ASME 2023 International Design Engineering Technical Conferences
and Computers and Information in Engineering Conference
IDETC-CIE2023
August 20-23, 2023, Boston, MA, USA

DETC2023-####

EXTENDED PRESENTATION ABSTRACT

DYNAMIC ESTIMATOR OF CAD PATTERNING FEATURE EXECUTION TIME

John Morris[1], Dave Procopio[1], John Wagner[1]
[1] Department of Mechanical Engineering, Clemson University, SC, USA

Ubiquitous in CAD software is some variation of a patterning tool capable of replicating select seed geometry in a prescribed manner. Patterning features are differentiated by layout; common layouts include uniform grids, radial arcs, and replication along a predefined path. In addition to layout, patterning tools can differ according to their method of replication. Though the associated conference paper [1] and this short article focus on tool methods provided by the Siemens NX software, these methods are commonly found with minor differences in many other packages. Provided in NX are three patterning methods: Feature Pattern (FeP), Face Pattern (FaP), and Geometry Pattern (GP). These three methods offer varying tradeoffs on fidelity, accuracy, speed, and memory usage. FeP recreates the selected seed feature at every instance, returning pattern features with individually solvable references. This greater accuracy and fidelity are more computationally expensive, with diminished speed and larger file sizes. Both FaP and GP generally reduce the fidelity of the replicated instances while decreasing execution time. Pattern complexity is a function of the complexity of the seed instance and the number of instances to be created. Seed features with geometric entities replicated many times create computationally expensive patterns that tax hardware systems during creation and subsequent product updates.

One example of products requiring computationally intense patterning features are gas turbine blades. The interior of these blades is often covered in small protuberances called turbulators that promote heat transfer through the blade by initiating turbulent air flow near the blade’s surface [2]. The fluid dynamics resulting from these turbulators is difficult to establish analytically and requires numerical techniques —these digital analyses subsequently require a robust geometric model of the blade and turbulators be developed in CAD. These models can have hundreds or thousands of turbulator instances which require a significant amount of turbine blade design time to be spent on producing and updating large-scale pattern features.

In this extended abstract, a design tool is presented to estimate the time necessary to complete a pattern feature. This estimation

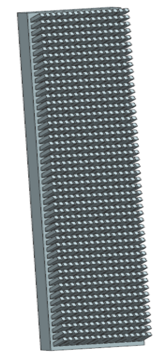

Figure 1: Example block showing patterned turbulators patterned in a rectilinear grid

can help guide design decisions in two ways. First, the tool highlights differences in pattern methods (FeP, FaP, and GP) which provides more information to the designer in selecting the optimal method. Second, the tool can alleviate the attention-demanding task of waiting for the pattern feature to complete execution. For instance, Siemens NX (and many other CAD packages) do not contain a time estimator for pattern completion. The knowledge of how long a pattern will take to execute can help the designer plan design activities, including when the overall model creation will be finished.

The tool described here is a plugin developed for Siemens NX, though its functionality could be employed with equal utility in other CAD platforms. When deployed, the tool appears as a graphic window that allows the designer to select the seed geometry in the model as well as enter the number of instances that would be created in a proposed pattern feature. Based on this, as well as the number of geometric entities in the selected geometry, the new tool displays an estimate for the execution time of each of the three pattern methods (FeP, FP, and GP). This time estimation is presented dynamically, without the need for further inputs from the designer.

The functional performance of the tool is based on a linear extrapolation that considers the two primary factors in pattern execution time: the number of instances in the pattern, and the complexity of the seed instance (as measured in number of geometric entities). The tool first establishes an empirical relationship between these factors and pattern execution time for an arbitrary pattern. This relationship is then applied to arbitrary inputs specified by the designer.

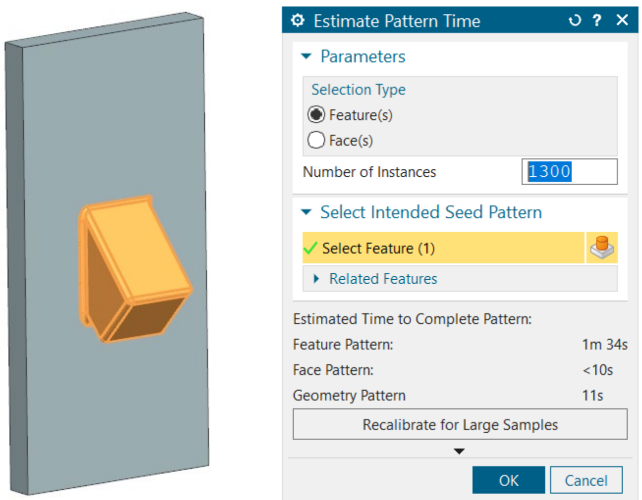

Figure 2: GUI window for design tool next to model with selected geometry. The selected geometry could then be used as the seed instance in a pattern feature.

Though the extrapolation is solely dependent on these two factors, the overall time must be scaled based on the performance of the computer platform. The tool does not assume anything about the platform (such as checking for available cores or memory usage), but rather performs an empirical analysis of the computer's performance. This calibration routine runs in the background when the tool is first deployed and consists of creating two pattern features of a simple cuboid in each of the three patterning methods (FeP, FaP, and GP). Using two pattern features provides the tool with two sample points necessary for the linear extrapolation, consequently the two patterns have a differing number of instances (set arbitrarily to ten and twenty). The tool calculates the execution time of each of these patterns to construct a linear relationship between number of geometric entities in a pattern feature and the pattern feature's execution time, specific to the computer at the current time. This relationship is specified in equation 1,

$$t(n_i, n_e) = \beta_1 n_i n_e + \beta_0 \tag{1}$$

$$\beta_1 = \frac{t^2 - t^1}{n_e(n_i^2 - n_i^1)} \tag{2}$$

$$\beta_0 = t^1 - \beta_1 n_i^1 n_e \tag{3}$$

where t is the estimated execution time of the patterning feature, n_i is the number of instances in the pattern, n_e is the number of geometric elements in each instance, and β_0 and β_1 are first-order scaling factors. The superscript (with values of 1 or 2) refers to an empirical value used or found from the respective patterning feature employed in the calibration routine. The calibration routine, in constructing the 6 unique patterning features, generally takes less than five seconds to process. It runs every time the tool is deployed to accurately map the current performance of the underlying hardware.

Extrapolation based on geometric elements is reasonably accurate for predicting pattern completion time. However, due to their high memory usage and intensive calculations, pattern feature execution can still vary in response to changing computational performance. Computer throttling, available RAM, and memory limits can cause unanticipated error into the tool's estimates. This is similar to many time estimators for computer tasks. The purpose of the tool is not to perfectly calculate pattern execution times, but rather estimate them based on current system performance. Tests of various patterns with 1,000 instances resulted in estimation errors of up to 12 seconds.

If a designer needs to test several large patterns, they may wish for better accuracy than that given by the initial calibration routine. Some aspects of throttling and large memory utilization can be accommodated by the extrapolation model if the initial sample points are sufficiently large to incur such system responses. The tool provides an option to recalibrate the model based on a much larger sample size if desired. Though this recalibration is generally more accurate for large sample sizes, it is takes far longer to compute. There is little benefit from utilizing this option for a single pattern feature, since it can take as long or longer to run.

The tool is still under active development. The latest versions of the program are available at *https://github.com/Clemson-PLMC/PatternEstimatorTool.git* under the MIT License. To utilize the plugin, users should download the source files and place them in their user directories according to the documentation produced by Siemens [3].

This plugin was written using NXOpen, the Application Programmer Interface (API) for Siemens NX, in VB.NET. It was written for NX 1980 but can purportedly be employed in any version of NX after NX 5.0.2.

REFERENCES

[1] D. Procopio, J. Morris, and J. Wagner, "Evaluation of CAE Design Change Updates—A Case Study on Gas Turbine Airflow Disruptors," presented at ASME IDETC-CIE Conference, Boston, USA, Aug 20-23, 2023.

[2] J. Han, and H. Chen, "Turbine Blade Internal Cooling Passages with Rib Turbulators," Journal of Propulsion and Power, vol. 22, no. 2, pp. 226-248, 2006. doi:10.2514/1.12793.

[3] Siemens, Getting Started with NXOpen. Siemens Product Lifecycle Management Software Inc., 2019. Accessed: Mar. 09, 2023. [Online]. Available: https://docs.plm.automation.siemens.com/data_services/resources/nx/1872/nx_api/common/en_US/graphics/fileLibrary/nx/nxopen/NXOpen_Getting_Started.pdf