Real-time Modeling and Tracking Manual Workflows from First-Person Vision

Nils Petersen [*] Alain Pagani [†] Didier Stricker [‡]

German Research Center for Artificial Intelligence (DFKI)

Abstract

Recognizing previously observed actions in video sequences can lead to Augmented Reality manuals that (1) automatically follow the progress of the user and (2) can be created from video examples of the workflow. Modeling 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 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 image region that is consistent with a homographic model and contains a given region of interest. We are then able to model the state of the environment and user actions using simple 2D region descriptors. The model elegantly handles estimation errors due to incomplete observation and is robust towards occlusions, *e.g.,* due to the user's hands. We demonstrate the effectiveness of our approach quantitatively and compare it to the current state of the art. Further, we show how we apply the approach to visualize automatically assessed correctness criteria during run-time.

1 Introduction

Workflow knowledge comprises both explicit, verbalizable knowledge and implicit knowledge, which is acquired through practice. While the first type can be well presented in the form of traditional paper documentation, the second requires or at least benefits from training with a permanent corrective. Current approaches to Augmented Reality (AR) manuals have mostly concentrated on making the transfer of the first type more efficient (*e.g.,* [1, 23, 6]). This has resulted in systems that allow the user to display instructions for a certain work step until the user requests the next instruction manually. The didactive gain of these systems is principally unchanged in comparison to paper manuals with the difference of omitting the cognitive load needed to associate a textural explanation or an instructive sketch with the current work environment.

Being able to track the manual workflow itself allows improving on two fronts: Firstly, to make the running system follow the user automatically while performing the task. Secondly, exploiting the fact that we deal with an interactive system by also conveying feedback over the quality or (whenever possible) the correctness of the task execution. Figure 1 shows examples of this visual feedback.

As [18] has shown recently, it is possible to automatically create interactive Augmented Reality manuals from video examples of a reference performance. The approach mainly focused on unsupervised temporal segmentation of the workflow video recording with a fixed camera.

[*]e-mail: nils.petersen@dfki.de
[†]e-mail:alain.pagani@dfki.de
[‡]e-mail:didier.stricker@dfki.de

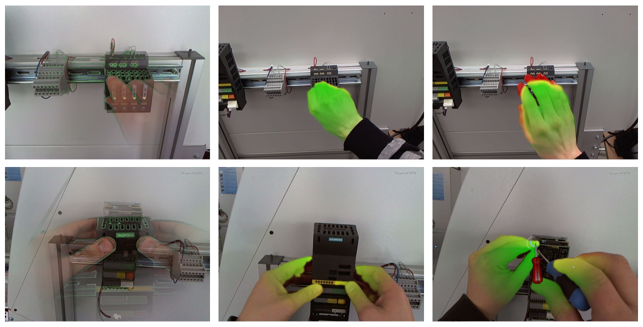


Figure 1: Example application showing an automatically authored AR-manual. The half-transparent overlays (left column) were automatically extracted from the reference sequence. The green coloring indicates that the current step is conducted correctly, red indicates a wrong posture or position

In this work, we extend the approach to extract the required information from a moving, for example head-worn camera. This allows the ad-hoc, in-situ documentation of workflows during the execution. The topic is getting particularly relevant (as reflected in [9]) with the advent of less obstructive, consumer-targeted first-person vision cameras in combination with AR displays like the Vuzix M100 and most prominently Google Glass.

Contributions: We propose a robust, piecewise homographic transform that we call Relevance Plane Transform (RPT) 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. As we aim for manual workflows, we select the planar subset that contains the region of most user interaction, derived from the temporal task structure and the viewing direction. The result is a piecewise two-dimensional spatiotemporal model of dynamic, changing environments. We use this to sample 2D probability maps of the hand location and to sample instructive snippets as described in [18] from a moving camera. It does not require knowledge (or consistency) of 3D scene geometry, explicitly copes with dynamic, changing environments, and works with uncalibrated cameras.

In addition to this main contribution, we present a method to automatically assess certain correctness indicators and propose means of visualizing them to the user during the workflow. The resulting system allows the fully automatic creation of Augmented Reality manuals from video examples as well as their context-driven presentation in AR. In contrast to the current state of the art in this area, our system is able to automatically follow the progress of the user, particularly without using markers or any other tracking aid. Due to the visual feedback that is provided while following the procedural instructions, it can be used as an interactive tutoring system that also conveys feedback over the assessed correctness of the execution.

Paper structure: We start with a review of the problem properties. We will then review related work followed by a detailed explanation of our main contribution, the Relevance Plane Transform (Section 3). After that, we show how we sample and apply the location probability maps (Section 4.1). Details regarding the visual feedback as well as a brief overview of the entire system is provided in Section 5 We evaluate the tracking performance in comparison to the approach presented in [18] and conclude our work in Section 7.

1.1 Problem properties

Analyzing video material to automatically create AR manuals leads to some quite unique and specific constraints. We briefly summarize the key aspects to motivate our tracking design decisions.

Camera motion and viewpoint: Video material is typically recorded from a head-worn camera leading to ego-perspective recordings. Camera motion during a certain manual work step will dominantly consist of orientation change and we cannot assume sufficient camera translation to reliably reconstruct geometry.

Environment: Additionally, the environment is susceptible to change due to user interaction, which affects scene geometry and trackable features.

User: When using the resulting AR manual, we can assume a cooperative user that supports the system when given appropriate feedback. However, this assumption does not necessarily hold for the training material.

2 Related work

There has been extensive work on the general use case of procedural assistance using Augmented Reality. The early work of Caudell and Mizell [3] promoted this use case for head-up displays, thereby coining the term *Augmented Reality*. Since then, according systems were presented based on fiducials [20] or CAD models [19] and evaluated for various application domains like automotive [14], military [6], object assembly [24], and manufacturing [16].

Recently, the authors of [7] have evaluated AR in the psychomotor phase of a workflow, the phase wherein the user actually executes each work step. They use markers, attached to all tracked objects and on the head-worn display. On moving one of the incorporated objects, indicating the beginning of the psychomotor phase [15], their system presents new overlays (dynamic arrows, highlights, or labels) to assist the user during the execution. While we share the distinction between an informational phase and the psychomotor phase, we focus on the technical realization of markerless spatiotemporal tracking. Additionally, the general scope of our work is to automate the creation process of systems for procedural assistance. For details on the authoring aspects of our approach, please refer [18].

The authors of [25] evaluate a comprehensive AR-based training system and discuss various aspects of skill transfer, AR-based training, and tele-consultation. In addition to *adaptive visual aids* that are adjustable with respect to their guidance level, they also apply additional haptic feedback using a vibrotactile bracelet. Their approach builds on pre-authored multimedia content, which is additionally complemented and extended by a remote expert, whereas we propose to extract this type of information from video examples showing a reference performance.

One major challenge when gathering information from an unconstrained, dynamically changing environment is to define a data structure that can hold and describe the findings. One straightforward approach would be to anchor information at 3D locations, *i.e.,* to annotate the scene's 3D geometry (or an online reconstruction of it). Examples for this method are all marker based approaches [11], 3D model-based approaches [26], or SLAM-related approaches [10]. As discussed in section 1.1 this is not feasible in our scenario.

Another popular method is to associate information with 2D image features. In this case, information is anchored with point-features [21, 2], region descriptors [8] or object detectors [27] that principally can operate separately on single frames of the sequence. While this is often sufficient and feasible, it has one major disadvantage as it does not allow a spatially continuous annotation of the scene.

We propose to anchor information within a dynamic scene using a temporal series of spatially continuous 2D representation. These 2D maps are registered with the scene trough a planar structure that contains a certain region of interest. In contrast to methods like [4, 12], we do not aim for an accurate reconstruction of the environment or the camera pose which relaxes most of the constraints on scene geometry. Particularly, our model does not imply any requirements on camera motion, like it is necessary for structure from motion (SfM) and SLAM-based methods.

The authors of [5] have recently proposed a tracking and mapping scheme that deals with the motion requirements through explicit model switching. In contrast, our approach is based on prior temporal segmentation that leads to distinct or only loosely coupled local 2D maps always using a homographic model. The temporal segmentation is derived from task structure analysis of the reference sequence [18] with the extension that strong violations of the homographic model can additionally lead to a new segment.

3 Relevance Plane Transform

The core idea is to identify the planar image structure (the so-called Relevance Plane RP) that contains a certain region of interest. All images that share the same region of interest (ROI) are then projected into a common 2D coordinate frame using homographies acquired from tracking the planar structure. The corresponding ROIs are selected according to the temporal task structure, estimating locations of user interaction. We assume that the user touches the environment in the course of each work step. Therefore, the contact points will always sharply project into the common frame. Content at different depths will show a reprojection error proportional to the distance to the RP unless camera motion is purely rotational. Figure 2 illustrates the model and this consideration.

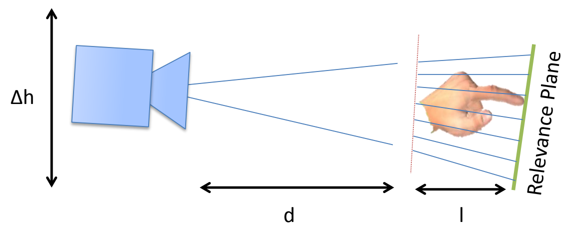


Figure 2: Illustration of the model assumption: The RPT provide accurate estimates if Δh and l are small compared to d

The camera motion during a single work step exhibits only little translational motion, as pointed out in Section 1.1, which leads to only small reprojection errors outside of the RP. The reason why simple image stabilization is not sufficient is that between different users, however, the translational offset might be significant. While our approach is equal to whole image stabilization in case of purely rotational motion, with translational motion, the tracking support converges to the planar structure. By exploiting the fact that with degenerate (*i.e.,* purely rotational) motion, the entire image sharply projects into the common frame, we can sample without actually estimating the Relevance Plane. The segmentation can still take place during online tracking.

In the following sections, we explain how we track the camera (Section 3.1). After that we show how the various ROIs are selected (Section 3.2). and how we robustly segment the largest support region (Section 3.3) to track the Relevance Plane (Section 3.4).

3.1 Camera tracking

We use a very simple but practical alignment scheme to track the camera.

Step 1: We start by selecting arbitrary corner features $\mathcal{P}_t$ within frame $t = 0$. After locating the correspondences within the next frame using KLT [22], we use RANSAC to find the largest subset of correspondences whose movement can be described using a homography, *i.e.,* we determine the largest set of correspondences $\mathcal{P}_{t+1}$ and the homography $\mathbf{H}_t^{t+1}$ that satisfies

$$\|\vec{p}_{t+1} - \mathbf{H}_t^{t+1}\vec{p}_t\|_2 \leq \varepsilon, \quad \vec{p}_t \in \mathcal{P}_t,\ \vec{p}_{t+1} \in \mathcal{P}_{t+1} \tag{1}$$

where ε is an error threshold. The impact of a suboptimal RANSAC solution is negligible as (1) the method gracefully deteriorates with suboptimal results and (2) the step gets repeated on every frame. Though, it is important not to set ε too low, since it affects how the tracking support is enlarged in case of occlusions. We found values between 4 and 6 pixels to produce good results w.r.t. an image size of 960×720 pixels.

Step 2: For the next frame $t = 1$, we repeat KLT and RANSAC with the already determined set of points $\mathcal{P}_{t+1}$ to estimate $\mathbf{H}_t^{t+1}$ for $t = 2$. After that, we select new corner features across the entire image in frame t, find correspondences in $t+1$ and directly apply (1) to reject points that do not comply with the homographic model. We continue to track by repeating Step 2 for all subsequent frames $t = 3..n$. In case of a complete loss of tracking, *i.e.,* $\mathcal{P}_t = \emptyset$, we repeat Step 1.

This simple scheme only provides camera location w.r.t. a random subset of the image and tends to drift quickly (which is negligible in our framework since we segment the sequence into several independent and rather short segments). Nevertheless, it has some useful properties that we will later use to segment the Relevance Plane: Firstly, it operates consistently on translational and rotational-only camera motion. SLAM-related methods mostly require a certain initialization and movement pattern (*e.g.,* [10]) or use explicit model switching to cope with degenerate motion (*e.g.,* [5]). Our method gradually converges to a planar subset with according motion, which is very important for our segmentation scheme. Secondly, in case of strong occlusions or change of environment, this scheme gradually deviates from the coplanar point set (steered by the value of ε). Camera tracking continues despite a fully occluded target, although of course with a higher tracking error.

3.2 Determining the set of Relevance Planes

We switch the ROI and therefore the Relevance Plane on every task of the workflow. To that end we adopt the segmentation criteria from [18] and extend it by additionally using strong camera movement as a cue for a changed region of interest. The top rows of Figure 8 and 9 show the segmentation result.

3.2.1 Temporal task structure

For the sake of completeness, we provide a brief summary of the original segmentation process. For details, please refer to the original publication [18].

The approach uses whole-image distance functions of the sort $d(\boldsymbol{I}_1, \boldsymbol{I}_2) \to \mathbb{R}$ where $\boldsymbol{I}_1$ and $\boldsymbol{I}_2$ are two arbitrary images of an ordered image sequence. In order to cope with lighting changes and small perspective deformations, $d(\boldsymbol{I}_1, \boldsymbol{I}_2)$ is implemented using the DOT region descriptor [8]. To further minimize cross speaking 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.

Formally, this is introduced as a distance threshold T and only appreciating violations $d(\boldsymbol{I}_1, \boldsymbol{I}_2) > T$, a so-called *novelty*. The segmentation is then based on minimizing the shortest-path, *i.e.,* finding a set of frames with the least amount of novelties that connects between the first and the current frame of the sequence. 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 and thus, 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.

3.2.2 Strong camera movement

For the application at hand, this adopted segmentation strategy is extended with segmenting on strong camera movement. Since we want to distinguish different types of movement and weight them differently, we derive three measures for different components of a homography $\mathbf{H}$. For in-plane translation, we simply measure the translational shift of the image center:

$$\tau(\mathbf{H}) = d\left((c_x, c_y, 1)^T, H(c_x, c_y, 1)^T\right) \tag{2}$$

where $d(\vec{h}_1, \vec{h}_2)$ is the Euclidean distance of the points after "unhomogenizing", and c_x, c_y are the pixel coordinates of the image center (or the optical center, if available).

For assessing out-of plane rotation, we score the perspective distortion of the image center:

$$\phi(\mathbf{H}) = \frac{\max_{i=1..4} d_i}{\min_{i=1..4} d_i} \tag{3}$$

with $d_{1..4}$ being the lengths of the 4 edges of a distorted square:

$$\begin{aligned}
d_1 &= d(\mathbf{H}(c_x{-}1, c_y{-}1, 1)^T, \mathbf{H}(c_x{+}1, c_y{-}1, 1)^T) \\
d_2 &= d(\mathbf{H}(c_x{+}1, c_y{-}1, 1)^T, \mathbf{H}(c_x{+}1, c_y{+}1, 1)^T) \\
d_3 &= d(\mathbf{H}(c_x{+}1, c_y{+}1, 1)^T, \mathbf{H}(c_x{-}1, c_y{+}1, 1)^T) \\
d_4 &= d(\mathbf{H}(c_x{-}1, c_y{+}1, 1)^T, \mathbf{H}(c_x{-}1, c_y{-}1, 1)^T)
\end{aligned}$$

Finally, movement along the optical axis is scored as

$$\sigma(\mathbf{H}) = \log^2 \frac{d_1 + d_2 + d_3 + d_4}{8} \tag{4}$$

Since we deal with a head-worn camera, we want to ignore short, likely unintentional movements. To that end, we filter values within a sliding window of length w, only appreciating the minimum motion value. We segment the sequence if one of these measures exceeds a certain threshold throughout the entire sliding window, *i.e.,* if $\min\limits_{k=t-w..t} \tau(\mathbf{H}_{t-w}^{-1}\mathbf{H}_k) > T_\tau$, analogous for ϕ, σ. The camera movement thresholds T_τ, T_ϕ and T_σ are hereby determined experimentally. Since rotations around the optical axis do have a negligible effect, we are entirely ignoring this kind of movement.

Movement along the camera axis often occurs because the user performs work that deals with details or requires a high accuracy. Due to this, we grant a high threshold T_σ as segmentation condition, although it has a strong impact on perspective distortion and sampling precision of the respective maps.

3.2.3 Selecting the region of interest

We exploit the task structure to determine the region of most user interaction. We start with temporal segments $\boldsymbol{\mathcal{S}}_i$ that have been classified as containing user actions. We represent the viewing direction of each image $\boldsymbol{I}_t \in \boldsymbol{\mathcal{S}}_i$ using an attention mask $\boldsymbol{M}_t$ that is 1 in the image center and radially fades out to 0. After aligning these masks using the homography $\hat{\mathbf{H}}_t = (\frac{1}{|\mathcal{S}_i|}\sum_k \mathbf{H}_k^{k+1})^{-1}\mathbf{H}_t^{t+1}$ we determine the region of interest of $\mathcal{S}_i$ as

$$ROI_i = \text{thres}_\kappa\left(\sum \text{warp}_{\hat{\mathbf{H}}_t}(\boldsymbol{M}_t)\right) \tag{5}$$

where $\text{warp}_{\hat{\mathbf{H}}_t}$ warps the image using the homography $\hat{\mathbf{H}}_t$ and thres_κ is a binary image threshold operator with threshold value κ. We reuse the same ROI within the directly adjacent static or movement segments. In case of two neighboring action segments the subsequent one propagates the ROI.

We also tried to define the ROI as the area of greatest optical flow, around the centroid of the hand silhouette, at the location of the fingers estimated through [17], and through combinations of the three but found that this approach worked most reliably in practice.

3.3 Segmenting the Relevance Plane

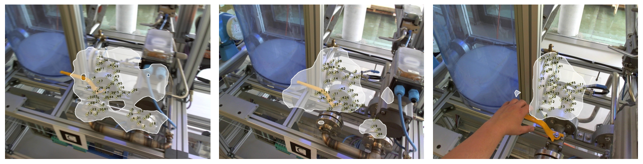


Figure 3: Support region while tracking the Relevance Plane

Segmenting the RP is quite analogous to our camera tracking scheme described in Section 3.1 with simple adjustments:
Altered Step 1: The corner selection is constrained to a support region initialized with the region of interest.
Altered Step 2: The support region is being updated by thresholding the density map of the currently tracked features before rejecting points that do not comply with $\mathbf{H}_t^{t+1}$. Figure 3 illustrates how the support region is propagated.

Without occlusions and with sufficient camera motion, the support will converge to a planar subset of the scene that strongly overlaps the region of interest. In presence of occlusions, the support drifts to a planar subset that satisfies (1). Note that due to the radial distortions of an uncalibrated camera, the support will not span the entire planar structure.

The homography to transform an image $\boldsymbol{I}_t$ into the common coordinate frame is given by:

$$\hat{\mathbf{H}}_t^{RP} = (\frac{1}{|\mathcal{S}_i|}\sum_k \mathbf{H}_k^{k+1})^{-1}\mathbf{H}_t^{t+1} \tag{6}$$

where $|\mathcal{S}_i|$ is the number of images in the segment. This type of interpolation of the homographies is along the circular secant, not the arc; therefore it degenerates in case of strong rotation. However, since we also separate common frames according to movement cues, this type of interpolation becomes feasible within this application.

3.4 Matching and Tracking

For initialization, we begin with the region matching approach as proposed by [18] to get a rough four degrees of freedom (4-DOF) quantized pose estimate (scale, rotation, x- and y-translation). Continuing from this pose estimate, we use a point matcher to refine it into a 6-DOF pose estimate. Although this way of recovering the pose estimate is considerably slower than using just the point matching approach, it is highly robust towards lack of texture or occlusions.
Build point descriptors: We compute ORB [21] keypoints and descriptors within the Relevance Plane support for each image projected into its common frame. Thereafter, we merge all points that are close in image and descriptor space through replacing them by the averaged keypoint position and the descriptor with the lowest distance sum towards all others within the merge set.
Matching: We start executing DOT matching which returns a rough (quantized) 4-DOF pose, denoted as $\mathbf{H}_4$. Additionally, we calculate a 6-DOF $\mathbf{H}_6$ pose by detecting and matching point features within the segment's RP support projected into the image using $\mathbf{H}_4^{-1}$.

We reject the point matching homography $\mathbf{H}_6$ if it does not comply with $\mathbf{H}_4$ by examining the values of $\tau(\mathbf{H}_4^{-1}\mathbf{H}_6)$, $\phi(\mathbf{H}_4^{-1}\mathbf{H}_6)$, and $\sigma(\mathbf{H}_4^{-1}\mathbf{H}_6)$. In case of sufficiently low values, we initialize the tracking using $\mathbf{H}_{t=0} = \mathbf{H}_6$. Otherwise, we use $\mathbf{H}_{t=0} = \mathbf{H}_4$ but repeat the matching procedure with one of the following camera frames. In case of successful initialization, the homography $\mathbf{H}_{t=0}$ is written forward using $\mathbf{H}_t^{t+1}$ from Section 3.1 while maintaining the support region of the Relevance Plane which results in the homography:

$$\bar{\mathbf{H}}_t^{RP} = \prod_{k=0..t}(\mathbf{H}_k^{k+1})\mathbf{H}_{t=0} \tag{7}$$

Since KLT is also not dependent on point features (only on sufficient rank 2 image gradients within each patch), the method also works with severely occluded or mostly textureless environments.

4 Workflow Tracking

In the two following subsections, we explain how we capture and store hand locations and how this is combined in an extended scoring function.

4.1 2D location maps

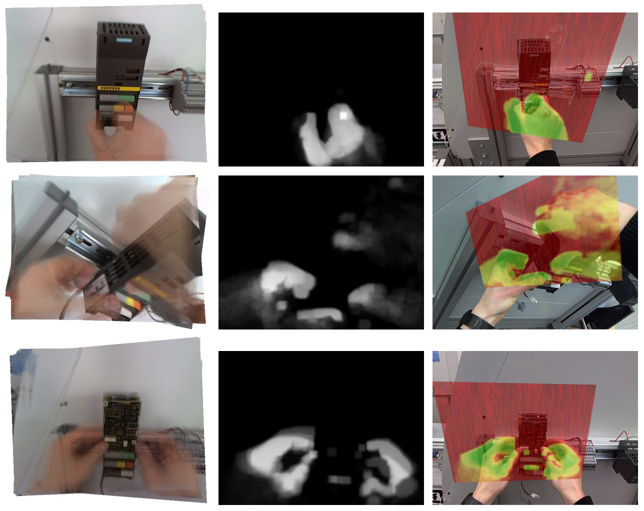


Figure 4: Images projected into common frame and averaged **(left)**, hand location map **(middle)**, hand location map projected into the field of view **(right)**

We store the location probability of the user's hands in a 2D map for each temporal segment separately. We first segment the hand silhouette mask $\boldsymbol{S}_t$ based on skin color segmentation for every image $\boldsymbol{I}_t \in \mathcal{S}_i$. While simple pixel-wise segmentation based on HSV histograms is sufficient for the evaluated scenarios, a more robust substitute for this step is the segmentation procedure from [13]. The location probability map is then the normalized average $\boldsymbol{S}_i^{RP} = \frac{1}{|\mathcal{S}_i|}\sum \text{warp}_{\bar{\mathbf{H}}_t^{RP}} \boldsymbol{S}_t$, where $|\mathcal{S}_i|$ is the number of images in segment i. Figure 4 illustrates this procedure.

We also use this to provide visual feedback by color-coding this map and projecting it into the field of view of the user, compare right column of Figure 4. A very low or zero location probability is indicated as red, low as yellow, and high probability as green. Compare Section 5 for a discussion of this feedback from an application perspective.

4.2 Tracking score

In [18], an image received a classification score for each candidate segment using a k-NN classifier using the robust distance function, reviewed in Section 3.2.1:

$$\text{score}^i_{\text{NN}}(\boldsymbol{I}_t) = \frac{1}{k} \sum_{l=1..k} 1/d(\boldsymbol{I}_t, \text{NN}(l)) \tag{8}$$

where NN(l) denotes the l^{th} nearest neighbor. Using the location maps, we extend the scoring function to

$$\begin{aligned} \text{score}^i(\boldsymbol{I}_t) = &\ \alpha\, \text{score}^i_{\text{NN}}(\boldsymbol{I}_t) \\ &- \beta\, \text{count}(\text{thres}_\kappa(\boldsymbol{S}^{RP}_i) \otimes \text{warp}_{\bar{\mathbf{H}}^{RP}_t}(\boldsymbol{S}_t)) \end{aligned} \tag{9}$$

where α and β are weights and count() is the non-zero pixel count and $\otimes$ denotes the pixel-wise XOR operator.

One important aspect to note is that we do not apply $\text{score}^i_{\text{NN}}(\boldsymbol{I}_t)$ within the normalized frame but in the original image space. This is due to two reasons: (1) Since $\text{score}^i_{\text{NN}}(\boldsymbol{I}_t)$ implicitly has some affine invariance and robustness towards arbitrary local deformation, it also handles a certain degree of perspective distortion. (2) The term also appears in tracking (re-)initialization to determine $\mathbf{H}_4$. To allow an instantaneous reinitialization, we chose to apply it to the image space directly. Otherwise, in case of a tracking loss, the user is required to adopt a valid initialization position. In our framework, we use an attention funnel to guide the user back, if he wanders off too far.

5 Application

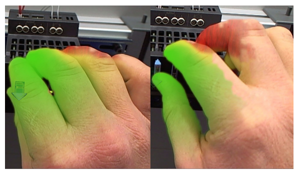
Although the paper is mainly focused on the technical aspects of spatiotemporal tracking, we have also made two contributions to the general use case of AR-based procedural assistance. In this section, we discuss these more application-centric aspects of the approach.

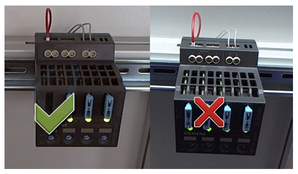
The contribution in this respect is the automatic assessment of correctness indicators and their presentation to the user in the form of visual real-time feedback: *Enactive feedback*, during the psychomotor phase, for which we propose a novel kind of visual representation and *optical validation* that compares the outcome of a work step with a desired target state.

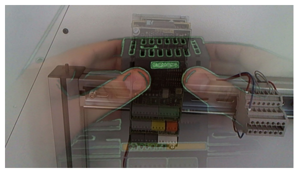
Figure 5 shows examples of the four kinds of visual overlays that are used for guidance, including the procedural and annotational overlays that have already been presented in our previous publication [18]. We will now briefly explain the two new correctness indicators and their technical realization.

Enactive feedback Through back projecting the color-coded location probability maps, described in Section 4.1 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. 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 Figure 5(a) and 4.

In addition to indicating clearly incorrect hand positions, it is also reassuring the user of the fine-grained support through the system. The enactive feedback hereby replaces the procedural overlays during the psychomotor phase to reduce visual clutter of the interface.


(a) Enactive feedback


(b) Optical validation


(c) Procedural overlays

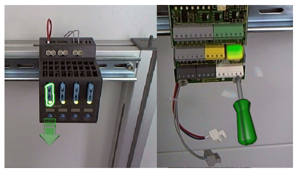

(d) Annotational overlays

Figure 5: Visual feedback provided by the system

Optical validation 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. During run-time, we then perform an optical validation of the observed state after the user has finished the work step. Depending on the outcome, we either acknowledge a correct (green check mark) or indicate an incorrect (red 'x') completion, see Figure 5(b).

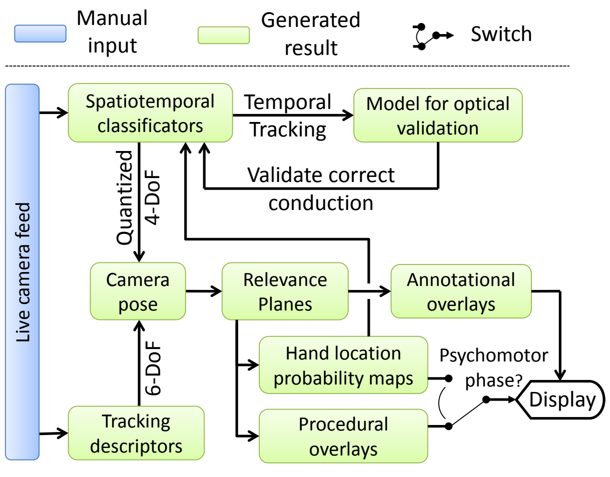
In contrast to [18], where changed regions could only be determined for recordings from a fixed camera, we can now also handle a moving camera. This is achieved by registering the Relevance Planes for the preceding and subsequent static segments using the approach described in Section 3.4 prior to the comparison. We then 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 translational (+6, 0, −6 pixels in x & y direction), rotational (+5°, 0°, −5°), and scaling (90%, 100%, 111%) deviation using brute-force matching of the resulting 81 positions and orientations. It is not straight-forward 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 the matching score accounts for a positive match, we use this as the matching threshold.

The spatiotemporal tracking is reliably identifying when the user has reached a potential target state. Though, it is generally not able to discriminate between the possibly small appearance discrepancies that indicate errors. The pixel-wise comparison of the identified image regions is far more specific in this respect.

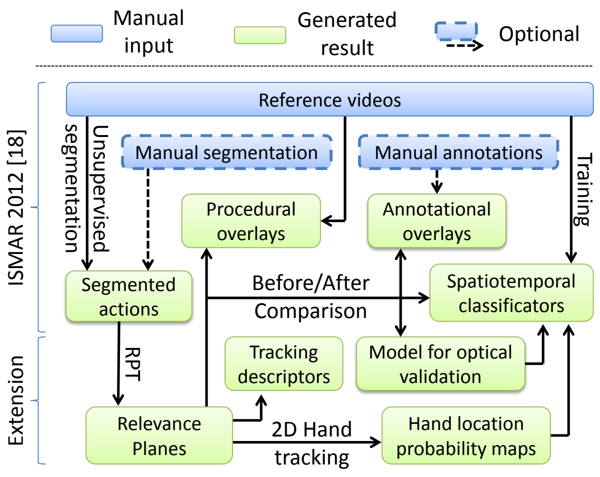
Figure 6(a) shows a schematic overview of the application during run-time, including how the model for optical validation is incorporated. Figure 6(a) shows an analogous overview of the authoring process, marking the extensions covered in this paper.

6 Evaluation

We have evaluated the applicability of the tracking approach in two industrial workflows that were recorded with a head-worn camera and a notebook-maintenance scenario that was recorded with a fixed camera. Our three datasets differ fundamentally in their properties: For baseline, we included the notebook sequence from [18] (Figure 7). Due to the fixed camera, this is a direct evaluation of

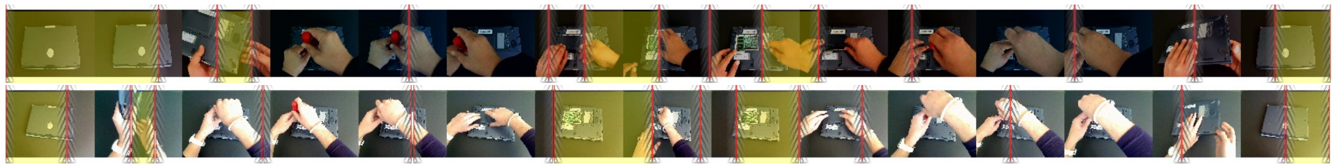


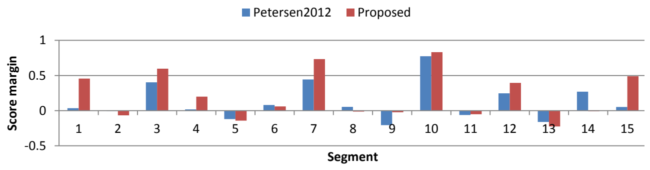
(a) Tracking process



(b) Authoring process

Figure 6: Schematic data flow diagrams for the tracking and the authoring process





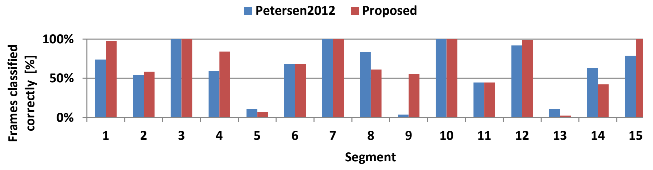


Figure 7: *"Notebook"*: Temporal segmentation of reference and test sequence showing action segments and yellow-tinted static segments and **(top)**. Descriptor score margins and correctness **(bottom)**

the impact of the location descriptor described in 4.1 and we therefore omitted the (trivial) common frames in the figure. The *"Plugs & circuit board"* (Figure 8) exhibits many large planes, coarsely aligned with the image plane and relatively steady camera motion. *"Lever & lid"* (Figure 9), which was also used in [18] exhibits few trackable planes, the angle between the Relevance Planes and the image plane is quite large, and it comprises erratic camera motion.

To analyze the tracking performance, we recorded each scenario twice and used the first for reference and the second for testing. To generate ground-truth, the second recording was manually segmented to exactly match the temporal segmentation of the reference sequence. We then tracked each sequence once with the approach described in [18] and once with our proposed extension. Since maximum vote is used as decision rule in both cases, we were interested in the percentage of correctly classified frames according to this rule and the "confidence" of this decision. Therefore, we measured the *score margin* of frame t, *i.e.,* the score of the correct (according to ground-truth) segment classifier minus the highest adjacent segment classifier: $m^i(t) = \text{score}^i - max(\text{score}^{t+1}, \text{score}^{t-1})$. The number of correct classifications is then given through counting $m^i(t) > 0$. The results are shown in Figure 7, 8, and 9. Additionally, Table 1 lists overall performance numbers.

The *"Notebook"* sequence only slightly improves with the proposed approach. While there are large improvements in certain segments (segments 1, 4, 9, and 15, compare Figure 7), these are evened out by the unchanged or even slightly decreased correctness percentage of the other segment classifiers. The decision margin, though, almost doubles from 12 to 22, which is an indicator for the increased robustness.

The tracking performance for the sequence *"Plugs & circuit board"* increases drastically from 30% to 74% overall correctly classified frames. The score margin was likewise improved from the negative margin -21 to 15.8 and these increases are spread among almost all segment scores, compare Figure 8.

On the other side, the *"Lever & lid"* dataset only marginally benefits from the approach. This is mostly due to the already high tracking score of 76.5%. One interesting aspect is that the according score margin is quite small in both methods: 0.2 and 0.29, respectively. This is owed to the employed dominant orientation templates in combination with highly cluttered background. As the region descriptor only stores the orientations of the k strongest gradients within the descriptor support, much of the cluttered background gets encoded. This leads to the decreased match score margin, as differences in the foreground have less impact.

Additionally, we evaluated the reprojection accuracy in the two datasets recorded with a moving camera. To that end, we first computed the RPT for each segment as described in Section 3.3. Then, we masked the desired region of interest within each common frame as ground truth. This ground-truth annotation is illustrated as the blue grids in Figure 8 and Figure 9, respectively. For every im-

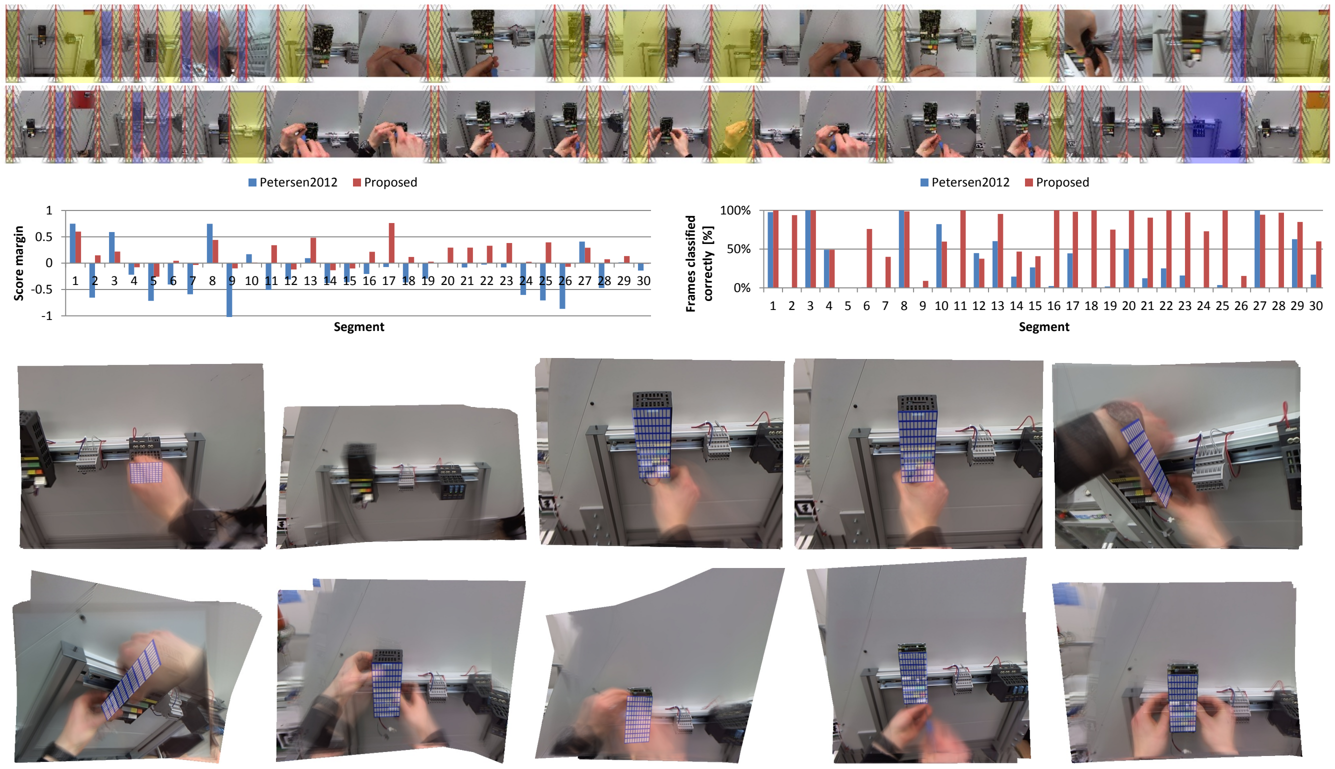


Figure 8: *"Plugs & Circuit board"*: Temporal segmentation of reference and test sequence showing yellow-tinted static-, blue-tinted movement-, and action-segments **(top)**. Descriptor score margins and correctness **(middle)**. Common frames for selected temporal segments **(bottom)**

age $\boldsymbol{I}_t \in \boldsymbol{\mathcal{S}}_i$ projected into the common frame, we selected points $\vec{p}_t$ within the ground-truth mask and tracked them using KLT to get the entire point trajectory $\vec{p}_t$ for every t in the segment. The reprojection error one point $\vec{p}_t$ is then taken as: $e(\vec{p}_t) = \vec{p}_t - \frac{1}{|\boldsymbol{\mathcal{S}}|}\sum \vec{p}_t$ and the overall reprojection error is determined as the average of $e(\vec{p}_t)$ over all selected points and all segments. The results are shown in Figure 10.

The reprojection error is lower for the easier data set *"Plugs & circuit board"*. Over 60% of the pixels reproject into an area of 2 pixels diameter compared to only 30% in the *"Lever & lid"*. In both sets, the tracking error in pixels does not exceed 20, measured w.r.t. to an image of 960×720 pixels. For example videos, please refer to the supplementary material.

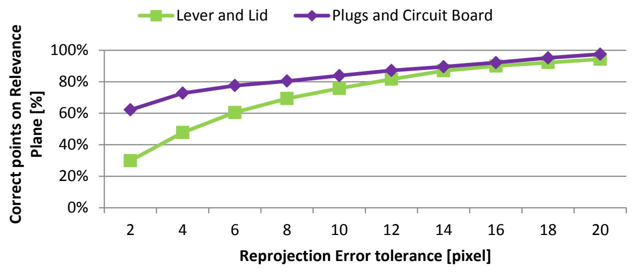


Figure 10: Reprojection error in pixel (image size is 960x720)

7 Conclusion

We have presented a method to model user interaction and accordant change of the environment by temporally segmenting user actions storing descriptors in a series of 2D maps. We have shown and evaluated examples how this can be used in the application of workflow tracking for Augmented Reality manuals that are automatically created from a single first-person view video example.

Table 1: Tracking performance comparison

	Petersen2012		**Proposed**	
Data set	Margin	Correct	Margin	Correct
Notebook	12.2	62.8%	21.6	68.0%
Lever & Lid	0.2	76.5%	0.29	80.5%
Plugs & Circuit Board	-21	30.4%	15.8	74.5%

The higher precision of our tracking approach improves the general experience in terms of higher accuracy of the visual overlays as well as substantially reduced the spatial jitter compared to the approach, described in [18].

In future work, we plan to study effectiveness of the proposed visual feedback during the psychomotor phase with a focus on skill transfer. In this context, we will also concentrate on using hand and finger tracking to further improve the feedback that is given to the user.

Acknowledgements

The work presented in this paper was performed in the context of the Software-Cluster projects EMERGENT and SINNODIUM (www.software-cluster.org). It was funded by the German Federal Ministry of Education and Research (BMBF) under grant no. "01IC10S01". The authors assume responsibility for the content. We would also like to thank the SmartFactoryKL (smartfactory-kl.de) for providing the factory-related scenarios.

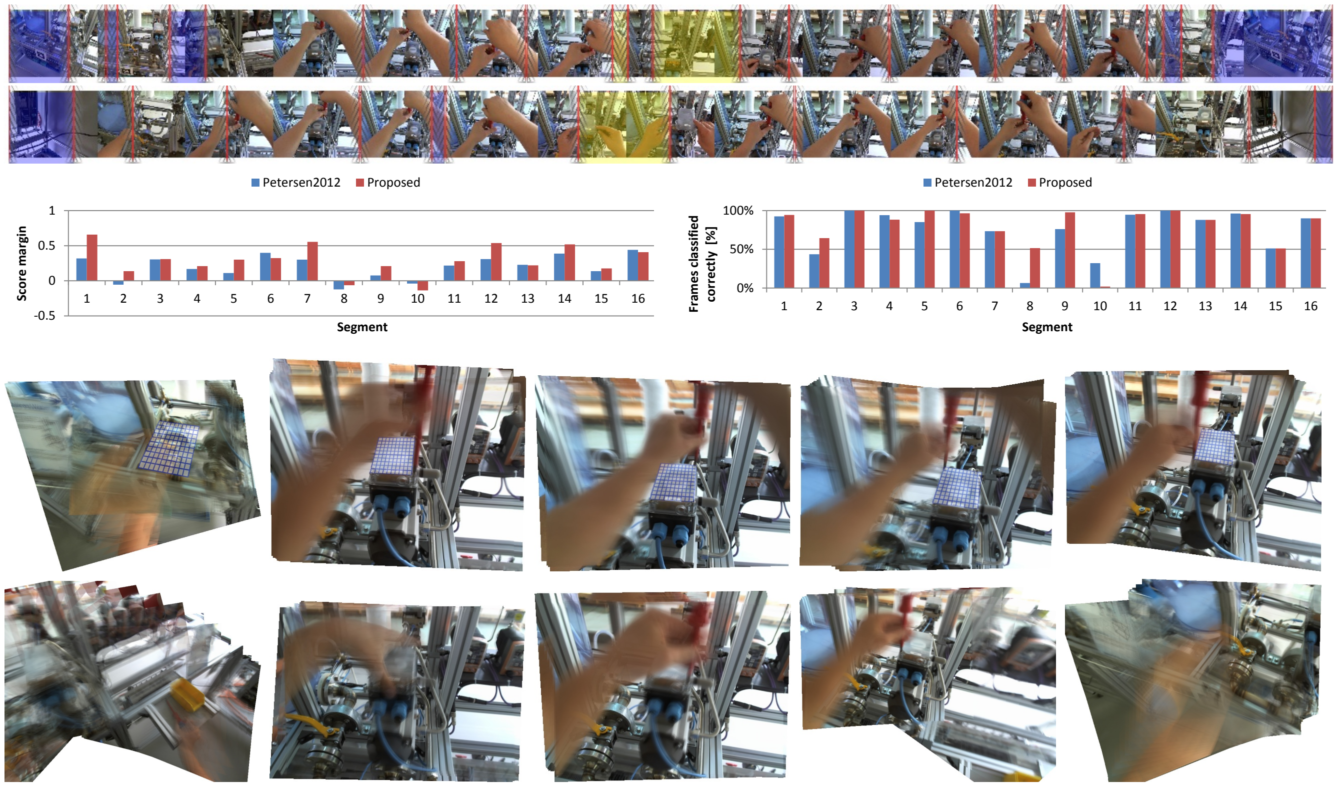


Figure 9: *"Lever & Lid"*: Temporal segmentation of reference and test sequence showing yellow-tinted static-, blue-tinted movement-, and action-segments **(top)**. Descriptor score margins and correctness **(middle)**. Common frames for selected temporal segments **(bottom)**

References

[1] K. M. Baird and W. Barfield. Evaluating the effectiveness of augmented reality displays for a manual assembly task. *VR*, 1999.

[2] M. Calonder, V. Lepetit, C. Strecha, and P. Fua. BRIEF : Binary Robust Independent Elementary Features. In *ECCV*, 2010.

[3] T. Caudell and D. Mizell. Augmented reality: an application of heads-up display technology to manual manufacturing processes. In *ICSS*, 1992.

[4] O. Faugeras and F. Lustman. Motion and structure from motion in a piecewise planar environment. *IJPRAI*, 1988.

[5] S. Gauglitz, C. Sweeney, J. Ventura, M. Turk, and T. Hollerer. Live tracking and mapping from both general and rotation-only camera motion. In *ISMAR*, 2012.

[6] S. Henderson and S. Feiner. Exploring the benefits of augmented reality documentation for maintenance and repair. *Visualization and Computer Graphics*, 2011.

[7] S. J. S. Henderson and S. S. K. Feiner. Augmented reality in the psychomotor phase of a procedural task. In *ISMAR*, 2011.

[8] S. Hinterstoisser, V. Lepetit, S. Ilic, P. Fua, and N. Navab. Dominant orientation templates for real-time detection of texture-less objects. In *CVPR*, 2010.

[9] T. Kanade and M. Hebert. First-Person Vision. *Proc. of the IEEE*, 2012.

[10] G. Klein and D. Murray. Parallel Tracking and Mapping for Small AR Workspaces. In *ISMAR*, 2007.

[11] D. Koller, G. Klinker, E. Rose, D. Breen, R. Whitaker, and M. Tuceryan. Real-time vision-based camera tracking for augmented reality applications. In *VRST*, 1997.

[12] D. S. Kumar and C. V. Jawahar. Robust Homography-Based Control for Camera Positioning in Piecewise Planar Environments. In *LNCS, CVGIP*, 2006.

[13] C. Li and K. M. Kitani. Pixel-level Hand Detection in Ego-Centric Videos. In *CVPR*, 2013.

[14] B. Morkos, J. Taiber, J. Summers, L. Mears, G. Fadel, and T. Rilka. Mobile devices within manufacturing environments: a BMW applicability study. *IJIDeM*, 2012.

[15] U. Neumann and A. Majoros. Cognitive, performance, and systems issues for augmented reality applications in manufacturing and maintenance. In *Virtual Reality*, 1998.

[16] S. K. Ong, M. L. Yuan, and A. Y. C. Nee. Augmented reality applications in manufacturing: a survey. *Production Research*, 2008.

[17] N. Petersen and D. Stricker. Fast hand detection using posture invariant constraints. In *LNCS, KI*, 2009.

[18] N. Petersen and D. Stricker. Learning Task Structure from Video Examples for Workflow Tracking and Authoring. In *ISMAR*, 2012.

[19] J. Platonov, H. Heibel, P. Meier, and B. Grollmann. A mobile markerless AR system for maintenance and repair. In *ISMAR*, 2006.

[20] D. Reiners, D. Stricker, G. Klinker, and S. Müller. Augmented reality for construction tasks: doorlock assembly. In *IWAR*, 1998.

[21] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. ORB: an efficient alternative to SIFT or SURF. In *ICCV*, 2011.

[22] J. Shi and C. Tomasi. Good features to track. In *CVPR*, 1994.

[23] A. Tang, C. Owen, and F. Biocca. Experimental evaluation of augmented reality in object assembly task. In *ISMAR*, 2002.

[24] A. Tang, C. Owen, F. Biocca, and W. Mou. Comparative Effectiveness of Augmented Reality in Object Assembly. *New Horizons*, 2003.

[25] S. Webel, U. Bockholt, T. Engelke, N. Gavish, M. Olbrich, and C. Preusche. An augmented reality training platform for assembly and maintenance skills. *Robotics and Autonomous Systems*, 2013.

[26] F. Wientapper, H. Wuest, and A. Kuijper. Reconstruction and Accurate Alignment of Feature Maps for Augmented Reality. In *3DIMPVT*, 2011.

[27] X. Wu, S. Kamijo, W. Zhang, and M. Sakauchi. Interactive Object Annotation for Construction of Video Information System. In *ISM*, 2006.