A practical methodology for creating robust parametric surface-based models in automotive engineering

Michal Forraí a, Jana Gavačová a,*, Ladislav Gulan a

a Faculty of Mechanical Engineering, Slovak University of Technology in Bratislava, Námestie slobody 17, 842 31 Bratislava, Slovakia

* Corresponding author. Tel.: +421-2-5729-6860. E-mail address: jana.gavacova@stuba.sk

Abstract

The paper deals with the general description of Generative Engineering Design (GED) methodology used during the development of surface-based components. The stages of GED presented here are applied on sport vehicle parts with visible surfaces. The focus is on the later stages, i.e. detail design. These include creation of robust parametric shape-based CAD model, which is capable of easy change propagation. The GED methodology builds on Knowledge-Based Engineering (KBE) principles, because the ability of quick and easy change is essential. The practical use of the methodology is demonstrated using CAD-based generic templates and a custom tool in CATIA.

Keywords: Generative Engineering Design; GED; Knowledge-Based Engineering; KBE; Shape Design; Power Copy

1. Introduction

Product development methodology has been improving for decades. There has been room for improvement in several fields, including CAD applications. Developed methods of designing are relevant and effective only for the complex products. Development of vehicle prototype is a suitable application for the Generative Engineering Design methodology. Car body components with visible surfaces are made from wide range of materials and could have wide range of shapes [1].

Nomenclature

API Application Programming Interface
C³ Curve continuity of the order
CAD Computer-Aided Design
CAE Computer-Aided Engineering
CAM Computer-Aided Manufacturing
CAS Computer-Aided Style
COP Cloud of points
GED Generative Engineering Design
GSD Generative Shape Design
GUI Graphic User Interface
KBE Knowledge-Based Engineering
VBA Visual Basic for Applications
Generative Engineering Design is a process in which the draft or model is made based on quantitative and qualitative parameters or aesthetic inputs, using algorithms created by individual human intervention for the purpose of generating a variable set of subsequent models following accurate relations and according to hierarchical connections.

Even though similar, the fundamentals of Generative Engineering Design are different from parametric modelling. Generative engineering relies partly on generic parametric models. Parametric models are numerically controlled deterministic representations of design solutions which result in a new product with similar geometrical values (quantity indicators such as dimensions, weight etc.), but dissimilar in quality (e.g. aesthetic indicators, subjective user requirements, and needs). It means that generative design in new design and innovation offers more than a geometric model. It offers a whole complex of information about a new product which has not only a deterministic nature, but also a heuristic one.

Today, parametric modelling is a well established approach. While lower levels of parametrization (i.e. parameters and formulas) are commonly used, higher levels are still underutilized. These higher levels are made of composite geometry, which is required to dynamically adjust to variable inputs. Complex geometry brings more constraints that have to be met to produce a valid result. The parametrization levels, and some practical solutions for its high-level use are well illustrated in [3,4].

Knowledge-Based Engineering describes a wide range of CAD and CAE applications that are used in cooperation with each other, with emphasis on knowledge reusability in repetitive tasks following normal engineering practices [5]. Part of such a cooperation is related to the definition of specific parameters and other kinds of requirements on behalf of automatic design creation. GED is generally a part of such a methodology [6–8]. The proposed methodology addresses two of current shortcomings of KBE as identified in [9]. It tries to avoid the resulting model to be a ‘black box’ by providing explanations, links and context to data and formulas. The knowledge reuse problem is partly addressed by implementation in a well-established CAD software. The reliance on proven software is suitable for long-term usability or work in diverse environment, although there are limitations (lack of knowledge sharing between applications). Another problem with knowledge reuse in KBE application is the form, which usually prevents the reuse for non-programmers [10]. This is addressed by the chosen approach by relying mostly on visual and graphical workflow.

An overview of selected KBE goals is shown in fig. 1. To further leverage the time-saving and solution-finding advantages of concurrent product development cycle, it is needed to improve the speed and accuracy of information exchange. The shortened development time is attributed to more automation possibilities of repetitive tasks [9,11] and reusability of design elements. This has an impact on earlier stages, where the effect leads to better product knowledge. The improved design knowledge leads to more change opportunities, allowing wider exploration of design space [12]. In this paper, the focus is on the geometrical definition and the reusability of geometrical features.

2.1. GED methodology for development of shaped components

The unique advantage within the shape component design lies in the possibility to define models by the shape and position of surfaces. For example surface of contact is difficult to be set in solid modelling. The presented methodology is based on a proper hierarchy of operations throughout the shaped (i.e. surface-based) component modelling. Such a component is built step by step as is shown in fig. 2. The main advantage lies in the hierarchy of separate files created in CAD software:

Class A surface represents input surface. Quality of connections between patches, aesthetics, aerodynamics and passive safety are considered here.

Class B surface represents derived surface (offset from A-surface with added technological adjustments). It is important to link all the operations in a way that allows its adaptation to any changes of the class A surface.

Interface represents connection between parts. For each part pair a separate file is used, referenced by both parts (generally, an interface can be defined between more than two parts). In each interface file, separate geometrical set exits for particular interfaces (e.g. shape boundary, contact surface, clip definition). It is responsible for additional changes that can be made between parts, propagating the change to all interfaced parts. Interface creation is the best benefit of automatic modification of assembly during the subsequent stages of product development process. It means that in the case of a class A surface restyling, any affected parts would be changed automatically. This can be called a generatively transformed design.

Result represents the final result, solid or surface for analysis created from hierarchy of input parts and handling interface realization (e.g. shape trim, clip surface instantiation).

Functional features (class C surface) represents geometry creation which leads to a design of functional features such as clips, ribs, holes etc. A class C surface refers to a functional surface, which originates between parts as interface and after assembling it is constantly covered (not visible from any side). Class C surface is the main construction element of functional features.

2.2. Application of GED methodology during the virtual sport vehicle development

The proposed scheme shown in fig. 3 is suitable for designing process of vehicle body components. Development of virtual sport vehicle illustrates such a process. It is one of objectives in the Laboratory of Generative Engineering Design at the Slovak University of Technology in Bratislava. Firstly, a CAS
was drafted by designer for the whole vehicle body, which was used directly as reference for class A surfaces for rapid workflow. From CAS, a more precise mathematical description of surfaces suitable for CAD can be created using following procedure:

1. Specialized class A surface software or module loads data from CAS as a package of frozen surfaces;
2. patches of some surface areas are extracted to be able to be formed as freeform surfaces;
3. patches are connected with lower class of continuity such as C^0 or C^1 to form an intermediate model with sharp connections;
4. interfacing curves are built on patches with respect to the shape of future output model;
5. connecting surfaces of patches are created with boundary of interfacing curves with higher class of continuity such as C^2, C^3 or even higher.

After the class A surface (fig. 4) is completed, it is then used as input to class B surface file. Then all required interface files are created, so the result file contains final model. For surface-based plastic components it is important to link interfacing surface with the die direction line (e.g. interface using swept surface in fig. 5). Resulting part forms a closed surface and volume properties can be applied to it. Result can only be determined by the surface of class A surface, class B surface and interference surfaces. Separate files of boundary surfaces are linked to the result and inserted as external references. Links are active during the whole designing process and any change applied to files is automatically propagated.

The next chapter gives more details on class B surfaces, particularly its constituent repeating features that can be further individually modified (fig. 6). These are predefined surfaces of functional features (e.g. clips, ribs, etc.). The methodology to create these adheres to Knowledge-Based Engineering.

3. Template creation

The methodology described here is intended to mainly improve possibilities of editing complex instantiated geometry in arrays or patterns. It describes the most important properties, that should be considered when creating template models. These include building models in such a way that improves stability of operations on boundary representation models in dif-
ferent orientations and their data structure organisation. An example is implemented in CATIA, where a patterning tool exists, but is limited to repeating identical objects and their individual instances cannot be edited. Tools for manual associative data duplication are also available. The combination of these can further improve product development process, but robust parametric-associative models are a basic requirement.

The proposed methodology is dependant on specific CAD software capabilities. Support of parametric-associative modelling using feature-based modeller is required. Application of Knowledge-based engineering approach is facilitated by imposing a strict modelling methodology and data structure. Grouping of features into sets allows the user to define logical structure to a model, which is essential for complex models or shape models (in which features do not have to be in the order they were created). A deep specification tree, which shows the parametric nature of model by including the parameters as leaf nodes is very useful. The associativity of CAD model is not presented in specification tree, but there is another visualization for this connections (e.g. historical graph, fig. 8). Each template can have its complexity evaluated [13].

Easily comprehensible parents-children relations, clearly separated purpose of different geometrical elements, and manageable structure are some of the basic means to clearly describe a complex model to the user. This is important in case of a parametric template model intended to be further modified in edge cases by hand. Using parametric models of different levels of parametrization can be considered in a specific sense as an analogy for automation (because it allows to reduce repetitive tasks by creating automatically adjustable models) with more inputs limitations.

3.1. Implementation description

Power Copy is a tool in CATIA for duplicating information from source file to a destination in working file. Thanks to the associative and parametric nature, the result is automatically adjusted to selected inputs (geometry or parameters).

Power Copy preserves easy access to its constituent elements and its hierarchical structure. Once instantiated, a change in the source file is not propagated to already cloned geometry. This allows the templates to be defined for generic uses, but sometimes automatic change propagation might be desirable. The accessibility of Power Copy structure might be a limitation if unique names are required in specification tree (e.g. name-based data identification using CATIA automation API).

A custom Power Copy instantiation management tool can solve some limitations: change propagation (replacing instances), name uniqueness and one-at-a-time instantiation with GUI. The manual instantiation is a time consuming process if more copies have to be instantiated at specific, in-advance prepared, specification-driven positions.

Important consideration when creating templates is their robustness (e.g. the template has to be able generate a valid model for all orientations). Thus surface-based templates are preferred rather than solids, because surface-based models allow to easily reference geometric features (nodes in the specification tree) instead of using boundary representation. Surface-based models are better suited for incorporating design intent, functional requirements and their presentation, and retrieval of requirements.

3.2. Structure of Power Copy template

Some of the shortcomings of advanced Power Copy usage can be avoided by definition of a proper structure combined with limited processing of geometric inputs using standard commands, together with properly parametrized models.

It might be preferable to limit the number of designs defined in a single template file (fig. 7) to a small number or even create only a single template per file, so it remains manageable and is clearly presented. The basic structure of a template consists of

![Fig. 5. Interface surface with die direction line and derived class B surface](image)

![Fig. 6. Resulting part (2 variants) with 6 instances of parametric functional features, that have some common inputs (e.g. die direction) and some of their parameters can be individually modified (e.g. number and angle of ribs)](image)
The Power Copy definition dialogue, tree branches to be copied are highlighted.

three branches in specification tree: Geometry, Parameters and Relations.

The structure of Geometry branch separates various types of information (e.g. inputs, design intent or limits). An auxiliary geometrical set contains helper objects to specify design intent and "sanitize" inputs. In this set, all features that define the local axis system of instanced Power Copy and bounding box planes are defined. Naturally, the Geometry branch contains actual geometrical features used to create the final representation and the final output is specifically marked.

CATIA has two types of parameters: intrinsic and user parameters. Intrinsic parameters drive the elements and features of a parametric geometrical model. They are located in specification tree under geometry feature nodes. User parameters contain additional information that is needed for product development (e.g. requirements, design intent, etc.).

Parameters branch with user parameters serves to decrease the model apparent complexity and to present only relevant parameters that are often adjusted by designer in a central location. They are an immediate interface for Power Copy definition and they facilitate instance customization. Thus, internal (intrinsic) parameters are visibly separated and accessed only when necessary: correcting errors from unintended or untested use of template (i.e. finding limits of its use), or for development of derived templates. Derived information about model is stored as parameters. This allows the template to contain everything necessary for rough optimization, with both input and output values grouped in one place.

Relations branch aggregates all formulas that define relations between parameters or geometrical features and store specific know-how of template models [14].

Depending on the user parameters context, the formulas may contain external inputs (i.e. parameters and variables not derived from CAD model). These inputs may be implicitly stored in the formula body, or as named user parameters. While implicit variables and constants in formulas do not increase the model complexity with the introduction of additional links to parameters, this approach is not suitable for KBE. The implicitly stored knowledge in formulas might be difficult to be reused or adapted. Trade-offs between direct storage and generalised knowledge has to be considered.

3.3. Automation example

A simple script to facilitate the usage of Power Copies was written using procedural programming techniques (fig. 8). Because the Power Copy templates are identified solely on their node names in specification tree, the script presents the data in more easily readable manner and serves as a simple higher level management tool for Power Copies.

This approach was chosen for its less demanding structure and mainly because there was no need to create and store additional data during runtime. Although VBA is object-based language and supports object encapsulation and their interface, it does not support other properties of fully object-oriented programming languages. This is not particularly relevant for chosen paradigm. In this example, the procedural programming paradigm was sufficient, mainly because the data was not very complex. The management tool helps with repetitive tasks and the model is relatively simple (the execution speed is not critical, even if VBA has lower performance [15]) and the logic is mainly implemented in CAD file as a part of geometrical model (using CATIA KnowledgeWare module or generalized geometrical features). Part in fig. 6 shows its intended use.

4. Conclusion

The paper presents a scheme in which a final model is build from multiple intermediate CAD models in separate files. This ensures a clear data flow, where various kinds of inputs are handled in different models for different specialists. The separations of input models allows to create checkpoints in workflow and forces a clear identification of intermediate steps during design. A similar approach (although without separate CAD files) is used to build a parametrized, surface-based template model. This template contains not only the geometry, but in accordance to KBE principles, it also gives additional information. Its structure (e.g. inputs and outputs) is clearly presented and stored in selected CAD software. Further, the template includes parametrized knowledge in form suitable for CAD users.

The paper gives some hints of current problems in KBE as identified in literature. The proposed methodology addresses some shortcomings of KBE in practical way from perspective of CATIA user. A macro was written using procedural pro-
gramming technique to verify the usefulness of such an approach. The majority of template model logic was implemented using available CAD commands, macro programming served only as a simple management tool.

Future work will be to explore further the application of presented methodology. Possibilities of different ways how to embed knowledge into models will be evaluated, specially the cases where knowledge can be easily reused without the requirement of programming. A more robust tool (object-oriented) for template management can be developed.

Acknowledgements

This contribution has been elaborated under the European Structural Funds No. 26240220076 - “Industrial research into the methods and procedures in generative design and knowledge engineering in car development”; Program for supporting excellent young researchers at Slovak University of Technology, and by the Slovak VEGA grant agency in project VEGA 1/0445/15.

References