Interactive Mapping between Knowledge Level and Symbol Level with Geometry

A KL-Model for Design Space Exploration

Teng-Wen CHANG[1], Rob WOODBURY[2] and Sambit DATTA[3]

[1] *Graduate Institute of Architecture, National Chiao Tung University, Taiwan*

[2]*Graduate Program in Computing Arts and Design Sciences, Simon Fraser University, Canada*

[3]*School of Architecture and Building, Deakin University, Australia*

Keywords: design space, geometric description, knowledge level, subsumption, unfolding

Abstract: Design space exploration is long-standing motivating ideas in computer-aided design. It realises this vision through a model of design states for making and moving amongst states and an organisation of states into a structure called a design space. Using a design space structuring mechanism based on a *subsumption* relation, this paper sketches a theory called Geometric Typed Feature Structures (GTFS) to preserve the formal properties of the design space movement algorithms for geometry. It also provides the theory for incorporation of user-guided exploration in the design space. Consequently, the clear division between knowledge level and symbol level, such that functional decomposition → formal symbol level and design → model symbol level, disappears. We can therefore use the same subsumption relation to structure the design space exploration interactively. Such interactive mapping between knowledge level and symbol level provides the fine-grained opportunities for user intervention in formal design space movement algorithms. In this paper, we summarize this approach with an example of GTFS subsumption process.

1 BACKGROUND

Design space exploration is one of the long-standing motivating ideas in computer-aided design (e.g. Gero 1994; Smithers 1998; Woodbury et al. 1998). Its claim is that computers can be useful to the designers by supporting a conceptual world of states that designers create and among which they move in an open-ended process of discovery. It realises this vision through a model of design states comprising three elements—goals and designs, operators for making and moving amongst states and an organisation of states into a structure called a design space. Amongst its mooted benefits are that designers are made more free to consider alternative problem descriptions and designs.

There are many characterisations of this model but few treat all of its three elements at significant depth. By far the most common pattern is a clear description of states and operators. The design space is taken to be the derivation relation of operators applied to states. Effective systems, for example, GENESIS (Heisserman 1991) can be implemented within this view, but such systems trade on the completion of single states, in other words, on following single trajectories through a design space. This has happened for good reasons. It turns out that design spaces are not adequately structured by derivation and going beyond derivation to a richer set of structuring relations is a very hard problem (Chien 1998), especially when geometry is involved (Chang 1999; Chang and Woodbury 2000).

In his forthcoming thesis, Burrow (Burrow 2002) develops a design space structuring mechanism based on a subsumption relation. In it, two design states (and recursively their subparts) are related if one subsumes the other, that is, if one contains strictly less information than the other. This scheme replaces rule-based derivation with a set of composable operators including π-resolution, unification, anti-unification, hysterical undo, indexing and path reuse (Woodbury, Datta and Burrow 2000; Woodbury et al. 1999). It turns out that subsumption imposes tight restrictions on the formal properties of a representation scheme. One necessary condition is that, if two states share a common specialisation, then that specialisation must be unique. Another is that properties of objects must be (partial) functions – set valued features cannot be represented. A third is that the obvious operator set over the representation is couched in terms of information preserving or destroying movements in a design space. At first glance such operators would seem not to apply to geometric objects as it is “common knowledge” that geometric operations in design involve both addition and removal of information from a representation.

A consequence of the apparently necessary restrictions on formal representational properties is an engineering decision to split a design representation into two parts: a part whose knowledge level submits to a symbol level representation suffering the system’s formal restrictions and a part in which the restrictions are deemed not to apply. Burrow calls the former a functional decomposition (Burrow 2002) and the latter a model. These terms originate in the SEED knowledge level (Flemming and Woodbury 1995), which posited distinct functional units and design units as devices supporting a conceptual separation of brief and design. Burrow’s functional decompositions correspond to the functional unit structure of the SEED knowledge level. Chang (Chang 1999) showed that this distinction, between formal properties for functional units and informal ones for design units, is not implied by Burrow’s representation scheme. Rather, there are classes of geometric objects that can be represented within the scheme and classes of geometric operators that make compliant moves in a design space. What is not clear in Chang’s thesis is to what extent these restricted operators are practical, that is, to what extent they are useful to design space exploration.

The current paper presumes that some of geometric operators will be useful in some circumstances, but that the so-called functional-decomposition/model remains a necessary part of a subsumption-based design space explorer.

The sheer Vastness (in Dennett's sense (Dennett 1996) of a design space[1]) implies the futility of automated search against any but vanishingly small and trivially simple problems. A high level of user-directedness is thus inevitable in a practical design space explorer.

2 REPRESENTING GEOMETRY—GEOMETRIC TYPED FEATURE STRUCTURES

2.1 A Knowledge Level

In the context of realizing design space explorer, a theory called knowledge level developed by Newell (Newell 1982) is used for mapping a set of specific behaviours and their mechanisms. Briefly speaking, knowledge level concepts are those unformalised conceptual objects that we deal with in our heads. Symbol level thus comprises the representation mapped from its knowledge level concepts. The mapping between knowledge level and symbol level is done by the rationalize/mechanize mapping from observed interaction between an agent and the environment. Our knowledge level concepts, adapted from SEED (Flemming and Woodbury 1995), comprise functional units (FU), design units (DU), design states, technologies and design spaces.

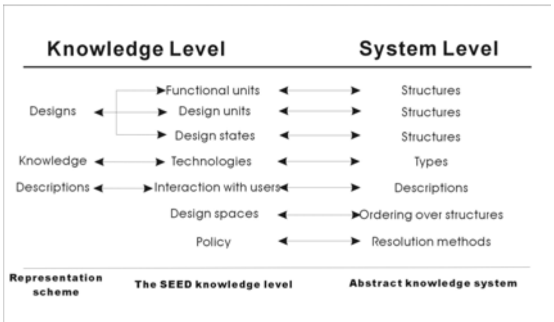

Figure 1 Mapping among a Representation Scheme, the SEED Knowledge Level Concepts, and a Generic Abstract Knowledge System

Functional units and design units describe the design information utilizing the function-form separation. Technologies are collection of computational mechanism for problem solving in a design context. In addition, design states and design spaces are the high level structures of the SEED knowledge level, which organise the form of generation of alternative designs by utilizing FUs, DUs and technologies. A

[1] Dennett would use the term "the design space" or simply "design space".

generic mapping from representation scheme to the SEED knowledge level concepts is illustrated in Figure 1. The mapping from KL to SL in a design space explorer implementation is also shown in the right side of Figure 1.

2.2 Subsumption Ordered Design Spaces

Using the functional decomposition/model mechanism described before, we propose a representation scheme called Geometric Typed Feature Structures (Chang 1999; Chang and Woodbury 2000) from theory called typed feature structures (TFS)(Carpenter 1992; Woodbury et al. 1998). Within such representation, a further mapping between knowledge level and symbol level for subsumption-based design space explorer is shown as Figure 2. We argue that such representation can provide a tractable symbol level formalism for functional and structural description of design, especially geometry. Figure 2 shows the mapping from abstract knowledge system to TFS mechanism.

The TFS symbol level prescribes some qualities that are also met in certain algebras over geometry. We can therefore use the same subsumption relation to structure the design space exploration. The trick is showing that geometric operators correspond to the design space movement operators reported in (Woodbury, Datta and Burrow 2000).

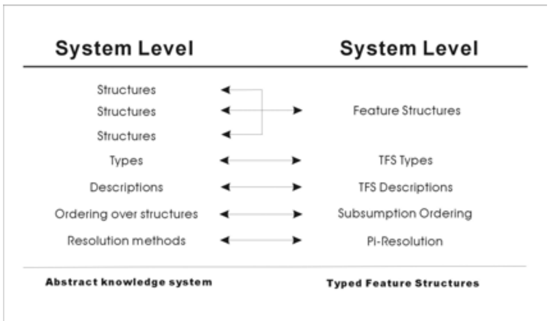

Figure 2 Mapping from an Abstract Knowledge System to Typed Feature Structures Mechanisms

2.3 Extending TFS for Geometry—Geometric Typed Feature Structures

An extended theory of typed feature structures gives a representation for what we have described as functional decompositions. Accounts of this representation appear elsewhere (Woodbury et al. 1999; Woodbury, Datta and Burrow 2000); here we summarise its essential structure.

Typed Feature Structures comprise four elements: a set of atomic elements called types organised in an inheritance hierarchy, a frame-like representation (feature structures), and the algorithms (mainly subsumption) over those. The main characteristics for this formalism are that the information has to be partial and specialization has to be unique.

The crucial insight behind extending TFS to geometry is to discover useful algebraic structures of geometric objects affording the mathematics required of TFS. Surprisingly there are many such structures each giving a way to sculpt geometric form within the tight constraints of the TFS theory.

Our extension to Typed Feature Structures begins with the observation that the geometric objects have indefinite numbers of sub-parts, none of which can be privileged over another. To say it explicitly, we are representing shape in Stiny's sense (Stiny 1980). Thus there can be no general a priori assignment of features to the geometric objects. This excludes the use of typed feature structure features as a representation device and leaves only the type hierarchy to carry geometric information.

Without features, type hierarchies are simply lower semilattices with the additional required property that, whenever upper bounds exist, there is a least upper bound amongst them. All lattices meet these conditions, as do other structures.

What this means for the geometry is that we can use as a type hierarchy of any geometric algebra whose operators induce a lattice onto the algebra. Of course, this is an overly stringent condition, but does admit many common structures. We interpret algebraic operations as movements to new types in the hierarchy. We call such types order types as distinguished from the succession types that employ the features by which the feature structures were originally named.

It suffices to have the following:

- A suitable algebra
- A way or ways of calling out elements of the algebra. We must be able to access arbitrary objects in the algebra
- Efficient algorithms for comparing and combining objects

By providing such algebra, each geometric object within our framework is tractable and has sufficient formal properties for inclusion into a formal design space. Two such algebras are described in the following example that is based on the operations of set.

2.4 An Example of GTFS

For a simple geometric operation such as insertion of a window, the formalism requires a suitable algebra for description of this geometric operation. For example, insertion of windows can be described as two algebras including adding a window within a wall, and punching a hole on the wall provided. These algebras specify two separated information-adding movements that are represented in two type hierarchies (shown in Figure 4). One derivation of a design space will use subsumption resolution method to create one instance of insertion of windows. The following sections will demonstrate such concept in details.

Algebras for insertion of a window are 1) punching a hole in a wall and 2) adding a window (shown in the Figure 3) that are represented as PunchHole and AddWindow respectively. Within GTFS, the algebra has to fulfil the *bounded complete partial* order (BCPO) conditions (thus, reflexivity, anti-symmetry and transitivity). Together with the geometric information, they form a set of GTFS types. In this case, they are $GTFS_{PunchHole}$ and $GTFS_{AddWindow}$.

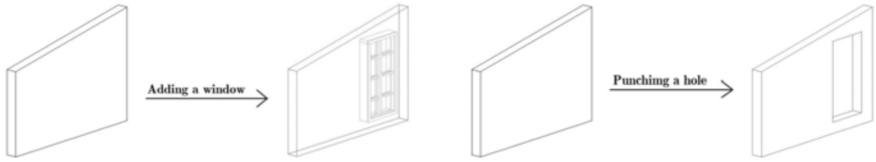

Figure 3 Punching a hole and adding a window are two informational adding algebras.

The algebra describes one way of informational adding to the extent of GTFS, and must satisfy the following condition:

For $a,b \in GTFS_{PunchHole}$, there exists algebra $\circ$ such that the following conditions hold:

- $a \circ a$ (for reflexivity)
- $a \circ b$ and $b \circ a$ implies $a \equiv b$ (for anti-symmetry)
- $a \circ b$ and $b \circ c$ implies $a \circ c$ (for transitivity)
- Existing a bottom and a least upper bound (for the bottom and a upper bound over this ordering)

Same conditions can be applied to algebra Θ for AddWindow, such that two GTFS type hierarchies ($GTFS_{PunchHole}$ and $GTFS_{AddWindow}$) exist (examples are shown in Figure 4). Each type hierarchy has its own algebra.

- $a\Theta a$ (for reflexivity)
- $a\Theta b$ and $b\Theta a$ implies $a \equiv b$ (for anti-symmetry)
- $a\Theta b$ and $b\Theta c$ implies $a\Theta c$ (for transitivity)
- Existing a bottom and a least upper bound (for the bottom and a upper bound over this ordering)

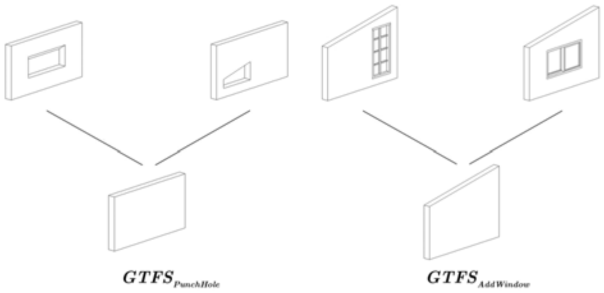

Figure 4 Two GTFS Type Hierarchies: $GTFS_{PunchHole}$ and $GTFS_{AddWindow}$

For the efficiency geometric computational mechanism, there must exist a way for calling out part of algebras and the efficient computational algorithms for PunchHole and AddWindow over geometry. For insertion of windows, the Boolean operation (for punching a hole) and normal geometric operation (for adding a window) are sufficient and efficient enough for generating the geometry for this example. Since insertion of window will not involve complex geometric operation, the simple two-manifold geometry will be sufficient enough for representing the geometry in the GTFS types. The external modelling purpose is represented as a subsumption process that will subsume two parts of design (punching the hole and adding a window) and unify them together as an instance of insertion of window. An exemplary subsumption process for insertion of a window is shown in Figure 5.

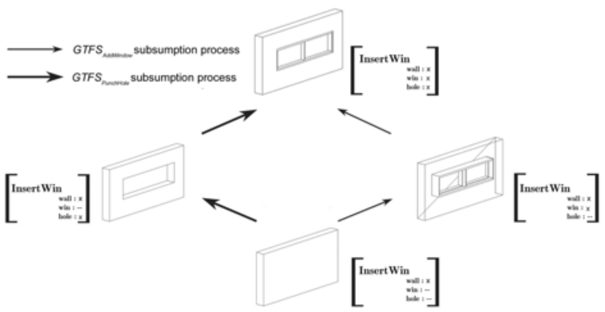

Figure 5 Insertion of a Window in a Subsumption Process

3 UNFOLDING GEOMETRY INTERACTIVELY

Within the framework of Geometric Typed Feature Structures (Chang 1999), sketched above, we have found a way to admit some geometric computations as proper movements in the formal space. Although, how much still requires a further investigation. Consequently, the clear division between the knowledge level and the symbol level, such that functional decomposition → formal symbol level and design → model symbol level, disappears. Again using insertion of windows as an example, there exists many way of windows and different hole that can be punched in a given design problem—the wall. Following the interactive subsumption process, the information that designers have in mind has to be elaborated using the symbol level mechanism, thus, the geometric description that will specify the geometry further down the space. In Figure 5, the final result is done through two different GTFS types in two different subsumption processes interactively by the mix-initiative metaphor.

Burrow's mechanism for incremental resolution anticipates the need for interaction by providing fine-grained opportunities for user intervention in the fundamental design space movement algorithms. Datta's forthcoming thesis develops unfolding as a user-level metaphor for dealing with this fine-grained interaction. Through unfolding we can interact with both the subsumption-based representation as well as some part of the geometry. With the interactive mapping of GTFS, all the complex geometric information can be formalized within this formalism.

Unfolding a design space interactively, as conceived in the above terms, entails that the following conditions be met:

- Preservation of the formal properties of the design space movement algorithms in both functional decomposition and geometric objects
- Incorporation of user-guided exploration for mapping geometric models to their corresponding decompositions.

Both these conditions are handled by conceiving the process of exploration as mixed-initiative interaction (Datta, Burrow and Woodbury 1998) between the user and the formalism. In the case of functional decomposition problems, mixed-initiative interaction enables both the designer and the formalism to explore the problem space through turn-taking. The designer is able to construct queries and to navigate around the possible decomposition hierarchy. The formalism is able to resolve queries into intermediate partial solutions that extend the space of decompositions and preserve their ordering under the relation of subsumption.

In the case of geometric exploration, the designer is able to define models through addition and subtraction operators, as described in the previous section. The fine granularity of design space exploration operators and interactions implies that, in conceiving geometric operations in the design space, it may well be useful to decompose common geometric operations, such as insertion of a window, into a composite operation in which each component is compliant with the formal design space restrictions. If this is possible, an apparently non-monotonic whole can be understood as a design space movement through a sequence of information adding and information "removing" moves in design space. Once these are defined, the interactive process of unfolding the subsumption hierarchy enables the user to move in the design space. While these advances have enabled a deeper understanding of complementary role of the user in formal exploration, an underlying assumption has been that the relation between functional decompositions and geometric structures (design units) is a one-to-one mapping. The extension to more flexible forms of mapping remains an area of investigation.

Summarising, user intervention in design space explorer can be formalized as the interactive mapping from the knowledge level to symbol level within our definition of design space exploration. This mapping provides the fine-grained opportunities for user intervention in formal design space movement algorithms incorporating geometric objects based on GTFS. This is to say that such movement commits us (as designers) to certain (and only certain) geometric features as we move along a subsumption branch.

Furthermore, using unfolding as a user-level metaphor for dealing with the fine-grained interaction, provides formalism for preserving the formal properties of the design space movement algorithms as well as incorporating user-guided exploration in the design space. Based on the GTFS approach, the interactive mapping is via mixed-initiative interaction that enables designer and the formalism to explore the problem spaces through turn-taking. The designer constructs the geometric description that satisfies the formalism through navigating the subsumption process. Therefore, the ordering nature of functional decomposition can be preserved under the subsumption relation. The interactive model with GTFS shows a richer potential, and we do not yet know its bounds. In addition, the interaction problem of design space exploration is magnified via this approach.

4 CONCLUSION

There are two main problems addressed in this paper for integrating geometry with design space exploration formalism. Firstly is the conflict between the relational natures of geometric objects with the function decomposition within the design space exploration formalism. Secondly, the grounded nature of geometric design descriptions is opposed to the partial characteristic of TFS descriptions. For both satisfying the tractability of DSE as well as ordering geometric information with other knowledge level concepts, formalism, Geometric Typed Feature Structures is exploited within the TFS theory. Using a set of geometric operations and suitable algebra, GTFS combines the formal descriptions of the functional decomposition with its informal model. The clear separation between knowledge level concepts and symbol level representation thus is replaced with an interactive unfolding process. This approach pushes the design space exploration research further into an integrated framework of geometry and user-interaction within a restricted formalism—subsumption based design spaces. By formally interacting through fine-grained mixed-initiative unfolding, the subsumption-based design spaces and some parts of the geometry can be explained as a design space movement through a sequence of information adding and information 'removing' moves in design spaces.

REFERENCES

Burrow, A. 2002. *Typed Feature Structures and Design Space Exploration,* Ph.D. Dissertation. Department of Computer Science, The University of Adelaide.

Dennett, D.C. 1996. *Darwin's Dangerous Idea: Evolution and the Meaning of Life*. Touchstone Books.

Carpenter, B. 1992. *The Logic of Typed Feature Structures*. Cambridge Tracts in Theoretical Computer Science, Cambridge University Press.

Chang, T.W. 1999. *Geometric Typed Feature Structures: Towards Design Space Exploration*, Ph.D. Dissertation. The University of Adelaide.

Chang, T.W. and R.F. Woodbury. 2000. Geometric Typed Feature Structures: Carrying Geometric Information using Typed Feature Structures. *Construction Information Technology 2000*, ed. G. Gudnason, 166-177.

Chien, S.F. 1998. *Supporting Information Navigation in Generative Design Systems*, Ph.D. Dissertation. Department of Architecture, Carnegie-Mellon University.

Datta, S., A. Burrow and R.F. Woodbury. 1998. Mixed Initiative in Design Space Exploration. Poster Abstract. *Proceedings of the 1998 Australasian Computer Human Interaction Conference*, *OZCHI `98, Adelaide*.

Flemming, U. and R. Woodbury. 1995. Software Environment to Support Early Phases in Building Design (SEED): Overview. Journal of Architectural Engineering. vol. 1, 1995, 147-152.

Gero, J. 1994. Towards a Model of Exploration in Computer-aided Design. *Formal Design Methods for CAD*, eds. J.S. Gero and E. Tyugu, 315-336. Elsevier.

Heisserman, J. 1991. *Generative Geometric design and Boundary Solid Grammars*, Ph.D. Dissertation, Department of Architecture, Carnegie-Mellon University.

Newell, A. 1982. The Knowledge Level. Artificial Intelligence, 18: 87-127, 1982.

Smithers, T. 1998. Towards a Knowledge Level Theory of Design Process. *Artificial Intelligence in Design '98*, eds. J.S. Gero and F. Sudweeks, 3-21, Kluwer.

Stiny, G. 1980. Introduction to Shape and Shape Grammars. *Environment and Planning B: Planning and Design* 7(3): 343-352.

Woodbury, R., S. Datta and A. Burrow. 2000. Erasure in Design Space Exploration, *Artificial Intelligence in Design 2000*, ed. J.S. Gero, 521-544. Key Centre For Design Computing, Worcester, Massachusetts.

Woodbury, R.F., A. Burrow, S. Datta and T.W. Chang. 1999. Typed Feature Structures and Design Space Exploration. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 13: 287-302.