Comparative evaluation of WIMP and immersive natural finger interaction: a user study on CAD assembly modeling

Marius Fechter¹ · Benjamin Schleich¹ · Sandro Wartzack¹

Received: 10 October 2020 / Accepted: 18 May 2021 / Published online: 15 June 2021
© The Author(s) 2021

Abstract

Virtual and augmented reality allows the utilization of natural user interfaces, such as realistic finger interaction, even for purposes that were previously dominated by the WIMP paradigm. This new form of interaction is particularly suitable for applications involving manipulation tasks in 3D space, such as CAD assembly modeling. The objective of this paper is to evaluate the suitability of natural interaction for CAD assembly modeling in virtual reality. An advantage of the natural interaction compared to the conventional operation by computer mouse would indicate development potential for user interfaces of current CAD applications. Our approach bases on two main elements. Firstly, a novel natural user interface for realistic finger interaction enables the user to interact with virtual objects similar to physical ones. Secondly, an algorithm automatically detects constraints between CAD components based solely on their geometry and spatial location. In order to prove the usability of the natural CAD assembly modeling approach in comparison with the assembly procedure in current WIMP operated CAD software, we present a comparative user study. Results show that the VR method including natural finger interaction significantly outperforms the desktop-based CAD application in terms of efficiency and ease of use.

Keywords Natural user interface · Computer-aided design · Assembly modeling · Virtual reality

1 Introduction

In contrast to WIMP (Window, Icon, Menu, Pointing) interfaces (van Dam 1997), virtual and augmented reality open up the possibility for the user to interact with virtual objects close to reality. If, for example, an object with any orientation is to be placed in 3D space, a virtual reality (VR) application may allow this to be done quickly via natural grasping techniques. The user would not have to think about operating steps because he knows them from everyday experience, which allows him to focus better on the actual task (Weiser 1991). In contrast, when using a WIMP interface, 3D widgets are used to split a manipulation task into individual

sequences that can be operated with a 2D mouse (Houde 1992). Due to the need of dividing the procedure into several subsequent steps, the 2D mouse is less suitable for such a 3D manipulation task (LaViola et al 2017). In this contribution, the term manipulation task refers to spatial rigid object manipulation (LaViola et al 2017).

In the context of assembly modeling, the user interfaces and menu guidance of current computer-aided design (CAD) software is highly adapted to the 2D mouse. This allows users to insert constraints between components easily by clicking on the geometric elements, like surfaces or edges. In reality, a user would simply pick up the components and assemble them, assuming he knows the sequence and spatial locations in which they belong together. Since the assembly process in reality takes place by relocating and rotating mechanical components in space, the question arises whether it is possible to achieve a faster assembly with the aid of natural 3D manipulation techniques and semi-automatic constraint detection. The constraint detection algorithm thereby utilizes information on the spatial location, geometry and topology of the CAD components.

The aim of this paper is to determine which advantages the utilization of natural interaction can have in CAD

✉ Marius Fechter
marius.fechter@fau.de
Benjamin Schleich
schleich@mfk.fau.de
Sandro Wartzack
wartzack@mfk.fau.de

¹ Institute of Engineering Design, Technical Faculty,
Friedrich-Alexander-University Erlangen-Nuremberg,
Martensstr. 9, 91058 Erlangen, Germany

context. Therefore, the first purpose of this paper is the investigation whether positioning objects in 3D space benefits from natural interaction as a prerequisite for further procedure. The second purpose of this paper is the determination of the suitability of the natural interaction in the CAD context, taking assembly modeling as an example. Assembly modeling is chosen because there is a evident equivalent in reality. If the approach for assembly modeling is showing advantages compared to WIMP interaction, this would make further research on extending this new form of interaction to other steps of CAD work reasonable and necessary. This is, for example, the design of preliminary drafts in the early development phase of a product.

To answer the research question of whether natural finger interaction can be superior to WIMP interaction for relocating and for assembly modeling tasks, we developed a user interface for natural interaction and an algorithm for automatic constraint detection in CAD applications. The subsequent comparative evaluation of the approach via a user study allows assessing the usability of the overall method.

2 Related work

2.1 Natural finger interaction

There are various approaches in the literature that allow finger interaction and virtual grasping of objects to be implemented. Physically based approaches (Borst and Indugula 2006) use physics simulation software and spring damping elements to recreate the real word contact behavior of hands and objects in (VE)s. Since mostly convex rigid bodies are used in these simulations, point contacts occur, which do not perfectly represent the real contact of a finger with an object. In order to remedy this shortcoming, there is the approach of making the fingers deformable and thus achieving contact surfaces (Jacobs and Froehlich 2011). The necessary simulation is computationally intensive. One approach that does not require a complete physics simulation is the god object method (Ortega et al 2007; Jacobs et al 2012; Talvas et al 2013). Thereby, the god object is defined as the location of a virtual object representation, that always remains outside of other virtual objects (Zilles and Salisbury 1995). A third possibility is the utilization of heuristic methods. These methods determine the virtual object a user wants to grab based on intersections and finger positions or movements. Thus, a very early approach (Ullmann and Sauer 2000) checks the direction of the two surface normals at the points where the object intersects with the fingertips. If these normals are at an angle that is smaller than a defined value, the virtual object is considered grasped. Others use ray casting, for example, to investigate whether there is an object between certain fingers (Moehring and Froehlich 2005). An

interesting approach (Periverzov and Ilieş 2015) uses Voronoi regions to determine the objects the user was closest to in a defined number of previous time steps before a grasping action occurs. In principle, all these approaches can be used in virtual, augmented, or mixed reality environments.

2.2 Automatic constraint detection

In addition to natural interaction, the automatic detection of constraints between assembly components in CAD software without predefinition is a central aspect of our research. The focus of this publication is on improving the assembly process in CAD software rather than optimizing the assembly process in an immersive virtual environment. Therefore, we distinguish approaches working with predefined constraints from those, which automatically detect constraints. Although there is extensive literature on the subject of assembly process validation (Seth et al 2011; Wang et al 2016; Pascarelli et al 2018), the approaches often rely on predefined constraints (Jayaram et al 1999; Yao et al 2006; Valentini 2009, 2018).

To be able to assemble arbitrary assemblies without predefined constraints, the used algorithm must analyze the geometry of the components automatically. An early publication (Fa et al 1993) describes the idea to detect constraints between geometric shapes automatically by looking at opposite elements. If two geometric elements meet the conditions for a certain type of relationship within a certain tolerance, a constraint is applied. In the case of large assemblies with several hundred geometric surfaces, the analysis of possible constraints when changing the position of a component can be computationally intensive. For the purpose of reducing the number of potential surface pairs to be checked, several approaches (Marcelino et al 2003; Seth 2007) focus on components with intersecting bounding boxes. This method still results in a large number of potential surface pairs to be analyzed (Wang et al 2013). Another approach is that of only scanning surfaces that are in contact (Wang et al 2013).

An obvious requirement of the approaches considered above is that the objects must intersect to allow the detection of constraints between them. This limits the applicability for many assemblies. Mechanical components are often connected using screws in clearance holes, not having direct contact between the thread and the cylindrical clearance hole surface. However, the algorithm developed in this paper must be as flexible as possible in order to be able to process arbitrary assemblies.

2.3 Studies comparing 2D mouse and 3D interaction

When it comes to comparing desktop-based 2D mouse interaction and 3D manipulation techniques for spatial rigid

object manipulation tasks, divergent results can be found in literature. Some indicate no difference in performance efficiency for 3D positioning tasks (Bérard et al 2009) or 3D manipulation tasks (Alkemade et al 2017). Some even found insufficient performance in accuracy and efficiency of a multi-modal control interface, using hand gestures and eye tracking, compared to the mouse for manipulation tasks (Song et al 2014). Apart from that for a docking task as well as a positioning task a standard one-handed wand interface was found to be faster than 3D widgets operated with a mouse (Schultheis et al 2012). Overviews of further studies can be found in literature (Mendes et al 2018). Since the studies available in the current research do not distinguish tasks according to the number and type (translation/rotation) of degrees of freedom, this will be addressed in the present work.

Regarding the comparison of assembly modeling steps between WIMP and natural interaction, Toma et al (2012) describe a comparative evaluation of a desktop-based 2D mouse and a multimodal interface. It is described there that no constraints have to be defined in the virtual environment, but the components only must be brought into the correct spatial location. A user study showed that the assembly procedure in common CAD software was a lot slower, than specifying the location of the assembly in 3D space with a hand gesture. Unfortunately, this study compares a relocation task in the immersive environment with a CAD assembly task in a desktop setting.

In summary, although several research activities exist in the field of automatic constraint detection, we are not aware of any user study in the literature that compares conventional WIMP interaction with markerless natural finger interaction within an immersive environment for CAD assembly modeling tasks.

3 Method for natural CAD assembly modeling

Since the method for semi-automatic CAD assembly modeling has two primary components, both the finger interaction and the algorithm for constraint recognition will be presented in this section.

3.1 A combination of heuristic and physical grasping approaches

The method of our natural user interface (NUI) is based on an approach presented in Fechter and Wartzack (2017). However, for the sake of reproducibility, it is briefly highlighted in the following. Our method bases on two fundamental approaches: physics simulation and heuristics.

Therefore, we use a combination of physics-based and heuristic approaches.

In the following, we refer to three different parts of our approach: the tracked hand, the rigid body spring hand and the heuristic hand. The *tracked hand* contains the data captured by a markerless optical finger tracking sensor. The *spring hand* consists of 15 rigid body capsules representing the three outer phalanges of each finger (red in Fig. 1; shown for thumb and index finger only) and a cuboid representing the palm of the hand. Virtual spring-dampers, first presented in Borst and Indugula (2006), connect the rigid body finger segments. These virtual spring-dampers are controlled in such a way that the spring hand tries to follow the tracked hand. When a virtual object is grasped, in reality the tracked fingers would intersect with the virtual object (see Fig. 1b). In contrast, the physics simulation ensures the components of the spring hand to always stay outside the grasped object. Since the virtual spring-dampers make the spring hand try to follow the tracked hand, the rigid bodies exert forces at the contact points. This allows the user to push objects, for example.

The *heuristic hand* allows the detection of the user's intention to grasp an object. The first of two pieces of information necessary to recognize this intention is the identification of the specific object, the user wants to manipulate. Therefore, the heuristic hand consists of 15 trigger shapes, which are represented in Fig. 1a with green dotted lines. The trigger shapes are intentionally displaced relative to the bone of the tracked hand. In case of a grasping action this ensures the trigger shapes coming into contact with the virtual object prior to the rigid bodies of the spring hand. The position of each trigger shape $\mathbf{p}_t$ is calculated by Eq. 1, where $\mathbf{v}_d$ is the vector holding the displacement values, $\mathbf{X}_b$ is the 3x3 rotation matrix of the tracked bone and $\mathbf{p}_b$ is the position of the tracked bone. The rotation of the trigger shape is identical to the rotation of the corresponding finger bone. All vectors and matrices are given in the same global coordinate system.

$$\mathbf{p}_t = \mathbf{v}_d \cdot \mathbf{X}_b + \mathbf{p}_b \quad (1)$$

Using the trigger shapes allows the detection of intersections between them and virtual objects (depicted in orange in Fig. 1a). Assuming that the user grasps the object with his thumb and one or more of the other four fingers, the intended object can be identified. In detail, this requires an intersection between the virtual object and a thumb trigger shape as well as an intersection between the virtual object and at least one other finger trigger shape. The second piece of information that is necessary to identify the user's intention to grasp an object, is the detection of a physical grasping movement. Therefore, the angle between the direction of the four fingers, except the thumb, and the direction of the hand is considered. To avoid false detection due to tracking data noise, this angle is considered for the current frame and

Fig. 1 Approach to detect the user's intention of grasping a virtual object

several former frames. If the change of the angle is negative and the absolute value exceeds a defined threshold in multiple successive frames, the heuristic detects a grasping movement.

In case the heuristic detects the user's intention to grasp a virtual object, this object is bound to the movement of the hand. Then the spring hand no longer has any influence on the spatial location of the virtual object until the user releases it. In the specific frame, the virtual grasp is detected, we store the position and rotation of the virtual object ($\mathbf{p}_{o,g}$ and $\mathbf{X}_{o,g}$) and the position and rotation of the tracked hand ($\mathbf{p}_{th,g}$ and $\mathbf{X}_{th,g}$). Since the current spatial location of the tracked hand ($\mathbf{p}_{th,c}$ and $\mathbf{X}_{th,c}$) is known, the rotation between both spatial locations can be calculated:

$$\mathbf{X}_{th,c} = \mathbf{R}_{th} \cdot \mathbf{X}_{th,g} \rightarrow \mathbf{R}_{th} = \mathbf{X}_{th,c} \cdot \mathbf{X}_{th,g}^{-1} \quad (2)$$

With this knowledge, we can calculate the current position $\mathbf{p}_{o,c}$ and the current rotation $\mathbf{X}_{o,c}$ of the virtual object depending on the current spatial location of the hand:

$$\mathbf{p}_{o,c} = \mathbf{R}_{th} \cdot (\mathbf{p}_{o,g} - \mathbf{p}_{th,g}) + \mathbf{p}_{th,c} \quad (3)$$

$$\mathbf{X}_{o,c} = \mathbf{R}_o \cdot \mathbf{X}_{o,g} \stackrel{\mathbf{R}_o = \mathbf{R}_{th}}{=} \mathbf{R}_{th} \cdot \mathbf{X}_{o,g} \quad (4)$$

Releasing a virtual object works opposite to grasping. In order to detect the user's intention of releasing an object, the algorithm scans the data of the tracked fingers for an opening movement of the hand (see Fig. 2). For this purpose,

the angle between the direction of the four fingers, with the exception of the thumb, and the direction of the hand over several successive frames is used again. Other possibilities of releasing an object are an almost completely opened hand and the absence of intersections of the trigger shapes with the virtual object.

The precision of the hand tracking data allows a relatively accurate manipulation of virtual objects. However, since the detection of a release action involves several frames and the spatial location of the hand changes as a result, precise positioning can be affected. To avoid reducing the accuracy

Fig. 2 Release movement must occur over several time steps

of the placement by unintentional hand movements when releasing the object, the algorithm saves the spatial object locations for several previous frames. This enables us to retrospectively set the object location that was present at the beginning of the release action. In addition, the objects' spatial location is locked after a grasp release for a short period of time in order to avoid unintended displacements.

Since our approach for natural finger interaction lacks haptic feedback, which is an important source of information when interacting with physical objects in reality, we need to address other ways of feedback. In an immersive virtual environment, it is possible to provide visual and auditory stimuli to address the respective human senses. Therefore, when the user grasps a virtual object, its edges change their color from black to the shining color of the object. In addition, the user hears the sound of a finger tapping on a tabletop. In the case of a release action, the edges turn from object color to black and the release sound (a plastic cap falling onto a tabletop) is played.

3.2 Constraint recognition

3.2.1 Semi-automatic constraint detection algorithm

Initially, each component of an assembly can be moved in six degrees of freedom. Three for the position and three for the rotation. By inserting constraints, the degrees of freedom of a component are restricted successively.

In this paper, we aim at evaluating whether the automatic detection of constraints between components based

on their spatial location in VR is efficient and accepted by the user. We developed the algorithm specifically to evaluate the usability of the fundamental method exemplarily for a frequently occurring type of assembly case. Therefore, arbitrary rotationally symmetrical components can be assembled by coincident constraints. For example, assembly using hole patterns or distance constraints is currently not possible. Of course, during the development and the implementation of the algorithm, it was ensured that the extensibility for further assembly cases is given. The technical contribution of the presented method compared to prior work is that no collision detection is required to detect constraints anymore. As explained in the following section, collision detection is only used to avoid false constraint detection in the case of two parts intersecting.

In order to fixate rotationally symmetrical components usually three constraints are necessary. The detection of each of the three constraints requires the nonrecurring analysis of potentially matching surface pairs and the subsequent verification of the fulfillment of certain conditions. Therefore, as described in detail below, (1) the radius, (2) the information whether it is an inner or outer surface and (3) the center axis of the cylindrical surface are considered. In a first, nonrecurring step, the detection algorithm analyzes all components for cylindrical surfaces that could potentially match cylindrical surfaces of other components. The criteria include the approximate equivalence of the radius and the combination of one inner and one outer surface. An example for a cylindrical surface pair is depicted in Fig. 3a in light blue. After every object manipulation, the algorithm checks whether a cylindrical surface pair of any two components currently

Fig. 3 Detection of three constraints to restrict the degrees of freedom of the movable component

approximately fits together. Therefore, the algorithm uses data from the direction and the position of central axes belonging to the cylindrical surfaces identified in advance. On the one hand, the angle between the two directions α (see Fig. 3a) has to be smaller than the threshold $\alpha_{\min}$.

$$\alpha = \arccos(\mathbf{v}_m \cdot \mathbf{v}_f) < \alpha_{\min} \quad (5)$$

On the other hand, the position of the movable components surface $\mathbf{p}_m$ has to be located within a cylinder with radius $r_{\min}$ around the axis of the fixed component.

$$d = \|\mathbf{v}_f \times (\mathbf{p}_m - \mathbf{p}_f)\| < r_{\min} \quad (6)$$

As soon as the algorithm inserts the first constraint, the component will only be movable along and around the cylinders middle axis.

The ability to move the component only along the middle axis of the cylindrical constraint means that it will contact a specific surface of another component. Therefore, the algorithm performs a scan to identify the planar surface pair that will come together first. It uses ray casting (LaViola et al 2017) in the normal direction of all existing planar surfaces with a large number of linearly distributed rays per surface. If a ray hits another component, using the intersection points and the start points of the respective rays, the shortest distance between a planar surface pair can be determined. In Fig. 3b for example ray 1 defines the nearest surface pair, while ray 2 and ray 3 do not intersect with another assembly component. Once the nearest pair has been identified, during user manipulations we calculate the distance separating this pair. As soon as it falls below a specific value, the constraint is applied. Thereafter, the component is only rotatable around the middle axis of the first cylindrical constraint.

In order to restrict the last rotational degree of freedom (DOF), the algorithm nonrecurrently analyzes the components for parallel, matching drillings. The values for matching diameters of threads and clearance holes are stored in

tables. In order to calculate the distance between two single matching drilling positions, the coordinates are projected onto one common plane with their joint middle axis as normal vector. Once the distance between two single matching drilling positions of the moving component $\mathbf{p}_{m,d}$ and the fixed component $\mathbf{p}_{f,d}$, see Fig. 3c, falls below a defined value, the last constraint is added. Thus, all degrees of freedom of the component are restricted and it is assembled completely.

3.2.2 Constraint detection feedback

Providing the user with clear and understandable feedback is important in VEs. In order to give feedback on a recently inserted constraint, we address visual and auditory perception. For the visual feedback, the surfaces used for a constraint will be colored for a short period, depicted in Fig. 4a–c. This visualization is visible to the user, even if another object or surface would normally obscure it. In action, it happens that a constraint is detected when the user has grasped an object and is currently manipulating it. In order to clearly signal to the user that a new constraint has been inserted, we use auditory feedback additionally. A deep gong-like sound signals the recognition of a task fulfillment, such as inserting a constraint.

To prevent the algorithm from inserting a constraint between surfaces that cannot come together in reality, collision detection is permanently performed during object manipulation. If the algorithm detects a collision between two unconstrained assembly components, it stops constraint insertion until the collision is resolved. In case of a collision present, the grasped object is marked red, depicted in Fig. 4d.

When inserting a new constraint, it is also possible to display a dialog box to the user in order to confirm or discard the constraint in the immersive VE. This dialog box will not be utilized to maintain comparability with the CAD software used.

Fig. 4 Procedure for mounting the two pipes with the surfaces used for the constraints marked in light blue color (a–c). Intersecting components marked red (d). The red and white hand is the virtual repre-

sentation of the user's hand. You can find a video demonstrating the method at <https://vimeo.com/497012463> (Color figure online)

3.2.3 Limitations of the method

A limitation of the current method is the restriction to rotationally symmetric components that can be assembled semi-automatically. This frequently occurring case, however, serves only as an exemplary use case for the consideration of the suitability of natural finger interaction for assembly modeling. The focus of this paper is not on the development of algorithms for all possible constraints, but rather on the evaluation of the usability of this “natural” form of assembly modeling. Nevertheless, during the development of the method, attention was given to the need to allow parts of the algorithms to be easily extended to include constraints such as distance, parallelism, angular offset, tangentiality or drilling patterns. In addition, functionality such as undoing an action or deleting inserted constraints are essential for a productively used application. However, since this system is used exclusively as a demonstrator system, these functions are currently not required.

4 User study

In the following, we want to present a summative evaluation in the form of a usability study (Sauro and Lewis 2016) to prove the usability of the overall method including the interaction method and the constraint detection algorithm. The usability of a user interface (UI) considers aspects like task performance, ease of use, user comfort, and system performance (Hix and Hartson 1993). Most importantly, we want to look at the task performance, in our case the efficiency in terms of time necessary for task fulfillment, and the ease of use.

4.1 Objectives and hypotheses

As a first step, we want to examine the differences in task performance between frequently used 3D widgets operated with the mouse and keyboard and the newly developed finger interaction method. Therefore, we formulate the following one-tailed null hypotheses and alternative hypotheses:

- *Hypothesis $H1_0$* : The sample mean in the time necessary to complete *1D tasks* in the VR environment is less than or equal to the desktop environment. The alternative hypothesis $H1_1$ is that the mean of the sample from the VR environment is greater than the sample from the desktop environment.

$$H1_0 : \mu_{1D,VR} \leq \mu_{1D,desktop}$$

$$H1_1 : \mu_{1D,VR} > \mu_{1D,desktop}$$

- *Hypothesis $H2_0$* : The sample mean in the time necessary to complete *3D tasks* in the VR environment is greater than or equal to the desktop environment. The alternative hypothesis $H2_1$ is that the mean of the sample from the VR environment is less than the sample from the desktop environment.

$$H2_0 : \mu_{3D,VR} \geq \mu_{3D,desktop}$$

$$H2_1 : \mu_{3D,VR} < \mu_{3D,desktop}$$

- *Hypothesis $H3_0$* : The sample mean in the time necessary to complete *6D tasks* in the VR environment is greater than or equal to the desktop environment. The alternative hypothesis $H3_1$ is that the mean of the sample from the VR environment is less than the sample from the desktop environment.

$$H3_0 : \mu_{6D,VR} \geq \mu_{6D,desktop}$$

$$H3_1 : \mu_{6D,VR} < \mu_{6D,desktop}$$

The main objective of this summative usability study is to compare the common CAD assembly procedure and the developed finger interaction method in conjunction with the automatic constraint recognition algorithm. Therefore, we formulate the following one-tailed null hypotheses and alternative hypotheses:

- *Hypothesis $H4_0$* : The sample mean in the time necessary to complete assembly tasks in the VR environment is greater than or equal to the desktop environment. The alternative hypothesis $H4_1$ is that the mean of the sample from the VR environment is less than the sample from the desktop environment.

$$H4_0 : \mu_{A,VR} \geq \mu_{A,desktop}$$

$$H4_1 : \mu_{A,VR} < \mu_{A,desktop}$$

- *Hypothesis $H5_0$* : The sample mean in the usability score from the System Usability Scale (SUS) questionnaire in the VR environment is less than or equal to the desktop environment. The alternative hypothesis $H5_1$ is that the mean of the sample from the VR environment is greater than the sample from the desktop environment.

$$H5_0 : \mu_{SUS,VR} \leq \mu_{SUS,desktop}$$

$$H5_1 : \mu_{SUS,VR} > \mu_{SUS,desktop}$$

4.2 Study design

4.2.1 Participants and stereo blindness

Twelve participants were recruited from the Friedrich–Alexander University, Engineering Design. Two of them were

excluded from the study due to stereo blindness. The test for stereo blindness performed is similar to the one described by Dankert et al (2013) and Liu et al (2017). The ten tested subjects (9 male, 1 female; 10 right-handed) ages 22–32 averaged 26.7 (SD = 3.37). Eight participants hold a master's degree in engineering design, one holds a master's in electrical engineering, one is a undergraduate student in mechatronics. All except one have experience in CAD software. Five participants with no experience in VR, four participants with moderate experience in VR and one participant with considerable experience in VR.

4.2.2 Apparatus

Our user study compares two environments. The first environment (hereafter referred to as the *desktop environment*) is a usual engineering workplace with a 22-inch 2D monitor, mouse and keyboard. The second environment (hereafter referred to as the *VR environment*) uses a HMD (Oculus Rift CV1), to which a sensor for markerless hand tracking (Leap Motion) is attached. In our setup, the HMD is tracked by two infrared cameras. Both environments use the same CAD software (ANSYS SpaceClaim) on the same computer (i7-6700K, GeForce GTX 980 Ti, 16 GB RAM).

4.2.3 Tasks

Each participant was asked to perform the same tasks in both the desktop and VR environment. There are two types of tasks: relocate and assemble.

The objective of the task type *relocate* is getting an object into a target position and rotation as quickly as possible. During the test, the user must perform a total of six tasks of this type. Two each with one (hereafter referred to as 1D task), three (3D task) and six DOF (6D task). In the case of a 1D task, the object must be moved in one direction (see Fig. 5). For tasks with three degrees of freedom, the object must be moved in two translational directions and around one rotational axis (see Fig. 6). In case of a task with six DOF, the participant must manipulate all translational and rotational degrees of freedom (see Fig. 7).

The objective of the task type *assemble* is assembling several components. In the user study, the participants have to assemble two different assemblies as fast as possible. The

Fig. 5 Exemplary one-dimensional displacement task (1D task): the object is translated along the *x*-axis

first assembly is a pipe connection with two parts, depicted in Fig. 8a. The second assembly is a 1-cylinder engine of a model aircraft with the housing, a cover, the exhaust, and the cylinder head. The components spatial starting location of the assembly are depicted in Fig. 8. Each of the two assembly tasks has to be performed two times for each user. Table 1 gives an overview of the various tasks.

4.2.4 Procedure

Since the user study is conducted in within-subjects design, each user tests both environments. This ensures that the effects of individual task fulfillment capabilities of a participant on the results are eliminated as far as possible. In order to prevent the results from being falsified by learning effects during task execution, the environment the user begins with, is selected by lot. This means that the half of the subjects starts with one of the two environments and vice versa.

In both environments, prior to the task execution the user can perform one 1D, one 3D and one 6D task as well as the two assembly tasks for training purposes (see Table 1). Before the task measurement begins, the participant can take a close look at the current task. When he signals that he is ready to start, a timer counts down from three in both environments before the time measurement begins. For each relocate task, the movable object is colored green, and the spatial target location of the object is marked red and semi-transparent, depicted in Fig. 9. The fulfillment of the task is automatically detected when the position and rotation of the relocatable object is close to that of the target. The threshold was chosen to be 1.5 centimeters for the position and 5° per axis for the rotation. For each assembly task, fulfillment is recognized when three constraints are assigned to every moveable component.

Fig. 6 Exemplary three-dimensional displacement task (3D task): the object is moved along the *x* and *y* axis, as well as rotated around the *y* axis

Fig. 7 Exemplary six-dimensional displacement task (6D task): the object is moved along as well as rotated around *x*, *y* and *z* axis

Fig. 8 Models used in the user study for the two tasks of the type assembly

Fig. 9 3D task of relocate type with the 3D widget in the desktop environment

In case the desktop environment is tested first, the test coordinator demonstrates the functionality of the CAD program operation for all types of tasks and shows the two assembly procedures. In addition, the participant is provided with printouts containing information about necessary buttons, shortcuts and constraints. During the test execution, the study coordinator will provide verbal support as necessary. If the user is ready to start a task, he clicks onto a button to start the countdown for a measurement.

The first step in the VR environment is setting the inter-pupillary distance (IPD) for the subject. Subsequently, the test for stereo blindness described above is performed. To teach the user how to interact by fingers, an avatar is present in the immersive VE. This avatar demonstrates how objects

Table 1 Overview of the tasks for both environments

Task no.	Type	Description	Name
Training	Relocate	1D task number 1	
Training	Relocate	3D task number 1	
Training	Relocate	6D task number 1	
1	Relocate	1D task number 1	1D-t1
2	Relocate	1D task number 2	1D-t2
3	Relocate	3D task number 1	3D-t1
4	Relocate	3D task number 2	3D-t2
5	Relocate	6D task number 1	6D-t1
6	Relocate	6D task number 2	6D-t2
Training	Assemble	Assembly 1 training	
7	Assemble	Assembly 1 trial 1	A1-t1
8	Assemble	Assembly 1 trial 2	A1-t2
Training	Assemble	Assembly 2 training	
9	Assemble	Assembly 2 trial 1	A2-t1
10	Assemble	Assembly 2 trial 2	A2-t2

can be nudged, grasped and assembled. To start the measurement for a task, the subject must give the avatar thumbs up for a countdown to begin.

The task sequence has been designed to ensure a maximum duration of 45 minutes for one test run.

4.2.5 Methods

During the test, the duration to complete a task is measured automatically, hereafter referred to as task performance. In

addition, we measure the accuracy of the placements in the tasks of the type relocate. However, since the fulfillment of a task is automatically recognized as soon as the limit values are met, it is not useful to assess the accuracy values. At the end of each test for an environment, the participant was asked to answer System Usability Scale (SUS) questions (Brooke et al 1996) in the form of a five-point Likert scale (Joshi et al 2015). The System Usability Scale (SUS) questionnaire is a frequently used instrument to gather the subjective assessments of usability. Here it is used to map the subjective assessments of the subjects regarding the usability of the interfaces in both test environments.

4.3 Results

Time measurements of task type relocate

In the experiments, the participants performed a total of 200 tasks, 120 tasks of the type relocate and 80 tasks of the type assemble. The time measurement results for the relocate tasks are depicted in Fig. 10. As we are interested in the differences between the two environments, we calculate the differences between the relocate task measurements for each user, depicted in Fig. 11.

In order to determine whether the mean difference between the data sets for the single tasks in both

environments is zero, we use the one-tailed paired-sample t test. To be able to use the one-tailed paired-sample t test, some prerequisites must be fulfilled. Among other things, the difference data of the single tasks should be approximately normally distributed and should not contain any outliers. Therefore, the data belonging to the six tasks was checked for normal distribution using the Kolmogorov–Smirnov test including Lilliefors significance correction (Öztuna et al 2006), reported in “Appendix 2.”

For the 1D task number 1 (1D-t1) higher values ($M = 4.06$, $SD = 1.74$) were observed in the VR environment than in the desktop environment ($M = 1.65$, $SD = 0.71$). This difference could be demonstrated as significant ($t(9) = -4.18$, $p = 0.001$, $d = 1.32$). Since the p value is less than $\alpha = 0.05$, we reject the null hypothesis H_{I0} in favor of the alternative hypothesis H_{I1} . For the 1D task number 2 (1D-t2) higher values ($M = 5.37$, $SD = 3.99$) were observed in the VR environment than in the desktop environment ($M = 1.32$, $SD = 1.10$). Due to outliers, no one-tailed paired-sample t test was conducted for the second 1D task.

For the 3D tasks lower values (3D-t1: $M = 5.33$, $SD = 2.44$ and 3D-t2: $M = 9.51$, $SD = 12.34$) were observed in the VR environment than in the desktop environment (3D-t1: $M = 14.90$, $SD = 10.87$ and 3D-t2: $M = 15.74$, $SD = 3.82$). Due to outliers, the prerequisites to be able to conduct an one-tailed paired-sample t test was

Fig. 10 Time required to fulfill the 1D, 3D and 6D tasks in both environments. The box plot shows the median (red line), outliers (red plus sign), the area from 25. (q_1) to 75. (q_3) percentile (blue box) and the whiskers (black dashed line). The whiskers extend to the most

extreme data value that is not an outlier. Outliers here are values that are either over $q_3 + 1.5 * (q_3 - q_1)$ or below $q_1 - 1.5 * (q_3 - q_1)$. In the presence of a normal distribution, this corresponds to a coverage of approximately $\pm 2.7\sigma$ and 99.3% (Color figure online)

Fig. 11 Difference in the time users needed to complete the 1D, 3D and 6D tasks between the both environments. For an explanation of the box plot elements see caption of Fig. 10. *Indicates normal distribution based on Kolmogorov–Smirnov test, Lilliefors significance correction

not given. Therefore we cannot provide a conclusion regarding null hypothesis $H2_0$ and the alternative hypothesis $H2_1$.

For the first 6D task (6D-t1) lower values ($M = 12.05, SD = 9.25$) were observed in the VR environment than in the desktop environment ($M = 55.10, SD = 24.83$). This difference could be demonstrated as significant ($t(9) = -6.08, p < 0.001, d = 1.92$). For the second 6D task (6D-t2), lower values ($M = 14.07, SD = 11.18$) were observed in the VR environment than in the desktop environment ($M = 60.66, SD = 14.54$). This difference is significant ($t(9) = -9.42, p < 0.001, d = 2.98$). Since the p value is less than $\alpha = 0.05$ for both 6D tasks, we reject the null hypothesis $H3_0$ in favor of the alternative hypothesis $H3_1$.

Time measurements of task type assemble

The time measurement results for the assemble tasks are depicted in Fig. 12. As we are interested in the differences between the two environments, we calculate the differences between the assemble task measurements for each user, depicted in Fig. 13.

Due to outliers in the differences (see Fig. 13), the prerequisites to conduct a one-tailed paired-sample t test are given for two samples (A1-trial2 and A2-trial2).

For the first trial of the first assembly (A1-trial1) lower values ($M = 11.92, SD = 6.51$) were observed in the VR environment than in the desktop environment ($M = 31.00, SD = 16.04$). Due to outliers, no one-tailed paired-sample t test was conducted for the first trial of

Fig. 12 Time required to fulfill the assembly tasks in both environments. For an explanation of the box plot elements see caption of Fig. 10

the pipe assembly. For the second trial of the first assembly (A1-trial2), lower values ($M = 8.04$, $SD = 3.79$) were observed in the VR environment than in the desktop environment ($M = 24.55$, $SD = 8.80$). This difference could be demonstrated as significant ($t(9) = -5.78$, $p < 0.001$, $d = 1.83$).

For the first trial of the second assembly (A2-trial1) lower values ($M = 30.70$, $SD = 7.18$) were observed in the VR environment than in the desktop environment ($M = 63.72$, $SD = 11.63$). Due to outliers, no one-tailed paired-sample t test was conducted for the first trial of the model airplane assembly. For the second trial of the second assembly (A2-trial2) lower values ($M = 32.35$, $SD = 7.38$) were observed in the VR environment than in the desktop environment ($M = 60.96$, $SD = 16.68$). This difference could be demonstrated as significant ($t(9) = -4.91$, $p < 0.001$, $d = 1.55$). Since the p value is less than $\alpha = 0.05$ for both assembly tasks, we reject the null hypothesis H_{0_0} in favor of the alternative hypothesis H_{4_1} .

SUS questionnaire

The answers to the SUS questions (listed in “Appendix 1”) are converted into values with 0 minimum (strongly disagree) and 4 maximum (strongly agree). Subsequently, for negative questions, the score is reversed by 4-value (Brooke et al 1996). The scores are depicted in Fig. 14 for the two single environments plus their difference per participant. The VR environment achieves a total of 87.75 points, while the desktop environment achieves

54.5 points. A Kolmogorov–Smirnov test including Lilliefors significance correction indicates that the differences in the SUS scores per user follow a normal distribution ($D(10) = 0.246$, $p = 0.089$).

An one-tailed paired-sample t test was conducted to compare the SUS scores in VR and desktop environment conditions. Higher values ($M = 3.51$, $SD = 0.31$) were observed in the immersive environment than in the desktop environment ($M = 2.18$, $SD = 0.37$). This difference could be demonstrated as significant ($t(9) = 7.51$, $p < 0.001$, $d = 2.37$). Since the p value is less than $\alpha = 0.05$, we reject the null hypothesis H_{5_0} in favor of the alternative hypothesis H_{5_1} .

4.4 Discussion

Time measurements of task type relocate As can be seen in Fig. 10a, the time required to complete relocate type tasks in the desktop environment strongly increases with higher number of task DOF. In contrast, the measured times in the VR environment (see Fig. 10b) increase only moderately. We believe that the massive increase in the desktop environment is due to the subdivision of the task into several sub-steps for operation using a 3D widget. The results in Table 2 indicate that 1D tasks in the VR environment need about triple of the time of the desktop environment. Users could perform 3D tasks twice as fast and 6D tasks in a quarter of the time necessary in the desktop environment. These are approximate values which may vary depending on the characteristics of the task (e.g., object size, distance and rotation angle). The very small p values and the high effect sizes (> 0.8) for the evaluable samples indicate small probabilities of error and strong effects (Cohen 1988). Outliers in the virtual environment can occur if finger tracking

Fig. 13 Difference in the time users needed to complete assembly tasks in the both environments. For an explanation of the box plot elements see caption of Fig. 10. *Indicates normal distribution based on Kolmogorov–Smirnov test, Lilliefors significance correction

Fig. 14 SUS scores per environment and the difference per user. For an explanation of the box plot elements see caption of Fig. 10

Table 2 Mean time required over all task with the same number of DOF to be changed

	Time VR (s)	Time desktop (s)	$\frac{\text{Time VR}}{\text{Time desktop}}$ (%)
1D	4.7	1.5	316.88
3D	7.4	15.3	48.41
6D	13.1	57.9	22.56

Table 3 Mean time required over all task with the same assembly

	Time VR (s)	Time desktop (s)	$\frac{\text{Time VR}}{\text{Time desktop}}$ (%)
Pipe	10.0	27.8	35.92
Engine	31.5	62.34	50.58

problems appear or the user inadvertently pushes objects out of their current range (e.g., toward the floor plane). The values of the standard deviations are influenced, among other factors, by the fact that the subjects have different levels of experience, different levels of personal ability and that the solution path for the individual tasks is not strictly prescribed.

Comparison to studies from the literature for task type relocate In a user study Schultheis et al (2012) compare mouse operation via 3D widgets with interaction via a 3D controller. The time required to complete the task is 3.4 times slower when operating via the computer mouse than via the 3D controller. Since it can be assumed that the described task in their study requires work steps with more than two degrees of freedom, the results can be roughly compared to the results from this section. The results of the study presented in this section show that, on average, the tasks with more than two degrees of freedom were completed about 3.6 times slower in the desktop environment than with the finger interaction interface in the immersive environment. This magnitude is consistent with the efficiency benefits of interacting via a 3D controller in Schultheis et al (2012).

Time measurements of task type assemble As the values in in Table 3 indicate, the users could complete the same assembly tasks nearly twice fast or faster in the VR environment than in the desktop environment. We strongly believe that this is a result of automatic constraint recognition based on efficient manipulation through natural finger interaction. Implementing the detection algorithms into the mouse and keyboard operated desktop environment would not be reasonable. Again, the small p values and the high effect sizes for the evaluable samples indicate small probabilities of error and strong effects (Cohen 1988).

Comparison to studies from the literature for task type assemble In the literature there is a user study (Toma et al 2012) which compares assembly modeling in a CAD application via computer mouse and keyboard with the arrangement of parts in a CAVE environment via hand interaction.

It must be noted that in the immersive environment, the correct arrangement of the components is detected using predefined matrices with position and orientation, and no constraints are inserted. Thereby, the assembly modeling in the desktop environment takes about three times as long as the arrangement of the components in the correct spatial position. Although the studies are not directly comparable, the ratio of time measurements roughly corresponds to the results from the user study presented in this section.

SUS questionnaire On average, the VR environment performs better than the desktop environment in every single question. Particularly with regard to the complexity and the learnability of the operation on the single environments, there were strong differences in favor of our newly developed method. The detailed scores of the single questions in the SUS questionnaire is reported in “Appendix 1.”

Limitations of the study With regard to the results of the SUS survey, it must be noted that the majority of the subjects have no or moderate experience with VR. In contrast, only one study participant has no experience with CAD applications. The novel experience of spatial representation of virtual content and natural interaction could cause the virtual environment to be viewed as advanced and interesting. This could lead to an overly positive attitude toward the VR environment. Unfortunately, this influence is difficult to determine. As a solution, only VR-experienced subjects could be tested, which unfortunately are not available in the conducted study.

In the context of semi-automatic constraint detection, it would be of interest to see if this would provide an efficiency advantage in the desktop environment. However, since the manipulation of virtual objects with more than three degrees of freedom is significantly slower compared to natural finger interaction, this advantage can be considered negligible.

5 Conclusions

A novel natural finger interaction method is presented in conjunction with an algorithm to detect constraints to facilitate CAD assembly modeling in an immersive virtual environment. The natural finger interaction approach follows the philosophy to recreate the human everyday experience in interacting with physical object. Moreover, the automatic constraint detection algorithm uses the geometry and the spatial location of rotationally symmetric components to detect constraints without any predefinitions. We compare a VR environment with natural finger interaction and a mouse & keyboard operated desktop environment in a user study. The results show the superiority of our approach in relocation tasks with higher DOF and in CAD assembly modeling tasks. Therefore, the main contribution of this study is the finding that for user interfaces in the context of Computer-Aided Design, there is much potential in natural interaction approaches.

Future research will focus on extending the concept combining the intuitive interaction approach and the constraint detection method to cover additional assembly cases such as drilling hole patterns or distance constraints. In addition, the suitability of natural interaction for creating preliminary designs will be investigated.

Appendix 1: SUS questionnaire and responses

The following SUS questions were answered by the participant subsequently to each test environment trial. The answers are given in a 5-point Likert scale, ranging from strongly disagree to strongly agree (Fig. 15).

Fig. 15 SUS answers for the single questions per environment. The upper bar of each question represents the VR environment, while the lower bar belongs to the desktop environment. Please note that every

second question is formulated negatively, so that less agreement with the questions or small values imply a higher ranking

Appendix 2: Tables

See Tables 4, 5, 6 and 7.

Table 4 Results for the Kolmogorov–Smirnov test for normal distribution of the differences of the relocate tasks at the 5% significance level, Lilliefors significance correction

Name	Statistic	df	Significance
1D-t1	0.249	10	0.079
1D-t2	0.135	10	0.200*
3D-t1	0.237	10	0.120
3D-t2	0.420	10	< 0.001
6D-t1	0.241	10	0.105
6D-t2	0.150	10	0.200*

*This is a lower bound of the true significance

Table 5 Results for the Kolmogorov–Smirnov test for normal distribution of the differences of the assemble tasks at the 5% significance level, Lilliefors significance correction

Name	Statistic	df	Significance
A1-trial1	0.336	10	0.002
A1-trial2	0.169	10	0.200*
A2-trial1	0.165	10	0.200*
A2-trial2	0.171	10	0.200*

*This is a lower bound of the true significance

Table 6 Results for the one-tailed paired-sample *t* test of the differences of the relocate tasks

Name	Mean	SD	<i>T</i>	df	Sigma
1D-t1	2.402	1.819	4.177	9	0.001
1D-t2*	4.045	4.523	2.829	9	0.010
3D-t1*	− 9.569	10.336	− 2.928	9	0.008
3D-t2*,**	− 6.239	13.967	− 1.413	9	0.096
6D-t1	− 43.050	22.391	− 6.080	9	< 0.001
6D-t2	− 46.595	15.638	− 9.423	9	< 0.001

*Sample has outliers. **Normal distribution not indicateable

Table 7 Results for the one-tailed paired-sample *t* test of the differences of the assemble tasks

Name	Mean	SD	<i>T</i>	df	Sigma
A1-trial1*,**	− 19.083	17.017	− 3.546	9	0.003
A1-trial2	− 16.512	9.039	− 5.777	9	< 0.001
A2-trial1*	− 33.014	11.808	− 8.841	9	< 0.001
A2-trial2	− 28.606	18.427	− 4.909	9	< 0.001

*Sample has outliers. **Normal distribution not indicateable

Supplementary Information The online version contains supplementary material available at <https://doi.org/10.1007/s10055-021-00543-0>.

Acknowledgements We gratefully acknowledge the support of NVIDIA Corporation with the donation of a high-performance GPU used for this research.

Funding Open Access funding enabled and organized by Projekt DEAL. This research did not receive any grant from funding agencies in the public, commercial or not-for-profit sectors.

Data availability The datasets generated for this study are available on request to the corresponding author.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

- Alkemade R, Verbeek FJ, Lukosch SG (2017) On the efficiency of a VR hand gesture-based interface for 3d object manipulations in conceptual design. *Int J Hum Comput Interact* 33(11):882–901. <https://doi.org/10.1080/10447318.2017.1296074>
- Bérard F, Ip J, Benovoy M, El-Shimy D, Blum JR, Cooperstock JR (2009) Did “minority report” get it wrong? Superiority of the mouse over 3d input devices in a 3d placement task. In: Gross T, Gulliksen J, Kotzé P, Oestreicher L, Palanque P, Prates RO, Winckler M (eds) *Human–computer interaction—INTERACT 2009*. Springer, Heidelberg, pp 400–414
- Borst C, Indugula A (2006) A spring model for whole-hand virtual grasping. *Presence* 15(1):47–61. <https://doi.org/10.1162/pres.2006.15.1.47>
- Brooke J et al (1996) SUS—a quick and dirty usability scale. *Usability Eval Ind* 189(194):4–7
- Cohen J (1988) *Statistical power analysis for the behavioral sciences*. Lawrence Erlbaum Associates, Mahwah
- Dankert T, Heil D, Pfeiffer T (2013) Stereo vision and acuity tests within a virtual reality set-up. In: *Virtuelle und Erweiterte Realität-10. Workshop der GI-Fachgruppe VR/AR*
- Fa M, Fernando T, Dew PM (1993) Direct 3d manipulation techniques for interactive constraint-based solid modelling. *Comput Graph Forum* 12(3):237–248
- Fechter M, Wartzack S (2017) Natural finger interaction for CAD assembly modeling. In: *ASME 2017 international design engineering technical conferences and computers and information in engineering conference*, ASME, volume 1: 37th computers and

- information in engineering conference, p V001T02A041. <https://doi.org/10.1115/DETC2017-67555>
- Hix D, Hartson HR (1993) Developing user interfaces: ensuring usability through product & process. Wiley, New York
- Houde S (1992) Iterative design of an interface for easy 3-d direct manipulation. In: Proceedings of the SIGCHI conference on human factors in computing systems, ACM, New York, NY, USA, CHI'92, pp 135–142. <https://doi.org/10.1145/142750.142772>
- Jacobs J, Froehlich B (2011) A soft hand model for physically-based manipulation of virtual objects. In: 2011 IEEE virtual reality conference (VR), pp 11–18. <https://doi.org/10.1109/VR.2011.5759430>
- Jacobs J, Stengel M, Froehlich B (2012) A generalized god-object method for plausible finger-based interactions in virtual environments. In: IEEE symposium on 3D user interfaces (3DUI), pp 43–51
- Jayaram S, Jayaram U, Wang Y, Tirumali H, Lyons K, Hart P (1999) Vade: a virtual assembly design environment. IEEE Comput Graphics Appl 19(6):44–50
- Joshi A, Kale S, Chandel S, Pal DK (2015) Likert scale: explored and explained. Curr J Appl Sci Technolpp 396–403
- LaViola J, Kruijff E, Bowman D, McMahan R, Poupyrev I (2017) 3D user interfaces: theory and practice. Addison-Wesley usability and HCI series. Addison-Wesley, Boston
- Liu O, Rakita D, Mutlu B, Gleicher M (2017) Understanding human–robot interaction in virtual reality. In: 2017 26th IEEE international symposium on robot and human interactive communication (RO-MAN), IEEE, pp 751–757
- Marcelino L, Murray N, Fernando T (2003) A constraint manager to support virtual maintainability. Comput Graph 27(1):19–26
- Mendes D, Caputo F, Giachetti A, Ferreira A, Jorge J (2018) A survey on 3d virtual object manipulation: from the desktop to immersive virtual environments. In: Computer graphics forum, Wiley Online Library
- Moehring M, Froehlich B (2005) Pseudo-physical interaction with a virtual car interior in immersive environments. In: Proceedings of the 11th eurographics conference on virtual environments, pp 181–189
- Ortega M, Redon S, Coquillart S (2007) A six degree-of-freedom god-object method for haptic display of rigid bodies with surface properties. IEEE Trans Vis Comput Graph 13(3):458–469
- Öztuna D, Elhan AH, Tüccar E (2006) Investigation of four different normality tests in terms of type 1 error rate and power under different distributions. Turk J Med Sci 36(3):171–176
- Pascarella C, Lazoi M, Papadia G, Galli V, Piarulli L (2018) CAD-VR integration as a tool for industrial assembly processes validation: a practical application. In: De Paolis LT, Bourdot P (eds) Augmented reality, virtual reality, and computer graphics. Springer International Publishing, Cham, pp 435–450
- Periverzov F, Ilieş H (2015) Ids: the intent driven selection method for natural user interfaces. In: IEEE symposium on 3D user interfaces, pp 121–128
- Sauro J, Lewis JR (2016) Quantifying the user experience: practical statistics for user research. Morgan Kaufmann, Burlington
- Schultheis U, Jerald J, Toledo F, Yoganandan A, Mlyniec P (2012) Comparison of a two-handed interface to a wand interface and a mouse interface for fundamental 3d tasks. In: 2012 IEEE symposium on 3D user interfaces (3DUI). IEEE, pp 117–124
- Seth A (2007) Combining physical constraints with geometric constraint-based modeling for virtual assembly
- Seth A, Vance JM, Oliver JH (2011) Virtual reality for assembly methods prototyping: a review. Virtual Real 15(1):5–20
- Song J, Cho S, Baek SY, Lee K, Bang H (2014) GaFinC: gaze and finger control interface for 3d model manipulation in CAD application. Comput Aided Des 46:239–245. <https://doi.org/10.1016/j.cad.2013.08.039>
- Talvas A, Marchal M, Lécuyer A (2013) The god-finger method for improving 3d interaction with virtual objects through simulation of contact area. In: 2013 IEEE symposium on 3D user interfaces (3DUI). IEEE, pp 111–114
- Toma MI, Gîrbacia F, Antonya C (2012) A comparative evaluation of human interaction for design and assembly of 3d cad models in desktop and immersive environments. IJIDeM 6(3):179–193
- Ullmann T, Sauer J (2000) Intuitive virtual grasping for non haptic environments. In: Proceedings the eighth Pacific conference on computer graphics and applications, pp 373–457
- Valentini PP (2009) Interactive virtual assembling in augmented reality. IJIDeM 3(2):109–119. <https://doi.org/10.1007/s12008-009-0064-x>
- Valentini PP (2018) Natural interface for interactive virtual assembly in augmented reality using leap motion controller. IJIDeM 12(4):1157–1165. <https://doi.org/10.1007/s12008-018-0461-0>
- van Dam A (1997) Post-wimp user interfaces. Commun ACM 40(2):63–67. <https://doi.org/10.1145/253671.253708>
- Wang Z, Ong S, Nee A (2013) Augmented reality aided interactive manual assembly design. Int J Adv Manuf Technol 69(5–8):1311–1321
- Wang X, Ong SK, Nee AY (2016) A comprehensive survey of augmented reality assembly research. Adv Manuf 4(1):1–22
- Weiser M (1991) The computer for the 21st century. Sci Am 265(3):94–105
- Yao YX, Xia PJ, Liu JS, Li JG (2006) A pragmatic system to support interactive assembly planning and training in an immersive virtual environment (i-vapts). Int J Adv Manuf Technol 30(9):959–967. <https://doi.org/10.1007/s00170-005-0069-y>
- Zilles CB, Salisbury JK (1995) A constraint-based god-object method for haptic display. In: Proceedings 1995 IEEE/RSJ international conference on intelligent robots and systems. Human robot interaction and cooperative robots, vol 3, pp 146–151

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.