RR-9
Research

Research in Constraint-Based Layout, Visualization, CAD, and Related Topics: A Bibliographical Survey

Walter Hower and Winfried H. Graf

Deutsches
Forschungszentrum
für Künstliche
Intelligenz GmbH

**Research
Report**
RR-95-12

Research in Constraint-Based Layout, Visualization, CAD, and Related Topics: A Bibliographical Survey

Walter Hower and Winfried H. Graf

December 1995

Deutsches Forschungszentrum für Künstliche Intelligenz GmbH

Postfach 20 80
67608 Kaiserslautern, FRG
Tel.: + 49 (631) 205-3211
Fax: + 49 (631) 205-3210

Stuhlsatzenhausweg 3
66123 Saarbrücken, FRG
Tel.: + 49 (681) 302-5252
Fax: + 49 (681) 302-5341

Deutsches Forschungszentrum
für
Künstliche Intelligenz

The German Research Center for Artificial Intelligence (Deutsches Forschungszentrum für Künstliche Intelligenz, DFKI) with sites in Kaiserslautern and Saarbrücken is a non-profit organization which was founded in 1988. The shareholder companies are Atlas Elektronik, Daimler-Benz, Fraunhofer Gesellschaft, GMD, IBM, Insiders, Mannesmann-Kienzle, Sema Group, Siemens and Siemens-Nixdorf. Research projects conducted at the DFKI are funded by the German Ministry of Education, Science, Research and Technology, by the shareholder companies, or by other industrial contracts.

The DFKI conducts application-oriented basic research in the field of artificial intelligence and other related subfields of computer science. The overall goal is to construct systems with technical knowledge and common sense which - by using AI methods - implement a problem solution for a selected application area. Currently, there are the following research areas at the DFKI:

- ☐ Intelligent Engineering Systems
- ☐ Intelligent User Interfaces
- ☐ Computer Linguistics
- ☐ Programming Systems
- ☐ Deduction and Multiagent Systems
- ☐ Document Analysis and Office Automation.

The DFKI strives at making its research results available to the scientific community. There exist many contacts to domestic and foreign research institutions, both in academy and industry. The DFKI hosts technology transfer workshops for shareholders and other interested groups in order to inform about the current state of research.

From its beginning, the DFKI has provided an attractive working environment for AI researchers from Germany and from all over the world. The goal is to have a staff of about 100 researchers at the end of the building-up phase.

Dr. Dr. D. Ruland

Director

Research in Constraint-Based Layout, Visualization, CAD, and Related Topics: A Bibliographical Survey

Walter Hower and Winfried H. Graf

DFKI-RR-95-12

A version of this paper has been published in:
I. F. Cruz, K. Mariott, and P. Van Hentenryck (Editors). Proceedings of the International Workshop on Constraints for Graphics and Visualization (CGV '95) in conjunction with the International Conference on Principles and Practice of Constraint Programming (CP95), Cassis, France, September 18, 1995.
An updated version of this document is available via http://www.uni-koblenz.de/~walter/ and http://www.dfki.uni-sb.de/~graf/

This work has been supported by a grant from The Federal Ministry of Education, Science, Research and Technology (FKZ ITWM-9400).

ISSN 0946-008X

Research in Constraint-Based Layout, Visualization, CAD, and Related Topics: A Bibliographical Survey

Walter Hower
Institut für Informatik
Universität Koblenz-Landau
Rheinau 1
D-56075 Koblenz, Germany
walter@uni-koblenz.de

Winfried H. Graf
German Research Center for
Artificial Intelligence (DFKI) GmbH
Stuhlsatzenhausweg 3
D-66123 Saarbrücken, Germany
graf@dfki.uni-sb.de

Abstract

The present work compiles numerous papers in the area of computer-aided design, graphics, layout configuration, and user interfaces in general. There is nearly no conference on graphics, multimedia, and user interfaces that does not include a section on constraint-based graphics; on the other hand most conferences on constraint processing favour applications in graphics. This work of bibliographical pointers may serve as a basis for a detailed and comprehensive survey of this important and challenging field in the intersection of constraint processing and graphics. In order to reach this ambitious aim, and also to keep this study up-to-date, the authors appreciate any comment and update information.

Contents

1 Introduction

Research in the area of layout, graphics, computer-aided design (CAD), and visualization comprises more and more novel techniques of the artificial intelligence (AI) discipline—see e.g., [Smithers, 1989, Foley *et al.*, 1990, Catarci *et al.*, 1992, Maybury, 1993]. The present work focuses on *constraint processing* and compiles numerous papers in conjunction with (intelligent) graphical design and visualization.[1]

Already here we would like to point to a few papers just to illustrate the existence of the intersection area of constraint processing and graphical design. For instance, [Maleki, 1987] employs interactive graphics to build the constraint network, whereas [Maulsby *et al.*, 1988] uses constraint solving in interactive graphics. [Gleicher, 1995] discusses constraint-based interactive graphical applications in a broad sense. [Sistare, 1991] provides both the constraint description and the underlying geometry in one graphical framework to make it easier for the user to understand the representation of both. [Amarel *et al.*, 1991] also points to intelligent problem solving methods as, e.g., constraint processing, due to the design complexity. As graphics infrastructure becomes more and more sophisticated, also commercial activities have concentrated on constraint-based graphical applications [Cras, 1993, Kurlander *et al.*, 1994].

The constraint satisfaction problem (CSP) comprises a set of n variables with associated finite domains (FDs), and some combinations of value assignments ("constraints") to the variables; then, in order to get the globally consistent solution, we need to compute the set of all n-tuples consistent with the given constraints.[2] ([Tsang, 1993] may serve as a further reference. Please further consult [Freuder *et al.*, 1995].)

By and large, there are mainly two different qualities of constraint solving techniques: *local constraint propagation* and *global constraint satisfaction*. Briefly speaking, the distinction is as follows: local propagation just considers those parts which are closely related to each other and works only on a subset of the problem; global satisfaction however copes with the matter as a whole. Or from another point of view: a local technique tries to rule out values which are identified to be of no further use and thereby potentially reduces the domains of the variables; a global algorithm however keeps track of the interrelation of the values and maintains complete combinations. The result of local propagation, the so-called "locally consistent solution", is an n-tuple of (domain) sets, whereas the result of global satisfaction, the so-called "globally consistent solution", is a set of n-tuples enumerating all possible combinations.[3] Although local propagation is a fast technique we would already now like to take the opportunity to point to the fact that local consistency does not imply global consistency—cf. also [Kramer, 1992] (p. 62, footnote).[4] Thus, the locally consistent solution may just serve as a pre-processing state; the correct instantiations of the variables

---

[1] Although a constraint-based philosophy is found in nearly all the papers there are cases where the implementation of this idea has not been performed by constraint processing techniques in a narrow sense; nevertheless, we have cited some of them due to their intention.

[2] Striving for *all* combinations is important in a lot of real-world applications; for instance, modelling a set of traffic lights requires all possible colour assignments ([Hower, 1995]). Furthermore, in computer-aided layout design all potential configurations should be offered to the user; this feature is one of the reasons to employ a constraint-based approach. (You may consult [Van Hentenryck, 1995].)

[3] By 16 August 1995 the first author has similarly reported this clarification to the internet newsgroup *comp.ai*.

[4] You may consider [Dechter and van Beek, 1995] regarding the way of the quality of a locally consistent solution to a globally consistent one.

must still be tried. Please note that this view of local propagation is different to the one used by most of the local propagation solvers mentioned in this survey. However, the statement above is an important classification in regard to the classical CSP; local consistency in the sense just described above is incorporated in the FD solvers mentioned in our paper. In [Borning, 1981] the weakness of local propagation has already been recognized, and it refers to the need of a global analysis—see also [Helm *et al.*, 1992] (p. 302). [Faltings and Sun, 1993] states the fact that binary constraint networks are often not expressive enough and prefers user interaction when some modifications come up (pp. 1454/1456 there). [Faltings, 1994b] nicely motivated a constraint-based approach to CAD. [Haroud and Faltings, 1994] deals with global consistency for continuous domains.

It is no surprise that in the material considered for this work it has really turned out that constraint satisfaction techniques are fundamental to intelligent CAD. Let's have a look at some prominent citations: "Constraints will be taking an increasingly prominent position in our paradigms for programming in the years to come."—see [Borning, 1981] (p. 386). "(...) in the CAD/CAM community (...) constraint systems have become *de rigueur* for 'serious' CAD systems."—see [Kramer, 1992] (p. xviii). "The development of computer graphics constraint systems has been of considerable interest in recent years."—see [Rankin, 1995] (p. 71). In the papers taken into consideration local propagation, in the sense mentioned by the first item of subsection 2.2, has been the method mostly preferred—probably due to the efficiency of the procedure. Unfortunately, it sometimes leaves to the reader to imagine how global consistency is achieved.[5]

Although nearly each paper mentioned in this survey includes a section on related work, there seems to be a large demand for a complete overview about the current state of the art. Our bibliography points to research literature in the field of constraint-based graphical design.[6] We have tried to form clusters collecting papers belonging to a common theme. However, in order to suppress multiple citings of articles at various places in this survey and also to save space we have decided to sometimes cite them just once. Please note that due to time limitations we are only able to mention most of the contributions without commenting on all of them in detail.

This paper is organized as follows: First, in section 2 we briefly discuss different views on constraints and constraint processing and overview the programming paradigms and constraint solving techniques that have been applied to graphical applications. We propose a classification of the different approaches according to the models they use for constraint solving. Section 3 surveys constraint-based graphical applications according to the classification given in section 2, while section 4 addresses geometrical layout problems that can be formalized as CSPs. In section 5 we describe some special features of constraint solvers for graphical applications whilst section 6 spots further techniques. Final remarks conclude the present work.

---

[5] In some cases just simple assignment techniques seem to get employed. (Backtracking-like algorithms are discussed in [Zahn and Hower, 1996].)

[6] It was a laborious job to collect all these articles from so many diverse sources; the present paper probably provides such an exhaustive collection for the first time. This work has benefited from extensive overviews given in [Leler, 1988], [Borning *et al.*, 1992], [Sannella, 1994], [Fron, 1994], and [Gleicher, 1995], where some information presented here has been derived from.

2 Constraints for Graphics and Visualization

As it has been shown in previous work many graphical and multimedia automation tasks can be facilitated efficiently using constraint processing techniques. Especially, constraints have been used extensively in automated geometric layout to maintain visibility and consistency in interactive user interfaces. The declarative semantics of constraint languages allows one to specify graphical objects and their inter-relationships while avoiding extraneous concerns about the realization of the visualization algorithms. Moreover, applying constraints to graphical displays automatically allows a better control of the design space and facilitates an incremental re-design of a generated presentation on the fly. This decoupling of the visualization from the application code results in easy modification and elegant specification capabilities and facilitates adaptive and flexible presentations.

Another major advantage is the ability of constraints to describe complex objects simply and naturally. Thus constraint networks provide an elegant mechanism to state design-relevant knowledge about heterogeneous geometrical and topological relationships declaratively, while characterizing properties between different kinds of graphical objects that can be maintained by the underlying system. They can easily be used to specify layout requirements in graphical environments in order to guarantee local circumscriptions of the presentation such as format restrictions, margins, distances, and non-overlapping, or to maintain consistency among objects in the whole document. Furthermore, constraint programming techniques have been used to declaratively represent aesthetic knowledge, e.g., basic design principles expressing perceptual criteria (cf. [Graf, 1991]).

2.1 Paradigms and Methodology

Our classification is mainly based on the distinction between two different research fields—also represented by different communities—which have been addressed by constraint processing techniques: the classical field of *constraint satisfaction (CS)* versus the broader area of *constraint-based inference (CBI)*. While CS refers to problem solving *using* constraints, CBI means problem solving *on* constraints. In our application area related work on CS is concerned with the search for all solutions of a CSP such as in combinatorial placement problems in discrete, finite domains, while CBI is mainly concerned with the modification of (sub)solutions such as in interactive graphical interfaces.

In CBI, constraints usually do not state an NP-complete problem. Here the solution of a constraint problem (not a CSP in the traditional sense) frequently is only one issue of a more general design goal that can be achieved by using knowledge about the constraints such as in intelligent presentation systems (cf. WIP [André *et al.*, 1993]). CS is employed to find all, one, or the best solution of a CSP such as in combinatorial placement and optimization problems. It may be seen as a subset of the more general technology of CBI and can also be used as part of it in a particular problem scope.

According to [Borning *et al.*, 1992, Sannella, 1994], this classification reflects the two major approaches: the *perturbation model* and the *refinement model*. Most of the mentioned CBI systems are based on the perturbation model that uses the *value inference* technique. Value-inference constraint solvers assign constant values to some variables and use constraints to determine values of remaining uninstantiated variables allowing other constraints to be selected etc. Here constraints reduce the search space by *propagating values* as soon as possible. This technique enables a data-driven computation, i.e., each constraint has a set of procedures, *methods* in an object-oriented sense, that can be invoked alternately to satisfy the constraint.

The refinement model usually is used in CS systems, such as all finite domain solvers, which are based on *constraint propagation* techniques—consistency or search algorithms—referred to as *label inference* [Faltings, 1994a]. Some approaches have also applied refinement-based solvers to interactive graphical interfaces (cf. [Helm *et al.*, 1995]). For most graphics applications the perturbation approach has been approved rather than the refinement model since (1) it is more efficient and easier to implement, (2) it provides facilities of control to assign specific values to variables, and (3) variables with changeable values enable editing and interaction. However, value inference is rather insufficient for solving combinatorial problems since it mainly works for equational constraints. An additional feature of the value-based approach is provided by the well-known *Blue* algorithms [Freeman-Benson *et al.*, 1990, Sannella, 1995]: they are *structure-based* in the sense that they generate plans which can repeatedly be re-used.

Furthermore, constraint languages can be compared according to the variable domains and constraint types they support, e.g. equations/inequalities/disequations, numeric/symbolic/geometric, extensional/intensional, explicit/implicit, static/dynamic, one-way/multi-way methods, and one-/multi-output. The different approaches also show the trade-off between declarativity/universality vs. efficiency. Moreover, the constraint paradigm can be applied in an active or passive manner to graphical applications: It can serve as a knowledge representation formalism and a computational model as well as an efficient control mechanism.

2.2 Constraint Solving Techniques

The wide field of constraint-based graphics systems subsumes many classes of constraint solving techniques differing in the solving algorithm and propagation, especially methods for constraint optimization and the handling of cycles, over-constrained situations (partial CS), and dynamic input, among others. Here, crucial points include efficiency and incrementality. By and large, we can distinguish the following categories: local propagation, finite domain, numerical, symbolic, geometric as well as application- and domain-specific solvers.

Local propagation solvers Local propagation is the most commonly used constraint solving technique in graphical applications. It has a long tradition in constraint processing and has been investigated extensively (e.g., Sketchpad [Sutherland, 1963] and ThingLab [Borning, 1979]). Local propagation solvers propagate values, states, and degrees of freedom (DOFs). Besides, in the graphics domain, local propagation often works in an "assignment" mode to just inform other variables about some variable instantiation(s). Then, when k ($\leq n$) variables are given in a subnet, we need the instantiation of $k-1$ variables to determine the value of the remaining one; this is also called one-step deduction. (However, some systems even allow the propagation of algebraic expressions—as the one presented in [Gosling, 1983].) Advantages of local propagation for graphical applications are: (1) efficiency, (2) generality, and (3) debugging facilities for variable computation. A typical disadvantage is that local propagation cannot handle cyclic constraint nets and non-equational constraints.

Finite-domain solvers Finite-domain constraint techniques, derived as a subset of general constraint processing, state an important paradigm emerging from AI (cf. [Mackworth, 1992, Kumar, 1992]). They have a wide applicability for highly-complex combinatorial configuration and optimization problems, such as geometric placement. The innovative idea of finite-domain constraint solving in graphics lies in the active use of constraints to prune the search space *a priori* by removing combinations of values which cannot appear together in a solution, in order to avoid combinatorial explosion. Consistency algorithms and search procedures, including looking-ahead and intelligent backtracking methods, are in use to efficiently solve CSPs (cf. [Van Hentenryck, 1989]). Geometric CSPs are treated in more detail in section 4. [Hower, 1995][7] may serve as one representative which ensures global constraint satisfaction in finite domains.

Numeric solvers Numeric solvers, that evolved from research in the field of operations research, use iterative approximation techniques, such as classical relaxation, the gradient, Newton-Raphson or Levenberg-Marquardt method [Press *et al.*, 1994], for solving large sets of linear and non-linear constraints among real-valued variables. But usually they are not efficient enough for graphical applications and may not converge on a solution for all constraints. For instance, the Newton-Raphson iteration (e.g., used in Juno [Nelson, 1985, Heydon and Nelson, 1994] and Converge [Sistare, 1991]) changes all constrained variables at each iteration step. The Levenberg-Marquardt iteration, a least-squares method that is used in Chimera [Kurlander and Feiner, 1993], achieves better numerical efficiency. Frequently, iterative numerical techniques are employed in situations when local propagation fails (because of cycles or simultaneous constraints).

Further examples of numerical solvers are the systems TRIP/COOL [Kamada, 1989][8], QOCA [Helm *et al.*, 1995], and Bramble [Gleicher, 1992]. Numerical solvers which are limited to linear, algebraical constraints are provided by, for example, METAFONT [Knuth, 1986] and IDEAL [van Wyk, 1982]—both systems do not have a WYSIWYG interface. Numerical solvers for a fixed set of non-linear constraints are used within the drawing programs Briar [Gleicher and Witkin, 1994], Converge [Sistare, 1991], and IntelliDraw, the Aldus's commercial drawing program—they are WYSIWYG but do not have a programming view.

Symbolic solvers Another class of solvers —so-called graph transformation solvers that use algebraic term rewriting rules—exploit symbolic techniques. For example, the solver in the system GITS [Olsen Jr. and Allan, 1990] uses a rather simple approach by referring to a table of pre-compiled constraints to resolve cycles. The graphical editor Magritte [Gosling, 1983], that supports only a small set of algebraic constraints, exploits a more elaborated approach of replacing cycles through algebraic transformations. General constraint languages based on term rewriting solvers have also been used in the context of graphical applications, for example the systems Bertrand [Leler, 1988], Equate [Wilk, 1991], and Siri [Horn, 1992].

Geometric solvers Here, we point to the so-called geometric constraint engine by [Kramer, 1992], which provides a geometric constraint solver that allows direct inferences along DOFs of geometric entities.

Domain-specific solvers Many graphics and visualization systems make use of application-specific and domain-specific constraint solvers that restrict the represention of and reasoning about constraints to a certain domain. Many of them support equations

---

[7] [Hower and Jacobi, 1994] illustrates a distributed approach

[8] you may also consult [Takahashi *et al.*, 1995]

and inequalities among real-valued variables and provide numeric as well as symbolic techniques for solving them. But usually they are limited to problems that can be formalized as sets of constraints within the corresponding domain. Examples are the graphical editors Bramble [Gleicher, 1992] and QOCA [Helm *et al.*, 1995] as well as a surface modelling tool developed by [Welch and Witkin, 1992].

2.3 Classification of the Different Viewpoints

The various approaches strongly depend on the structure and complexity of the specific application domain. For example, we may consider the graphics type (e.g., rectangles/triangles, 2D informational graphics such as graphs, flowcharts, and networks, 3D realistic graphics, multimedia items, and animation) and the question whether dynamic/temporal information and user interaction is required. Furthermore, we can distinguish between applications where constraints are generated or inferred automatically on the fly and others that require user-defined constraints.

Hence, constraint formalisms and constraint solving algorithms have been used in different forms and qualities. According to major differences in the use of constraints, we will classify constraint-based graphics approaches as follows: (1) general-purpose constraint programming languages and systems (extending programming languages by constraint formalisms) that sometimes include domain-specific constraint libraries, (2) general constraint solvers, (3) constraint solvers integrated in user interfaces, and (4) pure applications that, besides others, make use of constraints.

In many cases general local propagation solvers such as DeltaBlue [Freeman-Benson *et al.*, 1990] and SkyBlue [Sannella, 1995] have been used in a number of applications, including the user-interface toolkit MultiGarnet [Sannella, 1994], the constraint imperative programming language Kaleidoscope [Freeman-Benson, 1991], the drawing program CoolDraw [Freeman-Benson, 1993], the 3D animation system TBAG [Elliott *et al.*, 1994], and the virtual reality system VB2 [Gobbetti and Balaguer, 1993][9].

Frequently, constraint solvers and imperative programming languages have been integrated by modifying an object-oriented system. For example, Garnet [Myers *et al.*, 1990], Rendezvous [Hill *et al.*, 1993, Hill, 1993], and PICASSO [Rowe *et al.*, 1991] use the facilities of an object-oriented system in order to associate constraints to particular object slots and invoke a local propagation solver when these slots are changed.

In contrast to general constraint satisfaction, many approaches in this area use constraints only on a pre-processing stage and exploit other mechanisms (e.g., optimization algorithms) for problem solving. In the following, we will compare the bulk of local propagation solvers in interactive graphical applications to the diverse set of other techniques, especially constraint satisfaction. For this bibliographical survey, we have chosen a classification following the different application areas.

3 Constraint-based Graphical Applications

Related work on geometric layout, graphics, and visualization issues is immanent to a large spectrum of application domains which includes the following categories: interactive graphical (drawing and editing) systems, user interface management systems (UIMSs), layout and

---

[9] see also [Balaguer and Gobbetti, 1995]

display managers, intelligent CAD systems, multimedia presentation systems, programming-by-example, 2D/3D computer graphics and animation systems, and visual programming languages. Furthermore, there are a lot of graphical applications which apply constraints only rudimentarily, while there seems to be less effort on principled constraint-based graphics systems.

3.1 Graphical Editing and Drawing

Since constraint satisfaction techniques have become more and more sophisticated during the last decade, and with the growing availability of advanced graphics hardware, there has been an upward trend in applying constraint technology to computer graphics. Thus, most of the related work on applications of constraint languages and systems has been done in the area of interactive graphical interfaces, especially in geometric layout. Much of the recent work on geometric layout in interactive graphics systems also comprises new constraint processing techniques. Obvious applications of constraint-based drawing programs are CAD systems.

Based on Sketchpad, the pioneering system in constraint-based languages and object-oriented programming as well as in interactive graphics, subsequent ideas led to the constraint-oriented simulation laboratory ThingLab. Further research activities on constraint-based graphics systems focussing geometric layout include the systems IDEAL, Magritte, Juno, and Bertrand. These early approaches supported only a limited set of graphic primitives and constraint types. Most of them, including Sketchpad and ThingLab, did not use a real declarative representation of constraints. Juno-2 [Heydon and Nelson, 1994] is a constraint-based double-view drawing editor which combines numeric and symbolic techniques. It enables solving of cyclic constraint nets and provides a powerful declarative language in order to enrich its extensible class of non-linear constraints. Constraint solvers used for surface modelling in user interfaces and in graphical editors [Gleicher, 1993, Helm *et al.*, 1995] are mostly limited to problems where the constraints can be expressed in terms of numeric relations. LayLab's constraint-based graphical editor InLay [Graf and Neurohr, 1995] supports interactive multimedia layout, authoring, pre/post-editing, and beautification tasks. TRIP's [Kamada, 1989, Kamada and Kawai, 1991, Takahashi *et al.*, 1991, Miyashita *et al.*, 1992] integrated constraint-based object layout system COOL formats graphs by minimizing the total energy of "springs" between the nodes (expressed as a quadratic function of the node positions). Further drawing and editing programs are CoDraw [Gross, 1992], CoolDraw, and Oak [Tonouchi *et al.*, 1992].

3.2 Intelligent CAD

Graphical constraints [Szwillus, 1994] already have a long history in the area of artificial intelligence in design [Pfefferkorn, 1975]. [Sutherland, 1963] already proposes to use graphics as a medium of communication and illustrates the "constraint"-oriented approach from the first sketch to its complete finish. [Johnson, 1963] and [Thennarangam and Singh, 1994] cope with 3D modelling. Constraint processing as a knowledge representation paradigm as well as an inference mechanism is used in a broad spectrum of applications; a more recent paper is, for instance, [Tanimoto, 1993] in the area of configuration (of user interfaces). [Klein, 1994] also points to interesting issues in the frame of configuration problem solving. [Young *et al.*, 1991] and [Nagai and Terasaki, 1993] prefer a constraint-based modelling in engineering. [Gross *et al.*, 1988] illustrates the use of a "constraint manager" system in the wider area of "design". [Shimada *et al.*, 1989] suggests a constraint-based framework for intelligent CAD systems. [Liu and Popplestone, 1990] lists several sorts of constraints. [Kalra and Barr, 1990] proposes that the system should maintain the relations among objects. In [Nourani and Magalhães, 1993] the user checks global consistency. [Phillips *et al.*, 1990] rises the problem of both under- and over-specification of constraints. [Hel-Or *et al.*, 1994] proposes the use of inexact satisfaction of constraints as a means to express a general outline to avoid overspecification. [Veltkamp, 1995] prefers an incremental technique (instead of the usage of a constraint logic programming system) for interactive design purposes and deals with under-constrained situations. [Olsen Jr. and Allan, 1990] prefers interactive techniques, too. [Rappoport, 1993] points to a higher level interface when indicating the constraints among geometrical objects. [Tsang, 1990] briefly mentions the notion of k-consistency which means that it is possible to consistently assign values to k ($\leq n$) variables of the CSP.

3.3 Automated Layout Design of Graphical and Multimedia Presentations

Since the layout of a presentation often conveys the structure, intention, and significance of the underlying information, besides many other approaches (such as rule-/case-based reasoning, simulated annealing, genetic algorithms) especially constraint-based layout design facilities play a crucial role for the design of automatically generated as well as user-defined graphical and multimedia presentations. Recent approaches investigate the use of constraint-based formalisms for declaratively representing graphical design knowledge in order to produce aesthetical as well as adaptive and coherent layouts.

Marks investigated the encoding of arc-node diagrams in his ANDD system that grouped nodes sharing common graphical values to re-inforce perception of graphical properties. Besides a rule-based realization and an expensive genetic algorithm, a constraint-driven approach is detailed in [Dengler *et al.*, 1993] that uses a generalization of the simple "mass-spring" layout technique for constraint solving.

Graf has developed the constraint-based multimedia layout framework LayLab [Graf, 1992, Graf, 1993, Graf, 1995a] which incorporates a collection of two dedicated solvers including the local propagation solver SIVAS+ and the finite domain solver FIDOS. SIVAS+ extends the constraint hierarchy solver DeltaBlue by indirect reference constraints in order to allow for dynamic input and user interaction; FIDOS encodes placement heuristics as domain-specific constraint abstractions and uses forward checking for efficiently solving them. Furthermore, LayLab deals with page layout as a rhetorical force, influencing the intentional and attentional state of the user [Graf, 1995b].

3.4 Design of User Interfaces

Rapidly expanding activities on graphical and multimedia interface toolkits have addressed the area between visual interfaces and constraint-based systems. User interface construction systems are similar to drawing programs since they have to determine size, position, and behaviour of the interface items. In user interface tools, by far the most common technique to solve constraints is *local propagation*. It has the advantage of being rather efficient and very general, which is required in user interface construction.

Many user-interfaces design systems have provided integrated constraint solvers, including the prominent examples ThingLab II [Maloney *et al.*, 1989], Garnet, and Rendezvous.

ThingLab II was concerned with extensions of the original ThingLab supporting constraint hierarchies, incremental compilation, and graphical facilities for defining new kinds of constraints.

The constraint solver in Garnet supports only one-way constraints of the same strength but provides some additional features [Vander Zanden *et al.*, 1991]: indirect reference constraints of source variables (experimenting both with a lazy evaluation and an eager evaluation algorithm), tolerance of side effects and a defaulting mechanism for dealing with uninitialized variables. This full-fledged toolkit has achieved considerable flexibility when extended to MultiGarnet by multi-way constraints and hierarchies.

The Rendezvous language and architecture which is developed for distributed applications (as in computer supported cooperative work) also allows only one-way constraints and supports for indirection in sources and targets (implemented with an eager algorithm), tolerance of side effects, and a special mechanism for handling uninitialized variables. The user interface construction tools TRIP [Kamada and Kawai, 1991], TRIP2/DeltaTrip (based on DeltaBlue) [Takahashi *et al.*, 1991], and TRIP3 [Miyashita *et al.*, 1992] map abstract objects and relations onto sets of graphical items and geometric relations for visualization.

The user interface editor Opus [Hudson and Mohamed, 1990] provides several display techniques for constraints that reduce the typical cluttering effect of display objects.

The use of constraints with priorities for specifying inter- and intra-window relations in systems for window management and arrangement is illustrated in the non-hierarchical window manager RTL/CRTL [Cohen *et al.*, 1986] (that is partially rule-based) and the Constraint Window System CWS [Epstein and LaLonde, 1988]. The latter uses constraint hierarchies in a layout system for Smalltalk windows, e.g., to define relations among the canvas size, window size, and scaling factors. Like a module of a UIMS, Chisel [Singh and Green, 1989] creates interactive screen layouts from information about dialog requirements, display devices, and user preferences. Here the placing of so-called interaction techniques is based on constraints which are associated with a measure of specificity.

Further user interfaces and interface design systems that use constraints include GROW [Barth, 1986], Peridot [Myers, 1988], Lapidary [Myers *et al.*, 1989], MEL [Hill, 1991], GITS, Animus [Duisberg, 1987], and the FilterBrowser user interface construction tool [Ege *et al.*, 1987]. The references [Borning *et al.*, 1992, Sannella, 1994, Fron, 1994] contain additional pointers to constraint-based user interfaces.

3.5 Programming-by-Example and Beautification

An increasing number of interface-design systems mostly based on a graphical editor have been developed during the last few years to make the interface design and constraint specification process more efficient and comfortable than with conventional techniques. Especially, visual programming and graphical editing tasks have been addressed by constraint-by-example techniques to enhance the interactive specification and editing of graphical presentations.

Some systems eliminate any need for the user to specify constraints explicitly. The user sketches the graphical elements and layout and the system infers the corresponding constraints to be applied and maintained. Here, constraints provide a means of stating layout requirements; e.g., the Peridot system deduces constraints automatically as the user demonstrates the desired behaviour. The Chimera editor designed by Kurlander and Feiner supports two functionalities by the help of constraints: (1) It is able to infer constraints from multiple snapshots [Kurlander and Feiner, 1993] and (2) allows interactive graphical search and replace operations [Kurlander and Feiner, 1992].

The Metamouse system [Maulsby *et al.*, 1988] is a demonstrational interface for graphical editing tasks within a drawing program. The user can specify a procedure by performing an example execution trace, manipulating objects directly on the screen and creating graphical tools. The spreadsheet-based system NoPumpG [Lewis, 1990] allows for an easy creation of (animated) graphics without any need for programming.

Another important research topic deals with drawing and beautification of graphs. Early work includes an automatic beautifier for line drawings and illustrations that makes use of automatic constraint generation as part of the 2D graphics editor PED is described in [Pavlidis and Wyk, 1985]. In this approach, vertices are re-drawn precisely following certain constraints, such as nearly adjacent or coincident lines, that are inferred from an initial sketch which are imposed on the beautified version. But most of the drawing programs are limited to syntactical constraints. An extensive overview about algorithms for drawing graphs—some of them are constraint-based—provides the annotated bibliography by Di Battista and colleagues [Di Battista *et al.*, 1994].

3.6 Constraint-Based Animation and Visual Programming

Some systems incorporate the notion of time by allowing the representation and/or processing of temporal constraints: The animation system Animus (see also [Borning and Duisberg, 1986]) is one of the first systems that allows for easy construction of an animation with minimal concern for lower-level graphics programming. Here temporal constraints are used to describe the appearance and structure of a picture, as well as how those pictures evolve in time. TBAG, that is also based on SkyBlue, is a toolkit for the creation of interactive, dynamic 3D graphics. The graphical toolkit Bramble supports user interaction through the technique of differential manipulation and represents also lights and cameras as constrainable objects. In an application of the Kaleidoscope language temporal constraints are used to update the display of graphical objects which are manipulated by mouse actions interactively and maintain their consistency requirements.

Other research in the area of constraint-based layout in dynamic and animated settings is concerned with topics like *program visualization* and *visual programming* (see [Glinert, 1990a, Glinert, 1990b] for an overview). Examples of program visualization systems are the visual programming environment Fabrik [Ingalls *et al.*, 1988] and the systems Gelo [Duby *et al.*, 1989] and its 3D extension PLUM [Reiss, 1993], which provide general-purpose packages to visualize information about programs. Here, the layout of linked hierarchical objects is described via constraints. Gelo includes predefined data views and allows the graphical specification of topological constraints by the user. [Cruz, 1995] presents the U-term language, a declarative language with a visual syntax for specifying the display of database objects that is compatible with an object-oriented framework. The environment Escalante [McWhirter and Nutt, 1994] supports the rapid prototyping as well as automated generation of complex visual language applications. LayLab has also been extended to achieve topologically consistent layouts in visual program design [Graf and Neurohr, 1995].

By far the largest effort on algorithm animation exhibits the system BALSA [Brown, 1988] that creates animated courseware for an electronic classroom and provides an extensive library of animated presentations. Further algorithm animation system are TANGO [Stasko, 1990], Zeus [Brown, 1991], and a gestural interface by [Duisberg, 1990] for visual programming of program visualizations. The generation and animation of virtual worlds is treated in [Thalmann and Thalmann, 1992].

4 Geometric Layout as a CSP

As with many other interesting artificial intelligence design problems, geometric layout as a highly-complex configuration task can be formalized as a CSP. In addition to research on constraint-based inference systems (described previously), much work was done on the modelling of combinatorial geometric problems in a finite discrete search space and on the development of constraint satisfaction techniques for solving them efficiently [Van Hentenryck, 1989, Henz *et al.*, 1995]. Here the main problem consists of finding any solution that satisfies all topological and geometrical restrictions with regard to certain optimization criteria. So the design of an optimal geometric layout can be treated as a combination of a general search problem and an optimization problem. In contrast to other configuration or scheduling problems, an optimal geometric layout frequently has to satisfy certain additional aesthetic criteria. Generally, the problem with most of the constraint-based approaches to space planning problems is that they are (1) only adapted rudimentarily to real applications, (2) are very specific to a particular problem or application domain, (3) or their run-time efficiency is inadequate. Therefore, many previous constraint techniques could only be applied to a narrow set of problems.

Since efficient constraint satisfaction is crucial when using constraints for graphical tasks, general constraint (logic) programming languages currently available seem to be inconvenient for handling real-world placement problems in realistic domains because of their increased runtime and the required memory capacity, especially when they are restricted to the use of constraint techniques only—although they have already proven their adequacy for such problem classes for academic examples (cf. [Müller *et al.*, 1995])—and thus cannot compete with specialized layout algorithms. Therefore, the CHIP sytem [Van Hentenryck, 1989] has been extended by a new primitive constraint in finite domains, the so-called *cumulative constraint* [Aggoun and Beldiceanu, 1993], that allows an improvement in the efficiency of CLP languages for solving hard scheduling and placement problems.

A number of CLP languages have been marketed as commercial systems with a C-like syntax and are frequently applied to solve combinatorial problems (cf. [Cras, 1993, Fron, 1994, Jaffar and Maher, 1994]). For example, the language CHARME, that is based on a procedural framework, arose from CHIP by omitting its logic programming part. Ilog Solver establishes a library of constraint algorithms, designed to work with C++ programs, which has also been applied to geometric placement problems.

Some further guidelines for solving practical CSPs are given in [Van Hentenryck, 1989]. Here it has been shown by the example of the cutting-stock problem that complex geometric placement problems require a combination of an algorithmic and a constraint-oriented treatment. The algorithmic approach is used to generate configurations that can easily be used by the constraint solver to satisfy and optimize the constraints. In a similar way, in the commercial application YPPS of the LayLab layout tool (cf. [Graf, 1995a]), its finite domain solver FIDOS is enriched by a special constraint abstraction that represents the heuristic placement of display advertisements (similar to the cumulative constraint) in order to allow for an automated logical pagination of yellow-pages telephone directories.

The following research has also addressed the problem of solving geometric layout as a CSP: [Tokuyama *et al.*, 1991], [Charman and Trousse, 1993], and [Aggoun and Beldiceanu, 1993] deal with rectangles. [Hower *et al.*, 1993][10] employs a global constraint satisfaction solver where all the various layout possibilities get generated by a bottom-up approach—without the need of an optimization function. Please consult [Hower, 1996]. There, in order to reformulate a problem to get a discrete search space, the reasoning takes place along boundaries of value intervals. (See also [Elliman *et al.*, 1993] where interval domains represent ranges of values.) A different approach (to obey the *non-overlapping* constraint) favouring evolutionary computing is demonstrated in [Hower *et al.*, 1995] where even triangles can be handled.[11] [Lozano-Pérez, 1983] treats more general polygons. [du Verdier, 1993] just copes with binary constraints.

5 Special Constraint Solver Features in Graphical Environments

Today's real-world applications cannot be solved by standard constraint technology. Therefore, sophisticated constraint solvers provide special mechanisms to handle relaxation (cf. the CP95 workshop on over-constrained systems), constraint nets containing cycles, incremental compilation, graphical constraint specification, variable indirection, constraint libraries, efficient consistency and search algorithms, among others.

5.1 Priority Weights

In [Badler *et al.*, 1987] and [Ege, 1989] weights are assigned to the constraints; the latter one additionally reports on the usefulness of constraint processing in connection with user interfaces. [Freeman-Benson *et al.*, 1990] also orders the constraints partially according to their importance to be fulfilled; there, it is a distinction between "required" constraints which must be satisfied and "preferred" ones with lower-level priorities. It performs local propagation; however, no soft relaxation is possible—the constraints will get deleted completely; please consult [Maloney, 1991], [Sannella *et al.*, 1993], and [Sannella, 1994] w.r.t. further developments. [Darses, 1990] already mentions (there on p. 135) that "an increasing number of CAD systems have been developed from the constraint satisfaction paradigm"; it proposes a least-commitment strategy along with an ordering of the constraints according to a priority hierarchy. [Emmerik, 1990] recommends to assign priorities to the constraints to allow some guidance inside the procedure during the detection of an inconsistency; additionally, it refers to the usage of a constraint-based specification not only for geometric modelling (where constraints contain the DOFs) but also for the design of user interfaces. [Faltings *et al.*, 1992] deals with constraint variables with value intervals employing a justification-based reason maintenance system integrated into an intelligent CAD system. It orders the constraints according to some priority and distinguishes between irrevocable constraints and those ones which may be subject to further modification. It distinguishes constraints into two categories: "preference constraints" and "fixed constraints". "Preference constraints" belong to various levels in a hierarchy where the higher levels deal with the more global layout and the lower ones know the details. In case of inconsistency the informed constraints belonging to the lower levels of the abstraction hierarchy are tried to get fulfilled whereas the more abstract

---

[10] section 4 of [Berling *et al.*, 1993] (pp. 27/28) presents a more detailed german description

[11] Thanks to Anna Thornton for her encouragement during lively discussions after the CoPiCAD-94 workshop ([Hower *et al.*, 1994a]) to try such an approach.

constraints have to be adjusted. "Fixed constraints" however must be obeyed first on the higher levels according to the global layout restrictions; in case of inconsistency low-level constraints must be withdrawn in order to try to achieve just small changes. [Freeman-Benson, 1991] describes a hierarchy of constraint priorities and states that constraint systems are often in use for graphical layout and interaction. It mentions that a parallelization would be desirable, and furthermore, it realizes an integration of imperative as well as declarative features; see also [Lopez *et al.*, 1994].

5.2 Object-Orientation

The paper [Borning and Duisberg, 1986] promotes the object-oriented view—also due to its modularity—which should be a good basis in changing situations to know what happens, and it illustrates how helpful the constraint processing paradigm is w.r.t. the construction of user interfaces. [Sunde, 1988] models the geometric relations via object-oriented constraints denoting relations among objects; changing the constraint specification causes the system to compute the effects. [El Dahshan and Barthes, 1989] favours an object orientation and proposes to introduce priorities to order the constraints. [Fertey *et al.*, 1990] uses an object-oriented representation in a 3D environment. [Arbab and Wang, 1991] performs geometric reasoning to catch the domain knowledge in a better way. It also prefers the object-oriented paradigm and proposes to manage on local changes within an incremental technique when constraints are added which is important in interactive CAD—instead of algebraic methods. Furthermore, it allows to deal with some kind of qualitative constraints. [Hill, 1991] tries to avoid unnecessary recomputations. [Rankin, 1995] reports on good experience with the combination of constraint processing and object-oriented programming; furthermore, it states that the algorithm used is suitable for a parallel architecture. [Du, 1995] maintains invariant relationships in geometrical modelling via CAD-oriented constraint processing in an object-oriented system. [Paltrinieri, 1995] features object-oriented abstraction to visualize the constraint network.

5.3 CSP Modification and Inconsistency Handling

The work [Brüderlin, 1986] deals with 2- and 3-dimensional objects; in case of an inconsistency of the underlying CSP the user has to replace a constraint or even to delete one. [Serrano and Gossard, 1987] mentions the constraint-oriented philosophy of design and also allows constraints to get modified. [Vander Zanden, 1988] reports on the need to have an update mechanism which allows to propagate the modifications just along those constraints which are directly influenced by minimal change; it proposes an incremental approach in order to keep the alterations as local as possible ("principle of least astonishment") in case of constraint modifications.[12] Furthermore, it provides a comfortable overview regarding user interface management systems and compiles literature w.r.t. constraint-based graphic systems until 1988. [Keirouz *et al.*, 1990] recalls the fact that during the design phase one would like to be able to change the constraints. [Yamaguchi and Kimura, 1990] proposes just to perform the necessary changes when inconsistency occurs which would require a kind of

---

[12] [Newbery Paulisch, 1993] also intends to yield just small effects on the (re)computation due to small changes in the layout; in the case of an inconsistency, however, constraints get completely deactivated. Additionally, it mentions the feature of adaptability when using an object-oriented design. [Szwillus, 1993] pursues the "least astonishment" principle, too.

meta-level control; furthermore, it illustrates geometric reasoning based on geometric constraints for variational geometry. [Murtagh and Shimura, 1990] reports on a redesign just by the adjustment of only a few parameters. [Helm *et al.*, 1995] incrementally adds and deletes constraints in an object-oriented architecture to perform "re"- and "un-do" operations in a constraint-based graphical editing system.[13] ([Rosendahl, 1993] points to "un-do" operations in CAD.) [Baykan and Fox, 1991] commends the constraint-directed search as a principal problem solving mechanism for intelligent CAD and proposes some abstraction to reduce the complexity; when inconsistency occurs the user shall relax the constraints—you may consult [Flemming *et al.*, 1992] (discussing local "path" consistency—coping with at most 3 variables at once), too. [Žalik *et al.*, 1992] performs local propagation during its incremental design and wishes an immediate visualization of the effects; the changes should be performed just where necessary and a parallel computation would be recommendable. [Pineda, 1992] reasons symbolically when changes are needed. [Matoušek, 1994] discards some constraints. [Bahler *et al.*, 1994] handles negotiated conflict resolution in design.

6 Diverse Techniques

The article [Cournarie and Beaudouin-Lafon, 1995] reports on local constraint propagation in graphical user interfaces. It identifies problems with an object-oriented implementation and introduces a "prototype"-based model; it also proposes to employ (several) special purpose constraint solvers in a modular way. [Freeman-Benson and Borning, 1992] requires one corresponding constraint solver for each special domain to foster the integration of objects and constraints. [Nelson, 1985] translates geometric constraints into numerical ones to solve them numerically; please further consult also [Heydon and Nelson, 1994]. [Sriram and Maher, 1986] represents the constraints as objects or as production rules. [Duisberg, 1986] refers to the time aspect of the computation; it mentions the fundamental hardness of the CSP as well as the fortunate fact that in some specific applications it turns out that it is still feasible. [Witkin *et al.*, 1987] expresses constraints as "energy" functions; in the case of a local (not global) minimum user interaction is required. ([Steinberg, 1987] also prefers control by the user.) [Iba and Inoue, 1991] and [Owen, 1991] use an algebraic method. ([Kondo, 1992] employs algebraic methods with potentially high complexity.[14]) [Agrawal *et al.*, 1993] employs an equation system. [Buchanan and de Pennington, 1993] illustrates the use of computer algebra. [Isaacs and Cohen, 1987] already propagates along DOFs. [Kramer, 1992] (reviewed also in [Bouzy, 1993] and [Sacks, 1993]) allows to reformulate the problem so that the search space is discrete, and it declines to prefer iterative methods because these ones cannot exploit intelligently the case when just a small change in the problem specification has to be considered; it therefore prefers the "DOF analysis" incorporating a constraint-based geometric reasoning. It incrementally reduces the DOFs and proposes symbolic reasoning taking into consideration geometric knowledge instead of blind manipulation of algebraic equations. Domain-specific strategies are employed, and it is stated that the complexity of their method is polynomial in the number of constraints. (In general there are three translational DOFs, three rotational ones and sometimes also di-

---

[13] [Ruttkay, 1995] presents an editor based on constraint processing, too.

[14] "The analysis of the complexity of this method is left for future work." ([Kondo, 1992], p. 146)

mensional DOFs.[15]) You may further consult [Kramer, 1994].[16] [Bouma *et al.*, 1995] reports on geometrical constraint solving, too. [Solano and Brunet, 1994] comments on parametric design. [Chung, 1994] promotes constraint-based variational design. [Pabon *et al.*, 1992] integrates parametric geometry and variational modelling. [Karsenty *et al.*, 1992] tries to identify/infer the constraints. [Kass, 1992] uses interval arithmetic—see also [Navinchandra and Rinderle, 1989]. [Guan and Friedrich, 1992] proposes a reasonable inconsistency handling in its object-oriented design and differentiates between pre-defined parameters and changeable ones; its main contribution is the initiation of “fuzzy constraint satisfaction”. [Towhidnejad *et al.*, 1993] uses constraint processing techniques in the area of automatic knowledge acquisition of CAD data. [Guesgen and Hertzberg, 1993] points to the interesting relationship of spatial and temporal constraint-based reasoning. [Thornton, 1994] discusses genetic algorithms and simulated annealing approaches. [Tsuchida, 1995] deals with tree-like diagrams where the nodes are rectangles. [Tamassia, 1995] illustrated the need of the management of constraints in graph drawing.

7 Final Remarks

The intent of the present paper is to provide an extensive collection of connected work (hopefully including also the most recent material) for the first time—which up to now is not yet available otherwise to such an extent. A thorough elaboration of the constraint-oriented view enables the designer and the user of a system to naturally express the meaning of the intended message in mind such that the computer may support the human in maintaining even the semantics of the implementation—a welcome feature of sensible CAD.

Acknowledgements

We would like to thank the numerous reviewers who took their time to comment on our draft; this version really gained from their feedback. Thanks also to Randy Goebel for his valuable remarks and Ingar Uhe who struggled with our English. Additional thanks go to Tomihisa Kamada for the non-material support of this piece of work.

---

[15] Rigid bodies do not have such dimensional degrees of freedom.

[16] [Celaya Llover, 1992] also works in the kinematic domain.

References

[Aggoun and Beldiceanu, 1993] Abderrahmane Aggoun and Nicolas Beldiceanu. Extending CHIP in order to solve complex scheduling and placement problems. *Mathematical and Computational Modelling*, 17(7):57–73, 1993. Pergamon Press Ltd.

[Agrawal *et al.*, 1993] R. Agrawal, G. L. Kinzel, R. Srinivasan, and K. Ishii. Engineering Constraint Management Based on an Occurrence Matrix Approach. *Transactions of the ASME, Journal of Mechanical Design*, 115(1):103–109, March 1993.

[Amarel *et al.*, 1991] Saul Amarel, Alvin Despain, Penny Nii, Louis Steinberg, Marty Tenenbaum, and Peter Will. Panel on AI and design. In [Mylopoulos and Reiter, 1991], pages 563–565, 1991.

[André *et al.*, 1993] E. André, W. Finkler, W. Graf, T. Rist, A. Schauder, and W. Wahlster. WIP: The automatic synthesis of multimodal presentations. In Maybury [1993], pages 75–93.

[Arbab and Wang, 1991] Farhad Arbab and Bin Wang. A Geometric Constraint Management System in Oar. In [ten Hagen and Veerkamp, 1991], pages 205–231, 1991.

[Badler *et al.*, 1987] Norman I. Badler, Kamran H. Manoochehri, and Graham Walters. Articulated Figure Positioning by Multiple Constraints. *Computer Graphics and Applications*, 7(6):28–38, June 1987. IEEE.

[Bahler *et al.*, 1994] D. Bahler, C. Dupont, and J. Bowen. An axiomatic approach that supports negotiated resolution of design conflicts in concurrent engineering. In [Gero and Sudweeks, 1994], pages 363–379, 1994.

[Bajcsy, 1993] Ruzena Bajcsy, editor. *IJCAI-93, Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence*, Chambéry, Savoie, France, August 28 – September 3, 1993. IJCAII, distributed by Morgan Kaufmann Publishers, San Mateo, California, USA.

[Balaguer and Gobbetti, 1995] Jean-Francis Balaguer and Enrico Gobbetti. Supporting Interactive Animation Using Multi-way Constraints. In [Veltkamp and Blake, 1995], pages 17–27, 1995.

[Barth, 1986] P. Barth. An object-oriented approach to graphical interfaces. *ACM Transactions on Graphics*, 5(2):142–172, 1986.

[Baykan and Fox, 1991] Can A. Baykan and Mark S. Fox. Constraint Satisfaction Techniques for Spatial Planning. In [ten Hagen and Veerkamp, 1991], pages 187–204, 1991.

[Belli and Radermacher, 1992] Fevzi Belli and Franz Josef Radermacher, editors. *Industrial and Engineering Applications of Artificial Intelligence and Expert Systems*, volume 604 of *Lecture Notes in Artificial Intelligence, Subseries of Lecture Notes in Computer Science*. Proceedings, Springer-Verlag, Berlin/Heidelberg, June 9–12, 1992. 5th International Conference, IEA/AIE - 92, Paderborn.

[Berling and Hower, 1993] Roland Berling and Walter Hower. Solving Geometrical Constraint Systems. *AI Communications*, 6(3/4):232–234, Sept./Dec. 1993. IOS Press, Amsterdam, The Netherlands. Book review of [Kramer, 1992].

[Berling *et al.*, 1993] Roland Berling, Chun Du, Walter Hower, and Manfred Rosendahl. Modellierung geometrischer Constraints für CAD-Anwendungen. In Klaus Kansy and Peter Wißkirchen, editors, *Neue Architekturkonzepte zur Gestaltung graphischer Systeme*, pages 19–29. Proceedings, GMD-Studien Nr. 223, Gesellschaft für Mathematik und Datenverarbeitung mbH, Sankt Augustin; GI-Workshop, Bonn, November 18/19, 1993.

[Borning and Duisberg, 1986] Alan Borning and Robert Duisberg. Constraint-Based Tools for Building User Interfaces. *ACM Transactions on Graphics*, 5(4):345–374, October 1986.

[Borning *et al.*, 1992] A. Borning, B. Freeman-Benson, and M. Wilson. Constraint hierarchies. *LISP and Symbolic Computation: An International Journal*, 5(3):223–270, 1992.

[Borning, 1979] A. Borning. *ThingLab - A Constraint-Oriented Simulation Laboratory.* PhD thesis, Department of Computer Science, Stanford University, Stanford, CA, 1979.

[Borning, 1981] Alan Borning. The Programming Language Aspects of ThingLab, a Constraint-Oriented Simulation Laboratory. *ACM Transactions on Programming Languages and Systems*, 3(4):353–387, October 1981.

[Borning, 1994] Alan Borning, editor. *Principles and Practice of Constraint Programming*, volume 874 of *Lecture Notes in Computer Science.* Proceedings, Springer-Verlag, Berlin/Heidelberg, May 2–4, 1994. Second International Workshop, PPCP '94, Rosario, Orcas Island, WA, U.S.A.

[Bouma *et al.*, 1995] William Bouma, Ioannis Fudos, Christoph Hoffmann, Jiazhen Cai, and Robert Paige. Geometric constraint solver. *Computer-Aided Design*, 27(6):487–501, June 1995. Butterworth-Heinemann, Elsevier Science Ltd.

[Bouzy, 1993] Arnaud Bouzy. Book review of *Solving Geometric Constraint Systems, A Case Study in Kinematics* (G A Kramer, The MIT Press, Artificial Intelligence series, Cambridge, Massachusetts, USA / London, England, UK, 1992). *Computer-Aided Design*, 25(10):678–679, October 1993. Butterworth-Heinemann Ltd.

[Brown, 1988] M. H. Brown. *Algorithm Animation.* ACM Distinguished Dissertations. MIT Press, Cambridge, MA, 1988.

[Brown, 1991] M. H. Brown. Zeus: A system for algorithm animation and multi-view editing. In *Proceedings of the 1991 IEEE Workshop on Visual Languages*, pages 4–9, Kobe, Japan, 1991.

[Brüderlin, 1986] Beat Brüderlin. Constructing Three-Dimensional Geometric Objects Defined by Constraints. In F. Crow and S. M. Pizer, editors, *Workshop on Interactive 3D Graphics*, pages 111–129, Chapel Hill, North Carolina, October 23/24, 1986. ACM SIGGRAPH, New York, NY, U.S.A. Conference proceedings.

[Buchanan and de Pennington, 1993] S Alasdair Buchanan and Alan de Pennington. Constraint Definition System: a computer-algebra based approach to solving geometric-constraint problems. *Computer-Aided Design*, 25(12):741–750, December 1993. Butterworth-Heinemann Ltd.

[Catarci *et al.*, 1992] T. Catarci, M. F. Costabile, and S. Levialdi, editors. *Advanced Visual Interfaces, Proceedings of the International Workshop AVI '92.* World Scientific Series in Computer Science - Vol. 36. World Scientific Press, Singapore, 1992.

[Celaya Llover, 1992] Enric Celaya Llover. *Geometric Reasoning for the Determination of the Position of Objects Linked by Spatial Relationships.* PhD thesis, Institut de Cibernètica, Dept. de Llenguatges i Sistemes Informàtics, Universitat Politècnica de Catalunya, Barcelona, Spain, September 1992. "Tesi doctoral".

[Charman and Trousse, 1993] Philippe Charman and Brigitte Trousse. EAAS: environnement d'aide à l'aménagement spatial. In EC2, editor, *Avignon '93*, volume 1, pages 613–624, France, May 24–28, 1993.

[Chung, 1994] Jack C. H. Chung. Constraint-based variational design. In [Hower *et al.*, 1994b], pages 44–54, 1994.

[Cohen *et al.*, 1986] E. Cohen, E. Smith, and L. Iverson. Constraint-based tiled windows. *IEEE Computer Graphics and Applications*, 6(5):35–45, 1986.

[Cournarie and Beaudouin-Lafon, 1995] Eric Cournarie and Michel Beaudouin-Lafon. ALIEN: A Prototype-Based Constraint System. In [Laffra *et al.*, 1995], pages 92–110, 1995. Revised version of a paper presented at the second EUROGRAPHICS workshop on Object-Oriented Graphics, Texel, The Netherlands, June 4–7, 1991.

[Cras, 1993] J.-Y. Cras, editor. *A Review of Industrial Constraint Solving Tools.* AI Perspectives Report, AI Intelligence, 1993.

[Cruz *et al.*, 1995a] Isabel Cruz, Kim Marriott, and Pascal Van Hentenryck, editors. *Proceedings of the International Workshop on Constraints for Graphics and Visualization* (*CGV'95*), Cassis, France, 18th September 1995. Department of Computer Science, Monash University, Clayton, Australia.

[Cruz *et al.*, 1995b] I. F. Cruz, J. Marks, and K. Wittenburg, editors. *Proceedings of the Workshop on Effective Abstractions in Multimedia Layout, Presentations, and Interaction in conjunction with ACM Multimedia '95.* San Francisco, CA, U.S.A., November 4, 1995. Also available as Electronic Proceedings via http://www.cs.tufts.edu/~isabel/mmwsproc.html.

[Cruz, 1995] I. Cruz. Expressing Constraints for Data Display Specification: A Visual Approach. In [Saraswat and Van Hentenryck, 1995], pages 445–469, 1995.

[Darses, 1990] Françoise Darses. Constraints in design: towards a methodology of psychological analysis based on AI formalisms. In D. Diaper, D. Gilmore, G. Cockton, and B. Shackel, editors, *Human-Computer Interaction – INTERACT '90*, pages 135–139, Cambridge, U.K., 27–31 August 1990. Elsevier Science Publishers B.V. (North-Holland), Amsterdam, The Netherlands. Proceedings of the IFIP TC 13 Third International Conference on Human-Computer Interaction.

[Dechter and van Beek, 1995] Rina Dechter and Peter van Beek. Local and Global Relational Consistency—Summary of Recent Results. In [Montanari and Rossi, 1995], pages 240–257, 1995.

[Dengler *et al.*, 1993] E. Dengler, M. Friedell, and J. Marks. Constraint-driven diagram layout. In *Proceedings of the 1993 IEEE Symposium on Visual Languages*, pages 330–335, Bergen, Norway, 1993.

[Di Battista *et al.*, 1994] G. Di Battista, P. Eades, R. Tamassia, and I. G. Tollis. Algorithms for drawing graphs: an annotated bibliography. *Computational Geometry: Theory and Applications*, 4:235–282, 1994. Also available via http://www.cs.brown.edu:80/people/rt/gd-biblio.html.

[du Verdier, 1993] Franck du Verdier. Solving Geometric Constraint Satisfaction Problems for Spatial Planning. In [Bajcsy, 1993], volume 2, pages 1564–1569, 1993.

[Du, 1995] Chun Du. *RelCAD: A Relational CAD System with Constraint-Based Geometric Modelling.* Verlag Dr. Kovač, Hamburg, March 1995.

[Duby *et al.*, 1989] C. Duby, S. Meyer, and S. P. Reiss. Using GELO to visualize software systems. In *Proceedings of the UIST'89 (ACM SIGGRAPH Symp. on User Interface Software and Technology)*, pages 149–157, Williamsburg, VA, 1989.

[Duisberg, 1986] Robert A. Duisberg. Animated Graphical Interfaces using Temporal Constraints. In *Human factors in computing systems*, pages 131–136. ACM, April 13–17, 1986. CHI'86 conference proceedings.

[Duisberg, 1987] R. Duisberg. Animation using temporal constraints: An overview of the animus system. In *Human-Computer Interaction*, pages 275–307. Lawrence Erlbaum Associates, Hillsdale, NJ, 1987.

[Duisberg, 1990] R. Duisberg. Visual Programming of Program Visualizations - A Gestural Interface for Animating Algorithms. In T. Ichikawa, E. Jungert, and R. Korfhage, editors, *Visual Languages and Applications*, pages 161–173. Plenum Press, New York, NY, 1990.

[Ege *et al.*, 1987] R. Ege, D. Maier, and A. Borning. The Filter browser – defining interfaces graphically. In *Proceedings of ECOOP-OOPSLA '87 (European Conference on Object-Oriented Programming)*, pages 155–165, Paris, June 1987.

[Ege, 1989] Raimund K. Ege. Direct Manipulation User Interfaces Based on Constraints. In *COMPSAC'89, International Computer Software and Applications Conference*, pages 374–380. IEEE, September 1989. Proceedings.

[El Dahshan and Barthes, 1989] Kamal El Dahshan and Jean Paul Barthes. Implementing Constraint Propagation in Mechanical CAD Systems. In Varol Akman, Paul J.W. ten Hagen, and Paul J. Veerkamp, editors, *Intelligent CAD Systems II, Implementational Issues*, pages 217–227. Record of the Second Eurographics Workshop on 'Intelligent CAD Systems – Implementational Issues', Koningshof Congress Center, Veldhoven, The Netherlands, April 19–23, 1988, Springer-Verlag, Berlin/Heidelberg, EurographicSeminars, Tutorials and Perspectives in Computer Graphics, EUROGRAPHICS, The European Association for Computer Graphics, 1989.

[Elliman *et al.*, 1993] D.G. Elliman, E.K. Burke, and M.I. Heard. A Constraint Based Approach to Engineering Design. *AISB Quarterly*, 83/84:35–38, Spring/Summer 1993. Newsletter of the Society for the Study of Artificial Intelligence and Simulation of Behaviour, Brighton, UK.

[Elliott *et al.*, 1994] C. Elliott, G. Schlechter, R. Yeung, and S. Abi-Ezzi. TBAG: A high level framework for interactive, animated 3d graphics applications. In *Proceedings of the ACM SIGGRAPH '94*, Orlando, FL, July 1994.

[Emmerik, 1990] Maarten J.G.M. van Emmerik. A System for Interactive Graphical Modeling with Three-Dimensional Constraints. In Tat-Seng Chua and Tosiyasu L. Kunii, editors, *CG International '90, Computer Graphics Around the World*, pages 361–376. Springer-Verlag, Tokyo/Berlin/Heidelberg, 1990.

[Epstein and LaLonde, 1988] D. Epstein and W. LaLonde. A Smalltalk window system based on constraints. In *Proceedings of OOPSLA '88 (ACM Conference on Object-Oriented Programming Systems, Languages, and Applications)*, pages 83–94, September 1988.

[Faltings and Sun, 1993] Boi Faltings and Kun Sun. Computer-aided Creative Mechanism Design. In [Bajcsy, 1993], volume 2, pages 1451–1457, 1993.

[Faltings *et al.*, 1992] Boi Faltings, Djamila Haroud, and Ian Smith. Dynamic Constraint Satisfaction with Continuous Variables. In David C. Brown, Manjula B. Waldron, and Hiroyuki Yoshikawa, editors, *Intelligent Computer Aided Design*, pages 427–445. Proceedings of the IFIP WG 5.2 Working Conference on Intelligent Computer Aided Design (IntCAD91, Columbus, OH, U.S.A., 30 September – 3 October 1991), Elsevier Science Publishers B.V. (North-Holland), 1992.

[Faltings, 1994a] Boi Faltings. Arc-consistency for continuous variables. *Artificial Intelligence*, 65(2):363–376, February 1994. Elsevier Science Publishers B.V., Amsterdam, The Netherlands.

[Faltings, 1994b] Boi Faltings. Constraint Satisfaction Problems in CAD Systems, August 14, 1994. Invited talk, International Workshop on *Constraint Processing in Computer-Aided Design (CoPiCAD-94)*, Swiss Federal Institute of Technology, Lausanne, Switzerland.

[Fertey *et al.*, 1990] G. Fertey, B. Peroche, and J. Zoller. Creating 3d scenes with constraints. Technical Report N° 90.4, Département Informatique Appliquée, Ecole Nationale Supérieure des Mines de Saint-Etienne, France, Février 1990.

[Flemming *et al.*, 1992] U. Flemming, C. A. Baykan, R. F. Coyne, and M. S. Fox. Hierarchical generate-and-test vs constraint-directed search—A comparison in the context of layout synthesis. In John S. Gero, editor, *Artificial intelligence in design '92* (Part 14 *Design Processes*), pages 817–838. Kluwer Academic Publishers, Dordrecht, The Netherlands, June 1992. Papers from the Second International Conference on Artificial Intelligence in Design (Pittsburgh, PA, U.S.A.).

[Foley *et al.*, 1990] J. Foley, A. van Dam, S. Feiner, and J. Hughes. *Computer Graphics: Principles and Practice.* The System Programming Series. Addison-Wesley, Reading, MA, 2nd edition, 1990.

[Freeman-Benson and Borning, 1992] Bjorn N. Freeman-Benson and Alan Borning. Integrating Constraints with an Object-Oriented Language. In Ole Lehrmann Madsen, editor, *ECOOP '92, European Conference on Object-Oriented Programming*, volume 615 of *Lecture Notes in Computer Science*, pages 268–286, Utrecht, The Netherlands, June 29 – July 3, 1992. Proceedings, Springer-Verlag, Berlin/Heidelberg.

[Freeman-Benson *et al.*, 1990] Bjorn N. Freeman-Benson, John Maloney, and Alan Borning. An Incremental Constraint Solver. *Communications of the ACM*, 33(1):54–63, January 1990.

[Freeman-Benson, 1991] Bjorn N. Freeman-Benson. Constraint Imperative Programming. Technical Report 91-07-02, Department of Computer Science and Engineering, University of Washington, Seattle, WA, U.S.A., August 14, 1991. Revised Ph.D. dissertation.

[Freeman-Benson, 1993] B. Freeman-Benson. Converting an existing user interface to use constraints. In *Proceedings of the UIST'93 (ACM SIGGRAPH Symp. on User Interface Software and Technology)*, pages 207–215, Atlanta, GA, 1993.

[Freuder *et al.*, 1995] Eugene C. Freuder, Rina Dechter, Matthew L. Ginsberg, Bart Selman, and Edward Tsang. Systematic Versus Stochastic Constraint Satisfaction. In Chris S. Mellish, editor, *IJCAI-95, Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, volume 2, pages 2027–2032, Montréal, Québec, Canada, August 20–25, 1995. IJCAII. Distributed by Morgan Kaufmann Publishers, Inc., San Mateo, California, USA. Panel.

[Fron, 1994] A. Fron. *Programmation par contraintes.* Addison-Wesley France, SA., 1994.

[Gero and Sudweeks, 1994] John S. Gero and Fay Sudweeks, editors. *Artificial Intelligence in Design '94.* Kluwer Academic Publishers, Dordrecht, The Netherlands, 1994.

[Gleicher and Witkin, 1994] M. Gleicher and A. Witkin. Drawing with constraints. *The Visual Computer*, 11(1), 1994.

[Gleicher, 1992] Michael Gleicher. Integrating Constraints and Direct Manipulation. *Computer Graphics*, pages 171–174, 1992. Special issue, Proceedings, 1992 Symposium on Interactive 3D Graphics, Cambridge, Massachusetts, 29 March – 1 April, A publication of ACM SIGGRAPH, acm press, New York, NY, U.S.A.

[Gleicher, 1993] M. Gleicher. A graphics toolkit based on differential constraints. In *Proceedings of the UIST'93 (ACM SIGGRAPH Symp. on User Interface Software and Technology)*, pages 109–120, Atlanta, GA, 1993.

[Gleicher, 1995] Michael Gleicher. Practical Issues in Graphical Constraints. In [Saraswat and Van Hentenryck, 1995], pages 407–426, 1995.

[Glinert, 1990a] E. P. Glinert, editor. *Visual Programmimg Environments: Applications and Issues.* IEEE Computer Science Press, Los Alamitos, CA, 1990.

[Glinert, 1990b] E. P. Glinert, editor. *Visual Programmimg Environments: Paradigms and Systems.* IEEE Computer Science Press, Los Alamitos, CA, 1990.

[Gobbetti and Balaguer, 1993] E. Gobbetti and J.-F. Balaguer. VB2: An architecture for interaction in synthetic worlds. In *Proceedings of the UIST'93 (ACM SIGGRAPH Symp. on User Interface Software and Technology)*, pages 167–178, Atlanta, GA, 1993.

[Gosling, 1983] James Gosling. *Algebraic Constraints.* PhD thesis, Department of Computer Science, Carnegie-Mellon University, May 1983.

[Graf and Neurohr, 1995] W. H. Graf and S. Neurohr. Constraint-based layout in visual program design. In *Proceedings of the 1995 IEEE Symposium on Visual Languages*, Darmstadt, Germany, 1995.

[Graf, 1991] W. H. Graf. Constraint-based processing of design knowledge. In *Working Notes of the AAAI-91 Workshop on Intelligent Multimedia Interfaces*, Anaheim, CA, U.S.A., July 1991.

[Graf, 1992] W. H. Graf. Constraint-based graphical layout of multimodal presentations. In [Catarci *et al.*, 1992], pages 365–385, 1995.

[Graf, 1993] Winfried H. Graf. LAYLAB - A Constraint-Based Layout Manager for Multimedia Presentations. In Gavriel Salvendy and Michael J. Smith, editors, *Human-Computer Interaction: Software and Hardware Interfaces*, volume 19B of *Advances in Human Factors / Ergonomics*, pages 446–451. Elsevier Science Publishers B.V., Amsterdam, The Netherlands, August 8–13, 1993. Proceedings of the Fifth International Conference on Human-Computer Interaction (HCI International '93, Orlando, Florida, USA), Volume 2.

[Graf, 1995a] W. H. Graf. The constraint-based layout framework LayLab and its applications. In [Cruz *et al.*, 1995b], 1995.

[Graf, 1995b] Winfried H. Graf. Intent-Based Layout in Interactive Multimedia Communication. In *Proceedings of the First International Workshop on Intelligence and Multimodality in Multimedia Interfaces (IMMI-1)*, Edinburgh, Scotland, July 13–14, 1995.

[Gross *et al.*, 1988] Mark D. Gross, Stephen M. Ervin, James A. Anderson, and Aaron Fleisher. Constraints: Knowledge representation in design. *Design Studies*, 9(3):133–143, July 1988. Butterworth & Co (Publishers) Ltd.

[Gross, 1992] M. D. Gross. Graphical constraints in CoDraw. In *Proceedings of the 1992 IEEE Workshop on Visual Languages*, pages 81–87, Seattle, WA, 1992.

[Guan and Friedrich, 1992] Qi Guan and Gerhard Friedrich. Extending Constraint Satisfaction Problem Solving in Structural Design. In [Belli and Radermacher, 1992], pages 341–350, 1992.

[Guesgen and Hertzberg, 1993] Hans Werner Guesgen and Joachim Hertzberg. A Constraint-Based Approach to Spatiotemporal Reasoning. *Journal of Applied Intelligence*, 3:71–90, 1993. Kluwer Academic Publishers, Boston, MA, U.S.A.

[Haroud and Faltings, 1994] Djamila Haroud and Boi Faltings. Global Consistency for Continuous Constraints. In [Borning, 1994], pages 40–50, 1994.

[Hel-Or *et al.*, 1994] Yaacov Hel-Or, Ari Rappoport, and Michael Werman. Relaxed parametric design with probabilistic constraints. *Computer-Aided Design*, 26(6):426–434, June 1994. Special Issue *Solid Modeling '93*, Butterworth-Heinemann Ltd.

[Helm *et al.*, 1992] Richard Helm, Tien Huynh, Catherine Lassez, and Kim Marriott. A Linear Constraint Technology for Interactive Graphic Systems. In *Graphics Interface '92*, pages 301–309, Canada, 1992. Proceedings, Morgan Kaufmann Publishers, USA.

[Helm *et al.*, 1995] Richard Helm, Tien Huynh, Kim Marriott, and John Vlissides. An Object-Oriented Architecture for Constraint-Based Graphical Editing. In [Laffra *et al.*, 1995], pages 217–238, 1995. Revised version of a paper presented at the third EUROGRAPHICS workshop on Object-Oriented Graphics, Champéry, Switzerland, October 28–30, 1992.

[Henz *et al.*, 1995] M. Henz, G. Smolka, and J. Würtz. Object-Oriented Concurrent Constraint Programming in Oz. In [Saraswat and Van Hentenryck, 1995], pages 29–48, 1995.

[Heydon and Nelson, 1994] Allan Heydon and Greg Nelson. The Juno-2 Constraint-Based Drawing Editor. SRC Research Report 131a, Systems Research Center, Digital Equipment Corporation, Palo Alto, California, USA, December 1994.

[Hill *et al.*, 1993] R. Hill, T. Brinck, J. Patterson, S. Rohall, and W. Wilner. The rendezvous language and architecture. *Communications of the ACM*, 36(1):62–67, January 1993.

[Hill, 1991] Ralph D. Hill. A 2-D Graphics System for Multi-User Interactive Graphics Based on Objects and Constraints. In E. H. Blake and P. Wißkirchen, editors, *Advances in Object-Oriented Graphics I*, pages 67–91. Springer-Verlag, Berlin/Heidelberg, 1991.

[Hill, 1993] R. Hill. The RENDEZVOUS constraint maintenance system. In *Proceedings of the UIST'93 (ACM SIGGRAPH Symp. on User Interface Software and Technology)*, pages 225–234, Atlanta, GA, 1993.

[Horn, 1992] B. Horn. Properties of User Interface Systems and the Siri Programming Language. In B. Myers, editor, *Languages for Developing User Interfaces*, pages 211–236. Jones and Barlett, Boston, MA, 1992.

[Hower and Jacobi, 1994] Walter Hower and Stephan Jacobi. A distributed realization for constraint satisfaction. In H. Kitano et al., editors, *Parallel Processing for Artificial Intelligence 2*, volume 15 of *Machine Intelligence and Pattern Recognition series*, pages 107–116. Elsevier Science Publishers B.V., Amsterdam, The Netherlands, 1994.

[Hower *et al.*, 1993] Walter Hower, Manfred Rosendahl, and Roland Berling. Constraint processing in human-computer interaction with an emphasis on intelligent CAD. In Michael J. Smith and Gavriel Salvendy, editors, *Human-Computer Interaction: Applications and Case Studies*, volume 19A of *Advances in Human Factors / Ergonomics*, pages 243–248. Elsevier Science Publishers B.V., Amsterdam, The Netherlands, August 8–13, 1993. Proceedings of the Fifth International Conference on Human-Computer Interaction (HCI International '93, Orlando, Florida, USA), Volume 1.

[Hower *et al.*, 1994a] Walter Hower, Djamila Haroud, and Zsófia Ruttkay, editors. *Constraint Processing in Computer-Aided Design* (*CoPiCAD-94*), August 14, 1994. Workshop notes, Third International Conference on Artificial Intelligence in Design (AID'94), Swiss Federal Institute of Technology, Lausanne, Switzerland, 15–18 August 1994.

[Hower *et al.*, 1994b] Walter Hower, Djamila Haroud, and Zsófia Ruttkay (editors). Proceedings of the AID'94 workshop W9 on *Constraint Processing in Computer-Aided Design* (*CoPiCAD-94*). Fachberichte Informatik 18/94, Institut für Informatik, Universität Koblenz-Landau, Koblenz, 1994.

[Hower *et al.*, 1995] Walter Hower, Derrick Köstner, and Manfred Rosendahl. Computer-aided layout by evolutionary computing. In [Veltkamp and Blake, 1995], pages 251–269, 1995.

[Hower, 1995] Walter Hower. Constraint satisfaction — Algorithms and complexity analysis. *Information Processing Letters*, 55(3):171–178, 11 August 1995. Elsevier Science Publishers B.V.

[Hower, 1996] Walter Hower. Bottom-up layout generation. *Informatica*, 20(1), 1996. An International Journal of Computing and Informatics.

[Hudson and Mohamed, 1990] S. E. Hudson and S. P. Mohamed. Interactive specification of flexible user interface displays. *ACM Transactions on Information Systems*, 8(3):269–288, 1990.

[Iba and Inoue, 1991] Hitoshi Iba and Hirochika Inoue. Reasoning of Geometric Concepts based on Algebraic Constraint-directed Method. In [Mylopoulos and Reiter, 1991], pages 143–149, 1991.

[Ingalls *et al.*, 1988] D. Ingalls, S. Wallace, Y. Chow, F. Ludolph, and K. Doyle. Fabrik: A visual programming environment. In *Proceedings of OOPSLA '88 (ACM Conference on Object-Oriented Programming Systems, Languages, and Applications)*, pages 176–190, 1988.

[Isaacs and Cohen, 1987] Paul M. Isaacs and Michael F. Cohen. Controlling dynamic simulation with kinematic constraints, behavior functions and inverse dynamics. *Computer Graphics*, 21(4):215–224, 1987. SIGGRAPH '87, Anaheim, California, July 27–31, Proceedings, A publication of ACM SIGGRAPH, acm press, New York, NY, U.S.A.

[Jaffar and Maher, 1994] Joxan Jaffar and Michael J. Maher. Constraint logic programming: a survey. *The Journal of Logic Programming*, 19/20:503–581, May/July 1994. Special Issue: Ten Years of Logic Programming, North-Holland, Elsevier Science Publishing Co., Inc., New York, NY, U.S.A.

[Johnson, 1963] Timothy E. Johnson. Sketchpad III, a computer program for drawing in three dimensions. In *Spring Joint Computer Conference*, page 347 ff., Detroit, Michigan, U.S.A., May 21–23, 1963. Proceedings.

[Kalra and Barr, 1990] Devendra Kalra and Alan H. Barr. A Constraint-Based Figure-Maker. In C.E. Vandoni and D.A. Duce, editors, *Eurographics '90, Proceedings of the European Computer Graphics Conference and Exhibition*, pages 413–424, Montreux, Switzerland, 4–7 Septemver 1990. Elsevier Science Publishers B.V. (North-Holland), Amsterdam, The Netherlands.

[Kamada and Kawai, 1991] T. Kamada and S. Kawai. A general framework for visualizing abstract objects and relations. *ACM Transactions on Graphics*, 10(1):1–39, January 1991.

[Kamada, 1989] Tomihisa Kamada. *Visualizing Abstract Objects and Relations — A Constraint-Based Approach*, volume 5 of *Series in Computer Science*. World Scientific Publishing Co Pte Ltd., Singapore, 1989.

[Karsenty *et al.*, 1992] Solange Karsenty, James A. Landay, and Chris Weikart. Inferring Graphical Constraints with Rockit. Technical Report 17, DEC, Paris Research Laboratory, France, March 1992. HCI'92, University of York, U.K., 15–18 September.

[Kass, 1992] Michael Kass. CONDOR: Constraint-Based Dataflow. *Computer Graphics*, 26(2):321–330, July 1992. SIGGRAPH '92, 19th annual ACM Conference on Computer Graphics and Interactive Techniques, Chicago, Illinois, July 26–31, 1992, Conference Proceedings, A publication of ACM SIGGRAPH, acm press, New York, NY, U.S.A.

[Keirouz *et al.*, 1990] Walid Keirouz, Jahir Pabon, and Robert Young. Integrating parametric geometry, features, and variational modeling for conceptual design. In J. R. Rinderle, editor, *International Conference on Design Theory and Methodology*, pages 1–9. American Society of Mechanical Engineers (ASME), 1990. Proceedings.

[Klein, 1994] Rüdiger Klein. How to Use Constraints – Exemplified in Configuration Problem Solving. In [Hower *et al.*, 1994a], pages 14–19, 1994.

[Knuth, 1986] D. E. Knuth. *The METAFONT book.* Computer and Typesetting Series. Reading, MA, Reading, MA, 1986.

[Kondo, 1992] Koichi Kondo. Algebraic method for manipulation of dimensional relationships in geometric models. *CAD, computer-aided design*, 24(3):141–147, march 1992. Butterworth-Heinemann Ltd.

[Kramer, 1992] Glenn A. Kramer. *Solving Geometric Constraint Systems: A Case Study in Kinematics.* Artificial Intelligence series. The MIT Press, Cambridge, Massachusetts, USA / London, England, UK, 1992. Reviewed in [Berling and Hower, 1993].

[Kramer, 1994] Glenn A. Kramer. A geometric constraint engine. In Eugene C. Freuder and Alan K. Mackworth, editors, *Constraint-Based Reasoning*, pages 327–360. Special Issues of *Artificial Intelligence*, A Bradford Book, The MIT Press, Cambridge, Massachusetts, USA / London, England, UK, 1994.

[Kumar, 1992] V. Kumar. Algorithms for constraint-satisfaction problems: A survey. *AI Magazine*, 13(1):32–44, 1992.

[Kurlander and Feiner, 1992] D. Kurlander and S. Feiner. Interactive constraint-based search and replace. In *Proceedings of the CHI'92 (Human Factors in Computing Systems)*, pages 609–618, Monterey, CA, May 1992.

[Kurlander and Feiner, 1993] D. Kurlander and S. Feiner. Inferring constraints from multiple snapshots. *ACM Transactions on Graphics*, 12(4):277–304, October 1993.

[Kurlander *et al.*, 1994] David Kurlander, Jean-François Puget, and Jeff Heisserman. Panel Session: Commercial Applications of Constraint Programming. In [Borning, 1994], pages 350–360, 1994. Panel organizer: Alan Borning.

[Laffra *et al.*, 1995] Chris Laffra, Edwin H. Blake, Vicki de Mey, and Xavier Pintado, editors. *Object-Oriented Programming for Graphics*, Focus on Computer Graphics, Tutorials and Perspectives in Computer Graphics. Springer-Verlag, 1995. Revised versions of papers presented at the second and third EUROGRAPHICS workshop on Object-Oriented Graphics.

[Leler, 1988] W. Leler. *Constraint Programming Languages: Their Specification and Generation.* Addison-Wesley, Reading, MA, 1988.

[Lewis, 1990] C. Lewis. NOpumpG: Creating interactive graphics with spreadsheet machinery. In Glinert [1990b], pages 526–546.

[Liu and Popplestone, 1990] Yanxi Liu and Robin J. Popplestone. Symmetry Constraint Inference in Assembly Planning — *Automatic Assembly Configuration Specification.* In *AAAI-90, Eighth National Conference on Artificial Intelligence*, volume Two, pages 1038–1044, Boston, MA, July 29 – August 3, 1990. American Association for Artificial Intelligence, Proceedings, AAAI Press, Menlo Park, CA, U.S.A.

[Lopez *et al.*, 1994] Gus Lopez, Bjorn Freeman-Benson, and Alan Borning. Kaleidoscope: A Constraint Imperative Programming Language. In Brian Mayoh, Enn Tyugu, and Jaan Penjam, editors, *Constraint Programming*, volume 131, Series F: Computer and Systems Sciences of *NATO ASI Series*, pages 313–329. Springer-Verlag, Berlin/Heidelberg, 1994. Proceedings of the NATO Advanced Study Institute on Constraint Programming, held in Pärnu, Estonia, August 13–24, 1993.

[Lozano-Pérez, 1983] Tomás Lozano-Pérez. Spatial Planning: A Configuration Space Approach. *IEEE Transactions on Computers*, C-32(2):108–120, February 1983.

[Mackworth, 1992] Alan K. Mackworth. Constraint Satisfaction. In Stuart C. Shapiro, editor, *Encyclopedia of Artificial Intelligence*, volume 1, pages 285–293. John Wiley & Sons, 1992. Second edition.

[Maleki, 1987] Jalal Maleki. VIVID, The Kernel of a Knowledge Representation Environment Based on the Constraints Paradigm of Computation. In *Proceedings of the Twentieth Annual Hawaii International Conference on System Sciences*, pages 591–600. North-Holland, January 1987.

[Maloney *et al.*, 1989] John H. Maloney, Alan Borning, and Bjorn N. Freeman-Benson. Constraint Technology for User-Interface Construction in ThingLab II. In *Conference on Object-Oriented Programming Systems, Languages, and Applications*, pages 381–388, New Orleans, October 1–6, 1989. Proceedings, ACM. Technical Report 89-05-02, Department of Computer Science and Engineering, University of Washington, Seattle, WA, U.S.A.

[Maloney, 1991] John Harold Maloney. Using Constraints for User Interface Construction. Technical Report 91-08-12, Department of Computer Science and Engineering, University of Washington, Seattle, WA, U.S.A., 1991. PhD thesis.

[Matoušek, 1994] Jiří Matoušek. On geometric optimization with few violated constraints. In *10th Annual Symposium on Computational Geometry*, Stony Brook, New York, NY, U.S.A., June 6–8, 1994.

[Maulsby *et al.*, 1988] D.L. Maulsby, K.A. Kittlitz, and I.H. Witten. Constraint-Solving in Interactive Graphics: A User-Friendly Approach. In N. Magnenat-Thalmann and D. Thalmann, editors, *New Trends in Computer Graphics*, pages 305–318. Proceedings of CG International '88, Springer-Verlag, 1988.

[Maybury, 1993] M. Maybury, editor. *Intelligent Multimedia Interfaces.* AAAI Press, Menlo Park, CA, 1993.

[McWhirter and Nutt, 1994] J. D. McWhirter and G. J. Nutt. Escalante: An environment for the rapid construction of visual language applications. In *Proceedings of the 1994 IEEE Symposium on Visual Languages*, St. Louis, MO, 1994.

[Miyashita *et al.*, 1992] K. Miyashita, S. Matsuoka, S. Takahashi, A. Yonezawa, and T. Kamada. Declarative programming of graphical interfaces by visual examples. In *Proceedings of the UIST'92 (ACM SIGGRAPH Symp. on User Interface Software and Technology)*, pages 107–116, Monterey, CA, 1992.

[Montanari and Rossi, 1995] Ugo Montanari and Francesca Rossi, editors. *Principles and Practice of Constraint Programming — CP '95*, volume 976 of *Lecture Notes in Computer Science.* Proceedings, Springer-Verlag, Berlin/Heidelberg, September 19–22, 1995. First International Conference, Cassis, France.

[Müller *et al.*, 1995] Tobias Müller, Konstantin Popov, Christian Schulte, and Jörg Würtz. Constraint Programming in Oz. DFKI Oz Documentation Series, Programming Systems Lab, German Research Center for Artificial Intelligence, Saarbrücken, June 28, 1995. http://ps-www.dfki.uni-sb.de/oz/documentation/#constraints.

[Murtagh and Shimura, 1990] Niall Murtagh and Masamichi Shimura. Parametric Engineering Design Using Constraint-Based Reasoning. In *AAAI-90, Eighth National Conference on Artificial*

Intelligence, volume One, pages 505–510, Boston, MA, July 29 – August 3, 1990. American Association for Artificial Intelligence, Proceedings, AAAI Press, Menlo Park, CA, U.S.A.

[Myers *et al.*, 1989] B. A. Myers, B. T. Vander Zanden, and R. B. Dannenberg. Creating graphical interactive application objects by demonstration. In *Proceedings of the UIST'89 (ACM SIGGRAPH Symp. on User Interface Software and Technology)*, pages 95–104, Williamsburg, VA, November 1989.

[Myers *et al.*, 1990] B. A. Myers, D. A. Guise, R. B. Dannenberg, B. T. Vander Zanden, D. S. Kosbie, E. Pervin, A. Mickish, and P. Marchal. Comprehensive support for graphical, highly-interactive user interface: The Garnet user interface development environment. *IEEE Computer*, 23(11):71–85, November 1990.

[Myers, 1988] B. A. Myers. *Creating User Interfaces by Demonstration.* Academic Press, New York, NY, 1988.

[Mylopoulos and Reiter, 1991] John Mylopoulos and Ray Reiter, editors. *12th International Joint Conference on Artificial Intelligence*, Darling Harbour, Sydney, Australia, 24 – 30 August 1991. IJCAII, Proceedings, Volume 1; distributed by Morgan Kaufmann Publishers, San Mateo, California, USA.

[Nagai and Terasaki, 1993] Yasuo Nagai and Satoshi Terasaki. A Constraint-Based Knowledge Compiler for Parametric Design Problem in Mechanical Engineering. ICOT Technical Memorandum TM-1270, Institute for New Generation Computer Technology, Tokyo, Japan, 1993.

[Navinchandra and Rinderle, 1989] D. Navinchandra and J. R. Rinderle. Interval approaches for concurrent evaluation of design constraints. In *Symposium on Concurrent Product and Process Design*, pages 101–108, San Francisco, CA, U.S.A., 1989. Proceedings.

[Nelson, 1985] Greg Nelson. Juno, a constraint-based graphics system. *Computer Graphics*, 19(3):235–243, July 1985. SIGGRAPH'85, San Francisco, CA, July 22–26, Proceedings, acm press, New York, NY, U.S.A.

[Newbery Paulisch, 1993] Frances Newbery Paulisch. *The Design of an Extendible Graph Editor*, volume 704 of *Lecture Notes in Computer Science.* Springer-Verlag, Berlin/Heidelberg, 1993.

[Nourani and Magalhães, 1993] Farid Nourani and Léo Pini Magalhães. Management of consistency constraints in a CAD database system. In Zesheng Tang, editor, *New Advances in Computer Aided Design & Computer Graphics*, volume 2, pages 770–776, Beijing, China, August 23–26, 1993. Proceedings of the Third International Conference on CAD and Computer Graphics (CAD/Graphics'93), International Academic Publishers.

[Olsen Jr. and Allan, 1990] Dan R. Olsen Jr. and Kirk Allan. Creating Interactive Techniques by Symbolically Solving Geometric Constraints. In *UIST 90, 3rd Annual Symposium on User Interface Software and Technology*, pages 102–107, Snowbird, Utah, USA, October 1990. ACM SIGGRAPH/SIGCHI.

[Owen, 1991] J.C. Owen. Algebraic Solution for Geometry from Dimensional Constraints. In *Symposium on Solid Modeling Foundations and CAD/CAM Applications*, pages 397–407, Austin, Texas, USA, 1991. ACM. Proceedings.

[Pabon *et al.*, 1992] Jahir Pabon, Robert Young, and Walid Keirouz. Integrating Parametric Geometry, Features, and Variational Modeling for Conceptual Design. *International Journal of Systems Automation: Research and Applications (SARA)*, 2:17–36, 1992.

[Paltrinieri, 1995] Massimo Paltrinieri. A Visual Environment for Constraint Programming. In *11th International IEEE Symposium on Visual Languages*, pages 118–119. IEEE Computer Society Press, 1995. Poster Session.

[Pavlidis and Wyk, 1985] T. Pavlidis and C. Van Wyk. An automatic beautifier for drawings and illustrations. *Computer Graphics*, 19(3):225–234, 1985.

[Pfefferkorn, 1975] Charles E. Pfefferkorn. A Heuristic Problem Solving Design System for Equipment or Furniture Layouts. *Communications of the ACM*, 18(5):286–297, May 1975.

[Phillips *et al.*, 1990] Cary B. Phillips, Jianmin Zhao, and Norman I. Badler. Interactive Real-time Articulated Figure Manipulation Using Multiple Kinematic Constraints. *Computer Graphics*, 24(2):245–250, March 1990. Special issue on 1990 Symposium on Interactive 3D Graphics, Snowbird, Utah, 25th – 28th March, Proceedings, A publication of ACM SIGGRAPH, acm press, New York, NY, U.S.A.

[Pineda, 1992] Luis A. Pineda. Reference, Synthesis and Constraint Satisfaction. In A. Kilgour and L. Kjelldahl, editors, *Eurographics '92*, volume 11, number 3, pages C–333 – C–344. Eurographics Association, Blackwell Publishers, 1992.

[Press *et al.*, 1994] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C.* Cambridge University Press, Cambridge, England, 2nd edition, 1994.

[Rankin, 1995] John R. Rankin. A Graphics Object-Oriented Constraint Solver. In [Laffra *et al.*, 1995], pages 71–91, 1995. Revised version of a paper presented at the second EUROGRAPHICS workshop on Object-Oriented Graphics, Texel, The Netherlands, June 4–7, 1991.

[Rappoport, 1993] Ari Rappoport. Direct Manipulation Devices for the Design of Geometric Constraint Networks. In N. Magnenat-Thalmann and D. Thalmann, editors, *Computer Graphics International '93 — Communicating with virtual worlds*, pages 294–305, Swiss Federal Institute of Technology, Lausanne, Switzerland, June 21–25, 1993. Springer.

[Reiss, 1993] S. P. Reiss. A framework for abstract 3d visualization. In *Proceedings of the 1993 IEEE Symposium on Visual Languages*, pages 108–115, Bergen, Norway, 1993.

[Rosendahl, 1993] Manfred Rosendahl. UNDO in CAD Systems. In W. Cellary, K. Vidyasankar, and G. Vossen, editors, *Versioning in Database Management Systems*, page 12, Dagstuhl-Seminar-Report 55, IBFI GmbH, Internationales Begegnungs- und Forschungszentrum für Informatik, Schloß Dagstuhl, Wadern, February 1–5, 1993.

[Rowe *et al.*, 1991] L. A. Rowe, J. A. Konstan, B. C. Smith, S. Seitz, and C. Liu. The PICASSO application framework. In *Proceedings of the UIST'91 (ACM SIGGRAPH Symp. on User Interface Software and Technology)*, pages 95–106, Hilton Head, SC, 1991.

[Ruttkay, 1995] Zsófia Ruttkay. A Co-operative Graphical Editor Based on Dynamically Constrained Objects. In [Laffra *et al.*, 1995], pages 33–53, 1995. Revised version of a paper presented at the second EUROGRAPHICS workshop on Object-Oriented Graphics, Texel, The Netherlands, June 4–7, 1991.

[Sacks, 1993] Elisha P. Sacks. What's in a linkage. *Artificial Intelligence*, 61(2):343–349, June 1993. Elsevier Science Publishers B.V., Amsterdam, The Netherlands. Book Review of: Glenn Kramer, *Solving Geometric Constraint Systems.*

[Sannella *et al.*, 1993] Michael Sannella, John Maloney, Bjorn Freeman-Benson, and Alan Borning. Multi-way versus One-way Constraints in User Interfaces: Experience with the DeltaBlue Algorithm. *Software—Practice and Experience*, 23(5):529–566, May 1993. A Wiley-Interscience Publication.

[Sannella, 1994] M. Sannella. *Constraint Satisfaction and Debugging for Interactive User Interfaces.* PhD thesis, Department of Computer Science and Engineering, University of Washington, Seattle, WA, 1994.

[Sannella, 1995] Michael Sannella. The SkyBlue Constraint Solver and Its Applications. In [Saraswat and Van Hentenryck, 1995], pages 385–406, 1995.

[Saraswat and Van Hentenryck, 1995] Vijay Saraswat and Pascal Van Hentenryck, editors. *Principles and practice of constraint programming: the Newport papers.* The MIT Press, Massachusetts Institute of Technology, Cambridge, MA, U.S.A., 1995.

[Serrano and Gossard, 1987] D. Serrano and D. Gossard. Constraint Management in Conceptual Design. In D. Sriram and R. A. Adey, editors, *Knowledge Based Expert Systems in Engineering: Planning and Design*, pages 211–224. Computational Mechanics Publications, 1987.

[Shimada *et al.*, 1989] Kenji Shimada, Masayuki Numao, Hiroshi Masuda, and Shinji Kawabe. Constraint-based object description for product modeling. In F. Kimura and A. Rolstådas, editors, *Computer Applications in Production and Engineering*, pages 95–106. Elsevier Science Publishers B.V. (North-Holland), 1989. Proceedings of the Third International IFIP Conference on Computer Applications in Production and Engineering, CAPE '89, Tokyo, Japan, 2–5 October 1989.

[Singh and Green, 1989] G. Singh and M. Green. Chisel: A system for creating highly interactive screen layouts. In *Proceedings of the UIST'89 (ACM SIGGRAPH Symp. on User Interface Software and Technology)*, pages 86–94, Williamsburg, VA, 1989.

[Sistare, 1991] Steve Sistare. Graphical Interaction Techniques in Constraint-Based Geometric Modeling. In *Graphics Interface '91*, pages 85–92, Toronto, Canada, June 1991. Proceedings.

[Smithers, 1989] T Smithers. AI-based design versus geometry-based design or Why design cannot be supported by geometry alone. *CAD, computer-aided design*, 21(3):141–150, april 1989. Butterworth & Co (Publishers) Ltd.

[Solano and Brunet, 1994] Lluis Solano and Pere Brunet. Constructive constraint-based model for parametric CAD systems. *Computer-Aided Design*, 26(8):614–621, August 1994. Butterworth-Heinemann Ltd.

[Sriram and Maher, 1986] D. Sriram and M.L. Maher. The Representation and Use of Constraints in Structural Design. In *Applications of Artificial Intelligence in Engineering, 1st International Conference*, pages 355–368, Southampton, U.K., April 1986. Proceedings, Springer-Verlag.

[Stasko, 1990] J. T. Stasko. Tango: A framework and system for algorithm animation. *IEEE Computer*, 23(9):27–39, September 1990.

[Steinberg, 1987] Louis I. Steinberg. Design as Refinement Plus Constraint Propagation: The VEXED Experience. In *AAAI 87, Sixth National Conference on Artificial Intelligence*, volume 2, pages 830–835, Seattle, WA, U.S.A., July 13–17, 1987. Proceedings, American Association for Artificial Intelligence.

[Sunde, 1988] Geir Sunde. Specification of shape by dimensions and other geometric constraints. In Michael J. Wozny, H. W. McLaughlin, and José Luis Encarnação, editors, *Geometric Modeling for CAD Applications*, pages 199–213. Elsevier Science Publishers B.V. (North-Holland), Amsterdam, The Netherlands, 1988. Selected and expanded papers from the IFIP WG 5.2 Working Conference, Rensselaerville, NY, U.S.A., 12–16 May 1986.

[Sutherland, 1963] Ivan E. Sutherland. Sketchpad, a man-machine graphical communication system. In *Spring Joint Computer Conference*, pages 329–346, Detroit, Michigan, U.S.A., May 21–23, 1963. Proceedings.

[Szwillus, 1993] Gerd Szwillus. Solving Linear Graphical Constraint Expressions. In *Poster Sessions: Abridged Proceedings HCI International '93*, page 267, 8–13 August 1993. 5th International Conference on Human-Computer Interaction jointly with 9th Symposium on Human Interface (Japan), Hilton at Walt Disney World Village, Orlando, Florida, U.S.A.

[Szwillus, 1994] Gerd Szwillus. Graphical Constraints. CHI'94 Tutorial, Conference on Human Factors in Computing Systems, April 25, 1994. Boston, MA, U.S.A.

[Takahashi *et al.*, 1991] S. Takahashi, S. Matsuoka, A. Yonezawa, and T Kamada. A general framework for bi-directional translation between abstract and pictorial data. In *Proceedings of the UIST'91 (ACM SIGGRAPH Symp. on User Interface Software and Technology)*, pages 165–174, Hilton Head, SC, 1991.

[Takahashi *et al.*, 1995] Shin Takahashi, Satoshi Matsuoka, Ken Miyashita, Hiroshi Hosobe, Akinori Yonezawa, and Tomihisa Kamada. A Constraint-Based Approach for Visualization and Animation. In [Cruz *et al.*, 1995a], pages 103–117, 1995.

[Tamassia, 1995] Roberto Tamassia. Constraints in Graph Drawing. In [Cruz *et al.*, 1995a], page 85, 1995. Invited talk.

[Tanimoto, 1993] Toshikazu Tanimoto. A Constraint Decomposition Method for Spatio-Temporal Configuration Problems. In *Eleventh National Conference on Artificial Intelligence*, pages 145–151, Washington, D.C., July 11–15, 1993. American Association for Artificial Intelligence, Proceedings AAAI-93, AAAI Press, Menlo Park, CA, U.S.A.

[ten Hagen and Veerkamp, 1991] Paul J.W. ten Hagen and Paul J. Veerkamp, editors. *Intelligent CAD Systems III, Practical Experience and Evaluation*, EurographicSeminars, Tutorials and Perspectives in Computer Graphics. Springer-Verlag, Berlin/Heidelberg, 1991. Third Eurographics Workshop on 'Intelligent CAD Systems', Texel, The Netherlands.

[Thalmann and Thalmann, 1992] N. Magnenat Thalmann and D. Thalmann. *Creating and Animating the Virtual World.* Computer Animation Series. Springer-Verlag, Berlin, Germany, 1992.

[Thennarangam and Singh, 1994] Suresh Thennarangam and Gurminder Singh. Inferring 3-dimensional constraints with DEVI. In [Borning, 1994], pages 78–85, 1994.

[Thornton, 1994] A. C. Thornton. Genetic algorithms versus simulated annealing: satisfaction of large sets of algebraic mechanical design constraints. In [Gero and Sudweeks, 1994], pages 381–398, 1994.

[Tokuyama *et al.*, 1991] Takeshi Tokuyama, Takao Asano, and Shuji Tsukiyama. A Dynamic Algorithm for Placing Rectangles without Overlapping. *Journal of Information Processing*, 14(1):30–35, 1991.

[Tonouchi *et al.*, 1992] T. Tonouchi, K. Nakayama, S. Matsuoka, and S. Kawai. Creating visual objects by direct manipulation. In *Proceedings of the 1992 IEEE Workshop on Visual Languages*, pages 95–101, Seattle, WA, 1992.

[Towhidnejad *et al.*, 1993] M. Towhidnejad, H. R. Myler, and A. J. Gonzalez. Constraint Mechanisms in Automated Knowledge Generation. *Applied Artificial Intelligence*, 7(2):113–134, April–June 1993. Taylor & Francis.

[Tsang, 1990] Jean Patrick Tsang. Constraint propagation issues in automated design. In Georg Gottlob and Wolfgang Nejdl, editors, *Expert Systems in Engineering — Principles and Applications*, pages 135–151, Vienna, Austria, September 24–26, 1990. Proceedings, Lecture Notes in Artificial Intelligence, Subseries of Lecture Notes in Computer Science, Volume 462, Springer-Verlag, Berlin/Heidelberg. International Workshop.

[Tsang, 1993] Edward Tsang. *Foundations of Constraint Satisfaction.* Computation in Cognitive Science. Academic Press, Harcourt Brace & Company, Publishers, London, UK, 1993.

[Tsuchida, 1995] Kensei Tsuchida. The Complexity of Drawing Tree-Structured Diagrams. *IEICE Transactions on Information and Systems*, E78-D(7):901–908, July 1995. The Institute of Electronics, Information and Communication Engineers, Tokyo, Japan.

[Van Hentenryck, 1989] P. Van Hentenryck. *Constraint Satisfaction in Logic Programming.* MIT Press, Cambridge, MA, 1989. Revision of Ph.D. thesis, University of Namur, 1987.

[Van Hentenryck, 1995] Pascal Van Hentenryck. Constraint Solving for Combinatorial Search Problems: A Tutorial. In [Montanari and Rossi, 1995], pages 564–587, 1995.

[van Wyk, 1982] C.J. van Wyk. A high-level language for specifying pictures. *ACM Transactions on Graphics*, 1(2):163–182, 1982.

[Vander Zanden *et al.*, 1991] B. T. Vander Zanden, B. A. Myers, D. A. Guise, and P. Szekely. The importance of pointer variables in constraint models. In *Proceedings of the UIST'91 (ACM SIGGRAPH Symp. on User Interface Software and Technology)*, pages 155–164, Hilton Head, SC, 1991.

[Vander Zanden, 1988] Bradley T. Vander Zanden. Incremental Constraint Satisfaction and its Application to Graphical Interfaces. Technical Report TR-88-941, Department of Computer Science, Cornell University, Ithaca, New York, USA, October 1988.

[Veltkamp and Blake, 1995] Remco C. Veltkamp and Edwin H. Blake, editors. *Proceedings of the fifth Eurographics workshop on Programming Paradigms in Graphics, EUROGRAPHICS '95*, Maastricht, September 2–3, 1995. CWI, Amsterdam, The Netherlands.

[Veltkamp, 1995] Remco C. Veltkamp. A Quantum Approach to Geometric Constraint Satisfaction. In [Laffra *et al.*, 1995], pages 54–70, 1995. Revised version of a paper presented at the second EUROGRAPHICS workshop on Object-Oriented Graphics, Texel, The Netherlands, June 4–7, 1991.

[Welch and Witkin, 1992] W. Welch and A. Witkin. Variational surface modeling. *Computer Graphics*, 26(2), July 1992.

[Wilk, 1991] M. R. Wilk. Equate: An object-oriented constraint solver. In A. Paepcke, editor, *Proceedings of OOPSLA '91 (ACM Conference on Object-Oriented Programming Systems, Languages, and Applications)*, pages 286–298. ACM Press, New York, NY, October 1991.

[Witkin *et al.*, 1987] Andrew Witkin, Kurt Fleischer, and Alan Barr. Energy Constraints On Parameterized Models. *Computer Graphics*, 21(4):225–232, 1987. SIGGRAPH '87, Anaheim, California, July 27–31, Proceedings, A publication of ACM SIGGRAPH, acm press, New York, NY, U.S.A.

[Yamaguchi and Kimura, 1990] Yasushi Yamaguchi and Fumihiko Kimura. A constraint modeling system for variational geometry. In Michael J. Wozny, J. U. Turner, and K. Preiss, editors, *Geometric modeling for product engineering*, pages 221–233. Elsevier Science Publishers B.V. (North-Holland), Amsterdam, The Netherlands, 1990. Selected and Expanded Papers from the

IFIP WG 5.2 / NSF Working Conference on Geometric Modeling, Rensselaerville, NY, U.S.A., 18–22 September 1988.

[Young *et al.*, 1991] R. E. Young, A. Greef, and P. O'Grady. SPARK: an artificial intelligence constraint network system for concurrent engineering. In J.S. Gero, editor, *Artificial Intelligence in Design '91*, pages 79–94. Butterworth Heinemann, Oxford, UK, 1991.

[Zahn and Hower, 1996] Martin Zahn and Walter Hower. Backtracking along with constraint processing and their time complexities. *Journal of Experimental & Theoretical Artificial Intelligence*, 8(1), 1996. Taylor & Francis, London, UK.

[Žalik *et al.*, 1992] Borut Žalik, Nikola Guid, and Aleksander Vesel. Representing Geometric Objects Using Constraint Description Graphs. In [Belli and Radermacher, 1992], pages 505–514, 1992.