Video-Annotated Augmented Reality Assembly Tutorials

Masahiro Yamaguchi[1,2], Shohei Mori[2], Peter Mohr[2,3], Markus Tatzgern[4], Ana Stanescu[2], Hideo Saito[1], Denis Kalkofen[2]

[1]Keio University, [2]Graz University of Technology, [3]VRVis GmbH, [4]Salzburg University of Applied Sciences

{yamaguchi | saito}@hvrl.ics.keio.ac.jp, {mori | mohr | ana.stanescu | kalkofen}@icg.tugraz.at, markus.tatzgern@fh-salzburg.ac.at

ABSTRACT

We present a system for generating and visualizing interactive 3D Augmented Reality tutorials based on 2D video input, which allows viewpoint control at runtime. Inspired by assembly planning, we analyze the input video using a 3D CAD model of the object to determine an assembly graph that encodes blocking relationships between parts. Using an assembly graph enables us to detect assembly steps that are otherwise difficult to extract from the video, and generally improves object detection and tracking by providing prior knowledge about movable parts. To avoid information loss, we combine the 3D animation with relevant parts of the 2D video so that we can show detailed manipulations and tool usage that cannot be easily extracted from the video. To further support user orientation, we visually align the 3D animation with the real-world object by using texture information from the input video. We developed a presentation system that uses commonly available hardware to make our results accessible for home use and demonstrate the effectiveness of our approach by comparing it to traditional video tutorials.

Author Keywords

Augmented reality; video label; retargeting; assembly tutorial.

CCS Concepts

•Human-centered computing → Mixed / augmented reality; Graphical user interfaces; Human computer interaction (HCI); Mobile computing;

INTRODUCTION

Assembly and disassembly procedures are recurring activities in today's industrialized society. Popular examples include the assembly of new furniture and the maintenance of household appliances. Publicly available knowledge databases, such as YouTube[1] or Instructables[2], provide information on how to build or maintain complex objects. Hence, such tasks are no longer limited to occupations requiring specialized training.

[1] www.youtube.com

[2] www.instructables.com

UIST '20, October 20–23, 2020, Virtual Event, USA

ACM ISBN 978-1-4503-7514-6/20/10 ...$15.00.

https://doi.org/10.1145/3379337.3415819

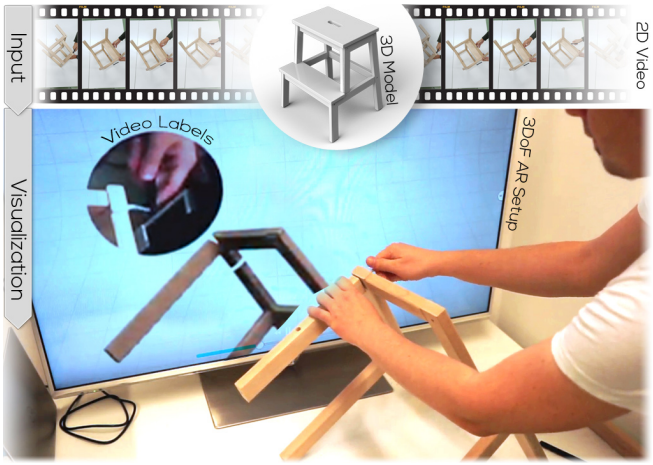

Figure 1. Video annotated assembly tutorial. Our system converts an assembly video tutorial into a 3D animation of a video textured 3D model and annotates it with the corresponding video sequence. These video labels provide missing details, such as tools and complex motions that are required to perform the actions.

Traditionally, assembly and disassembly procedures are communicated using textual descriptions that are often combined with a series of images for illustrating motions [34]. As complex manipulations are difficult to visualize in a series of images, illustrations are often complemented with video instructions [37]. Such video tutorials are widely available on the Internet and an abundant source of information.

Following video tutorials may be challenging, when the user's current viewpoint is not aligned with the one in the video [6]. Interactive 3D tutorials utilize object tracking to overcome the issue of mismatching viewpoints [19] by presenting 3D instructions that are aligned with the user's current view of the object. This enables users to perceive both, the real-world object and the associated instructions, from a single viewpoint. However, creating 3D animations for this kind of interactive tutorials is often difficult and, thus, quickly becomes time-consuming and expensive.

In order to support the authoring process of such interactive 3D tutorials, researchers have investigated *video retargeting* approaches that create these tutorials from existing 2D video tutorials [38, 12]. Video retargeting heavily relies on Computer Vision (CV) techniques to detect and track objects in the video [37, 12, 55] and consequently generate the required instructional 3D animations. However, video tutorials often present cases that cannot be handled gracefully by the utilized CV algorithms, thereby restricting the extracted instructions to the capabilities of the object detector that analyzes the video.

For instance, existing algorithms commonly fail when parts are occluded or have a small footprint in the 2D video. In such situations, the extracted instructions may only represent a subset of the complete assembly procedure. Due to these shortcomings, many retargeting approaches are only demonstrated on assemblies which consists of large detectable parts.

In this paper, we propose a novel approach for retargeting assembly video tutorials, which allows us to obtain a complete and detailed instructional 3D tutorial that also encompasses small and occluded parts. The extracted instructions are presented in an Augmented Reality (AR) "magic mirror" setup [15], as shown in Figure 1. Users work in front of a display showing the instructions overlaid onto a copy of the current work piece. The workpiece is tracked to infer orientation changes so that the viewpoint of the instruction is aligned with the one of the user.

We base our instruction extraction method on the principle of assembly-by-disassembly [11], a common technique in assembly planning [26]. Assembly-by-disassembly determines the assembly steps by disassembling the object of interest and reverting the calculated sequence. We follow a similar approach and parse the input video in reverse, starting with the completely assembled object and analyzing its disassembly. Therefore, by reverting the determined removal sequence, we retrieve the actual assembly steps that we use to generate our instruction visualization. This simple but effective idea allows tracking the removal of parts instead of having to detect the presence of new parts, which provides significant benefits to object detection and tracking. A major advantage is that we can use direct observations in the input video for object tracking, whereas previous approaches had to rely on object detection using synthetic 3D renderings [55].

Our approach extracts complete assembly instructions including manipulations of visually undetectable parts and presents them as animated 3D instructions. We achieve this by combining CV object tracking with queries of an assembly graph [27] of the target object that encodes blocking relationships between parts and allows us to identify small or occluded parts in the video. We enhance the presentation of 3D instructions by annotating the object with video sequences from the input video tutorial to preserve details, such as complex assembly motions, tool usage or non-rigid object deformations. To make interactive 3D AR tutorials widely available to users, we also aim for a practical setup that uses commonly available hardware such as laptops and TVs for presenting assembly instructions. In addition, we utilize a typical smartphone for tracking the assembly pose, to align the viewpoint shown in the AR mirror with that of the user.

Our system significantly adds to the state-of-the-art of AR tutorials, as we are the first to automatically generate comprehensible and complete 3D visualizations from video tutorials containing difficult-to-detect parts and complex object manipulations. In summary, we make the following contributions.

- We introduce a novel approach for extracting complete 3D assembly instructions from a video tutorial utilizing the object's assembly graph to infer otherwise undetectable assembly steps.
- We introduce a novel visualization technique for assembly instructions combining 3D animations with video information to encompass all information from input video tutorials. Video annotations provide information on otherwise undetectable manipulations.
- We introduce a practical approach for an AR tutorial setup using commonly available hardware by utilizing a smartphone for object tracking and a monitor as AR mirror.
- We demonstrate the completeness of our extraction and visualization approach by performing a user study comparing against typical video tutorials.

Our approach relies on model-based tracking of parts in a video frame. Hence, it relies on video data of certain quality, e.g. with large enough depth of field and little motion blur.

RELATED WORK

Due to their spatial anchoring, assembly instructions are ideally suited for AR applications. In comparison to traditional media, AR systems have been shown to reduce mental workload, improve task performance [52] and localization [22]. Users following AR instructions not only benefit from an improved spatial understanding, but also from comprehensible instruction visualization [25].

Advances in 3D pose estimation and tracking [44, 58] have opened up new possibilities for scene understanding, and enable the design of sophisticated AR assembly tutorial systems. With this motivation in mind, we introduce an end-to-end approach to automatically generate AR tutorials from input videos and apply them to real-world assembly applications.

Video-based Tutorials

Video-based tutorials have not only been proposed for desktop applications [43], but also for AR tutorials that display 2D videos overlaid over the AR view to communicate human motion [28], general task workflows related to real-world objects [42, 10], or assembly instructions [18]. Goto et al. [18] overlay 2D videos prepared in advance over the task work space using homographies. Petersen et al. [42] overlay automatically segmented input video directly onto the relevant real-world objects. However, placing videos on top of objects typically causes occlusions. Damen et al. [10] avoid this problem by placing segmented 2D videos next to the instruction area.

While these approaches show the general applicability of registered videos to visualize instructions in AR, their effectiveness has not been extensively evaluated. Only Goto et al. [18] report a user study indicating benefits of video overlays, even though occlusions remain problematic. Furthermore, these approaches are not applicable to complex assembly structures, as investigated in this paper.

Augmented Reality Tutorials

AR tutorials register instructions in 3D space directly in a user's view, as demonstrated by seminal work in the area of maintenance and assembly [9, 14]. Early AR systems, such as Reiners et al. [45] for doorlock assembly or Zauner et al. [60]

for furniture assembly, require parts to be fitted with fiducial markers for identification, which limited their usability.

Recent approaches avoid fiducial markers by employing novel CV techniques. For instance, Wu et al. [59] developed an AR instruction system with markerless tracking based on RGBD input. Similarly, Wang et al. [55] provided real-time feedback during an object assembly procedure by using a probabilistic model to compute the most likely assembly configuration captured via an RGB input video. Gupta et al. [19] introduce an assembly tutorial for Duplo structures. The system provides an authoring and a guidance mode, but is limited by assumptions about the structure of the Duplo components, such as the orthogonality of the assembly direction.

In contrast to previous work, our approach is not limited to certain object types or classes, but determines complete assembly instructions of objects, including their small and hidden parts, by utilizing an assembly graph of the analyzed object. The system implements a "magic mirror" setup [15], that avoids the need of specialized display and tracking hardware such as an AR head mounted display (HMD).

Authoring Tutorials

AR assembly tutorials are traditionally realized using rule-based systems [14] and scripting languages [8, 29]. However, assembly instructions can also be inferred from a 3D CAD model directly [36, 33, 1, 27], for instance, by calculating blocking relationships and creating an assembly graph containing information about the assembly sequence of all parts. We integrate assembly graphs into our system to support the extraction of assembly sequences from input videos and to identify small or occluded parts.

More recent approaches extract assembly instructions from existing sources, for instance, by analyzing written manuals [37, 48, 47] or video tutorials that demonstrate the procedure. Gupta et al. [19] investigated the creation of assembly tutorials for Duplo blocks by allowing a user to demonstrate assembly instructions via a semi-automatic authoring system. Our approach is related to this category of demonstration-based authoring and uses video tutorials as input. Using video as a source expands the applicability of our approach due to the wide availability of such content in online platforms. In a recent work, Mohr et al. [38] presented a similar system that extracts AR tutorials for tools with surface contact. Similarly, our method also extracts instructions from unmodified input videos, but we broaden the scope to support general instructions for the assembly of complex objects.

Visualizing Tutorials

Tutorials often utilize animated or video instructions that allow users to follow procedures when solving a task. These animations are usually segmented into distinct steps, so that users can work along at their own pace [18, 19]. The cognitive load theory (CLT) provides an explanation for the benefits of animated instructions. It considers the influence of instructional design and the cognitive architecture on information processing [51]. In the CLT, information processing depends on the intrinsic nature of the material that must be comprehended (intrinsic cognitive load) and the presentation of the material itself (extraneous cognitive load). Both intrinsic and extraneous cognitive load strain the limited working memory [35] of observers, which affects understanding of instructions. While intrinsic cognitive load cannot be changed, extraneous cognitive load is reduced by choosing an appropriate instruction design. A meta-review of Höffler and Leutner [24] revealed that animated and video instructions outperformed static images. The effect was stronger for instructions that related to the acquisition of procedural-motor skills, such as assembly tasks. Furthermore, findings of Ayers et al. [2] and Wong et al. [57] indicate that animations are especially suited for instructions that require observers to follow human movement. While Höffler and Leutner [24] focus on the benefits of continuous animations, animations have been found to be more effective and reduce extraneous cognitive load when presented in logical segments [5, 49] and when viewpoints of the instruction and the viewer are aligned [17].

Heiser et al. [21] investigated the design of effective assembly instructions, which have been successfully applied to automatically create step-by-step instructions [1]. They identified action diagrams depicting this type of assembly instructions as preferred representation. An single action diagram shows the attachment of a major part to an assembly, including the required smaller parts, such as fasteners. Furthermore, occlusions should be avoided, which may require viewpoint changes in the instructional visualization. In the initial orientation phase of users, a realistic depiction of the assembly instructions would be beneficial. Höffler and Leutner [24] also determined that realistic animations were more effective than CAD-style animations.

Our system has the goal to reduce extraneous cognitive load by automatically creating effective assembly instructions from input videos. Therefore, we adopt the design principles suggested by CLT research to create effective assembly instructions. Instructions are segmented into steps, so that, in each step, only a single major part and its fasteners are attached. Using segmented 3D animations allows users to follow instructions at their own pace. We allow free viewpoint changes to avoid occlusions, improve part visibility and enable users to align the viewpoint of the 3D instructions with their own viewpoint. To improve realism, inpainting techniques apply the texture of the input video to the CAD parts. Additional details are provided by registering the original instructional videos in 3D. We visualize the video viewpoint relative to the 3D assembly using a 3D view frustum indicating the view direction of recorded video. Such viewpoint visualizations support the spatial orientation of users [53] and communicate viewpoint locations [13].

DETERMINING THE ASSEMBLY SEQUENCE

In this section, we provide an overview of our system for generating a free-viewpoint animation. Input to our system is a video tutorial and a 3D model of the object of interest (Figure 2a). The 3D model must be structured such that individually movable parts can be distinguished. Our system animates each of the parts following the assembly procedure that is demonstrated in the input video.

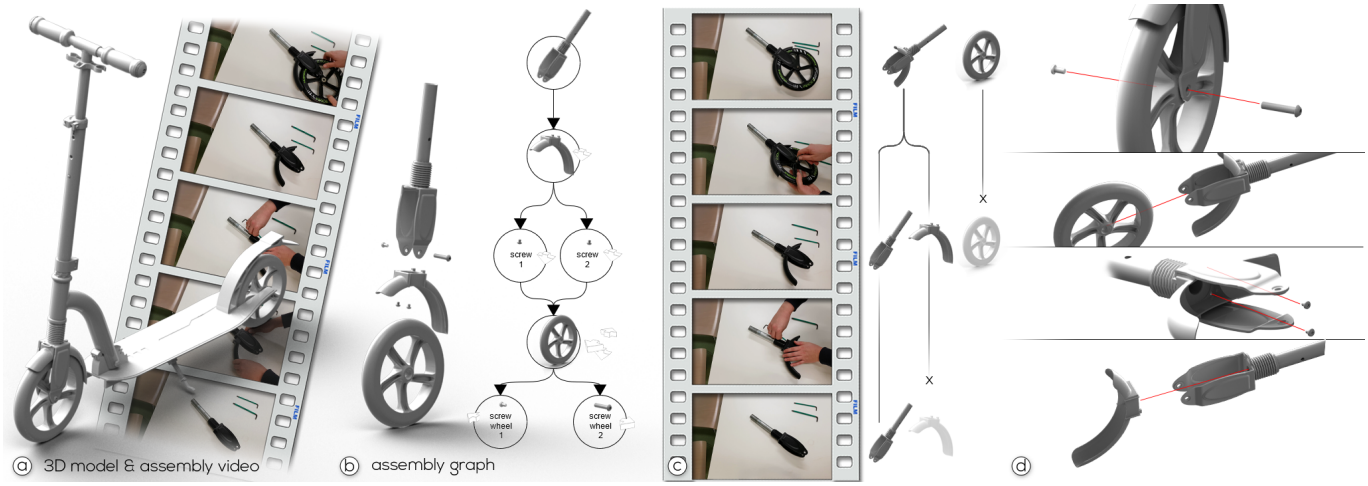

Figure 2. Animating the 3D assembly model. (a) Given a 3D model of the object of interest, (b) we construct its assembly graph, (c) and we identify object parts which are large enough to be detected. (d) After detecting the removal of large parts, we use the blocking information in the assembly graph to infer the removal of small parts, which cannot be detected otherwise. We use the detected order of removal to generate a 3D animation that encodes the removal actions as depicted in the video.

Step 1 - Computing the Assembly Graph. We use the assembly graph to support the detection of the removal of assembly parts in the video. To identify unblocked parts that can be removed, we compute blocking relations between all parts of the 3D model. We store the blocking information in a directed acyclic assembly graph (Figure 2b), which represents the parts of the object as nodes, and blocking relations between parts as edges. We compute the assembly graph using the approach presented by Kerbl et al. [27], which detects blocking relations by analyzing the 3D model of the object.

Step 2 - Extracting the Assembly Sequence. We compute the assembly sequence from the input video by analyzing the video frames in reverse order, thus, observing the disassembly of the object and detecting the removal of object parts. Hence, inverting the observed disassembly sequence provides us with the actual shown assembly sequence. For each disassembly step, we use the assembly graph to identify all currently removable parts in the 3D model (Figure 2c). Then, we compute the six degrees of freedom (6DOF) pose of the object in order to project all of its parts into the corresponding video frame. These projections result in 2D footprints of each part, which we use to initialize a segmentation-based tracker. To detect a part's removal, we project the current removable parts while tracking the assembly and calculate the probability of each part's existence at the location of its expected 2D footprint. Once a part has been removed, the probability in the corresponding 2D footprint decreases significantly, providing a reliable measure for removal detection. When a part is detected as removed, we add it to the assembly sequence and remove it from the 3D model and the assembly graph.

This process iterates, until the assembly graph is empty. However, since we rely on CV methods, we can only monitor the presence of object parts that provide a sufficiently large, detectable footprint. Therefore, in each frame, we first test the detectability of all removable parts by measuring the size of their 2D footprint in image space. We mark undetectable parts as potentially removed in the assembly graph, so that blocked, but detectable parts become removable. Note that we add an undetectable part to the assembly sequence only after we detect the removal of a larger part that it blocks.

Since multiple undetectable parts may block a single detectable part, we must determine the correct removal order of undetectable parts as shown in the video. We infer their removal order by segmenting the user's hands and tools from the video and analyzing the positions relative to assembly parts, assuming they are removed sequentially using hands and tools.

Step 3 - Generating Step-by-Step Animations. Once a complete disassembly sequence has been computed, we generate visual instructions to communicate the assembly actions. We generate commonly used assembly visualizations such as motion lines and 3D animations [40] by translating each part in the direction where no other parts block the removal. When multiple directions are possible, we follow the approach of Li et al. [31] and select the direction that allows the part to escape the assembly bounding box the fastest (Figure 2d).

Model Registration

Our approach is based on the registration and tracking of an existing 3D model of the assembly in the video. We aim at a wide range of assembly tutorials. Therefore, we focus on common RGB video formats. We make use of an image-based tracker [44] that tracks the 2D segmentation and the 6DOF pose of a 3D object simultaneously.

Initially, using GrabCut [46], we generate an object mask M_N of the fully assembled object $\mathbf{O}$ at the last frame N of the input video I_N (Figure 3b). Subsequently, we derive the initial 6DOF pose of $\mathbf{O}$ from the mask M_N. For pose estimation, we train a convolutional neural network (CNN) with renderings of $\mathbf{O}$ (Figure 3c). In order to remove any misleading color information, we convert both, the renderings for training and the segmentation, at run time, to binary image masks. Note that this is similar to the approach of Wu et al. [58]. However,

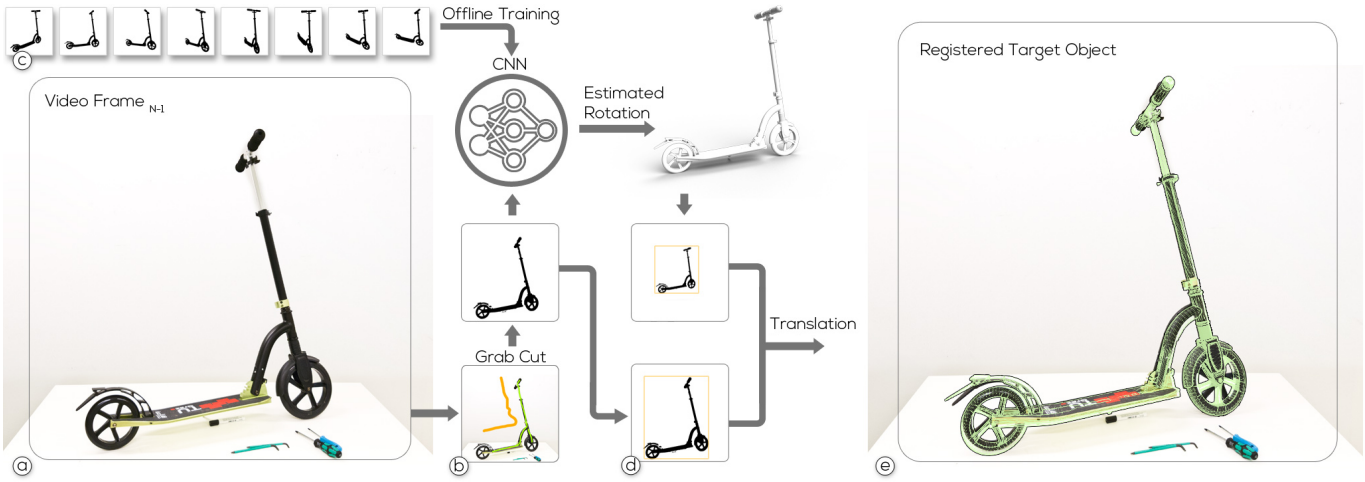

Figure 3. Registration. (a) We register the assembly to the last frame of the input video. (b) Our approach uses a segmentation of the object, (c) and a neural network to derive the camera orientation from the segmentation. (d) Using the estimated orientation we render the object of interest, and subsequently detect its bounding rectangle. We also compute the bounding rectangle of the segmentation in the input frame, and we derive the distance of the camera from the ratio of the rectangles sizes. (e) We use the resulting translation and rotation to register the assembly.

in contrast to Wu et al., we estimate the translation separately, using only the bounding rectangle of the mask M_N. We can detect the translation separately since we can rely on a very precise mask M_N generated by GrabCut. Hence, we focus the network on the estimation of the rotation only, which significantly reduces the size of the network and the amount of data required for training. Our 3DOF pose was trained with approximately 10% of the training data required for state-of-the-art 6DOF pose estimation techniques [58].

Rotation. We generate the training data of the network by rendering the object's 3D model from various camera positions on its bounding sphere (Figure 3c). We crop each training image using the bounding rectangle that encloses the rendering and subsequently resize it to 224×224 pixels to match the size of the CNN input structure. The separation of rotation and translation allows us to set up a small network. Our network consists of only five convolution-and-pooling layers with Rectified Linear Unit (ReLU) activation and one fully connected layer. The first convolution layer uses a 5×5 kernel, while other convolution layers use a 3×3 kernel size. After every convolution, a max-pooling operation down-samples the image to the half of its size. The loss function makes use of a standard mean squared error in quaternion space.

Translation. For the full pose, we calculate the translation that aligns the segmented object mask M_N with the mask M_R of the rendered object. We generate M_R by rendering $\mathbf{O}$, using the estimated rotation R_N and an initial distance d from the center of $\mathbf{O}$. We choose an initial distance d so that the rendering contains the entire object. Both masks represent the projection of the object under the same rotation and thus show very similar shapes, while differing in size. To align both masks, we compute the ratio of the longest edges of both bounding rectangles, and adjust the translation accordingly (Figure 3d).

Evaluation. Our pose estimator requires only a small set of images for training and generates results that are sufficiently accurate to initialize incremental tracking. We demonstrate this on the LINEMOD dataset [23], from which we used 10,000 images for training and 1,000 for testing. We used ten objects from the dataset. We excluded symmetric objects that make our pose estimation ambiguous. We measured the overlap of the ground truth masks and estimated masks of a 3D model. The overlap and the standard deviation were $81\%(\pm 12)$, which is substantially larger than what has been reported for initializing image based pose tracking [44].

Removable and Detectable Part Selection

After we register the 3D model of the object to the video, we start monitoring its removable parts. We identify removable parts by searching the assembly graph for nodes without outgoing edges. The resulting list contains all parts that can potentially be removed from the assembly at the current point in time. However, some parts contained in this list may not be detectable in the camera view due to their small size in image space. Therefore, we determine the detectability of the parts under observation by computing their footprint in image space. We remove parts from the list which are too small to be detected (we consider a part too small if its footprint is smaller than 25×25 pixels). Afterwards, we update the list of removable parts by searching for parts that have now become removable. We repeat this process, until no footprint in the list of removable parts is considered being too small. We repeatedly perform testing the size of the parts under consideration and we re-categorize them when their detectability state changes. In the rest of this section, we discuss the tracking and detection of removable parts that have a sufficiently large footprint. Smaller parts are inferred from the removal detection of these larger trackable parts.

Tracking

After we register the assembly and determine its initial 6DOF pose, our system tracks the rigid 3D assembly using the approach of Prisacariu and Reid [44]. However, since the observed object is disassembled in a video tutorial, its 3D shape changes over time. This will eventually lead to tracking failure.

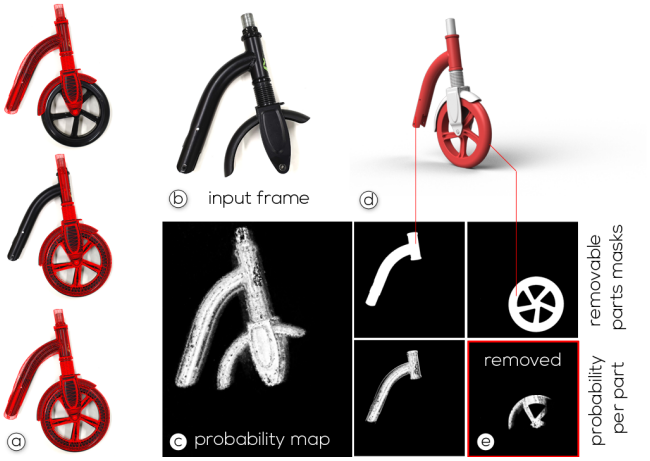

Figure 4. Tracking and removal detection. We generate masks for large object that we use for tracking by configuring each tracker with all but the removable part, in addition to the entire assembly. (a) The example shows the resulting three trackers for two removable parts. (b) From the input frame, (c) we compute the probability of all tracker and we use the one with the highest value, i.e. the one without the removed part, for updating the camera pose. (d) We use the camera pose to project removable parts (illustrated in red), (e) and we compute the probability of their existence in corresponding footprints.

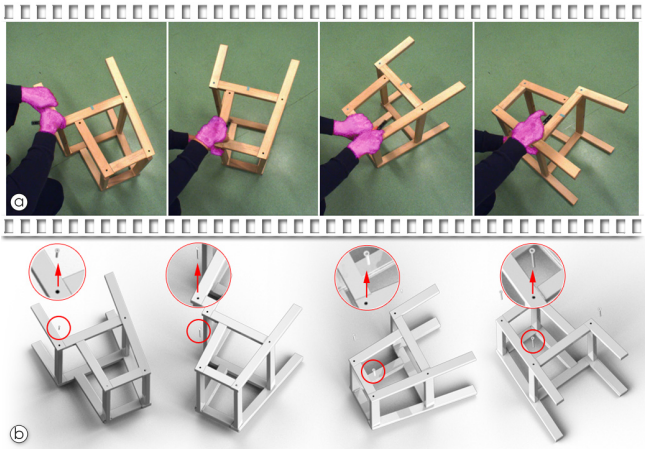

Figure 5. Sequence refinement. If multiple small parts block a large part, their order of removal is unknown. To refine the removal order for such parts, we additionally search for the users hands and tools in the corresponding video sequence. (a) We use scaled hand masks to compute the overlap projection of small parts, and (b) we infer order of their removal from the order in which the hands overlap the parts. This example shows four screws that have been ordered based on the overlap of the hand masks with the projected footprint of the screws.

To improve tracking stability, we make use of the previously determined removable parts list. We use multiple trackers, one for each removable part, and we use the tracker with the highest confidence for updating the pose of the entire assembly.

Since the stability of each tracker improves with the number of pixels covered by the observed object, we aim at maximizing the footprint of the tracked parts. Therefore, we track the current assembly without a removable part, instead of tracking a single removable part in isolation. We apply this approach for each removable part of a frame, thus, running multiple object tracker instances in parallel, each ignoring a different removable part (Figure 4a). One of these trackers will not be affected by the removal of a part, because the tracked assembly configuration matches the real configuration after the part has been removed, thereby improving its tracking stability. To further improve stability, we cope with hands occluding the assembly by performing hand and tool segmentation using the approach of Li et al. [30] and excluding such pixels from the input frame of the trackers.

Among all trackers, we identify the one that we finally use for updating the pose of the entire assembly by comparing their confidence. We derive the confidence from their energy terms, after pose optimization has been completed for the current frame. We follow the formulation of the energy term by Prisacariu and Reid [44], which uses a level set embedding of the pose, and the likelihood models P_f and P_b. The likelihood models use color histograms, for calculating the probability of a pixel x to fall within the foreground region f and the background b. Note that the smaller the energy gets, the more confident the estimated pose becomes.

Removal Detection

We infer the removal of a part from the probability of its 2D projection to cover the part in the video frame. Therefore, we project each removable part O_k to image space, using the current pose of the assembly, and generate the footprints $\Omega(O_k)$ (Figure 4d). To compensate for hand occlusions, we remove pixels from $\Omega(O_k)$ that intersect with the hand segmentation $\Omega(H)$, i.e., $\Omega'(O_k) \equiv \Omega(O_k) \setminus \Omega(H)$. We skip a frame for a part O_k if more than n percent of the pixels in $\Omega(O_k)$ are occluded by the user's hands. We calculate the probability $P(O_k)$ over all pixels in a footprint of a removable part (Equation 1).

$$P(O_k) = 1/|\Omega'(O_k)| \sum_{\mathbf{x} \in \Omega'(O_k)} (P_f(\mathbf{x}) - P_b(\mathbf{x})). \qquad (1)$$

We consider a part to be removed, if its probability $P(O_k)$ falls below a threshold $t_{removed}$. To compute $P(O_k)$ we need the current foreground likelihood model P_f and background likelihood models P_b for a pixel at $\mathbf{x}$ in the region $\Omega'(O_k)$. Since the likelihood models change over time, we keep updating them for a tracker if its confidence value is higher than a threshold t_{track}. The results in this paper were generated using $t_{track} = 0.70$, $t_{removed} = 0.15$, and $n = 50$. Our current implementation processes input videos at approximately 3 frames per second (fps) on a system using an Intel Core i7-7700 CPU, 32 GB RAM and an NVIDIA GeForce GTX 1060 GPU. Optimizations such as multi-threading can further improve the performance of the multi-objects tracking and the removal detection.

Sequence Refinement

After detecting the removal of a larger part O_{large}, we search for the smaller parts that may have not been detected. Since only unblocked parts can be removed from the assembly, we assume that any part that blocked O_{large} must have been removed before. Thus, we use the assembly graph to identify all parts O_{small} that block O_{large} and that have not been removed yet.

Since multiple parts O_{small} may block O_{large}, we need to determine the removal order of all O_{small} as shown in the video. We derive this from analyzing the user's hands movement and the tools they are using. We assume that the user is removing a part using a tool or with her hands directly. This allows us to infer the removal sequence from the locations of the user's hands and tools relative to the parts of the object. We compute the hand and tool segmentation [30], and the footprint of all O_{small} in the frames between the one where we detect the removal of the current large part O_{large} and the one where we detected the previous large part. If no previous part exist, we start from the last frame of the input video.

We compute the footprint of a part O_{small} by projecting its corresponding 3D model into the video frame using the estimated camera pose. Then, we compute the overlap between a projected part and the hand and tools masks and assume that the user is removing the part O_{small} with the largest overlap. If two or more parts O_{small} overlap with the same mask, we group them and show their instruction simultaneously, instead of consecutively. Figure 5 illustrates our approach.

VISUALIZATION

Generating virtual 3D animations from the input video allows us to create free-viewpoint instruction visualizations. We use an interactive AR mirror to align the user's viewpoint of the physical object with the virtual 3D instruction view shown in the mirror. Our setup requires a monitor and a 3D object tracker attached to the physical assembly to update the virtual camera during user interaction. Instead of relying on CV object tracking that is typically error-prone, we make use of an additional tracking device. Inspired by the approach of Mohr et al. [39], we use an off-the-shelf smartphone, which we attach to the assembly to enable its tracking (see Figure 6).

Texturing

To minimize cognitive load, we match the appearance of the 3D model to the one in the video by using pixel colors from the video and applying them to the model texture. We create a texture atlas for each part and we fill it by projecting the pixel from a video frame onto the object using the estimated camera pose. Since the user's hands often occlude the object during interaction, we exclude hand pixels before we use a frame. Video tutorials usually show the object of interest from multiple viewpoints. Since we track the camera pose, we can project video frames onto the 3D assembly to fill the texture atlas. We only project video frames where an object has been identified. Some parts may not be visible, in which case we additionally use an inpainting solution to estimate missing pixel colors from those within the texture atlas [4] (Figure 7a).

Video Label Extraction

A video label is a cropped video excerpt focusing on the interaction with a part. The label is especially useful to communicate actions that are otherwise challenging to visualize in a 3D animation, such as tool usage or non-rigid object deformations. In such cases, the video label is able to present direct visual explanations of interactions. In order to focus the content of the video label, we remove frames which do not show any actions, and we crop the remaining frames to the area that shows the assembly part in addition to the the user's hands and tools that have been used to manipulate it. In particular, we search the video for hands and tools using the feature descriptor presented by Li et al. [30] that we already used for sequence refinement. We use the result to produce a pixel mask, which we refer to as the action mask and, consequently, remove frames where no action masks appear.

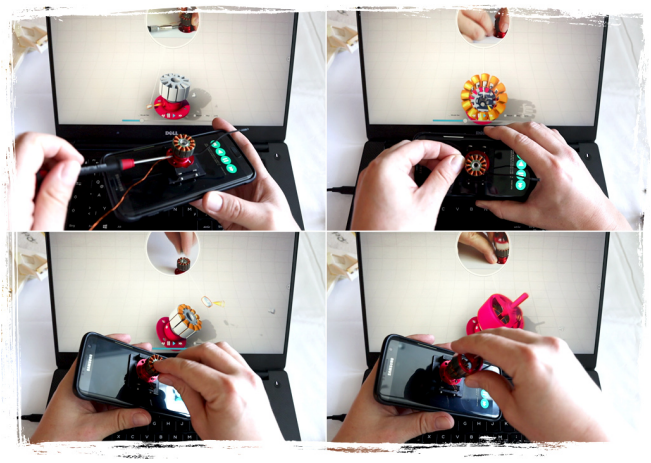

Figure 6. AR visualization. We can utilize a common smartphone for tracking the object-of-interest, and utilize widely available displays as AR mirror. In this example, the user follows a video-annotated AR tutorial that has been generated from a video tutorial showing the assembly of a motor on a laptop screen. The tutorial includes fastening small screws and coiling of a wire. We present these operations as video annotations, as they cannot easily be extracted with current CV techniques.

To focus on the content of a video label, we calculate the center $\mathbf{c}_\text{l}$ and the radius r_l of the most important area. First, we detect the contours of parts, hands, and tools [50]. When a tool and a hand, or two hands have been detected we combine their masks and treat them as one. Subsequently, we compute the minimal enclosing circles on each contour. We obtain the centers of the part and the hand regions as $\mathbf{c}_\text{p}$ and $\mathbf{c}_\text{h}$, and the radii of the circles, which enclose the masks as r_p and r_h, respectively. Finally, we calculate the video label center $\mathbf{c}_\text{l}$ and its radius r_l using Equation 2 and 3. To stabilize video label regions over time, we use rolling average filtering for $\mathbf{c}_\text{l}$ and compute the median of r_l throughout the video sequence. See Figure 7b,c for an example. Our current implementation calculates video labels at approximately 37 fps (see above for PC details).

$$\mathbf{c}_\text{l} = (r_\text{l} - r_\text{p}) \frac{\mathbf{c}_\text{h} - \mathbf{c}_\text{p}}{||\mathbf{c}_\text{h} - \mathbf{c}_\text{p}||} + \mathbf{c}_\text{p}, \tag{2}$$

$$r_\text{l} = (||\mathbf{c}_\text{h} - \mathbf{c}_\text{p}|| + r_\text{h} + r_\text{p})/2. \tag{3}$$

Video Label Placement

We visualize labels as annotations of the object of interest. They consist of a circular representation of the video and a thin anchor line which connects the video with the object.

Dynamic placement. We initially placed labels on a plane in 3D space [54], at the point where the corresponding video frame was captured from (Figure 8a). However, as a 3D video label disappears when the user explores the object from the

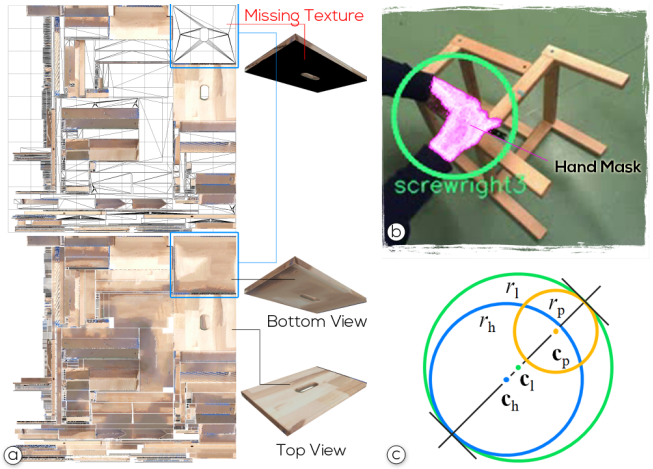

Figure 7. Texture filling and label extraction. (a) To visually align the 3D animation with the video label we fill the texture of the 3D model by projecting video pixels from the tracked camera onto the object. (b) We focus the information in video labels by cropping the input video to the circle that encloses the user's hand, tools, and the part that is assembled. (c) We use the center and radii of the detected part (yellow) and hand-and-tool (blue) to focus and stabilize the video label region (green).

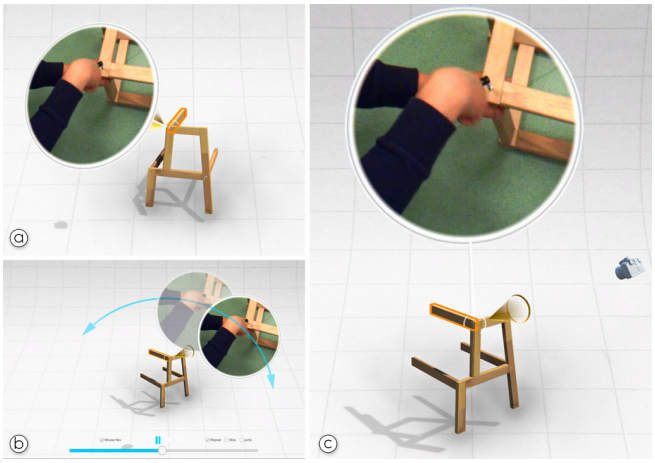

Figure 8. Label placement. (a) We place the video on a plane in 3D space using the estimated pose of the video camera. In addition, we render a 3D cone to indicate a predefined field of view and to connect the label with the object. (b) We show the 3D label only when the virtual camera is in the field of view. For all other camera positions we project the center of the label to the scaled bounding circle of the object. This causes the label to move on a circle around the object when the camera is outside the viewing cone. (c) Based on user feedback, we refined the placement strategy. To avoid label motion and occlusions, we place the label on the object's up vector, at a fixed distance outside its bounding circle. We place it close enough to observe both, the 3D animation and the video. To indicate the camera pose we add a 3D camera icon.

side or the back, we present the video on a 2D billboard, which we place on the bounding circle of the object, when the 3D label leaves a predefined viewing cone (55°) (Figure 8b).

Thus, depending on the current viewpoint, we switch automatically between the 3D and the billboard placement. However, based on user feedback in a pilot study with three participants, we decided to update the placement strategy. All users raised the issue of overlap of the video and the object, when the virtual camera is aligned with the one in the video in the 3D presentation. This eventually obstructs their view of the object. Even when videos show more information than the 3D animation, all users mentioned they felt insecure about the instruction when the 3D animation was not visible anymore. In addition, all of them commented on the distracting motion that is introduced by the moving label. This is in line with the findings of Madsen et al. [32], who study the impact of label motion on user performance. We initially wrongly assumed that the 3D placement will allow anticipating label motions and thus, do not distract.

Static placement. To compensate for label motion and object occlusion, we refined the placement strategy. We first placed the label in the corner of the display (Figure 9) and disabled positional updates. In addition to video labels, we present a 3D cone visualization and a 3D camera icon, which indicate the 3D camera pose and the field of view that was used for capturing the video (Figure 8c shows a screenshot that includes the camera icon and the view cone).

While the static placement was well received in a second pilot with three users, the position was criticized as being too far away from the object. Users reported that the distance made it difficult to look at both, the 3D animation and the video, simultaneously. Therefore, we kept the static placement but moved the label center to the point where the object's up vector intersects its scaled bounding circle. Our final placement strategy avoids motion and occlusion, and the placement on the up vector enforces alignment between the object and the label, therefore bringing the whole content to the user's view.

EXPERIMENT

We validate the results of our approach to automatically create step-by-step AR instructions from an input video in a user study where participants followed these instructions to assemble an object. We compare the extracted AR instructions to the baseline method of using traditional video tutorials to investigate if our approach generates a complete set of instructions from the input video. The evaluation also allowed us to collect subjective feedback regarding the usability and cognitive load of the system. Our AR visualization follows established guidelines for optimal instruction design and, thus, uses step-by-step instructions, animations and free viewpoint changes to reduce a user's cognitive load. Due to these design guidelines, we expected the AR condition to be superior to a video tutorial in a typical assembly task.

Design. We designed a within-subject study with an independent variable "instruction method" with two conditions: video and AR. The video condition corresponded to a video tutorial without zoom effects or cuts. The video showed the assembly from a viewpoint defined by the video's author. As is typical with current online video tutorials, the video did not automatically pause playback between assembly steps. However, users could pause the video at any time. In AR, users worked with the presented AR mirror system. The system paused automatically at each segmented instructional step, and users could manually forward by pressing a button on a keyboard. Furthermore, users could freely change the viewpoint.

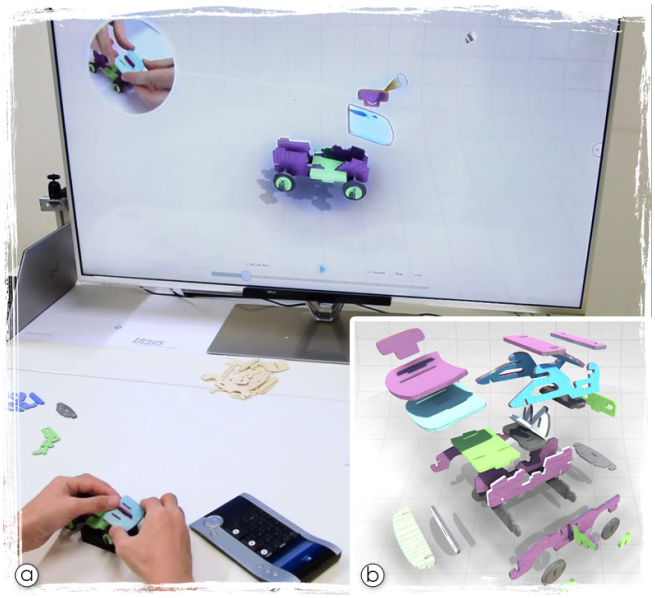

Figure 9. Apparatus. (a) In the study, we used a 50 inch screen in front of the user. (b) Exploded view of the assembly used in the study.

We measured task completion time (TCT), i.e., time to finish the assembly, total number of assembly errors, cognitive load as mental effort using the commonly used rating scale of Paas [41], usability with the System Usability Scale (SUS) [7], task load using the NASA TLX questionnaire [20], and overall preference. If not indicated otherwise, numerical values in the text of this Section are reported in the format "mean (sd)". For our study, 16 participants (2 female, $\overline{X}$ =27.8 (3.85) years) volunteered. On a scale from one to five (best), the mean of self-rated AR experience was 3 (sd=1.03, median=3), and the mean of the handicraft work experience was 2.93 (sd=0.99, median=3). 6.25% of participants rated AR experience as 1, 31.5% as 2, 18.75% as 3, 43.75% as 4. Hence, the majority of participants had at least intermediate AR experience. All participants except one had a computer science background. One participant had an engineering background and self-rated AR experience of 3, thus, also experience with AR.

Apparatus. Video and AR condition were displayed on a 50 inch screen. The screen was located in front of the assembly area. Participants were seated in front of the screen and followed the presented instructions. Users had to assemble a car model with 34 parts attached to a wireless IMU for tracking. In the video condition, they could interact with the video and navigate between assembly steps in the AR condition using a computer mouse and a keyboard with labeled buttons for video controls. Figure 9 shows the apparatus.

Procedure. After filling out an informed consent form and demographics questionnaire, participants familiarized themselves with the system using a trial assembly. To avoid learning effects, the object used in the trial task was different from the one used in the main task. If they were familiar with the interfaces of both methods, they started the task with one instructional method. Participants were instructed to be fast and accurate. After completing a condition, users filled in the mental effort, SUS and NASA TLX questionnaires and continued with the remaining instructional method. After completing the final task, the users filled out the preference questionnaire.

The starting order of the instruction method was counterbalanced. After assembling half of the car model, participants continued assembling the car using the other instruction method. We followed this procedure, because using the same toy car allows us to generate comparable quantitative data between both instruction methods. In our study, both halves of the car have been assembled using both methods an equal number of times in order to compensate for differences in assembly complexity. We calculate the overall mean of our measurements for each instruction method.

Hypotheses. We expected that the AR condition would outperform the video condition with respect to TCT (**H1**) and total number of errors (**H2**) due to the design of the system that takes into account research from instruction design and CLT. Consequently, we also expected to see better usability (SUS) (**H3**), less mental effort (**H4**), less task load (NASA TLX) (**H5**) and overall preference (**H6**) for the AR condition.

Results. The data was evaluated using a significance level of 0.05. As the data did not fulfill the normality requirements it was analyzed using Wilcoxon signed-rank tests; effect sizes are calculated as $r = \frac{Z}{\sqrt{N}}$, as proposed by Fritz et al. [16]. The analysis was performed using the statistics software *R*.

Wilcoxon signed-rank tests revealed statistically significant differences in mental effort (video 3.9 (1.9); AR 3.0 (1.0); Z=2.36, p<0.05, r=0.42), SUS (video 70.5 (14.3); AR 86.3 (8.2); Z=2.9, p<0.01, r=0.51) and the factor temporal demand of the NASA TLX (video 48.1 (24.3); AR 28.4 (22.1); Z=2.42, p<0.05, r=0.43). There were no significant differences in TCT (video 183 (43); AR 182 (51)), error (video 0.6 (1.5); AR 0.5 (0.8)) or raw NASA TLX (video 37 (18); AR 31 (13)).

We checked for differences in assembly complexity between the two halves of the car by calculating Wilcoxon signed-rank tests for each instruction method and assembly half. The tests did not reveal statistically significant differences.

Discussion. We did not find statistically significant differences in TCT and number of errors. Therefore, we reject H1 (TCT) and H2 (error). However, the subjective results clearly demonstrate the advantages of our AR system over traditional video tutorials. We accept H6 (preference), as 100% of the participants preferred the AR system over the traditional video tutorial. 62.5% of participants had a self-rated AR experience above 3. Therefore, this strong preference for the AR condition cannot be fully explained by a novelty effect, but is an indicator of the perceived superiority of the AR condition over video only. Based on the result of the statistical evaluation, we accept hypothesis H3 (SUS) and H4 (mental effort). The superior mean SUS score of 87 for our AR system indicates a usability rating above the average score of 70, that according to the analysis of Bangor et al. [3] translates into the adjective "excellent". Participants also had to invest significantly less mental effort in solving the tasks using our AR system.

These positive results for the AR condition are underlined by the qualitative feedback of the participants. As unsolicited

feedback in an unstructured after-study interview, 37.5% of the participants stated that the AR visualization was easy to understand and very clear. Furthermore, 50% of the participants positively remarked on the additional interactivity and viewpoint control in the AR condition that it allowed them to better understand the task and assembly shapes. 19% of the participants explicitly mentioned the video annotations and stated that they were especially useful to understand details of the instructions, when the 3D rendering was not sufficiently clear. These observations indicate that our algorithm for extracting assembly steps and the visualizations work as expected.

We observed that the video inpainting of the 3D parts shown in the instruction animation supported participants in identifying the corresponding real-world parts during the assembly task. In the interview, only few participants (12.5%) explicitly pointed out the advantages of visualizing assembly parts with the same texture as the real-world parts. However, throughout the assembly tasks, 37.5% of the participants mixed up two similarly colored parts, thereby making the same assembly errors. This clearly indicates that participants used the parts' visual appearance to identify the subsequent part and highlights the value of using inpaining for the 3D visualization.

While we did not find a statistically significant difference in the NASA TLX, we partially accept H5 as the test revealed a significant difference in temporal demand. This difference may be explained by the design of the instructions. 25% of the participants stated that they were less pressured by the AR condition due to the presentation as step-by-step instructions compared to the video condition that had to be paused manually. In the video condition, participants usually tried to work at the speed of the video. Hence, the video was subjectively found to be putting more pressure on the participants than the step-by-step instructions of the AR condition.

The difference of temporal demand between AR and video instructions may also explain the lack of a significant difference in TCT. According to our observations and the participants' feedback, in the video condition, participants tried to work at the speed of the video without pausing it, while they took their time in the AR condition and investigated the instructions from different viewpoints. Hence, participants sped up their interaction in the video condition, while they slowed down in the AR condition, thus, influencing the TCT. Furthermore, the lower mental effort for the AR condition indicates that understanding and following step-by-step AR instructions was easier than following the video tutorial, even though the latter could have also been paused on demand.

Overall, the evaluation showed that users were able to complete the tasks with both video and AR instructions. This indicates that the extracted instructions and generated visualizations of our system are complete so that users can follow the generated instructions without difficulty. One major advantage of the AR mirror approach lies in the interactive visualization of assembly steps most notably of small and occluded parts. In such cases, video tutorials require zooming in on small parts, or a change of camera angles, which leads to a mismatch between the user's view of the real-world object and the video's viewpoint. Videos are also often edited to cut between viewpoints, which requires additional mental rotations from the viewer. The interactive viewpoints of AR facilitate this reorientation as is indicated by the user feedback and the decreased mental effort. In addition, the 3D animation of the assembly instruction naturally avoids occlusions, e.g., by hands or tools. A future study should isolate the impact of these factors by controlling occlusions in the video and by controlling viewpoint discrepancies between video and the real-world object.

The usability of the video condition was rated as 70.5 (SUS), which corresponds to average usability and the adjective "good" [3]. This shows that the video condition was perceived positively despite its limitations. Usability could likely be improved by segmenting the video like in the AR condition. We noticed that participants rarely used the pause button of a video. This is in line with the results of Biard et al. [5], who additionally showed that users performed better when videos were automatically pausing between steps. Interestingly, participants did not remark upon such a feature for videos in our unstructured after-study interview but requested additional AR features such as more animation control (e.g., speed), or a preview of next steps. However, although segmented video instructions may improve in terms of temporal demand, participants would still be required to mentally align the video viewpoint with the real object and the video would suffer from occlusions (e.g., by the author's hands), both of which may contribute to increased mental effort. A future study should isolate the effect of instruction segmentation for videos on the task performance when compared to the AR mirror system.

CONCLUSION AND FUTURE WORK

We showed that our approach successfully extracts assembly instructions from video tutorials. Given only a video recording and the object's 3D model, we infer the object's assembly steps in their correct order. We take inspiration from the technique of assembly-by-disassembly and parse input videos from end to beginning to facilitate object extraction. Furthermore, we utilize the object's assembly graph to identify the manipulation of small or occluded parts. We evaluated the completeness of the extracted instructions in a user study that demonstrated that users were indeed able to follow the generated instructions and to complete the assembly tutorial.

In the following, we enumerate a series of design recommendations for visual instructions of retargeted videos. Generally, our AR mirror instruction visualization caused less mental effort and, thus, cognitive load, than a common video tutorial. This is a result of grounding our design in findings from current state-of-the-art: (1) we show assembly instructions segmented into step-by-step animations that are shown at a speed chosen by the user [5, 49], (2) we allow users to control the viewpoint of the instructions so that they are always aligned with the corresponding real-world object [17] and can be used to resolve occlusions or to reorient during the task [21]. We add to the current research, as we are the first to introduce (3) video annotated AR instructions combining free viewpoint navigation and detailed instruction visualization. Video annotations are essential to capture assembly steps that cannot easily be extracted with current CV algorithms (e.g.,

tool usage, complex assembly motion, non-rigid object deformations), while AR instructions enable aligning the viewpoint of the real-world assembly with the visualization. Furthermore, our user study clearly showed that color is a strong cue during users' visual search for the next assembly part [56]. This aspect needs further investigation in the context of video retargeting, as (4) video texturing is a useful tool to visually align the virtual 3D instructions with the real-world scene and, thus, facilitates visual search. However, a future system must consider visually similar assembly parts and provide automatic guidance to resolve similarities of color and/or shape that could lead to assembly errors.

In addition to generating and evaluating instructions, we presented a feasible implementation of a real-time AR mirror tutorial system that allows home users access to these instructions. Our system requires only commonly available hardware, such as a consumer desktop PC to display the visualisation, and a mobile phone that enables object tracking by attaching it to the object of interest. As future work, we plan to provide a web-based interface to allow content providers and consumers access to the video extraction software and the AR mirror system. This allows us to collect test data that follows the guidelines above to produce videos suitable for our approach for further analysis, and will in the long term push online tutorials to the next level.

Our approach is aimed at processing detailed assembly videos. Currently our system can process videos that show objects that are continuously manipulated in a series of video frames without interruptions. While cuts in the video can be handled by reinitializing the object tracking, stop-motion sequences or slideshows will require frequent reinitialization, which decreases usability. Furthermore, while our system can detect small parts, it relies on the trackability of larger parts. For optimal detection results, content providers should take care of appropriate lighting, improving contrast to the background and calibrating the recording camera. However, we believe that these simple guidelines are easy to follow by professional video content providers and can stimulate the generation of interactive 3D tutorials. Our system currently does not use audio information from the video. However, a combination of extracted video annotations and voice overlay could further enhance the presented information. The audio track could also be used to focus the automatic detection of assembly steps by recognizing audio related to certain objects or tasks.

Currently, our system uses 3D models of the assembled object, required for calculating an assembly graph. Such detailed models are often available in the do-it-yourself community (e.g., 3D printing), are provided by producers or may be crowdsourced. In the future, 3D models could also be inferred from the video itself. Many objects that require assembly are composed of standardized pieces. In such a case, once an object is detected in the video frame, its 3D model could be retrieved from an object database.

Overall, our work presents a novel method to make the generation of intuitive 3D assembly instructions feasible for providers of assembly video tutorials. By relying on commonly available hardware, our AR mirror system also lowers the threshold of accessing these 3D instructions, thereby making them available to a wide audience.

ACKNOWLEDGMENTS

This work was enabled by the Competence Center VRVis, the FFG (grant 859208 - Matahari) and the Austrian Science Fund grant P30694, and partly by JST CREST under Grant JPMJCR1683, Japan. VRVis is funded by BMVIT, BMWFW, Styria, SFG and Vienna Business Agency in the scope of COMET, Competence Centers for Excellent Technologies (854174), which is managed by FFG.

REFERENCES

[1] Maneesh Agrawala, Doantam Phan, Julie Heiser, John Haymaker, Jeff Klingner, Pat Hanrahan, and Barbara Tversky. 2003. Designing effective step-by-step assembly instructions. *ACM Transactions on Graphics* 22, 3 (July 2003), 828–837.

[2] Paul Ayres, Nadine Marcus, Christopher Chan, and Nixon Qian. 2009. Learning hand manipulative tasks: When instructional animations are superior to equivalent static representations. *Computers in Human Behavior* 25, 2 (March 2009), 348–353.

[3] Aaron Bangor, Philip Kortum, and James Miller. 2009. Determining What Individual SUS Scores Mean: Adding an Adjective Rating Scale. *J. Usability Studies* 4, 3 (May 2009), 114–123.

[4] Connelly Barnes, Eli Shechtman, Adam Finkelstein, and Dan B. Goldman. 2009. PatchMatch: A Randomized Correspondence Algorithm for Structural Image Editing. *ACM Transactions on Graphics* 28, 3 (2009).

[5] Nicolas Biard, Salomé Cojean, and Eric Jamet. 2018. Effects of segmentation and pacing on procedural learning by video. *Computers in Human Behavior* 89 (December 2018), 411–417.

[6] Paul Breedveld. 1997. Observation, Manipulation, and Eye-Hand Coordination Problems in Minimally Invasive Surgery. In *Proc. XVI European Annual Conference on Human Decision Making and Manual Control*. 9–11.

[7] John Brooke. 1996. SUS: A quick and dirty usability scale. *Usability evaluation in industry* 189, 194 (1996), 4–7.

[8] Andreas Butz. 1994. BETTY: Planning and Generating Animations for the Visualization of Movements and Spatial Relations.. In *Proc. of Advanced Visual Interfaces*. 53–58.

[9] Thomas P Caudell and David W Mizell. 1992. Augmented reality: An application of heads-up display technology to manual manufacturing processes. In *Proc. of Hawaii Int. Conf. on Syst. Sc.*, Vol. 2. IEEE, 659–669.

[10] Dima Damen, Teesid Leelasawassuk, Osian Haines, Andrew Calway, and Walterio Mayol-Cuevas. 2014. You-Do, I-Learn: Discovering Task Relevant Objects and their Modes of Interaction from Multi-User Egocentric Video. In *BMVC*. 3.

[11] L.S. Homem De Mello and Arthur C. Sanderson. 1991. A Correct and Complete Algorithm for the Generation of Mechanical Assembly Sequences. *IEEE Trans Robotics and Automation* 7, 2 (April 1991), 228–240.

[12] Daniel Eckhoff, Christian Sandor, Christian Lins, Ulrich Eck, Denis Kalkofen, and Andreas Hein. 2018. TutAR: augmented reality tutorials for hands-only procedures. In *ACM SIGGRAPH Int. Conf. on VR Continuum and its Applications in Industry*. 1–3.

[13] Steven Feiner. 2014. ParaFrustum : Visualization Techniques for Guiding a User to a Constrained Set of Viewing Positions and Orientations. *UIST* (2014), 331–340.

[14] Steven Feiner, Blair Macintyre, and Dorée Seligmann. 1993. Knowledge-based augmented reality. *Commun. ACM* 36 (July 1993), 53–62. Issue 7.

[15] Mark Fiala. 2007. Magic Mirror System with Hand-held and Wearable Augmentations. *2007 IEEE Virtual Reality Conference* (2007), 251–254.

[16] Catherine O. Fritz, Peter E. Morris, and Jennifer J. Richler. 2012. Effect size estimates: Current use, calculations, and interpretation. *Journal of Experimental Psychology: General* 141, 1 (2012), 2–18.

[17] T. B. Garland and Christopher A. Sanchez. 2013. Rotational perspective and learning procedural tasks from dynamic media. *Computers and Education* 69 (2013), 31–37.

[18] Michihiko Goto, Uematsu Yuko, Hideo Saito, Shuji Senda, and Akihiko Iketani. 2010. Task support system by displaying instructional video onto AR workspace. In *ISMAR*. 83–90.

[19] Ankit Gupta, Dieter Fox, Brian Curless, and Michael Cohen. 2012. DuploTrack: A Real-time System for Authoring and Guiding Duplo Block Assembly. In *UIST*. 389–402.

[20] Sandra G Hart and Lowell E Staveland. 1988. Development of NASA-TLX (Task Load Index): Results of empirical and theoretical research. *Advances in Psychology* 52 (1988), 139–183.

[21] Julie Heiser, Doantam Phan, Maneesh Agrawala, Barbara Tversky, and Pat Hanrahan. 2004. Identification and Validation of Cognitive Design Principles for Automated Generation of Assembly Instructions. In *Proc. of the Working Conference on Advanced Visual Interfaces*. ACM Press, New York, NY, USA, 311–319.

[22] Steven Henderson and Steven Feiner. 2011. Exploring the Benefits of Augmented Reality Documentation for Maintenance and Repair. *TVCG* 17, 10 (October 2011), 1355–1368.

[23] Stefan Hinterstoisser, Vincent Lepetit, Slobodan Ilic, Stefan Holzer, Gary Bradski, Kurt Konolige, and Nassir Navab. 2012. Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes. In *Asian Conference on Computer Vision*. Springer, 548–562.

[24] Tim N. Höffler and Detlev Leutner. 2007. Instructional animation versus static pictures: A meta-analysis. *Learning and Instruction* 17, 6 (December 2007), 722–738.

[25] Denis Kalkofen, Erick Mendez, and Dieter Schmalstieg. 2009a. Comprehensible Visualization for Augmented Reality. *IEEE TVCG* 15, 2 (2009), 193–204.

[26] Denis Kalkofen, Markus Tatzgern, and Dieter Schmalstieg. 2009b. Explosion Diagrams in Augmented Reality. In *VR'09*. 71–78.

[27] Bernhard Kerbl, Denis Kalkofen, Markus Steinberger, and Dieter Schmalstieg. 2015. Interactive Disassembly Planning for Complex Objects. In *Computer Graphics Forum*, Vol. 34. Wiley Online Library, 287–297.

[28] Tobias Langlotz, Mathäus Zingerle, Raphael Grasset, Hannes Kaufmann, and Gerhard Reitmayr. 2012. AR Record Replay: Situated Compositing of Video Content in Mobile Augmented Reality. In *OzCHI*. 318–326.

[29] Florian Ledermann and Dieter Schmalstieg. 2005. APRIL A High-Level Framework for Creating Augmented Reality Presentations. In *Proc. of IEEE Virtual Reality*. 187–194.

[30] Cheng Li and Kris M. Kitani. 2013. Pixel-level hand detection in ego-centric videos. In *IEEE CVPR*. 3570–3577.

[31] Wilmot Li, Maneesh Agrawala, Brian Curless, and David Salesin. 2008. Automated Generation of Interactive 3D Exploded View Diagrams. *ACM Transactions on Graphics* 27, 3, Article 101 (2008), 101:1–101:7 pages.

[32] Jacob Boesen Madsen, Markus Tatzgern, Claus B. Madsen, Dieter Schmalstieg, and Denis Kalkofen. 2016. Temporal Coherence Strategies for Augmented Reality Labeling. *IEEE TVCG* 22, 4 (2016), 1415–1423.

[33] Sotiris Makris, George Pintzos, Loukas Rentzos, and George Chryssolouris. 2013. Assembly support using AR technology based on automatic sequence generation. *CIRP Annals* 62, 1 (2013), 9–12.

[34] Paul Mijksenaar and Piet Westendorp. 1999. *Open Here. The Art of Instructional Design*. Thames & Hudson.

[35] George A. Miller. 1956. The magical number seven, plus or minus two: some limits on our capacity for processing information. *Psychological Review* 63, 2 (March 1956), 81–97.

[36] Niloy J. Mitra, Yong-Liang Yang, Dong-Ming Yan, Wilmot Li, and Maneesh Agrawala. 2010. Illustrating How Mechanical Assemblies Work. *ACM Transactions on Graphics* 29, 4, Article 58 (July 2010), 58:1–58:12 pages.

[37] Peter Mohr, Bernhard Kerbl, Michael Donoser, Dieter Schmalstieg, and Denis Kalkofen. 2015. Retargeting Technical Documentation to Augmented Reality. In *ACM CHI*. ACM, New York, NY, USA, 3337–3346.

[38] Peter Mohr, David Mandl, Markus Tatzgern, Eduardo E. Veas, Dieter Schmalstieg, and Denis Kalkofen. 2017. Retargeting Video Tutorials Showing Tools With Surface Contact to Augmented Reality. In *ACM CHI*. ACM, 6547–6558.

[39] Peter Mohr, Markus Tatzgern, Tobias Langlotz, Andreas Lang, Dieter Schmalstieg, and Denis Kalkofen. 2019. TrackCap: Enabling Smartphones for 3D Interaction on Mobile Head-Mounted Displays. In *ACM CHI*. 1–11.

[40] Marc Nienhaus and Jürgen Döllner. 2003. Dynamic glyphs - depicting dynamics in images of 3D scenes. In *Int. Conf. on Smart Graphics*. Springer, 102–111.

[41] Fred G. Paas. 1992. Training strategies for attaining transfer of problem solving skills in statistics : a cognitive-load approach. *Journal of Educational Psychology* 84, 4 (1992), 429–434.

[42] Nils Petersen and Didier Stricker. 2012. Learning task structure from video examples for workflow tracking and authoring. In *2012 IEEE ISMAR*. 237–246.

[43] Suporn Pongnumkul, Mira Dontcheva, Wilmot Li, Jue Wang, Lubomir Bourdev, Shai Avidan, and Michael F. Cohen. 2011. Pause-and-play: Automatically Linking Screencast Video Tutorials with Applications. In *ACM UIST*. 135–144.

[44] Victor A. Prisacariu and Ian D. Reid. 2012. PWP3D: Real-time segmentation and tracking of 3D objects. *Int. J. of Computer Vision* 98, 3 (2012), 335–354.

[45] Dirk Reiners, Didier Stricker, Gudrun Klinker, and Stefan Müller. 1999. Augmented reality for construction tasks: Doorlock assembly. In *Proc. of the International Workshop on AR: Placing artificial objects in real scenes: placing artificial objects in real scenes*. AK Peters, Ltd., 31–46.

[46] Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. 2004. Grabcut: Interactive foreground extraction using iterated graph cuts. In *ACM Transactions on Graphics*, Vol. 23. ACM, 309–314.

[47] Tianjia Shao, Dongping Li, Yuliang Rong, Changxi Zheng, and Kun Zhou. 2016. Dynamic furniture modeling through assembly instructions. *ACM Transactions on Graphics* 35, 6 (2016).

[48] Tianjia Shao, Wilmot Li, Kun Zhou, Weiwei Xu, Baining Guo, and Niloy J. Mitra. 2013. Interpreting concept sketches. *Transactions on Graphics* 32, 4 (2013).

[49] Ingrid A.E. Spanjers, Tamara Van Gog, Pieter Wouters, and Jeroen J.G. Van Merriënboer. 2012. Explaining the segmentation effect in learning from animations: The role of pausing and temporal cueing. *Computers and Education* 59, 2 (2012), 274–280.

[50] Satoshi Suzuki and Keiichi Abe. 1985. Topological structural analysis of digitized binary images by border following. *Computer Vision, Graphics, and Image Processing* 30, 1 (1985), 32–46.

[51] John Sweller, Jeroen J.G. Van Merrienboer, and Fred G.W.C. Paas. 1998. Cognitive Architecture and Instructional Design. *Educational Psychology Review* 10, 3 (1998), 251–296.

[52] Arthur Tang, Charles Owen, Frank Biocca, and Weimin Mou. 2003. Comparative effectiveness of augmented reality in object assembly. In *ACM CHI*. ACM, 73–80.

[53] Markus Tatzgern, Raphael Grasset, Eduardo Veas, Denis Kalkofen, Hartmut Seichter, and Dieter Schmalstieg. 2015. Exploring real world points of interest: Design and evaluation of object-centric exploration techniques for augmented reality. *Pervasive and Mobile Computing* 18 (2015), 55–70.

[54] Markus Tatzgern, Denis Kalkofen, Raphael Grasset, and Dieter Schmalstieg. 2014. Hedgehog Labeling: View Management Techniques for External Labels in 3D Space. In *Proc. of IEEE Virtual Reality (VR)*. 1–6.

[55] Bin Wang, Guofeng Wang, Andrei Sharf, Yangyan Li, Fan Zhong, Xueying Qin, Daniel CohenOr, and Baoquan Chen. 2018. Active Assembly Guidance with Online Video Parsing. In *IEEE VR*. IEEE, 459–466.

[56] Jeremy M. Wolfe. 1994. Guided Search 2.0 A revised model of visual search. *Psychonomic Bulletin & Review* 1 (1994), 202–238.

[57] Anna Wong, Nadine Marcus, Paul Ayres, Lee Smith, Graham A. Cooper, Fred Paas, and John Sweller. 2009. Instructional animations can be superior to statics when learning human motor skills. *Computers in Human Behavior* 25, 2 (March 2009), 339–347.

[58] Jimmy Wu, Bolei Zhou, Rebecca Russell, Vincent Kee, Syler Wagner, Mitchell Hebert, Antonio Torralba, and David MS Johnson. 2018. Real-time object pose estimation with pose interpreter networks. In *IEEE IROS*. IEEE, 6798–6805.

[59] Li-Chen Wu, I-Chen Lin, and Ming-Han Tsai. 2016. Augmented reality instruction for object assembly based on markerless tracking. In *ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*. ACM, 95–102.

[60] Jürgen Zauner, Michael Haller, Alexander Brandl, and Werner Hartmann. 2003. Authoring of a Mixed Reality Assembly Instructor for Hierarchical Structures. In *IEEE/ACM ISMAR*. 237–246.