Parametric modeling in AutoCAD

By Neil C. Katz
Skidmore, Owings & Merrill LLP
October 11, 2008

A version of this article appeared in “AECbytes” in 2007.

Building information modeling and parametric modeling are very current themes in today's architecture. Many people associate BIM and parametric modeling with particular software tools. My belief is that both building information modeling and parametric modeling are more related to the approach taken in the design process than the software tools an architect uses.

The examples provided here illustrate the use of a simple tool (in this case, “simple” is not necessarily a negative adjective) used to study and model complex geometric forms, particularly at the earliest phases of a design project, analyzing and even adjusting these forms with respect and in response to environmental conditions such as sun and shadows, zoning criteria, views, and size (floor areas and program verification, façade surface areas, volume).

The examples also illustrate that the approach one takes in solving a problem—thinking creatively and not being constrained by one’s own thinking or by a limiting set of tools—is the key to innovative designs and design processes. Often a simple and flexible tool can be more helpful in solving a problem a very specific way than a very sophisticated tool design.

What is parametric modeling?
Building models are representations of buildings. Architects are experts at modeling buildings. They can conceptualize a building in their minds. They must often document the building to be able to record their concepts and ideas, and especially to be able to share them. These documents are building models. Plans, elevations, and sections are examples of building models. Renderings are examples of building models. Physical scale models are examples of building models. Virtual representations in a computer of a building are also building models.

Building models can be explicit. Every aspect of the model is well-defined and can be described, typically independently without referring to other parts of the model. In a computer model in which a building is represented explicitly, we can, for example, get the coordinates of any point in the building, and from this information, even create various different representations (such as plans and sections, renderings, physical models), and even create the real building. During the design process, where decisions about the building are being made as the model is being built, aspects of the model are built, tested (by various methods), and continually modified.
Parametric modeling in AutoCad

Parametric building models are a bit different. Aspects of the model depend on relationships between parts of the building. Creating and modifying these relationships is an important part of the design process. A parametric model is often defined by rules and constraints, which define aspects of the building and their relationships to each other. Changing a rule or constraint, or modifying a part of the model itself, often has implications in the entire model.

In the first type of model, the geometry is explicit and the rules are implicit (there are always rules and constraints in an architectural model, but the modeling tool will not keep track of them so we have to); in the second type, the rules are explicit and the geometry is implicit.

There are several tools available which are designed to create parametric architectural models, including Digital Project (an application based on Catia by Dessault Systems and developed for architects by Gehry Technologies) and Generative Components (running within MicroStation by Bentley Systems). These tools are fairly recent, developed in the past decade. SOM does currently use these types of tools.

Parametric tools allow relationships to be defined among components in the model, and parameters which control aspects of the model to be defined and changed. We can describe a building, for example, as an extruded rectangular form with a pitched roof. As we change the dimensions or shape of the rectangle or the height of the extrusion, the roof may change to still fit perfectly and perhaps not exceed a total height constraint, and even warn us if conditions cannot be satisfied. We can say we are modeling the building by describing the rules which the building must follow. In conventional modeling (where we explicitly describe the building), when we change one aspect of the model, we must often make several changes to satisfy our design intent (the implicit rules of the design), and in addition, the software does not keep track of the rules and we must decide whether and when they are broken.

In AutoCAD, we can create models in a conventional way. AutoCAD, however, comes with a programming environment (more than one), with which we can create a set of instructions, including the rules and constraints of the design (as well as parameters defining certain aspects of the design), which can be used to build the model. We can use these instructions to always build a model from scratch, each time using the same parameters, or experimenting with different ones. The parameters can be numeric values (FootprintWidth = 120'-6", FootprintDepth = 84'-0", MaxBuildingHeight = 78'-0", …), relationships (FootprintDepth = FootprintWidth*0.6666), and can even include graphic parameters already existing in the model (a building lot, context buildings, a zoning envelope, …). The programming environment allows variables to be defined, allows conditional branching to different sets of instructions on the program, can repeat instructions until a condition in the program or model is met, and can interact with the model getting information from and adding to the model.
In late 2001, SOM was asked to study a master plan and buildings to replace those at the World Trade Center in New York. The images here are of a tall tower on the site, and developed as a result of a close collaboration between our design team and structural engineering team. Subsequent to this effort, a competition was held for the design of the master plan of the site. While these designs were not realized, they are an excellent example of computational design with parametric modeling. Some of the images shown here were displayed at the Venice Biennale in 2002.

The examples above are views of a model generated by a lisp program. The building’s structure is modeled, and based on a diagrid supporting a building which is cylindrical in form. Two examples (of almost one hundred modeled) are shown here. In each case a single diagrid member, which spirals from the base of the building to the top, is created by the program, and repeated (by rotation and reflection) to create the entire structure. In the second set of images, as in most of the models, the member varies from bottom to top, in this case by splitting (one member at the base splits into four members) and by tapering.
Another example of a lisp-based model is illustrated here, in which a frit pattern is generated based on simple graphics in the model which control the parameters (the "rules") which affect the generation of the pattern. In each case, the red line is the centerline of the pattern, the blue line determines the spacing of the dots, and the green line determines the size of the dots.

These models were created for the Memorial Sloan-Kettering Mortimer B. Zuckerman Research Center in Manhattan.
Parametric Modeling in AutoCad

The architects working on this project set out to minimize curtain wall opacity as much as possible in order to create a dialogue between the interior and exterior of this research facility. Located in the heart of a mostly residential neighborhood on New York’s Upper East Side, transparency was a key design criteria as the client wanted to create a friendly exchange between residents and researchers.

After researching various types of low-emission coatings, designers realized that these metallic films alone would not allow the desired transparency. So they developed a method of combining low-emission coating with a ceramic frit pattern to minimize the amount of sunlight that could penetrate the building’s façade, thereby reducing solar heat gain and allowing the building to surpass city code requirements for energy efficiency.

The research done specifically on the frit patterns has since been extended, and has been used for other applications as well, such as perforations in metal panels.

Solar Incidence Angle Analysis | Lotte Super Tower, Seoul

The Lotte Super Tower project was won through a competition.

Many of the processes described here were used in the design, analysis, and documentation of this building.

The building model is created both as a 3-dimensional model and as an unfolded model (for laser-cutting, as well as for representation). The program generates one quarter of the structure, and uses symmetry to complete the models. Parameters in the program control the diagrid; the parameters are refined after many iterations to optimize structural performance, program area contained within the building, and aesthetic judgments.
Parametric Modeling in AutoCad

Solar incidence angles: morning, springtime

Solar incidence angles: afternoon, autumn

Solar incidence angles: complete set

Solar incidence angles: analysis summary

The analysis summary was part of our competition entry package, and represents a solar-incidence-angle analysis of the tower. An analysis was performed for each facet of the tower model, each hour from morning to evening for one day of each month, comparing the normal vector of each facet to the direction to the sun. If this angle is small, that facet is getting direct sunlight, causing a probable negative impact on energy performance and building occupant comfort. If the angle is small, we colored that facet red, and as the angle increased—and the effect got less negative—the colors changed to orange, yellow, green, and blue (blue represents the least negative effect). While the analysis was done on a three-dimensional model of the project, the results are shown on an unfolded model, allowing us to see a ‘report’ of the analysis for the entire building in a single image. These results were summarized to create the image above.