Computers & Graphics 53 (2015) 82–91

Contents lists available at ScienceDirect

Computers & Graphics

journal homepage: www.elsevier.com/locate/cag

40 years of CG in Darmstadt

Cognitive Augmented Reality

Nils Petersen [a,c,*], Didier Stricker [a,b]

[a] Deutsches Forschungszentrum für Künstliche Intelligenz, Trippstadter Straße, 122, 67663 Kaiserslautern, Germany
[b] Department Augmented Vision, Kaiserslautern University, Gottlieb-Daimler-Straße, 67663 Kaiserslautern, Germany
[c] ioxp GmbH, Trippstadter Straße 122, 67663 Kaiserslautern, Germany

ARTICLE INFO

Article history:
Received 19 August 2015
Accepted 22 August 2015
Available online 10 September 2015

Keywords:
Computer vision
Augmented reality
Visualization

ABSTRACT

Although the concept of Augmented Reality (AR) has already been proposed more than 20 years ago, most AR-applications are still limited to simple visualization of virtual objects onto spatially limited scenes. Ideas such as the "augmented reality manual", showing step-by-step instructions to a user wearing a Head-Mounted Display (HMD), have been implemented, but the developed systems did not pass the barrier of demonstration prototypes. One major reason, beside remaining ergonomic and hardware limitations, consists of the large effort required for creating the content of such virtual instructions and for building models allowing accurate tracking. In this paper, we introduce the concept of Cognitive Augmented Reality, which radically revises existing approaches on AR-based assistance systems and proposes a fundamentally new paradigm exploiting prior visual observation and learning of a complete manipulative workflow. More precisely, we present in this paper a complete approach for creating augmented reality content for procedural tasks from video examples, and give then details about the presentation of such content at runtime. We show that this new approach, in spite of its very challenging aspects, is scalable, and valuable from a practicable point of view.

1. Introduction

There has been extensive work on procedural assistance using Augmented Reality (AR) since the early seminal contribution of Caudell and Mizell [1] promoted this use case for head-up displays, thereby coining the term itself. Various different industrial tasks have been since then considered, mostly assembly and maintenance [5–8,14] but also process control [9], inspection and quality control [10], picking tasks [11,12], and virtual discrepancy checks [13]. Augmented Reality assistance was also subject to several large scale research projects, such as the ARVIKA [2,3], followed by ARTESAS, AVILUS/AVILUS+ [4] all pioneer work initiated or co-initiated by the institutions "GRIS – Graphisch-Interaktive Systeme" of the Technical University and Fraunhofer Institute für Graphische Datenverarbeitung (Fraunhofer IGD) in Darmstadt, Germany. Beside many works on marker-based and marker-less tracking [24–26], there is a large body of literature on design guidelines for AR presentations and visualization [15–19], procedures [20,21], and visual communication [22]. Several groups have presented graphical tools for authoring [23,27,31,33], but again with the goal to merely simplify a manual content creation and get animated 2D or 3D graphics objects registered onto the real world.

A closer look to those systems shows that all those approaches still have two major drawbacks: (1) content creation is still a tedious process and requires expert knowledge and (2) the existing systems have little "intelligence" in the sense of awareness about the current state of the scene and the user's context. In particular the latter would be a prerequisite to properly shape the provided information to the user's needs regarding situation and level of skills. Information level is identical for all users, and scene/user-action analysis is absolutely missing.

On the other side, it has been long understood that automated systems aiming to assist or interact with human activity need to have a degree of understanding of human behavior in order to be effective [38]. Actions and responses need to align with our expectations and information needs to be presented in a manner, which reflects our own perception. What is less understood is how that understanding of behavior is to be obtained. Following this basic idea, we propose to consider and design AR system as an autonomous and intelligent system, relying on classical robotics perception-action loop, or, *e.g.*, in the sense of concepts similar to the parasitic humanoid [32].

In this paper we show that an effective approach for procedural workflows can be developed using visual observation only. An illustration of the simplified authoring and implementation process is shown Fig. 1. The main steps can be summarized as:

* Corresponding author.

http://dx.doi.org/10.1016/j.cag.2015.08.009
0097-8493/

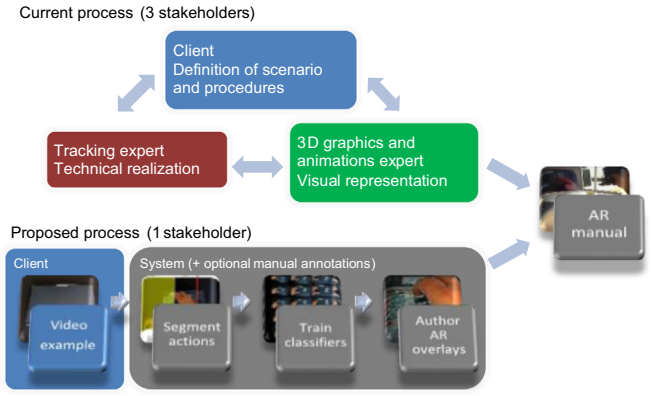

Fig. 1. Current implementation process (top) compared to the proposed process (bottom) where creation follows a machine learning-based approach.

Visual observation: A static or head-mounted camera captures the actions of an expert during the execution of a task.
Workflow segmentation, modeling, and content creation: The recorded video is then segmented into a set of single atomic actions via an unsupervised method. The single steps are analyzed to estimate required level of accuracy, repetitiveness, etc. Virtual information can be added and registered with the scene at the level of each single action what, together with the processed videos, provides a full representation of the demonstrated workflow [27,28]. The system selects descriptive minimal video snippets from the reference recording for instructing the user, based on the classification cues.
Context-aware presentation and workflow tracking: At run-time, the live video images of the head-mounted camera of the user are compared with the recorded reference workflow. This allows identifying and tracking the currently executed action. The selected snippets are then visualized onto the scene as half transparent overlays, both spatially and temporally registered with the user's actual progress. The system is hereby aware of the user's actions and accommodates displayed information for the user's needs.

2. Task assessment and modeling of the workflow

In this paragraph, we summarize the modeling aspects of workflow capturing. The details and evaluation of the single proposed methods can be found in previous literature [27–29,35], and in the Ph.D.-thesis [30] from which the following contributions (paragraphs 3–6) are extracted.

2.1. Unsupervised temporal segmentation

Since we aim for manual workflows viewed from the user's perspective, we generally have to deal with close-up images with frequent or even permanent occlusion of large parts of the observed image by the hands of the user. As we cannot assume observability of tools or interaction objects, a profound scene analysis is often infeasible as the already difficult object detection is additionally hindered. Instead, we propose a novel measure derived from image distance that evaluates image properties jointly without prior interpretation.

One of the main challenges of using image distance functions is that function results do not always coincide with the perceived similarity between two images. Therefore, it is not straightforward to formulate suitable compactness criteria based on this. We use whole-image dissimilarity functions between two arbitrary images of an ordered image sequence $\boldsymbol{S}$. In order to cope with lighting changes and small perspective deformations, this function is implemented using a region descriptor [34] and to further minimize crosstalk due to small camera movements, the function is explicitly made invariant to small affine image transforms.

The main premise is the following: while it is not decidable whether dissimilar images were produced by the same or different actions, it is relatively safe to assume that very similar pairs were produced by the same action. Whenever a frame cannot be safely assigned to an action, formally introduced as a dissimilarity threshold between carefully selected frame pairs, we call it a *novelty*. The segmentation is then based on minimizing the so-called shortest-path, *i.e.*, finding a set of frames with the least amount of novelties that connects between a hypothesized start and stop frame of the segment. For example, a scene with little visual change will produce a small shortest path as well as a scene with a very high but repetitive change. As soon as the visual change increases or alters in movement pattern, this will result in a strong lengthening of the shortest path, which we interpret as a segment boundary. After determining the segment boundary, the length of the shortest path in relation to its theoretical maximum is used to distinguish segments with user actions from static segments. A detailed description is given in [27]. From comparison of multiple reference recordings, we are able to identify work steps that need to be processed accurately based on the reoccurrence of single key postures and further to distinguish important steps from erratic motion based on the reoccurrence of longer trajectories.

2.2. Workflow modeling and tracking

After the unsupervised segmentation, we establish a tracking model of each work step both for camera tracking and for tracking the user. Creating this model is challenging as the environment is susceptible to change drastically due to user interaction, and camera motion may not provide sufficient translation to robustly estimate geometry.

We propose the *relevance plane transform*: a piecewise homographic transform that projects the given video material onto a series of distinct planar subsets of the scene. These subsets are selected by segmenting the largest planar image region that contains a given region of interest determined through estimating the focus of attention within each of the temporal segments. This results in a piecewise two-dimensional, spatiotemporal model of dynamic, changing environments. As this fits 2D coordinate frames into the workspace, it is viable to directly apply 2D descriptors or to anchor 2D information associated both spatially and temporally with the time-evolving 3D workspace, compare Fig. 2. In our experiments, we use this to sample 2D probability maps of the hand location and to extract instructive snippets within the recorded video from a moving camera. As it elegantly handles cases of incomplete observation, it does not require any prior knowledge of 3D scene geometry, explicitly copes with dynamic, changing environments, and works with uncalibrated cameras.

2.3. Explicit generalization of the training data using image-based rendering

As free-hand activities introduce a large amount of visual variance to the observation, a single recording is generally not sufficient for our non-parametric classification approach. To overcome the resulting problem, such as high user dependency, we need to generalize the model. Ideally, this is achieved through training the classifiers with additional reference performances. In order to make the system work reliably trained only with a single reference performance, we propose an image-based rendering approach to

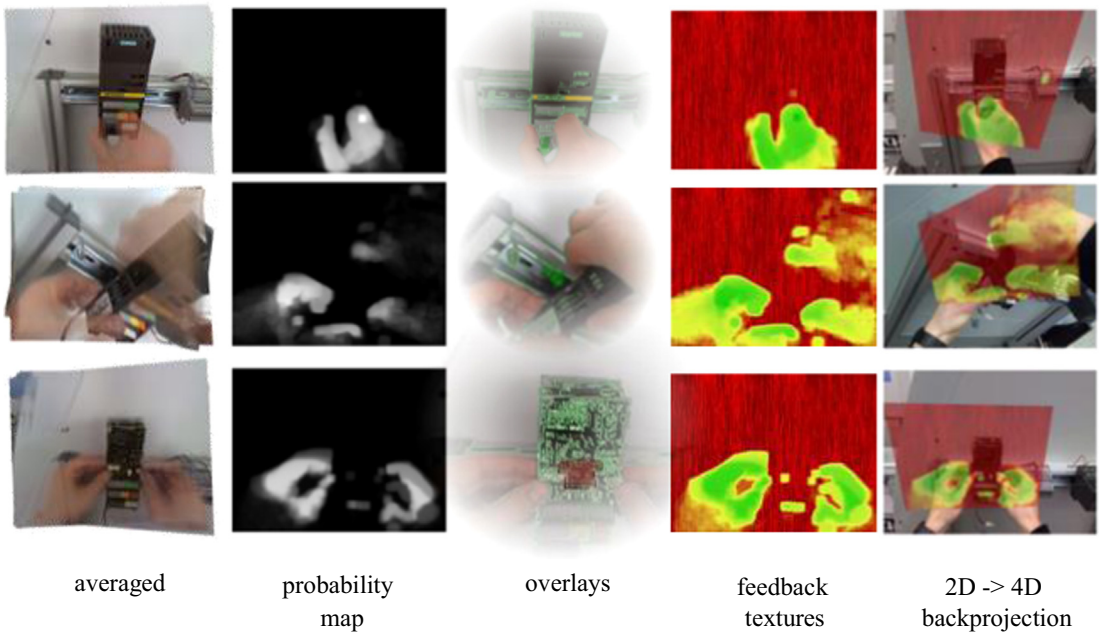

Fig. 2. The *relevance plane transform* allows the projection of a time-progressing 3D workspace into a piecewise 2D representation, which allows to elegantly apply 2D descriptors or anchor 2D overlays.

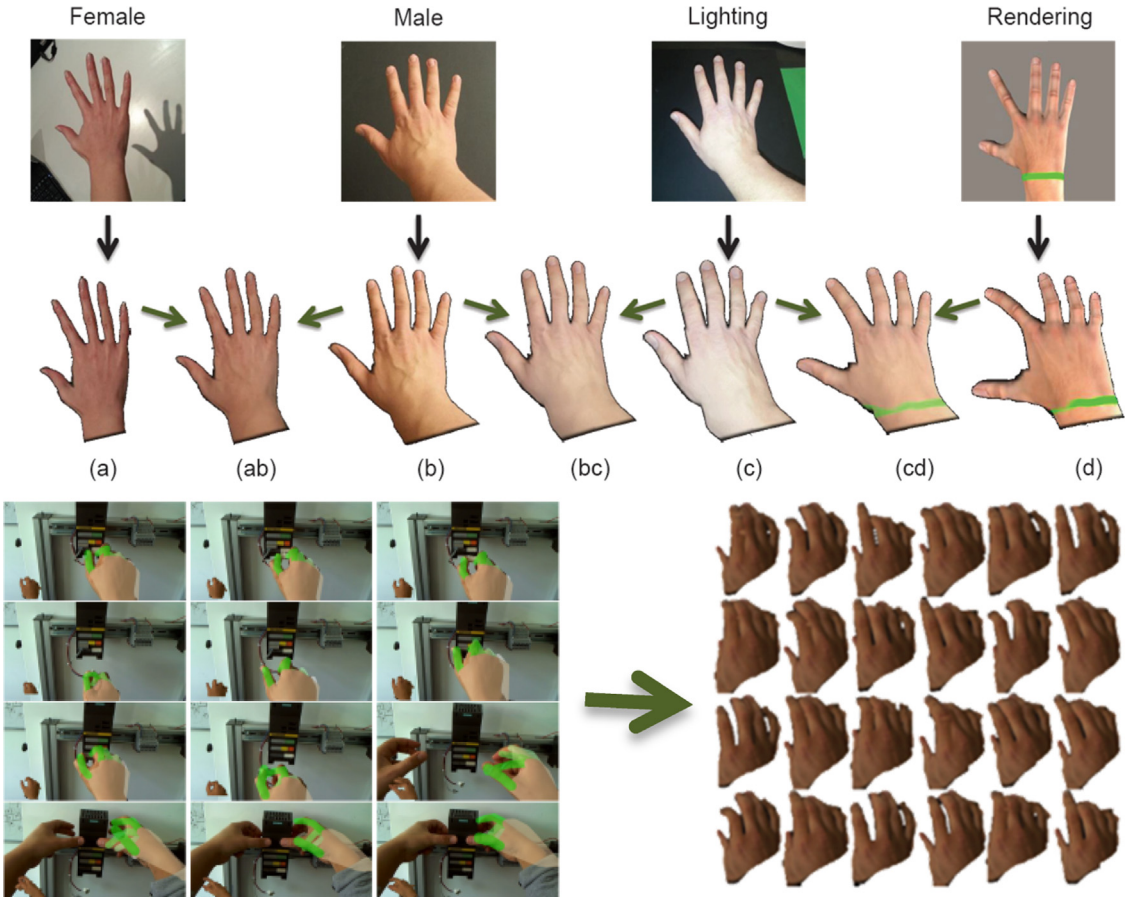

Fig. 3. Hand appearance model based on image-based rendering. During classifier training, postures of tracked frames (lower left) are synthesized in additional configurations to cope with minimal training data, starting with a single reference recording.

explicitly generalize the reference material through a model-guided approach. Fig. 3 shows an illustration and example images.

The method is computationally extremely lightweight to afford usage as hand appearance model for hand and finger tracking. We can show that this allows the formulation of a pixel-wise objective function that significantly outperforms the state of the art in monocular hand tracking with a generative model [35]. For the sake of robustness, we nevertheless only synthesize new postures

within a small vicinity of a recovered hand configuration. Thus, tracking errors do not strongly impact the training data. To accommodate for incomplete training data, we additionally deliberately selected all building blocks of the system to afford incremental learning. This is most notable in the proposed approaches to camera and hand tracking, both relying on nearest-neighbor search trees but also valid for the temporal workflow tracking. We carefully designed the entire approach to not crucially depend on a fragile high-level feature or pre-processing step. The core of our approach is based on very robust methods and all fragile building blocks are consequently incorporated in an extending but optional way: while their successful completion will improve the accuracy or the level of understanding, the working result is still usable without these steps. Examples of this are the incorporation of 3D hand tracking results and the approach to camera tracking that improves with the availability of point features but is not dependent on it. Fig. 4 shows a schematic of the data flow during the authoring process.

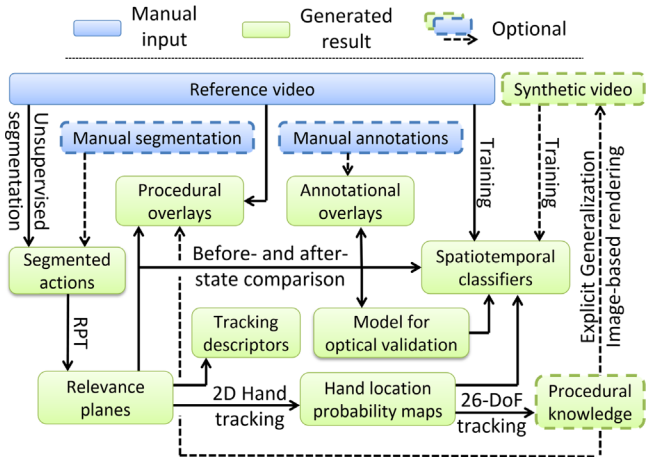

Fig. 4. Data flow diagram of the authoring process: after applying the *relevance plane transform* to the distinct segments, the workspace state before and after each user action is used to train the classifiers and to process the visual overlays. Further, the sequence is analyzed using hand and finger tracking to provide enactive feedback and to explicitly generalize the training data through image-based rendering.

3. Automatic overlay generation and visual representation

Building upon the acquired information, we are able to automatically generate a rich set of visual overlays. Part of these overlays are dynamically created during run-time based on the actions of the user. The following subsections explain the processing steps for the technical realization of each type.

We distinguish four types of visual overlays, see Fig. 5, for examples for each type:

Procedural overlays: Overlays that instruct on the given task by displaying an animated summary of the subsequent task.
Enactive feedback: Real-time enactive feedback during the execution of the task, indicating a correct conduction of the task by coloring the user's hand green, respectively red in case of wrong postures.
Annotational overlays: Overlays that emphasize certain workspace regions or further illustrate the current instruction, also using (manually entered) texts or graphics.
Assessing overlays: Overlays that indicate the outcome of an optical validation that compares the state of the workspace after a work step with the desired target state.

3.1. Procedural overlays

Since we use first-person view videos as input material, the straightforward way to generate an instructional animation of a work step is to use the corresponding parts of the reference sequence, directly, as illustrated in Fig. 6 Through the temporal segmentation of the reference sequence, we are able to identify the time segment containing the conduction of the task. Using the entire sequence to illustrate the step might not always be the optimal strategy, as the selection needs to be a trade-off between completeness and conciseness. The distinction between static, repetitive, and progressive segments allows for some improvement on this respect.

The decision what to show is dependent on the classifications of the current and following segments. In case of a progressive segment, we actually do use all frames for the animation, since it is difficult to determine whether a shorter snippet would be sufficient. Using the hand tracking information, we would indeed be

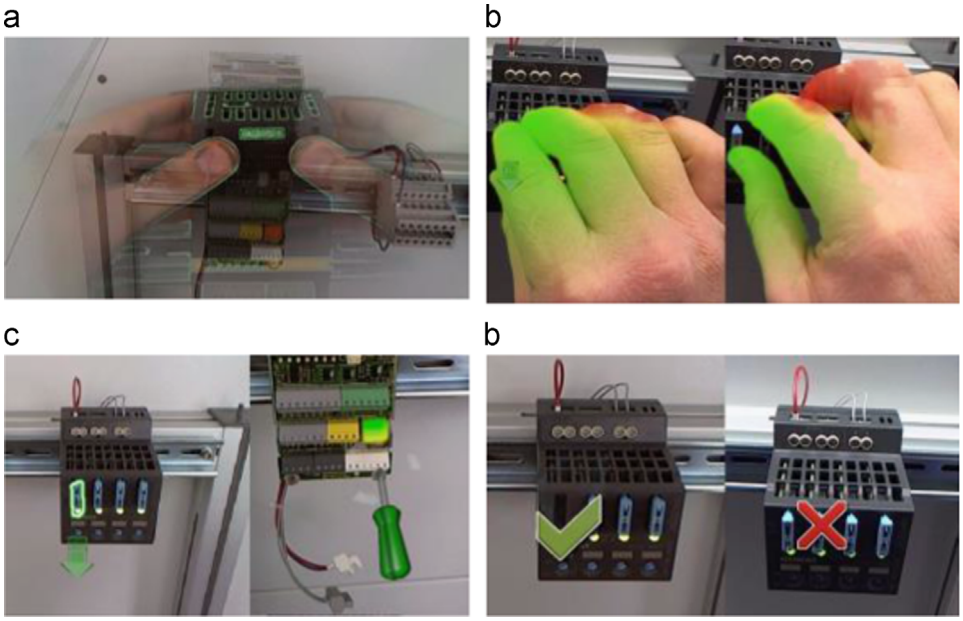

Fig. 5. Visual feedback provided by the system. (a) Procedural overlays, (b) enactive feedback, (c) annotational overlays, and (d) assessing overlays/optical validation.

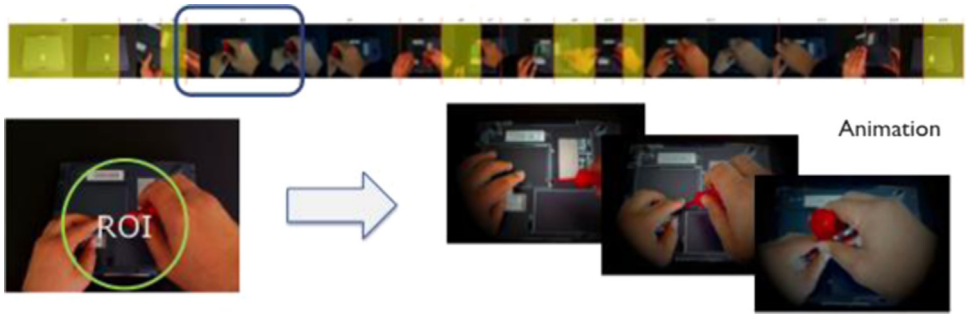

Fig. 6. Illustration of our method to automatically generate procedural overlays based on the segmentation results.

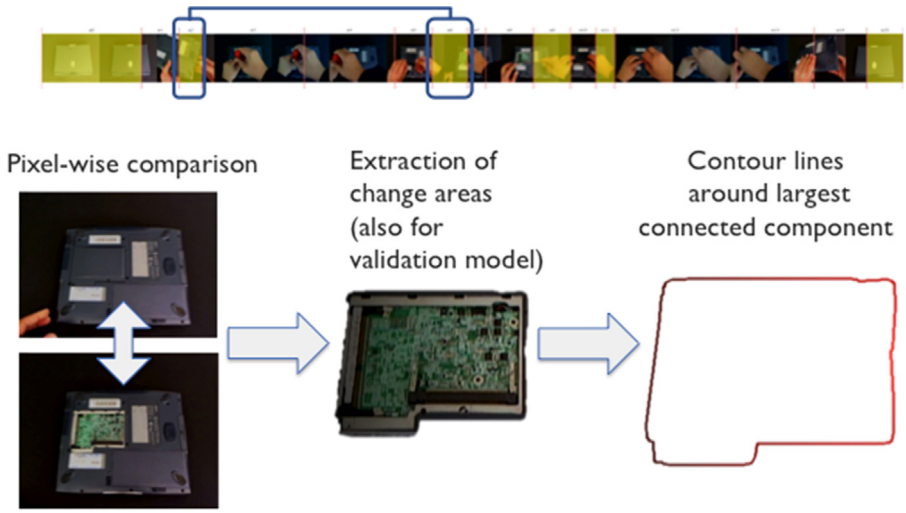

Fig. 7. Illustration of our method to automatically generate annotational overlays indicating changed workspace areas based on the segmentation results.

able to determine events such as grasping. However, we cannot safely decide, whether the footage before or after this salient event is important to illustrate the action.

However, when showing overlays sampled from repetitive segments, we do not playback the segment in its entirety. In case of a repetitive segment, we could use the cycle period, if detectable. In practice, we sample a snippet of fixed length from the middle of the temporal segment. There is an additional distinction depending on the subsequent segment: If the current segment is classified as being repetitive and the following as non-static, it is not determinable which cycle or repetition is the last one, in order to switch over to displaying instructions for the next step. Hence in this case, we always append the instructions for the following action to the current one. Although this means that the user is instructed on two consecutive actions at once, it ensures that both instructions will be shown to the user.

3.2. Enactive feedback

Through back projecting the color-coded location probability maps, into the field of view, we are able to provide real-time feedback about whether the user's hands are at locations that comply with the reference material. The color-coding is done, using a static look-up table. A very low or zero location probability is indicated as red, low as yellow, and high probability as green.

These colored maps are then projected into the current camera frame using the inverse Relevance Plane Transform and then used to tint the largest connected skin-colored regions, see Fig. 5(b). In addition to indicating clearly incorrect hand positions, it also reassures the user of the ongoing support through the system.

3.3. Indication of changed areas

Through comparing the reference sequence before and after a segmented user action, we are able to automatically identify image regions that have been altered in the course of the action, see Fig. 7. This is achieved through registering the *Relevance Planes* for the preceding and subsequent static segments [28]. Using a pixel-wise comparison, we segment discrepant regions between the geometrically registered common frames and select the largest connected component.

We can use this, to indicate workspace regions that are about to be altered in the following step. To that end, we display an annotational overlay containing the contour of the connected component at the beginning of the task, see left part of Fig. 5(c).

3.4. Optical validation

Further, we can use the altered regions to perform an optical validation of the state of the workspace. For that, we extract the corresponding image patches from the static segments before (prior state) and after (target state) the actual action takes place.

During run-time, we compare the target state patch with the tracked camera image when the user is assumed to have completed the respective work step using normalized cross-correlation. Hereby, we tolerate small using a pyramidal matching approach of subdivisions of the reference template. It is not straightforward to determine a threshold value for a successful match, as we do not know, whether a low score comes from an incorrect execution by the user or general image distortion effects due to changed lighting or viewpoint. We therefore use the known prior state to determine a suitable threshold. We match the prior

state patch using the same procedure to the live camera image just before the execution of the work step. Since we know that this matching score accounts for a positive match, we can use this as a threshold value for the subsequent comparison.

While the spatiotemporal tracking is reliably identifying when the user has reached a potential target state, it is not designed to discriminate between the possibly small appearance discrepancies that indicate errors. The normalized cross-correlation of the identified image regions is far more specific in this respect. Depending on the outcome of this comparison, we either acknowledge a correct (green check mark) or indicate an incorrect (red "x") completion, see Fig. 5(d).

4. Manual authoring

While we are able to automatically generate a rich set of visual representations, the system is unable to infer task goals and domain knowledge from the observation. For example, it is crucial to communicate any hazards to the user, e.g., from residual electrical charges, pressure, or chemicals that might not be obvious to an observer of the video references. Additionally, due to the occlusion or unobservability issues, the reference material simply might not depict all necessary information. Hence, it is mandatory to provide a way to manually augment the scene with instructional assets.

While this could involve things that novice observers could effectively infer from watching the reference material, there are important hints about safety regulations, warnings, or conventions that require expert domain knowledge to contribute. One of the principal aims of our work is to allow domain experts, *e.g.*, a maintenance worker, rather than a person knowledgeable in 3D creation tools to self-dependently implement the system described within this work. We briefly present our authoring-tool that does not require the author to have any knowledge about 3D content creation or tracking systems.

4.1. Structuring view

The tool divides the authoring procedure in two steps that are covered by two different views. The first step allows to review and to correct the automatically discovered workflow structure, see Fig. 8.

In particular, the user can add new recordings to the analyzed data body (Fig. 8, section "Import/record").

After adding a sequence, it is presented as a film strip with the segmentation result overlaid using different tints (Fig. 8, section "Segments"). Yellow indicates a static segment, blue a movement segment, and white indicates a segment containing a detected user action. The segment borders are indicated through vertical lines that the user can readjust freely through moving the line handles. Additionally, the user is able to easily delete unimportant or unintentional actions, as well as to combine or divide segments. While this view provides an easy interface for correcting possible segmentation or (in case of multiple recordings) synchronization errors, the actual authoring is handled in another view.

4.2. Authoring view

The main difficulty of the authoring process is dealing with the 3D nature of the problem. In more detail, the process requires to associate 3D coordinate frames, spanned by the (possibly disjunctive and local) tracking models with the assets provided by a 3D graphics designer. Neither the creation of 3D assets, nor the association with a 3D tracking system can generally be conducted by a domain expert. An exemplary screenshot of the authoring

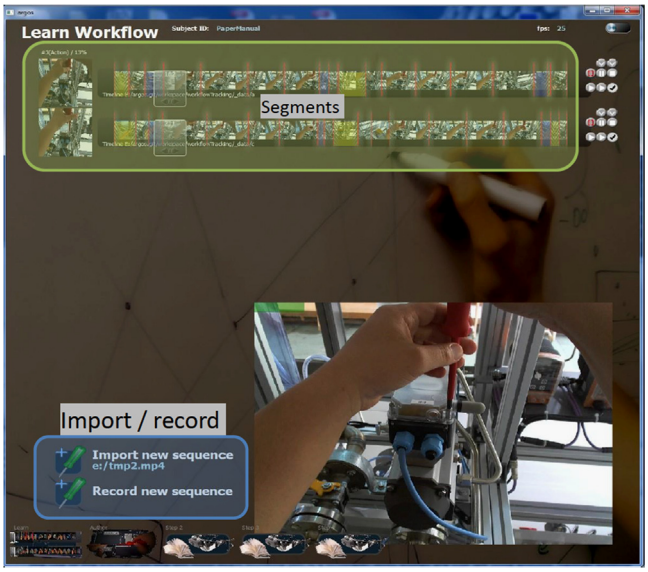

Fig. 8. Labeled screenshot from the learning view of the authoring tool: the recognized actions within all available workflow recordings are presented to the user. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

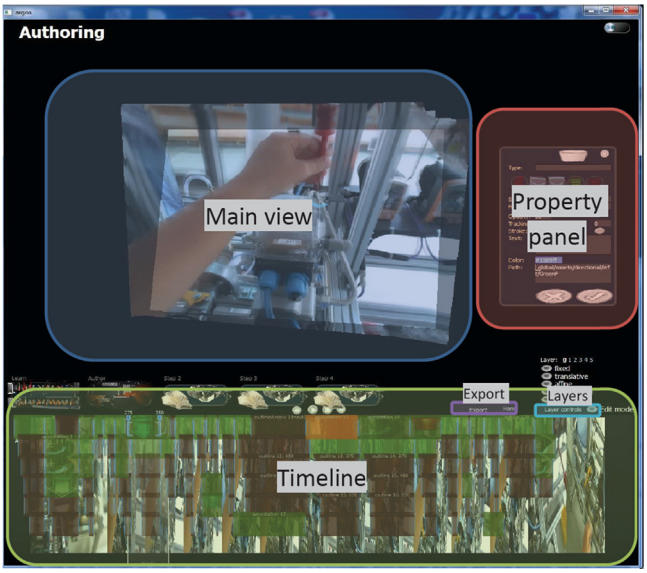

Fig. 9. Labeled screenshot from the authoring view of the authoring tool: the current frame is projected onto the common frame to allow easy annotation within a stabilized frame of reference.

view is shown in Fig. 9. The editing takes place within the stabilized common frame of the according Relevance Plane (Fig. 9, section "Main view"). This leads to a workflow that is more similar to annotating a still image than to annotating a 3D environment. To further simplify the procedure, we allow adding annotations using a set of predefined pen-stroke gestures. Fig. 10 shows the set of currently supported gestures and illustrates the procedure.

5. Context-aware user assistance

In contrast to the current state of the art in the field of computer-aided assistance and AR-based manuals, our system is able to automatically follow the progress of the user without the use of markers or other tracking aids. We distinguish several phases in the course of each work step. These are used to further filter the

theoretically available information, *e.g.*, to hide the procedural overlay when the user is already executing the instruction. In exchange, the user is provided with visual feedback for reassurance whether the task is currently conducted correctly.

Fig. 11 shows examples of the provided visual feedback and Fig. 12 illustrates the schematic data flow during run-time.

5.1. Scoping of displayed information

The point in time when AR overlays for the next action are displayed is crucial for the understanding. If the user is left unaware about the pending task for too long, the performance of the workflow will be stalled. If the next step is hinted too early, while the current step is not yet completed, it could potentially confuse the user. So, a helpful overlay must accomplish both, reassure the user of the correctness of the current behavior and announce subsequent actions early enough to minimize perplexity.

The fine-grained tracking of the user's performance even allows us to further break down the temporal granularity, subdividing each work step into several, partly overlapping phases. We can use these phases to exactly time, when information is displayed to the user. Therefore, we can massively increase the specificity in the selection of displayed information. As a result, this also effectively avoids visual cluttering, which might overburden or stress the user. Fig. 13 shows an example of visual clutter already occurring with a single type of visual overlay when not scoping the displayed information.

Fig. 14 shows the temporal order of the distinguished phases, in particular the instruction phase, psychomotor phase, validation phase, and transition phase:

Instruction phase: When a user has finished the preceding work step and is ready to be instructed about the new task. Technically, we determine the instruction phase either as the static segment before the next segment containing a user activity. In case of adjacent non-static segments, we subdivide the non-

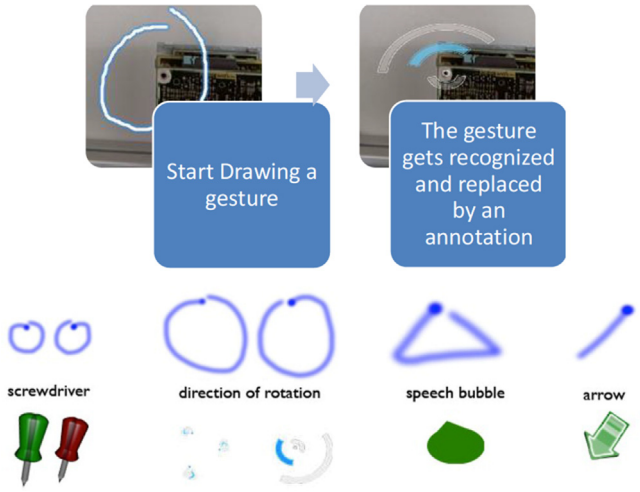

Fig. 10. Illustration of the annotation procedure and selection of supported gestures.

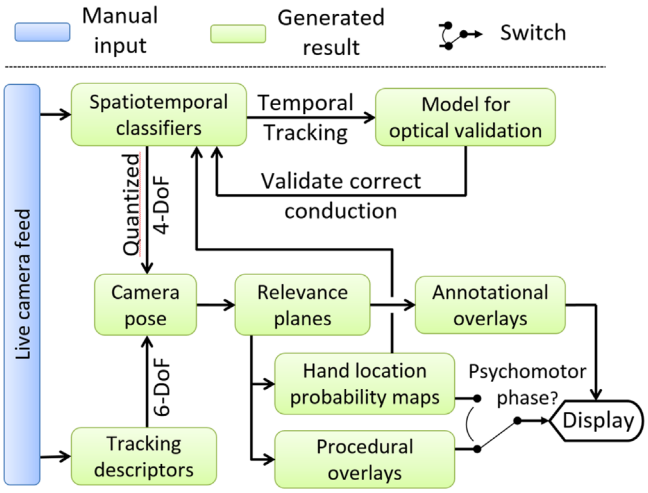

Fig. 12. Data flow diagram of the run-time process: using the set of classifiers, the temporal segment is determined. When possible, the resulting rough camera pose is refined using point descriptor matching and used to back project the relevance planes, in order to display the overlays.

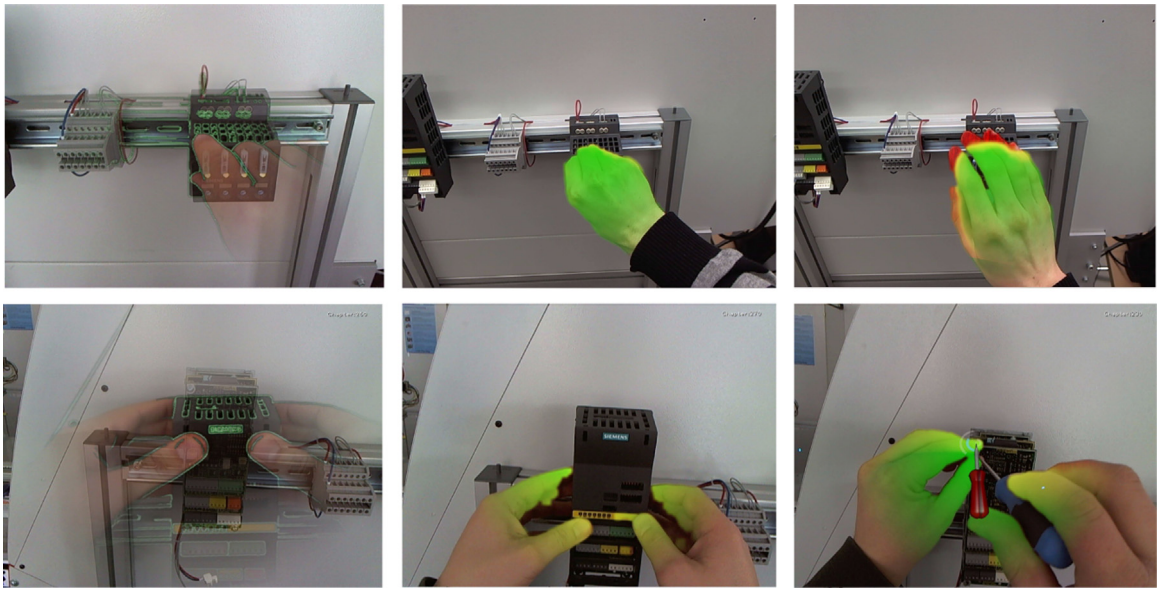

Fig. 11. Examples from an automatically authored AR-manual: the half-transparent overlays (left) were automatically extracted from the reference sequence. The green coloring (middle) indicates a correct conduction, red a wrong posture or position. The augmented tools (lower right) have been added manually. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

static segment and set the instruction phase as the beginning of the non-static segment before detecting the user's hand entering or coming close to the target posture.

Psychomotor phase: As soon as the user begins with the actual execution of a work step. The system hides the procedural overlays and instead displays the enactive feedback. As the procedural overlays would very likely interfere with the actual appearance of the scene, hiding these during the psychomotor phase largely reduces visual clutter of the interface. Additionally, through providing visual feedback over the correctness of conduction, we are able to reassure the user of a correct execution and the ongoing support by the assistance system. Technically, we detect the beginning of this phase by solely observing the matching scores of the hand location probability maps.

To the best of our knowledge, this is the first AR-based assistance system that provides this level of support during the psychomotor phase. While [37] also proposes exchanging the provided information, their system is heavily dependent on markers. In contrast, this work describes the first realization using solely natural features.

Validation phase: After the conduction of the step, the system displays a reference image for manual inspection, or, if possible, performs an automatic validation in which case it simply displays the outcome of this inspection.

The beginning of this phase is conditioned on the segment being surrounded by static segments before and after and the user actually retracting his or her hands after completing the step. The reason is simple, as the validation is conducted by analyzing the workspace appearance as seen from the head-worn camera, which trivially requires the workspace being observable. This also needs to be true for the reference material, including the view on the unobstructed workspace before and after the user interaction to allow for an automatic identification of altered regions.

Fig. 13. Visual clutter due to procedural overlays interfering with the current appearance of the workspace.

Transition phase: Between steps, if there is a change of the region of interest or even a change of location involved. This phase is active during a movement segment, if a change in position needs to be communicated before instructing the next step. Alternatively, this phase starts after a fixed period of time after the validation phase. For example, this could be used to present the user with an overview of his or her current progress within the workflow, by displaying a short summary of completed and forthcoming work steps.

We chose the term phase to emphasize that this represents an additional subdivision of each segment, as the main unit of temporal/procedural progress. While the segments are entirely determined through the reference material, the different phases are additionally conditioned on how the user conducts the workstep. For example the transition phase can be entirely omitted by immediately continuing with the subsequent action.

A user can also repeatedly cycle between instruction, psychomotor, and validation phase for each work step. This could be further extended by detecting certain operational modes, like a state of confusion of the user. For example, remaining in the instructional phase for a long time, or cycling more than two times through the aforementioned phases could be interpreted as a sign of confusion and be answered with additional support through the system.

5.2. Viewpoint guidance

During the workflow, the user might need to be instructed to reposition his or her viewpoint for several reasons. The obvious one is that this change in viewpoint also occurred in the reference recording. In this case, the viewpoint guidance could actually be interpreted as procedural overlay. This might be due to didactic reasons, in order to facilitate the understanding of the recorded procedure. In addition to that, there is of course the technical necessity to not severely deviate from the viewpoint of the reference recording. We therefore reuse the same visual hints to guide the user back to the point of view that matches the one from the reference material. We use a slightly modified implementation of attention funnels [36] to guide the user towards the target viewpoint.

Fig. 15 shows an example of the visual representation. From the approach described in paragraph 2, we receive a homography relative to the reference material, which we can propagate frame-by-frame using the camera tracking approach as described in [27,28]. This visual hint is hidden from the user, if the relative deviation from the closest matching references material is sufficiently small.

6. Conclusion

This paper presents an overview of a novel learning-based authoring approach towards procedural task assistance using

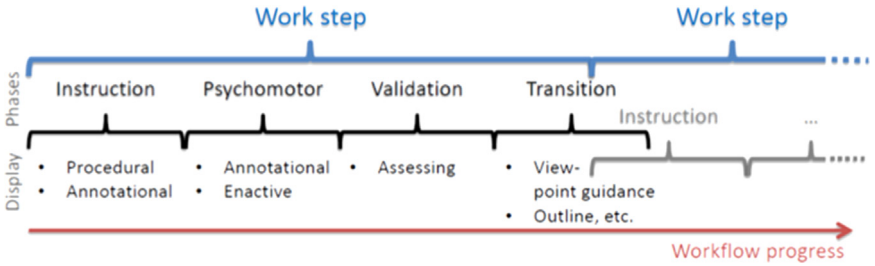

Fig. 14. Illustration of the four partly overlapping phases distinguished within each work step and the respectively displayed information.

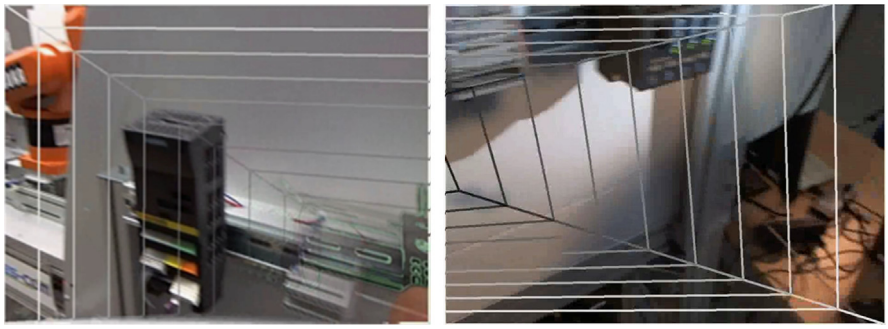

Fig. 15. Example of the attention funnels used to guide the user to a target viewpoint.

Fig. 16. Left: Protocol as HTML-page generated automatically out of the recorded video; right: Cognitive Augmented Reality – live Augmented Views created from the exemplary recorded workflow and reacting to user actions.

Augmented Reality. The resulting system is comprehensive and allows the fully automatic creation of Augmented Reality manuals from video examples as well as their context-driven presentation in AR. The recording of a single reference recording of a workflow is sufficient for a practical system. With availability of additional reference recordings, the system not only improves in precision and recall but is also able to estimate certain task-specific properties, like required level of accuracy and distinction of erratic and intended actions. The presented approach is the first to combine classical AR with machine learning, classification and basic reasoning methods, leading towards a cognitive system, aware about scene state and user actions. To underscore the extension of the merely spatial paradigm of Augmented Reality with the cognitive components, we call this combination *Cognitive Augmented Reality*.

Beside the presented live augmentation with Head-Mounted Display, the proposed technology and methodology opens many additional fields of application, such as the automated generation of written task documentation (see Fig. 16), support for documenting error indications, or analysis of maintenance procedures on a process level. In the future work, we will focus on task recognition while processing a reference recording to be able to incorporate planning algorithms and gain further understanding about the underlying workflow.

Acknowledgment

The work presented in this paper was performed in the context of the Software-Cluster project EMERGENT (www.softwarecluster.org) and the project COGNITO (www.ict-cognito.org, ICT-248290). It was funded by the German Federal Ministry of Education and Research (BMBF) under Grant no. “01IC10S01” and under the “ICT-2009.2.1”.

References

[1] Caudell TP, Mizell DW. Augmented reality: an application of heads-up display technology to manual manufacturing processes. In: Proceedings of the Hawaii international conference on system sciences; 1992.
[2] Weidenhausen J, Knöpfle C, Stricker D. Lessons learned on the way to industrial augmented reality applications, a retrospective on ARVIKA. Comput Gr 2003;27(6):887–91.
[3] Encarnação JL, Stricker, D. Augmented-Reality für industrielle Anwendungen in Entwicklung. Produktion und Service am Beispiel des Leitprojekts ARVIKA (1999 bis 2003), Reuse, Bernd (Hrsg.); Vollmar, Roland (Hrsg.). Informatikforschung in Deutschland. Springer, Verlag, Berlin, Heidelberg, New York; 2008.
[4] Werner Schreiber, Peter Zimmermann, editors. Virtuelle Techniken im industriellen Umfeld, Das AVILUS-Projekt - Technologien und Anwendungen. Berlin, Heidelberg: Springer-Verlag; 2011 http://dx.doi.org/10.1007/978-3-642-20636-8, eBook ISBN: 978-3-642-20636-8, Hardcover ISBN: 978-3-642-20635-1.
[5] Echtler F, Sturm, F, Kindermann K, Klinker G, Stilla J, Trilk J, Najafi H. The intelligent welding gun: augmented reality for experimental vehicle construction. In: Virtual and augmented reality applications in manufacturing. Springer; 2004. p. 333–60.
[6] Alvarez H, Aguinaga I, Borro D. Providing guidance for maintenance operations using automatic markerless Augmented Reality system. In: Proceedings of the international symposium on mixed and augmented reality (ISMAR); 2011.
[7] Salonen T, Sääski J. Dynamic and visual assembly instruction for configurable products using augmented reality techniques. In: Advanced design and manufacture to gain a competitive edge. Springer; 2008. p. 23–32.
[8] Ke C, Kang B, Chen D, Li X. An augmented reality-based application for equipment maintenance. Affect. Comput. Intell. Interact. 2005;3784:836–41.
[9] Benbelkacem S, Zenati-Henda N, Belhocine M, Bellarbi A, Tadjine M, Malek S. Augmented reality platform for solar systems maintenance assistance. In: Proceedings of the international symposium on environment friendly energies in electrical applications (EFEEA); 2010.
[10] Ockerman JJ, Pritchett AR. Preliminary investigation of wearable computers for task guidance in aircraft inspection. In: Proceedings of the international symposium on wearable computers (ISWC); 1998.
[11] Schwerdtfeger B, Klinker G. Supporting order picking with augmented reality. In: Proceedings of the international symposium on mixed and augmented reality (ISMAR); 2008.
[12] Besbes B, Collette SN, Tamaazousti M, Bourgeois S. An interactive augmented reality system: a prototype for industrial maintenance training applications. In: Proceedings of the international symposium on mixed and augmented reality (ISMAR); 2012.

[13] Webel S, Becker M, Stricker D, Wuest H. Identifying differences between CAD and physical mock-ups using AR. In: Proceedings of the international symposium on mixed and augmented reality (ISMAR); 2007.
[14] Reiners D, Stricker D, Klinker G, Müller S. Augmented reality for construction tasks: doorlock assembly. In: Proceedings of the international workshop on augmented reality (IWAR); 1998.
[15] Mura K, Gorecky D, Meixner G. Involving users in the design of augmented reality-based assistance in industrial assembly tasks. In: Applied Human Factors and Ergonomics; 2012.
[16] Tversky B, Heiser J, Lee P, Daniel M-P. Cognitive design principles for automated generation of visualizations. In: ALLEN G, editor. Applied spatial cognition: from research to cog- nitive technology. Hillsdale, NJ: Lawrence Erlbaum Associates, Inc.; 2006. p. 53–74.
[17] Wang X, Dunston PS. Design, strategies, and issues towards an augmented reality-based construction training platform. ITcon J 2007;12:363–80.
[18] Gavish N, Gutierrez T, Webel S, Rodriguez J, Tecchia F. Design guidelines for the development of virtual reality and augmented reality training systems for maintenance and assembly tasks. BIO Web Conf 2011;1:1–4.
[19] Webel S, Bockholt U, Keil J. Design criteria for AR-based training of maintenance and assembly tasks. In: Virtual and Mixed Reality-New Trends. Springer; 2011. p. 123–32.
[20] Agrawala M, Phan D, Heiser J, Haymaker J, Klingner J, Hanrahan P, Tversky B. Designing effective step-by- step assembly instructions. Trans Gr (TOG) 2003;22(3):828–37.
[21] Driskill E, Cohen E. Interactive design, analysis, and illustration of assemblies. In: Proceedings of the symposium on interactive 3D graphics; 1995.
[22] Agrawala M, Li W, Berthouzoz F. Design principles for visual communication. Communications of the ACM; 2011.
[23] Zauner J, Haller M, Brandl A, Hartmann W. Authoring of a mixed reality assembly instructor for hierarchical structures. In: Proceedings of the international symposium on mixed and augmented reality (ISMAR); 2003.
[24] Bleser G, Stricker D. Advanced Tracking through efficient image processing and visual-inertial sensor fusion. . Reno, Nevada, US: IEEE Virtual Reality Conference (VR); 2008.
[25] Wuest H, Pagani A, Stricker D. Feature management for efficient camera tracking. . Tokyo, Japan: Asian Conference on Computer Vision (ACCV); 2007. p. 769–78.
[26] Vacchetti L, Lepetit V, Fua P. Combining edge and texture information for real-time accurate 3D camera tracking. In: Proceedings of the 3rd IEEE/ACM international symposium on mixed and augmented reality (ISMAR'04). Washington, DC, USA: IEEE Computer Society; 2004. p. 48–57.
[27] Petersen N, Stricker D. Learning task structure from video examples for workflow tracking and authoring. In: Proceedings of the 2012 IEEE international symposium on mixed and augmented reality (ISMAR) (ISMAR'12). Washington, DC, USA: IEEE Computer Society. p. 237–46.
[28] Petersen N, Pagani A, Stricker D. Real-time modeling and tracking manual workflows from first-person vision. In: Proceedings of the 2013 IEEE International Symposium on Mixed and Augmented Reality (ISMAR); 1–4 October 2013. p.117, 124.
[29] Petersen N, Stricker D. Morphing Bi l lboards for accurate reproduction of shape and shading of articulated objects with an application to real-time hand tracking. In: Proceedings of computational modeling of objects presented in images (ComplImage); 2012.
[30] Petersen N, Acquiring and transferring workflow knowledge using Augmented Reality [Ph.D. thesis]. TU Kaiserslautern.
[31] Knöpfle C, Weidenhausen J, Chauvigne L, Stock I. Template based authoring for ar based service scenarios. In: Proceedings of the virtual reality conference (VR); 2005. p. 237–40.
[32] Maeda T, Ando H. Wearable robotics as a behavioral interface – the study of the parasitic humanoid. Proceedings of the sixth international symposium on wearable computers (ISWC 2002); 2002. p.145, 151.
[33] Carvalho E, Domingues H, Mações G, Santos LP. Augmented reality visualization and edition of cognitive workflow capturing. In: Proceedings of the experiment@ international conference (exp.at); 2011.
[34] Hinterstoisser S, Lepetit V, Ilic S, Fua P, Navab N. Dominant orientation templates for real-time detection of textureless objects. In: Proceedings of the computer vision and pattern recognition (CVPR); 2010.
[35] Petersen N, Stricker D. Morphing billboards – an image based appearance model for hand tracking. Comput Methods Biomech Biomed Eng: Imaging Vis 2015;3(2):63–75.
[36] Biocca F, Tang A, Owen C, Xiao F, Lansing E. Attention funnel: omnidirectional 3D cursor for mobile augmented reality platforms. In: Proceedings of the SIGCHI conference on human factors in computing systems, CHI'06; 2006.
[37] Henderson SJ, Feiner SK. Augmented reality in the psychomotor phase of a procedural task. In: Proceedings of the international symposium on mixed and augmented reality (ISMAR); 2011.
[38] Clark A. An embodied cognitive science. Trends Cogn Sci 1999;3(9):345–51.