; designers create temporary model states that can later be deleted or edited once their purpose is exhausted.

Backtracking in PCAD goes beyond the working definition given by Akers that only includes undo and erase as signals for backtracking (Akers, 2009). A broader definition is required to map designer action in parametric modeling accurately. Designers trigger backtracking when they revisit a previously established parameter or feature in the model past state and make a change. These modifications include adding new features, suppressing or deleting features and simple undo. On the other hand, deferral in design is closely tied to the very nature of parametric modeling (Woodbury, 2010). With a deferral strategy, designers build or use representations that can admit changes to earlier decisions without much change on the representation. We believe that the reasons behind the deferral strategy are determined by at least four factors. The first two are deferral of parametric values and deferral of structural elements of the parametric system. The other two factors are the deferral of design decisions and the deferral of work.

**METHOD: DEVELOPING RESEARCH APPARATUS AND STUDIES**

We conducted two studies to understand the design behavior of designers using PCAD tools and to suggest potential tool solutions to support the design process using design history. Below we describe the process we followed, and insights gained from each study.

*Apparatus development: building the design process graph*

Before we conducted the studies, we developed a design process-graph scheme as an apparatus to visualize and analyze patterns of use in parametric design. We also identified signals that reveal backtracking and deferral. We achieved these by running a pilot study with one participant who was asked to design a bus stop and a beach changing room. The participant was a designer with advanced SW skills. The encoding of the pilot study is based on the participant’s feedback as well as the measures described by Akers (2009). It helped us refine the study design and encoding guidelines. We identified the following signals as measures to be used in encoding the data in both studies:

- **Undo:** Reversing the previous action performed
- **Delete:** Deleting parts of the CAD model or features
- **Add actions:** Inserting new geometric features to the model.
- **Modifying actions:** Editing existing feature properties.
- **Within- and between-states:** Actions that are executed and applied within the same state or in between different states.
The first graph-model: Using the insights from the pilot study, we created the first process graph by encoding video of designers to reconstruct the design history (Figure 1). It was an iterative process that consisted of researchers viewing, identifying and generating accounts of the sequential actions. Actions are marked as nodes, and undo, delete, add, modify actions were shown as arcs connecting from the node where an action is performed to the initial node where the object being edited is first introduced. Backtracking actions created new variations and alternatives shown as branches on the graph. The initial encoding was very detailed and captured more user actions than were necessary; adding noise to the data masking relevant actions. The initial graph was not included in further analysis except as a source to provide insight on the overall process.

On the graph, the actions shown as nodes are ordered from left to right. Alternatives (branches) are created when signaled by explicit user intention or observation of a “major” change to the model; and distinct symbols identify revisiting variables and deletions. In this process, the flow of control is managed by the designer and system as described in (Figure 2).

The first iteration was visually complex and hard to grasp. We identified three apparent sources. First, many local edits (direct or short indirect arcs) appear to be simple error correction (caused by either designer or system) that were corrected using undo or erase. Second, distinguishing operations by type (revisiting variables, deletions and structure changes) suppressed the overall picture of change. Hence, we decided to remove suspected undo actions and made all nodes and arcs of one type. Finally, the

Figure 1
First iteration of graph modeling action data with backtracking shown as arcs. The long backtracking arcs towards the end of the process suggest the deferral of decisions.

Figure 2
During any point in the design process (a) designers add new elements to the design (b). At this point the designer decides to change a value of a preceding element (c). The system updates all states downstream and takes the designer to the state where the change is initiated. The graph keeps the record of change as a backtracking arc.

Figure 3
The second iteration of the process graph includes the abstraction of designer actions into a defined set. Actions that build the model are shown in grey. Backtracking, in large orange nodes and backtracking arcs link both.
branching was not a good model of PCAD, primarily because of the downstream propagation of change. Backtracking became the most salient behavior captured in this version.

The second iteration of the graph-model: After studying actions, their relationships, visual representation and overall structure, we decided to include only add, subtract and modify actions in the graph. Other low-level actions such as UI-commands, error correction, zooming or changing display styles were excluded.

The second process graph scheme includes three discrete elements of parametric modeling: constructs, backtracking and design variations (Figure 3). Constructs are all the actions designers take within the tool to build the parametric model such as inserting a feature or creating a parametric relationship, or encapsulating a selection of low-level actions. In this graph, constructs are shown as grey nodes; backtracking nodes (larger orange nodes) are placed when designers backtrack and make changes to constructs, for example, by changing the value of a parameter or deleting a section of the model. With each backtracking node, a corresponding arc is created representing the relationship between the construct and its backtracking node. The third element is design variations, which are shown below the constructs as unfilled nodes. These are a record of the changes to two elements and the addition of one more in the design model.

```plaintext
transaction modelChange 'Add Floor_solid; change curve02, plane02' {
    node User.Objects.plane02 Bentley.GC.Features.Plane
    {
        GraphLocation = {807.6, 21.2};
    }
    node User.Objects.curve02 Bentley.GC.Features.Curve
    {
        GraphLocation = {1084.0, 39.0};
    }
    node User.Objects.Floor_solid Bentley.GC.Features.Solid
    {
        Technique = 'OffsetFromClosedCurve';
        ClosedCurveToOffset = curve02;
        OffsetAboveCurve = 0;
        OffsetBelowCurve = 1;
        AboveCurveDirection = baseCS.ZDirection;
        AutoHideInputs = true;
        Density = 1.0;
        GraphLocation = {1447.01055457457, -113.228175561609};
    }
}
```
Figure 6
Design process graphs of GC and SW designers using backtracking as metric. Actions encoded from five different real-world projects using GC, study 2 (Left), and actions from participants using SW that created different solutions for a beach changing room and bus stop in Study 1. Note: graphs are not at the same scale.
changes on the parameters made by the designer either immediately after establishing them or after backtracking. We built the second graph using the encoded data from one of the participants in Study 1 (see below). A partial view is shown in Figure 3.

In the second scheme, each design variation is equivalent to a branch. However, the resulting graph does not show branching as a tree graph as branches are implicit. Given sequential nodes A, B and C and design state α, an arc from C to A will create an implicit branch and changes will propagate downstream. The value of B and all subsequent nodes will depend on the new value of A and the parametric relationships associated with A creating design state β. Design state α no longer exists as a model state and has no explicit branch. In addition, we changed arcs from rectilinear to curved (in this instance circles) as in ThreadArcs (Kerr, 2003) as these disambiguate arc crossings and improve visual tracing in and out of nodes. The next refinement of the second graph shows part-editing spans under corresponding nodes as color-coded bars below the graph (Figure 4).

**Study 1: observing design moves using SW**
The first study consisted of 16 participants who were asked to complete two separate design tasks in the course of approximately two hours using SW. Participants for the study were upper-division undergraduate, graduate and former students from SIAT, Simon Fraser University. All undergraduate participants had previously taken advanced design courses that included the use of SW. Graduate students had a design background. All participants were screened through a questionnaire and interviewed to confirm that they were either intermediate or advanced users of SW.

For this study, we asked the participants to design a bus stop and a beach changing room. We assert that these two tasks are comparable, given that the prototypical design of these two structures is similar in overall size, number of individual parts, spatial complexity, structural complexity and difficulty. In the first stage, we used the apparatus we developed to visualize the encoded process graphs of 7 design solutions out of 33. Figure 6 (right column) shows these visualizations.

**Study 2: analysis of real world projects created by GC**
The second study used GC models collected from the “wild” as records of design work. Transactions in GC are records of the discrete changes that build a parametric model. A transaction can include single design steps or a group of actions at the user’s discretion. Figure 5 shows a sample transaction that adds Floor_solid in the model and changes curve02 and plane02.

GC transaction files were parsed to produce process graphs. The parsing revealed some issues. Individual transactions ignore the action order and may include information that is ambiguous and unnecessary for our purpose, such as minor edits. GC transaction files are user-editable but do not record this action, therefore they may not accurately reflect all work done. We selected transaction files that were not substantially changed once the designer completed their design task. Care was taken to make sure that the criteria used in encoding SW and parsing GC files was similar given the differences between the systems. The resulting analysis graphs are shown on the left column in Figure 6.

**FINDINGS AND DISCUSSION: COMPARISON OF GRAPHS FROM TWO DIFFERENT TOOLS**
The studies revealed that backtracking actions are highly common in both systems. They can have different span sizes covering few actions or the entire graph. There are three general backtracking patterns observed. First, in partial construct-backtrack moves designers build a part of the model, backtrack to the start of that part, refine and continue working on the next. The GC model iv, and the SW models a, b, c are of this type. Second, long construct moves and long backtracking spans may connote focus on specific aspects followed by reflection. The GC models i, ii, iii, v and the SW models d, e, and f show this pat-
tern. The third pattern combines both patterns such as the models i, ii, iii, iv in GC and e in SW. We believe that the SW g is an exception: we suspect that the participant executed a preconceived design. The intensity of backtracking in the GC models is much higher than the SW models. This can be attributed either to the tool, task, or designer. The similarities between the SW graphs, and the similarities between the GC graphs may reveal that it is most likely the tool and task rather than the designer.

Backtracking can be a sign of exploration and deferral. The long-span backtracking particularly shows that designers are willing and able to refine even the earliest of actions and parts. The actions that prompt the backtracking are either explicitly set up by the designer or implicitly present in the parametric model but nevertheless allow the designer to defer decisions. All graphs (except g) show certain backtracking-intense clusters, for example in the SW model a, the backtracking is in the end, whereas in the GC model i it is spread evenly throughout the process. We need further investigation to understand why backtracking moves differ so dramatically between the two systems.

**Initial Prototype: enabling time travel in design**

Based on our findings, we envision a software prototype providing designers with real-time feedback in parallel to the construction of the parametric model. The prototype captures, synthesizes and generates a visualization of the design process derived directly from the user’s interaction with the parametric tool. It is proposed to be an add-on to existing parametric systems. We suggest the prototype provide the following capabilities.

1. View and interact with design variations that are not part of the current CAD model. Through the UI, designers should be able to review past design variations, comparing alternative versions of their design and ultimately backtrack to previous designs.
2. Navigate the hierarchical structure by adjusting the granularity independently across the process graph to manage complex models and provide designers a way to identify the relationships of nodes on different levels of the hierarchical structure.
3. Provide information about individual and groups of nodes, arcs and nested hierarchies through tooltips and brushing.
4. Provide secondary notation to mark nodes and arcs.
5. Save, share and compare process graphs for training, archiving, supervising, collaborating or accountability.

The prototype would include a hierarchical structure that allows constructs, backtracking nodes and design variations to be grouped together to create nested hierarchies (Figure 7). The hierarchical organization would enable semantically meaningful chunks of actions to be grouped together, making the identification of the relationships between individual actions and higher-level parts or tasks in the graph.
possible (Figure 8). Due to page limitations, a detailed description of the prototype is left for another paper. Arc rendering is another major change in the prototype. Arcs are bundled together to reduce occlusion, clustering and enhancing overall readability. These graphically compact rounded rectangular arcs preserve the start-end point detection and help to reduce the crossing ambiguity of the circular arcs of our previous diagrams. The design of these arcs is based on techniques developed by Holten (2006).

CONCLUSIONS AND FUTURE WORK

The process graph as a model of design history shows a partial narrative of the design process. Through visual analysis we can identify backtracking and deferral; two strategies consistent with the literature and relevant to parametric modeling. These strategies support exploration and are present in both PCAD systems.

Our initial findings show that there is enough evidence to motivate the creation of a tool to support these two strategies in PCAD. This contention finds support in other studies on backtracking (Tidafi, Charbonneau, and Araghi, 2011). The support tool should particularly address design refinement, what-if-scenarios, why-not-try-this-scenarios and deferral. We demonstrated a possible solution in this paper, in the process of implementing it along with experimentation on backtracking and deferral.

ACKNOWLEDGMENTS

This research was partially funded by the Graphics, Animation and New Media Network of Centres of Excellence of Canada and Consejo Nacional de Ciencia y Tecnología (CONACYT), Mexico.

REFERENCES


