10th International Conference on Mass Customization and Personalization – Community of Europe (MCP - CE 2022)
Toward the Sustainable, User-Centric and Smart Industry 5.0
September 21-23, 2022, Novi Sad, Serbia

INTERACTIVE GEOMETRIC CONFIGURATION USING SKETCH-BASED CAD MODELS

Paul Christoph Gembarski, Pauline Gast
Institute of Product Development, Leibniz Universität Hannover,
An der Universität 1, 30823 Garbsen, Germany

Abstract: *Knowledge-based geometry models reduce variant design to the input of parameter values. Especially knowledge-based CAD models that incorporate geometrical data and implemented explicit knowledge offer additional possibilities. One is interactive drag-and-drop control of geometric features. This poses new requirements for the setup of CAD models as each geometric constraint, dimension and 3D feature contributes to the variability of the model. In this paper, the authors give methodological guidance to such modeling tasks by extending the CommonKADS approach with a correlation model for CAD model entities. The guidelines are visualized for the creation of an interactive, configurable steel construction model.*

Key Words: ***CAD Configurators, Knowledge-Based Design, Solution Space Development, Modeling Guidelines***

1. INTRODUCTION

Design configurators and knowledge-based geometry models largely reduce variant design activities to the input of parameter values [1, 2]. Common functionalities are template configurations as starting point for a convenient exploration of the solution space or consistency checks, e.g. if manufacturing or other constraints are violated [3, 4]. But especially knowledge-based computer aided design (CAD) models that incorporate a complete set of geometrical data, including parametrics and features, a chronology, behavioral rules and additionally implemented explicit knowledge, offer further input possibilities for geometry adaptation [5].

Virtual reality (VR) applications in mechanical engineering design use gestures to control and modify the CAD model in order to create an immersive and natural way of interaction [6, 7]. One of these interactions is interactive drag-and-drop control of geometric features, e.g., spanning an extrusion or moving parts in an assembly [8].

Basically, such functionalities for an intuitive way of model interaction can also be implemented for product configuration in non-VR applications. From a methodological point of view, such interactive configuration poses new requirements for the setup of a CAD model as each geometric constraint, each dimension and each 3D feature contributes to the variability of the model [9]. If, e.g., a feature references parameters from a suppressed parent component or a parameter is not well-defined between tested limits, the rebuild operation will probably fail [10].

Additionally, such a geometry change needs to be processed and proofed. One use case is that freely modified features are adjusted to a given increment of a dimension, another use case is that the change leads to an automated consistency check.

In the present paper, the authors give methodological guidance to such modeling tasks by extending the CommonKADS approach with a correlation model for CAD model entities. The guidelines are then visualized for the creation of an interactive, configurable steel construction model that is controlled by drag points.

2. THEORETICAL BACKGROUND

2.1. Knowledge-Based CAD

Compared to other 3D modeling principles, knowledge-based CAD is a paradigm shift as it leads the focus to the development of solution spaces rather than single product variants [2]. The basic idea is to implement domain and control knowledge leading to the ability to draw conclusions from the design situation and from requirement changes (Fig. 1) [11].

Thereby, domain knowledge is used to build the solution space. Modern CAD systems offer e.g. the following possibilities [12-15]:

- Constraints between model elements like parameters or features express, e.g., design formulas or logical dependencies. Especially when the CAD system is able to express physical parameters, complete dimensioning calculations may be implemented.
- Parameter tables describe part families, so e.g. standard parts can easily be built from a master

model which parameters are then fed from a row of the table.

- Features implement semantic information objects from contiguous geometry elements together with parametrics and behavioral rules. Within defined limits, features can adapt to changes of the modeling context.
- (Geometric) Templates aggregate multiple features and parametrics as reusable building blocks for virtual prototyping.
- Design rules are if-then-else-statements to control occurrences of model elements, trigger parameter calculations, value assignments or fire subordinate rules and program code.

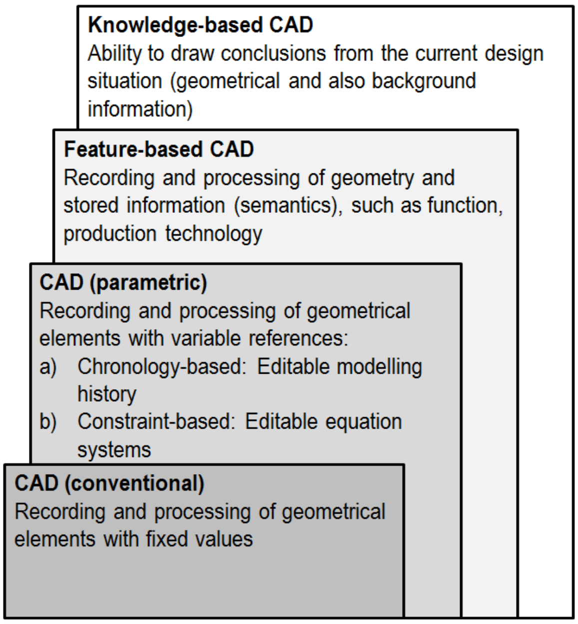

Fig. 1. *Overview of the principles of 3D modeling* [11]

In contrast to domain knowledge, control knowledge formalizes the way a solution space is explored. Three traditional reasoning types, known from expert systems, are also found in knowledge-based CAD [16, 17]:

- Rule-based reasoning uses a network of if-then-else-statements. The search process for the solution can be directional but also flexible since rules may initiate or exclude subordinate rules from further processing.
- Model-based reasoning, e.g. also in the context of product configuration, relies on abstractions of features or product components that are constrained to each other. The type of constraints may be, e.g., of logical or mathematical type, but also rely on mechanics like resource provision and consumption.
- Case-based reasoning widely mimics thinking in analogies, as problem-solving relies on previously solved and archived design tasks. A case is formally a problem-solution pair. Inference is done by searching for equal or similar solutions to a stated problem. Similarity in this context is modelled either by hierarchical concepts or similarity indices.

2.2. Methodologies for Building Knowledge-Based Engineering Systems

The existing methodologies for building knowledge-based engineering systems (KBES) set different foci to support the development of knowledge-based engineering systems in general. To those belong, among others, CommonKADS, MOKA, MIKE, KNOMAD and KAMET II [18-22]. While the last integrate their own specific tools for individual activities in KBES development, the first two were developed with wide applicability. As a consequence, CommonKADS and MOKA have reached dissemination. While the latter has a strong emphasis on knowledge engineering and informal modeling, the first draws strong attention on formal modeling. The CommonKADS method has been developed and validated by many universities and companies as part of the ESPRit Project P1098 KADS (Knowledge Acquisition and Documentation Structuring). This method offers the advantage that the development is discussed through the introduction of several stages of development [23]. CommonKADS is mainly applied as a tool for knowledge management in organizations, to support selected business processes and to develop configuration-based systems. The following CommonKADS models can be viewed as requirement specifications for these systems, being divided in the categories *context*, *concept* and *artefact* as shown in Fig. 2 [2].

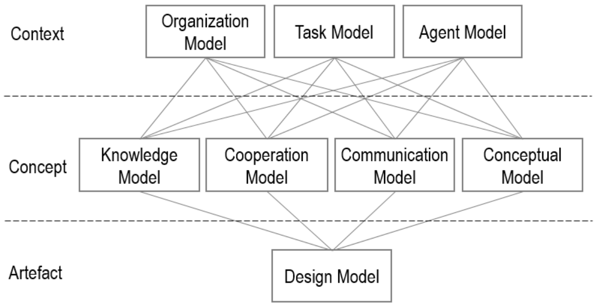

Fig. 2. *Categorization of the CommonKADS models*

Context includes the organization model, the task model and the agent model. These represent the task background, including analyzing the organization and the task itself, as well as the level of training of a potential user or the developer and functionalities of the system the KBES is supposed to be implemented in [19]. These models answer the question why such a system is needed and what benefits it brings [24].

General design concepts are collected in the second category, answering the question what kind of system is needed and pointing out its relations. These contain concepts of how knowledge components contribute to finding a solution, concepts for user interaction, the distribution of tasks between user and system, as well as the argumentation structure the KBES is supposed to run through [23]. This category includes the knowledge model, the communication model, the model of cooperation and the conceptual model.

The design model connects the preceded models and leads to the final design. Here the system is specialized in relation to the type of implementation platform, the form of representation and the calculation mechanisms for implementing the models of the categories context and concept. This model can be implemented without having to make further decisions on the implementation of system functions [25].

3. METHODICAL APPROACH

As knowledge-based CAD models consist of multiple geometric features and domain knowledge artefacts that are strongly intertwined, a special attention needs to be drawn on the control and rebuild concept of the model itself. Constrained, correlated and referenced parameters result in dependency chains that are to be kept flat. Known planning aids for this are parameter trees and parameter plans, in which dependencies of parameters are defined [26-28].

In interactive models, like mentioned in the introduction, a fast and stable model rebuild is a quality characteristic. Considering e.g. a system, in which drag points are implemented in a skeleton sketch of the CAD model, the dependencies between drag point, its driven parameters, user inputs and reasoning requires the introduction of a planning aid specialized on the resulting correlations. This tool has to support the designer to model correlations only top-down, avoid circular references and distinguish between different hierarchy levels. Additionally, it should visualize the length of a dependency chain in order to estimate model efficiency.

Since such a planning aid contributes to the concept level of the CommonKADS model architecture, the authors introduce the *correlation model* (Fig. 3). It is related to the models on context level since especially task and agent model pose requirements to the later user interaction with the knowledge-based CAD model which is then represented in the design model.

As shown in Fig. 4, the correlation model is divided into three layers. The *user input layer* contains all parameters that the user communicates to the system.

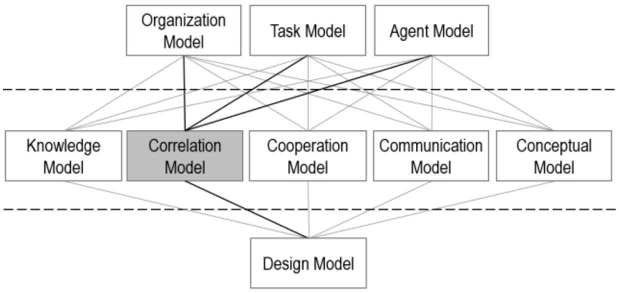

Fig. 3. *Extended CommonKADS with Correlation Model*

On the example of a steel platform, this is the desired dimensions, a maximum deflection and, resulting from the context / agent model, the input of the available raw stock that possibly restricts beam sizes for the support or the gratings.

As the interaction with e.g. drag points introduces a new geometric abstraction, the *skeleton layer* combines all respective constraints and visualizes the parameters that are influenced by drag points or other interaction elements.

The *component layer* then relates elements of the knowledge-based CAD model, e.g., geometric features in a part model or components in an assembly model, to the other layers.

As relations between the parameters, direct linkage is defined as parameter assignment. Additionally, calculations and restrictions are introduced. In order to realize a strict control of the later CAD model, two rules are further introduced: Only top-down correlations are allowed to get from one layer to another but correlations within a layer may be both unary and binary.

It is noticeable that correlations can be modeled as loop like between *Input of maximum deflection, grating: length* and *support: cross-section* in the depicted example. This shows that this relation can be met both by reducing the length of the platform or by increasing the strength of the supporting beams. Thus, loops also indicate potential for later optimization or finding alternatives.

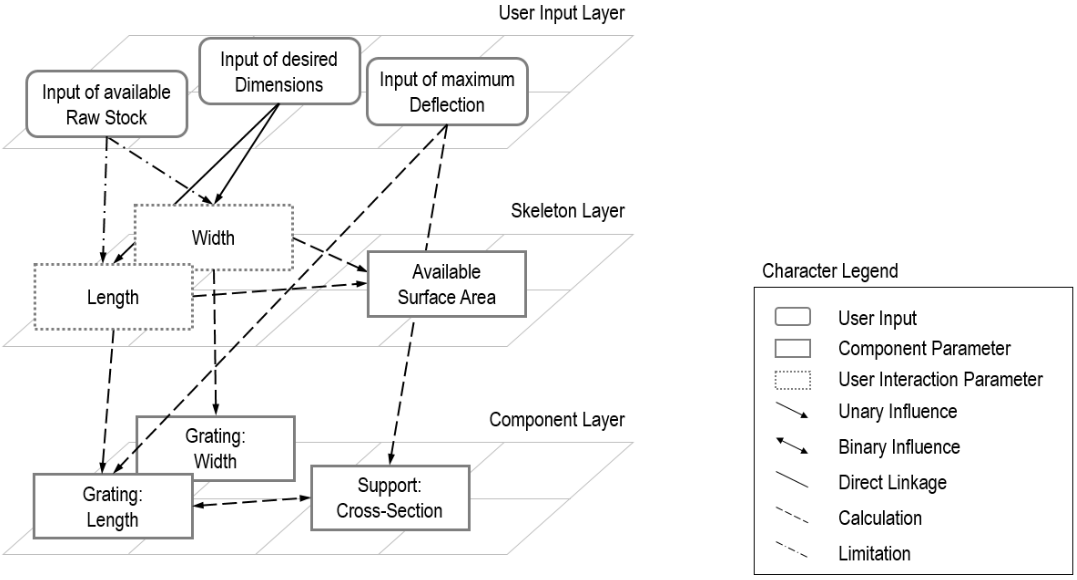

Fig. 4. *Correlation Model for a Steel Platform (Excerpt)*

Since the integration of the correlations is the most difficult part of planning the final system design, the correlation model should serve as the basis for the design model after completion.

4. APPLICATION EXAMPLE

Fig 5. *Configurable, modular crane pathway*

The application example is to implement a sketch-based dynamic configurator for crane pathways in Autodesk Inventor Professional (Fig. 5). Subtasks to be fulfilled are the placement of the components (straight and curve segments) via Drag&Drop, the calculation of the geometric pattern of supports (portals or cantilevers) along the track, the calculation of the resulting deflection based on the load specified by the user and, following this, warning the user if the maximum deflection is exceeded. If this occurs, the user should have the possibility either to reduce the distance between the supports or to change the cross-section of the whole pathway. Use cases of the system can be seen in Fig. 6.

The Correlation Model shown in Fig. 7 contains the referenced parameters for the patterns of the supports, which are calculated within the I-beams of the trail and called up by the pattern. The load information required to calculate the deflection is requested via user forms and saved as user input as well as the maximum deflection to be reached and the selection between portals and beams for the first pattern.

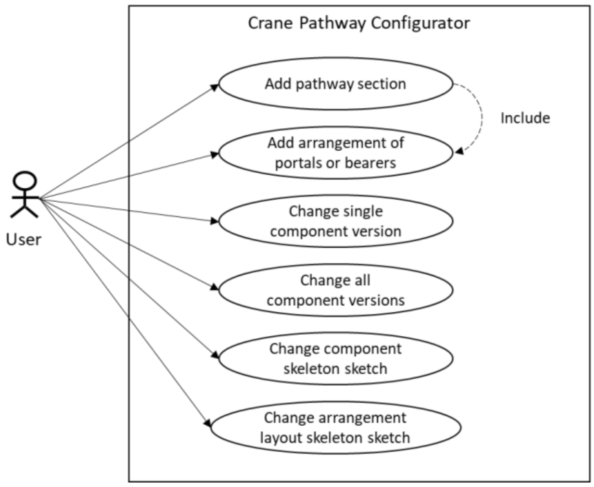

Fig. 6 *UseCase Diagram of the Configurator*

The specified maximum deflection is then saved as a limit value. In addition, the user specifies a size of the beams for which five variants are implemented in this example.

The user inputs are processed within the assembly by the other correlations. One of these is the deflection calculation with the layout of the pattern, as the distance of the supports is used in it. The calculation of the distances and the number of supports is carried out at the component level of the I-beam. Using the length of the I-beam and the approximate distance of the elements, both are specified by the user through drag points. Later, a rounded number of elements is to be calculated as well as an even distance. The layout is modeled as skeleton sketch and used as a reference by the pattern at assembly level.

To implement the knowledge base for this system, as well as the previously listed correlations, Autodesk Inventor Professional offers a variety of KBE modeling options. These include iParts (part families) and an Excel integration for the realization of e.g. template models, or

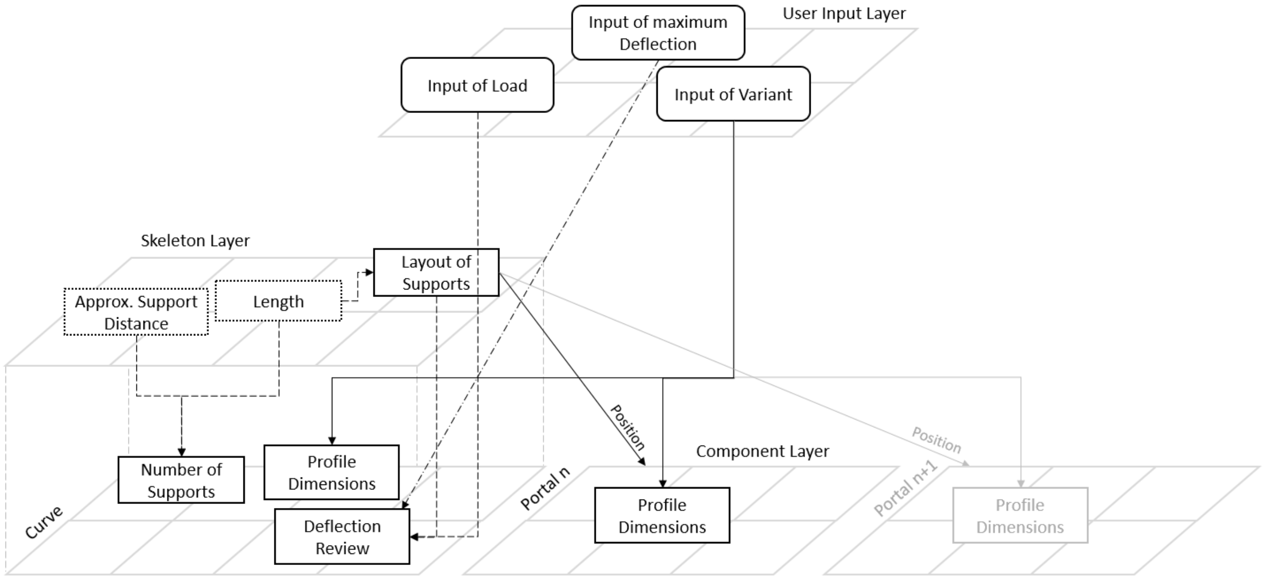

Fig. 7. *Correlation Model - Creation of a Pattern*

geometric and sketch-based patterns. For knowledge and rule integration, handling of user parameters and the export and import of parameters, the two programming environments iLogic and VBA as application programming interface (API) are available. Both also allow for designing user interfaces, though iLogic focuses more on parameter handling and has limited control elements.

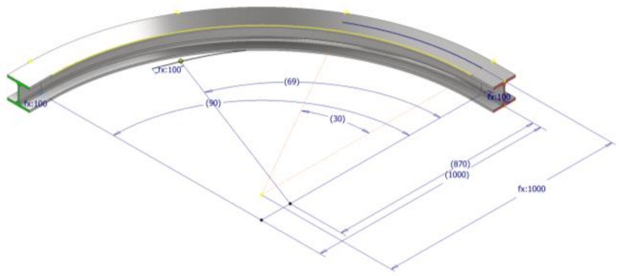

Fig 8. *Curve Segment with Interfaces and Drag Point*

The implementation is shown on the example of a generic curve segment (Fig. 8). The list of cross-sections according to DIN 1025-2 is part of the knowledge model, as well as the additional parameters that are necessary for the deflection calculation according to the corresponding substitute model (in this case beam supported on two sides). The geometry model needs to contain a representation of the cross-section (in this case a sketch that is fed from a parameter table) and a representation of the radius and angle of the curve path, which later will be controlled by a drag point (in this case a 2D sketch).

Building up the component one starts by defining the cross-section in the first sketch. It is preferable to construct symmetrically to the origin planes. Here the rough shape of the I-profile should be modeled first without the fillets, as those should only be added as 3D feature later to achieve better performance. When applying the dimensions to the sketch, the parameters are named and initialized according to the smallest parameter set of the standard. The next step is to create a second sketch which is arranged perpendicular to the plane of the first one. Here an arc is created beginning in the component origin, which is defined by center point, radius and trajectory angle which both are defined as driven dimensions. This creates the drag point which is located at the end of the arc.

The geometry is then created as sweeping since this allows better controllability compared to a revolution where the parameters from the drag point would need to be linked to the revolution angle, while the radius would be controlled from the sketch itself. Subsequently, the mentioned fillets are added to the profile. Another skeleton sketch is added onto the profile which is used for the pattern of supports. For this purpose another arc is created which contains the imported radius and center point of the curve, but a different angle. Here, too, the parameters are marked as driven dimensions.

Once the geometry has been completed, iMates are added to the component which are half geometric constraints. To connect to the contact components of the previous and following section, six iMates are defined in two sets. The three iMates of each set determine the flush constraint of the upper surfaces and the side surfaces, as well as the mate constraint of the contact cross-section surfaces. In addition six further iMates are defined for the pattern of the supports, these are combined into two sets for the connection on the right and left side of the curve. They define the flush constraint of the cross-sectional surface of the curve with the side surface of the left or right side surface of the element, the flush constraint of the left or right side surface of the curve with the front surface of the element as well as the mate constraint of the upper surface of the curve and the lower surface of the cantilever support. When defining iMates, however, only one half of these intended connections is defined for each component. They are brought together in the assembly by their naming.

After defining the iMates, an embedded spreadsheet is created that contains the different sizes from the standard. This is necessary to implement the automatic change of the variant of all components to another size in case that e.g. the deflection is too high and no additional supports are possible. In order to be able to access the embedded spreadsheet directly through iLogic, the name of the table must not be changed.

After integrating the parameter sets, the iLogic rule for the deflection calculation is created. Here, the modulus of elasticity E, the area moment of inertia I, the mechanical load F, as well as the distance between the supports or the full circumference of the curve l are processed. After the calculation of the deflection, it is compared with the user's specification of the maximum deflection. If it is exceeded, a message box is opened, on which the name of the component section which exceeds the deflection is specified and the control parameter for exceeding the deflection is set to true.

Another iLogic rule which is created within the component is the one for automatically changing the variant via the spreadsheet link. This is set up in such way that a control parameter is checked to see if the variant has to be changed. If so, the spreadsheet is opened in the background. By specifying the column and row, each parameter is updated using the command "GoExcel.CellValue".

Before creating an iPart from the component, all functions should be tested as iParts are write-protected. To do this, one should place the component in a new assembly and test, whether the skeleton sketches are visible and accessible, whether the component connects the iMates to new components and whether the iLogic rules are executed properly. Here, the control parameters, which are to trigger the rules automatically in the system later on, can be changed manually in the test assembly.

After testing all functions, an iPart can be created from the component to realize the manual change of the variant of a single component. The important parameters and user parameters are selected here and loaded into the iPart table.

The master assembly of a new pathway initially contains one component of each of the five basic components (straight pathway segment, right and left curve segments, cantilever and portal), which are inactive and invisible. These serve as a product model that is copied and saved seperately when a new component is added. This method of insertion bypasses

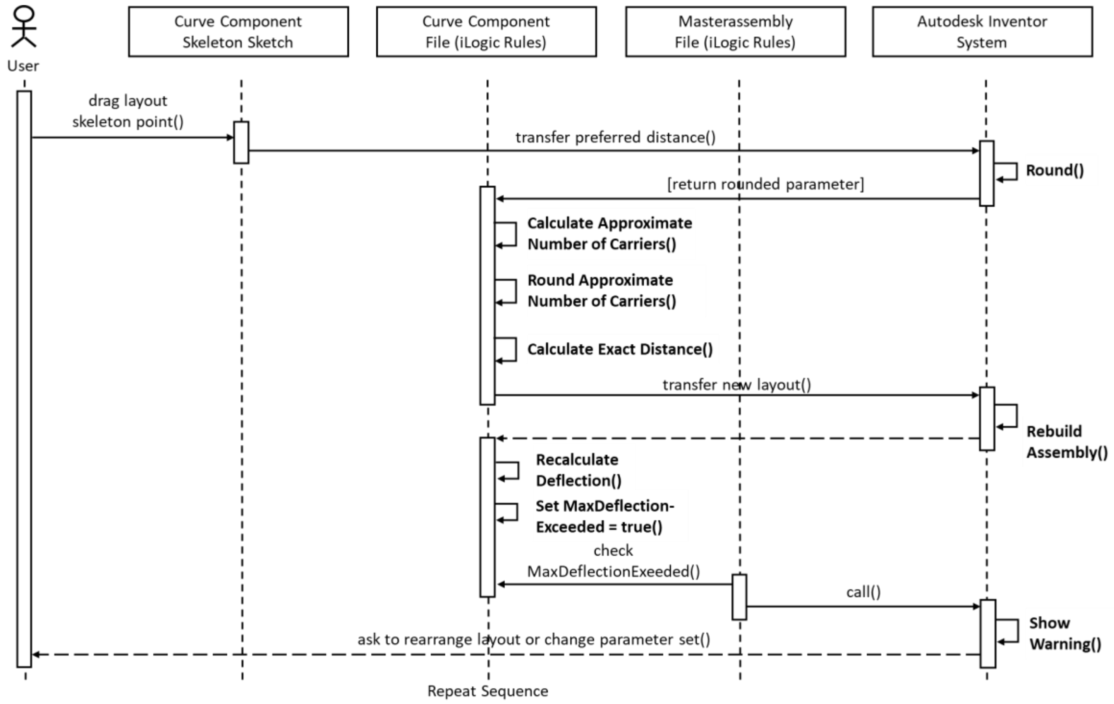

Fig. 9. *Sequence Diagram for User Modification of the Curve Segment Drag Point*

the write protection associated with the Autodesk Inventor iParts and allows the components to be edited independently.

When a user starts a new pathway design, the first step is to input the initial requirements and boundary conditions as specified on the user input level of the correlation model. Afterwards, a first straight segment is added to the assembly to which the user can add further segments by drag&drop. Due to the iMates, the new component is automatically constrained to the existing ones. By a second user form, the single segments can be equipped with supports. For this purpose, one component is placed at the section via iMates and, starting from this, a pattern is created according to the skeleton sketch. When inserting a new cantilever pattern, the user additionally specifies on which side they should be placed. The side is indicated in relation to the running direction, which is defined along the positive x-axis of the component.

The sequence diagram in Fig. 9 shows what happens when the user drags the drag point of the layout skeleton sketch in the curve segment. Here, the values for radius and trajectory angle is first rounded by the system and then an approximate number of supports is calculated based on the distance defined by the user. This is then rounded and the exact distance between the cantilevers or portals is adjusted. This new layout is then updated by the system. The deflection is recalculated and if exceeded, the corresponding Boolean parameter is set to true. This parameter is permanently checked by the master assembly and if the deflection is confirmed to have been exceeded, a warning is opened by the system automatically.

If no cantilevers or portals have been placed along a section, the entire length of the section is used for the deflection calculation instead of the distance between the supports according to the skeleton sketch.

If the user decides to change the component variant as a measure for falling below the maximum deflection, the components are called up individually and the corresponding parameters are updated from the variant list in the Excel table. The user has the possibility to change the component variant independently of the exceeding of the deflection. To do this, he or she can either use the function ‘Change Size’ for an individual component, which accesses the iPart tables stored in the component, or change all component variants in one step according to the previous procedure via user forms. If a single segment is changed, an adapter piece would be necessary to maintain the height of the pathway and geometric consistency. Such a function has not yet been included in the configuration system.

5. LESSONS LEARNED

When developing a sketch-based, interactive configuration, detailed models of the extended CommonKADS methodology should be completed before starting to model the components. Besides being a specification and modeling tool, it is also a checklist for the later implementation. Especially the conceptual model is to be seen a basis for the system’s information flow.

As in all parametric CAD models, attention should be paid to flat hierarchies and a central control flow. This ensures the correct update of all parameters for which a detailed *Correlation Modell* is essential, as the depth of the hierarchies as well as the direction of the control flows can be read from the model. In addition, a clear breaking down of the correlations supports the planning of embedded calculations and avoids loops which would lead to the necessity of multiple, sequential model rebuild operations in order to achieve a complete update.

During control planning, the interaction between the system and components has to be planned, as well as the influence the user is supposed to have, like being able to change sketches, occurrences or parameters. This influences the access rights the user and the system need to have, which in turn influences which type of component the parts of the assembly must be inserted as. The access rights associated with the different types of components also depend on the used CAD system. For example, access to sketches is generally blocked on inventor iParts, whereas control or user parameters and parameter sets can be changed easily. Another type of component, available in Autodesk Inventor is content center parts which are library iParts. These generally have similar advantages and disadvantages as user defined iParts if they are placed as according to standard. However, their placement from the content center is associated with a better user interface as the components inserted into the content center can be categorized in folders and appealing input dialog. If they are placed with the option “individual”, however, the write-protection can be bypassed and part of the functions associated with the iParts can be retained. Thus, although the iPart tables can be accessed manually via the “change size” function in the feature tree, they can no longer be accessed automatically via iLogic. Consequently, this way of implementation is eliminated.

As a rule of thumb, all sketches in a parametric CAD model need to be fully determined to optimize performance. This is not possible with sketches including drag points so that their use should be restricted only to the skeleton sketches. The geometry features then should be linked to this or rely on reference geometry for their boundaries.

6. SUMMARY AND CONCLUSION

The basic idea to use drag points as interactive geometry modifier for CAD-based configuration could be realized. As the organization of parameter dependencies and an errorless model rebuild after modification raised the need for an additional planning aid, the correlation model as CommonKADS extension proofed to be applicable. As shown with the deflection calculation, additional plausibility checks or the check for any type of restriction is still possible and a cooperative way of problem-solving involving both user and configuration system results.

The chosen application example is a placeholder for many assembly types in mechanical, electrical and civil engineering. Thinking of other steel constructions, piping, wiring harness or even circuit boards, drag points are able to convey a more natural configuration experience compared to the input of parameter values. Related to the application of virtual reality, users can import external reference like the image of a floor plan or a construction side, e.g., as configuration aid or layout.

Additional examinations are necessary in order to estimate the performance of more complex assemblies. Thinking of piping when supports not only consisting of three but thirty or more components are influenced, latencies when dragging the control elements need to be as low as in the above application example. A reliable approach here could be to use models with low level of detail for the interactive configuration and generate the models with full manufacturing bill of materials in a subsequent step.

Another implication of the drag points and interactive configuration in general is its potential for gamification of the configuration process. What about a game of roller-coaster tycoon?

7. REFERENCES

[1] A. Trentin, E. Perin, and C. Forza, “Product configurator impact on product quality,” *International Journal of Production Economics*, vol. 135, no. 2, pp. 850-859, 2012. DOI: 10.1016/j.ijpe.2011.10.023

[2] P.C. Gembarski, R. Lachmayer, “Solution Space Development: Conceptual Reflections and Development of the Parameter Space Matrix as Planning Tool for Geometry-based Solution Spaces,” International Journal of Industrial Engineering and Management (IJIEM), vol. 9, no. 4, pp. 177-186, 2018. DOI: 10.24867/IJIEM-2018-4-177

[3] J.J. Cox, “Product Templates - A Parametric Approach to Mass Customization” *CAD Tools and Algorithms for Product Design*. Berlin, Heidelberg, Germany: Springer, pp. 3-15, 2000. DOI: 10.1007/978-3-662-04123-9_1

[4] H. Li, P.C. Gembarski, R. Lachmayer, “Template-Based Design for Design Co-Creation” in Proceedings of the *5th International Conference on Design Creativity* (ICDC2018), Bath, United Kingdom, 31.01.-02.02.2018, 2018.

[5] M. Hirz, W. Dietrich, A. Gfrerrer, and J. Lang, *Integrated computer-aided design in automotive development*. Berlin, Heidelberg, Germany: Springer, 2013. DOI: 10.1007/978-3-642-11940-8

[6] R. Neugebauer, D. Weidlich, H. Zickner, and T. Polzin, “Virtual reality aided design of parts and assemblies,” *International Journal on Interactive Design and Manufacturing* (IJIDeM), vol. 1, no. 1, pp. 15-20, 2007. DOI: 10.1007/s12008-007-0008-2

[7] A. Seth, J.M. Vance, and J.H. Oliver, “Virtual reality for assembly methods prototyping: a review,” *Virtual reality*, vol. 15, no. 1, pp. 5-20, 2011. DOI: 10.1007/s10055-009-0153-y

[8] M. Fechter, and S. Wartzack, “Natural finger interaction for CAD assembly modeling,” in *37th Computers and Information in Engineering Conference*, 2017, p. V001T02A041. DOI: 10.1115/DETC2017-67555

[9] E. Ostrosi, J. Stjepandić, and Ž. Tomasić, “Robust CAD Modeling: Concepts and Principles for Industrial Applications,” in *Transdisciplinary Engineering for Complex Socio-technical Systems– Real-life Applications*, 2020, pp. 612-621. DOI: 10.3233/ATDE200122

[10] K. Amadori, M. Tarkian, J. Ölvander, and P. Krus, “Flexible and robust CAD models for design automation,” *Advanced Engineering Informatics*, vol.

26, no. 2, pp. 180-195, 2012. DOI: 10.1016/j.aei.2012.01.004

[11] VDI, *VDI Guideline 2209 - 3D Product Modeling*, Berlin, Germany: Beuth, 2009.

[12] VDI, *VDI Guideline 5610-2 - Knowledge management for engineering – Knowledge based engineering (KBE)*, Berlin, Germany: Beuth, 2017

[13] P.C. Gembarski, “Three Ways of Integrating Computer-Aided Design and Knowledge-Based Engineering.” *Proceedings of the Design Society: DESIGN Conference*, vol. 1, pp. 1255-1264, 2020, DOI: 10.1017/dsd.2020.313

[14] G. La Rocca, “Knowledge based engineering: Between AI and CAD. Review of a language based technology to support engineering design” *Advanced Engineering Informatics*, vol. 26, no. 2, pp. 159-179, 2012. DOI: 10.1016/j.aei.2012.02.002

[15] W. Skarka, “Application of MOKA methodology in generative model creation using CATIA.” *Engineering Applications of Artificial Intelligence*, vol. 20, no. 5, pp. 677-690, 2007. DOI: 10.1016/j.engappai.2006.11.019

[16] L. Hvam, N.H. Mortensen, and J. Riis, *Product Customization*, Berlin, Heidelberg, Germany: Springer, 2008. DOI: 10.1007/978-3-540-71449-1

[17] A.A. Hopgood, *Intelligent Systems for Engineers and Scientists: A Practical Guide to Artificial Intelligence*, Boca Raton, USA: CRC press, 2021. DOI: 10.1201/9781003226277

[18] M. Stokes, *Managing engineering knowledge: MOKA: methodology for knowledge based engineering applications*, London, Great Britain: Professional Engineering Publishing, 2001.

[19] G. Schreiber, B. Wielinga, R. de Hoog, H. Akkermans, W. Van de Velde, “CommonKADS: A comprehensive methodology for KBS development” *IEEE expert*, vol. 9, no. 6, pp. 28-37, 1994. DOI: 10.1109/64.363263

[20] R. Curran, W.J. Verhagen, M.J. Van Tooren, and T.H. van der Laan, “A multidisciplinary implementation methodology for knowledge based engineering: KNOMAD,” *Expert Systems with Applications*, vol. 37, no. 11, pp. 7336-7350, 2010. DOI: 10.1016/j.eswa.2010.04.027

[21] S. Neubert, “Model Construction in MIKE (Model Based and Incremental Knowledge Engineering),” in *Knowledge Acquisition for Knowledge-Based Systems*, Berlin, Heidelberg, Germany: Springer, 1993. DOI: 10.1007/3-540-57253-8_55

[22] O. Cairó, and S. Guardati, “The KAMET II methodology: Knowledge acquisition, knowledge modeling and knowledge generation,” *Expert Systems with Applications*, vol. 39, no. 9, pp. 8108-8114, 2012. DOI: 10.1016/j.eswa.2012.01.155

[23] A.T. Schreiber, G. Schreiber, H. Akkermans, A. Anjewierden, N. Shadbolt, R. de Hoog, W. Van de Velde, B. Wielinga, and N.R. Shadbolt. *Knowledge engineering and management: the CommonKADS methodology*, Cambridge, USA: MIT press, 2000.

[24] N. R. Milton, *Knowledge Technologies*, Milano, Italy: Polimetrica sas, 2008

[25] J.K. Kingston, “Designing knowledge based systems: the CommonKADS design model,” *Knowledge-Based Systems*, vol. 11, no. 5-6, pp. 311-319, 1998. DOI: 10.1016/S0950-7051(98)00071-9

[26] J. Abulawi, *Ansatz zur Beherrschung der Komplexität von vernetzten 3D-CAD-Modellen*, PhD-Thesis, Universität der Bundeswehr Hamburg, 2012.

[27] H. Li, T. Brockmöller, P.C. Gembarski, and R. Lachmayer, “An investigation of a generative parametric design approach for a robust solution development,” *Proceedings of the Design Society: DESIGN Conference*, vol. 1, pp. 315-324, 2020, DOI: 10.1017/dsd.2020.273

[28] P.C. Gembarski, and R. Lachmayer, “Forward variance planning and modeling of multi-variant products”. *Procedia CIRP*, vol. 21, pp. 81-86, 2014. DOI: 10.1016/j.procir.2014.03.161

CORRESPONDENCE

Dr.-Ing. Paul Christoph Gembarski
Institute of Product Development,
Leibniz Universität Hannover,
An der Universität 1
30823 Garbsen, Germany
gembarski@ipeg.uni-hannover.de

Pauline Gast
Institute of Product Development,
Leibniz Universität Hannover,
An der Universität 1
30823 Garbsen, Germany
pauline.gast@stud.uni-hannover.de