Paper presented at the ICAI'11 - The 2011 International Conference on Artificial Intelligence
http://www.world-academy-of-science.org/worldcomp11/ws/program/ica18

Artificial Intelligence Techniques for Understanding Gothic Cathedrals

Stefaan Van Liefferinge[1], Charles Hollingsworth[2], Rebecca A. Smith[1], Michael A. Covington[2], and Walter D. Potter[2]
[1]Lamar Dodd School of Art, The University of Georgia, Athens, Georgia 30602, USA
[2]Institute for Artificial Intelligence, The University of Georgia, Athens, Georgia 30602, USA

Abstract— *This paper introduces a work in progress on a new research topic for artificial intelligence. It presents the first stages of research that investigates the capacities of AI to understand the architectural rules that define historic architecture: specifically those of Gothic cathedrals. Gothic architecture has its own logic. It follows rules that are defined by physical and structural constraints as well as by the style of the architecture. This project will analyze specialized descriptions of Gothic architecture and translate them into computer code. On the long run this will result in an intelligent computational model, a program that can understand the structure of a Gothic cathedral and reason about its architecture. This paper introduces the background of this project, sketches a small knowledge base illustrating the approach, and describes the potential impact of this type of research on the study of historic architecture.*

Keywords: Gothic Cathedrals, Architectural Descriptions, Knowledge Representation, Text Understanding, Logic, Computational Reasoning

1. Introduction and overall method

The limits prescribed by architectural design make built structures a good subject for AI experimentation.[1] It is thus not surprising that a number of projects have brought architecture within the scope of AI. For example, architectural historians have indicated how patterns in the language of architecture can be approached as if it were code [1]. Also engineers have investigated how to use AI for architectural research and design. Some have proposed systems to obtain the logical configuration of built structures [2]. Others have developed algorithms for producing computer-generated models of historic architecture [3], [4]. These initiatives underscore the affinity between AI and architecture. However, up to now, research has focused on the buildings themselves without addressing an intermediate medium as, for example, written architectural descriptions.

[1]This research benefited from the generous support of a Digital Humanities Start-Up Level I Grant from the National Endowment for the Humanities (Grant Number HD5110110), a University of Georgia Research Foundation Grant, and from The University of Georgia President's Venture Fund.

This research project involves a method that utilizes a translation of architecture into a logical system. More precisely, it takes advantage of the existence of architectural descriptions in the literature, descriptions that are expressed in a natural language and that follow specific rules. Indeed, the conventions which architecture generally obeys have defined a language, a structured and logical means of describing architecture. Architectural historians commonly manipulate this tool to communicate their analysis of buildings. These descriptions reflect the structure and rules of architecture, and, conversely, the conventions that architectural texts follow are well adapted to describing built structures.

Thus, instead of searching for a direct mapping between actual buildings and an intelligent computer system, our research investigates the application of AI technologies for understanding written descriptions of buildings. For this task a subset of architecture, Gothic cathedrals, was chosen [5]. This subset is well adapted because Gothic cathedrals are known for being organized following recurrent patterns. Their internal organization is valuable and serves as a heuristic to codify the logic of their architecture.

While describing architecture is a practice in obeying conventions, it is nonetheless a human activity that combines presupposed knowledge, intelligence, and creativity. These factors put this research project well beyond coding or modeling. It searches to catch the processes involved in a specific discipline - architectural history - and to bring these within the reach of computational logic. Ultimately, we would like to implement a system that can automatically translate between English and a computer, and which can reason about any description of architecture.

In broad lines, the adopted method of research is as follows. Schemes commonly used to describe Gothic cathedrals are analyzed and categorized. This classification is used to produce an architectural description language (ADL), a simple subset of English that is sufficient for describing the architectural features of interest. In this ADL we write a generic description that captures those features common to most Gothic cathedrals. Descriptions of particular cathedrals need only describe how they differ from this canonical description, as described in [6]. However, the translation of architectural descriptions into formal logic is no simple

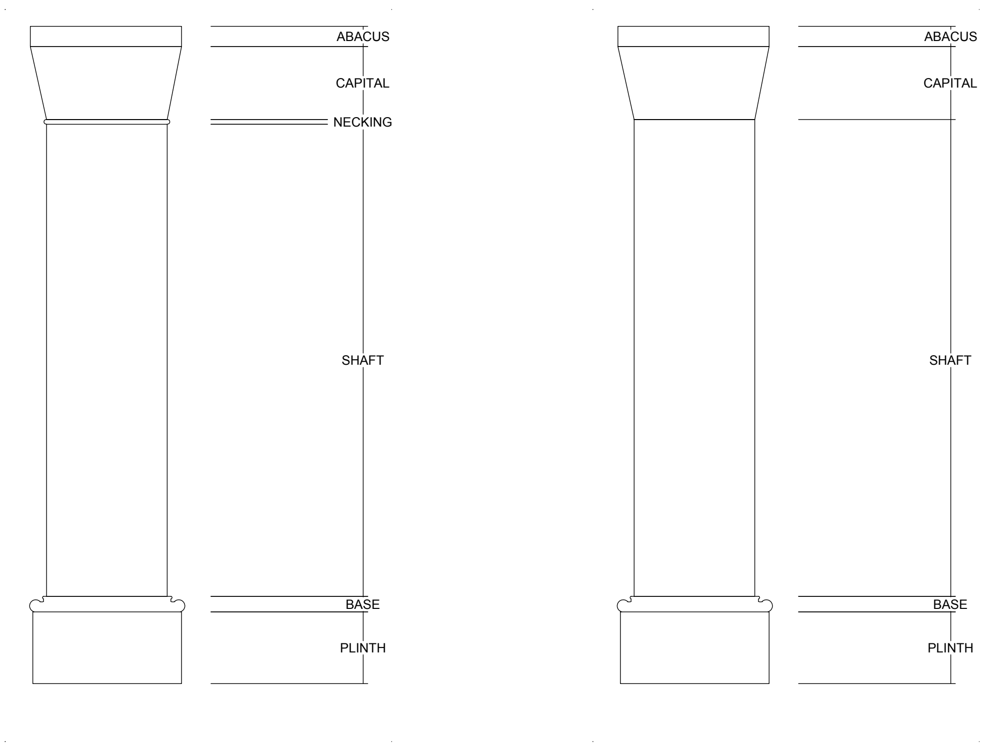

Fig. 1: Column, with and without necking

```
shaft([shaft_inst(1),X]) :- column(X).
has(X, [shaft_inst(1),X]) :- column(X).
```

Fig. 2: Prolog translation of "Every column has a shaft."

```
immediately_above(X,Y) :-
    necking(X),
    shaft(Y),
    has(ParticularColumn, X),
    has(ParticularColumn, Y).
```

Fig. 3: Prolog translation of "If a column has a necking, it is immediately above the shaft."

task. For this, at this stage, we are experimenting with the translation of a small subset of architecture, that of a column, into a declarative language and testing how a computer can operate and interactively reason with this subset.

2. A Minimal Case Study

Because the project aims at identifying knowledge presupposed in descriptions of architecture, knowledge representation is of central importance to our project. Architectural descriptions must be translated into a formal, machine-readable format that can be used for responding to queries and drawing inferences. We decided to render these descriptions as source code for the logic programming language Prolog, because the Prolog interpreter already incorporates question answering and inference. Additionally, much work has already been done on translating from natural language to Prolog [7], [8].

We chose first to devise a knowledge base for describing Gothic columns, with the intention of expanding this later to include other architectural features. Columns were chosen because, while they play a fundamental role in a cathedral, they are fairly simple structures. Furthermore, the features in a column are arranged very simply, one on top of the other, and the order does not vary. However, certain features, such as the base, capital, and shaft, are found in all columns, while others, such as neckings and plinths, are not always present (Fig. 1). We thus needed to be able to represent concepts such as "Every column has a shaft" and "If a column has a necking, then the necking is immediately below the capital." Finally, we needed to account for the fact that many architectural features are repeated, so that one might encounter statements such as "Every vaulting unit has four columns." For this last requirement, we made extensive use of skolemization [9], a well-known method for eliminating existential quantifiers. The use of skolemization in logical programming is covered in [7], [10]; our implementation is a modified version of that described in [8].

The Prolog translation of the assertion that "Every column has a shaft" is in Figure 2. The Prolog term [shaft_inst(1),X] is a Skolem function, a way of providing a unique identifier for each shaft that is a function of the name of the column of which it is a component. In other words, we are asserting two things: for each column X, there exists a unique shaft named [shaft_inst(1),X]; and column X has that shaft as one of its components. These identifiers are admittedly unwieldy, but should only be used internally by our program: the end user should never have to type them as part of a query. The listing in Figure 3 illustrates how to say "The necking is immediately above the shaft," while bearing in mind that not all columns will have a necking.

This states that, if some particular column has both a necking and a shaft, the necking is immediately above the shaft. Elsewhere we deal with columns without neckings, in which case it is the capital which is immediately above the shaft. This is in fact the default, and if we simply mention a column without specifying whether it has a necking, the system assumes it has none. However, if we subsequently assert that the column does have a necking, the previously derived facts about the relative locations of the capital and shaft must be revised. To do this we make use of nonmonotonic reasoning [11], [12], a form of reasoning that allows previously deduced assertions to be defeated by new evidence. Our implementation is similar to that presented in [13].

At present, here is a very basic set of queries which the user can make to our knowledge base. To get a list of constituent parts of a column named column1, for example, the user can type

```
?- has(column1, X).
```

into the Prolog interpreter. For an exhaustive list of the components of vaulting unit v1 and their various sub-components, type:

```
?- has(v1,X), has(X,Y).
```

To get a list of all the components below the capital in column1, type:

```
?- has(column1, X), capital(X), above(X, Y).
```

While this simple example already captures some essential capabilities needed for architectural description, we will expand this knowledge base to deal with more and more sophisticated concepts. Eventually, a natural language front end will take the place of the Prolog query interface, making the system more usable, and paving the way for eventually being able to handle more typical architectural texts.

3. Future Impact

Both architectural history and artificial intelligence will benefit from the combination of technology and traditional methods of building analysis. It will allow architectural historians to understand better how we write architectural descriptions, will be a beneficial study tool for students and professionals alike, and will eventually be able to give us accurate visuals of buildings. Engineers working in artificial intelligence can expect from this project an entirely new scope for natural language processing, and to apply artificial intelligence to a previously little-explored field.

Writing for architectural history is a complex and often subjective process. The number of assumptions made when writing a description of a building, particularly when writing about Gothic architecture, is vast. A program that could "read" and analyze architectural descriptions would be entirely new. Such a program would be able to detect trends among various essays and treatises that scholars are often unaware exist. These trends could be as simple as the order in which a building elevation is described, to something as complex as the nature of the geometrical pattern from which the plan is derived. The discovery of patterns within written text could further the study of natural language processing by highlighting how humans describe buildings. It could show how we translate the experience of a building into written description – and possibly underscore the limitations within writing. Previously unknown peculiarities within individual buildings could be found and studied by experts, without affect from the voice of a particular author or the subjective analysis of future scholars.

One of the long-term goals of this project is to create a program that can generate models that integrate 3D images of buildings, expert knowledge of architectural historians, and archaeological data. This can have a profound effect on the study of architectural history and provide a fascinating field of exploration for artificial intelligence. For example, by using written sources, the system could create a building design or plan that is historically more valid than that of a human scholar drawing by CAD or by hand. Applications of this type of image creation program should also be of interest to engineers applying AI methods in the virtual game design industry. Engineers could create buildings and cities that were previously limited to the human imagination. The program could use historic written accounts to understand better what the original author actually meant to describe, even if the account is mistaken or lacking. This will allow us to create models and imagery of buildings that exist only in descriptive accounts.

Often drawings of buildings that no longer exist or may have never existed are inaccurate and misleading. This project opens a door to the generation in the future of accurate and detailed elevations, floor plans, and virtual tours. Such models could, for example, inform preservationists about previous structural interventions in buildings. This would enable more accurate repairs, which would lead to the continuing survival of many of the world's treasures. The cooperation between these two seemingly remote fields will be mutually beneficial, and likely foreshadows the coming of new interdisciplinary studies – combining artificial intelligence with both architectural history and other fields in the humanities and sciences.

References

[1] William J. Mitchell. *The logic of architecture: Design, computation, and cognition*. MIT Press, Cambridge, MA, USA, 1990.

[2] Atsushi Saito, Junzo Munemoto, and Daisuke Matsushita. Acquiring Configuration Rules of Form Elements from "Historic" Architectural Facade Employing Inductive Logic Programming. T. Washio et al. (Eds.): *JSAI 2005 Workshops, LNAI 4012*, 190–200, 2006. Springer-Verlag, Berlin-Heidelberg.

[3] Yong Liu, Congfu Xu, Qiong Zhang, and Yunhe Pan. The smart architect: Scalable ontology-based modeling of ancient Chinese architectures. *IEEE Intelligent Systems*, 23:49–56, 2008.

[4] Yong Liu, Yunliang Jiang, and Lican Huang. Modeling Complex Architectures Based on Granular Computing on Ontology. *IEEE Transactions on Fuzzy Systems,* vol. 18, no. 3, 585–598, June 2010

[5] Paul Frankl. *Gothic Architecture.* Revised by Paul Crossley. Yale University Press: New Haven, 2000.

[6] Charles Hollingsworth, Stefaan Van Liefferinge, Rebecca A. Smith, Michael A. Covington, and Walter D. Potter. The ARC Project: Creating logical models of Gothic cathedrals using natural language processing. *Proceedings of ACL 2011 Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities* (forthcoming)

[7] Patrick Blackburn and Johan Bos. *Representation and inference for natural language: A first course in computational semantics*. CSLI Publications, Stanford, CA, USA, 1999.

[8] Michael A. Covington, Donald Nute, Nora Schmitz, and David Goodman. *From English to Prolog via discourse representation theory.* Technical report, Advanced Computational Methods Center, The University of Georgia, 1988. http://www.ai.uga.edu/ftplib/ai-reports/ai010024.pdf (available as of May 2011).

[9] Thoralf Skolem. Über die mathematische Logik (Nach einem Vortrag gehalten im Norwegischen Mathematischen Verien am 22. Oktober 1928). In *Selected Works in Logic.* Jens Erik Fenstad, ed. Universitetsforlaget, Oslo - Bergen - Tromsö, 1970, 189–206.

[10] Jeffrey Cua, Ruli Manurung, Ethel Ong, and Adam Pease. Representing Story plans in SUMO. In *Proceedings of the NAACL HLT 2010 Second Workshop on Computational Approaches to Linguistic Creativity.* Association for Computational Linguistics, Los Angeles, California, June 2010, 40–48.

[11] Grigoris Antoniou. *Nonmonotonic Reasoning.* The MIT Press, Cambridge, MA, USA, 1997.

[12] Raymond Reiter. Nonmonotonic Reasoning. *Annual Review of Computer Science,* 1987.2: 147–186.

[13] Donald Nute. Defeasible Logic. In *Proceedings of the Applications of Prolog 14th International Conference on Web Knowledge Management and Decision Support* (INAP'01). Oskar Bartenstein, Ulrich Geske, Markus Hannebauer, and Osamu Yoshie, eds. Springer-Verlag, Berlin-Heidelberg, 151–169.