Automatic Presentation of Multimedia Documents Using Relational Grammars[1]

Louis Weitzman
Visible Language Workshop
MIT Media Laboratory
20 Ames Street, E15-443
Cambridge, Massachusetts 02139-4307, USA
E-mail: weitzman@media-lab.mit.edu
Mosaic: http://www.media.mit.edu/students/weitzman/home-page.html

Kent Wittenburg
Computer Graphics & Interactive Media Research
Bellcore
445 South Street, MRE 2A-347
Morristown, New Jersey 07962-1910, USA
E-mail: kentw@bellcore.com
Mosaic: http://community.bellcore.com/kentw/home-page.html

1: Abstract

This paper describes an approach to the automatic presentation of multimedia documents based on parsing and syntax-directed translation using Relational Grammars. This translation is followed by a constraint solving mechanism to create the final layout. Grammatical rules provide the mechanism for mapping from a representation of the content of a presentation to forms that specify the media objects to be realized. These realization forms include sets of spatial and temporal constraints between elements of the presentation. Individual grammars encapsulate the "look and feel" of a presentation and can be used as generators of that style. By making the grammars sensitive to the requirements of the output medium, parsing can introduce flexibility into the information realization process.

Keywords: Automatic design, grammar-directed design, visual languages, relational grammars, parsing, constraints

2: Introduction

A fully functioning multimedia system requires a wide range of stages to achieve effective automatic presentations. These include the processes of *content selection*, which identifies what to say; *media allocation*, which identifies in what media to say it; and *media realization*, which identifies how to say it in these media [May93]. However, in order to communicate effectively, adaptive multimedia systems must not merely present information, but must present information that has been specifically *designed* for a given context and task. The dynamics of information in the future will require a more careful crafting of the documents we author. Information will constantly be changing, users will have different requirements, and display devices on which they view the information will require vastly different design solutions. At the same time, documents will be including more structured knowledge of their content. In order to support the dynamics of this information-rich environment and exploit the nature of these structured documents, we will need new techniques and paradigms for the automatic design and presentation of this information.

The research described here focuses on the media realization phase and describes a formalism, Relational Grammars, for encoding design knowledge along with a methodology -- parsing, syntax-directed translation, and constraint resolution -- as a realization procedure that may deliver different documents under differing circumstances.

Kochhar, Marks, and Friedell [Koc91] characterize the articulation of a designed artifact along the axis of automaticity, from completely manual to completely automatic. Relational Grammars [Wit91] [Wit92] [Wit93] provide a number of affordances along this axis, some of which have been explored in working prototypes. These paradigms include:

- Incremental improvement
- Graphic design completion
- Design verification and error checking
- Syntax directed editing
- Structural zooming, and
- Automatic presentation.

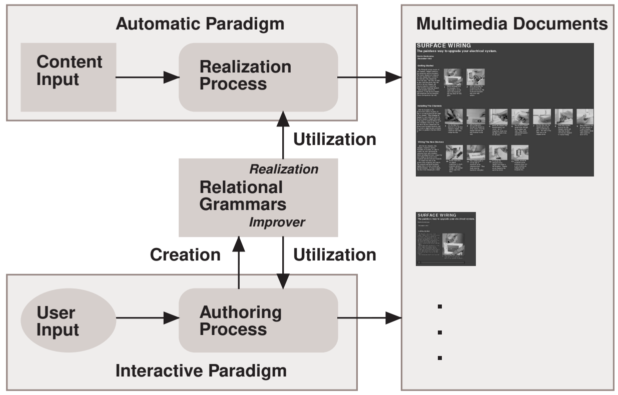
In a previous paper [Wei93], we described an interactive improver-based paradigm using Relational Grammars to support the authoring phase of a design process. In this paper we apply this same formalism in the generation phase to support an automatic

[1] This paper was the winner of the Best Student Paper award at the 1994 ACM Multimedia conference.

articulation of a multimedia document. This articulation includes graphic (e.g., font specification), spatial (e.g., relative positioning), and temporal constraints (e.g., sequence of presentation). Our larger vision includes an authoring component which produces a grammar, rather than a finished, static document. The grammar can then be used dynamically, as will be described here, to present content selected through a variety of mechanisms.

As can be seen from Figure 1, Relational Grammars play a central role in this vision. In the interactive paradigm, the improver-based grammar watches the user's authoring actions and suggests improvements, creates composite objects, and so on; in the automatic paradigm, the

Figure 1 . Vision of an integrated environment for interactive authoring and automatic articulation of multimedia documents.



realization grammar, a product of the authoring process, maps from sets of content objects to multimedia documents. This mapping can be sensitive to differing properties of the content or the context resulting in different realization options. In the present paper, we focus on the realization grammar only and the role of parsing in the realization process. Authoring of grammars and the relationship between the improver grammar and the realization grammar are issues we will address in future work.

Our language of design follows in the tradition of rule-based paradigms. One assumption this approach makes is that the domain knowledge can be encoded in the form of rules. In the domain of graphic design, layout designs are typically described in this manner. Existing publications can be analyzed and the rules of their construction can be inferred and generalized by a designer to new domains such as multimedia documents. Some rules are generic and transcend single design layouts (e.g., equal sizing and alignment of similar elements), while other rules are specific to a particular design (e.g., vertical indentation for a particular layout).

After an overview of the architecture of our presentation system, we turn to a working example, a "home screen" of a multimedia on-line document. It takes its look and feel from the table of contents of a popular magazine. This first example, which will describe the parameters of spatial and temporal layout, will allow us to focus on the architecture of our realization system. We discuss the form of the input and the output of the realization process, and provide examples of rules that articulate these particular design styles. We then move on to the realization of a dynamic presentation in which the grammar constrains the elements of the presentation both spatially and temporally. It also serves as an example of how syntax-directed translation can achieve differing results depending on characteristics of the output display. Related work in automated layout and multimedia presentation is compared. Finally, we conclude with future directions of this research.

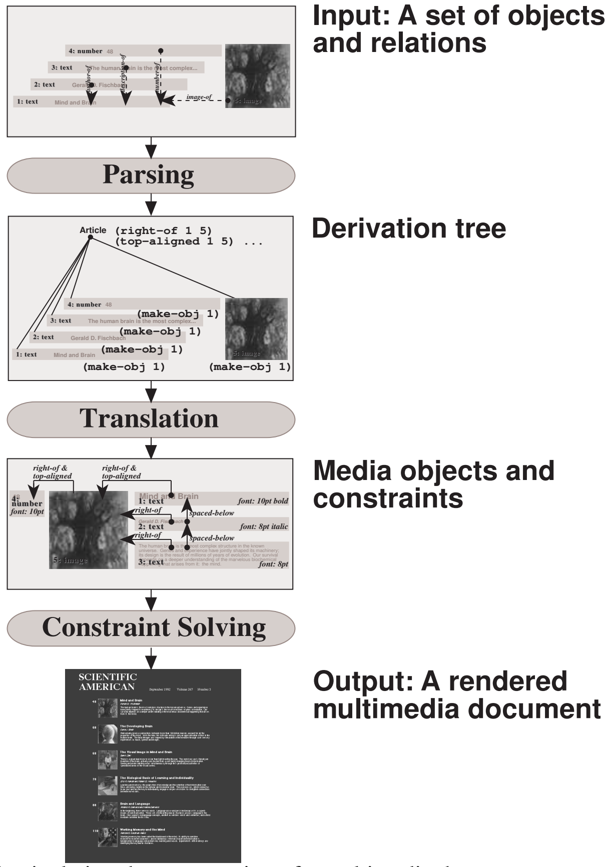


Figure 2. Overview of the process of articulating the presentation of a multimedia document.

3: Architectural overview

Relational Grammars with semantic attributes provide a mechanism for the articulation phase of the larger multimedia presentation problem. An overview of the system architecture is presented in Figure 2 and is characterized as follows. Given a representation of the content to be communicated by some design, create one or more instances of a fully articulated design. Here we are not concerned with the important problem of accessing and filtering of information. The assumption is that the information to be presented has already been chosen and relationships between the elements are known. Another process, or the user, first selects the information to be presented. The system parses content elements and relations building a derivation tree. In the first example, this corresponds to the hierarchical composition of the set of articles and headers to be included in the home screen.

Then, a translation phase begins. Following in the tradition of syntax-directed translation [Aho86], each grammar rule has an associated set of attributes which are used to compute the output forms from a syntactic derivation tree. Here the output determines a set of media objects to be created and a set of spatial and temporal constraints to be installed. Through familiar methods of computing inherited and

Figure 3. Automatic layout of *Scientific American* table of contents in the traditional style.

Figure 4. Automatic layout of *Scientific American* table of contents (i.e., the same input as in Figure 3) using the style of WIRED magazine's table of contents.

synthesized attributes, the semantic output of the parse tree is produced. A constraint resolution procedure is then invoked to solve the constraints among media objects that in this case determine the actual numerical values for spatial positioning. Finally, the media objects are rendered on the display.

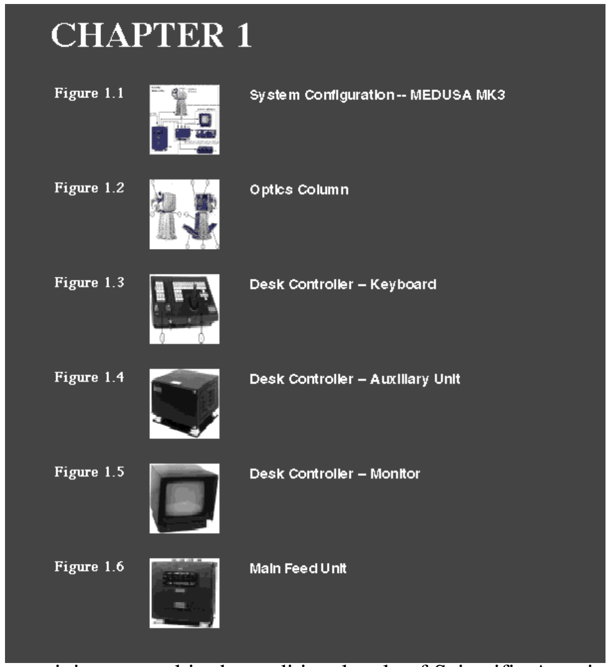


Figure 5. Automatic layout of an online training manual in the traditional style of Scientific American. Variations of the basic rules parse the articles even though information is missing (i.e., authors and descriptions).

4: Examples of spatial layout

Figures 3-5 illustrate the results of our system in the realm of spatial layout and graphical style. Figures 3 and 4 use the same content but different grammars to achieve unique styles of presentation. Figure 3 is an automatically generated on-line version of the table of contents modeled directly from an issue of Scientific American.[2] Figure 4 uses the grammar based on another publication's style to present the same information.[3] Figure 5 illustrates different content that utilizes the same grammar as in Figure 3. However, note that in Figure 5 there is less information present (i.e., no authors or descriptions). The grammar in question contains rule variants that permit successful parses even though such content differences exist.

5: The articulation components

We will use Figure 3 as our running example in the following discussion of the sequence of processing steps in our articulation process.

Input to the parser

Input to the parser is a set of content objects as well as domain-dependent relations which hold between them. Conceptually the input is a database, which can be thought of abstractly as a graph with primitive objects as nodes and relations as (hyper)arcs. In the online magazine example, we read in a file describing the content information and construct our own internal database. Figure 6 illustrates the form of the input data to the parser. For example, the figure indicates that "`Gerald D. Fischbach`" is in the *`author-of`* relation to "`Mind and Brain`." The basic types of objects that comprise the input to the parsing process include `text`, `numbers`, and `images`.[4] In this example we order the articles as they are in the original publication (i.e., the *precedes* relation), but could have used other relations to determine the presentation sequence, such as importance, type of article (e.g., lead with a general science article), highest priority based on user profile, etc.

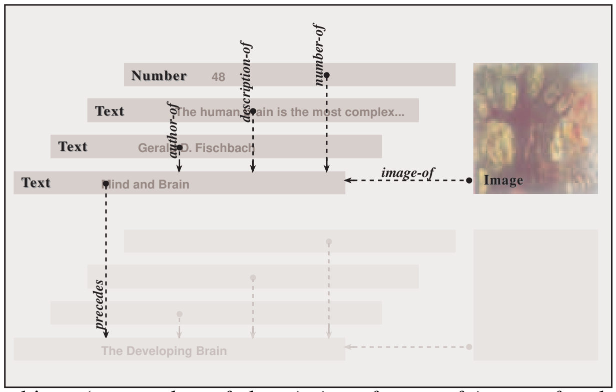
---

2. Scientific American, September 1992, Vol. 267, No. 3, special issue on Mind and Brain.

3. This layout is based on a grammar derived from Wired, January 1994, Vol 2.01.

4. Depending on the needs and purpose of the application, the input could also contain higher structured objects such as a prestructured article composite. In this case, the system would not have to create an article composite object through the parsing process. The advantage of using more primitive elements as input is that the parser can do a certain amount of selection in assembling pre-existing pieces, an advantage we don't really exploit in the examples discussed here.

Figure 6. An example of partial input to the parser for the online magazine example. Input is a set of primitive objects (e.g., *text*, *numbers*,



and *images*) and relations between the objects (e.g., *author-of*, *description-of*, *page-of*, *image-of*, and *precedes*).

Grammar

Relational Grammars are an extension to string-based grammars in which composition relations other than just string concatenation are allowed. They are a member of a wide-ranging family of higher-dimensional grammar frameworks, e.g., array, tree, or graph grammars [Ros90] or unification-based constraint grammars [Hel91]. Where string grammars generate or compose expressions consisting of one-dimensional arrays of symbols, these higher-dimensional grammars deal in structures characterized by, for example, geometric positions in two or three dimensional space, topological connectivity, or, as in the case at hand, arbitrary semantic relations holding among information objects. The motivation for using higher-dimensional grammars rather than string-based grammars is that strings alone will not be a rich enough structure to capture relationships in this problem domain. We believe that information content and context serving as the input to a presentation procedure is not naturally or easily coerced into a fully ordered, linear structure such as a string. However, the architecture proposed here would still be appropriate for use with more traditional string-based frameworks such as attribute grammars [Knu68].

Relational Grammars depend on generalized relations between the right-hand-sides of rules to constrain the rule applications and direct the parsing. In our running example, content relations such as *author-of, description-of, page-of*, and *image-of* are utilized. Figure 7 shows a rule which is utilized repeatedly in the derivation behind Figure 3. The context-free backbone of this rule corresponds to the rewrite rule:

Article $\rightarrow$ *Text Text Text Number Image.*

Thus *0* indicates the left-hand-side rule element and *1...n* represent the right-hand-side rule elements. (See [Wit92] [Wit93] for details concerning the grammar formalism.)

```
(Defrule (Make-Article The-Grammar)
  (0 Article)
  (1 Text)
  (2 Text   (Author-Of      2 1))
  (3 Text   (Description-Of 3 1))
  (4 Number (Page-Of        4 1))
  (5 Image  (Image-Of       5 1))
  :OUT
  ((right-of     1 5)
   (right-of     2 5)
   (right-of     3 5)
   (right-of     5 4)
   (top-aligned  1 5)
   (top-aligned  5 4)
   (spaced-below 2 1)
   (spaced-below 3 2)
   (set-font 1 10pt :bold)
   (set-font 2  8pt :italic)
   (set-font 3  8pt :plain)
   (set-font 4 10pt :plain)))
```

Figure 7. The definition of the `Make-Article` rule. The conditions for rule matching include relations between the elements (e.g., *author-of*). `Article` is the resulting composite category that is created when the five basic categories (numbered 1 through 5) are matched and the indicated relations satisfied.

The forms following `:OUT` in the rule definition represent an extension of Relational Grammars as previously conceived to include "semantic" attributes. Consistent with standard practice in compiler design, where attributes are used to generate compiler code, here we use attributes to generate code for creating media objects. In the current implementation, we use only synthesized attributes, i.e., the output attribute of each node of the derivation tree depends only on the values of attributes of nodes below it [viz Knu68], but we are extending the framework to incorporate inherited attributes as well.

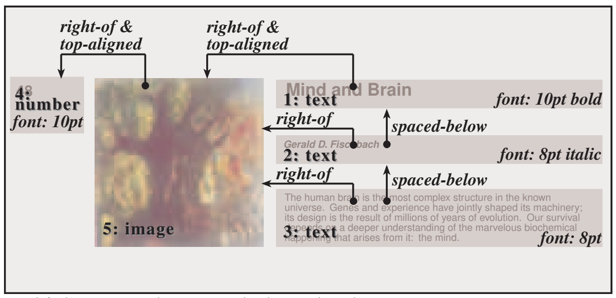
Parsing

The parser's goal is to build a derivation tree that covers the input. In our current implementation we use the bottom-up, non-deterministic algorithm presented in [Wit92] with an additional control feature that allows a depth-first search, i.e., the parser returns as soon as a new derivation is found. Subsequently, parses may be sought until the search space is exhausted. The output of parsing is then one or more derivation trees, each of which yields an independent presentation. In the future work section, we discuss other parsing algorithms that we intend to explore for the purpose of multimedia design and presentation.

Translation

When a derivation is found that covers all of the input, the set of `:OUT` forms is collected through a depth-first left-to-right walk through the derivation tree. The `Make-Article` rule, in Figure 7, includes a number of forms constraining spatial (*right-of*, *top-aligned*, and *spaced-below*) as well as graphic (*set-font*) attributes. In this example, basic lexical items include an output form which creates the realized element in the presentation. Figure 8 illustrates these output constraints graphically.

Figure 8 . A part of the translation for the presentation in Figure 3 generated with the rule in Figure 6. The translation has created media objects and installed spatial constraints which must subsequently be solved.



Constraint Solving

The use of a constraint solving algorithm is a natural match with the rule formalism. Each rule installs only local constraints between the elements of the rule body. Rules for creating composite structures then create the constraints that link these smaller constraint networks together. The output forms of the final rule in the derivation then seeds this network with actual x and y values that are propagated during the constraint satisfaction phase of realization. In the final presentation, we allow the user to interact with the elements. In our online table of contents example, the user can move and resize individual elements and the constraint system interactively maintains the proper relationships installed by the grammar.

The constraint propagation system being used is DeltaBlue developed at the University of Washington [Fre90, Mal91]. DeltaBlue is designed for non-cyclic constraint networks to be used in interactive applications with up to ~20,000 constraints. In our experience, a grammar's constraints can easily be crafted to avoid cycles in the final presentation.

6: Examples of dynamic documents

In the previous examples, all output was constrained geometrically. The following examples include temporal constraints as well. With these additional constraints, the grammar can specify dynamic relationships between the elements of a user controlled presentation. In addition, interaction dynamics is supported with the inclusion of hyperlinks between documents. These documents are created using HTML and presented in NCSA Mosaic [NCS93].

A repair procedure taken from a Popular Mechanics article [Hen93] serves as the basis for our next example. This repair procedure is composed of three major steps, each

Figure 9. Presentation of a home repair procedure from Popular Mechanics magazine on a high-resolution display.

Figure 10. Presentation of the same home repair procedure as in Figure 9 but constrained temporally as well as spatially. The user can manually step through the procedure by interacting with the horizontal bar at the top of the page.

containing a number of minor steps. Figure 9 shows the layout of the complete procedure on a large, high-resolution display. However, if we consider the characteristics of the output medium as part of the input to the parser, we can make the presentation sensitive to the output requirements.[5] For instance, if this is a repair procedure being carried out by a person in the field with a hand-held digital assistant, the grammar can generate quite a different presentation. Figure 10 shows this second interpretation displaying the first step of the complete procedure. As part of the presentation, the horizontal bar at the top of the page becomes an active object which controls the presentation of elements in the repair procedure. As the user interacts with the bar, the visibility of elements is modified. This is based on the tapping mechanism developed in the Steamer project [Hol84] and generalized in the Icon Editor [Ros90].

In order to visualize complex processes, Steamer presented a graphical interface to an underlying simulation of steam propulsion. Elements in the interface were *tapped* into the simulation maintaining the visual representation of the relevant part of the mathematical model. Here, the simulation is the presentation. As the user interacts with the horizontal bar, elements appear and disappear in the proper sequence. The display area of the device used to articulate the second presentation is much smaller than the first presentation. The grammar trades off the spatial resolution of the high-quality display with temporal resolution on the

5. The mechanism for making this property of the output display part of the parse input is through the predicate mechanism of Relational Grammars. That is, certain rules will be fired only if a global predicate such as *large-display-p* is true or not.

much smaller screen.

Another way to support interaction dynamics is through the use of hypermedia documents. This class of documents provides easy access to related information by simply clicking on links presented to the user. Figure 11 illustrates an online table of contents "home page" presented in Mosaic. In this example, the title of the article is the link to the actual article online.[6] This example is created using the interactive improver-based scenario. As the document author adds elements to the design, an improver grammar watches and suggests refinements. In the process, composite structures are created. What is interesting in the context of this paper is that these composites can now support the realization of the authored document in different formats.

At any point during the design process, the author can request the generation of an HTML file describing the input. The system identifies the most general composite structure created thus far. This composite, in the traditional object-oriented programming style, can produce the HTML description of itself and all of its children. This is true for a single article, a sequence of articles, or the completed table of contents. Currently, the links themselves are automatically produced by a database look up, but could be automatically installed as a result of the parsing action.

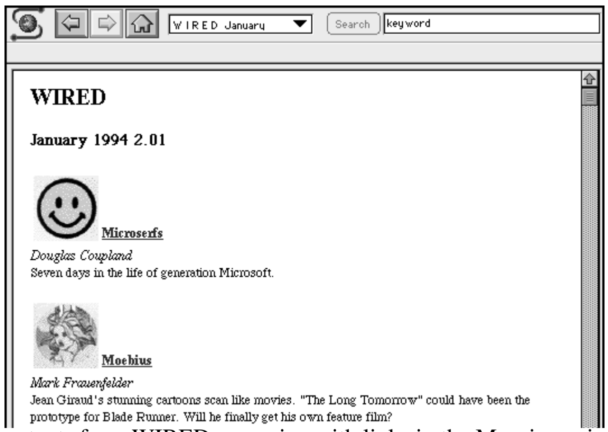


Figure 11. Presentation of a table of contents from WIRED magazine with links in the Mosaic environment.

7: Related work

It is important to note that the approach described in this paper addresses only a very specific part of the more general multimedia presentation problem: the problem of articulation. Our main concern is with the representation and articulation of graphic design knowledge. A standard technique for controlling design articulation is the use of the grid [Hur78]. Early research that focuses on articulation using the grid for the layout of information is [Fei88]. The system uses information about the kind of material to be displayed, the user, and display hardware. The system then creates displays using more information about the kinds of objects to be presented. This system first generates a grid and then uses it in the presentation of information. In our work, we characterize the same categories of information but the notion of a grid is implicitly represented in the output constraints of the grammar.

Dynamic interface vs static layout

In order to support the view that the end result of multimedia generation is a dynamic interface rather than a static page (or a static sequence of frames), we allow each presentation element to be connected to a value within an application or simulation. Then, each element in the interface has the potential to become dynamic by visualizing values within the application or simulation and/or modifying those values through user interaction. For example, in a multimedia training document, the simulation of a device to be repaired could control how the element is presented and how it behaves when the student interacts with the document. This behavior can be encoded in the output of the grammar constraints as illustrated in the example shown in Figure 10.

The separation of interface from application is common to the position developers of user interface management systems (UIMS) take. It enhances modular development and supports automatic construction and reuse of interface elements. Early work in the visualization of simulations can be found in Steamer [Hol84] and the Process Visualization System [Fol86]. This approach separates the dynamics of the application from the specific presentation to the user. Typical UIMS in fact support this same separation [Are88, Wie90, and Kim93]. DON [Kim93] consists of an application model (containing both data and control

---

6. Accessed through WIRED's World Wide Web server at http://www.wired.com.

models), a design process model supporting top-down iterative design and graphic design knowledge to support the layout process. An expert system, rule-based approach is used. ITS [Wie90] is similar but emphasizes the usability of the style language by the interface designer. By making the underlying rule-base more accessible, the designer has the ability to modify the rule-base itself.

Multimedia presentation systems

Today, many systems address the larger problem of multimedia presentation and include articulation as just one component of a larger system. COMET [Fei93] is a knowledge-based system that produces interactive and coordinated explanations that combine text with 3D graphics. It uses "media generators" which can communicate with each other to produce a presentation. The underlying generator within COMET, called IBIS [Sel91], uses a rule-based approach which generates and tests its presentation alternatives. Each rule invokes a set of stylistic strategies that specify high-level visual effects (e.g., highlighting an object). These strategies are in turn realized by other, lower-level rules. Another general multimedia platform for generation is WIP [And93]. WIP approaches the problem of multimedia design as a planning problem to achieve coherent multimodal presentations of text and graphics. The articulation of the layout is the same constraint satisfaction algorithm used in our research. They encode the graphical design knowledge through constraints which express semantic/pragmatic relations (alignment, grouping, symmetry or similarity) and geometrical/topological relations (absolute and relative position).

Coordinating complex temporal relationships has been attempted in some of the larger multimedia systems. [Kar90] examines these complex relationships in the context of building animations between elements of the presentation. This includes the temporal synchronization of special editing effects (e.g., cuts, wipes, fades, dissolves), camera control movements, and multiple views of the same information. As part of this more complete model, user interactions are also modeled. The use of temporal constraints in our system is a simple extension of the spatial constriants and is based upon Allen's categories of temporal logic [All83]. We construct a constraint network based on the relative time of an element's presentation suggested by the grammar. Then, using the same constraint satisfaction approach for spatial layout, we identify the relative time slots for realization.

Other systems that automatically design artifacts are characterized by the fact that the domain can be expressed as a set of discrete rules that easily encode design knowledge. In architecture, this has been limited to applications in highly constrained specialties such as kitchen design [Oxm87] or simple floorplan layout [Mit76]. Other domains, like information graphics [Mac86], also lend themselves to this approach. One example, ANDD [Mar90] is a system that does fully automatic design of network diagrams. It is an effective rule-based system that designs and articulates network diagrams to communicate information represented in arbitrary attributed graphs and is part of a multimedia explanation facility.

The main difference between these other systems and the work described here is that they all employ some form of forward-chaining, rule-based system. These systems use rules and metarules to control the search of the design space and use the generate-and-test paradigm to determine the appropriateness of a solution. More often than not, this aspect of these systems receives little attention in the literature, perhaps because adequate control mechanisms can be difficult to specify. We, on the other hand, use an independently motivated parsing algorithm. The issue of control is thus folded into the more general problem of finding efficient parsing algorithms for higher-dimensional grammars, a continuing research topic. As progress is made on this front, the results can be incorporated into future versions of our system. In the meantime, authors of the rules used in our approach need not concern themselves with issues of control.

8: Future directions

In support of the design process, Relational Grammars have proven useful as the basis for a number of different interaction paradigms. An open question is how much further they can be pushed to support the creative activity of design. This larger vision suggests a number of future research directions.

Parsing algorithms

We believe that further research on parsing algorithms in the service of multimedia articulation is called for. For example, depending on properties of the content database, deterministic LR-style algorithms for Relational Grammars may be possible, which of course would be more efficient that the one we are currently using. Research on such algorithms for multidimensional grammars is ongoing. On the other hand, nondeterminism in parsing, along with the possibility of ambiguity in derivations, may play the role of generating more than one possible presentation, which could in turn be critiqued by a higher-level control structure in more "intelligent" applications. Still another idea is to use predictive-style parsers to help do some of the content selection [see Wit93].

Ambiguity in design

Traditionally, ambiguity in parsing is something to be avoided. However, to support a creative process, this may not be the case. Ambiguity should be viewed as beneficial and something that needs to be maintained until later design decisions have been able to disambiguate the designer's intention. These ambiguous interpretations can be viewed as alternative solutions in the larger design space. Building an environment to explore this space enabling better solutions is one of our goals.

Cooperative design

We have shown two distinct paradigms that use Relational Grammars for design support, completely automatic and improver-based. A different approach embraces both methods within a single paradigm. Cooperative CAD [Koc90] is a technique that mixes automatic articulation with manual control. This cooperative paradigm puts the user in control to manually articulate design decisions but also supports automatic design exploration by the system. Relational Grammars can provide a formalism for this approach.

Design by demonstration

The very nature of design suggests that the solution to the problem is not known a priori. In fact, as the design progresses, initial assumptions and decisions may be redefined or dropped altogether. This suggests that the language of a design, the basic vocabulary and the rules for combination, evolves as the solution is explored. What is important is having an environment that will respond to these needs. A designer should have the ability to redefine rules and add new ones to support their design process. An area of research that can provide some insight is in the area of *programming-by-demonstration* [Cyp93]. With this approach, designers could modify an existing grammar or create new grammars without coding. This would help create a design environment in which nonprogrammers can modify the existing rule sets.

New applications

Exploration of the use of Relational Grammars to support online training manuals and the design of dynamic presentations is a continuing research focus. In addition, as interest in the Internet continues to grow, it is becoming more apparent that we will need automated techniques to support the design and presentation of information. One interesting application for Relational Grammars is in the production of timely, personalized newspapers. Intelligent agents would retrieve information over the net and submit it to the grammar for automatic presentation.

9: Acknowledgements

We would like to thank both the Media Lab's Visible Language Workshop, directed by the late Professor Muriel Cooper, and Bellcore's Computer Graphics & Interactive Media Research Group, directed formerly by Jim Hollan and currently by George Furnas, for their support of this research. In addition, the first author's work is sponsored in part by the Media Lab's NIF project, JNIDS, and the Alenia Corporation. This paper is dedicated to the creative spirit of Professor Cooper.

10: References

[Aho86] Aho, A.V., R. Sethi, and J.D. Ullman. *Compilers: Principles, Techniques, and Tools*, Addison-Wesley, Reading, Ma., 1986.

[All83] Allen, J. Maintaining Knowledge About Temporal Intervals, *Communications of the ACM*, 26, 11 (1983), pp. 832-843.

[And93] Andre, E., Finkler, W., Graf, W., Rist, T., Schauder, A. and Wahlster, W. WIP: The Automatic Synthesis of Multimodal Presentations, *Intelligent Multimedia Interfaces*, Maybury, M., Ed, AAAI Press/MIT Press, Cambridge, MA. 1993, pp. 75-93.

[Are88] Arens, Y., Miller, L., Shapiro, S. and Sondheimer, N. Automatic Construction of User-Interface Displays, In *Proceedings of the American Association of Artificial Intelligence*, Vol. 2 (August 21-26, St. Paul, MN). 1988, pp. 808-813.

[Cyp93] Cypher, A., Ed. *Watch What I Do: Programming by Demonstration*, MIT Press, Cambridge, Ma., 1993.

[Fre90] Freeman-Benson, B., Maloney, J. and Borning, A. An Incremental Constraint Solver, *Communications of the ACM*, 33, 1 (January, 1990), pp. 54-63.

[Fei88] Feiner, S. A grid-based approach to automating display layout, In *Proceedings of Graphics Interface '88* (June 6-10, Edmonton, Canada). 1988, pp. 192-197.

[Fei93] Feiner, S. and McKeown, K. Automating the Generation of Coordinated Multimedia Explanations, *Intelligent Multimedia Interfaces*, Maybury, M., Ed, AAAI Press/MIT Press, Cambridge, MA., 1993, pp. 117-138.

[Fol86] Foley, J. and McMath, C. Dynamic Process Visualization, *IEEE Computer Graphics and Application* 6, 3 (March, 1986), pp. 16-25.

[Fol91] Foley, J., Kim, W., Kovacevic, S. and Murray, K. UIDE - An Intelligent User Interface Design Environment, In *Architectures for Intelligent Interfaces: Elements and Prototypes*, Sullivan, J. and Tyler, S. Eds, Addison-Wesley, Reading, MA. 1991, pp. 339-384.

[Hel91] Helm, R., and Marriott, K. A Declarative Specification and Semantics for Visual Languages, *Journal of Visual Languages and Computing*, 2 (1991), pp. 311-331.

[Hen93] Henkenius, Merle. Surface Wiring, *Popular Mechanics* 170, 12 (December, 1993). pp. 63-66.

[Hol84] Hollan, J., Hutchins, E. and Weitzman, L. Steamer: An Interactive Inspectable Simulation-Based Training System, *AI Magazine* 5, 2 (1984), pp. 15-28.

[Hur78] Hurlburt, A. *The Grid*, Van Nostrand Reinhold Co., New York, 1978.

[Kar90] Karp, P. and Feiner, S. Issues in the Automated Generation of Animated Presentations, In *Proceedings of Graphics Interface*, (May 14-18, Halifax, Canada). 1990, pp. 39-48.

[Kim93] Kim, W. and Foley, J. Providing High-level Control and Expert Assistance in the User Interface Presentation Design, In *INTERCHI'93 Proceedings*, (April 24-29, Amsterdam, The Netherlands). 1993. pp. 430-437.

[Knu68] Knuth, D. E. Semantics of context-free languages, *Mathematical Systems Theory* 2 (1968), pp. 127-146.

[Koc90] Kochhar, S. and Friedell, M. User control in cooperative computer-aided design, In *Symposium on User Interface Software and Technology '90*, (October 3-5, Snowbird, Utah). 1990, pp. 143-151.

[Koc91] Kochhar, S., Marks, J., and Friedell, M. Interaction Paradigms for Human-Computer Cooperation in Graphical-Object Modeling, in *Proceedings of Graphics Interface '91* (June, Calgary, Canada). 1991, pp. 180-191.

[Mac86] Mackinlay, J. Automating the design of graphical presentations of relational information, *ACM Transactions on Graphics* 5, 2 (1986). pp. 110-141.

[Mal91] Maloney, J. *Using constraints for user Interface Construction*, PhD Thesis, University of Washington, Technical Report 91-08-12, Seattle, Wa., 1991.

[Mar90] Marks, J., A syntax and semantics for network diagrams, In *Proceedings of the IEEE Workshop on Visual Languages* (October 4-6, Skokie, Illinois). 1990, pp. 104-110.

[May93] Maybury, M. Ed. *Intelligent Multimedia Interfaces*, AAAI Press/MIT Press, Cambridge, Ma., 1993.

[Mit76] Mitchell, W., Steadman, J. and Liggett, R. Synthesis and optimization of small rectangular floor plans, *Environment and Planning B: Planning and Design* 3, 1 (1976), pp. 37-70.

[NCS93] NCSA. Internet Manuscript from mosaic@ncsa.uiuc.edu, 1993.

[Oxm87] Oxman, R. and Gero, J. Using an expert system for design diagnosis and design synthesis, *Expert Systems* 4, 1 (February, 1987), pp. 4-15.

[Ros90] Rosenfeld, A. Array, Tree, and Graph Grammars, in Bunke, H., and Sanfeliu, A., Eds. *Syntactic and Structural Pattern Recognition: Theory and Applications*, World Scientific, Singapore, 1990.

[Ros90] Rosenstein, M. and Weitzman, L. Icon Editor: The Specification of Graphic Behavior without Coding, In *Proceedings of the 23rd Annual Hawaii International Conference on Systems Sciences* (January 2-5, Kailua-Kona, HI.). 1990, pp. 523-530.

[Sel91] Seligmann, D. and Feiner, S. Automated Generation of Intent-Based 3D Illustrations, In *Proceedings of the ACM Siggraph'91* (July 28-Aug. 2, Las Vegas, NV.). 1991, pp. 123-132.

[Wei93] Weitzman, L. and Wittenburg, K. Relational Grammars for Interactive Design, In *IEEE Symposium on Visual Languages* (August 24-27, Bergen, Norway). 1993, pp. 4-11.

[Wie90] Wiecha, C., Bennett, W., Boies, S., Gould, J. and Greene, S. ITS: A tool for rapidly developing interactive applications, *ACM Transactions on Information Systems* 8, 3 (July, 1990), pp. 204-236.

[Wit91] Wittenburg, K., Weitzman, L. and Talley, J. Unification-Based Grammars and Tabular Parsing for Graphical Languages, *Journal of Visual Languages and Computing* 2 (1991), pp. 347-370.

[Wit92] Wittenburg, K. Earley-style Parsing for Relational Grammars, In *Proceedings of IEEE Workshop on Visual Languages* (Sept. 15-18, Seattle, Wa.). 1992, pp. 192-199.

[Wit93] Wittenburg, K. Adventures in Multidimensional Parsing: Cycles and Disorders. In *Proceedings of the Third International Workshop on Parsing Technology* (August, Tilburg, Netherlands and Durbuy, Belgium). Association for Computational Linguistics and Tilburg University, 1993, pp. 333-348.