History in Motion: Interactive 3D Animated Visualizations for Understanding and Exploring the Modeling History of 3D CAD Designs

TOM VEUSKENS, Hasselt University - Flanders Make
Expertise Centre for Digital Media, Belgium
RAF RAMAKERS, Hasselt University - Flanders Make
Expertise Centre for Digital Media, Belgium
DANNY LEEN, Hasselt University - Flanders Make
Expertise Centre for Digital Media, Belgium
KRIS LUYTEN, Hasselt University - Flanders Make
Expertise Centre for Digital Media, Belgium

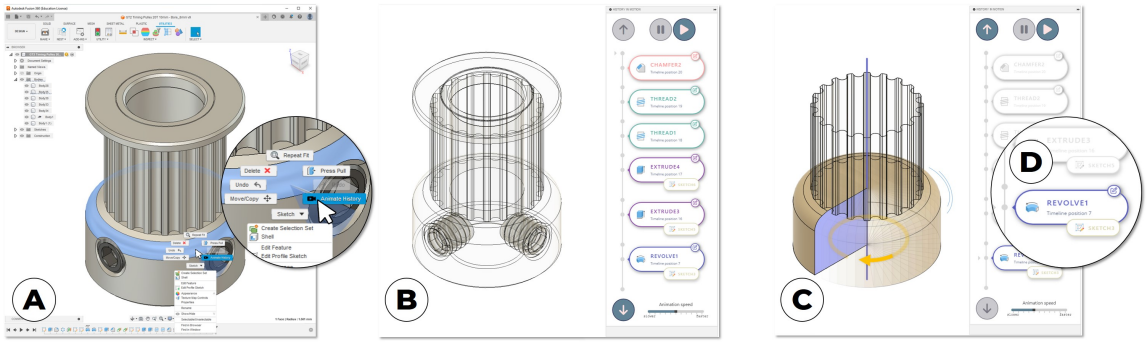

Fig. 1. An overview of History in Motion (HiM): (a) The designer selects a 3D geometry element. (b-c) HiM extracts the modeling features and an interactive animated visualization demonstrates how the modeling features contribute and interact to realize the selected geometry element. (d) HiM's control panel provides shortcuts for modifying modeling features.

We present History in Motion (*HiM*), an interactive visualization tool that enables CAD designers to interactively explore the design history of 3D CAD models. In contrast to manually exploring the modeling history of a CAD project, HiM finds relevant modeling features for geometry elements selected by the designer. We contribute a novel 3D interactive animation that visualizes how the modeling features interact, and are used on top of the CAD model, to realize the geometry. A control panel allows for a deeper exploration of the modeling features, with shortcuts for making modifications.

CCS Concepts: • **Human-centered computing → Human computer interaction (HCI)**; **Interactive systems and tools**; • **Computing methodologies → Animation**; • **Applied computing → Computer-aided design**.

Additional Key Words and Phrases: CAD, modeling history, animations, 3D

Manuscript submitted to ACM

ACM Reference Format:
Tom Veuskens, Raf Ramakers, Danny Leen, and Kris Luyten. 2023. History in Motion: Interactive 3D Animated Visualizations for Understanding and Exploring the Modeling History of 3D CAD Designs. In *Symposium on Computational Fabrication (SCF '23), October 8–10, 2023, New York City, NY, USA.* ACM, New York, NY, USA, 19 pages. https://doi.org/10.1145/3623263.3623358

1 INTRODUCTION

Feature-based Computer-Aided Design (CAD) modeling has become the industrial standard for designing engineering models and is widely adopted by major CAD software tools, such as Autodesk Fusion 360, Autodesk Inventor, PTC CREO, CATIA, or SolidWorks [Camba et al. 2016]. In this modeling paradigm, parametric CAD models are realized through a series of modeling operations or features, often referred to as the modeling history of a 3D model. Modeling features embed parameters and link to geometry elements of the 3D model, such as profiles and edges. Examples include an extrude feature in which a 2D profile is turned into a 3D geometry by extruding it with a parametrized value along its normal. Alternatively, a fillet or chamfer operation adds respectively a rounding or bevel parameter to an edge of the 3D geometry. Making changes to parameters or profiles requires recomputing all modeling features that build upon them to render the changes in the CAD model. As such, in feature-based modeling, a minor modification can result in significant changes and the re-rendering of the entire model.

Reusing and adapting existing 3D CAD models is a common practice to save time and leverage prior engineering efforts during design processes [Andrews et al. 1999]. Reusing models often requires making several changes to meet new design requirements [Altmeyer et al. 1994; Camba et al. 2016]. While some changes in feature-based CAD models can be as simple as changing the parameters of the modeling features, many updates are often challenging for the following reasons: (a) Many models consist of hundreds of features, and one has to find the appropriate parameters to change and understand all implications stemming from these changes. (b) The majority of 3D models do not allow for changes in parameters as it requires significant investments in time to make models robust [Peng 2012]. (c) Many changes go beyond updating parameter values and require altering, replacing, or appending feature definitions, which requires an in-depth understanding of how the model was designed. Making changes to geometries in feature-based CAD modeling environments, for example, is a very challenging endeavor when designers are unfamiliar with the modeling history of the CAD design. Even for CAD models one authored recently, getting familiar with the modeling history again is necessary before being able to implement further changes. Hence, designers often turn to push/pull operations for realizing minor edits quickly. However, these operations are less robust to changes in the future, as they are not defined by modeling features [Fu et al. 2017]. When major changes are necessary, designers often prefer to redesign a model from scratch instead of investing time and effort in understanding and making adaptions to existing models [Busby 1999].

To facilitate understanding the modeling history of CAD models in feature-based CAD modeling environments, we present *History in Motion (HiM).* HiM visualizes modeling features that contribute to a selected geometry element, such as faces or edges, and offers interactive exploration of these modeling features. Upon selection of a 3D geometry element, such as the curved face in Figure 1a, HiM renders an animated visualization of the sequence of modeling features that were used to realize the selected geometry element (Figure 1b). This animation shows, for example, that the curved face is defined in a sketch and not the result of a fillet feature applied to the edge, as one might have guessed at first sight (Figure 1c). For every modeling feature in the animated visualization, HiM also offers a shortcut for editing the feature. As shown in Figure 1d, a shortcut to the sketch is provided for editing the curvature of the face.

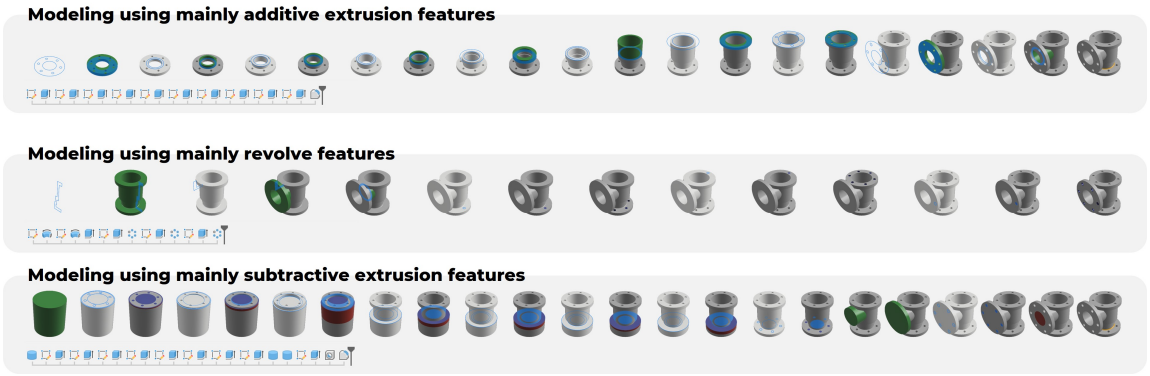

Fig. 2. Three very different modeling histories in feature-based CAD modeling for realizing the exact same CAD model.

This paper contributes *History In Motion (HiM)*, a new interactive visualization tool to ease understanding and exploring the modeling histories of feature-based CAD models. To realize this, we contribute novel animated visualizations for explaining modeling features in CAD environments. While our demonstrator system is implemented as a plugin for Autodesk Fusion 360, the interactive visualization technique, as well as the algorithms we contribute go beyond the specifics of our proof-of-concept implementation.

2 MOTIVATION

In feature-based CAD tools, 3D models are realized by sequentially applying modeling features to 2D sketches or 3D geometries. Different sequences of modeling features can, however, result in identical geometries. Figure 2 shows three very different modeling histories for realizing the exact same design. While in the first approach, the object is modeled mainly by extruding shapes on top of each other, the second approach mainly uses revolve features, and in the third approach, extruded shapes are subtracted from a larger cylinder shape. Hence, it is impossible to understand how a 3D model was designed by simply looking at the final shape. Furthermore, to make changes to a model, one first needs to understand the modeling history to find the appropriate modeling features to modify.

To visualize the modeling features that constitute a 3D model, commercially available feature-based CAD tools simply represent the modeling history of the entire project as a long list of modeling features. In Autodesk Fusion 360, this modeling history is visualized horizontally as a timeline, whereas Autodesk Inventor and SolidWorks render this list vertically and add auto-generated names to modeling features as identifiers, such as "Fillet 1". Selecting or hovering these features in this list traditionally highlights the 3D geometry change introduced by the feature. A double tap often opens the appropriate dialogs to edit the modeling feature.

In contrast to visualizing these geometry changes of modeling features, finding the modeling features that contributed to a geometry element is less trivial. 3D models of even moderate complexity typically consist of hundreds of modeling features. Going through this entire list to find a specific modeling feature is time-consuming. This is especially true as new modeling features are always appended to the end of the modeling history. As a result, modeling features related to the same geometry element are not necessarily positioned together in the list and might even be scattered across the modeling history. Figure 3 shows two modeling histories for realizing the same CAD model. We visualized the modeling histories similarly to the visualization style in Fusion 360, but manually edited colored bars on top to visualize the geometry elements they contribute to. Although both modeling histories use highly similar modeling features,

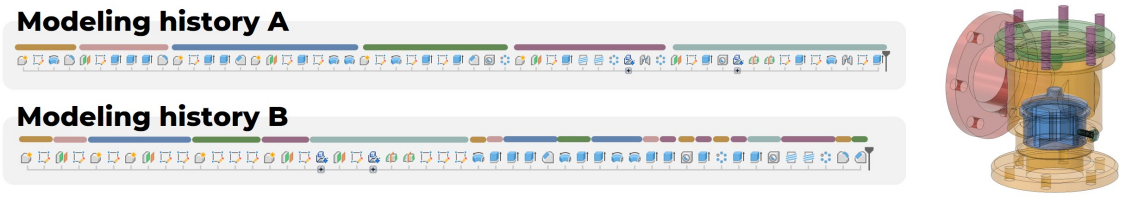

Fig. 3. Modeling histories consisting of the same modeling features, can look very different depending on the order in which modeling features are applied. Modeling history A is easier to process compared to B, as features related to the same geometry element are created after each other. This figure visualizes the modeling histories similar to the visualization style in Fusion 360, but we manually edited colored bars on top to visualize the geometry elements they contribute to.

modeling history B is significantly harder to understand as modeling features, related to the same geometry element, are scattered across the list.

To ease finding modeling features that created specific geometry elements, commercially available feature-based CAD tools, such as recent versions of Autodesk Fusion 360, allow designers to right-click on a face and initiate the "edit feature" or "edit profile sketch" function. These functions aim to find and edit respectively the feature or sketch that realized the geometry. However, in the majority of situations, the geometry of a face is the result of several modeling features and thus cannot be represented by a single feature. Figure 4, for example, shows a cube with rounded edges that is realized, according to modeling history A, by extruding a sketch and applying fillet features to the edges. Changing the dimensions of the side panel, highlighted in Figure 4, requires editing different features depending on the desired change. To change the height of the face, one must edit the extrude feature, while the width of the face is altered via the sketch of the ground plane. The radii of the corners are in turn updated via the parameters of the fillet feature. If, however, the cube was constructed, starting from a sketch of the side panel, as shown in modeling history B of Figure 4, one simply turns to the sketch to edit either the width, height, or corner radius of the face. Hence, to trigger either "edit feature" or "edit profile sketch" in Autodesk Fusion 360, one first needs to understand modeling history. Furthermore, when multiple features contribute to a selected face, such as the extrude and fillet feature in the example above, state-of-the-art CAD tools, such as Fusion 360, only point the designer to the first feature that created the geometry element, thereby ignoring subsequent modeling features that further edited the geometry element. Other relevant modeling features can thus only be found by manually exploring the modeling history of the entire 3D model. We argue that multiple modeling features, as well as their interactions, are relevant for understanding and editing geometries.

Instead of manually processing a modeling history to find relevant modeling features for editing a specific geometry element, or alternatively, instead of pointing only to a single modeling feature, we contribute HiM. HiM is a new interactive tool that visualizes how multiple relevant modeling features contribute to a selected geometry element.

3 RELATED WORK

This work draws from, and builds upon, work related to CAD modeling and model reuse as well as visualizing workflows.

3.1 CAD Modeling and Model Reuse

While feature-based CAD tools are highly powerful as they allow for designing almost any engineering model, they are one of the most complex software tools to use, requiring extensive training [Kasik et al. 2005]. Despite their origin in

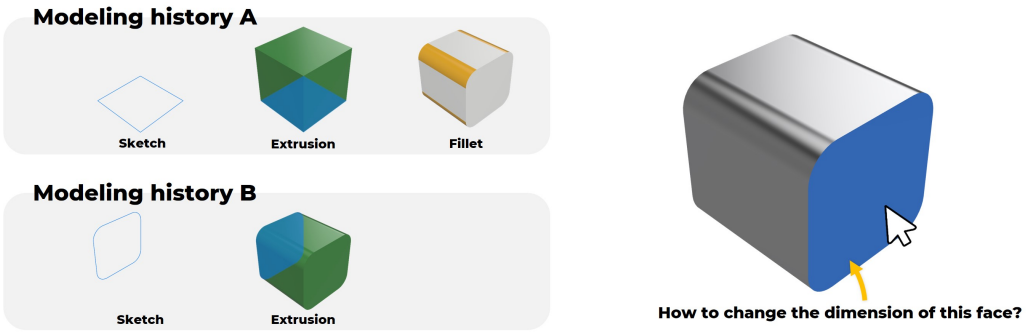

Fig. 4. Modifying the dimensions of the selected face of the cube requires edits to very different features depending on the modeling history. In Modeling history A, the width of the face is altered via the sketch, whereas the height is edited via the extrusion feature. The radii of the corners are updated via the parameters of the fillet feature. In modeling history B, however, all three parameters can be changed via the sketch.

industry, these tools are also highly popular in the maker community as CAD models are essential for prototyping using digital fabrication machinery. The maker community however consists of a diverse group of enthusiasts, who often do not have a formal background in engineering [Kiani et al. 2019]. The lack of 3D modeling skills has even been identified as a primary hindrance to the adoption of personal fabrication equipment [Shewbridge et al. 2014].

To lower the barrier for CAD modeling, various solutions have been proposed. GamiCAD [Li et al. 2012] and CADament [Li et al. 2014] employ gamification techniques to introduce users to new features in the design environment. Similarly, Blocks-to-CAD [Lafreniere and Grossman 2018] transitions users from a familiar 3D sandbox game environment to a CAD environment, allowing them to leverage knowledge in the game for CAD modeling. Masson et al. [Masson et al. 2022] observed that learning CAD modeling often involves a trial-and-error process, and augmented a CAD tool to support users throughout this phase. Chateau [Igarashi and Hughes 2001], on the other hand, suggests relevant CAD modeling operations during a design workflow to speed up modeling and introduce new modeling features to users. In an alternative track of research, researchers explore the use of physical blocks to design objects [Follmer et al. 2010; Follmer and Ishii 2012; Leen et al. 2017]. These physical constructions are then digitized into computer models [Ramakers et al. 2023]. Although these approaches leverage users' innate skills to manipulate physical objects, the fidelity of the resulting models is often significantly lower compared to CAD modeling.

As CAD modeling is time-consuming and hard, the reuse of existing designs is a widely adopted practice in CAD modeling workflows [Jackson and Buxton 2007]. Realizing high-quality parametric models that are robust to changes, however, requires expertise and additional investments in time. Research shows that the characteristics of CAD modeling workflows significantly affect their reuse later [Camba et al. 2016]. Hence, Peng et al. observed that models created by novice designers are oftentimes hard to modify and repurpose [Peng et al. 2012]. Additionally, experienced designers more frequently turn to their prior models, as they have a larger set of existing models [Ahmed et al. 2003].

Although designing reusable CAD models is hard for novices, it has important benefits as prior expertise and effort can be leveraged. This is also demonstrated by the popularity of maker community platforms for sharing 3D models, such as Thingiverse[1]. Models available via such platforms are combined, modified, or build upon to realize new designs, an approach referred to as remixing [Friesike et al. 2019; Oehlberg et al. 2015; Roumen et al. 2018]. Research however

[1]https://www.thingiverse.com/

shows that while the reuse of existing models is popular in the maker community, only a few users, with sufficient expertise in CAD modeling, contribute such high-quality parametric models to Thingiverse [Flath et al. 2017]. To lower the barrier to making high-quality parametric models several approaches have been investigated recently [Veuskens et al. 2022]. CODA [Veuskens et al. 2021], for example, analyzes CAD models and suggests constraints and parametric relations during a CAD modeling workflow. Such constraints make CAD models reusable and robust to changes. Instead of dealing with low-level geometry constraints, Hofmann et al. [Hofmann et al. 2018] propose PARTs which allows designers to visually specify high-level parametric behavior.

While these existing approaches lower the barrier for novice designers to start with CAD modeling and realize more high-quality models, HiM offers a distinct advantage. It empowers users to understand the modeling history used to create existing CAD models. This is an area that is often overlooked, but highly important. Modifying or building on top of existing models requires a good understanding of the modeling features that drive the model.

Besides feature-based CAD modeling, another popular modeling approach to create 3D models is Constructive Solid Geometry (CSG). In CSG, models are algorithmically defined in terms of geometric primitives and boolean operations [Voelcker and Requicha 1977]. CSG models are especially useful as recent research has explored how to go from an hard-to-edit mesh model to an editable CSG model [Du et al. 2018; Nandi et al. 2018]. However, automatically converted CSG models are often still hard to edit as they lack structure that designers typically add to the algorithm. To overcome this, Nandi et al. [Nandi et al. 2020] propose an automated approach to restructure CSG models to make them easier to edit. When reusing models created using CSG, designers need to understand the program defining the model before making changes. While the current version of HiM does not provide support for CSG models, similar tools could be developed in the future that extract relevant parts of the CSG model and animates how the primitives and boolean operations work together to realize parts of the 3D model.

3.2 Visualizing workflows

Researchers also extensively investigated techniques for capturing and visualizing workflows in software design tools. Chronicle [Grossman et al. 2010] is a capturing and analysis tool implemented in Paint.NET which performs a screen recording and also records all design operations while editing images. Users can use such recordings to learn how specific visual effects are realized. Similar to HiM, Chronicle allows users to select geometry regions and retrieve all design operations applied to that region. Instead of implementing workflow capturing features in specific tools, Nakamura and Igarashi [Nakamura and Igarashi 2008] present an approach for recording interactions with applications implemented in Java using Awt/Swing. To properly analyze extensive workflows, Chen et al. [Chen et al. 2016] present the idea of an adaptive history in which recorded editing operations are clustered at various granularities. To analyze and compare design workflows in more detail, Delta [Kong et al. 2012] offers a tool that visualizes differences and trade-offs between similar workflows. Such approaches help users in selecting and learning the proper workflow for making design edits.

In the area of 3D modeling, systems for recording and analyzing design workflows mainly focus on mesh-based modeling. Unlike feature-based modeling, in which modeling features drive geometry changes, in mesh-based modeling, geometries are manipulated directly via edges and vertices. Mesh-based modeling tools, therefore, do not necessarily preserve the modeling workflow while designing, as manipulations are directly applied to the CAD model. 3D Timeline [Doboš et al. 2014], therefore, is an approach for reverse engineering the modeling history from a sequence of intermediate versions of the 3D mesh model. 3D Timeline also allows for interactive exploration of geometries throughout the editing history. MeshFlow [Denning et al. 2011] presents a technique for rapid exploration of mesh

modeling workflows by filtering and clustering operations. 3DFlow [Denning et al. 2015] builds upon this work and allows for filtering operations by selecting 3D geometry elements. HiM, similarly, allows for analyzing modeling operations by selecting geometry elements. As different modeling strategies can be used for realizing similar results, CrossComp [Denning and Pellacini 2014] visualizes similarities and differences in modeling workflows of multiple designers. MeshHisto [Salvati et al. 2015] extends these approaches and visualizes mesh modeling workflows created by multiple designers, working together on a single model. HiM is very different from the approaches discussed in this paragraph, as it is implemented for feature-based modeling instead of mesh modeling. While mesh modeling, such as spline modeling, is popular for character design in, for example, games and animations, feature-based modeling is the industrial standard for modeling in engineering. For feature-based modeling, Workflow Graphs [Chang et al. 2020] proposes a data structure for representing differences in 3D modeling workflows that realize the same model. Unlike our approach, Workflow Graphs does not yet offer an interface for interactive exploration of modeling workflows.

4 HISTORY IN MOTION DESIGN RATIONALE

HiM helps designers in gaining insights into how specific geometry elements of existing 3D models are designed in feature-based CAD modeling environments. Upon selection of a geometry element, HiM opens an additional 3D interactive window that renders an animated visualization, explaining how a selected geometry element was realized. According to the user's preferences, this additional window can (partially) overlay the original CAD modeling application or have a permanent presence using a split view or external monitor when inspecting geometries frequently. HiM consists of 3 essential elements for which the design rationale is explained in this section: (a) Filtering modeling features upon selection of geometry elements, (b) 3D animated interactive visualizations, (c) Interactivity and control.

4.1 Filtering Modeling Features

Instead of requiring users to manually explore the modeling history of a CAD model to understand how geometries are realized, HiM filters all relevant modeling features when a geometry element is selected, including faces, edges, and vertices. When selecting, for example, a face, HiM extracts the feature that established the face, such as the extrusion or revolve of a sketch or duplication of an existing face, as well as all modeling features that further manipulate the shape or position of the face, such as extrusions that remove parts of the original shape or features that add curved or slanted edges. Doing so, HiM retrieves all modeling features in the modeling history until a sketch is found on which further modeling features build.

We believe that selecting a geometry element is a better "expressive match" [Olsen 2007] for finding the appropriate modeling features for editing a geometry element, in contrast to exploring the full modeling history of a model, in which modeling features are sorted based on their time of creation. In Figure 5, for example, HiM extracts a circular pattern feature, two thread features, and an extrude feature after the designer selected one of the threaded rods. The set of these features fully defines how the selected geometry element was realized. The circular pattern feature, furthermore, informs the designer that all threaded rods in the model are copies of the same design. Changes to modeling features of the rod are thus reflected in all rods unless the pattern feature is also modified.

4.2 3D Animated Interactive Visualizations

In contrast to state-of-the-art approaches that visualize modeling features using abstract iconic representations as shown in Figure 3, HiM visualizes the filtered set of modeling features by rendering 3D animated visualizations. These visualizations are very different from iconic representations as HiM renders the effect of the modeling feature onto the

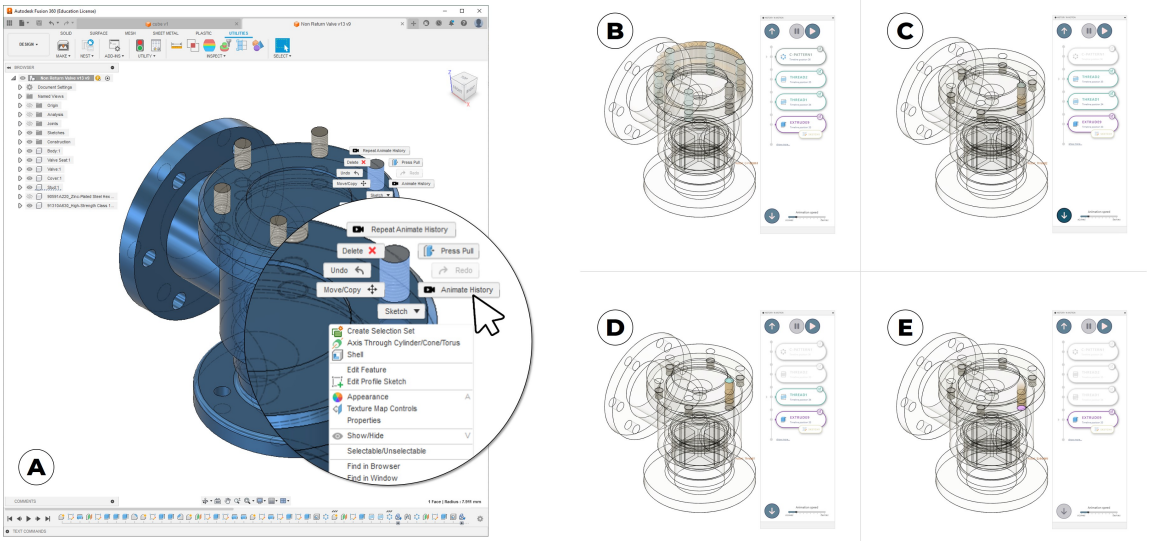

Fig. 5. (a) HiM extracts relevant modeling features selecting one of the threaded rods. An animated 3D visualization renders how the modeling features contribute and interact to realize the selected threaded rod: (b) A pattern feature duplicates a threaded rod design along a circle, (c-d) the threads are added through two thread features, (e) the rods originate from extruding a sketch of a circle.

3D model, thereby explaining the effect of the modeling feature in the context of the model. To visualize the effect of a modeling feature, HiM renders an animation that gradually transitions the 3D model from its state before the feature was applied until after its application. We designed these animations to look highly similar to the visual cues that are rendered by CAD tools when applying the modeling feature. For sporadic and frequent users of feature-based CAD tools, our animations are, therefore, immediately recognizable and clear depictions of the modeling features. We concatenate the animated visualizations of every modeling feature in the filtered set to render a short animated clip, visualizing how the selected geometry element is realized. As this animation is based on a filtered set of modeling features, it also shows how these modeling features interact to realize the selected geometry element. We also decided to play all animations of the filtered modeling history in reverse order, starting with the modeling feature that was added last. This reverse order improves recognition and facilitates understanding, as it ensures that the clip starts with exactly the same representation of the model as the one he/she is currently interacting with.

Our animated visualizations are different from an exact record and replay of a design workflow [Grossman et al. 2010] in two ways: (1) While we preserve the order of modeling features, our animations visualize only the modeling features that are relevant for the selected geometry element. These features might be scattered across a modeling history, as shown in Figure 3. (2) The rendering of custom animations in an interactive 3D environment allows designers to change the viewing perspective. Designers can thus change their viewport before, during, or after the animation to improve their understanding of the modeling features.

4.3 Interactivity and control

As our 3D animated visualizations are rendered in real-time, they support interactivity, such as changing the viewport. Additionally, HiM offers a neatly organized control panel (Figure 5b-e), offering an overview of the filtered set of

modeling features as well as controls over the 3D animated visualization. At any moment during the animated rendering, the modeling feature that is currently visualized is highlighted in the control panel. While our animations are instantly rendered upon selection of a 3D geometry element, the control panel allows for pausing, replaying, and changing the playback speed. Furthermore, users can step forward and backward through the animation to more closely inspect the impact of modeling features. Stepping backward through the filtered set of modeling features reverses the playback of the animation. Every modeling feature in the control panel also offers a shortcut to edit the feature. As such, designers can directly make adjustments to modeling features once they understand their impact on the geometry of the model without having to navigate manually to the modeling feature.

As explained in the previous sections, the filtered set of modeling features consists of all modeling features in the modeling history that contribute to the geometry element until a sketch is encountered that defines the basis for the geometry element. Such sketches, however, are often defined on faces. In turn, understanding how these faces are established can further help designers. Therefore, our control panel offers a "show more" feature which finds an additional set of modeling features that contributed to the face on which the sketch is built. This additional set of modeling features and corresponding animated 3D visualizations are then appended to the control panel. Designers keep on using the "show more" feature, to further explore the modeling history in more depth, until a sketch is found that is defined on a ground plane instead of a face.

5 BENEFITS AND EXAMPLE USES

In this section, we highlight several example cases that demonstrate HiM's utility in addressing various challenges during CAD modeling workflows.

(1) **Understanding interactions between modeling features**: As visualized in Figure 2, various design workflows can result in CAD models that look exactly the same. Designers, therefore, need to understand the modeling history before implementing changes. This is especially true when designers require changes to geometry elements that are the result of interactions between many modeling features. Figure 6a, for example, highlights a face of which the dimensions are not explicitly defined. Instead, its dimensions are the result of several modeling features, such as fillets, chamfers, and extrusions. When selecting this face, HiM visualizes the different geometry features that realized this face and established its dimensions. Furthermore, HiM's control panel can be used to more closely investigate which modeling feature contributed to which dimension and offers shortcuts for editing those features.
(2) **Understanding interactions between geometry elements**: In feature-based CAD, geometry elements are frequently realized by duplicating existing geometry using modeling features, such as copy and mirror, or duplication patterns, such as circular or rectangular. Changes to the original geometry are thus reflected in duplicated geometry elements, which can lead to unexpected changes if one is not aware of these features. As HiM filters the modeling history when selecting a geometry element, we draw designers' attention to modeling features that duplicate geometry. In Figure 6b, for example, one would like to change the diameter of the highlighted hole. Through HiM, the designer immediately notices that a mirror feature was used, and any change to the hole on the right will also be reflected on the left side of the model. The designer can now make an informed decision on whether this is desired or whether the model should be further adapted to remove the mirror feature.

Fig. 6. Some examples demonstrating the utility of HiM: (a) the dimensions of the selected face are not explicitly defined and are the result of several modeling features filtered and animated by HiM. (b) HiM informs the designer that the selected face is the result of a mirror feature. Changes to this face are thus also reflected in other faces. (c) HiM can help in understanding low-quality CAD models in which geometries are realized by duplicating modeling features instead of editing an existing feature. (d) The CAD model breaks when the radius parameter of two adjacent fillet features increases and the resulting curved edges start to overlap. HiM helps to recover such broken models.

(3) **Editing and avoiding the design of low-quality CAD models**: While various modeling workflows can result in CAD models that look exactly the same, their quality can differ depending on the modeling features that were used and how they interact. Low-quality CAD models [Aranburu et al. 2022] are hard to adapt, as a change to a geometry element often requires the restructuring of modeling features or changes to many unrelated modeling features. Oftentimes, this is the result of modifications or additions to CAD models that do not leverage existing

modeling features and instead are realized by adding new modeling features that do not properly interact with existing modeling features. Figure 6c shows an example of a cube that is further enlarged by extruding the faces of the cube instead of editing the dimensions of the original sketch as well as the height of the extrusion feature. This introduces additional extrusion features that all contribute to the dimensions of the cube. HiM can avoid such practices, as it offers convenient techniques for inspecting existing modeling features before making changes. Furthermore, when designers are editing a CAD model which already has a low quality, HiM helps with understanding confusing interactions between modeling features.

(4) **Repairing broken models**: Even now, 30 years after the introduction of feature-based modeling, some peculiar combinations of modeling features can lead to edge cases that are not properly rendered in CAD modeling environments and result in broken models. These edge cases differ per CAD tool, as software manufacturers use different modeling kernels [Gerbino et al. 2003]. Figure 6d shows an edge case in Fusion 360 in which two fillet features are added to the inner and outer edges of a tube. The model entirely collapses when the radius parameters of the fillet features are too large and the curved faces overlap, resulting in a geometry different from the intended tubular shape. Such encounters leave designers confused and are especially problematic when broken models are accidentally saved, as it very hard to find and repair these conflicting modeling features in the future. Using HiM, conflicting modeling features are easy to diagnose. When selecting, for example, the remaining geometry element of the collapsed model in Figure 6d, the animation of the fillet features enlarges the radius parameter of the fillet feature from zero to its actual value. HiM's visualizations clearly show that when the two fillet features overlap, the extrude feature breaks and the model collapses.

(5) **Ad-hoc learning from high-quality models**: HiM turns every CAD model into a potentially powerful learning tool for feature-based CAD modeling. CAD tools traditionally render the entire modeling history as a long list of modeling features, making it hard to find sequences of modeling features that contribute to a specific geometry element. HiM allows novices to select intricate 3D geometry elements in existing CAD models that they don't know how to model themselves. The animated visualizations of HiM offer a first step to learning about the modeling features that are used, and how they interact, to realize the specific geometry element. Also, for more seasoned users, HiM is helpful for inspecting existing CAD models and learning about alternative clever strategies from other designers (Figure 2).

6 IMPLEMENTATION

Our proof-of-concept implementation of HiM is implemented as a plugin for Autodesk Fusion 360 using the Fusion 360 Python API[2]. The plugin is created using Python 3.9 and works on the May 2023 release of the plugin API and V.2.0.15299 of Fusion 360. Similar to other CAD environments, Fusion 360 stores the modeling history of a CAD model as a list of modeling features, which is accessible through the API. As our approach solely uses this information, we do not require additional instrumentation or annotations of CAD models. Our proof-of-concept implementation is implemented as a plugin for Autodesk Fusion 360 and thus only works on CAD models with a *.f3d* file format. The approaches and algorithms, outlined in this section, however, go beyond the specifics of Fusion 360 and can be used as a reference work when supporting HiM in other feature-based CAD tools.

[2]https://help.autodesk.com/view/fusion360/ENU/?guid=GUID-A92A4B10-3781-4925-94C6-47DA85A4F65A

6.1 Algorithm for Filtering Modeling Features

To filter all relevant modeling features, when selecting a geometry element, HiM finds all features that created or modified the geometry element, from the modeling history of the full project available through the Fusion API. When selecting an edge, HiM finds the modeling features that created or modified the two adjacent faces. For vertices, HiM considers modeling features relevant for the faces that connect to all edges shared by the selected vertex. As such, we translate the problem of finding relevant modeling features for the selected geometry element into finding relevant modeling features for one or multiple faces. For every face, our algorithm works in two stages. First, we find modeling features that created the face. Then, we find modeling features that modified the face. This two-step process is required to extract all relevant modeling features as geometry elements often get changed after their creation. HiM, however, does not differentiate between these modeling features when presenting them to the user.

In very simple cases, a single modeling feature is responsible for creating a face, such as faces created by extruding or revolving a sketch. Oftentimes, however, multiple modeling features contribute to the creation of a face. For example, a curved face, realized using a fillet modeling feature on an edge, while the edge was created by applying an extrusion modeling feature on a sketch. As covered in Section 4.3, by default, HiM finds all modeling features for a selected geometry element until a sketch is found as this is an entity that is very defining for a geometry and easy to modify. The "show more" button loads additional modeling features in cases where the sketch that was found is not defined on one of the ground planes (XY, XZ, YZ) of the design.

Algorithm 1 (lines 6-14) shows pseudocode for finding modeling features that created a selected geometry element by processing all modeling features in the project through the API. For every modeling feature, a list of faces, created by this feature, is available through the API. The function *find_creation_features()* (defined in Algorithm 2, line 1), uses this information to verify whether features in the modeling history are relevant for the selected geometry element. The function *get_input_geometries()* (defined in Algorithm 2, line 28) is then used to find additional geometry elements to which the modeling features are applied. The next iteration of the algorithm then finds the modeling features which created these additional geometry elements. This process continues until a sketch is encountered, and no new geometry elements need to be processed.

HiM then proceeds to extract all features that further modify the selected geometry element. This includes, for example, extrusion operations that cut holes in a selected face or a fillet operation that introduces a rounded edge to the face. As the Fusion API does not expose the geometry elements that are modified by modeling features, a custom implementation is needed. As shown in Algorithm 1 (lines 17-21), this is realized by verifying, for every feature in the modeling history, if it modifies the geometry element by comparing the geometry element before and after applying the feature. We consider a vertex or edge to be modified when the 3D coordinates change. A face is modified when the number of edges or vertices changes or when the area or centroid changes.

6.2 Real-Time Rendering of 3D Animated Interactive Visualizations

HiM renders a short clip in real-time consisting of animated visualizations of every modeling feature in the filtered set. The slider in the control panel allows for changing the speed of the animation between 2 and 10 seconds per modeling feature.

To render animations of modeling features in real-time and also allow for viewport changes, animations are realized by incrementally updating parameter values of modeling features over time from zero to the actual parameter value. For example, animating an extrusion, revolve feature, or fillet operation is realized by temporarily varying respectively the

Algorithm 1 Algorithm to extract the relevant modeling features from the modeling history based on a selected geometry element

```
function EXTRACT_RELEVANT_FEATURES(selected_entity : {vertex, edge, face})
    relevant_features ← []
    geometry_elements ← [selected_entity]

    //Extract all features that (in)directly created the selected entity
    while geometry_elements ≠ empty do
        new_geometry_elements ← []
        for geometry_element in geometry_elements do
            creation_features ← find_creation_features(geometry_element)
            relevant_features.extend(creation_features)
            new_geometry_elements.extend(get_input_geometries(creation_features))
        end for
        geometry_elements ← new_geometry_elements
    end while

    //Extract all features that modified the selected entity after its creation
    for all feature in timeline.features do
        if selected_entity.modified_by(feature) then        ▷ Compare entity before and after applying feature
            relevant_features.append(feature)
        end if
    end for

    return relevant_features
end function
```

distance, angle, or radius parameters. For circular and rectangular patterns, on the other hand, we update respectively the angle and distance in X and Y direction. As such, users can change the viewport of the modeling environment while HiM renders animations. To ensure our animations are also clear in more complex models that are prone to clutter, we render all other geometry elements transparent and additionally highlight the profile or edge on which the modeling feature is applied. Our rendering approach also ensures that animations are immediately recognizable, as they are consistent with visual cues rendered by Fusion 360 when applying and updating the modeling feature (Section 4.2).

Updating and thus rendering modeling features requires time, which depends on the overall performance of the system and the complexity of the modeling feature as well as the geometry to which it is applied. To ensure animations are smooth and finish within the user-configured timeframe, we adjust the step increment of the parameter value, which defines the frame rate of the animation. As we do not know beforehand how much time it takes to render a modeling feature, we start with an initial estimate of the frame rate: 15 frames for 2-second animations and 75 frames for 10-second animations. This frame rate is then divided by the actual parameter value in the modeling feature and used as an initial guess for the step increment of the parameter value of the modeling feature. After rendering a frame of the animation, the step increment and thus also the frame rate is further fine-tuned based on the execution time of the last frame. Using this approach, complex modeling features are executed for every frame of the animation. Therefore, our frame rate is on average significantly lower compared to production-quality animations. While the frame rate is still sufficient for visualizing modeling features, implementing future versions of HiM directly in Autodesk Fusion 360 avoids additional delays introduced by the plugin API.

Algorithm 2 Utility functions used by Algorithm 1 to find modeling features

```
1:  function FIND_CREATION_FEATURES(geometry_element : {vertex, edge, face})
2:      creation_features ← []
3:      faces ← extract_faces(geometry_element)
4:      for feature in timeline.features do
5:          for face in faces do
6:              if feature not in creation_features & feature.created(face) then      ▷ Exposed by Fusion API
7:                  creation_features.append(feature)
8:              end if
9:          end for
10:     end for
11:     return creation_features
12: end function
13:
14: function EXTRACT_FACES(geometry_element : {vertex, edge, face})
15:     if type(geometry_element) is face then
16:         return geometry_element
17:     else if type(geometry_element) is edge then
18:         return geometry_element.faces                          ▷ faces connected through this edge
19:     else if type(geometry_element) is vertex then
20:         faces ← []
21:         for edge in geometry_element.edges do                  ▷ edges bounded by this vertex
22:             faces.extend(edge.faces)
23:         end for
24:         return faces
25:     end if
26: end function
27:
28: function GET_INPUT_GEOMETRIES(features)
29:     input_geometries ← []
30:     for feature in features do
31:         for input_geometry in feature.input_geometries do
32:             if type(input_geometry) is not sketch then         ▷ input_geometry is {vertex, edge, face}
33:                 input_geometries.append(input_geometry)
34:             end if
35:         end for
36:     end for
37: end function
```

For modeling features that do not have parameter values, such as *import*, *export*, *split*, *mirror*, and *combine*, we render custom fade-in and fade-out effects by changing the scale and opacity level of the geometries. Figure 7 shows animations for the 20 modeling features supported in the current version of HiM. While this set represents the large majority of features present in 3D CAD models (Section 7), supporting additional modeling features is straightforward. It requires specifying the parameters to animate the feature, or alternatively write a custom visualization for features that do not have parameters. As such, our approach is extensible and could even support the animation of third-party feature implementations[3].

[3]https://help.autodesk.com/view/fusion360/ENU/?guid=GUID-FA7EF128-1DE0-4115-89A3-795551E2DEF2

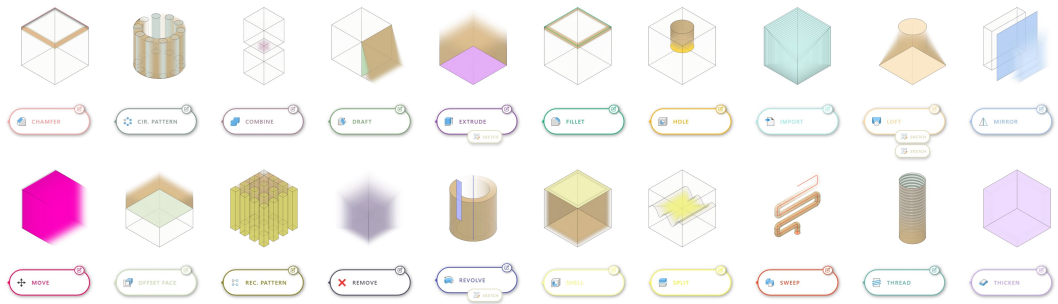

Fig. 7. An overview of the animated visualizations for each of the 20 modeling features supported by HiM.

6.3 Rendering an Additional 3D Interactive Window in Fusion 360

Instead of animating the actual CAD model, HiM renders the animated visualizations in a separate 3D interactive window. As such, the original CAD model is preserved and always visible, in addition to the animations which explain selected geometry elements. Autodesk Fusion 360, however, does not offer support for rendering an additional 3D interactive window. We realize this in our proof-of-concept implementation by launching a second instance of Fusion 360 in which we remove all traditional menu items and render a control panel through the Fusion API using HTML and Javascript. In this second Fusion 360 instance, animations are rendered on a copy of the 3D model. Updates to the model and requests for explaining a geometry element are communicated from the original Fusion 360 instance to this second instance via web sockets. In the future, CAD manufacturers can leverage our contributions to realize more elegant implementations of HiM directly in the source code of CAD tools.

7 COMPATIBILITY OF CURRENT VERSION OF HIM WITH EXISTING CAD MODELS

While the proof-of-concept implementation of HiM only supports CAD models in the *.f3d* file format, our approach solely uses the information available in these files and does not require additional metadata. As such, HiM can be used to analyze any existing *.f3d* CAD model. While our current version supports animations for 20 popular modeling features, more niche modeling features and custom features can be supported in the future. To verify the compatibility of the current version of HiM with existing 3D models and gain insights into which modeling features still require support in future versions, we ran a small analysis using the *ABC dataset* [Koch et al. 2019]. This dataset consists of 1,000,000 feature-based parametric CAD models designed in OnShape[4]. 4,786 models were excluded that were saved as, for example, *STEP* files and thus did not include any modeling features. This leaves us with 995,214 models for the analysis embedding 7,762,087 modeling features.

The analysis of all the modeling features in our dataset shows that our 20 supported modeling features represent 95.3% of all modeling operations in the dataset. The remaining 4.7% of modeling features represent 550 unique modeling features. Many of which were very niche third-party modeling features implemented in OnShape, such as modeling features that act as a shortcut for adding a dove-tail or a CNC dog bone. Our analysis also shows that 90% of the models (898,468 models), are constructed using only the 20 features supported by HiM. This demonstrates that our proof-of-concept implementation already supports animating the most important modeling features. The current

[4]https://www.onshape.com/

version of HiM, however, also offers partial support for CAD models embedding modeling features that are not yet supported. Those features still show up in HiM's control panel, but animations cannot be rendered as the parameters for animating these modeling features are unknown.

8 DISCUSSION AND FUTURE WORK

While the current version of HiM can already be used to gain important insights in the modeling history of CAD models, our work opens several avenues for future research.

First, in this article, we explained the rationale behind our proof-of-concept implementation and demonstrated its effectiveness using example cases. To further validate the effectiveness HiM, future research can compare the strengths and weaknesses of our approach to traditional approaches, that require manual exploration of the entire modeling history, through user studies. Although HiM is designed to offer insights into how specific geometry elements of CAD models are realized, different sets of modeling features are revealed depending on the selected geometry element. As such, only portions of the modeling history are communicated over time. While this is effective for modifications to the selected geometry elements, manual exploration of the entire modeling history might offer benefits for gaining a holistic understanding of the entire design workflow when many changes are desired.

Second, while HiM points designers to modeling features that are relevant for making changes to selected geometry elements, future research can investigate how to support designers further in making those modifications. In feature-based CAD modeling, parameter values can be used across different modeling features. Hence, making changes to a parameter might result in unexpected changes elsewhere in the model. This opens novel opportunities for introducing intelligibility techniques in CAD modeling environments. Examples include feed-forward techniques [Vermeulen et al. 2013] to communicate the impact of modifications before committing the changes.

Third, as discussed in this article, HiM also brings novel opportunities for learning CAD modeling techniques from existing designs. There are, however, major quality differences in design workflows, and one needs to be careful to only learn and adopt best practices. Future research can focus on assessing the quality of design histories by embedding metadata, such as the designer's expertise, in the model. Furthermore, machine learning techniques can also offer new opportunities for automated quality assessment of CAD design workflows.

Finally, HiM is currently a proof-of-concept implementation and several additional features can further improve the ease of use. For example, automated viewport changes in the animated visualization could further speed up the workflow as small visual renderings are immediately in focus.

9 CONCLUSION

In this paper, we presented History in Motion (*HiM*), an interactive visualization tool to help designers understand and explore modeling histories of 3D CAD models. HiM achieves this by filtering and visualizing relevant modeling features after the selection of geometry elements. Visualizations of modeling features are rendered as interactive animations on top of the CAD model which offers designers insights in how modeling features interact to realize the geometry. A control panel allows for a deeper exploration of the modeling features, with shortcuts for making modifications. We believe HiM is a significant contribution to facilitating the reuse of models and empowering designers to understand and make modifications to advanced CAD models. Design practices, in which existing CAD models are combined and modified, gain significant importance given the popularity of online CAD model repositories, such as Thingiverse.

ACKNOWLEDGMENTS

This research was supported by the Special Research Fund (BOF) of Hasselt University. We thank the anonymous reviewers for their invaluable and constructive comments. Special thanks go to Tovi Grossman for his insightful discussions throughout the development of this work.

REFERENCES

Saeema Ahmed, Ken M. Wallace, and Luciënne T. Blessing. 2003. Understanding the differences between how novice and experienced designers approach design tasks. *Research in Engineering Design* 14, 1 (Feb. 2003), 1–11. https://doi.org/10.1007/s00163-002-0023-z

Joachim Altmeyer, Stefan Ohnsorge, and Bernd Schürmann. 1994. Reuse of design objects in CAD frameworks. In *IEEE INTERNATIONAL CONFERENCE ON COMPUTER AIDED DESIGN*. Citeseer, 754–754.

P.T.J. Andrews, T.M.M. Shahin, and S. Sivaloganathan. 1999. Design reuse in a CAD environment — Four case studies. *Computers & Industrial Engineering* 37, 1 (1999), 105–109. https://doi.org/10.1016/S0360-8352(99)00033-9 Proceedings of the 24th international conference on computers and industrial engineering.

Aritz Aranburu, Daniel Justel, Manuel Contero, and Jorge D. Camba. 2022. Geometric Variability in Parametric 3D Models: Implications for Engineering Design. *Procedia CIRP* 109 (2022), 383–388. https://doi.org/10.1016/j.procir.2022.05.266 32nd CIRP Design Conference (CIRP Design 2022) - Design in a changing world.

J. S. Busby. 1999. The Problem with Design Reuse: An Investigation into Outcomes and Antecedents. *Journal of Engineering Design* 10, 3 (1999), 277–296. https://doi.org/10.1080/095448299261335 arXiv:https://doi.org/10.1080/095448299261335

Jorge D. Camba, Manuel Contero, and Pedro Company. 2016. Parametric CAD modeling: An analysis of strategies for design reusability. *Computer-Aided Design* 74 (2016), 18–31. https://doi.org/10.1016/j.cad.2016.01.003

Minsuk Chang, Ben Lafreniere, Juho Kim, George Fitzmaurice, and Tovi Grossman. 2020. Workflow Graphs: A Computational Model of Collective Task Strategies for 3D Design Software. In *Proceedings of Graphics Interface 2020* (University of Toronto) *(GI 2020)*. Canadian Human-Computer Communications Society / Société canadienne du dialogue humain-machine, 114 – 124. https://doi.org/10.20380/GI2020.13

Hsiang-Ting Chen, Li-Yi Wei, Björn Hartmann, and Maneesh Agrawala. 2016. Data-Driven Adaptive History for Image Editing. In *Proceedings of the 20th ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games* (Redmond, Washington) *(I3D '16)*. Association for Computing Machinery, New York, NY, USA, 103–111. https://doi.org/10.1145/2856400.2856417

Jonathan D. Denning, William B. Kerr, and Fabio Pellacini. 2011. MeshFlow: Interactive Visualization of Mesh Construction Sequences. *ACM Trans. Graph.* 30, 4, Article 66 (jul 2011), 8 pages. https://doi.org/10.1145/2010324.1964961

Jonathan D. Denning and Fabio Pellacini. 2014. CrossComp: Comparing Multiple Artists Performing Similar Modeling Tasks.

Jonathan D. Denning, Valentina Tibaldo, and Fabio Pellacini. 2015. 3DFlow: Continuous Summarization of Mesh Editing Workflows. *ACM Trans. Graph.* 34, 4, Article 140 (jul 2015), 9 pages. https://doi.org/10.1145/2766936

Jozef Doboš, Niloy J. Mitra, and Anthony Steed. 2014. 3D Timeline: Reverse engineering of a part-based provenance from consecutive 3D models. *Computer Graphics Forum* 33, 2 (May 2014), 135–144. https://doi.org/10.1111/cgf.12311

Tao Du, Jeevana Priya Inala, Yewen Pu, Andrew Spielberg, Adriana Schulz, Daniela Rus, Armando Solar-Lezama, and Wojciech Matusik. 2018. InverseCSG: Automatic Conversion of 3D Models to CSG Trees. *ACM Trans. Graph.* 37, 6, Article 213 (dec 2018), 16 pages. https://doi.org/10.1145/3272127.3275006

Christoph M. Flath, Sascha Friesike, Marco Wirth, and Frédéric Thiesse. 2017. Copy, Transform, Combine: Exploring the Remix as a Form of Innovation. *Journal of Information Technology* 32, 4 (Dec. 2017), 306–325. https://doi.org/10.1057/s41265-017-0043-9

Sean Follmer, David Carr, Emily Lovell, and Hiroshi Ishii. 2010. CopyCAD: Remixing Physical Objects with Copy and Paste from the Real World. In *Adjunct Proceedings of the 23nd Annual ACM Symposium on User Interface Software and Technology* (New York, New York, USA) *(UIST '10)*. Association for Computing Machinery, New York, NY, USA, 381–382. https://doi.org/10.1145/1866218.1866230

Sean Follmer and Hiroshi Ishii. 2012. KidCAD: Digitally Remixing Toys through Tangible Tools. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Austin, Texas, USA) *(CHI '12)*. Association for Computing Machinery, New York, NY, USA, 2401–2410. https://doi.org/10.1145/2207676.2208403

Sascha Friesike, Christoph M. Flath, Marco Wirth, and Frédéric Thiesse. 2019. Creativity and productivity in product design for additive manufacturing: Mechanisms and platform outcomes of remixing. *Journal of Operations Management* 65, 8 (April 2019), 735–752. https://doi.org/10.1016/j.jom.2018.10.004

Jun Fu, Xiang Chen, and Shuming Gao. 2017. Automatic synchronization of a feature model with direct editing based on cellular model. *Computer-Aided Design and Applications* 14, 5 (2017), 680–692. https://doi.org/10.1080/16864360.2016.1273585 arXiv:https://doi.org/10.1080/16864360.2016.1273585

Salvatore Gerbino et al. 2003. Tools for the interoperability among CAD systems. In *Proc. XIII ADM-XV INGEGRAF Int. Conf. Tools and Methods Evolution in Engineering Design*.

Tovi Grossman, Justin Matejka, and George Fitzmaurice. 2010. Chronicle: Capture, Exploration, and Playback of Document Workflow Histories. In *Proceedings of the 23nd Annual ACM Symposium on User Interface Software and Technology* (New York, New York, USA) *(UIST '10)*. Association for Computing Machinery, New York, NY, USA, 143–152. https://doi.org/10.1145/1866029.1866054

Megan Hofmann, Gabriella Hann, Scott E. Hudson, and Jennifer Mankoff. 2018. Greater than the Sum of Its PARTs: Expressing and Reusing Design Intent in 3D Models. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (Montreal QC, Canada) *(CHI '18)*. Association for Computing Machinery, New York, NY, USA, 1–12. https://doi.org/10.1145/3173574.3173875

Takeo Igarashi and John F. Hughes. 2001. A Suggestive Interface for 3D Drawing. In *Proceedings of the 14th Annual ACM Symposium on User Interface Software and Technology* (Orlando, Florida) *(UIST '01)*. Association for Computing Machinery, New York, NY, USA, 173–181. https://doi.org/10.1145/502348.502379

Chad Jackson and Maura Buxton. 2007. The design reuse benchmark report: seizing the opportunity to shorten product development. *Aberdeen Group, Boston* (2007).

D.J. Kasik, W. Buxton, and D.R. Ferguson. 2005. Ten CAD challenges. *IEEE Computer Graphics and Applications* 25, 2 (2005), 81–92. https://doi.org/10.1109/MCG.2005.48

Kimia Kiani, George Cui, Andrea Bunt, Joanna McGrenere, and Parmit K. Chilana. 2019. Beyond "One-Size-Fits-All": Understanding the Diversity in How Software Newcomers Discover and Make Use of Help Resources. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (Glasgow, Scotland Uk) *(CHI '19)*. Association for Computing Machinery, New York, NY, USA, 1–14. https://doi.org/10.1145/3290605.3300570

Sebastian Koch, Albert Matveev, Zhongshi Jiang, Francis Williams, Alexey Artemov, Evgeny Burnaev, Marc Alexa, Denis Zorin, and Daniele Panozzo. 2019. ABC: A Big CAD Model Dataset For Geometric Deep Learning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Nicholas Kong, Tovi Grossman, Björn Hartmann, Maneesh Agrawala, and George Fitzmaurice. 2012. Delta: A Tool for Representing and Comparing Workflows. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Austin, Texas, USA) *(CHI '12)*. Association for Computing Machinery, New York, NY, USA, 1027–1036. https://doi.org/10.1145/2207676.2208549

Ben Lafreniere and Tovi Grossman. 2018. Blocks-to-CAD: A Cross-Application Bridge from Minecraft to 3D Modeling. In *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology* (Berlin, Germany) *(UIST '18)*. Association for Computing Machinery, New York, NY, USA, 637–648. https://doi.org/10.1145/3242587.3242602

Danny Leen, Raf Ramakers, and Kris Luyten. 2017. StrutModeling: A Low-Fidelity Construction Kit to Iteratively Model, Test, and Adapt 3D Objects. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology* (Québec City, QC, Canada) *(UIST '17)*. Association for Computing Machinery, New York, NY, USA, 471–479. https://doi.org/10.1145/3126594.3126643

Wei Li, Tovi Grossman, and George Fitzmaurice. 2012. GamiCAD: A Gamified Tutorial System for First Time Autocad Users. In *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology* (Cambridge, Massachusetts, USA) *(UIST '12)*. Association for Computing Machinery, New York, NY, USA, 103–112. https://doi.org/10.1145/2380116.2380131

Wei Li, Tovi Grossman, and George Fitzmaurice. 2014. CADament: A Gamified Multiplayer Software Tutorial System. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Toronto, Ontario, Canada) *(CHI '14)*. Association for Computing Machinery, New York, NY, USA, 3369–3378. https://doi.org/10.1145/2556288.2556954

Damien Masson, Jo Vermeulen, George Fitzmaurice, and Justin Matejka. 2022. Supercharging Trial-and-Error for Learning Complex Software Applications. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems* (New Orleans, LA, USA) *(CHI '22)*. Association for Computing Machinery, New York, NY, USA, Article 381, 13 pages. https://doi.org/10.1145/3491102.3501895

Toshio Nakamura and Takeo Igarashi. 2008. An Application-Independent System for Visualizing User Operation History. In *Proceedings of the 21st Annual ACM Symposium on User Interface Software and Technology* (Monterey, CA, USA) *(UIST '08)*. Association for Computing Machinery, New York, NY, USA, 23–32. https://doi.org/10.1145/1449715.1449721

Chandrakana Nandi, James R. Wilcox, Pavel Panchekha, Taylor Blau, Dan Grossman, and Zachary Tatlock. 2018. Functional Programming for Compiling and Decompiling Computer-Aided Design. *Proc. ACM Program. Lang.* 2, ICFP, Article 99 (jul 2018), 31 pages. https://doi.org/10.1145/3236794

Chandrakana Nandi, Max Willsey, Adam Anderson, James R. Wilcox, Eva Darulova, Dan Grossman, and Zachary Tatlock. 2020. Synthesizing Structured CAD Models with Equality Saturation and Inverse Transformations. In *Proceedings of the 41st ACM SIGPLAN Conference on Programming Language Design and Implementation* (London, UK) *(PLDI 2020)*. Association for Computing Machinery, New York, NY, USA, 31–44. https://doi.org/10.1145/3385412.3386012

Lora Oehlberg, Wesley Willett, and Wendy E. Mackay. 2015. Patterns of Physical Design Remixing in Online Maker Communities. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems* (Seoul, Republic of Korea) *(CHI '15)*. Association for Computing Machinery, New York, NY, USA, 639–648. https://doi.org/10.1145/2702123.2702175

Dan R. Olsen. 2007. Evaluating User Interface Systems Research. In *Proceedings of the 20th Annual ACM Symposium on User Interface Software and Technology* (Newport, Rhode Island, USA) *(UIST '07)*. Association for Computing Machinery, New York, NY, USA, 251–258. https://doi.org/10.1145/1294211.1294256

X. Peng. 2012. Assessing Novice CAD Model Creation and Alteration. *Computer-Aided Design and Applications* PACE (Aug. 2012), 9–19. https://doi.org/10.3722/cadaps.2012.pace.9-19

X. Peng, P. McGary, M. Johnson, B. Yalvac, and E. Ozturk. 2012. Assessing Novice CAD Model Creation and Alteration. *Computer-Aided Design and Applications* PACE (Aug. 2012), 9–19. https://doi.org/10.3722/cadaps.2012.pace.9-19

Raf Ramakers, Danny Leen, Jeeeun Kim, Kris Luyten, Steven Houben, and Tom Veuskens. 2023. Measurement Patterns: User-Oriented Strategies for Dealing with Measurements and Dimensions in Making Processes. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems* (Hamburg, Germany) *(CHI '23)*. Association for Computing Machinery, New York, NY, USA, Article 214, 17 pages. https://doi.org/10.1145/3544548.3581157

Thijs Jan Roumen, Willi Müller, and Patrick Baudisch. 2018. Grafter: Remixing 3D-Printed Machines. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (Montreal QC, Canada) *(CHI '18)*. Association for Computing Machinery, New York, NY, USA, 1–12.

https://doi.org/10.1145/3173574.3173637

Gabriele Salvati, Christian Santoni, Valentina Tibaldo, and Fabio Pellacini. 2015. MeshHisto: Collaborative Modeling by Sharing and Retargeting Editing Histories. *ACM Trans. Graph.* 34, 6, Article 205 (nov 2015), 10 pages. https://doi.org/10.1145/2816795.2818110

Rita Shewbridge, Amy Hurst, and Shaun K. Kane. 2014. Everyday Making: Identifying Future Uses for 3D Printing in the Home. In *Proceedings of the 2014 Conference on Designing Interactive Systems* (Vancouver, BC, Canada) *(DIS '14)*. Association for Computing Machinery, New York, NY, USA, 815–824. https://doi.org/10.1145/2598510.2598544

Jo Vermeulen, Kris Luyten, Elise van den Hoven, and Karin Coninx. 2013. Crossing the Bridge over Norman's Gulf of Execution: Revealing Feedforward's True Identity. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Paris, France) *(CHI '13)*. Association for Computing Machinery, New York, NY, USA, 1931–1940. https://doi.org/10.1145/2470654.2466255

Tom Veuskens, Florian Heller, and Raf Ramakers. 2021. CODA: A Design Assistant to Facilitate Specifying Constraints and Parametric Behavior in CAD Models. In *Proceedings of Graphics Interface 2021* (Virtual Event) *(GI 2021)*. Canadian Information Processing Society, 87 – 96. https://doi.org/10.20380/GI2021.11

Tom Veuskens, Danny Leen, and Raf Ramakers. 2022. Identifying Opportunities to Reimagine Parametric Modeling for Makers. http://hdl.handle.net/1942/37369 CHI '22 - WORKSHOP.

Herbert Voelcker and Aristides Requicha. 1977. Constructive solid geometry. (1977).