Acquiring and transferring workflow knowledge using Augmented Reality

Nils Petersen

Submitted to the
Department of Computer Science
Technical University of Kaiserslautern
for the fullfillment of the requirements for the doctoral degree
Doctor of Engineering
(Dr.-Ing.)

January 2014

Abstract

Workflow knowledge comprises both explicit, verbalizable knowledge and implicit knowledge, which is acquired through practice. Learning a complex workflow therefore benefits from training with a permanent corrective. Augmented Reality manuals that display instructive step-by-step information directly into the user's field of view provide an intuitive and provably effective learning environment. However, their creation process is rather work intensive and current technological approaches lead to insufficient interactivity with the user.

In this thesis we present a comprehensive technical approach to algorithmically analyze manual workflows from video examples and use the acquired information to teach explicit and implicit workflow knowledge using Augmented Reality. The technical realization starts with unsupervised segmentation of single work steps and their categorization into a coarse taxonomy. Thereafter, we analyze the single steps for their modalities using a hand and finger tracking approach optimized for this particular application. Using explicit, work step specific generalization we are able to compensate for morphological differences among different users and thus to reduce the need for large amounts of training data. To render this information communicable, *i.e.,* understandable by a different person, we present the further processed data using Augmented Reality as an interactive tutoring system.

The resulting system allows for fully or semi-automatic creation of Augmented Reality (AR-)manuals from video examples as well as their context-driven presentation in AR. The method is able to extract and to teach procedural, implicit workflow knowledge from given video examples. In an extensive evaluation, we demonstrate the applicability of all proposed technical components for the given task.

Kurzzusammenfassung

Handlungswissen umfasst neben explizitem und verbalisierbarem Wissen auch implizites, prozedurales Wissen, dessen Aneignung durch Übung geschieht. Das Erlernen eines entsprechenden Arbeitsablaufes bedarf daher eines ständigen Korrektivs. Augmented Reality Handbücher, die schrittweise Anleitungen direkt ins Sichtfeld des Benutzers einblenden, bieten ein intuitives und nachweislich effektives Lernumfeld. Bislang ist deren Erstellung allerdings mit hohem Arbeitsaufwand verbunden und die Systeme haben, bedingt durch den grundsätzlichen technologischen Ansatz, eine begrenzte Interaktivität mit dem Benutzer.

In dieser Thesis wird ein umfassender technischer Ansatz vorgestellt, um manuelle Arbeitsprozesse aus Videobeispielen zu erfassen und daraus abgeleitetes explizites sowie implizites Handlungswissen mittels Augmented Reality zu schulen. Die technische Umsetzung beginnt mit der unüberwachten Segmentierung einzelner Handlungsschritte und deren Einteilung in eine grobe Taxonomie. Darauffolgend werden die einzelnen Handlungen durch ein, für diesen Anwendungsfall optimiertes, Hand- und Fingertracking Verfahren auf die genaueren Ausführungsmodalitäten hin untersucht. Durch die präzise Erfassung dieser konkreten Ausprägung des impliziten Handlungswissens, kann der Externalisierungsschritt umgangen werden. Um die hierbei aufgezeichneten Inhalte schulbar, d.h. für einen Menschen wieder erfassbar zu machen, werden die Daten aufbereitet und mittels Augmented Reality in Form eines interaktiven Tutor-Systems dargestellt. Im Gegensatz zum gängigen Stand der Technik auf diesem Gebiet erfasst dieses System präzise den aktuellen Kontext des Benutzers und überwacht und korrigiert Ausführungsfehler. Die Übertragung auf die Morphologie des Anwenders geschieht hierbei über eine explizite handlungsschrittspezifische Generalisierung der Trainingsdaten.

Das entstehende Gesamtsystem ermöglicht das voll- und teilautomatische Erstellen von Augmented Reality (AR-)Handbüchern aus Videobeispielen und deren vollständig kontextgetriebene Präsentation in AR. Das Verfahren ist in der Lage, prozedurales, implizites Handlungswissen aus Videobeispielen zu erfassen und zu schulen. In einer umfassenden Evaluierung wird die Eignung der vorgestellten technischen Komponenten für die Aufgabenstellung nachgewiesen.

Acknowledgements

In the course of creating this work, there is a number of people I would thoroughly like to thank for their efforts and their support.

First I would like to thank my doctoral supervisor Prof. Didier Stricker for his support and ideas during the last years. Further, my thank goes to Prof. Grudrun Klinker for accepting to be the second reviewer of this work and the extra work that is involved.

I would like to thank Gabriele Bleser, Lucia Sireis, and Dirk Petersen for proof-reading and commenting this work. Special thanks go to Gabi for the incredibly thorough review and the valuable comments. Thank you, Lucia, for also proof-reading roughly every paper I have ever submitted and for providing me with nocturnal food during the night-shifts to meet a paper deadline, especially at that time when the ISMAR deadline was not shifted for once.

I would also like to thoroughly thank my colleague Alain Pagani for his support, help, and friendship, interesting discussions, and a Christmas party story I am still recounting. Thank you Philipp Hasper for being a diligent student colleague and for joining in for a very enjoyable CeBIT '13, presenting the first fully functional demonstrator of this work.

Further I would like to thank many colleagues who have supported or taught me something during the last 5 years, most notably Gabriele Bleser, Markus Miezal, Marcus Liwicki, Yulian Pastarmov, and Gustaf Hendeby. In addition to that I would like to thank Ingo Boesnach and Tobias Feldmann as well as my parents for giving me the confidence to start my PhD studies in the first place.

Contents

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. Augmented Reality (AR) which denotes the augmentation of virtual information into the sensory perception of reality has proven to be promising in this regard: Augmented Reality manuals that provide context-aware step-by-step instructions directly in the field of view have been an important use case and selling proposition for AR in general.

Although the conceptual idea for these systems has already been proposed in 1992 [1] and in spite of their often reported usefulness [2, 3, 4], these systems are in no way widely used. Besides ergonomical problems related to the required hardware, significant factors are the complex technical requirements that make their creation sophisticated and expensive. These requirements can be categorized into authoring and tracking:

Authoring is the process of creating visual overlays and associating those with certain object parts or process steps, which requires a script-like description of the task structure. To be able to visualize these overlays on top of the associated object parts, a tracking system is required. Although there are marker-based and markerless approaches, both require intensive planning. Either because markers need to be placed or markerless tracking needs to be trained to detect single parts and object states.

All current approaches require a complex, time-consuming, and scenario-specific creation process with two specific consequences: Due to its complexity, the creation process cannot generally be conducted by a domain expert like a maintenance worker or mechanic but has to be supported by a person with technical knowledge in AR. While this is certainly a cost

(a) Google Glass [5] (b) Vuzix STAR 1200 [6]

Figure 1.1: The intended display devices, including all necessary sensors.

driver the main problem is that the solutions do not scale well with the problem size. Due to the considerable effort to implement a system for a single scenario, it becomes increasingly infeasible to provide AR-based assistance to a growing number of scenarios. Considering there might be hundreds or even thousands of different maintenance workflows at a single factory or garage, the implementation of comprehensive AR assistance is prohibitive following the current state of the art.

1.1 Aims

We aim to provide a set of algorithms to create interactive Augmented Reality assistance systems for procedural tasks from video examples. The goal is not only to grandly mitigate the technical effort but in fact to remove it entirely by eventually allowing authoring from in-situ observation of a workflow. To be of practical applicability, we deliberately avoid excessive instrumentation of the user and constrain the hardware requirements to a very lightweight system consisting of a single consumer-grade RGB camera, a display, and a mobile computer. The low hardware requirements comply with the sensor and performance specifications of current mobile phones, tablets, and most prominently, Google Glass [5], when supported by a remote PC for offloading parts of the computation. Therefore, the only necessary body-worn hardware can be reduced to a very lightweight, integrated combination of camera and display, shown in Figure 1.1.

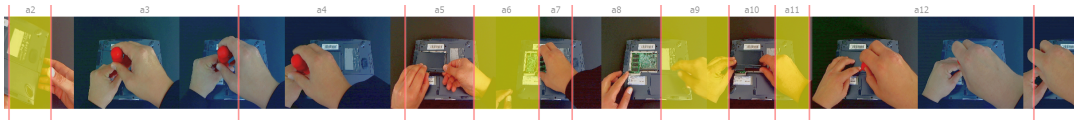
To achieve our general aim, we need to address a number of sub-problems. First, we need to automatically analyze and assess the task structure. This involves discovering the number of work steps that are comprised in the workflow, the possible step orders (which might vary between different performances), and possible variants of the workflow. Additionally, we need to decide whether an observed step needs to be conducted precisely, approximately, or whether

it has been performed unintentionally. Our framework then needs to create expressive visual representations to instruct the user while using the system. The according approach furthermore has to cope with a moving, head-worn camera, which is required for in-situ authoring. Also, to allow for field scenarios, the approach may not require dedicated tracking aids like markers or any kind of control over the infrastructure.

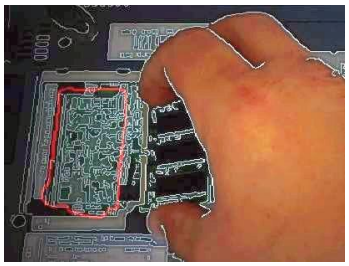
During run-time, we aim to present the information extracted during the authoring phase in a deeply interactive, context-aware form to the user: Current approaches to AR manuals have merely concentrated on migrating the paper manual paradigm to Augmented Reality (*e.g.,* [2, 3, 4]). This has resulted in systems that allow the user to display instructions for a certain work step until the user manually requests the next instruction. The didactic gain of these systems is principally unchanged in comparison to paper manuals with the difference of omitting the cognitive load needed to associate a textual explanation or an instructive sketch with the current work environment.

In our approach we aim at full awareness of the user by tracking his or her progress within the workflow. Being able to track the user's actions allows improving on two fronts: Firstly, to make the running system follow the user automatically while performing the task. Secondly, it allows to provide real-time feedback on the quality or (whenever possible) the correctness of the task execution. Figure 1.2 summarizes and illustrates our main contributions to authoring and run-time usage.

In order to achieve these goals, we do make a couple of assumptions regarding the environment that are however naturally met by many real-world industrial scenarios: We assume that the environment dominantly consists of a single (not necessarily connected) rigid object that is suitable to provide a frame of reference to the tracking system and the procedural model. This reference object may change in each work step, but needs to remain the same during the course of each individual step. The location of the user's activities in relation to this object is taken to be meaningful. For example, fastening a certain screw on a machine is well met by this assumption. Categories of counterexamples would be (1) everything dealing with non-rigid and organic objects, *e.g.,* performing surgery or placing a medical injection, (2) rigid objects undergoing unconstrained out-of-plane rotations that were not observed within at least one of the reference recordings, *e.g.,* the assembly of a hand-held object, and (3) if the rigid object is so small in one or all dimensions that it is not suited to provide a rough camera pose, *e.g.,* assembling a syringe.

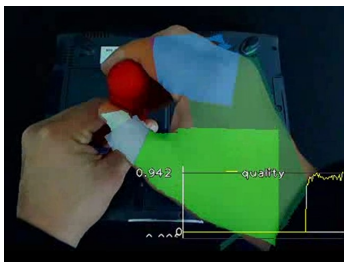


(a) Automatic task segmentation and analysis



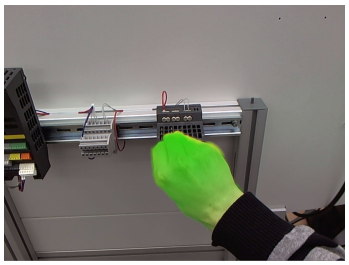
(b) Automatic authoring

(c) Explicit generalization



(d) Content-adaptive hand tracking

(e) Markerless workflow tracking



(f) Enactive feedback

(g) Optical validation

Figure 1.2: Illustration of our main contributions.

What we deliberately do not assume is texture or sufficient structure for using point-features for tracking, neither during authoring nor at run-time. Though, as we do not require any additional sources of geometrical information like CAD models, we require the viewpoint during run-time to be roughly the same as during authoring.

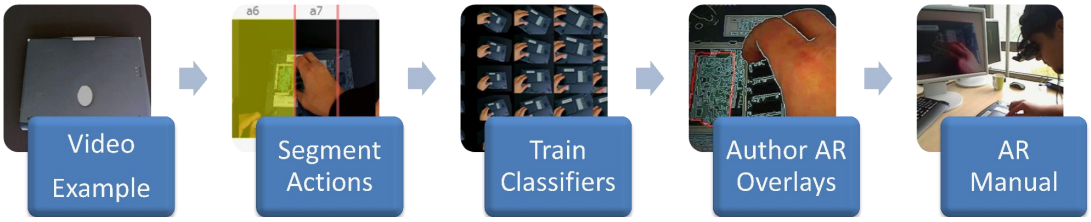


Figure 1.3: Simplified authoring process pipeline.

The simplification of the entire creation process (see Figure 1.3) up to the extent of automation opens up new usage scenarios: For industrial companies, the benefits lie in the much reduced price and the possibility to create AR-based documentation without disclosing confidential assembly steps to an external service provider. Aside from the scope of assistive systems, it can be used to document and protocol conducted work steps, for example in the context of safety-critical maintenance operations. Further applications are found in (1) quality management by instantaneously signaling omitted or incorrectly conducted work steps, (2) control, by automatically triggering certain actions when a work step is conducted, and (3) safety, *e.g.,* by displaying warning messages when hazardous work steps are executed.

Since no tracking aids as well as only commodity hardware are required, the system could be used by consumers to create ad-hoc documentations (*e.g.,* office scenarios like printer maintenance or general smaller assembly tasks). That there is a need for this kind of documentation is clearly reflected in the uncountable do-it-yourself videos found on the web. The topic is becoming particularly relevant (as reflected in [7]) with the current advent of less obstructive, consumer-targeted first-person vision cameras in combination with head-up displays like the Vuzix M100 and most prominently Google Glass. While it is hardly possible to foresee, whether these will have a permanent impact on the consumer market, the systems set a new standard for ergonomics of related hardware and substantially lower the entry price.

From a conceptual perspective on Augmented Reality, this work includes the user's context as a major driver in the perception of reality by incorporating temporal tracking and action recognition into the information selection process. This is an extension to the widespread

definition of Augmented Reality [8] that is focused on the spatial association with objects as the sole cue and driver for interactivity.

1.2 Approach

We build upon unsupervised temporal segmentation of the reference video sequences as a first step. Since we aim for manual workflows viewed from the user's perspective, we generally have to deal with close-up images and 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 straight forward to formulate suitable compactness criteria based on this.

We use whole-image distance or more general dissimilarity functions of the sort $d(\boldsymbol{\mathcal{S}}_i, \boldsymbol{\mathcal{S}}_j) \rightarrow \mathbb{R}$, where $\boldsymbol{\mathcal{S}}_i$ and $\boldsymbol{\mathcal{S}}_j$ are two arbitrary images of an ordered image sequence $\boldsymbol{\mathcal{S}}$. In order to cope with lighting changes and small perspective deformations, $d(\boldsymbol{\mathcal{S}}_i, \boldsymbol{\mathcal{S}}_j)$ is implemented using the DOT region descriptor [9]. 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.

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 shortest-path, *i.e.,* finding a set of frames with the least amount of novelties that connects between the start frame and the current 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.

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. 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.

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 from a single reference performance, we propose an image-based rendering (IBR) approach to explicitly generalize the reference material through a model-guided approach. Figure 1.4 shows a schematic of the data flow during the authoring process.

The IBR approach is further used as a 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. Using particle swarm optimization (PSO) to solve the proposed function we are able to estimate the 26 DoF posture of the hand. We use this to gain a 3D understanding of hand positions and postures that are characteristic for a certain task and to identify grasping positions. Further, we are able to identify work steps that need to be processed accurately and distinguish important steps from erratic motion by comparing several reference recordings.

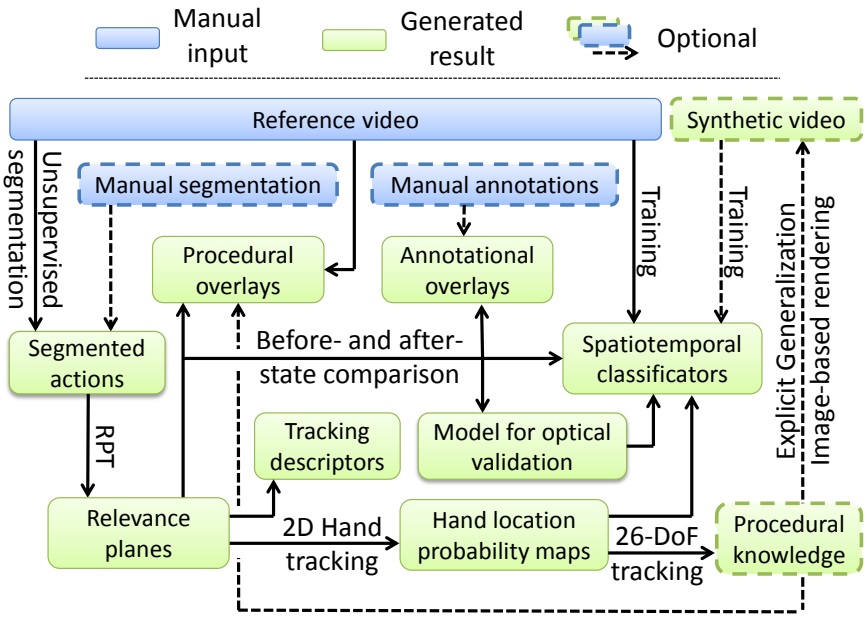


Figure 1.4: Data flow diagram of the authoring process: After applying the relevance plane transform (RPT) to the distinct segments, the workspace state before and after each user action is used to train the classifiers and to process the visual overlays. Further, the sequence is analyzed using hand and finger tracking to provide enactive feedback and to explicitly generalize the training data through image-based rendering.

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. Figure 1.5 shows examples of the provided visual feedback and Figure 1.6 illustrates the schematic data flow during run-time.

We carefully designed the entire approach to not crucially depend on a fragile high level feature or preprocessing 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

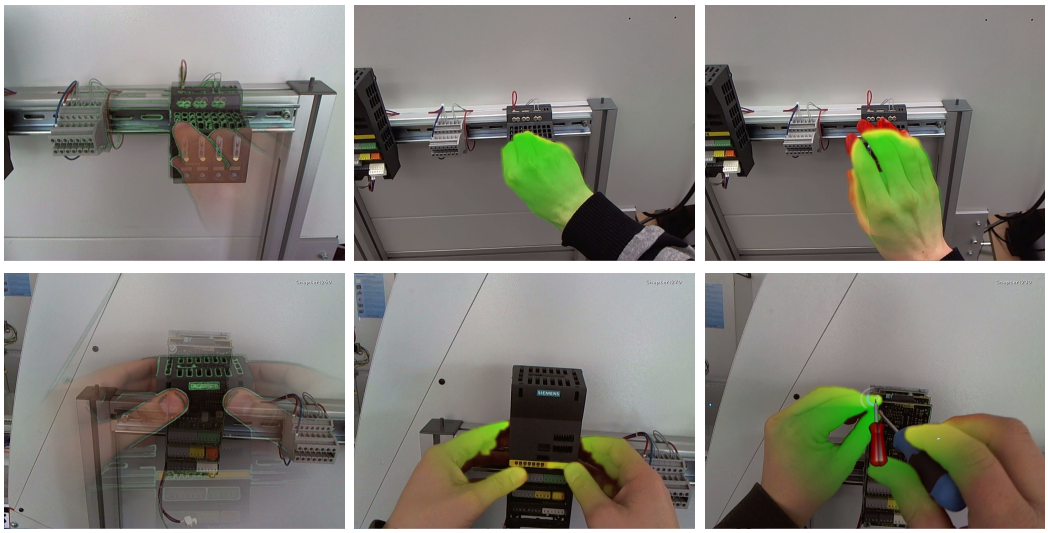


Figure 1.5: Examples from an automatically authored AR-manual: The half-transparent overlays (left column) were automatically extracted from the reference sequence. The green coloring indicates that the current step is conducted correctly, red indicates a wrong posture or position. The augmented tools have been manually added (lower right).

hand tracking results and the approach to camera tracking that improves with the availability of point features but is not dependent on it.

To accommodate for incomplete training data, we also consequently support incremental learning to adapt and extend the models during usage. Again, this is most notable in the proposed approaches to camera and hand tracking but also valid for the temporal workflow tracking. The entire framework affords to continue to adapt to the observation during run-time. 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.

1.3 Organization of the thesis

After having presented our principal aims and summarizing our approach, we will conclude this chapter with an overview over the thesis structure, followed by a comprehensive list of the contributions of this work in the following section.

In Chapter 2 we are reviewing related literature for the various conceptual and technical aspects of our work. The discourse starts with a brief overview over topics that are important for Augmented Reality in general. We then survey previous applications and surfaced concepts,

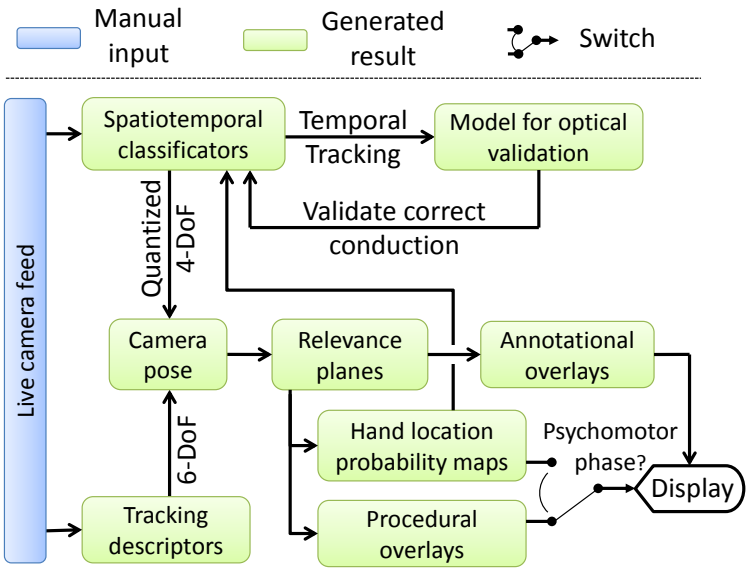


Figure 1.6: Data flow diagram of the run-time process: Using the set of classifiers, the temporal segment is determined. When possible, the resulting rough camera pose is refined using point descriptor matching and used to back project the relevance planes, in order to display the overlays.

related to our approach. As our work touches rather widespread fields of research, we have categorized the technical aspects into activity recognition, descriptors and models, and hand tracking.

In Chapter 3 we present our temporal segmentation approach based on image distance functions. The presentation begins with a detailed introduction to how we define robust dissimilarity functions based on region descriptors. After that, we show how these functions are extended to the proposed frame-to-frame distance measure, the shortest-path distance and how we use this distance measure to derive online evaluable compactness criteria for unsupervised temporal segmentation. Then, the approach used to infer the focus of attention is explained and the chapter concludes with the evaluation of the segmentation performance and its repeatability.

Chapter 4 describes our approach to camera and temporal tracking as well as the offline temporal alignment of multiple reference recordings. We first introduce the so-called *relevance plane transform* in detail that projects the time-evolving 3D workspace onto a series of distinct, spatially continuous 2D frames. We continue describing the 2D region descriptors and the hand location maps that are applied to the 2D frames and used for temporal tracking. While this concludes the online tracking (for tracking the user while performing the task), we show how

we can improve at the offline case, *i.e.,* temporally and spatially aligning multiple recordings to extract workflow variants. The chapter ends with an evaluation of the tracking and alignment performance.

Chapter 5 describes our proposed hand tracking approach to extract key postures, trajectories, and velocities of the user's hands during each work step in order to analyze and explicitly generalize the recorded reference material to reduce user dependency. First, the image-based appearance model is presented that is also used to synthesize additional reference material for the model guided explicit generalization. After that, we introduce the adaptive tracking approach that is used for tracking the user's hands with a kinematic model with 26 degrees of freedom and allows adapting to the observed material. Again, the reconstruction accuracy and tracking performance, as well as the results of explicit generalization are evaluated at the end of the chapter.

In Chapter 6 we explain how we sample instructive snippets from the video, and extract and generate the set of overlays, completing the tool set for the automatic authoring of AR manuals. We first discuss some general properties and difficulties related to Augmented Reality, and then explain the extraction of procedural overlays that illustrate the current instruction, and the generation of annotational overlays. We then lay out our approach to provide interactive visual feedback on the correctness of the task execution during run-time. To enable domain experts to extend the set of augmentations, we present our authoring tool that entirely abstracts from all 3D considerations typically required for authoring AR content.

Since we used a very dedicated approach to realizing the entire framework, which has a crucial impact on performance and portability, we cover the implementation aspects in Chapter 7. We present our approach to automatic parallelization by means of a component-based, data-driven programming model. After that, we show how we exploited some of the further properties of our approach, in order to systematically study different workloads between a remote server and a mobile client. We give an overview of the building blocks that comprise the presented framework and extensively evaluate the run-time behavior.

The thesis ends with a discussion of the results in Chapter 8 and the conclusions drawn from it. This chapter also identifies directions of future work and straight-forward extensions also aside of the principal use case of workflow assistance.

1.4 List of Contributions

The most important conceptual contributions of this work are the two following. Firstly, in extension to the general paradigm of AR, where augmented information is only spatially associated with real objects, we add context, visually inferred from the user's activities as a major driver for interactivity in the application. Secondly, we contribute a novel approach to the authoring process of procedural assistance systems through analyzing example videos of according activity sequences. To the best of our knowledge, this is the first system to achieve such a level of user-awareness in any Augmented Reality application without using additional tracking aids. This contribution is important, as it strongly alleviates the content creation problem that has been a key challenge for procedural assistance with AR.
The technical contributions in pursuit of these main contributions are

- A framework to derive robust, online evaluable criteria for unsupervised temporal segmentation from image dissimilarity functions. In principle, our segmentation framework is not limited to a computer vision context. It could be generally applicable to derive compactness criteria when no suitable distance function for arbitrarily distant entries in time series data exists.

  The presentation is based on the conference paper:

  > Nils Petersen and Didier Stricker, *Learning Task Structure from Video Examples for Workflow Tracking and Authoring*, in the Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR), 2012

- 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. This results in a piecewise two-dimensional, spatiotemporal model of a dynamic, changing environment.

  The presentation is based on the conference paper:

  > Nils Petersen, Alain Pagani, and Didier Stricker, *Real-time Modeling and Tracking Manual Workflows from First-Person Vision*, in the Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR), 2013

- A comprehensive method to automatically generate Augmented Reality manuals from video examples, comprising
  - The segmentation of states and state transitions.
  - An approach for authoring descriptive AR overlays.
  - The identification of object states before and after manipulation for optical validation.
  - Establishing a tracking model using explicit generalization to mitigate user dependency.
- A method to automatically assess certain correctness indicators and means of visualizing them to the user during the workflow.

  The presentation is based on the conference papers:

  > Nils Petersen and Didier Stricker, *Learning Task Structure from Video Examples for Workflow Tracking and Authoring*, in the Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR), 2012
  >
  > Nils Petersen, Alain Pagani, and Didier Stricker, *Real-time Modeling and Tracking Manual Workflows from First-Person Vision*, in the Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR), 2013

- A content-adaptive hand tracking scheme, based on an image-based appearance model, which is used to analyze the workflow video and to generalize the tracking model, comprising
  - An extension to billboard-rendering that we call 2.5D billboards that well describes ellipsoid 3D objects that is used to reproduce hand and finger segments.
  - An efficient morphing technique to minimize ghosting and preserve shape in presence of elastic deformation and model alignment errors.
  - An objective function resulting from this approach that significantly outperforms state of the art methods on RGB images.
  - The design of a database structure to store tracking-related information, leading to an increasing adaption to the image content over time.

  - A method to quickly establish locally optimal search-trees within this database for each tracker state allowing the system to run at interactive frame rates in spite of very large databases.

The presentation is based on the journal article:

> Nils Petersen and Didier Stricker, *Morphing Billboards - An Image Based Appearance Model for Hand Tracking*, In Computer Methods in Biomechanics and Biomedical Engineering: Imaging & Visualization (accepted), 2014

and on the conference papers:

> Nils Petersen and Didier Stricker, *Morphing Billboards for Accurate Reproduction of Shape and Shading of Articulated Objects with an Application to Real-time Hand Tracking*, in the Proceedings of Computational Modeling of Objects presented in Images (CompImage), 2012 **(Best paper award)**
>
> Nils Petersen and Didier Stricker, *Adaptive Search Tree Database Indexing for Hand Tracking*, in the Proceedings of Computer Graphics, Visualization, Computer Vision and Image Processing (CGVCVIP), 2012
>
> Nils Petersen and Didier Stricker, *Fast Hand Detection Using Posture Invariant Constraints*, in the Proceedings of Advances in Artificial Intelligence (KI), 2009

- A method to follow the actions of the user while performing a workflow, allowing context-aware playback and temporal alignment of multiple workflows for learning purposes.
- A robust, markerless camera tracking approach that deteriorates gracefully with lack of image features.
- A highly selective presentation technique that adapts the amount of visualized information to the current user context and the current phase of execution.

  The presentation is based on the conference paper:

  > Nils Petersen, Alain Pagani, and Didier Stricker, *Real-time Modeling and Tracking Manual Workflows from First-Person Vision*, in the Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR), 2013

- An implementation method that supports auto-parallelization and load balancing in combination with a mobile device without changing the program definition.

  The presentation is based on the conference papers:

  > Nils Petersen, Julian Pastarmov, and Didier Stricker, *ARGOS - a Software Framework to Facilitate User Transparent Multi-threading*, in the Proceedings of the MARC Symposium, 2011
  >
  > Philipp Hasper, Nils Petersen, and Didier Stricker, *Remote Execution vs. Simplification for Mobile Real-time Computer Vision*, in the Proceedings of the International Conference on Computer Vision Theory and Applications, 2014

Previously unpublished results are:

- An approach for the explicit generalization of the training data using image-based rendering.
- An approach for content-adaptive hand tracking using image-based rendering.

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.

2

Related work

In this chapter we will discuss previous approaches and thematically related work with respect to the various aspects of our work. We start with an introduction to task assistance using Augmented Reality, which includes an overview of related applications. After that we discuss general presentation concepts and our main field of contribution, the authoring process. Following this, we review related work for aspects that are technical prerequisites of our approach. This begins with a more technical view on the state of the art in unsupervised, supervised, and real-time activity recognition in Section 2.3. In Section 2.4.2 we discuss image cues and features that we use to model the dynamically changing scene in the course of a workflow. As the most prominent cue for monitoring manual activities is the movement of the user's hand, we review the state of the art in hand and finger tracking in Section 2.5.

2.1 Augmented Reality for task assistance

Augmented Reality (AR) denotes the overlay of virtual information onto the user's perception of the real environment. In principle, auditory or haptic feedback [10, 11] may serve as modalities of augmentation but in this work, the term AR is used synonymously with visual augmentation. There has been extensive work on the general use case of procedural assistance using Augmented Reality since the early work of Caudell and Mizell [1] promoted this use case for head-up displays, thereby coining the term itself.

Classically, this is realized through head-mounted displays (HMD) that directly augment the user's field of view. Though, due to the ergonomic shortcomings of head-worn displays, this kind of presentation still plays a niche role. The principle issues with head-worn displays

are addressed by a new generation of highly integrated, lightweight devices like Google Glass [5] or Vuzix M100 [12] that trade certain display properties for ergonomic acceptance. While these displays do not directly augment the field of view but rather display information in the peripheral vision of the user, they are absolutely suitable to be worn for an extended period of time.

In the next subsection, we will briefly introduce the current research and commercial landscape, discuss the various display approaches applicable to our use case, and then survey applications of AR related to task assistance in Section 2.1.3. Following to that, we review presentation and design approaches and in particular approaches for creating the necessary content and tracking models in Section 2.2.

2.1.1 Development landscape

The general topic of Augmented Reality was subject to several large scale research projects such as the ARVIKA project [13, 14], followed by ARTESAS [15], AVILUS/AVILUS+ [16, 17, 18], and the upcoming project ARVIDA. In addition to these projects that are largely dedicated to AR, the use case of workflow assistance was pursued in several related research initiatives. Examples would be the Cognito project [19], the Skills project [20], and even thematically far off projects like the software cluster [21] that is mainly focused on business software.

At the very latest since the appearance of smartphones, AR is also of commercial interest beyond industrial pilot projects. The so-called *Reality Browsers* like Wikitude [22], Layar [23], and Junaio [24] were the first large-scale, general audience commercial offers of Augmented Reality. The technology has the potential of high-order growth and market research firms have already identified the mobile Augmented Reality market as a stand-alone market [25]. Independent from its own commercial success, the consumer-targeted Google Glass will further promote applications and substantially raise the general awareness and acceptance of AR.

2.1.2 Display approaches

There are basically four configurations possible to display AR content:

ST-HMD See-through head-mounted display, *e.g.,* Vuzix STAR 1200. This configuration can be further subdivided into optical see-through (OST), using a half-transparent display and video see-through (VST), with mediated vision using a camera close to the gaze direction.

PV-HMD Peripheral view head-mounted display, *e.g.,* Google Glass, highly related to the general category of head-up displays.

HHS Hand-held screen with rear-side cameras, *e.g.,* smartphone/tablet. Hand-held displays follow the metaphor of augmenting the world like a *magic lense* to look through.

SS Stationary screen with unconstrained camera positioning, if both screen and camera are directly facing the user, this configuration is also called *magic mirror*.

The approaches have different strengths and weaknesses. Although ST-HMDs are closest to the ideal notion of Augmented Reality, they suffer from ergonomic issues, since the user's vision is permanently obstructed by the display. In case of a VST-HMD, where the user perceives the environment through a camera, geometrical offset and time delay due to processing may lead to symptoms, similar to motion sickness. Using an OST-HMD, the user sees the actual environment. The problem is that the optics of most current devices display the virtual content at a fixed focal distance, typically set to several meters away. This is a problem for applications like task assistance, as the focus plane is within hand's reach of about 30-60 cm. This means that the eye is not able to simultaneously focus on the environments and the virtual augmentation and needs to permanently reaccommodate, referred to as the *dual focus problem*. This leads to a benefit of video see-through (VST). As the virtual and real information get composited into a single image, the VST approach leads to a more seamless experience, in particular towards the eye accommodation issue. Although the PV-HMD configuration equally suffers from the accommodation issue, the impact on ergonomics is not as severe, as the user's view is not obstructed by unfocused visual clutter.

The impact on ergonomics was covered in several related studies. The authors of [26] evaluated the impact of hardware and tracking precision on ergonomics in an industrial context. Recently, a similar study was performed by [27], comparing different HMD setups and AR-based presentation to traditional instructions. The results indicate a generally good result of the AR-based presentation and the user's preference towards a monocular HMD instead of a binocular one. The issues with ergonomics are further increased with the potential need for additional sensors. Figure 2.1 shows a current setup from [28] which combines an RGB camera adjacent to the gaze direction, an over-head mounted Kinect camera and a body-worn IMU sensor network for body movement reconstruction. While the setup delivers comprehensive sensory data to follow and instruct the user during a workflow, it is prohibitively complex for everyday usage.

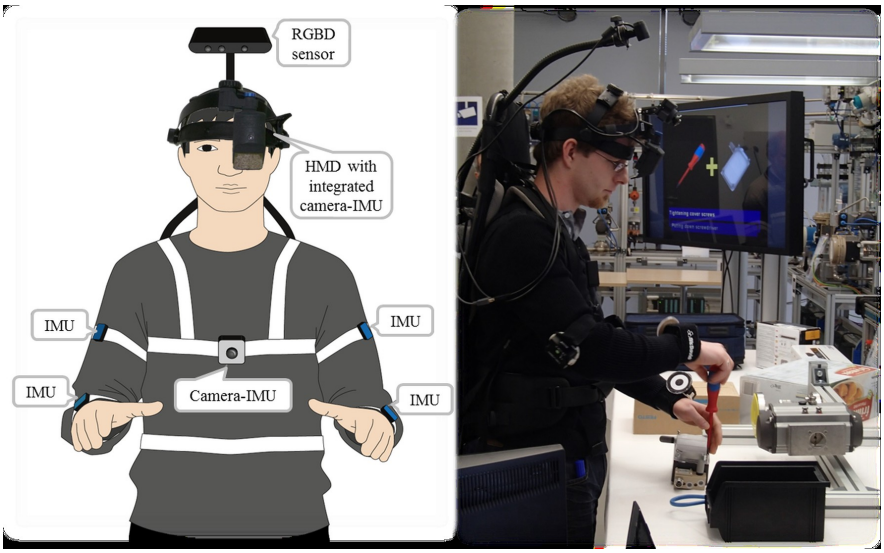


Figure 2.1: Required instrumentation of the user for the approach developed in the Cognito project [19]. The system incorporates readings from a body-worn IMU sensor network, a chest-worn spherical camera, a head-mounted front-facing camera, and an RGBD camera mounted over the head, looking down.

In order to circumvent the ergonomic shortcomings of current HMDs, several approaches employ stationary [29, 30, 31] or hand-held displays [32, 33]. For the case of hand-held displays, the benefits are the easy handling and, of course, the large spread and established user acceptance of the most prominent representatives: smartphones and tablets. The disadvantage is that this hinders hands-free operation. Stationary screens solve this issue but are typically constrained to one physical position and therefore not suited for mobile applications.

Congruent displays need to be calibrated in order to allow for a pixel-precise overlay of virtual information. For OST-HMDs, the calibration for the wearer's visual system requires to actively align real-world landmarks with virtual ones, *e.g.,* single-point active alignment [34]. This process can be supported by a pre-calibrated sensor for calibration, typically a HMD-mounted camera [35] or by a camera looking through the OST optics, directly [36].

2.1.3 Applications and case studies

The assistance of manual manufacturing processes has already been identified as one of the most important fields of application in the fundamental work of [1], which has also coined the term *Augmented Reality*. Since then, a variety of related work demonstrated the applicabil-

Aerospace	[37, 38, 39, 40, 41]	Automotive	[30, 42, 43]
Data center	[44]	Electrics	[32]
Furniture assembly	[45]	General factory	[46]
Instrument playing	[47]	Library services	[48]
Medical	[49, 50, 51]	Military	[4, 52]
Warehouse	[53, 54]		

Table 2.1: Application domains for Augmented Reality assistance.

ity and advantages of AR-based assistance systems in industrial environments within various domains, see Table 2.1.

These systems assist various different industrial tasks, mostly assembly and maintenance [43, 55, 56, 57] but also process control [32, 58], inspection and quality control [33], picking tasks [53, 54], and virtual discrepancy checks [59]. The tracking is mostly still based on fiducials [60] or tracking of tools [61, 62]. However, there also exist markerless approaches based on CAD models [63].

Besides the usage of AR as a method for a computer-aided instruction (CAI) system, there also exist additional applications in the manufacturing or assembly domain, *e.g.,* manufacturing planning [64]. Here, the user can plan the placement of the various machines and production systems using markers and review the final layout in Augmented Reality. Another category includes general supportive systems that may also be combined with a CAI approach. For example, [65] aims to facilitate understanding by augmenting occluded tools when operating a CNC (computer numerical control) machine. Henderson *et al.*in [66] use hand tracking and locate graphical user interface (GUI) elements on salient parts of real objects in AR to make use of their haptics. Particularly interesting are systems that combine and augment the classical paper paradigm with interactive functionality through AR, like [67, 68], though again not in an immediate context of CAI. The combination with the approach described in [68] is one of the directions of future work and will be sketched in Section 8.2.3.

A strong research emphasis was set on the comparison to classical instructional methods like paper manuals and videos [2, 4, 51, 69, 70] which clearly demonstrates the benefits of AR but also discovers several principle-related difficulties, which we will discuss and address in Section 6.1. Already when using mobile devices, aside from the immediate AR focus, there is a trade-off between benefits and risks, as [30] point out in their case study on mobile device usage

in the automotive manufacturing domain. [50] analyze the effect of augmented visual feedback from a virtual reality simulation system on manual dexterity training. The study investigates the impact of Virtual Reality (VR) feedback on the learning process of novel motor skills for novice dental students. The study shows a significant overall improvement for the test group training with the VR feedback if combined with tutorial input. One clear benefit that many studies (*e.g.,* [4, 71]) identify is reduced physical strain, when using an AR setup.

Since the field of related studies is quite vast, several publications focus on summarizing the field or identifying current and future research directions. A survey of Augmented Reality in manufacturing applications is found in [72, 73], whereas [8, 74] contain a more general overview over the topic itself. Further categorization methods for AR are found in [75], as well as in [76] for multi-sensory AR, and most recently [77] with a focus on applications. Furthermore, several authors [78, 79, 80] have sketched ideas and concepts that illustrate future research paths and applications on the way to a general adoption, while [81] lists the ”lessons learned” in previous activities.

An interesting further reading is also [82] that defends the general idea of procedure following, which is key to the step-by-step paradigm adopted by most approaches.

2.2 Design and authoring concepts

In this section, we will discuss the current approaches for realizing Augmented Reality manuals. The section first covers all the presentational aspects and afterwards the conceptual and technical approaches to creating the necessary content and models.

2.2.1 Presentation and interaction

Two aspects define the experience when using interactive manuals: the explanatory power of the presented content and the means of interaction. In the following, we discuss the current approaches and the literature in thematically related fields.

The explanatory power of the presentation, *i.e.,* the ease of understanding an instruction, is obviously a very important factor. There is a large body of literature on design guidelines for AR presentations and visualization [27, 83, 84, 85, 86], procedures [87, 88], and visual communication [89].

For workflow assistance, the dominant procedure employed by virtually every assistance system is based on a step-by-step presentation. This denotes that for each step, separately,

the system communicates the intended action along with an indication of associated locations. While this not only intuitively seems to be the straight-forward approach, it is also backed by scientific examination [82]. Though, the approaches differ greatly in the actual means of presentation and level of detail. As we intend to automate the entire process of creating interactive AR-based manuals, including the authoring of instructive overlays, a thorough study of the available principles is indicated.

The visual presentation may consist of isolated occurrences or combinations of text [2], diagrams [90], pictures [2], arrows [52], videos [91], and guiding cues, e.g. to guide the user's attention to a certain location or vantage point [53, 92]. Further, approaches may include side-information like the necessary tools or equipment for a step [42, 52]. Generally, the displayed information can be categorized into congruent (associated with objects using camera tracking) and contextual (no tracking).

An example of a presentationally rich AR training system is described in [93]. The authors 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.

Due to the high costs (discussed in the following section) of authoring instructive visual assets to illustrate a work step graphically, many approaches rely on textual information with only visualizing a locational or directional graphical annotation. Some more practical systems complement the AR experience with technical 2D or 3D sketches and video sequences illustrating a certain work step, *e.g.,* [93]. The presentation of additional content, not following the AR presentation principle, comes at the ergonomical costs of an increased cognitive burden, as indicated by [4, 52, 69, 94]. Nevertheless, there exist approaches that incorporate explosion diagrams in Augmented Reality [95] as well as methods to alleviate the creation process [96, 97].

The authors of [91] propose to overlay the previously recorded video directly onto the AR-workspace. Users in their study reported that the overlaid video instructions were easy to very easy to follow. We propose a combination of video-overlays and additional, spatial annotations. The topic of multimedia teaching and learning is widely researched and also covered in "hands-on" literature, *e.g.,* [98]. [99] investigate the so-called *Multimedia effect*.

Their findings indicate that the understandability of procedural tasks strongly benefits from adding multimedia information like text and pictures. In particular, this effect is strongest for procedural tasks, compared to conceptual or causal tasks. Further, the effect of animation and graphics were studied by [100] and there is a strong body of general research on cognitive aspects during manual assembly, see [31] for further reading.

Due to a general predetermination as visual or verbal learner (as studied by [101]) there is work that investigates how to combine language and visuals [102], studying the impact of verbal presentation [103], and comparing pictorial [104] or animated [105] demonstrations to texts and following a user-centric approach [106] to identify according design principles in a user study.

Also for basic, schematic overlays such as arrows there are several studies. [107] investigate arrows as diagrammatic, explanatory devices in technical sketches. In particular, their findings show that when users are asked to annotate sketch-based instructions, arrows are frequently used as an element to explain the functional nature of a machine part, whenever there is a dynamic, time-progressing aspect involved. [108] investigate diagrammatic communication with schematic figures.

Few approaches provide real-time feedback to the user during the actual psychomotor-phase. Henderson *et al.* [109], use object-attached markers to provide visual feedback for precisely aligning components. Kotranza and Lind [49] propose a Mixed Reality training system for training clinical breast exams that display the palpation pressure, which is read in real-time from pressure sensors within the training dummy.

The control of the system, foremost the navigation to the next procedural step, is most often effected by means of manual triggers like button presses [91], speech commands [33], or certain gestural changes of the gaze direction [110, 111]. There are few exceptions that perform the transition to the next procedural step autonomously and the author is not aware of a single system that provides the same level of autonomy and integration as the work presented in this thesis without using markers.

The approach that comes closest is [109] that relies on fiducial markers for tracking and derives the procedural state from the appearance and movement of markers. In contrast to this, our approach is able to autonomously follow the user through a workflow using solely vision-based action recognition.

2.2.2 Authoring and model creation

Several groups have presented graphical tools for authoring, [45, 112, 113]. The authors of [45] propose an interesting low-cost Mixed Reality approach using a computer mouse with fiducial markers as tangible tool. The user can use this tool to set up the location, orientation, and scaling of a certain AR-annotation or record an animation by moving it in free space. [114] continues this work and describes a more comprehensive authoring tool including guided calibration procedures. The general idea of a mixed reality application development framework is also described in [115]. There also exist approaches for alleviating certain sub-problems in content creation such as the early work in [116] for automated pictorial explanations, 3D exploded view diagrams [117], or automated assembly planning [118].

Furthermore, several approaches have been proposed using non-graphical authoring and development tools. Standard formats have been proposed or adapted to this end, like VRML and X3D [119, 120] with dedicated frameworks for display and interaction, *e.g., Instantreality* [121]. [122] have presented an XML-based authoring language that also comprises interaction aspects. In [123] a hierarchical representation of assembly steps is proposed as input for authoring.

An alleviation of the authoring process to a state that allows domain experts rather than AR experts to conduct the content creation is not only interesting for cost reduction. That domain knowledge is beneficial for creating understandable and didactic teaching material is not only an intuitive assumption but also scientifically supported. [124] investigated the impact of domain knowledge in a proband's ability to identify usability problems. They compare novice evaluators, regular specialists (with competence in the general field), and double specialists (with competence in the field and the specific domain). Their findings indicate that the double specialists performed best at identifying usability problems, which underscores the advantages of closely involving the domain expert in the content creation.

We will review three publications in more detail as they are representative for the three different approaches to the simplification of the authoring process:

Simplified manual authoring: The authors of [112] address the sophistication of manual authoring of AR documentation. They propose a set of predefined basic animated overlays (*e.g.,* release screw or connect-tool-with-part) that can be positioned in 3D using an editor or a script. The problem is that although authoring of standard tool interaction is alleviated in some respect there would still be a need for workflow-specific 3D modelling of object parts.

Additionally their approach does not address the tracking aspects at all, as it builds on markers for spatial tracking with user-triggered step progression.

Domain-specific automatisms: Given the 3D models of an assembled object and all contained parts, the system described in [55] automatically generates assembly and disassembly instructions. These are deduced from the geometry data using path-planning. During run-time, the system is able to track the object using markerless tracking with edge, junction, and point features. This strongly alleviates the authoring overhead for certain assembly workflows, given that all necessary information like the CAD models is available in a directly processable form.

Performance reproduction: Recently, [91] have proposed using instructional videos as AR overlays. These videos require including a visible marker which is afterwards removed through image inpainting. Additionally, the instructional videos have to be manually segmented into instructional steps and during run-time, the user manually changes the currently displayed work step by pushing a (virtual) button. Since their approach also needs a marker during run-time it is only applicable in training scenarios, when there is full control over the infrastructure. In contrast to this system, our approach does not rely on markers and allows the automatic authoring from geometrically more complex scenes than the mostly planar workspaces their approach requires. A conceptual discussion about the automation of the authoring process along with example implementations is presented in [125].

2.3 Activity recognition and event segmentation

Our approach is based on temporal event segmentation as a first processing step during authoring and defers exploiting information from scene analysis and hand tracking to subsequent steps. While the connection between event segmentation and instructions is well backed by cognition research [126] it is novel to the domain of Augmented Reality (AR) and computer-aided instructions (CAI). The viability of our approach with respect to this cognitive model has recently been experimentally proven by a follow-up work [127] of this thesis: Using the authoring tools presented within this thesis (Chapter 6), participants of a user study were asked to segment events within video examples of a workflow. These were then compared to analogously user-provided pictorial and textual manuals. The results showed that the event segments comply with borders from the instruction-creation task.

In the following subsections, we will review the literature for the unsupervised and supervised cases.

2.3.1 Unsupervised segmentation

Our approach is based on a statistical examination of manual workflows to identify certain characteristics occurring at times when an action changes. These characteristics are most dominant in motion pattern cues that can be extracted without knowledge about the affordances or goals of these actions. As psychological studies suggest [128, 129], these statistical cues also contribute to human segmentation decisions, even dominantly with infants [128].

Many of the approaches rely on high-level hand, body, and object detection to be able to exploit high-level relationships. Due to the highly error-prone process of object detection in unconstrained, dynamic setups, most of the approaches rely on additional sensors and/or fiducial markers. The authors of [130] use body-worn fiducial markers and inertial measurement units (IMUs) to evaluate different supervised and unsupervised classification methods on movement profiles in a kitchen scenario. In [131] body-worn fiducial markers are used to segment spatial relations using an extension of hidden Markov models (HMM) with special treatment to handle occlusions.

Although cognition research [128, 129] indicates that the exploitation of statistical properties plays a key role in how humans detect and segment actions, there is a relatively small body of research on unsupervised methods. In [132] the authors propose mean-shift clustering and *slow feature analysis* on super-pixels. In contrast to our approach, they require cyclic workflows, *i.e.,* the tasks have to be conducted repeatedly in the same order for extraction. In our approach we rely on motion and compactness cues that are ad-hoc and online evaluable.

Similar holds for [133] where low-level events are mined from multi-sensor data and represented using symbols from a finite alphabet. Frequent sequences of symbols are then scored according to order consistency to recover actions.

Another interesting approach is based on statistical irregularities [134]. Though not directly in the context of action detection, it could be adapted. They extract a patch-based representation from an image sequence. A new image is then reconstructed using the patches in their database. Using this, their method is able to identify untypical or salient image content. [135] discuss three approaches for event modeling based on HMMs, formal grammars, and ontologies with applications in event recognition, and temporal segmentation.

The authors of [136] reformulate activity detection as a binary classification problem by dividing the sequence into hierarchical train and test intervals. By contrast, our approach observes the learning rate of an online trained classifier to determine activity change. This classi-

fier is not determined on a specific internal distance measure and could in fact be realized with various distance or dissimilarity measures, *e.g.,* pixel-wise differences, histogram approaches, or chamfer edge distance.

[110, 111] investigate the usefulness of eye movement towards action recognition and anticipating the next task step. While we also exploit changes in gaze direction as a cue, we further incorporate additional sources of information into the segmentation decision.

2.3.2 Workflow tracking

While camera tracking is typically sufficient for general AR, additional understanding of the dynamic progression is necessary for our aimed level of interactivity in task assistance.

There exist various different approaches to camera tracking, including visual tracking with markers [137], markerless/hybrid [138], with additional inertial sensor fusion [139, 140], SLAM-based [141, 142, 143, 144], and recently, approaches that incorporate RGBD sensors [145, 146, 147]. In this section, we focus on approaches for tracking and internal representation of dynamically changing environments that are real-time capable. For a general overview of the directly adjacent field of action recognition, see [148].

There are three dominant approaches: (1) based on object recognition and the position of interaction objects, (2) based on hand and body trajectories, the position and posture of the hands or pose of tool, and (3) based on general motion or similarity cues. Our approach is clearly situated in the third category of approaches to bootstrap the procedure but then incorporates methods from both other categories to improve results in subsequent optional steps.

The first approach infers user activity and the state of the environment from the time-progressing spatial relationship of recognized objects, *e.g.,* [109] that use fiducial markers, [149] that use multiple RGB and IR cameras to detect changes in size, shape, and position of objects with a shape-from-silhouette approach, or [150, 151] that focus on textureless objects using an RGBD camera in first-person perspective.

The second approach infers the state from observing the hand and body motion trajectories over time [152, 153, 154] and recently, the two approaches often get combined as in [28, 130, 155, 156].

The third approach uses optical flow or general distance measures without prior identification of single objects. For example, the authors of [157] use motion cues within a regular grid to classify tasks in an industrial setting using echo state networks. Using the self-similarity matrix (SSM) to effectively detect repetitive patterns was demonstrated in [158]. The SSM can

also be used as a descriptor for entire workflows [159] as well as to relate two demonstrations of the task, compensating speed fluctuations using dynamic time warping (DTW), which works even under severe viewpoint changes.

However, the usage of tracking aids like fiducial markers to estimate the camera pose and the pose of interaction objects is still the dominant approach [28, 91, 109, 160]. Tracking the user's hands is mostly realized through skin color segmentation [155, 161], but there also exist approaches that, like us, estimate and exploit the full posture for analysis [156]. For body motion as well as for hand motion, the use of body-worn sensor networks is also an often pursued approach. Either without using cameras [152, 153] or in combination with first-person vision [28, 154, 162] or body-mounted cameras [163]. While those approaches are all "inside-out", the classical approach to motion capture involves external cameras. A survey on vision-based motion capture and analysis can be found in [164]. Additionally, the tracking can be augmented with additional sensor readings, *e.g.,* using a screwdriver with force torque sensor, or using sensors built into the interaction objects [49].

Sun *et al.* [156] analyze the image using a combination of 27 DoF finger tracking, 3D object tracking and camera tracking. They use a gaze-directed camera, where they perform hand tracking using a 27 DoF kinematic model of the hand with a skin color and contour based appearance model, rendered through quadrics. In contrast to our main premise they require known objects, as their detection is based on 3D CAD models. Their output are the 3D trajectories of the user's hand, the objects and the gaze direction. While a large focus of our work lies on authoring, from a run-time and tracking perspective alone, this approach is definitely one of the most similar.

[161] address the problem of learning object tracking models from egocentric video examples of everyday activities. Additionally, their approach is built around a profound method for tracking the observed workspace. Through using foreground segmentation and skin color, they automatically identify hand-held objects. As a background model, they generate panoramas from short-term snippets of the video stream. In contrast to our approach, they do not adapt the temporal segment that contributes to each of these panoramas to the ongoing activity, whereas our approach does not incorporate foreground segmentation, at all. Their approach comprises computationally demanding processing steps like super-pixel and graph-cut segmentation, as well as SIFT-descriptor extraction and matching, which makes real-time applications infeasible.

Recently, the authors of [109] have evaluated AR in the psychomotor phase (after Neumann and Majoros [165]) of a workflow, *i.e.,* the phase wherein the user actually executes a 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, their system presents new overlays (dynamic arrows, highlights, or labels) to assist the user during the execution. While we share the distinction between an instructional phase and the psychomotor phase, we focus on the technical realization of markerless spatiotemporal tracking. Additionally, an important goal of our work is to automate the creation process of systems for procedural assistance. [166] propose an event-based distance measure between several sequences that allows matching events from even a single example. The authors of [155] first segment the user's hand, face, and the manipulated object within the video stream. The work then focuses on the simultaneous recognition of the user action and the manipulated object using conditional random fields. In contrast to our approach, the system requires a frontal view on the scene, facing the user. [167] uses an HMM on a spatial 3D occupancy grid of the user's right hand derived from 2D skin color blob tracking on two cameras. In contrast to our method, their approach relies on a second camera for reconstructing the 3D positions, while we base the recognition upon 2D descriptors with prior camera movement compensation. In fact, the only way we incorporate full 3D information is in the way of full finger tracking, as we show in Chapter 5. [168] propose a similar approach with a single, wearable camera, where they use 2D positions of objects and the user's hands detected using color histogram. In contrast to our approach, they process the positions solely in the camera coordinate system, while we perform the classification using what we call the *relevance plane*, a 2D model of the current work step.

2.4 Descriptors and models

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. We will discuss current approaches in the following subsection and afterwards review approaches to region descriptors, usable to recognize wide-spread portions of an image.

2.4.1 Anchoring of augmentations and overlays

Anchoring is challenging as the extracted information has both a spatial and a temporal association with the environment that is susceptible to change drastically in the course of the workflow.

One straight-forward 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 the marker based approaches [169], 3D model-based approaches [170], or SLAM-related approaches [141].

We can exclude the entire body of literature dealing with markers or CAD-model tracking as we explicitly forbid any prior knowledge or control over the infrastructure in our approach.

The remaining SLAM-related approaches suffer from two problems: (1) The camera motion of the head-worn camera dominantly consists of orientation change and we cannot assume sufficient camera translation to reliably reconstruct geometry to bootstrap the mapping. (2) The environment is susceptible to change drastically and perpetually due to user interaction.

Very recently Tan *et al.* [142] proposed a promising method that explicitly handles dynamically changing environments and thus alleviates the second problem. A possible drawback towards our approach on camera tracking could be its dependence on point features, which might exclude sparsely textured workspaces.

These issues can be solved through the use of a combined RGB and depth sensor, which has recently become popular. [145, 146] propose methods for 6-DoF camera relocalization and tracking using RGBD cameras by combining the 2D image and the according depth map. [145] present a system for tracking and dense mapping by fitting the current depth frame within the global map using the iterative closest point (ICP) algorithm. The authors of [146] use regression over a set of synthesized views to relocate the camera under partial occlusion and sparse texture.

In this work, we restrict our focus to monocular RGB cameras. The arguments from a practical standpoint are the higher potential for miniaturization and the lower power consumption compared to active (light-emitting) depth sensors and the mere fact that as a consequence all currently available, suitable HMDs are exclusively equipped with RGB cameras.

To circumvent the challenges of 3D acquisition, a popular method is to associate information with 2D image features. In this case, information is anchored with point-features [171, 172], region descriptors [9] or object detectors [173, 174] 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 through a planar (not necessarily connected) structure within the environment that has a large overlap

with the region of interest. In contrast to methods like [175, 176], 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.

There are approaches to mitigate the effect of rotation-only camera motion by switching from depth reconstruction to a homographic model to track features. The authors of [177] have recently proposed a scheme that deals with the motion requirements through explicit model switching. This approach has also been adopted and extended in further SLAM approaches [144]. The disadvantage is that the depth of the feature points cannot be reconstructed from rotation-only observation. Hence, this method leads to rather decoupled models for the two types of features. In contrast, our approach is based on prior temporal segmentation that leads to locally optimal, decoupled 2D maps that are connected through a consistent model.

2.4.2 Region descriptors

Template matching proves to be particularly useful in case of sparsely textured objects that make it difficult to locate stable interest points. This prohibits the use of point descriptors and also the application of many affine region detectors. When applied to the entire image, these approaches can function as a robust dissimilarity measure. As region descriptors are explicitly designed to efficiently cope with certain image distortions, they generally outperform correlation-based dissimilarity measures. For our implementation and the experiments we propose an image dissimilarity function based on template matching, more precisely the *dominant orientation templates* (DOT) [9]. For each grid cell the template stores a list of eight booleans, seven to denote the presence of respective quantized gradient orientations, one to denote the absence of any strong gradients. The method is closely related to histograms of oriented gradients (HOG) [178]. In fact, DOT is essentially a binarized HOG using a locally adaptive thresholding value. Though, due to the binarization the method performs very fast as matching solely relies on bitwise operations that can be effectively handled by the vector units of modern processors. The matching distance evaluates the number of grid cells, where the dominant input image gradient was found in the list for that grid cell.

The authors of [9] report state of the art results for their method while being able to search the entire image in real-time without relying on any feature point detector. This is possible due to a binary representation of these lists, branch-and-bound clustering [179], and an explicit

invariance to small translational shifts. This allows skipping large parts of the image at the cost of reduced translational accuracy.

Another benefit of most template matching approaches is the possibility to add templates on the fly to allow online tracking [9, 178, 180]. This possibility is mainly depending on the storage strategy. In this work we are using branch-and-bound clustering [179], as proposed and adapted by [9] for usage with DOT. Another interesting approach is the use of locality sensitive hash functions [181, 182]. These hash functions are designed to more likely produce hash collisions also on merely similar entries.

A popular metric for matching edge based templates is the chamfer distance (*e.g.,* [183, 184]) which is efficient to compute using the distance transform. [185] present a template representation directly based on the distance transform which is invariant to scale changes and generalizes quite well. Unfortunately, their approach is only applicable to objects with roughly closed contours which narrows its field of usage. One of the major drawbacks that all methods based on distance transform have in common is the dependency on edge extraction, which still is an error prone step. [186] circumvent this by proposing an efficient to calculate approximation of the directed chamfer distance and a probabilistic line matching scheme to identify model edges with high probability.

In [187] the authors present a fast silhouette template matching approach. It is based on learning a list of axis-aligned rectangles covering the silhouette area that is efficiently matched using the integral image. Due to that and a hierarchical indexing of their templates, they achieve real-time performance independent of the image resolution. The downside is that object silhouette extraction is similarly fragile as edge extraction and does not preserve any information other than the object boundary which vastly reduces the applicability to arbitrary objects. [188] use the idea of bitwise comparison on binarized histogrammed intensity patches (HIP). Their approach has very efficient run-time behavior but requires an extensive training phase to counter the effects of a fast but inaccurate interest point localization. Eventually, with the recent popularity of depth cameras there now also exist template approaches that exploit both the color and the depth image, *e.g.,* [180].

2.5 Finger tracking

In this section we present related work for our hand and finger tracking approach. Our main idea is to incrementally update an underlying database with entries for successfully tracked

frames. The database is hereby initialized with a very large amount of synthetic entries. The straight forward approach of just adding an entry for each successfully tracked posture would adapt utterly slowly. Thus, in practice this approach would almost never have an impact on the actual tracking performance. We circumvent this through filling the parameter space between successful matches using image-based rendering. As soon as a couple of frames have successfully been matched, we synthesize the database entries in a perimeter of the adapted entries to increase the adapting rate.

We therefore divide related work into a section dealing with approaches to database query and general discriminative methods and a section for the appearance methods used to generalize the database.

2.5.1 Tracking by detection

There are numerous approaches with a tracking-by-detection architecture for articulated bodies based on database indexing. Examples for such work are dominantly based on edge templates [186] and silhouettes [187]. The essential approach is to establish a large database of object descriptors labeled with the corresponding generating parameters. At run-time, the generating parameters for the current frame are recovered through querying the database for the closest match with the current observation.

The accuracy of the approach is determined through the sampling density in the database and thus suffers strongly from the high dimensionality of the underlying search problem. This so-called ”curse of dimensionality” leads to an exponential performance decrease with higher degrees of freedom (DoF). To allow fast nearest neighbor searches in spite of this problem, the literature proposes similarity and parameter sensitive hashing functions [181, 182]. Although these methods allow faster general nearest neighbor searches, our proposed method has the benefit of allowing fast online database changes and a more flexible shaping of the search results incorporating the current tracking state.

Another approach is described in [187] where the authors build a database of hand silhouettes in different poses using hierarchically ordered axis-aligned rectangles and integral images for fast matching. The resulting approach is fast enough to run in real-time at least on smaller sets but silhouettes are hardly descriptive enough to recognize and distinguish complex postures.

In principal, our approach is a combination of beam-search and branch-and-bound clustering [179] in an appearance descriptor database. Since the choice of the descriptor is crucial to

our method, due to its computational footprint and robustness with respect to the sparsely textured hands, we briefly discuss the necessary properties. As the human hand does not provide sufficiently stable point features among all poses, point descriptors like [172] or [189] are not applicable. The authors of [190] use characteristic patches like the finger nail area, though, but these are only valid for small local regions of the parameter space.

Region descriptors for the entire hand patch are more suited since they describe the object as a whole. This is particularly beneficial when describing complex postures that are otherwise hard to capture in a regression/classification model. There is a huge variety of template descriptors, ranging from thumbnail views [191], over gradients [9], and silhouettes [187], to using the distance transform [185]. Silhouette templates are an often pursued approach, *e.g.,* [187] but have the disadvantage that they lose descriptiveness on complex postures. Edge templates ([186], [192]) on the other hand suffer from a particularly difficult and thus error-prone edge extraction step on hand images, especially when lighting is rather ambient. Thus, these templates are most stable on contour edges which share the same disadvantageous properties as silhouettes. The method proposed by [191] works with a distinctively colored glove that the user needs to wear. This tracking aid leads to less ambiguous images produced by different poses. The approach works with a large database of sampled views with a subsequent refinement of the closest match.

Classifiers such as [193] are likely to give excellent results at vastly reduced memory consumption compared to template techniques. However, the advantage of the template approach is that adapting to the tracking target can be performed online at very low computational costs. To us, this is a particularly desired property of the tracking system. Dominant orientation templates (DOT) [9] are computationally efficient and work very well with smoothly shaded objects such as hands. Since the descriptor encodes the image content quite locally, this facilitates clustering particularly for articulated structures like hands, where only parts of the image are affected from change of single parameters. Furthermore, using the same descriptor for workflow and hand tracking allows to streamline the tracking procedure, as we will show in Section 5.2.2.3.

2.5.2 Appearance models

Fast and accurate rendering of objects plays an important role in vision-based analysis-by-synthesis frameworks, particularly in the context of hand tracking with generative models. Therefore, a variety of methods has been proposed for accurately predicting the appearance of

a human hand in certain poses. However, most objective functions used in derived tracking methods are either solely or dominantly comparing hand silhouette and edge features between the current hypothesis and the observation, *e.g.,* [186, 187, 194, 195]. Especially in a monocular setting, this leads to a severely ill-posed problem as many postures create the same or very similar silhouette and edge information. A general overview of hand tracking techniques can be taken from [196].

The authors of influential work in [197] propose to use truncated quadrics to approximate the shape of the human hand. As quadrics can be efficiently projected to the image plane, this provides a computationally lightweight way to formulate an objective function based on silhouette and edge proximity. The authors of [194] propose using particle swarm optimization [198] on a similar objective function. While this works reasonably well on high quality frames they suffer from the fragility of skin color and edge extraction in presence of severe lighting or blur as we show in our experiments.

In the work of [195] the authors propose a ”cardboard model”. They use model-aligned rectangular shapes to approximate the hand's contour edges. To prevent collapsing of their rectangular segments they strictly limit the possible viewpoints of their model - a restriction that does not apply with our method.

In fact, only a few approaches incorporate additional, more complex image cues. In the work of [190] the authors propose the incorporation of salient points, *i.e.,* finger nails, into the objective function. Although this greatly alleviates the minimization problem, the requirement on the frame quality is prohibitive in many practical cases.

The authors of [199] have proposed the use of texture and shading information to alleviate the observation ambiguities using a textured, deformable 3D mesh model. Their method minimizes a pixel-wise distance between the current observation and a deformable mesh model with the texture taken from the last frame. The use of a deformable skinning model is computationally expensive and therefore achieving real-time performance is difficult. Also the shading information is captured from one frame alone and then approximated for the subsequent frame using Gouraud-shading. However, there is no straight forward way to extend this approach to greater pose difference. Additionally, the model requires well defined object boundaries. Therefore, it requires a very precise fit and non-blurred image material to produce decent results. These prerequisites are mostly prohibitive in realistic applications and do not occur in our approach.

The authors of [200, 201, 202] are incorporating depth information from a Kinect camera. They gain much increased stability from the additional information. However, the more complex hardware required for depth sensing (stereo setup or 'active' sensors like time-of-flight or Kinect) is complicating a use on highly mobile hardware.

Image-based rendering (IBR) provides a principled way to synthesizing realistic object appearance from unobserved viewing angles. Since the human hand is kinematically complex with 22-30 degrees of freedom with prevalent self-occlusion we resort to a strong geometric proxy to gain robustness. A related approach is the work of [203] that generates new views of football players from wide-baseline stadium cameras. The players are modeled as articulated bodies with billboard fans aligned to each bone. Thus, the authors call their approach "articulated billboards". Instead of performing a pixel-wise morph as proposed in our work, they create a separate billboard per segment and camera, arranged in a fan around the kinematic bone. Thus, they have less control over elastic deformation. However, just as in the cardboard model [195] their billboards are not stable at all relative viewpoints (which is not a requirement in their application).

The idea of morphing between prototype views images was pursued by [204] and [205]. Their approach is based on establishing point correspondences which is not easily possible in our application due to the high articulatory complexity of the hand and its lack of texture. Also since we are decomposing the necessary warp into a segment-wise rigid (predicted) transformation and an axis-aligned efficient pixel-wise warp we expect our method to be faster by orders of magnitude. Active appearance models (AAM) [205] describe the relationship between the movement of control-points and the change of pixel intensities through primary components analysis (PCA) to recover the pose parameters of unseen object views (faces in their case). The control-points hereby warp the image content. The main difference is that we use an anatomical kinematic model to predict coarse-grained deformation parameters and contour cues for fine-grained and elastic deformation. In contrast, AAM directly learns a linear model for both. Since using a strong geometric proxy provides an improved prediction of inter-pose change, our method is applicable to kinematically complex objects. Nevertheless, it would be interesting to investigate the feasibility of AAM on a per-segment level.

3

Unsupervised task segmentation

The segmentation of actions within a video sequence has a large number of applications comprising action and activity recognition, scoping of training data for classifier training, and (visual) data mining [206]. Most importantly, the temporal segmentation of actions is an essential building block towards discovering the structure of a workflow. In the following sections we will explain our approach to unsupervised segmentation of workflow videos based on image distances. We will begin with a review of related work. After that follows a short discussion of the problem and an introduction to the image dissimilarity functions that we use for clustering. We then show how we extend these functions to receive a frame-to-frame measure in time series data that we use to provide compactness criteria for segmenting a workflow. These measures can also be used to provide a coarse classification of each segment. We additionally interpret a change in the user's focus of attention as a cue for segment changes, which are estimated through changes in gaze direction, *i.e.,* through camera pose tracking.

The dissimilarity measures and the camera tracking approach described within this chapter are also being used during the run-time workflow tracking and the temporal alignment of workflows, both described in Chapter 4.

3.1 Image distance functions for task segmentation

The majority of approaches towards action recognition in video sequences is based on the analysis of high-level features, derived from detecting and tracking certain image parts such as tools and interaction objects [61, 62]. While the unsupervised isolation of these objects is already difficult and therefore a rather fragile preprocessing step, it foremost depends on the

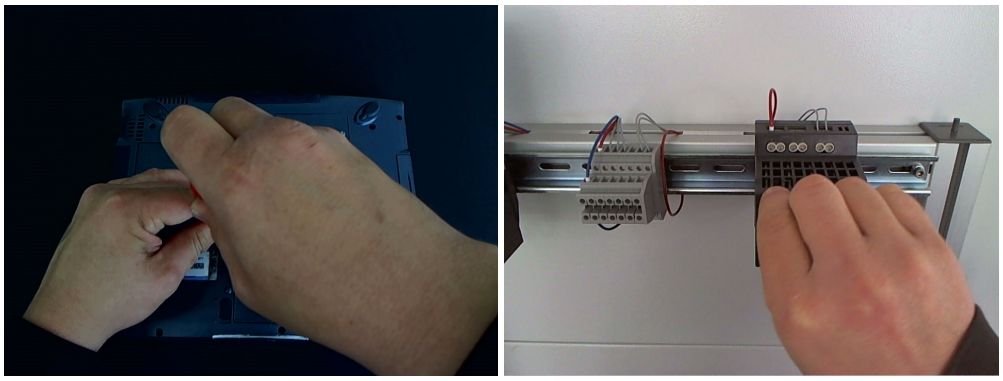


Figure 3.1: Examples of the user's hand occluding the used tool (left) or the interaction object (right).

actual observability of the objects. Especially in video from first-person view, tools and objects might not appear at all in the entire recording. Figure 3.1 shows examples of this problem.

Instead we propose using image distance measures of the kind $d : \mathbb{R}^2 \times \mathbb{R}^2 \to \mathbb{R}$ between two images $\boldsymbol{I}, \boldsymbol{J}$, based on low-level image comparison. The direct benefit is that a low-level comparison is not prone to failure due to a sophisticated preprocessing step. What makes it complicated to apply whole-image distance is that the return value of an image distance function does not necessarily reflect a semantic relation between two images. For example one could find three entirely different images, let us say: one image of a house and two images of cars that roughly exhibit the same (large) pair-wise image distance although the two images of cars are far more related. The reason is of course that the distance function does not directly evaluate the similarity of the latent concepts but rather a roughly correlated observation. If the images differ too much, the measure does not reflect the latent dissimilarity anymore. Figure 3.2 shows an example of this condition. We claim that the negation is indeed valid and postulate the

Surrogate assumption: Within its specific confidence radius, an arbitrary distance function on correlated phenomena can be used to infer a latent similarity.

In the application we pursue, we want to relate images of manual actions to infer whether they could originate from the same task. If the images are almost identical, the probability is low that both images could not be produced by the same task. Figure 3.3 illustrates this thought.

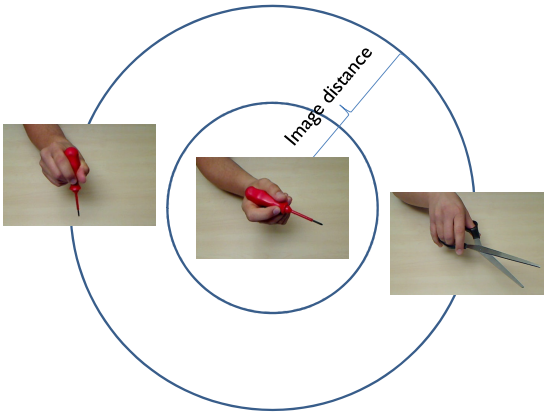


Figure 3.2: Image distance vs. latent similarity: The center image has almost the exact same distance towards the left and right image using our comparison function, although the left image is semantically more similar.

Given an image distance function $d(\boldsymbol{I},\boldsymbol{J})$ between two images $\boldsymbol{I},\boldsymbol{J}$, we assume that the images are latently similar if and only if $d(\boldsymbol{I},\boldsymbol{J}) < T$, where T is a threshold that we call the *confidence radius* that is specific to the given distance function. If $d(\boldsymbol{I},\boldsymbol{J}) \geq T$, then we cannot decide. Naturally, the surrogate assumption is not transitive, *i.e.,* we cannot propagate the local similarity from frame to frame in a video sequence to infer whether the first and the last frame are related.

In the following subsections we present our framework to partition an image sequence into conceptually similar images by exploiting the surrogate assumption. This formulation is independent from the actual choice of distance function. However, for didactic reasons, we first introduce the distance function that we use for the experiments.

3.1.1 Robust dissimilarity functions

In the context of our application an adequate image comparison function should predominantly be sensitive to task related image change. Since image brightness changes or minor camera movements are generally no indicators for task differences, the function should ideally be invariant to these types of changes. It is therefore easy to see that the desired function is not a metric, as it intentionally violates the condition $d(\boldsymbol{I},\boldsymbol{J}) = 0$ if and only if $\boldsymbol{I} = \boldsymbol{J}$. Hence, we will

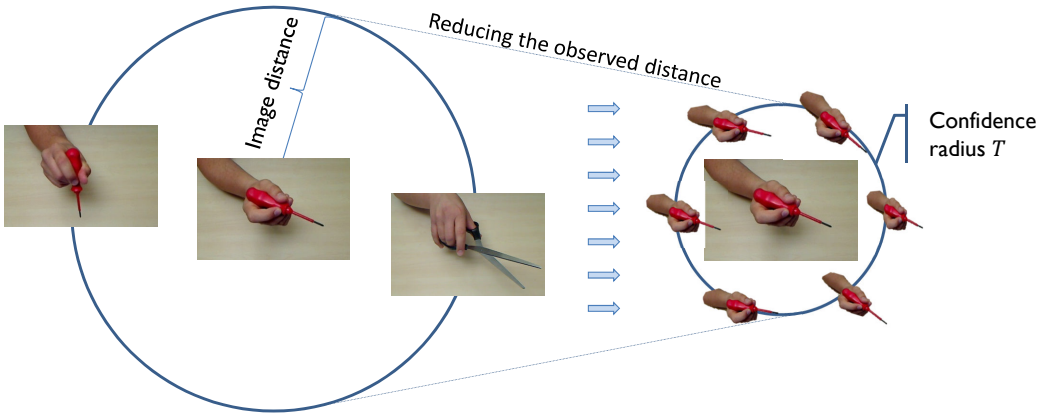


Figure 3.3: Illustration of the confidence radius: By reducing the observed distance to a very close perimeter, we can safely assume that the images within this radius were produced by the same action.

use the more general term *dissimilarity* instead of distance from here on. In fact, the dissimilarity measure that we propose is a so-called hemimetric, as it additionally violates the symmetry property.

Some comparison functions are implicitly invariant or robust towards certain image distortions, *e.g.,* normalized cross correlation is robust to lighting changes. However, every comparison function can be made explicitly invariant towards (predictable) distortions at the cost of increased computation time. This can be achieved by sampling the effects of a certain distortion and extending the function to return the dissimilarity to the minimum sample.

Given an image distortion function $\mathcal{T}(\boldsymbol{I}) \rightarrow \{\text{sample}_1, \text{sample}_2, \ldots\}$ that returns a set of transformed images, we return the minimum dissimilarity between $\boldsymbol{I}$ and the transformed images:

$$d_{\mathcal{T}}(\boldsymbol{I},\boldsymbol{J}) = \min_{\mathbf{K} \in \mathcal{T}(\boldsymbol{I})} d(\text{crop}(\mathbf{K}), \text{crop}(\boldsymbol{J})) \,, \tag{3.1}$$

where $\text{crop}(\boldsymbol{I})$ is a center-cropped version of the image to be able to compensate small amounts of camera translation.

We define $\mathcal{T}(\boldsymbol{I})$ to produce synthetic camera views within a small cone around the optical axis. Since we do not possess information about the scene geometry (*e.g.,* from a depth sensor), we only produce affine, more precisely scaled, rotated, and translated instances. This can be interpreted as an approximation using an orthogonal camera. In this work our dissimilarity

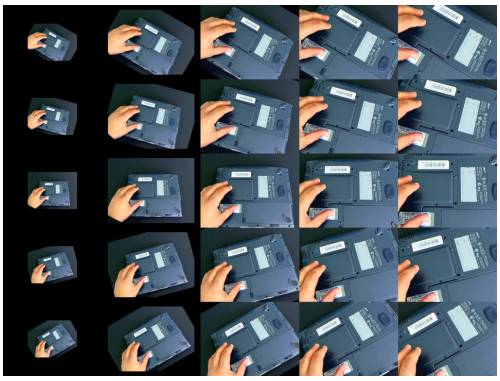


Figure 3.4: Scaled and rotated samples in $\mathcal{T}(\boldsymbol{I})$.

function is based on the region descriptor DOT (dominant orientation templates) [9], since it is invariant to small deformation and lighting change and can handle translation very efficiently. Since DOT already is handling translation, we define $\mathcal{T}$ to only cover scale and rotation change, see Figure 3.4 for an illustration. Due to the deformation invariance, this definition of $\mathcal{T}$ even suffices for small to medium changes of the elevation angle, depending on the range of depth within the camera image. Since DOT returns a score s_{DOT} between 0 (entirely unsimilar) and 1 (very similar) we define $d = 1/s_{DOT}$. Experimentally, we found $T = 1.11$ to be a good confidence threshold for the DOT-based approach.

3.1.2 Extension to time series data

Let $\boldsymbol{\mathcal{S}}$ be a given image sequence of size $|\boldsymbol{\mathcal{S}}|$ and $\boldsymbol{\mathcal{S}}_i$ denote the i-th frame from that sequence. From our given dissimilarity measure we aim to derive a time-series extension that is additionally using intermediate frames to infer whether a previous frame t_0 and the current frame t are dissimilar. We will use the notation $D_{t_0}(t)$ for this dissimilarity measure between $\boldsymbol{\mathcal{S}}_{t_0}$ and $\boldsymbol{\mathcal{S}}_t$ evaluating all $\boldsymbol{\mathcal{S}}_i$ with $t_0 \leq i \leq t$.

There are two intuitive choices for such a measure $D_{t_0}(t)$ that we will shortly discuss in advance. The first choice is to simply sum up all frame-to-frame dissimilarities between t_0 and t:

$$D^{sum}_{t_0}(t) = \sum_{i=t_0}^{t-1} d(\boldsymbol{\mathcal{S}}_i, \boldsymbol{\mathcal{S}}_{i+1}). \tag{3.2}$$

This would only be a valid dissimilarity measure if the sequence was entirely progressive, *i.e.,* there would be no repetition at all. In all other cases, the return value of this function would

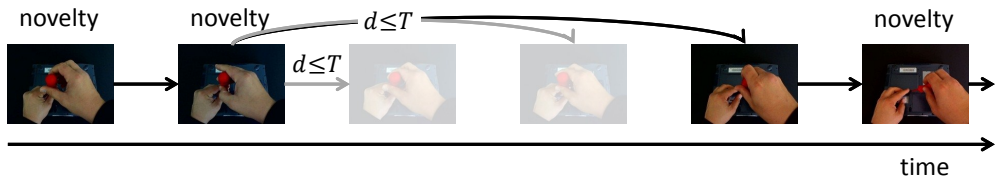


Figure 3.5: Illustration of the shortest path distance $D^{sp}_{t_0}$, compare Equation 3.4. Grayed out images indicate skipped frames.

be too large. One trivial example to emphasize this is to examine the case where $\boldsymbol{\mathcal{S}}_{t_0} = \boldsymbol{\mathcal{S}}_t$. One would generally expect the dissimilarity to be 0 if and only if the images are equal up to viewpoint change, but in presence of differing intermediate frames this is not the case.

The second intuitive choice would be to use the minimum dissimilarity of $\boldsymbol{\mathcal{S}}_t$ to $\boldsymbol{\mathcal{S}}_{t_0}$ and all intermediate frames.

$$D^{min}_{t_0}(t) = \min_{\forall t_0 \leq i < t} d(\boldsymbol{\mathcal{S}}_i, \boldsymbol{\mathcal{S}}_t). \tag{3.3}$$

Although Equation 3.3 would be a valid choice for a dissimilarity measure between an image and a set of images it is not a suitable measure between two frames. In fact, it is even likely to be entirely independent of frame $\boldsymbol{\mathcal{S}}_{t_0}$ as long as there is only one intermediate frame that is closer to $\boldsymbol{\mathcal{S}}_t$.

Instead, we propose a measure that we call *shortest path*. It not only solves the aforementioned problems but also 'protocols' whenever the dissimilarity function is evaluated above its confidence threshold. The term *shortest* path refers to the path through the image sequence from frame t_0 to t with the least amount of frames not within any confidence radius. Figure 3.5 shows an illustrating example. It is given by the following recursive definition:

$$D^{sp}_{t_0}(t) = \begin{cases} 0 & \text{if } t = t_0 \\ D^{sp}_{t_0}(m) + d(\boldsymbol{\mathcal{S}}_m, \boldsymbol{\mathcal{S}}_t) & \text{if } \exists m \in \boldsymbol{\mathcal{N}}_{t_0}(t-1): \\ & \min d(\boldsymbol{\mathcal{S}}_m, \boldsymbol{\mathcal{S}}_t) < T \\ D^{sp}_{t_0}(t-1) + d(\boldsymbol{\mathcal{S}}_{t-1}, \boldsymbol{\mathcal{S}}_t) & \text{else,} \end{cases} \tag{3.4}$$

where $\boldsymbol{\mathcal{N}}_{t_0}(t)$ is the set of frame-indices between t_0 and t with a pair-wise frame dissimilarity $d(\boldsymbol{\mathcal{S}}_i, \boldsymbol{\mathcal{S}}_j) \geq T$, for all $t_0 \leq i < j \leq t$. We call $\boldsymbol{\mathcal{N}}_{t_0}(t)$ the *novelty set* and approximate it with straight-forward greedy clustering, *i.e.,* as soon as the current frame has a pair-wise dissimilarity greater than T to all other novelty frames, we add its frame-index to this set.

Informally, $D^{sp}_{t_0}(t)$ can be interpreted like this: For the current frame $\boldsymbol{\mathcal{S}}_t$, look at all previous frames back to $\boldsymbol{\mathcal{S}}_{t_0}$ for frames that are so similar ($d < T$) that we can assume they come from

the same activity. If we have found such a frame $\boldsymbol{\mathcal{S}}_m$ with minimal dissimilarity we add the dissimilarity between $\boldsymbol{\mathcal{S}}_m$ and $\boldsymbol{\mathcal{S}}_t$ to our path and start over at frame-index m, directly (thus ignore all frames between m and t, the second case of Equation 3.4). If there was no such frame, we add the (typically small) dissimilarity between t and the previous frame $t-1$ and start over at the previous frame.

One should note that whenever no such frame was found (the third case in Equation 3.4) the current frame-index is added to the set of novelties. Thereby, we protocol that we encountered a frame for which we cannot certainly infer the task affiliation in the sense of the surrogate assumption.

When we compare a new frame to previous ones, we are in fact only incorporating the novelty frames $\boldsymbol{\mathcal{N}}_{t_0}(t)$. That is for two reasons: firstly, to speed-up computation, secondly and more importantly, the closest frame for $\boldsymbol{\mathcal{S}}_t$ would most probably always be the directly previous one. By restraining to $\boldsymbol{\mathcal{N}}_{t_0}(t)$ we implicitly disallow associating with frames between the last novelty and t.

Interestingly, the measure has closed-form upper and lower bounds that are reached in presence of characteristic content: In case of an entirely progressive (*i.e.,* non-repetitive) sequence $D^{sp}_{t_0} = D^{sum}_{t_0}$ and in case of an entirely static sequence $D^{sp}_{t_0} = D^{min}_{t_0}$. Since $D^{min}_{t_0} \leq D^{sp}_{t_0} \leq D^{sum}_{t_0}$ holds, we can use that for normalizing $D^{sp}_{t_0}$:

$$D^{|sp|_1}_{t_0} = \begin{cases} 0 & \text{if } D^{sum}_{t_0}(t) = 0 \\ \frac{D^{sp}_{t_0}(t) - D^{min}_{t_0}(t)}{D^{sum}_{t_0}(t) - D^{min}_{t_0}(t)} & \text{else.} \end{cases} \tag{3.5}$$

$D^{|sp|_1}_{t_0}(t)$ is a robust normalized measure for how progressive (*i.e.,* non-repetitive) the sequence is between t_0 and t. Values closer to 1 indicate a progressive activity, lower values a repetitive or cyclic activity, with a value of 0 in case of an entirely static segment. $D^{min}_{t_0}, D^{sp}_{t_0}$, and $D^{sum}_{t_0}$ for an entire segmented sequence are shown in Figure 3.6. The change rate of the novelty count, *i.e.,* the change rate of the number of elements in $\boldsymbol{\mathcal{N}}_{t_0}(t)$ constitutes an uncertainty measure for the assumption of task identity.

In the following we explain how these measures can be interpreted to derive online evaluable decision criteria for task segmentation.

3.1.3 Determining segment boundaries

Through examination of manual workflows, one can observe certain characteristics occurring at times when an action changes. These characteristics are motion pattern cues that can be

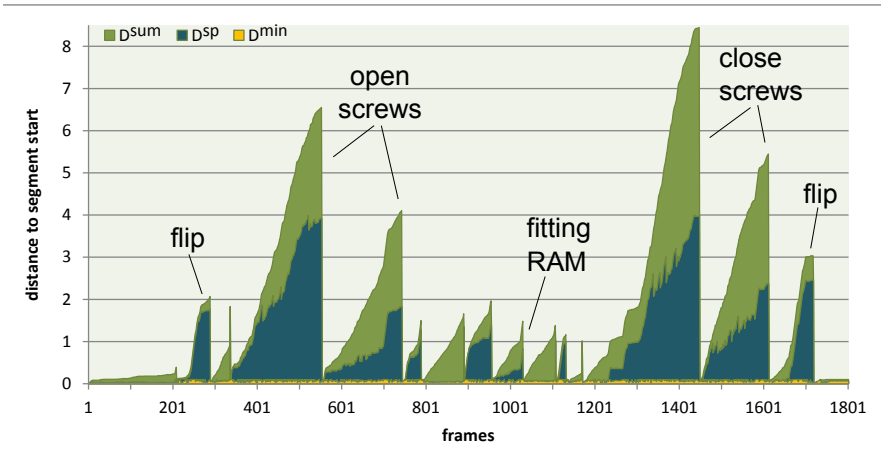


Figure 3.6: Plot of minimum, summed, and shortest path concatenated for all segments of the 'Install RAM' sequence. Repetitive actions like releasing/fastening screws and ensuring the fit of the RAM show a significantly smaller shortest path with respect to summed dissimilarity compared to progressive tasks like flipping the device.

extracted without knowledge about the affordances or goals of these actions. As psychological studies suggest [128, 129], these statistical cues contribute to human segmentation decisions, even dominantly with infants [128].

We have extracted three conditions that we use to determine segment boundaries:

- After a period of motion the activity comes to a significant slow-down or halt.
- After a period of motion a change in the motion pattern occurs.
- Before and after periods that are entirely static, *i.e.,* no image change except for noise.

Please note that we will complement these three conditions that are based on motion cues with a fourth that is based on changes in the focus of attention in the following section.

The latter two conditions are rather intuitive, the first requires a bit of illustration. This condition applies to cases like *e.g.,* placing an object or positioning a screwdriver, where aiming requires a slow-down of the motion. As the third condition is basically the absence of an activity, it reduces the search space for actions. Especially since these segments often do not contain the user's hands, the according frames are highly reliable for tracking as well as for the optical validation through comparing the appearance before and after an action (compare the according approach described in Section 6.3.4).

For detecting these conditions, we use the set of measures from the previous section. The normalized shortest path $D_{t_0}^{|sp|_1}$ is sensitive to the type of underlying image change: Since in the progressive case almost every frame gets flagged as a novelty, $D_{t_0}^{|sp|_1}$ remains close to 1. However, in case of repetitive or static content, the measure monotonically approaches 0. So, a sudden increase is a strong indicator for some kind of pattern change, as this means that a high number of frames cannot be associated to the previous observation. This allows to separate adjacent repetitive actions but is not applicable to adjacent progressive actions. Since a progressive action often coincides with a slow-down near the end, as formulated in the first condition, the framework will separate these cases. To also treat the cases where the condition does not hold, we additionally exploit a change in the focus of attention, as will be explained in Section 3.2. For segmentation we are most interested in the change rates of the measures. Since these are time series with a relatively high amount of local fluctuation, we determine the change rates of $\mathcal{N}_{t_0}(t)$ and $D_{t_0}^{|sp|_1}(t)$ using a sliding window of length w:

$$\Delta\mathcal{N}_{t_0}(t) = \frac{1}{w}\left(|\mathcal{N}_{t_0}(t)| - |\mathcal{N}_{t_0}(t-w)|\right) \tag{3.6}$$

$$\Delta D_{t_0}^{|sp|_1}(t) = \frac{1}{w}\left(\left|D_{t_0}^{|sp|_1}(t)\right| - \left|D_{t_0}^{|sp|_1}(t-w)\right|\right). \tag{3.7}$$

We choose w in all our experiments to be corresponding to a duration of one second. $D_{t_0}^{|sp|_1}$ is up to fluctuation monotonically decreasing roughly proportional to $1/(1+t-t_0)$ in case of static, repetitive (not necessarily cyclic) periods.

If we then encounter a cumulation of non-assignable images, this will result in an increase of $D_{t_0}^{|sp|_1}$, which we simply formalize as

$$\Delta D_{t_0}^{|sp|_1}(t) > 0, \tag{3.8}$$

as local fluctuations are already smoothed out through the sliding window. Through examination of $\Delta\mathcal{N}_{t_0}$ we gain information about the progressive motion. A significant decrease, defined by a ratio α of the recent peak rate $\Delta\mathcal{N}_{t_0}$, is used to identify a slow-down:

$$\max_{f=t_0}^{t} \Delta\mathcal{N}_{t_0}(f) > M \tag{3.9}$$

$$\Delta\mathcal{N}_{t_0}(t) < \alpha \max_{f=t_0}^{t} \Delta\mathcal{N}_{t_0}(f). \tag{3.10}$$

The threshold M determines the minimum motion that an action needs to exhibit before being a candidate for segmentation. Equation 3.9 formulates a heuristic to prohibit the segmentation

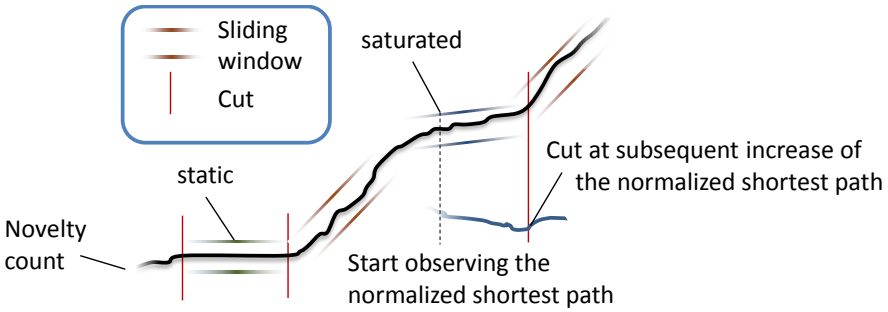


Figure 3.7: Illustration of our segmentation criteria: As soon as the observation gets saturated, indicated through a slower change rate of the novelty count $\Delta\mathcal{N}_{t_0}$, we cut at the next increase of the normalized shortest path $D_{t_0}^{|sp|_1}$.

of periods with potentially insufficient motion as distinct actions. Often, it is not required to threshold the minimum motion (*i.e.,* $M = 0$) and we analyze the influence of this threshold in our evaluation chapter.

Equation 3.10 is not solely an indicator for a slow-down or halt of motion. In fact, there are far more precise ways (*e.g.,* based on optical flow) to assess the motion velocity. We recall our main assumption that we can only infer a latent similarity within the confidence radius of a surrogate function. When the rate of novelties decreases, this means that we can actually relate the current images with our previous observation. The knowledge about the set of appearances produced by the current action has reached a state of saturation. Therefore, an increase of Equation 3.8 is caused by an actual change of the current appearance pattern. Figure 3.7 illustrates the process.

A premature increase of $\Delta D_{t_0}^{|sp|_1}$ in Equation 3.8, before condition 3.10 is met could either mean that there really is a new action or the current action was just not sufficiently observed. Experimentally we found $\alpha = 0.5$ is giving good overall values.

Switches from static to non-static content naturally coincide with an increase of $\Delta D_{t_0}^{|sp|_1}(t)$, compare Equation 3.8. However, since we await the saturation of the current segment and due to the way we compute temporal derivatives, we gain robustness with a dedicated handling. We segment static periods by extracting the longest period that satisfies

$$D_{t_0}^{min}(t) = d(\boldsymbol{\mathcal{S}}_i, \boldsymbol{\mathcal{S}}_{t_0}) < T, \forall i \in [t_0+1, t] \text{ and } t - t_0 \geq l_{min}. \tag{3.11}$$

So, if at least l_{min} consecutive frames are all similar to the first in the series, we extract this until the first frame that violates the criterion. Algorithm 1 lists our entire segmentation algorithm, including the dedicated handling of static segments.

Algorithm 1 Segmentation of a sequence.

Set current frame: $t \leftarrow 0$
while $t \leq n$ **do**
  Set start frame of current task segment: $t_0 \leftarrow t$
  Set start frame of potential static segment: $s_0 \leftarrow t$
  while $t \leq n$ **do**
    Go to next frame: $t \leftarrow t+1$
    // Check conditions for a static segment
    if $D_{t_0}^{min}(t) < T$ **and** $\mathcal{N}_{s_0}(s_0) = \mathcal{N}_{s_0}(t)$ **then**
      if $t - s_0 \geq l_{min}$ **then**
        $t \leftarrow$ position of next $D_{t_0}^{min}(t) \neq d(\mathcal{S}_{t-1}, \mathcal{S}_t)$
        Output static segment from s_0 to $t-1$
        Leave inner **while**-loop
      end if
    else
      Reset start of potential static segment $s_0 \leftarrow t$
    end if
    // Check condition for motion and pattern change
    if $t - t_0 \geq w$ // due to sliding window
    and $\max_{f=t_0}^{t} \Delta\mathcal{N}_{t_0}(f) \geq M$
    and $\Delta\mathcal{N}_{t_0}(t) < \alpha \max_{f=t_0}^{t} \Delta\mathcal{N}_{t_0}(f)$ **then**
      $t \leftarrow$ position of next $\Delta D_{t_0}^{|sp|_1}(t) > 0$
      Output segment from t_0 to $t-1$
      Leave inner **while**-loop
    end if
  end while
end while

3.2 Head gaze direction and attention

A change in the focus of attention is another important cue for action change. In this section, we will present how we estimate changes in the focus of attention from the head gaze direction, estimated from the pose of the head-worn camera.

3.2.1 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 [207], we use RANSAC to find the largest subset of correspondences that support 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

$$\|\vec{p}_{t+1} - \mathbf{H}_t^{t+1}\vec{p}_t\|_2 \leq \varepsilon, \quad \vec{p}_t \in \mathcal{P}_t, \ \vec{p}_{t+1} \in \mathcal{P}_{t+1}, \tag{3.12}$$

where ε is an error threshold. The impact of a suboptimal RANSAC solution is negligible as (1) our method gracefully deteriorates with suboptimal results and (2) the step gets repeated on every frame, thus quickly corrects exceptionally bad RANSAC solutions. 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 (3.12) 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. The homography from the first frame $t = 0$ to the current frame t is then given as

$$\mathbf{H}_0^t = \prod_{k=0}^{t-1} \mathbf{H}_k^{k+1}. \tag{3.13}$$

This simple scheme only provides camera location with respect to 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 so-called *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.,* [141]) or use explicit model switching to cope with degenerate motion (*e.g.,* [177]). 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). Hence, the camera tracking continues despite a fully occluded target, although of course with a higher tracking error.

3.2.2 Assessment of camera movement

We now extend the segmentation approach with an additional condition based on camera movement. This is particularly important for splitting adjacent progressive actions, that are generally inseparable through the image distance approach alone.

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:

$$\tau(\mathbf{H}) = d\left((c_x, c_y, 1)^T, H(c_x, c_y, 1)^T\right), \tag{3.14}$$

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 if known: the optical center).

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

$$\phi(\mathbf{H}) = \frac{\max_{i=1..4} d_i}{\min_{i=1..4} d_i}, \tag{3.15}$$

with $d_{1..4}$ being the lengths of the four 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

$$\sigma(\mathbf{H}) = \log^2 \frac{d_1 + d_2 + d_3 + d_4}{8}. \tag{3.16}$$

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 $\min_{k=t-w..t} \tau(\mathbf{H}_{t-w}^{-1}\mathbf{H}_k) > T_\tau$, analogous for ϕ and 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 and generally maintains the focus of attention. Due to this, we grant a high threshold T_σ as segmentation condition. However, due to the resulting perspective distortion this indeed has a technical impact on the sampling precision of the respective maps, as described in Chapter 4.

3.3 Evaluation

We tested our approach on three, real-world use cases: The first one shows the installation of a RAM module into a notebook, see Figure 3.8.

The second shows the replacement of an empty toner cartridge of a laser printer recorded from a head mounted camera, also containing unintentional movements, see Figure 3.9.

The third shows a prototypical maintenance task in a factory environment and contains erratic head mounted camera movements and heavily cluttered background, see Figure 3.10

The notebook sequence is the only one that was recorded using a fixed camera. The remaining two sequences are evaluated to investigate the limitations of the approach.

3.3.1 Repeatability of the segmentation

As we had the feeling that unsupervised task segmentation is an ill-posed problem by nature, we have conducted a user study to find out how subjective the segmentation decisions are. The probands first watched the entire sequences and were then asked to extract all actions with start and stop frame.

To our surprise, many tasks were quite differently assessed. In the notebook sequence, there were 6 'outlier' actions that were only tagged by a single person each, compared to 2 in the printer sequence and 7 in the factory sequence.

All actions with multiple reportings were used as ground truth for valid segmentation results in our analysis of the repeatability of our proposed method. We independently segmented 10 performances (by different persons) of the workflows using our proposed method. We rated an extraction as being correct, if the segment start and stop was consistent with the manual

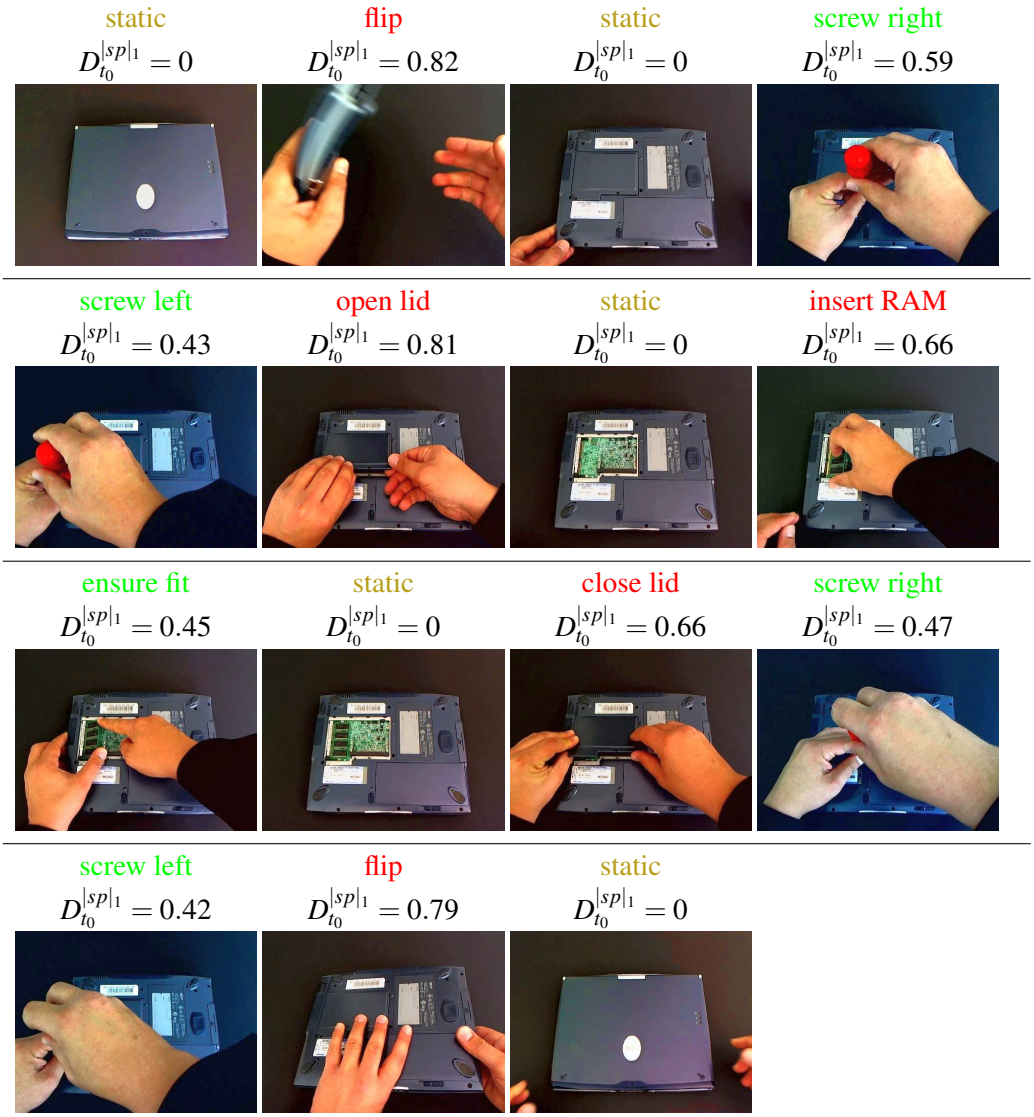


Figure 3.8: Photo story using the center frame of each automatically extracted segment from the notebook sequence. The labels are manually added and taken from the conducted user study, described in Section 3.3.1. The colors indicate progressive, repetitive, and static segments.

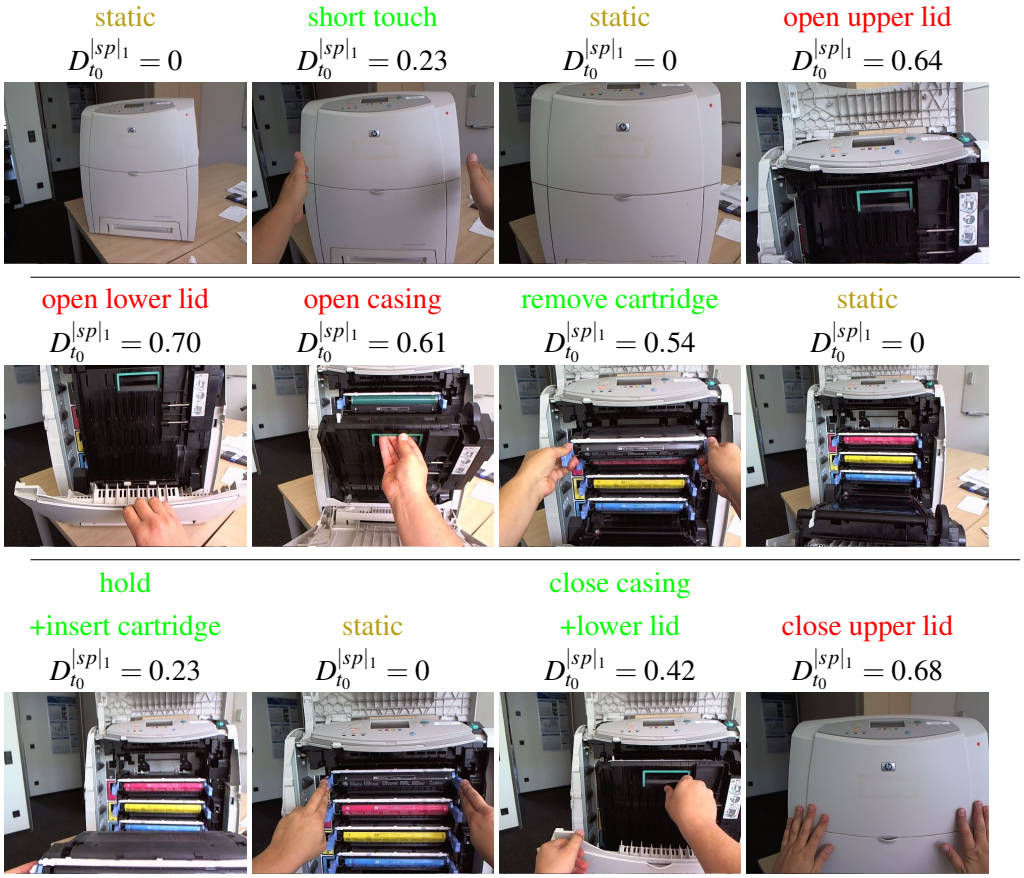


Figure 3.9: Photo story for the printer sequence. See Figure 3.8 for details.

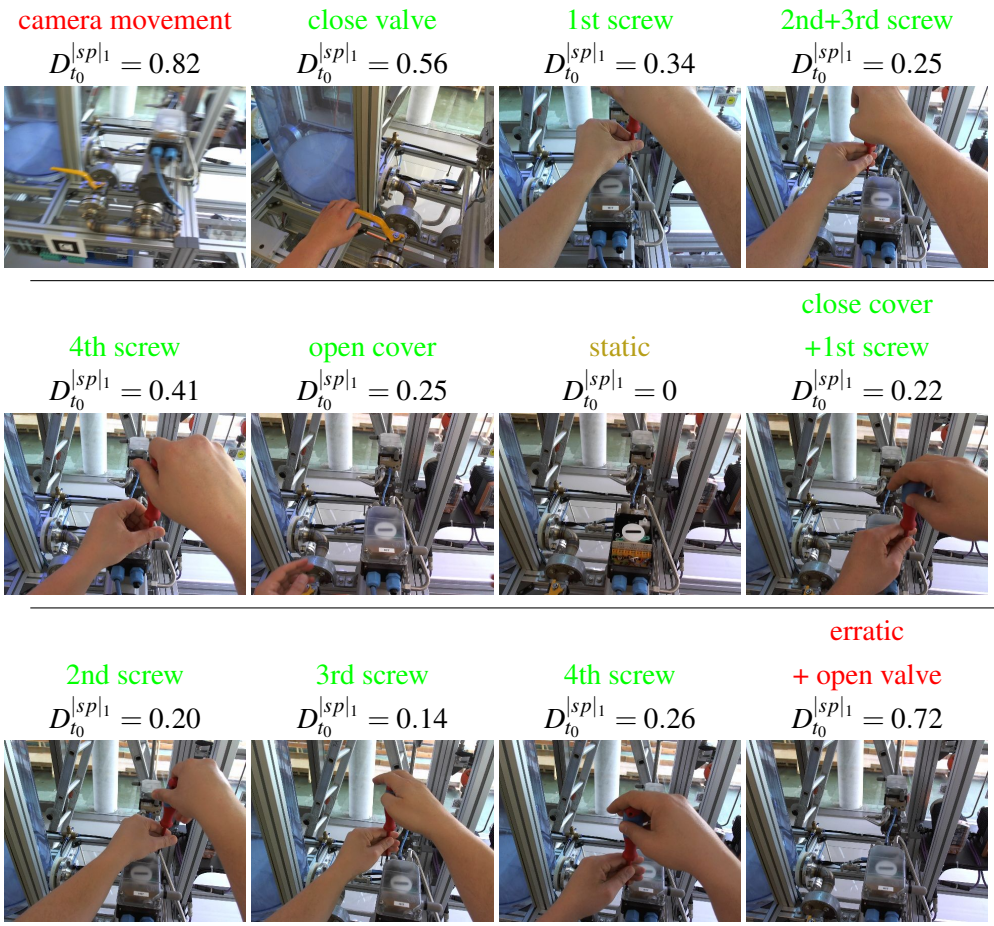


Figure 3.10: Photo story for the factory sequence. See Figure 3.8 for details.

segmentation result. Additionally, we also counted how often at least either start or stop of an action was recognized correctly, *i.e.,* only one side was correct.

For correct extractions this implies that each segment only contains exactly one action unless several users have segmented the exact same set of actions as a contiguous segment. For example, several users have not distinguished between different screws. The results of the manual and the automatic segmentation are shown in Figure 3.11.

Under consideration of cross-exclusive tasks (*together...*), our average per-task segmentation repeatability of the notebook sequence is 79% with 5 tasks being 90% and above. In comparison, the average manual task repeatability is 90%.

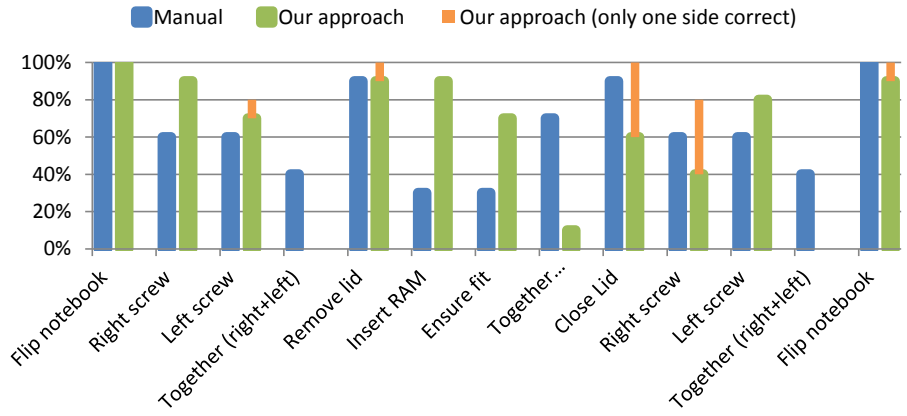
One thing to note is that in our approach the two screws were always distinguished. If there were errors, these were always in conjunction with the task before the first or after the second screw. This shows the effectiveness of our approach to distinguish repetitive segments.

In the other two sequences, missed activity changes are more frequent due to the additional challenge of a moving camera. This happens in particular between tasks that coincide with a great change in gaze direction (*e.g.,* printer sequence between upper and lower lid, factory sequence after/before closing/opening remotely located valve). While small changes in gaze direction are compensated by the design of our dissimilarity function, these larger changes conceal the user's actions and are not exploited as segmentation criteria on their own. Figure 3.11(b) and 3.11(c) give more detailed information about the per-task repeatability for the sequences with moving cameras.

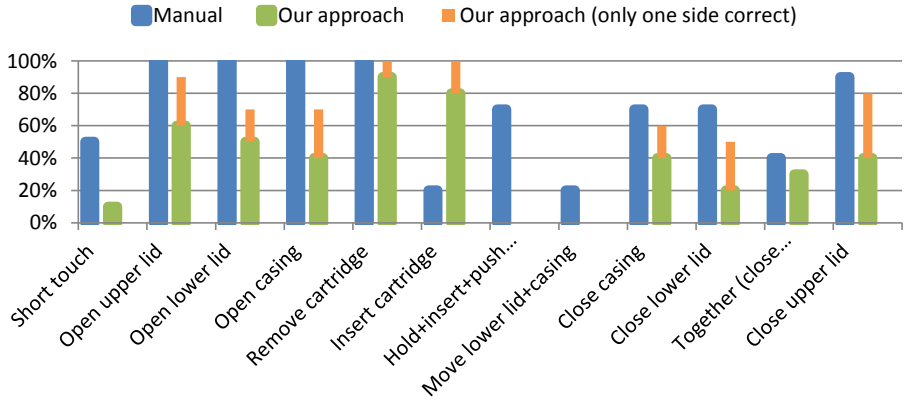
Using the center frame of each segment as representative for the segment we can automatically generate 'photo stories' as workflow documentation. We show the segmentation results for the three sequences in this representation in Figure 3.8, 3.9, and 3.10, respectively.

All pictures were automatically selected using our method, each representing one segment of the automatic segmentation. The task labels on each image are taken from the user study. The color of the task labels indicates the type of segment. If the value of $D_{t_0}^{|sp|_1}$ is above 0.6 we classify it as progressive and below as repetitive.

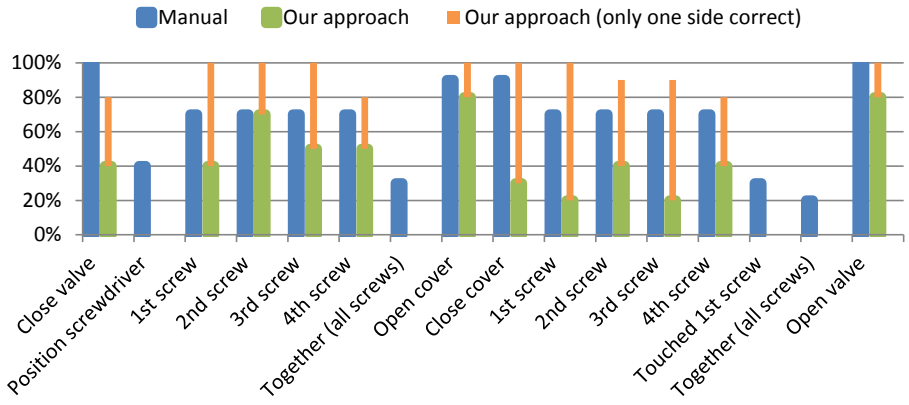
This classification works well for the sequence with a fixed camera. In both sequences with moving cameras the value tends to be too low due to large perspective distortions that are not handled by $d_{\mathcal{T}}(\boldsymbol{I},\boldsymbol{J})$ in Equation 3.1. This leads to an increase of $D_{t_0}^{sum}$ in the denominator of Equation 3.5. The effect is strongest in the factory sequence due to the highly cluttered background together with the high depth range of the recorded scene. Though, this effect only slightly affects the segmentation directly as only an increase of the function, *i.e.,* relative values



(a) Notebook



(b) Printer



(c) Factory

Figure 3.11: Task repeatability for the three sequences.

are taken into consideration. Only if the extent of camera movements prevents a saturation of the novelty rate this leads to missed activity changes.

To support this, we analyzed the temporal precision of correct segmentations in the following subsection.

3.3.2 Temporal accuracy

We consider a segmentation being temporally correct, if it falls within the minimum and the maximum frame value the probands have assigned to the start of a task. This interval is only 2 frames long for some tasks. We have investigated how often the cutting frame of a correct segmentation was within the correct interval respectively within a close perimeter of it. Figure 3.12 shows the results for an increasing amount of error (from 0 to 0.5 seconds). As expected, the notebook sequence with its fixed camera leads to the highest precision with 70% of the cut-frames being entirely consistent with the manual segmentation and 90% being not more than 0.3 seconds away. But also the factory sequence, in spite of a moving camera and a cluttered scene, leads to satisfactory results with 80% of the cut-frames being not more than 0.4 seconds away. As long as probands have not incorporated abstract task goals into their segmentation decision, our proposed criteria are well consistent with human decision making.

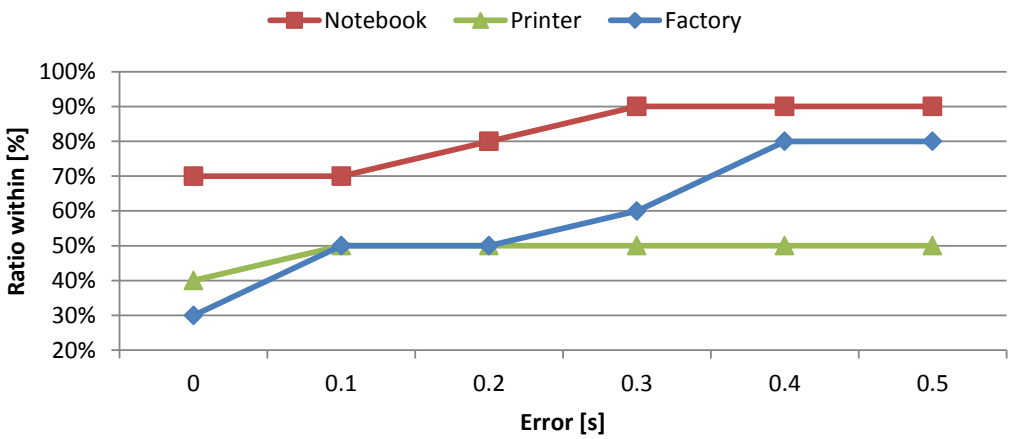


Figure 3.12: Segmentation precision for the three tasks.

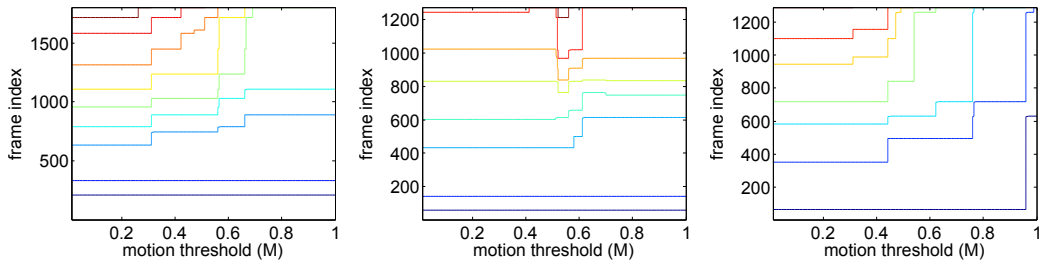
The reason why the printer sequence leads to significantly worse results (with 50% being more than half a second away from the correct interval) is not (solely) due to camera movement. The problem here is that the probands mostly attributed the first touch to the start of an action. As these are typically located very close to the image borders, they were widely ignored by

our approach due to the image crop applied in our dissimilarity function (which is needed to compensate small translational movements). Hence, only the subsequent movement of the respective printer part was then recognized as an action by our approach.

3.3.3 Impact of motion thresholding

The only parameter that was changed among the segmentation of the sequences was the motion threshold M. In the notebook sequence we used a value of $M = 0.5$ whereas all other segmentation results were conducted without thresholding the motion, *i.e.*, $M = 0$. Without the motion threshold, the segmentation result of the notebook sequence contains 2 additional segments: One additional phase before starting the screwing movement and one phase containing the approach of the hand before flipping the notebook.

We extensively analyzed the influence of the parameter M on the three sequences. Figure 3.13 shows all segmentation results for all M between 0 and 1 sampled at a step size of 0.01. A motion threshold above 0.6 is generally superseding the actual segmentation rules and leads to very few segments that are predominantly determined through the amount of visual motion. In contrast, values below 0.3 generally do not affect the segmentation result, leading to the same segmentation result as without motion thresholding. Hence, the value has a meaningful operational range between 0.3 and 0.6.



(a) Notebook (b) Printer (c) Factory

Figure 3.13: Evaluation of the influence of the motion threshold M: The graphs are sampled at a step size of 0.01. Each line represents a segment border.

4

Workflow modeling and tracking

Recognizing previously observed actions in video sequences is a prerequisite to (1) automatically follow the progress of the user and (2) still allow the automatic creation from video examples of the workflow. 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. The model elegantly handles estimation errors due to incomplete observation and is robust towards occlusions, *e.g.,* due to the user's hands. This allows to model the time-evolving state of the 3D workspace and the user actions using simple 2D region descriptors. We will present and discuss this approach to spatiotemporal modeling and tracking in Section 4.2.

While the spatiotemporal tracking is obviously necessary during run-time to provide context-aware assistance while the user is performing a workflow, many aspects are already needed during authoring. As we point out in our evaluation section, a single reference recording generally does not suffice to establish a model, due to user dependent variations. Therefore, we need to temporally and spatially align recordings already during authoring to capture the slightly differing performances of a workflow. Furthermore, this is a prerequisite to generalize the state transition model in order to also capture valid variants of the workflow.

Since we deal with a moving camera, we need to recognize and stabilize the region of interest. The first-person view workflow recordings that we deal with exhibit some quite unique

and specific properties. We briefly summarize the key aspects to motivate our tracking design decisions:

Camera motion and viewpoint: The 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. Especially, when aiming for creating AR documentation as a by-product of ordinary maintenance or assembly work, the system needs to deal with difficult conditions, erratic motion, and incomplete observation.

In the next section, we will introduce the relevance plane transform, an image transform that allows the piecewise modeling of a time-progressing environment using standard 2D descriptors. After that, we present the classification approach that we use to switch between temporal work states and to roughly initialize a camera pose. The third section explains how we apply this during authoring and the chapter concludes with an evaluation of the tracking approach.

4.1 Relevance plane transform

For modeling a dynamic, continuously changing environment, we propose a piecewise homographic transform that projects the given video material onto a series of distinct planar subsets of the scene. 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 within the ROI 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 4.1 illustrates the model and this consideration.

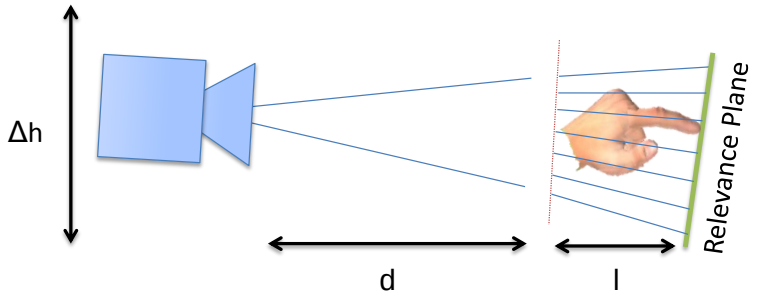


Figure 4.1: Illustration of the model assumption: The relevance plane transform (RPT) provides accurate estimates if Δh and l are small compared to d.

The idea is related to the influential tracking approach by Simon *et al.* [208] that also uses planar structures in the scene for tracking. In contrast to their approach, we propose a selection and segmentation approach that copes with the dynamic scene content. While our approach is equal to whole image stabilization in case of purely rotational motion, the tracking support incrementally converges to the planar structure with translational motion. By exploiting the fact that with degenerate (*i.e.,* purely rotational) motion, the entire image sharply projects into the common frame, we can sample information without actually estimating the relevance plane. The segmentation is then delayed but can still take place during online tracking. As pointed out in the beginning of this chapter, the camera motion during a single work step exhibits only little translational motion. However, the translational offset might be significant between different users.

We now show how the various ROIs are selected and how we robustly segment the largest support region for an ROI.

4.1.1 Selecting the region of interest

We change the ROI and therefore possibly the planar structure that constitutes the relevance plane on every task of the workflow. To that end we use the segmentation criteria from the previous Chapter 3, including strong camera movement as a cue for a changed region of interest.

We further exploit the task structure to determine the region that is currently the focus of attention. We start with temporal segments $\boldsymbol{\mathcal{S}}_i$ that have been classified as containing user

actions. For robustness, we use a fuzzy representation of the user's focus of attention within each image $\boldsymbol{I}_t \in \boldsymbol{\mathcal{S}}_i$. We simply use an attention mask $\boldsymbol{M}$ that is 1 in the image center and radially fades out to 0 to model the focus of attention. In the following subsection, we will show how the region of interest is updated to segment the relevance plane.

We also tried to define the region of interest using the area of greatest optical flow, around the centroid of the hand silhouette, at the location of the fingers estimated through [209], and through combinations of the three but found that this approach worked most reliably in practice.

4.1.2 Segmenting the relevance plane

Segmenting the RP is quite analogous to our camera tracking scheme described in Section 3.2.1 with simple adjustments:

Altered step 1: We again select corner features $\boldsymbol{\mathcal{P}}_t$ within frame $t = 0$. However, in contrast to the camera tracking approach, we constrain the selection to a support region, which in the first frame is given as

$$\boldsymbol{R}_{i,0} = \text{thres}_{\kappa} M, \tag{4.1}$$

where thres_{κ} is a binary image threshold operator with threshold value κ. The remainder of step 1 is analogous to the camera tracking method, *i.e.,* finding correspondences using KLT and using RANSAC to find a large subset of correspondences whose movement can be described using a homography.

Altered Step 2: The support region $\boldsymbol{R}_{i,t+1}$ is being updated using the density map $\boldsymbol{D}_t$ of the currently tracked features before rejecting points that do not comply with $\mathbf{H}_t^{t+1}$. Simply put, $\boldsymbol{D}_t$ is created by drawing blurred circles around each feature location in frame t. The updated support region is then given as the weighted average

$$\boldsymbol{R}_{i,t+1} = \text{thres}_{\kappa} \left(\alpha \boldsymbol{M} + \beta \, \text{warp}_{\mathbf{H}_t^{t+1}} \boldsymbol{D}_t \right), \tag{4.2}$$

where α and β are weights and $\text{warp}_{\mathbf{H}_t^{t+1}}$ warps the density map using the homography $\mathbf{H}_t^{t+1}$.

Figure 4.2 illustrates how the support region is propagated. Similar to the camera tracking approach, we then select corner features across the entire image that comply with the homographic model determined through the point trajectories within the support region.

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

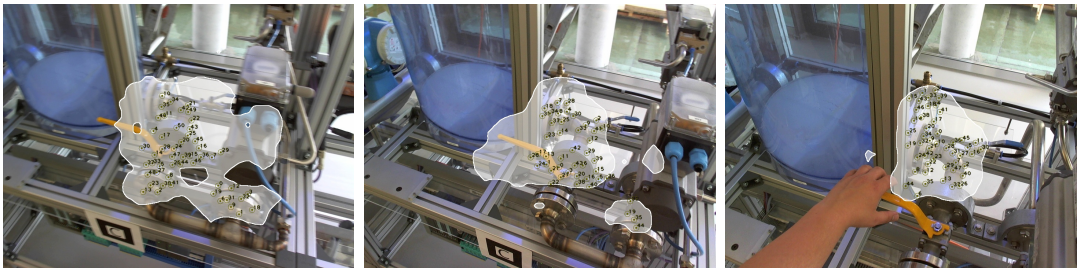


Figure 4.2: Support region while tracking the relevance plane.

support drifts to a planar subset that satisfies (3.12). Note that due to the radial distortions of an uncalibrated camera, the support will not span the entire planar structure.

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.

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

$$\hat{\mathbf{H}}_t^{RP} = (\frac{1}{|\boldsymbol{\mathcal{S}}_i|}\sum_{k=1}^{|\boldsymbol{\mathcal{S}}_i|}\mathbf{H}_0^k)^{-1}\mathbf{H}_0^t, \tag{4.3}$$

where $|\boldsymbol{\mathcal{S}}_i|$ is the number of images in the segment and $\mathbf{H}_0^t$ is the homography from first frame of the segment to t, as given in Equation 3.13. This type of linear interpolation between homographies is along the circular secant, not the arc. It will therefore degenerate 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.

We can use Equation 4.3 to project each frame of a temporal segment into a single 2D frame that affords the application of 2D descriptors such as skin color probability maps, illustrated in Figure 4.3. By backprojecting the common frame into the workspace, these descriptors can be applied during tracking. In the following section we explain how this backprojection is realized using a robust classification approach. As each relevance plane is specific for a temporal segment, this approach is intertwined with temporal tracking of the user's progress within the workflow.

4.2 Spatiotemporal classifiers

Our approach is based on an independent classification of each camera frame using a k-nearest neighbors (k-NN) classifier on the novelty frames $\boldsymbol{\mathcal{N}}_{t_0}(t)$. These frames were acquired through

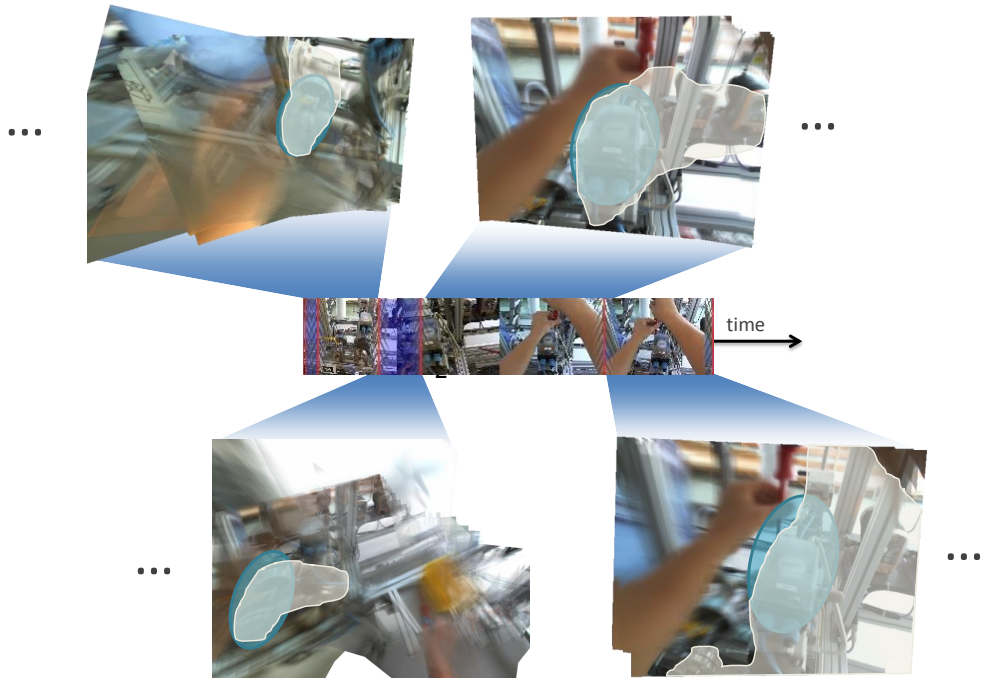


Figure 4.3: The relevance plane transform allows the projection of a time-progressing 3D workspace into a piecewise 2D representation.

the temporal segmentation approach described in the previous chapter. The classifier returns a set of hypotheses for the current segment index as well as a rough viewpoint estimate. The viewpoint is determined from the transform parameters leading to the minimal distance in the robust distance function $d_{\mathcal{T}}(\boldsymbol{I},\boldsymbol{J})$ (Equation 3.1). This leads to a quantized 4-DoF pose estimate (rotation around the camera axis, 2D translation, and scaling).

We propose a set of classifiers that are trained for every segment independently. To train a segment classifier k-NN$_i$ we first use all frames $\boldsymbol{I}$ from segment i as positive training examples. The procedure works as follows: We start with an empty set of neighbor samples. For every frame treated as a positive training example we create affinely transformed images $\mathcal{T}(\boldsymbol{I})$, as described in Section 3.1.1. Every image in $\mathcal{T}(\boldsymbol{I})$ that has a distance of at least T from all other neighbor samples is added as a new neighbor. In our implementation we use 9 rotation values on 9 different scales, thus resulting in 81 images per training example since translation is already handled by the underlying region descriptor.

We proceed differently on static and non-static segments. If segment i is non-static we use the previous and the following segment as negative training examples. If the workflow contains

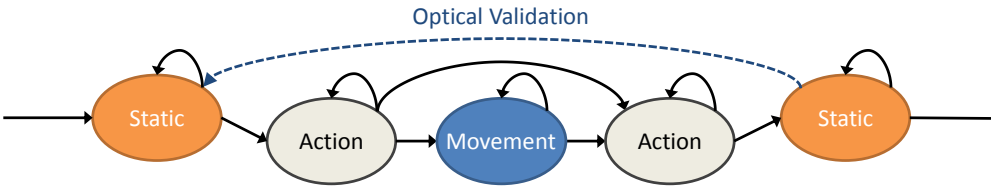


Figure 4.4: Illustration of the possible state transitions dependent on the type of segments.

structural variants in the transition model, there could be several successors and predecessors. In that case all of these segments are used as negative training examples.

For every affinely transformed image of each training example we calculate the distance towards all neighbor samples and remove samples that have a distance of less than T. So, for each frame from the segment i, there exists at least one nearest neighbor within a radius T. For each negative training example, there exists no neighbor sample within a radius T.

Each neighbor sample is labeled with the transformation parameters that were applied. These are the rotation value r, scaling s and translation t (provided by the region descriptor). With this information, we know how to transform overlaid images to produce roughly congruent overlays, *i.e.,* a quantized 4-DoF camera pose. As already mentioned, this suffices to produce acceptable results for small changes of the elevation angle.

We treat static segments differently for two reasons: Firstly, the probability for matching these images is higher since the segment potentially only shows rigid and static background. Secondly, the respective k-NN$_i$ will contain very few (but important) samples. Therefore, the procedure for removing negative training examples is not applied here. However, since we remove ambiguous matches from the neighboring classifiers, the separation is still guaranteed to be at least T.

With $NN(l)$ denoting the l^{th} nearest neighbor and $d(\boldsymbol{I},\boldsymbol{J})$ being the distance function defined in Section 3.1.1, the scoring function is

$$\text{score}^i_{\text{NN}}(\boldsymbol{I}_t) = \frac{1}{k}\sum_{l=1..k} 1/d(\boldsymbol{I}_t, \text{NN}(l)). \tag{4.4}$$

We experimented with different values of k and found that a value of $k = 2$ improved the precision and substantially helped to reduce the spatial jitter. Considerations in defense of the performance of nearest neighbor classifiers are found in [210]

The temporal state transitions are based on a Markov process, illustrated in Figure 4.4. In each segment state we may either remain in the current segment or move to its follow-up

segment state. Since we expect segments that were segmented due to strong camera movement not to be robustly recognizable, we allow skipping these and directly move to the next non-movement state. Generally, we do not allow transitions back to a previous state in the tracking process. However, as static segments might provide a direct view on the workspace, we utilize this to validate the intermediate work step results. Therefore, in case of clearly separable static segments that are not occluded by the user's hands, we allow a backwards transition to the previous static segment. A description of how the model for this optical validation is extracted from the reference material is provided in Section 6.3.3 and 6.3.4 .

Our transition rule is based on a simple hysteresis approach: For each possible follow-up state n we compute $score^i_{next}$, including the current state $score^i_{cur}$. If a follow-up state produces the maximum score for a certain number of consecutive frames we transit to this state. This implies that a single vote for the current state resets this counter.

The distance function for the k-nearest neighbors is an insufficient measure to compute the scores. Therefore, we compute the scores by applying the 2D descriptors within the relevance plane. In the following sections, we show how we robustly refine the camera pose to backproject the relevance plane. This is demonstrated using a simple hand location probability map as 2D descriptor, which is briefly explained subsequently. After this, we show how we compute the final scoring function used for temporal tracking.

4.2.1 Refinement of the camera pose

We begin with the region template matching approach proposed in Section 3.1.1 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. The reason why we resort to a two-step camera initialization method is due to the dependency of the point matching approach on sufficient texturing. Although the proposed method of recovering the pose estimate is considerably slower than exclusively relying on the point matching approach, it is highly robust towards lack of texture as well as occlusions. We did not pursue the alternative of using a contour or edge matching approach (*e.g.,* [211]). Since we have to deal with a high amount of occlusions, the robustness of a contour model is compromised whenever contours are partly occluded. In our approach using region descriptors, we counter this through joint sampling of hand and environment. The proposed approach consists of the following steps:

Build point descriptors: In an offline step, we compute ORB [171] 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 with the averaged keypoint position and the descriptor with the lowest summed distance towards all others within the merge set.

Matching: During tracking, 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 $\sigma(\mathbf{H}_4^{-1}\mathbf{H}_6)$ (compare Section 3.2.2). 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 3.2.1, while maintaining the support region of the relevance plane which results in the homography:

$$\bar{\mathbf{H}}_t^{RP} = \prod_{k=1}^{t} (\mathbf{H}_k^{k+1}) \mathbf{H}_{t=0}. \tag{4.5}$$

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.

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

4.2.2 Hand location probability maps

We store the location probability of the user's hands in a 2D map, using the common frame of the relevance plane, *i.e.,* for each temporal segment, separately. We first segment the hand silhouette mask $\boldsymbol{S}_t$ based on skin color segmentation for every image $\boldsymbol{I}_t \in \boldsymbol{\mathcal{S}}_i$. 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 [212]. The location probability map is then the normalized average $\boldsymbol{S}_i^{RP} = \frac{1}{|\boldsymbol{\mathcal{S}}_i|} \sum \text{warp}_{\bar{\mathbf{H}}_t^{RP}} \boldsymbol{S}_t$, where $|\boldsymbol{\mathcal{S}}_i|$ is the number of images in segment i. Figure 4.5 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 4.5. A very low or zero

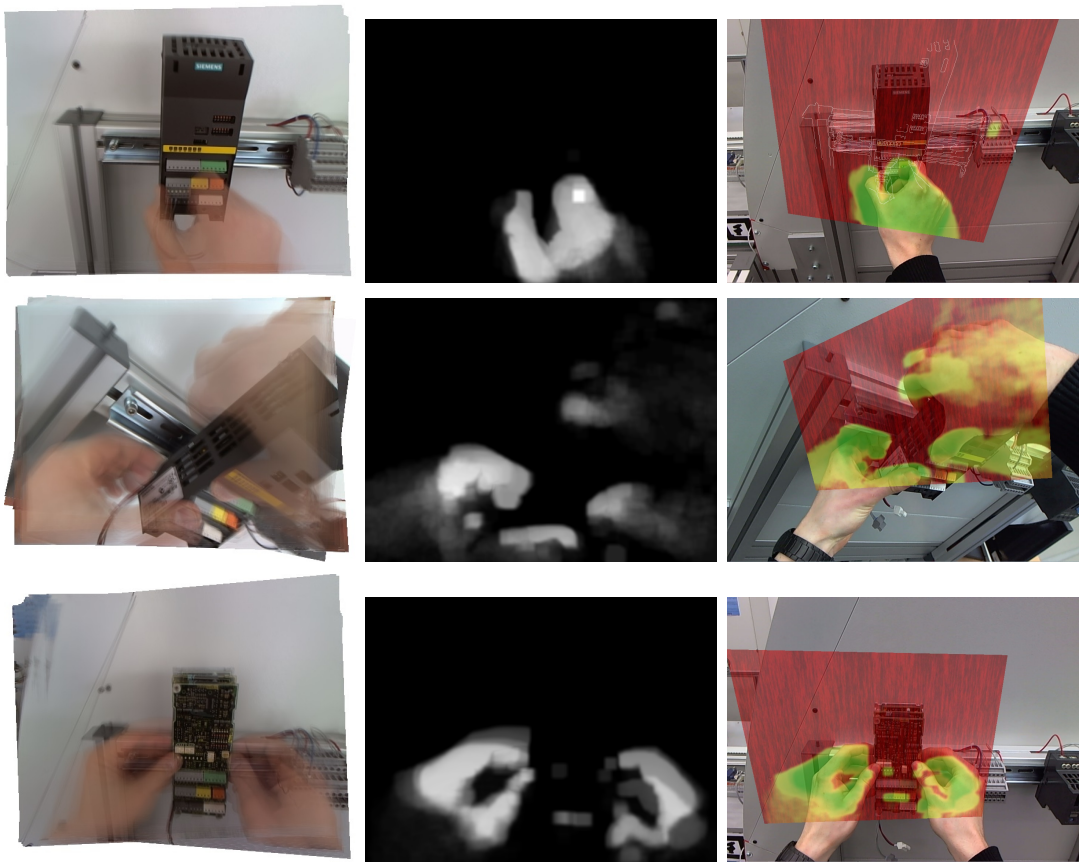


Figure 4.5: Images projected into the common frame and averaged **(left)**, hand location map **(middle)**, hand location map backprojected into the field of view **(right)**.

location probability is indicated as red, low as yellow, and high probability as green. Compare Section 6.2 for a discussion of this feedback from an application perspective.

4.2.3 Extended scoring function

We extend the original scoring function, which was formulated in Equation 4.4 with the 2D descriptor scores that are projected back into the camera frame using the homographic transform $\text{warp}_{\bar{\mathbf{H}}_t^{RP}}$. Using the 2D hand location maps, we therefore formulate the final scoring function as

$$\begin{aligned} \text{score}^i(\boldsymbol{I}_t) = \alpha\, &\text{score}^i_{\text{NN}}(\boldsymbol{I}_t) \\ &- \beta\, \text{count}(\text{thres}_\kappa(\boldsymbol{S}_i^{RP}) \otimes \text{warp}_{\bar{\mathbf{H}}_t^{RP}}(\boldsymbol{S}_t)), \end{aligned} \tag{4.6}$$

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

One important aspect to note is that we do not apply $\text{score}^i_{\text{NN}}(\boldsymbol{I}_t)$ in the common 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. While this explicitly narrows the allowed deviation of the user's point of view from that of the reference recording, it also assures that the tracking model is able to describe the observation. In our framework, we use an attention funnel to guide the user back, if he wanders off too far. To allow for a wider range of viewpoints, it is possible to add another reference recording from a different perspective. In order to incorporate this into a single tracking model, the recordings need to be registered spatially and temporally, which is described in the following section.

4.3 Learning from multiple sequences

There exist three different motivations for adding additional reference recordings into the training data body:

Viewpoint generalization: As the method extracts the entire workflow knowledge from monocular video examples, the generalization to arbitrary viewpoints is infeasible without very restrictive assumptions on scene geometry.

User dependency: Due to variation in the mode of execution, significant differences in the size or shape of the hands, or right vs. left handedness it is generally necessary to accommodate these differences with additional training examples.

Task variants: Finally, there might be different viable solutions to the same workflow. For example, the order of releasing screws might not be important and all orders lead to a correct completion of the task. These variants can be automatically extracted from the training material, analogously.

In contrast to the problem of online workflow tracking, we can assume and exploit the complete availability of the whole sequences. In the following, we explain how we adapt the tracking method to improve the recognition results for this offline case.

4.3.1 Temporal alignment

In this section we explain our method to temporally align another prerecorded workflow video in an offline process. This is required for associating learning data and allows the teach-in of appearance and viewpoint variations as well as task variants that change the order of work steps. Therefore, the approach needs to be able to detect inserted, left out, or reordered steps and accordingly update the transition model.

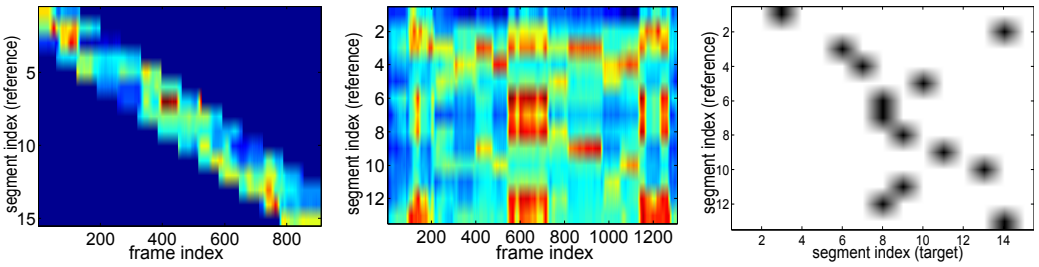
There exist approaches that are quite robust towards viewpoint changes in an offline case, *e.g.,* [159], through observing the self-similarity of the sequence and matching the resulting patterns. Although this constitutes an interesting approach, it is not applicable if the step order is permuted locally or otherwise changed. We handle both the online and the offline case with the k-nearest neighbors (k-NN) approach that was presented in the preceding section. As one of the main premises of our work is the assumption of a narrow confidence radius, a classification rule based on k-NN reduces the relevant evaluated distances to a minimum.

We distinguish two different cases of task variants:

Appearance variants that do not change the segment structure but change the appearance of the segments (*e.g.,* the same workflow performed by a left- and a right-handed person or from a significantly different viewpoint).

Structural variants that change the order or number of task segments (*e.g.,* releasing screws in a different order).

In case of appearance variants we use dynamic programming to find an optimal path through the given set of segments. The approach is very similar to dynamic time warping except that we do not frame-wise align the two sequences but rather on the granularity of frame-to-segment. If it is known that the additional recording constitutes an appearance variant, we can improve the stability of the alignment procedure by only allowing $\pm 20\%$ time fluctuation between the two sequences (Sakoe-Chiba band), compare Figure 4.6(a).



(a) Appearance variant (b) Structural variant (c) Association matrix

Figure 4.6: (a) and (b) show the score matrices used for alignment. (c) shows the association matrix to associate structural variants.

In the case of structural variants there exists an immanent problem: It is very hard to decide whether a low matching score of all segment classifiers is owed to strongly differing image appearance or because the image really does not belong to any known segment, see figure 4.6(b) for an example of the according score matrix. To solve this, we exploit the possibility to independently segment the second sequence with the method described in Section 3. We assume that we have successfully recovered all tasks in the target sequence (or at least, we live with the fact that wrongly segmented tasks consequently get associated incorrectly). We then calculate the average matching score over entire target segments and apply non-maximum suppression per source segment. Figure 4.6(c) shows the resulting association matrix for the printer workflow. After reordering the sequence using the association matrix, we can again apply DTW to fully align the sequences.

4.3.2 Spatial alignment

After the temporal alignment, we also need to register the relative camera poses among the frames of both sequences. Frames containing user interaction are very difficult to match among different recordings due to occlusion by the user's hands. We therefore exploit the fact that we already have tracked the camera independently within each sequence with the approach described in Section 3.2.1. Since we can associate all frames of the added video recording with a respective segment of the reference sequence due to the temporal alignment, we also know the respective classification into static and non-static. We therefore register the camera pose among different recordings only within static segments and compute the relative viewpoints of remaining frames separately for each sequence. Figure 4.7 illustrates the procedure.

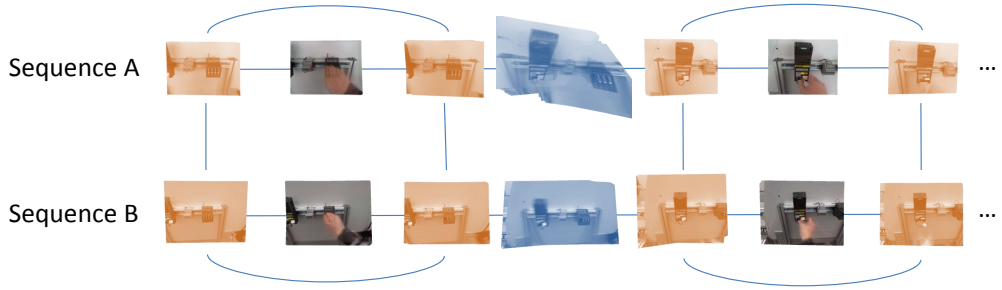


Figure 4.7: Illustration of camera pose registration between several recordings: The blue lines indicate a camera pose matching between yellow-tinted static segments and the pose updates for each sequence individually. In particular the segments that contain strong camera movement (blue-tinted) are handled individually.

For each frame pair from the associated static segments, we apply the procedure as described in Section 4.2.1. For the respective frames within each static segment of the added sequence, this leads to two different homography estimates: One which is relative to the linearly averaged homography of the segment of the added sequence, as formulated in Equation 4.3. And one that is relative to the analogously determined average of the segment from the reference sequence. For a shortened notation, we denote the reference sequence with A and the added sequence with B. Since we can relate the pose of each frame of the sequence via concatenation of $\mathbf{H}_k^{k+1}$, this leads to an overdetermined equation system, which we solve independently for each static segment:

$$\mathbf{H}_k^{k+1}\mathbf{H}_{A_l}^{B_k}\vec{p}_i = \mathbf{H}_{A_l}^{B_{k+1}}\vec{p}_i, \forall l,k,i, \tag{4.7}$$

where $\mathbf{H}_{A_l}^{B_k}$ is the homography from a frame l in sequence A to a frame k of sequence B, $\mathbf{H}_k^{k+1}$ denotes the homography from frame k to $k+1$ of sequence B and $\vec{p}_i$ denote the feature points in sequence A.

This results in a robust estimate of the relative homography for each static segment. Since the relevance plane is readjusted for each temporal segment, the estimates for the relative homographies between reference sequence A and added sequence B $\mathbf{H}_A^B$ only need to be propagated locally. To this end, we use the two closest static segments before and after a frame and assign the linearly interpolated homography as relative pose.

4.4 Evaluation

We have evaluated the applicability of the tracking approach with three data sets that differ fundamentally in their properties. For baseline, we included the *"Notebook"* sequence that was also used for evaluation in the preceding chapter (Figure 4.8). Due to the fixed camera, this is a direct evaluation of the impact of the applied 2D hand location descriptor, which has been described in Section 4.2.2.

"Lever & lid" (Figure 4.9), which was also used in the last chapter, exhibits few trackable planes, the angle between the relevance planes and the image plane is quite large, and it comprises erratic camera motion.

The newly added *"Plugs & circuit board"* data set (Figure 4.10) exhibits many large planes, coarsely aligned with the image plane and relatively steady camera motion. We added this sequence as a best-case scenario that is simplified but not uncommon in a factory environment.

We concentrate on three different aspects of the approach: (1) the performance of the classifiers for tracking the temporal progress of the user. This was examined through the ratio of correctly classified frames as well as the score margin as a measure for the robustness of the decisions. (2) The impact of adding additional training recordings to the classification performance. (3) The spatial accuracy of the reprojection. These are covered in the three following subsections.

4.4.1 Spatiotemporal classification

To analyze the tracking performance, we recorded each scenario twice and used the first for training 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 scoring function incorporating the 2D descriptor (Equation 4.6), thus using a model-representation based on the RPT. The results are denoted as "proposed" in the following graphs. We tracked the same sequences with the scoring function solely based on the k-NN classifier (Equation 4.4). This is denoted as "Petersen2012".

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 i minus the highest adjacent segment classifier: $m^i(t) = \text{score}^i(\boldsymbol{I}_t) - max\left(\text{score}^{i+1}(\boldsymbol{I}_t), \text{score}^{i-1}(\boldsymbol{I}_t)\right)$. The number of correct classifications is

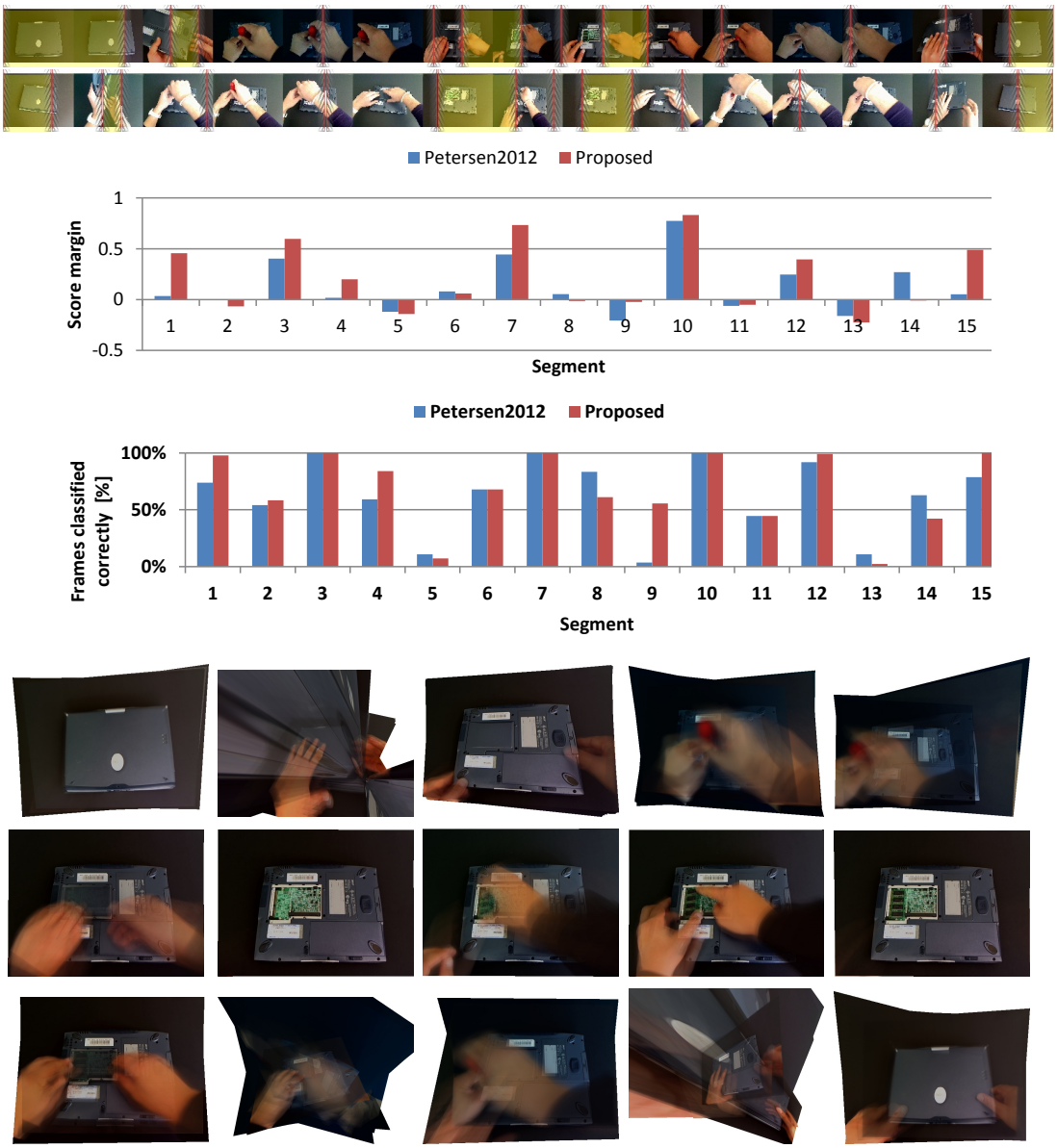


Figure 4.8: Results for the *"Notebook"* sequence: Temporal segmentation of reference and test sequence showing action segments and yellow-tinted static segments **(top)**, descriptor score margins and correctness **(middle)**, and common frames for all temporal segments **(bottom)**.

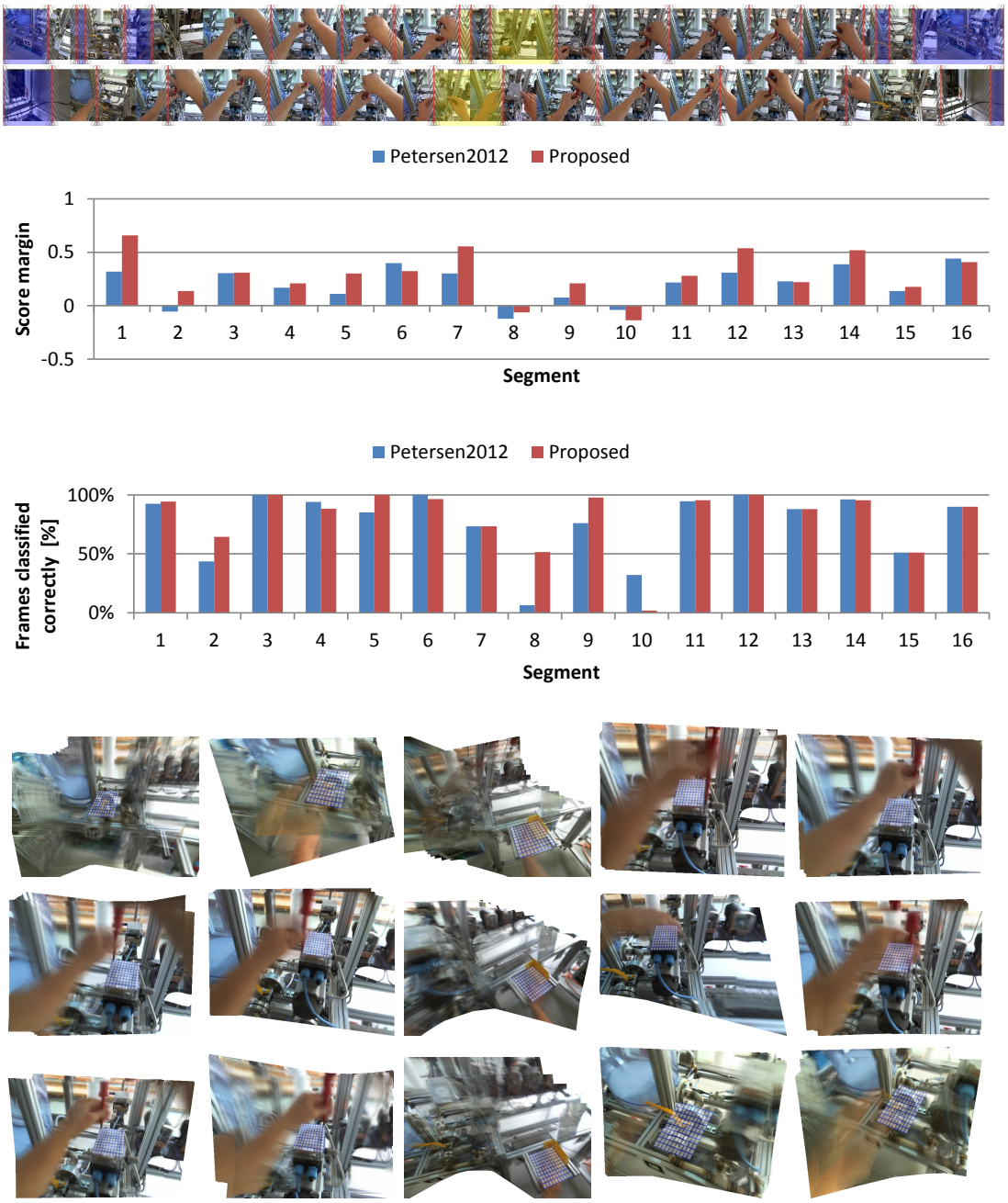


Figure 4.9: Results for the *"Lever & Lid"* sequence: Temporal segmentation of reference and test sequence showing yellow-tinted static-, blue-tinted movement-, and action-segments **(top)**, descriptor score margins and correctness **(middle)**, and common frames for all temporal segments **(bottom)**.

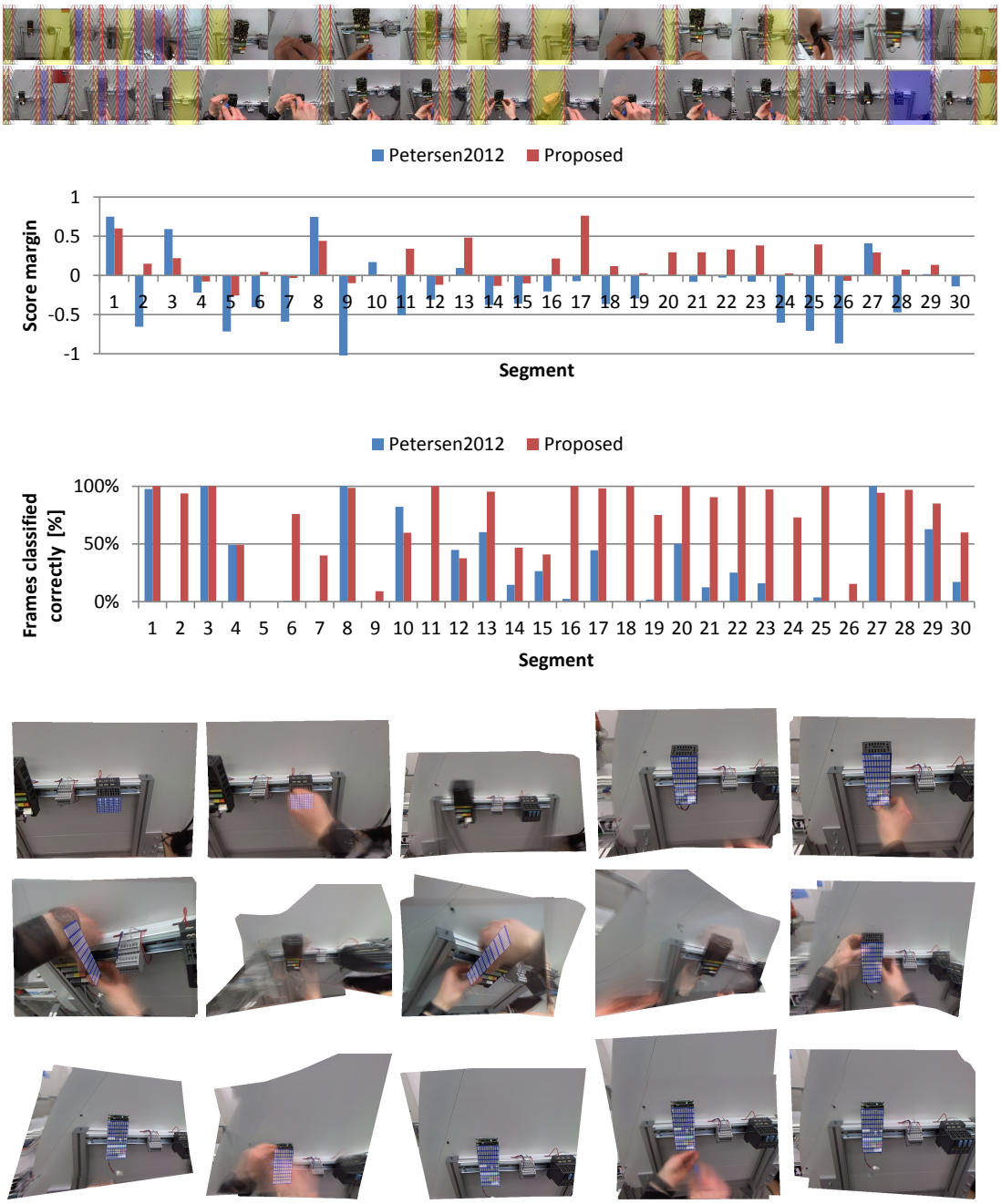


Figure 4.10: Results for the *"Plugs & Circuit board"* sequence: Temporal segmentation of reference and test sequence showing yellow-tinted static-, blue-tinted movement-, and action-segments **(top)**, descriptor score margins and correctness **(middle)**, and common frames for a selection of temporal segments **(bottom)**.

	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%

Table 4.1: Tracking performance comparison.

then given through counting $m^i(t) > 0$. The results are shown in Figure 4.8, 4.9, and 4.10 for the three data sets, respectively. Additionally, Table 4.1 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 4.8), 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 an on average negative margin -21 to 15.8 and these increases are spread among almost all segment scores, compare Figure 4.10.

On the other side, the *"Lever & lid"* data set 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.

4.4.2 Multiple training examples

We also investigated the influence of the number and kind of training examples on the example of the notebook sequence. We recorded the notebook sequence by 5 different persons (2 female, 3 male in the order F, M, M, M, F). Additionally, a 6th person (male) has recorded a total of 6 demonstrations of the workflow, 5 used for training and 1 for testing. Figure 4.11 shows

example frames in the order of appearance. The gender is important, since it has a large impact on the visual appearance and thus the achieved scores.

We then again manually aligned the respective performances to ensure that we do not measure the influence of alignment errors. We evaluated the influence on the segment classifier score in three experiments, see Figure 4.12:

- Trained with 1 to 5 examples from a single person and tested with a 6th example from the same person *(single → same)*.
- Trained with 1 to 5 examples from a single person and tested on the examples from the 5 other persons *(single → other)*.
- Trained with 1 to 5 examples from a different person, each and tested on an example of the 6th person *(multiple → other)*.

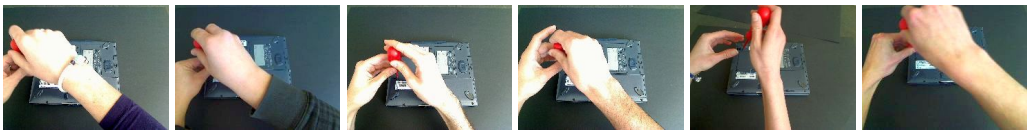


Figure 4.11: Example frames from the 6 different persons recorded for the training evaluation.

Experimentally, we determined that a minimum average score of 0.65 is required for the tracker to successfully follow the test sequence to its end. The reason, why we do not condition the value on the score margin is due to its higher dependency and thus higher variation among different temporal segments. Figure 4.12 shows the results.

As expected, the system achieves the highest scores when being trained with the same person that uses the system. With a single training example the system achieves an average

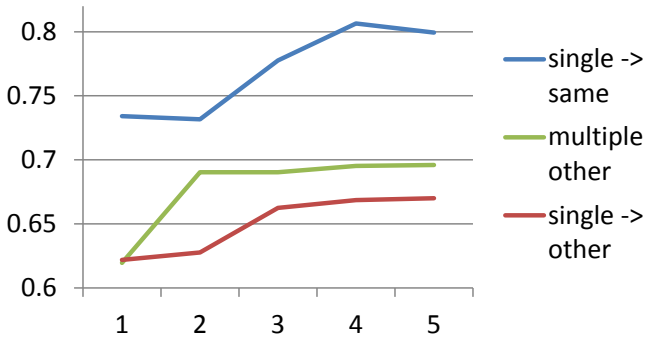


Figure 4.12: Average classifier score after 1-5 training examples.

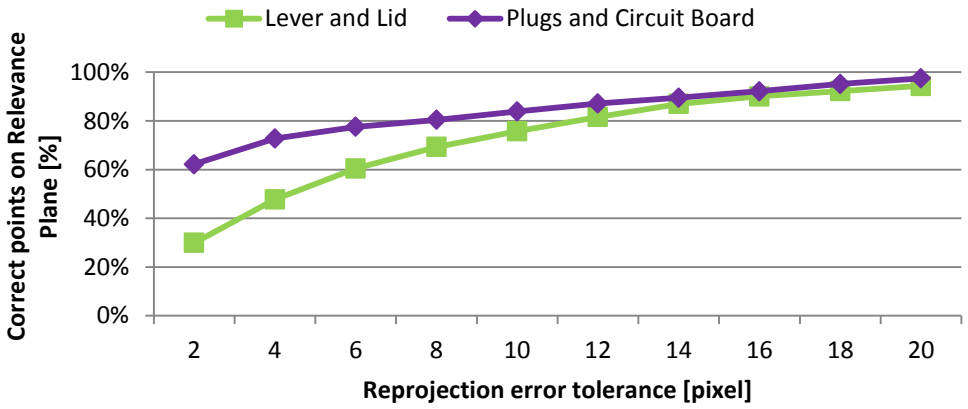


Figure 4.13: Reprojection error in pixel (underlying image size is 960×720).

score of 0.73 and climbs up to 0.8 after the 4th example. The 5th example does not further improve the result. Due to clustering and a similarly larger set of negative training examples the score even falls off slightly. When the system trained like this is applied to examples from unknown users, the score is only 0.62 on average which was not enough to track through the entire workflow. When being trained using all five examples, this value climbs up to 6.7, which is just enough to track the entire workflow. When the system is trained with examples from different users, the scores improve due to the increased variability. After only training with the first example (provided by a female user), we again achieve very low scores, since the test set is recorded by a male user. Already after training with the second example (showing a male user) this score climbs up to 0.69 and almost stays constant after training the remaining three examples. This clearly indicates that the morphological differences between users have a large impact on tracking performance. This also explains the significant improvements that can be achieved trough the model-guided generalization procedure, described in the following chapter.

4.4.3 Reprojection accuracy

Additionally, we evaluated the reprojection accuracy in the two data sets recorded with a moving camera. To that end, we first computed the RPT for each segment as described in Section 4.1.2. 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 4.9 and Figure 4.10, respectively. For every image $\boldsymbol{I}_t \in \boldsymbol{\mathcal{S}}_i$ projected into the common frame, we have

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 $t = 1..n$ in the segment. The reprojection error of a single point $\vec{p}_t$ is then taken as: $e(\vec{p}_t) = \vec{p}_t - \frac{1}{n}\sum \vec{p}_t$ 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 4.13.

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 with respect to an image of 960×720 pixels.

5

Hand and finger tracking

While our entire approach so far has been based upon robust but coarse models, we now extend this with detailed information gathered through hand and finger tracking. This has three important applications within our framework:

The first is acquiring detailed information about the execution modalities such as hand postures, trajectories, and velocities during each work step. We call this kind of information *enactive knowledge*, as it can be used to guide and support the user during the psychomotor phase [165].

The second is the assessment of the required level of precision within each work step. The classification models, presented in the previous chapter, already allow identifying unnecessary or unintentional work steps as a whole. Through analysis of recurring hand postures among several recordings of the same work step, we are able to further specify each step. Through this comparison, we are able to distinguish work steps that require a relatively high precision, *e.g.,* pressing a certain button from steps that are less determined, *e.g.,* picking up a randomly placed tool.

The third application is the model guided generalization of the tracking model. In order to reduce the required number of reference recordings, we are explicitly generating additional, synthetic views, based on the already available observation. To this end, we present a novel approach to image-based rendering of articulated objects that gives reliable estimates of the object's shape and shading in new, previously unseen poses. It faithfully approximates both shape and shading of a hand in an unseen target pose even despite large unobservable hand parts in the images that were used as prototype views.

In order to estimate the necessary hand posture parameters, we propose an entirely novel hand tracking approach that is able to adapt to the actual observation. Using this approach it is possible to track the challenging input material, exhibiting recurrent dis- and reappearing of the hand, occlusions through hand-held objects, and fast and erratic motions. To the best of our knowledge, this work contains the first description of a method that is able to operate on such input data using with a single RGB camera.

We begin with a detailed explanation of how we model the appearance of the hand (5.1) using our proposed extension to billboard rendering (5.1.1) and the method for efficient, axis-aligned morphing (5.1.2). After that, we present the adaptive tracking approach in Section 5.2 that allows the adaptation of the tracking model to the observerd content. In the extensive evaluation Section 5.3, we make the case that a derived pixel-wise distance function vastly outperforms distance functions based on skin or edge features, used in current state of the art methods. Using synthetic, ground truth labeled data, we demonstrate that the proposed function is highly robust against image blur, occlusion, cluttered or skin-colored background and exhibits significantly less local optima than the state of the art. Additionally, in Section 5.3.4, we are able to demonstrate the significant improvement of the generalization approach using actual workflow videos as an input.

5.1 Image-based appearance model

We present a method to synthesize a low-textured, articulated object from other views through image-based rendering (IBR) that can be used to explore the parameter space between a growing set of nearest neighbor templates. Figure 5.1 shows examples using synthetic images both with and without unobservable areas to illustrate the procedure.

The method can also operate on real observation, recorded with a monocular RGB camera. Figure 5.2 shows examples of interpolating between two frames of a real-world image sequence. The morphed views in between are synthesized using our approach with linear interpolation between the two original views, depicted left- and rightmost.

This allows the formulation of bias-free pixel-wise objective functions in an analysis-by-synthesis framework that significantly outperform the state of the art. Further, IBR and hand tracking results are eventually used to explicitly generalize the underlying training body for the workflow tracking (Section 5.2.2.2 and 5.2.2.3).

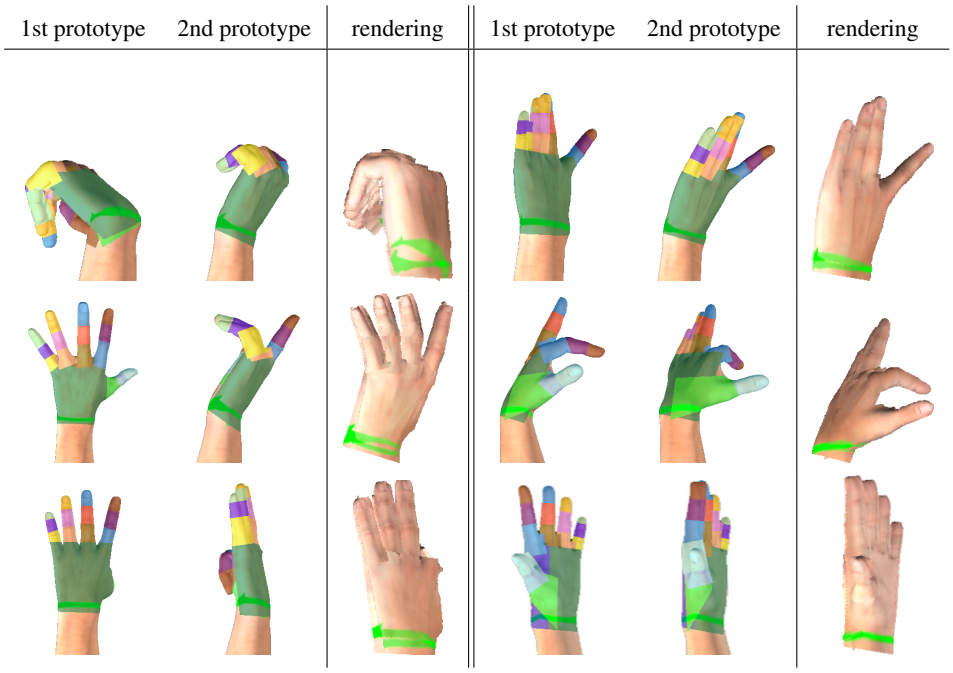


Figure 5.1: Examples using synthetic prototypes: Left column shows results for partly unobservable areas, right column for fully observable prototypes.

Since our method does not require skin-color segmentation, edge extraction, or any other preparatory feature extraction but operates directly on the pixel intensities, we gain additional robustness. In fact, we can show that our method leads to far less local optima than edge and skin-color based methods and is very robust towards blur, skin-colored background and even skin-colored occlusions. In practice, images often exhibit motion blur, due to the high movement speed of the human hand. We would like to point out that our method still produces stable results even in the combined presence of strong blurring, cluttered skin-colored background and skin-colored occlusions, whereas edge- and silhouette-based methods both fail.

Compared to related methods, our approach has the benefit of being computationally extremely lightweight and requiring only a coarse model fit and object segmentation. Although the prediction quality improves with shorter distance of the prototypes to the target pose, we will demonstrate that our approach can handle substantial differences in the input views. The method is not limited to hands but is in principle applicable to all articulated low-textured objects. Summarized, it is based on a kinematic skeleton with adjacent planar billboards, whose distinct textures are being morphed according to the view change.

Figure 5.2: Interpolation between observed hand postures: The leftmost and rightmost images show the frames used to extract the prototypes. The images in between show interpolated frames using our method.

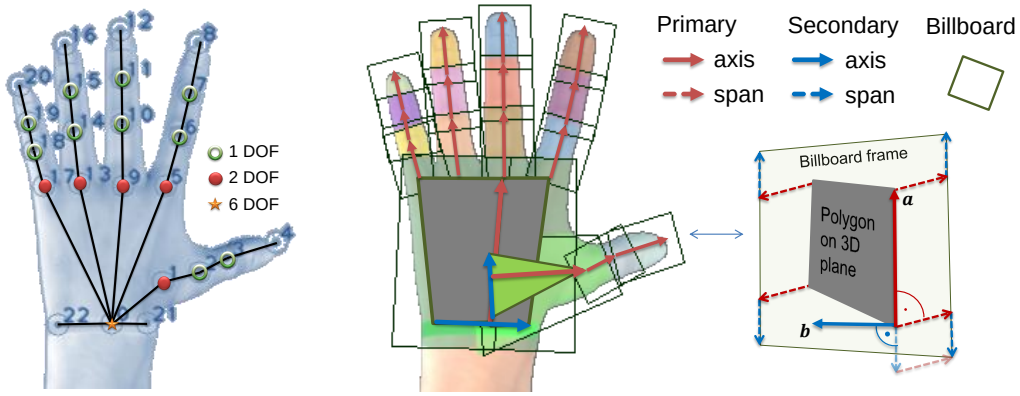


Figure 5.3: Schematic view of the kinematic hand model (**left**), the 2.5D billboards associated with this model (**middle**), and a single 2.5D billboard (**right**).

We distinguish between three main effects that affect the appearance of a kinematic object: segment-wise rigid transformation, elastic deformation, and shading change due to changed relative lighting. The rigid transformation which has the strongest impact on the appearance is carried out through positioning and deforming the billboards along the articulated model to match the target pose. Figure 5.3 (left) illustrates the kinematic model used to pose the hand. The rigid appearance change of each segment is thereby approximated using a proposed extension to billboard rendering, where billboards are transformed through the kinematic model (middle and right). To account for elastic effects and to compensate for model alignment errors, we propose an efficient axis-aligned warping method between the pre-sampled views. This set of labeled views is called the prototypes $\mathcal{P}$. As we assume relatively low texture, simple blending is then sufficient to cope with the remaining shading differences between two not too

distant prototypes. See [213] for a comment on why this is a sufficient interpolation in this case.

5.1.1 2.5D Billboards

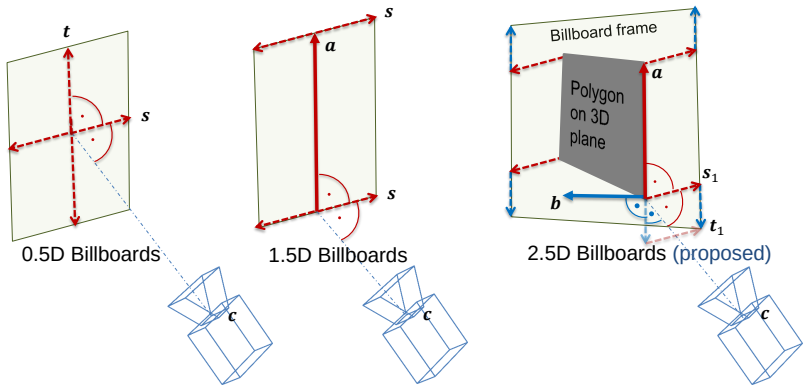


Figure 5.4: Billboard types in analogy to our proposed 2.5D billboards. While 0.5D can only change scale uniformly, 1.5D billboards extend this with a 3D axis that allows to model certain viewpoint changes at the cost of a possible collapse to a single line. Our proposed 2.5D billboards solve this through a second, perpendicular axis that spans a 3D polygon, surrounded by a 'billboard-frame' always aligned with the image plane.

As mentioned above, the hand is modeled as a set of billboards connected through a kinematic model. Billboards can faithfully simulate rotation-symmetric objects as long as the viewpoint is not changing substantially other than around the symmetry axis. This holds for cylindrical objects like finger segments but also for objects, where rotational appearance change is not directly apparent, *e.g.,* trees. Figure 5.4 illustrates the relation to conventional billboards.

Similar to [203], we are aligning each billboard with the bone vector $\vec{a}$ of the kinematic model. The billboard is then spanned using a vector $\vec{s}_i = \alpha_i(\vec{a} \times [\vec{j}_i - \vec{c}])$ perpendicular to $\vec{a}$ and parallel to the image plane, *i.e.,* perpendicular to the vector from camera center $\vec{c}$ to the adjacent joint $\vec{j}_i$. The scalar α_i is a normalization resp. scaling factor.

This billboard definition suffers from a singularity problem: If the camera view direction is aligned with $\vec{a}$, the billboard collapses to a single line. Additionally, for instance the palm of the hand exhibits a strong change in shape with change of viewpoint. Therefore, we propose an extension to billboards, which effectively solves both issues and is capable of reproducing

non rotation-symmetric objects more faithfully from arbitrary viewpoints. Compare the right illustration of Figure 5.3.

Our approach can be interpreted as a combination of a 3D planar polygonal patch and a surrounding billboard "frame" always aligned with the image plane. Therefore, we call this extension 2.5D billboards. It allows viewpoint dependent minimum and maximum shapes that can be used to describe rotational asymmetry and thus a larger class of convex 3D objects.

Formally, we achieve this by introducing a secondary axis $\vec{b}$ and a secondary perpendicular span vector $\vec{t}_i = \beta_i(\vec{b} \times [\vec{j}_i - \vec{c}])$ with the normalization factor β_i. The two axes then span a plane in 3D space. Let $\{\vec{x}_i\}, |\{\vec{x}_i\}| \geq 3$ be the set of vertices of a convex polygon in this plane. In fact, the convexity-constraint is too strict but simplifies the construction process with respect to avoiding self-intersection. Please note that the method described in this work is hereby effectively not limited to convex objects: Modeling of concave objects is achieved through the subsequent pixel-precise morphing step, while the billboard projection operates on the convex hull of the object. We establish a "billboard frame" around this polygon by adding axis aligned span vectors to obtain the billboard vertices

$$\vec{y}_i = \vec{x}_i + \vec{s}_i + \vec{t}_i. \tag{5.1}$$

Although the orientations of the billboard vectors are entirely determined, their lengths given by α_i and β_i are free parameters for each vertex $\vec{x}_i$. To avoid self-intersection of the resulting billboard polygon we do enforce the following three constraints when determining α_i and β_i:

- $\{\vec{y}_i\}$ represents a convex polygon.
- The billboard frame is on the outside of $\{\vec{x}_i\}$. Formally, let $\vec{m}$ be the centroid of $\{\vec{x}_i\}$, then the following must hold: $(\vec{y}_i - \vec{m})(\vec{x}_i - \vec{m}) > 1$.
- The order is maintained, *i.e.,* if $\vec{x}_i$ is the clockwise neighbor of $\vec{x}_{i+1}$ the same holds for $\vec{y}_i$ and $\vec{y}_{i+1}$.

Since inner and outer polygons are both convex, enforcing these constraints for an arbitrary viewpoint is sufficient for the constraints to hold for all viewpoints.

5.1.1.1 Constructing 2.5D billboards from projections

For constructing the 2.5D billboards there remains the question of the exact shape of the inner polygon as well as the outer billboard frame. While it would be possible to solve this through minimizing the reprojection error, there exists a simpler way.

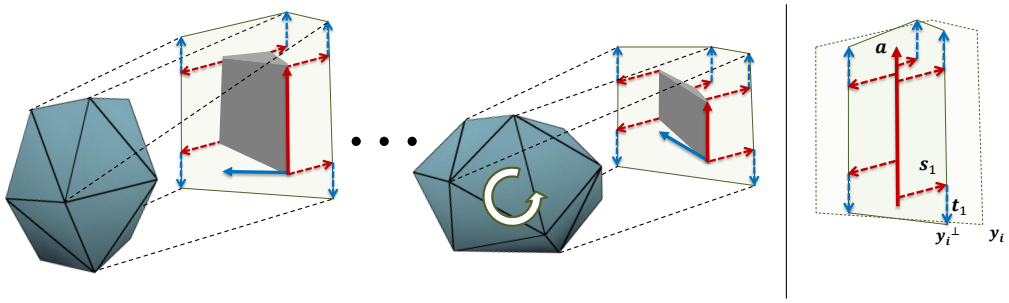


Figure 5.5: Illustration of 2.5D billboard projections of 3D objects at various orientations (**left**). Not more than one of the two axes of the inner polygon will vanish at any orientation (**right**).

The construction is conducted in several steps: First, we determine the vertices $\vec{y}_i$ of the billboard frame. This is done by examining the orientation that maximizes the projected silhouette of the object (which is trivial for finger segments). Then, we define $\vec{y}_i$ as the polygonal approximation of the convex hull of the projected silhouette.

In subsequent steps, we determine the billboard vectors $\vec{s}_i$ and $\vec{t}_i$, separately. We exploit the fact that the orientation that maximizes the projected object silhouette implies that also the inner polygon has to be in the orientation that maximizes its projected silhouette, *i.e.,* must be aligned to the image plane. Therefore, when rotating the object by 90° around one of the axes, the other axis vanishes in the projection of the inner polygon. Thus, when rotating around $\vec{a}$ (compare right of Figure 5.5), the inner polygon projects onto a line in the direction of $\vec{a}$, *i.e.,* has no extent in the direction of $\vec{b}$. We then define the intersection of the line $\vec{y}_i + v\vec{b}, v \in \mathbb{R}$ with the rotated object's projection as $\vec{y}_i^{\perp}$ and determine the corresponding billboard vector as

$$\vec{s}_i = \vec{y}_i - \vec{y}_i^{\perp}. \tag{5.2}$$

By repeating this procedure for the remaining axis $\vec{b}$ analogously, we can reconstruct the inner polygon using the determined billboard vectors $\vec{s}_i$ and $\vec{t}_i$.

Figure 5.3 shows the billboards that are used to model the hand. Applied to finger segments there is no obvious choice for a secondary axis. To solve this, we use the segment's bone as primary axis and reuse the primary span vector as secondary axis. If the bone aligns with the camera's viewing direction, this span vector would become zero and the billboard would collapse to a single point. In this case we substitute the secondary axis with the primary span vector of the preceding segment in the kinematic chain until we have found a non-zero span-vector.

5.1.1.2 Capturing prototype appearance

Each prototype consists of the set of generating parameters (position and joint angles) and the corresponding appearance information. To sample this appearance information from an image with given generating parameters, we assume that a rough segmentation is available to distinguish the hand from the background. Typically, this is achieved through skin-color segmentation but as this may happen in an offline process, more sophisticated segmentation methods can be applied, as well.

To identify the association between pixels and billboards we first articulate our kinematic model according to the given parameters. We then project the resulting model onto the image and label the segments accordingly. Pixels that are located within two billboard areas, *e.g.,* within overlaps along each finger, are copied into both billboard textures.

To avoid sampling neighboring fingers into the same billboard texture we prune each texture line-wise by only keeping the biggest connected segment. If there is at least a pixel gap between neighboring fingers this solves the issue. We have experimented with grab-cut segmentation to solve unintentional co-sampling for the case of unseparated occluding and occluded parts. The results however did not improve upon always co-sampling as the deteriorative effects are alleviated by the texture morphing phase.

Also, the prototype views often contain unobservable parts, *e.g.,* fingers occluded by the palm. We nevertheless sample the area where the occluded segment would be located. Although this leads to typically rectangular artifacts (since our billboards are mostly rectangular), it assures a smooth transition away from this prototype which is important for the derivability of the objective function. Our evaluation shows that this approach leads to a dominantly monotonic pixel-wise objective function despite the sampling errors.

5.1.2 Axis-aligned morphing between prototypes

The biggest influence on the rendered appearance, the segment-wise rigid transformation, is handled through the billboard articulation and accounts for rotational alignment of prototype and target billboard axes. As we show in the following evaluation, only relying on the 2.5D billboard transformation is already providing useful results.

However, to allow the exploitation of shading cues, a more precise approximation is required. Morphing the billboard texture, *i.e.,* blending and simultaneous warping could improve

the results, though the incorporated warping step is generally too costly to be used as an objective function. Due to the preceding rigid transformation, we can formulate an axis-aligned warping technique that produces satisfying results while being computationally comparable to a non-uniform scale.

We are exploiting two observations about the problem: Firstly, the fact that the visual effects due to warping within the object's silhouette are negligible compared to areas that contain the boundary, due to the relatively low texture. Secondly, we utilize that the "bones" of the kinematic chain are always roughly aligned with the object boundary.

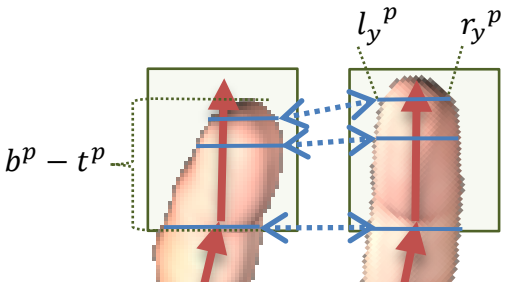


Figure 5.6: Illustration of the axis-aligned morphing scheme.

Figure 5.6 illustrates the method. When capturing the prototypes, we store the billboard texture always aligned to the according principal axis. We thereby compute the left and right contour boundaries of the billboard texture. Left and right denotes the directions $+\vec{s}_i$ and $-\vec{s}_i$, *i.e.,* perpendicular to the kinematic bone. Additionally, we calculate the top and bottom $(+\vec{t}_i, -\vec{t}_i)$ boundary of the texture, stored as the coordinate components t^p and b^p in texture space, $\forall p \in \mathcal{P}$. In practice, we only calculate top and bottom on billboard textures representing the fingertips and assume zero (top) and the texture height (bottom) for all other segments.

For the resulting texture we take the weighted average for top $t^{res} = \sum_{p\in\mathcal{P}} \gamma^p t^p$ and bottom $b^{res} = \sum_{p\in\mathcal{P}} \gamma^p b^p$ and row-wise left boundary $l_y^{res} = \sum_{p\in\mathcal{P}} \gamma^p l_y^p$ and right boundary $r_y^{res} = \sum_{p\in\mathcal{P}} \gamma^p r_y^p$. The computation of the weights γ^p will be described in Section 5.1.3. The morphing can then be formulated as

$$\boldsymbol{T}^{res}(x,y) = \sum_{p\in\mathcal{P}} \gamma^p \boldsymbol{T}^p(x^p(y^p), y^p) \tag{5.3}$$

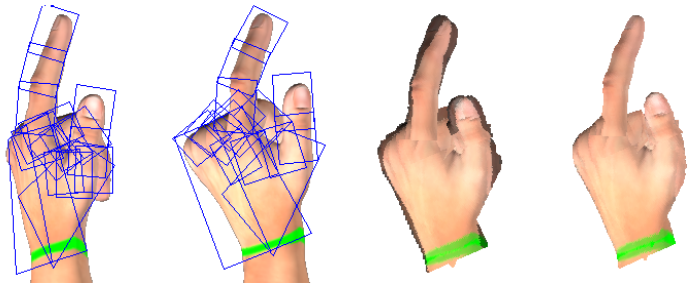
for all $y \in [t^{res}, b^{res}]$ with

$$y^p = (y - t^{res}) \frac{b^p - t^p}{b^{res} - t^{res}} + t^p \tag{5.4}$$

$$x^p(y^p) = (x - l^{res}_{y^p}) \frac{r^p_{y^p} - l^p_{y^p}}{r^{res}_{y^p} - l^{res}_{y^p}} + l^p_{y^p}, \tag{5.5}$$

where $\boldsymbol{T}(x,y)$ is a placeholder for billboard textures at pixel coordinate x,y with $\boldsymbol{T}^p(x,y)$ denoting the texture of prototype p and $\boldsymbol{T}^{res}(x,y)$ denoting the resulting texture.

The rearticulated and then pixel-wise warped prototypes are sufficiently similar so that the subsequent cross-fade is well approximating the fine-grained changes due to relative movement of the light source and fine-grained deformation. A formal explanation, why cross-fading is sufficient in this respect is pointed out in [213]: Simple cross-fading is indistinguishable from proper morphing for matching errors that are smaller than half the wavelength of the spatial frequency of the images. As the hands exhibit very low texture, this mostly resolves the remaining small-scale matching errors. Only small areas of the hand violate this assumption and exhibit a certain degree of ghosting, most noticeably the shirt-sleeve and the finger nail area. The warping prevents the occurrence of ghosting effects at the entire boundary, see



(a) **(b)** **(c)** **(d)**

Figure 5.7: Using prototype views **(a)** and **(b)** cross-fading the re-articulated models **(c)** leads to visible artifacts due to model-alignment errors and elastic deformation. This does not occur using our morphing technique **(d)**.

Figure 5.7. Particularly, since we do not dedicatedly treat unobservable image content, these boundary effects would be very evident.

As we do not perform any deformation to account for soft-tissue skinning at joints we do have visible boundaries between the single segments. A deformation method like [214] would allow a smooth transition, however, at the cost of a disproportionally increased computational

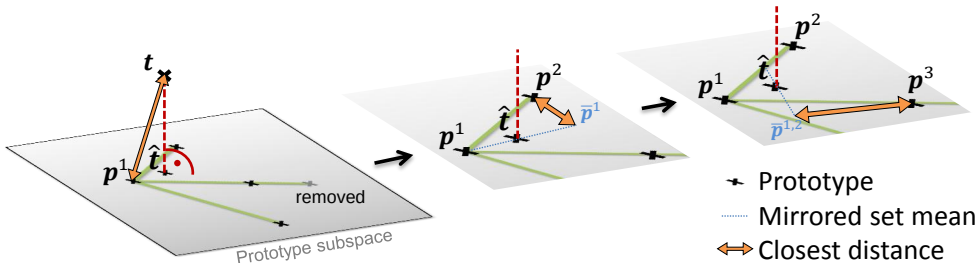


Figure 5.8: Illustration of the prototype subspace and the scheme used to calculate blending weights.

effort. Since the pixel-wise objective functions are not sensitive towards local texture discontinuities, this does not result in any loss of accuracy.

5.1.3 Determining blending weights

The definition of the blending weight γ^p that determines the weighting of prototype p in Equation 5.3 has a crucial influence. Simply defining γ^p reciprocal to the distance of the target parameter vector $\vec{t}$ (*i.e.,* the 26 hand pose/posture parameters) to the prototype's parameter vector $\vec{p}^p$ does not give appropriate results. The three arising problems are illustrated in the following examples using three prototypes with the (simplified for didactic reasons) parameter vectors $(0,0)^T$, $(10,0)^T$, and $(35,0)^T$:

Neighborhood: The synthesis of $\vec{t} = (5,0)^T$ would still incorporate the prototype $(35,0)^T$, though restraining to $(0,0)^T$ and $(10,0)^T$ produces a "cleaner" result.

Lateral position: The synthesis of $\vec{t} = (0,10)^T$ should only incorporate the prototype at $(0,0)^T$, as neither prototype contains information about the second component.

Prototype bias: Given an additional 4th prototype at $(10,10)^T$ and a synthesis target at $(15,5)^T$, the three closest prototypes would be $(0,0)^T$, $(10,0)^T$, and $(10,10)^T$ and thus all on the "left" of the target. This would lead to a significant bias in an objective function.

To solve this, we are using the following procedure to define the blending weights. First, we are sorting the prototypes in $\mathcal{P}$ such that $\vec{p}^1$ is the nearest neighbor to $\vec{t}$ in parameter space, $\vec{p}^2$ the second nearest and so on. We then choose the closest prototype $\vec{p}^1$ as support vector for a subspace, see left of Figure 5.8. To address the neighborhood problem, we prune linearly

dependent prototypes in the set: Whenever a span vector $\vec{p}^i - \vec{p}^1$ is representable using a linear combination of any span vector using a closer prototype $\langle \vec{p}^j - \vec{p}^1 \rangle, 2 \leq j < i$, it is removed from the set of prototypes until we have k linearly independent prototypes $\vec{p}^1$ to $\vec{p}^k$ in $\mathcal{P}_k$. We then collect these span vectors for the prototype subspace in $\mathbf{P} = \left(\vec{p}^2 - \vec{p}^1 \mid ... \mid \vec{p}^k - \vec{p}^1\right)$. The orthogonal projection $\hat{\vec{t}}$ of $\vec{t}$ into the subspace $\mathbf{P}$ then satisfies

$$\mathbf{P}^T(\vec{t} - \hat{\vec{t}}) = \vec{0}. \tag{5.6}$$

Since we want $\hat{\vec{t}}$ to be included in the subspace, we substitute $\mathbf{P}\bar{\vec{t}} + \vec{p}^1 = \hat{\vec{t}}$ and reorder to receive

$$\mathbf{P}^T(\vec{t} - \vec{p}^1) = \mathbf{P}^T\mathbf{P}\bar{\vec{t}}. \tag{5.7}$$

As the inverse of $\mathbf{P}^T\mathbf{P}$ exists, we can solve this as

$$\hat{\vec{t}} = \mathbf{P}(\mathbf{P}^T\mathbf{P})^{-1}\mathbf{P}^T(\vec{t} - \vec{p}^1) + \vec{p}^1. \tag{5.8}$$

Through measuring distances within the prototype subspace, we then effectively address the issues regarding lateral positions. However, using all remaining prototypes would increase the computation time while having little impact on the result. We therefore select a maximally bias-free subset $\mathcal{P}^{bf}$ of prototypes. Starting with the set $\mathcal{P}_1^{bf} = \{p^1\}$, we mirror the mean of all prototypes currently within the set $\vec{m}_i = \frac{1}{n} \sum_{p \in \mathcal{P}_i^{bf}} \vec{p}^p$ through $\hat{\vec{t}}$ and add the closest of the remaining prototypes to this set (compare Figure 5.8):

$$\mathcal{P}_{i+1}^{bf} = \mathcal{P}_i^{bf} \cup \underset{p \in \mathcal{P}, p \notin \mathcal{P}_i^{bf}}{\operatorname{argmin}} \|2\hat{\vec{t}} - \vec{m}_i - \vec{p}^p\|_2. \tag{5.9}$$

The blending weight γ^p of the p-th prototype is then defined to be

$$\gamma^p \propto 1/\|\vec{p}^p - \hat{\vec{t}}\|_2 \tag{5.10}$$

and normalized to satisfy $\sum_p \gamma^p = 1$ with $p \in \mathcal{P}^{bf}$. In our experiments we found that developing $\mathcal{P}^{bf}$ to a size of up to 4 already leads to good results.

5.2 Content adaptive hand tracking

From a technical perspective, the state of the art in hand tracking can be coarsely categorized into frame-to-frame and tracking-by-detection approaches. Both approaches aim to cope with

different but specific challenges of hand tracking. The first is based on the assumption that the previous trajectory of the hand movement can be exploited to acquire an accurate estimate of the hand configuration in the current frame. However, due to the ability of the hand to achieve high movement speeds and accelerations, this assumption is often violated and quick, unintentional movements will likely cause a tracking loss. Additionally, these approaches do require an entirely different strategy for initializing the tracker.

The second dominant approach is tracking-by-detection using an offline trained classifier. Typically, this classifier is based on nearest neighbor considerations, so this approach is often called database querying. During tracking the resulting classifier or template matcher is then applied to single images and does not (or only loosely) rely on the motion history.

The tracking-by-detection approach has several beneficial properties. Initialization of the tracking is straightforward by choosing the best hypothesis without prior knowledge. This strongly alleviates the requirements on a high input frame rate due to the possibility of quick reinitializations. Since the hand disappears often in the course of a workflow, we are crucially reliant on a quick reinitialization scheme.

Also in related work (*e.g.,* [191]), the two approaches are often combined. After generating a single or multiple hypotheses using an initialization database, a refinement step is performed that resembles the approach of frame-to-frame tracking. Here, we are combining the two approaches in an essentially different way. We use a database that is sufficiently large to be usable not only for initialization but for continuous tracking, also. In our experiments, we use a database with 4.5 million entries, which allows a comparatively dense sampling of the configuration space. Despite the large size of the database, the approach runs at interactive frame rates, while supporting to add, remove, and replace entries during run-time. This is possible through the organization of the database that facilitates generating locally optimal search trees for every query and therefore very fast beam-search runs. This leads to the major advantage that all successfully tracked frames can be used to incrementally adapt the underlying database to the observation.

We start with a database filled with synthetic hand templates. When a hand posture is successfully recognized using our database, we refine the match using nonlinear optimization of the pixel-wise objective function, detailed in Section 5.2.2.1. Figure 5.9 shows examples of recognized postures using our database. In case of finding a well-defined optimum, a new appearance template is generated from the according frame that replaces the synthetic template within the database. To vastly increase the convergence speed, we not only replace the single

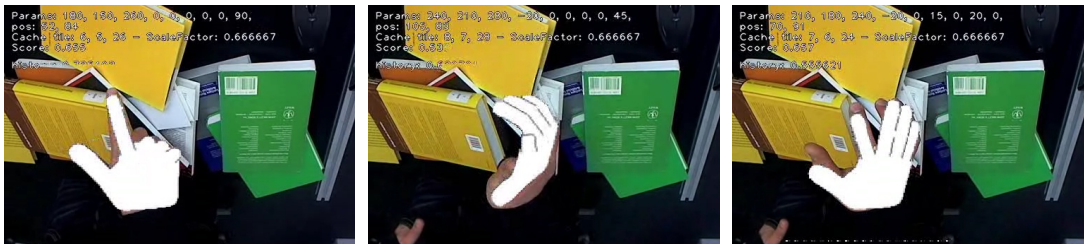


Figure 5.9: Sample tracking results on cluttered background.

nearest neighbor template, but also propagate the adapted appearance template to entries in the immediate vicinity using our image-based appearance model and interpolation between multiple entries.

The resulting tracking system retains most of the benefits from database indexing while still achieving frame rates between 10-15 frames per second on a single 2.8 GHz core and up to 30 frames with multi-threading on a quad core machine. Although the refinement scheme does not run in real-time, it may be outsourced to an asynchronous and even remotely executed thread. Since the eventual replacement in the database is computationally lightweight, the tracker can still be executed in real-time.

In the next subsections, we will describe the various aspects of this approach. We begin with explaining the structure and the initial content of the descriptor database. After that, we show how we combine this database with the adaptation scheme using the image-based appearance model.

5.2.1 Extendible descriptor database

The high performance of the database queries are possible through locally optimal search-trees (Section 5.2.1.2). We will briefly discuss the parameter subspace, which is used to initially create the synthetic hand views, the template construction, and the indexing method that contributes to the procedure.

5.2.1.1 Content and template construction

The high number of degrees of freedom (DoF) needed to express arbitrary hand configurations of up to 30 DoF for an anatomically accurate model exponentially increases the demand on the database size. However, as most of the mathematically possible hand configurations lead

to unnatural, unusual, or at least rarely seen hand postures, early tracking systems were using subspaces of about 8 to 15 dimensions (thereof 3 translational and 3 rotational DoF for the external pose) [197, 215].

Nevertheless, this parameter space is still very challenging, especially within the small time budget, owing to interactive frame rates. Additionally, any objective function for hand model fitting will be locally very non-linear due to several effects like self-occlusion. Hence, samples need to be collected at a relatively high rate among the parameter space. To synthesize views for our database, we are using the same kinematic model with 26 DoF that was introduced in Section 5.1.1, though only a subset is used to initially fill the database. There are methods that perform this reduction using PCA on data captured with a data glove [195]. Here, we explicitly attribute one flexion parameter and one abduction/adduction parameter to the thumb, the index finger and the combination of middle, ring, and small finger. The constrained model is still able to perform most of the natural poses, including an opposable thumb. Together with the three rotational DoFs of the wrist, this leads to a 9 DoF subspace that is stored densely sampled in the database. Since, the three translational DoFs of the wrist are handled by the template matching method, the effective recognizable DoFs of the tracking system is 12 DoFs. Please note that it is not a requirement that the parameter entries in the database must form a linear subspace. In fact, during the adaptation process, the database is extended with distinct observed poses.

The purely synthetic model is rendered as silhouette with boundary edges, see Figure 5.10. Although shading is a valuable cue, particularly to resolve ambiguities in the hand projection, it could only be exploited if lighting conditions are met by the synthetic rendering. As this would require known and static lighting, this would substantially reduce the universality and is therefore delayed until after the database adaptation takes place.

As discussed in Section 2.5.1, we use dominant orientation templates (DOT) [9] as appearance descriptors. Gradient orientation proves to be a good descriptor of hand appearance. Even if an edge between touching fingers is too weak to be recognized as such, the general gradient orientation is likely to be perpendicular to that edge. Furthermore, as the form of a finger segment is approximately cylindrical, the image gradients due to shading are mostly aligned and consistent with the boundary edge gradients. Misleading contrasts due to drop shadows only affect the matching scores in proportion to the affected area, as do occlusions, while brightness and contrast variations are implicitly normalized.

Unfortunately, the descriptor is neither scale nor rotation invariant. For translation along the z-axis, *i.e.,* scaling, we chose to not include database samples for that but rather perform

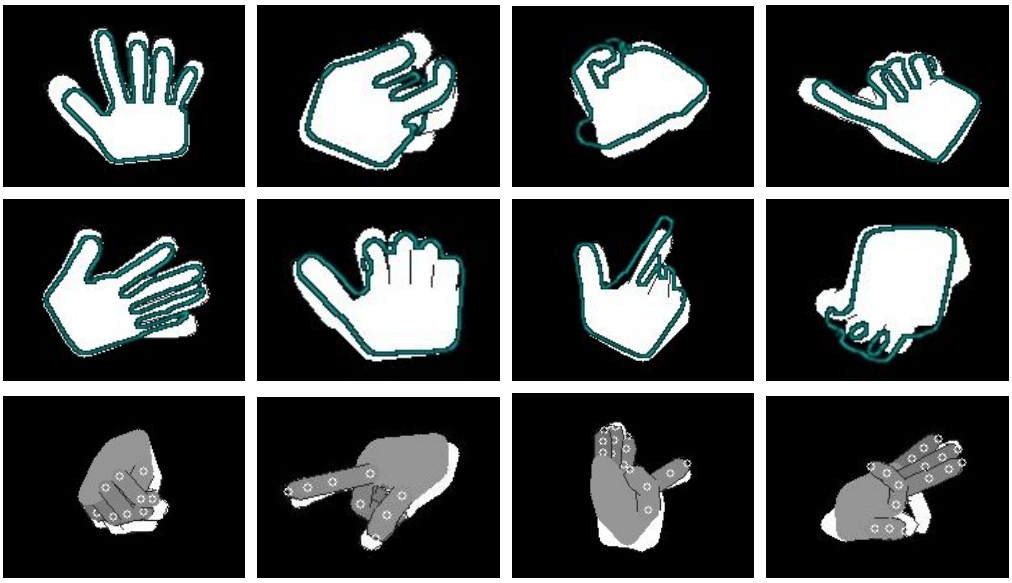


Figure 5.10: Results on the synthetic image sequences: Tracking result visualized as colored contour-overlay. Top row: dorsal views and only dorsal samples in the database (dorsal/dorsal), middle: dorsal views with full database (dorsal/full), bottom: unconstrained poses with full database (full/full).

iterated database searches with scaled query images. The reason is that changing the sample size leads either to cropping or aliasing effects: Larger scales reduce the area represented by the template to a small part of the hand; while at smaller scales the distinction between close-by edges is often lost, reducing the accuracy of the descriptor. Since the DOT template descriptor samples the observation at an effectively seven times lower spatial frequency, particularly the effects of antialiasing are mitigated through resampling the observation, directly.

Since the appearance-change is higher for in-plane rotation (our z-axis) than for the other two axes, we sample the z-axis at every 10° (36 templates), the x- and y-axis at 15° (24 templates). As we will show, this is sufficiently dense to ensure continuous detection of a smoothly moving target. For each external pose, we sample 216 different postures. In total this leads to $36 * 24 * 24 * 216 = 4,478,976$ DOT samples with their respective generating parameters in our database. In the next subsection we will show how we efficiently perform real-time beam-searches within this database.

5.2.1.2 Local search-tree generation

The tracking performance relies on local beam searches in direction of the currently estimated external parameters. The basic structure of our search tree is in accordance to the kinematic chain of the hand. Firstly, this reflects the hierarchic structure of the hand as an articulated body and thus tends to lead to similar image projections. Secondly, this also reflects different angular velocities occurring at the joints: Typically the velocities at internal joints are much higher than those of external joints and are vastly exceeding what can be captured at typical video frame rate of 30 fps. This property makes trajectory-based predictions of the internal joint angles rather inaccurate. Thirdly, the further a joint is located within the kinematic chain, the less image content is affected by the according parameters. Thus, as projections get increasingly similar, it becomes increasingly harder to estimate the respective parameters from the observation. Matching errors become more likely in these parameters. To accommodate all of this, it is desirable to broaden the beam at more uncertain degrees of freedom. In fact, we decided to include all internal parameters in our search-tree to be able to recover from tracking errors. This does not impact performance too much as most search runs are already rejected at higher tree stages.

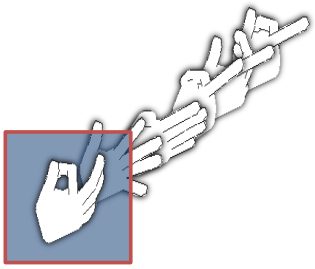


Figure 5.11: Illustration of a cache tile containing all hand postures of a certain extrinsic pose. Each cache tile can be loaded individually and removed from RAM when not used anymore.

In our experiments we use about 4.5 million samples with approximately 500 bytes per sample, but much higher values are possible. In order to reduce memory consumption and to cope with database sizes that do not fit in memory entirely, we incorporate a caching scheme, reducing the actually needed memory for loading parts of the database to about 6 MB of RAM. The whole database is stored in selectively loadable cache tiles, see Figure 5.11. Each tile contains all samples for a fixed set of external parameters and is loaded and released on demand by the tracking system.

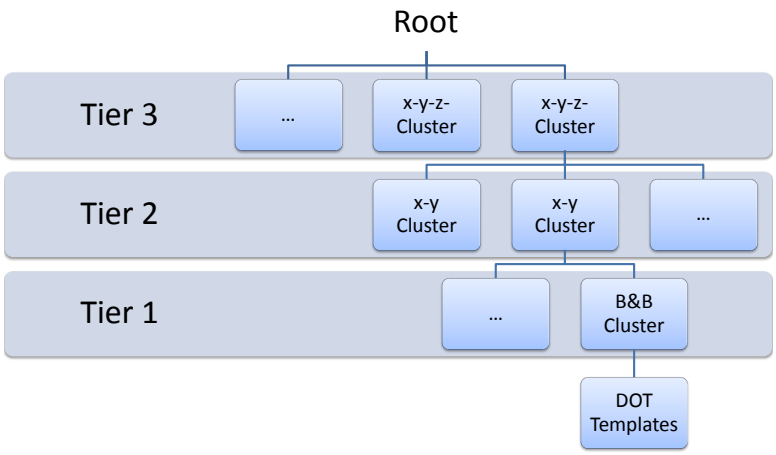


Figure 5.12: Illustration of the different tiers of the search tree: At every node an upper bound for the matching score can be quickly determined. Thus only branches that are able to exceed the score of the best hypothesis have to be examined.

We use a three-tiered search tree, see Figure 5.12. In the following, we list and afterwards describe the three tiers in detail:

- Tier 1 are the branch-and-bound clusters [179], as proposed and adapted by [9] for usage with DOT.
- Tier 2 is formed by precomputed local search-clusters containing the tier 1 clusters. The cluster comprises all samples from neighboring cache tiles for out-of-plane rotation, *i.e.,* around x- and y-axis.
- Tier 3 allows very fast reclustering of the precomputed local search-clusters. This is achieved through exploiting associative invariants among neighboring tier 2 clusters.

Tier 1: We use the branch-and-bound technique [179] as proposed and adapted by [9] for usage with DOT in tier 1. For a comprehensive understanding we will briefly sketch how the clustering works. A greedy algorithm is used to establish the clusters. The templates themselves are binary strings where each bit is associated with a gradient in a certain position and orientation. In the first step, the DOT template with the highest amount of dominant gradients (*i.e.,* the highest 1-bit count) that is not yet assigned to a cluster is used to start a new cluster. From the remaining unassigned templates, the one is added to the cluster that adds

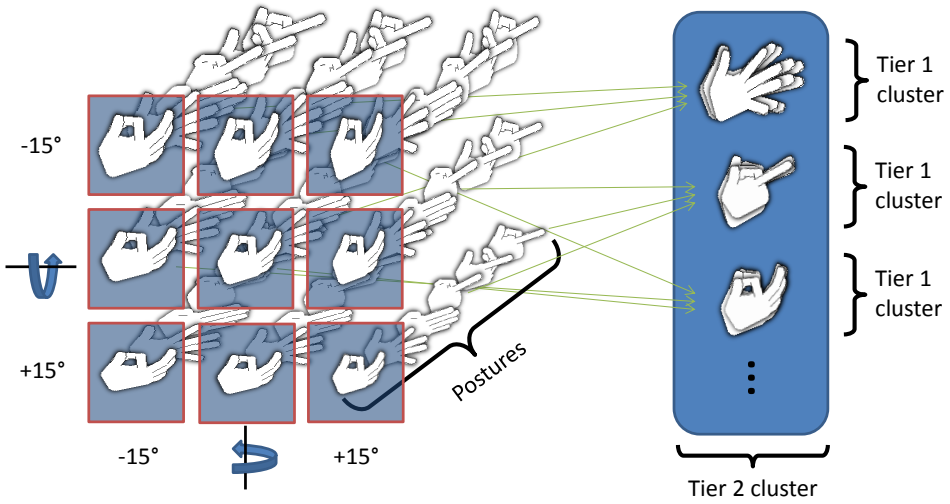


Figure 5.13: Illustration of the precomputed tier 2 clusters: The samples of 3×3 cache tiles seen on the left containing all $\pm 15^\circ$ out-of-plane rotations are clustered in tier 2.

the least additional dominant gradients to the template cluster. Adding, hereby, corresponds to bitwise OR-ing of the cluster-template and the new template. This is repeated until the cluster has reached a certain size, whereupon a new cluster is created. The whole algorithm iterates until no unassigned templates are left. For a more detailed pseudo-code representation, see Algorithm 2. Since clustering is $O(n^2)$ and becomes computationally heavy with large n, we precompute parts of it. The costly part is not the construction of the upper-bound template (OR-ing of short binary strings) but the selection of sufficiently similar templates. We will exploit this fact in several places within our method.

Tier 2: Tier 2 clusters out-of-plane rotations of the hand as a whole. We sample out-of-plane rotations at 15° with each set of according views stored in a separate cache tile. Each local tier 2 cluster contains the direct neighborhood in x- and y- direction (*i.e.,* rotation around the x- and y- axes). So, in total it comprises 9 cache tiles, see Figure 5.13 for illustration. Since each cache tile contains 216 templates in our implementation, each tier 2 cluster contains 1944 templates.

We precompute and store all tier 2 clusters. Since the computationally heavy part of the clustering process is finding the best-fitting templates for each cluster, we only store the associations by index and do not explicitly include the templates.

Tier 3: In tier 3 we exploit the underlying symmetry of the clustering with respect to rotation

Algorithm 2 Branch-and-bound clustering adapted from [9].

$\mathbf{U} \leftarrow$ set of all templates not assigned to a cluster
popcnt$\boldsymbol{t} \leftarrow$ number of 1-bits in template $\boldsymbol{t}$
$\mathbf{c} \oplus \boldsymbol{t} \leftarrow$ bitwise OR-ing of $\mathbf{c}$ and $\boldsymbol{t}$
while $\mathbf{U} \neq \varnothing$ **do**
  $\hat{\boldsymbol{t}} \leftarrow \underset{\boldsymbol{t} \in \mathbf{U}}{\operatorname{argmax}} \operatorname{bitcnt}(\boldsymbol{t})$
  $\mathbf{U} \leftarrow \mathbf{U} \setminus \hat{\boldsymbol{t}}$
  Create new cluster template $\mathbf{c} \leftarrow \hat{\boldsymbol{t}}$
  $s \leftarrow 1$
  while $s < maxSize, \mathbf{U} \neq \varnothing$ **do**
    $\hat{\boldsymbol{t}} \leftarrow \underset{\boldsymbol{t} \in \mathbf{U}}{\operatorname{argmax}} \operatorname{bitcnt}(\boldsymbol{t} \oplus \mathbf{c}) - \operatorname{bitcnt}(\mathbf{c})$
    $\mathbf{U} \leftarrow \mathbf{U} \setminus \hat{\boldsymbol{t}}$
    $\mathbf{c} \leftarrow \mathbf{c} \oplus \hat{\boldsymbol{t}}$
    $s \leftarrow s + 1$
  end while
end while

around the optical axis: Despite minor perspective effects or camera distortion, we can approximate rotation around the optical axis (our z-axis) as pure 2D image rotation also if the hand is not in the image center. The result is a faithful approximation, since the size of the hand is typically small compared to its distance to the camera and minor distortion effects are easily compensated through the template descriptor.

This simplification allows us to calculate a prototypical clustering that holds for all tier 3 clusters. A tier 3 cluster contains the tier 2 cluster rotated by $-10°$, $0°$, and $+10°$ around the z-axis. So, in total it contains the 27 cache tiles covering $[-15°;15°] \times [-15°;15°] \times [-10°;10°]$ around the center tile at our sampling rate. Since the hand postures in each cache tile contain finger configurations, partly compensating this extrinsic pose variation, the similarity of views facilitates a good clustering. Figure 5.14 shows an illustration of the cluster content.

We exploit the fact that the similarity between templates is maintained, if both templates are 2D-rotated by the same amount. This means that the optimal template association for clustering within each triple of neighboring tier 2 clusters is 2D-rotation invariant. Actually, there is one additional minor effect that violates this invariance: Since we do not rotate the grid cells of the DOT descriptor, the optimal clustering may differ slightly due to sampling effects. However, the effect is negligible as affected templates lead to a very similar upper bound compared to

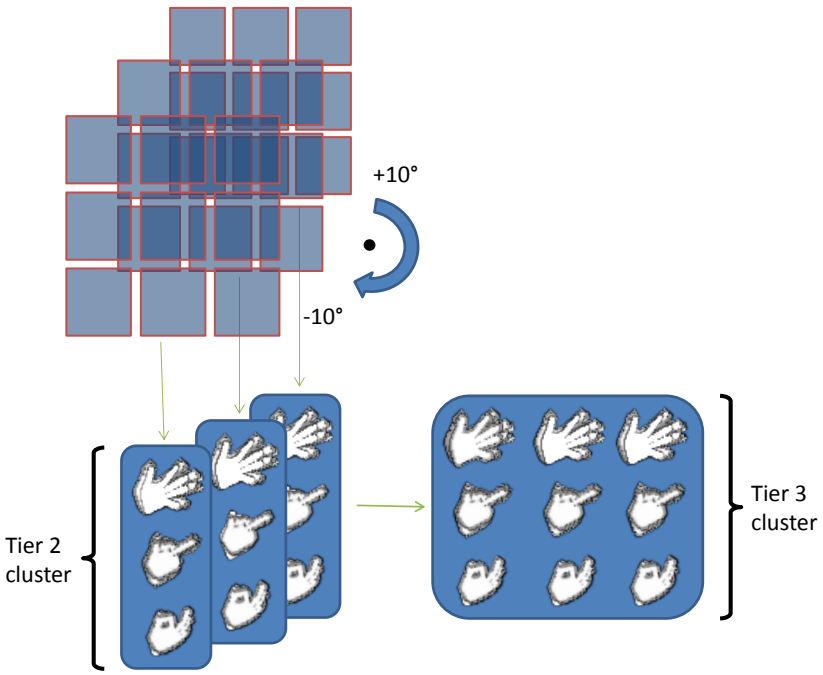


Figure 5.14: Illustration of tier 3 clusters generated on the fly: The tier 2 clusters containing the $\pm 10^{\circ}$ in-plane rotated samples are being grouped here.

the theoretical optimum.

For each tier 2 cluster we thus only have to compute one prototypical z-rotation neighborhood that can then be applied to all such triples. We again only store the associative information, *i.e.,* which templates of the tier 1 clusters were combined into the tier 2 cluster. We reapply the same association to create the tier 3 clusters for all other rotation values of the center tile. This reduces the amount of necessary precomputed clusters by the factor 36. This is a huge speed increase that allows exchanging parts of the database in real-time in order to adapt to observed images.

Only the clustering change of the single affected x-y-package has to be recomputed. Local search-trees are then implicitly available and updated for all rotated instances. Additionally, tier 3 leads to a natural choice of data parallelism for multi-threading, as the clusters within tier 3 can be processed by different threads. We therefore choose the cluster count to be a multiple of the number of (logical) cores.

In principle, the same approach is also viable for other geometric transforms, like finger joint rotations around the effective z-axis or even for out-of-plane rotations. However, (1)

these cases bear a significantly higher likelihood of self-occlusion and self-shadowing of the resulting hand appearance. (2) While we were able to align the symmetry of the underlying database entries under z-axis rotation with the database structure, this is not easily transferable to the small-scale symmetries. Since this requires rearranging the entries, it would break with the working principle of increasingly similar appearance down the search tree. The benefits would thus be countered by the significantly higher overhead.

5.2.2 Database tracking

In this section we show how we combine the appearance model and the possibility for fast local-beam searches into a content-adaptive tracking method. We start with explaining the procedure for database tracking and afterwards present the approach to refine a match and extend the database, accordingly.

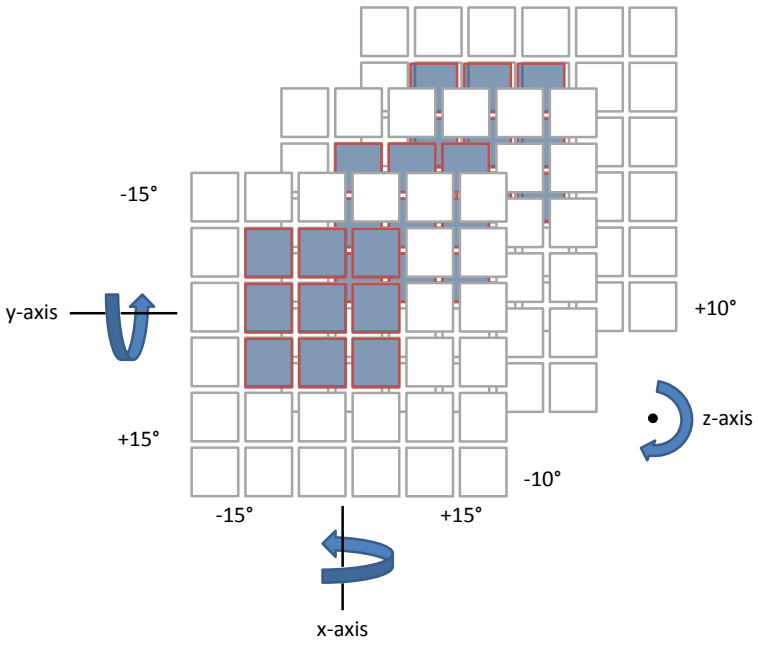


Figure 5.15: Illustration of the tracking support: Each cache tile is addressable through the according x-, y-, and z-Euler angles. At each iteration only the $3 \times 3 \times 3$ neighborhood centered at the current tracker state is searched.

As already stated, the tier 3 clusters contain templates of densely sampled intrinsic poses for an approximate external pose. For each frame in the sequence the $3 \times 3 \times 3$ cache tiles around

the current estimate of the external parameters are loaded, see Figure 5.15. The comprised DOT templates are then compared extensively at all 2D positions within each input frame. In reverse this means that the tracking system is only constrained by the three rotational external parameters and the z-axis translation, as explained below.

Given an estimate of the extrinsic hand orientation, the local search-tree is generated by loading the according tier 2 clusters and computing the tier 3 clusters through the prototypical clustering. At our sampling rate, the tracker compares the entire set of hand postures within $[-15°;15°] \times [-15°;15°] \times [-10°;10°]$ rotational tolerance of the extrinsic hand orientation.

Translation in $x-$ and $y-$ direction is handled by matching the templates at all possible positions, which is handled very efficiently through the DOT matching approach [9]. For further speed-up, we reject regions where no skin-color is found.

Translation along the z-axis, *i.e.,* scaling, is handled by consecutive runs with the query image being scaled accordingly (see the previous Section 5.2.1.1 for an explanation). We hereby repeat the scans on scaled instances of 90%, 100%, and 111% of the current estimate.

At our sampling rate, this results in the maximum viable velocities of 450° per second out-of-plane rotation and 300° per second in-plane rotation at a video frame rate of 30 frames per second. The angular velocities in the internal joints are hereby not limited, as each neighborhood contains a densely sampled set of internal parameters.

For every frame, we retain the $i = 1..k, k \leq 8$ matches with a score s_i above a certain threshold and a parameter vector $\vec{h}_i$. These matches are then combined by weighted averaging to give the parameter estimate $\vec{x}_t$ for the current frame:

$$\vec{x}_t = \alpha \sum_{i=1}^{k} \frac{\beta s_i}{\|\mathbf{W}(\vec{x}_{t-1} - \vec{h}_i)\|_1} \vec{h}_i \,, \tag{5.11}$$

where $\mathbf{W}$ is a parameter weighting matrix, α is a normalization factor, β is a relative weighting factor between the matching scores and scaling due to distance.

The matrix $\mathbf{W}$ is a diagonal matrix that weights the translational and rotational parameters inversely proportional to their position in the kinematic chain: Hand orientation is weighted with 1, the metacarpophalangeal (MCP) joint angles (flexion and abduction) with $1/2$, proximal interphalangeal (PIP) joint angles with $1/3$, and distal interphalangeal (DIP) joints angles with $1/4$. The weight coefficients for the translational parameters are normalized to match the scale of the rotational parameters. However, 2D translation is penalized stronger to avoid

incorporating spurious matches. The normalization factor α is then determined as

$$\alpha = \left(\sum_{i=1}^{k} \frac{s_i}{\|\mathbf{W}(\vec{x}_t - \vec{h}_i)\|_1} \right)^{-1}. \tag{5.12}$$

The value of α can also be interpreted as a confidence measure, as a small value reflects overall high matching scores and good parameter compliance.

5.2.2.1 Model refinement

For all frames that were successfully matched, as indicated through a small value of α in Equation 5.12, we perform an additional model refinement step using Particle Swarm Optimization (PSO) [198] and our proposed appearance model. PSO is an iterative optimization method that was proposed for usage with hand tracking by [201]. A general overview of further applications is found in [216, 217].

We propose an objective function based on pixel-wise differences to be minimized by PSO. The straightforward formulation would be:

$$f(\vec{p}) = \frac{1}{n} \sum_{x,y} |\boldsymbol{I}_t(x,y) - \boldsymbol{I}_{\vec{p}}(x,y)|, \tag{5.13}$$

where n is the number of pixels in the image and $\boldsymbol{I}_{\vec{p}}$ is the rendered hypothesis $\vec{p}$ using our proposed image-based appearance model.

Although the average per-pixel differences are well suited to evaluate the reconstruction accuracy of the proposed appearance model, we can do better in terms of an objective function for optimization purposes. When dealing with background that is similar to the pursued object, the average per-pixel differences will likely exhibit frequent local optima. We therefore limit the support area to the area covered by the current rendered hypothesis. To further reduce the amount of local optima, we reject areas, where the difference is too large to contain reliable information about the true global minimum:

$$\hat{f}(\vec{p}) = \sum_{x,y} \begin{cases} \min(\psi, |\boldsymbol{I}_t(x,y) - \boldsymbol{I}_{\vec{p}}(x,y)|) & \text{if } \boldsymbol{I}_{\vec{p}}(x,y) \neq \emptyset \\ \phi & \text{else,} \end{cases} \tag{5.14}$$

where $I_{\vec{p}}(x,y) \neq \emptyset$ denotes coordinates that are within the silhouette of our rendered hypothesis, ψ is the maximum acknowledged pixel value difference, and ϕ is a regularizer that adds a discount to the objective function in areas not covered by the rendered silhouette. The effect of ϕ is to influence the objective function towards favoring solutions with larger silhouette

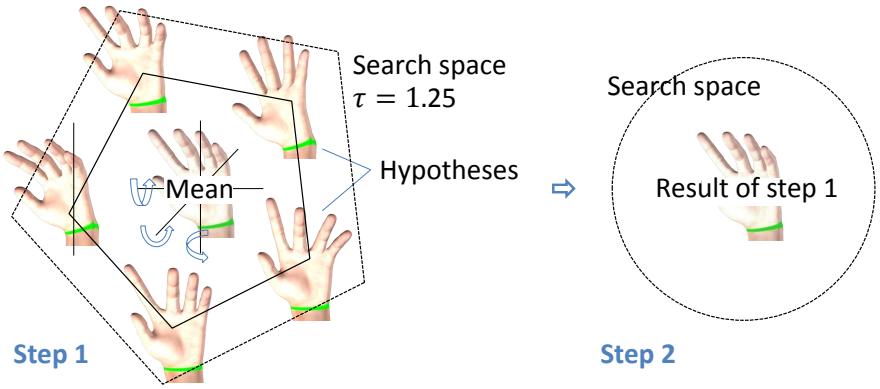


Figure 5.16: Illustration of the partitioned PSO solver: We first run the optimization on a smaller subspace spanned by the hypotheses, then refine the result using the full search space.

area. Our experiments have shown that without this discount, there is the potential of a 'trivial' optimum when the hand is projected onto a very small area. We have chosen $\psi = 50$ (with respect to intensity values in the range of 0-255) and $\phi = 0.8\psi$ for our experiments. We will show that the refined formulation leads to an increased robustness, especially on top of skin-colored background. In the following we will denote this as the *proposed* function and the average per-pixel difference (Equation 5.13) as *average*.

We slightly alter the optimization procedure as proposed by Kennedy and Eberhart [198]. We propose a partitioned approach that exploits the prior knowledge from the tracking system. Figure 5.16 illustrates the procedure.

We use the set of hypotheses $\vec{h}_i$, including the best hypothesis for the last frame $\vec{x}_{t-1}$. In contrast to the proposed random initialization of PSO, we instead initialize particles at every hypothesis $\vec{h}_i$ and at the weighted average, denoted as $\vec{x}_t^h$. We then define a low-dimensional search-space by spanning an affine linear subspace

$$\vec{x}_t^k = \vec{x}_t^h + \vec{p}^T \left[\vec{h}_1 - \vec{x}_t^h \mid \vec{h}_2 - \vec{x}_t^h \mid \ldots \right]. \tag{5.15}$$

We do not remove collinear span-vectors $\vec{h}_i - \vec{x}_t^h$ as PSO follows a randomized exploration scheme and is not dependent on operating on a vector base. The low-dimensional parameter vector $\vec{p} \in \mathrm{R}^k$ is therefore possibly possessing collinear components and k matches the number of hypotheses (including the last frame's best hypothesis).

We then further constrain the (convex) search space to $\|p\| < \tau$, where τ is a scale factor. With $\tau = 1$, the constrained search space would contain all hypotheses as outer corners of the

polyhedron. In our experiments, we have set $\tau = 1.25$. After a relatively small number of iterations within this low-dimensional search space, we switch to the full parameter space to refine the matches within a small perimeter around $\vec{x}_t^k$ to receive the estimated parameter vector $\vec{x}_t$ for the frame.

PSO has already been successfully applied to high-dimensional problems, even containing the joint parameter space for both hands [202]. However, these experiments have been conducted on less ambiguous, thus easier RGBD input material. We found that our partitioned approach gets less often stuck in local minima on monocular RGB observations.

5.2.2.2 Extending the database

Each database is not only labeled with the generating parameters but can also be associated with an according IBR appearance model. During tracking, the database is incrementally extended. For each confident parameter estimate, we replace the DOT descriptor of the closest database entry. Using the IBR model captured from that frame, we propagate the new information into the surrounding database entries. The rationale behind replacing instead of adding entries is to keep the database size and thus the query run-time constant. Figure 5.17 shows two examples of generalized renderings for matched frames.

Initially, there is the problem of deciding, when the tracker has successfully tracked a hand posture. Unfortunately, there is no reliable indicator for this. Here, we use an experimentally determined threshold value for the objective function, as there is no natural choice for deciding on the first frame. After the first matched frame, we use the reprojection error of all IBR models that are closest in parameter space as an additional criterion of validation.

Figure 5.18 shows interpolation examples between IBR models captured from different frames. Although the database only covers geometric (posture) parameters, our appearance model is also able to interpolate between different morphological and lighting properties. Figure 5.18(ab) shows interpolation between different morphological properties and Figure 5.18(bc) between different lighting scenarios. The appearance model is also able to interpolate between real and synthetic input material, which is shown in Figure 5.18(cd). In the future, these non-geometrical properties could be incorporated into the parameter space, explicitly. The synthetic rendering in this image was rendered using Poser [218].

To adapt the database, we load and rewrite the 27 cache tiles within the tier 3 cluster in the vicinity of $\vec{x}_t$. We thereby replace all entries within a certain radius around $\vec{x}_t$ with image-based

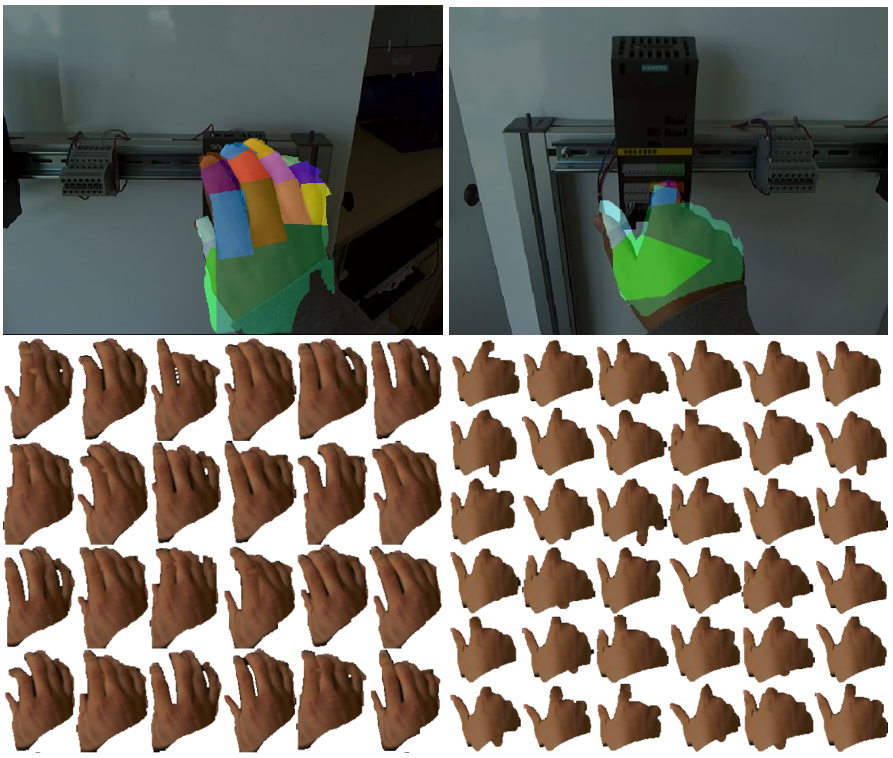


Figure 5.17: Two examples of generalized hand postures from a single tracked frame used to fill the database.

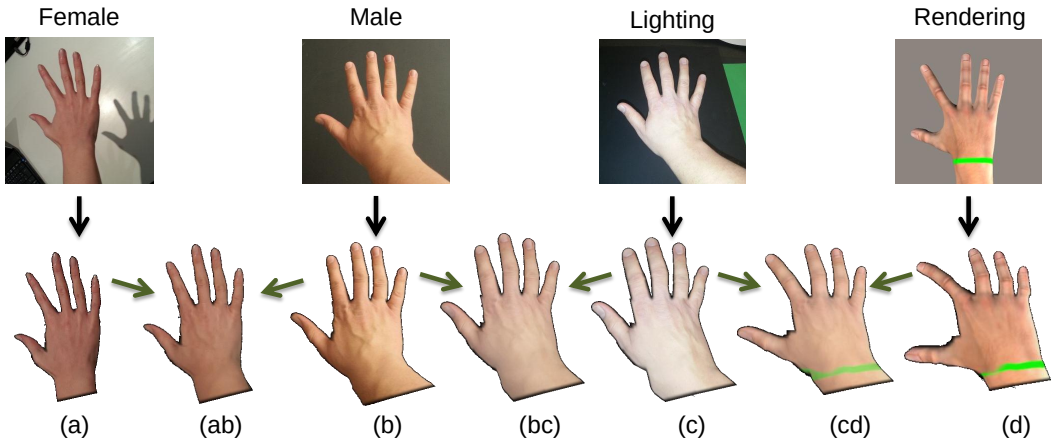


Figure 5.18: Synthesized hands using our image-based appearance model: The prototypes have been sampled from the four images in the top row. The bottom row shows reconstructions and interpolations between the different prototypes.

renderings. In our evaluation section, we will demonstrate how this affects the recognition of the user performance.

5.2.2.3 Incorporation into workflow tracking approach

In order to extend the workflow tracking model with the information gathered from hand tracking, we generalize the training sequence, directly. We hereby incorporate the adapted entries from the database into the spatiotemporal k-NN classifier, described in Section 4.2.

This is straightforward as both approaches are based on the DOT descriptor. In contrast to the approach used exclusively for hand tracking, we additionally need to incorporate the environment into each entry. We therefore retouch the input image and remove the hand using image inpainting [219]. We use the retouched image as background for the image-based renderings of the generalized hand postures and use the composited images as additional training material for the spatiotemporal classifiers, described in Chapter 4. The inpainting leads to certain smoothing artifacts. However, as the distortions are rather small-scale, this does not have a strong impact on the resulting DOT descriptors. Figure 5.19 shows an example of the inpainting result and the generalized training material.

In reverse, the spatiotemporal classifiers initialize the hand tracking during run-time, as the hand is likely to go through one of the already sampled postures in the course of a work step. Using hand parameter labels on the k-NN entries, this allows to bootstrap the hand tracker.

5.3 Evaluation

In the following subsections, we first examine the reproduction accuracy of the proposed image-based appearance model. Further, we analyze the properties of the pixel-wise objective function that can be formulated using this appearance model. We are able to show that our approach significantly outperforms the state of the art on monocular RGB sequences through demonstrating flaws in commonly used parts of the objective function. Afterwards, we investigate the initialization procedure using the database approach and the impact of the adaptation scheme, when applied to real workflow sequences.

5.3.1 Reproduction accuracy of the image-based appearance model

We propose to use our IBR-based appearance model in combination with a pixel-wise objective function, which makes its ability to accurately synthesize previously unseen poses an important

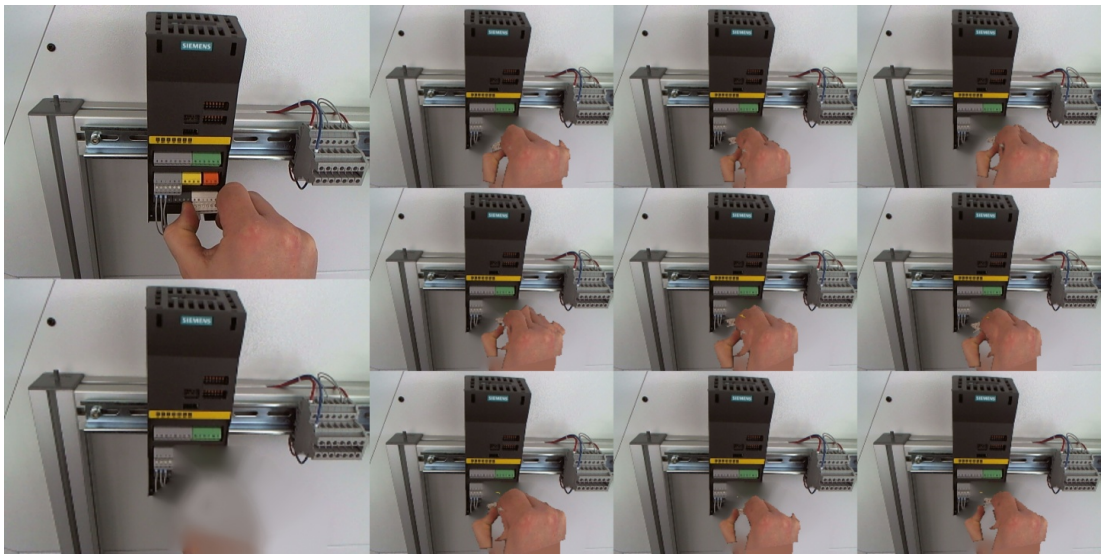


Figure 5.19: Example of the explicit generalization of a single frame: The upper left image shows the original frame. We remove the hand using image inpainting (lower left). The right side shows the generalized frames.

factor. To investigate its feasibility in this regard and to obtain quantitative measures, we use renderings generated with Poser [218] with ground truth available. We have augmented the Poser model with a green stripe around the wrist section to represent the shirt-sleeve.

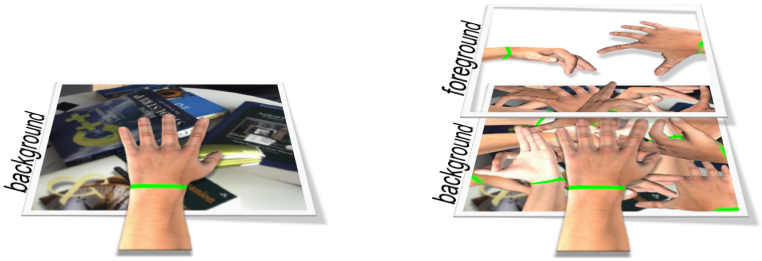


Figure 5.20: Experimental setups: Cluttered background for the *clutter* setup (**left**), background and occluding foreground containing hands for the *hands* setup (**right**).

For our experiments we have generated 2000 random hand poses including random finger articulations and random viewpoints except from impossible angles (*e.g.,* from within the arm joint, intersecting fingers) used as target views. We then generated two experimental setups, see Figure 5.20:

Clutter: The rendered Poser images were drawn on top of cluttered background.

Hands: The rendered Poser images were drawn on top of cluttered, skin-colored background showing hands and then additionally occluded by a foreground image showing hands.

Each target view was then rendered from viewpoints differing by $\pm 20°$, $\pm 40°$, and $\pm 60°$ degrees out-of-plane rotation. These views were used as prototype views $\vec{p}^{-}, \vec{p}^{+}$ to sample our model from. To measure the improvement due to morphing, we performed each experiment once with both prototypes and once only using the prototype $\vec{p}^{-}$ at $-20°$, $-40°$, resp. $-60°$. We did not enforce any further visibility constraints other than rejecting impossible poses. Particularly with higher viewpoint difference ($\pm 40°$ and $\pm 60°$), the prototype views exhibit substantial amounts of unobservable areas with respect to the object parts visible in the target view.

We first investigated the reconstruction error when interpolating between two prototype views with and without the axis aligned-morphing scheme. To this end, we used the clutter-setup, *i.e.,* we have rendered the target view onto the cluttered background using Poser to generate a synthetic observation image $\boldsymbol{I}^{obs}$ with known ground truth. Using our method, we have rendered the model for all parameters $\vec{p}_\alpha = (1-\alpha)\vec{p}^{-} + \alpha\vec{p}^{+}$ with $\alpha \in [0,1]$, denoted as $\boldsymbol{I}_\alpha^{render}$. Thus, for a perfect reconstruction $\boldsymbol{I}^{obs} = \boldsymbol{I}_{0.5}^{render}$ should hold. An illustration can be seen in Figure 5.21. To measure the image difference we use the average pixel-wise difference:

$$f_\alpha = \frac{1}{pixel\ count} \sum_{\forall x,y} |\boldsymbol{I}^{obs}(x,y) - \boldsymbol{I}_\alpha^{render}(x,y)|. \tag{5.16}$$

Figure 5.21: Difference images between (synthetic) observation and our rendering $|\boldsymbol{I}^{obs} - \boldsymbol{I}_\alpha^{render}|$ for several values of α. The center image shows $\alpha = 0.5$, where the ground truth optimum is located. The objective function for $0 \leq \alpha \leq 1$ was evaluated in 2000 randomized repetitions in both experimental setups (clutter setup shown here).

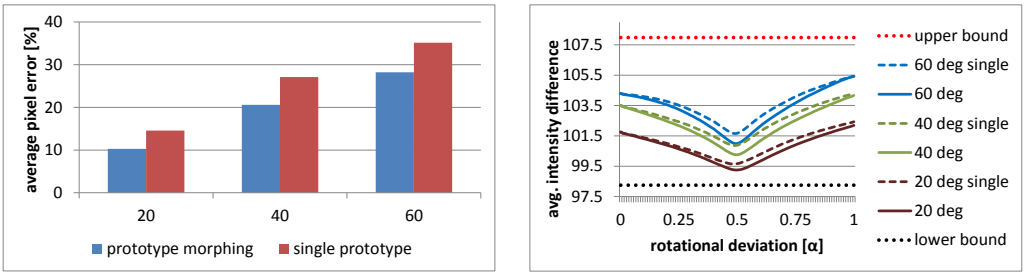


Figure 5.22: Average pixel error in percent between observation and our rendering given the true parameters (**left**) and the objective function between prototype parameters (**right**) for $\pm 20°$, $\pm 40°$, and $\pm 60°$ degrees prototype distance using one or two prototypes.

We performed these experiments with the viewpoints of the prototypes being $\pm 20°, \pm 40°$, and $\pm 60°$ rotated with respect to the target view and in order to assess the impact of our morphing scheme, once using both prototypes and once using only the $\vec{p}^{-}$ prototype. To aid interpretation, we calculated the upper bound (the average pixel value of $\boldsymbol{I}^{obs}$) and the lower bound (by subtracting the true target view from the image and then determining the average pixel value).

The resulting graphs are shown on the right of Figure 5.22. Averaged across our 2000 repetitions all experiments show clearly defined optima around $\alpha = 0.5$. As expected, closer prototype viewpoints lead to better results and morphing between two prototypes increases the accuracy of the prediction, significantly. Also, the closer the prototype views are to the target pose, the steeper the objective function gets around the true optimum.

One thing to notice, though, is that when only using the $\vec{p}^{-}$ prototype, the function's global optimum is slightly ($\leq 1.2°$) biased to the negative side; when using both prototypes the global optimum is located at the true ground truth optimum.

To better rate the improvements we have compared the function's value at the true optimum. We show these results on the left of Figure 5.22. The scale is normalized with respect to the determined bounds, *i.e.,* 0% corresponds to the lower and 100% to the upper bound. The improvement between using two prototypes compared to using just one is consistently between 20% and 30%. With two prototypes at $\pm 20°$ viewpoint deviation, our prediction and the ground truth image differ by only 10%. Also at higher prototype viewpoint distance, the improvement when using the morphing scheme is substantial. With 28% error using two prototypes at $\pm 60°$ our method almost achieves the same score as using one prototype at $\pm 40°$ (27% error).

5.3.2 Analysis of the proposed objective function

To compare our proposed pixel-wise objective function (Equation 5.14) with the state of the art in hand tracking on RGB images, we have implemented the method of Oikonomidis *et al.* [194]. Their approach optimizes an objective function using *particle swarm optimization* (PSO) [198], simultaneously maximizing the overlap between the rendered silhouette and the segmented skin-color and minimizing the mutual average edge distance. We have chosen this approach since skin-color based overlap (further denoted as *silhouette term*) and edge distance (denoted as *edge term*) appear - in different formulations - in most existing approaches, *e.g.,* [186, 187, 190, 194, 197, 220] as well as in the existing variants [200, 201, 202, 221, 222, 223, 224]. Additionally, we analyze the simpler formulation of our proposed method, directly based on average per-pixel differences (Equation 5.13), denoted as *average*.

We implemented the skin-color segmentation by back-projecting an HSV histogram sampled from the observation. Following the paper, we implemented the edge extraction using Canny edge extraction. For computing the silhouette and the edge maps we use the same articulated model as in our approach with fixed segment shapes. For our image-based rendering approach, we always use two prototypes at $\pm 20^\circ$.

Due to the high movement speed of the human hand, observed images often suffer from motion blur. We simulated the impact of motion blur by repeating the experiments with increasing levels of Gaussian blur applied to the images. Figure 5.23 shows plots of the numerical derivatives of the three objective functions in the two setups.

Expectedly, on cluttered background, the silhouette term leads to very good results at all levels of blur. Without blur, also the edge term shows a clearly defined optimum. However, with increasing level of blurring, the edge term becomes constant and thus becomes useless as an objective function. This is due to the increasingly difficult edge extraction and is therefore a principle problem of all approaches based on edge cues. Our proposed method is not dependent on a feature extraction and shows a clearly defined optimum at all levels of blur.

In the hands setup, the silhouette term suffers from numerous local optima and also exhibits compromised validity due to a high outlier rate, as we will show later. Without blur, the edge term is mostly able to determine the correct optimum but fails doing so, even at light to moderate levels of blur. In contrast, our approach is not largely affected by the skin-color fore- and background.

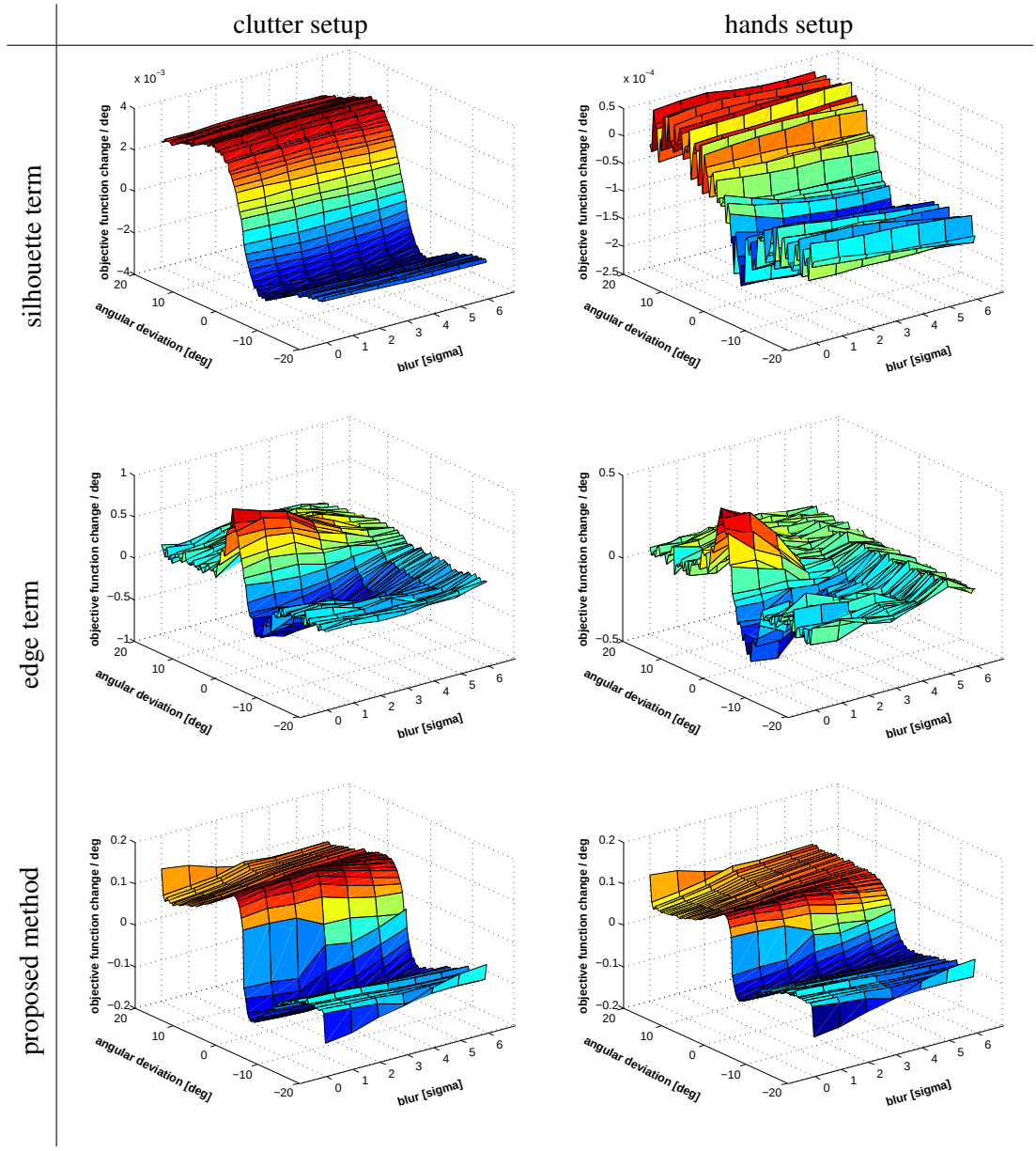


Figure 5.23: First derivative of the analyzed objective functions in the two setups under increasing levels of blur. The skin-color based silhouette term (top row) does not perform well on skin-colored back- and foreground (right column) while edge based terms (center row) are not robust towards blur. Our proposed method (bottom row) exhibits a well-defined optimum in both setups and is very robust towards blur.

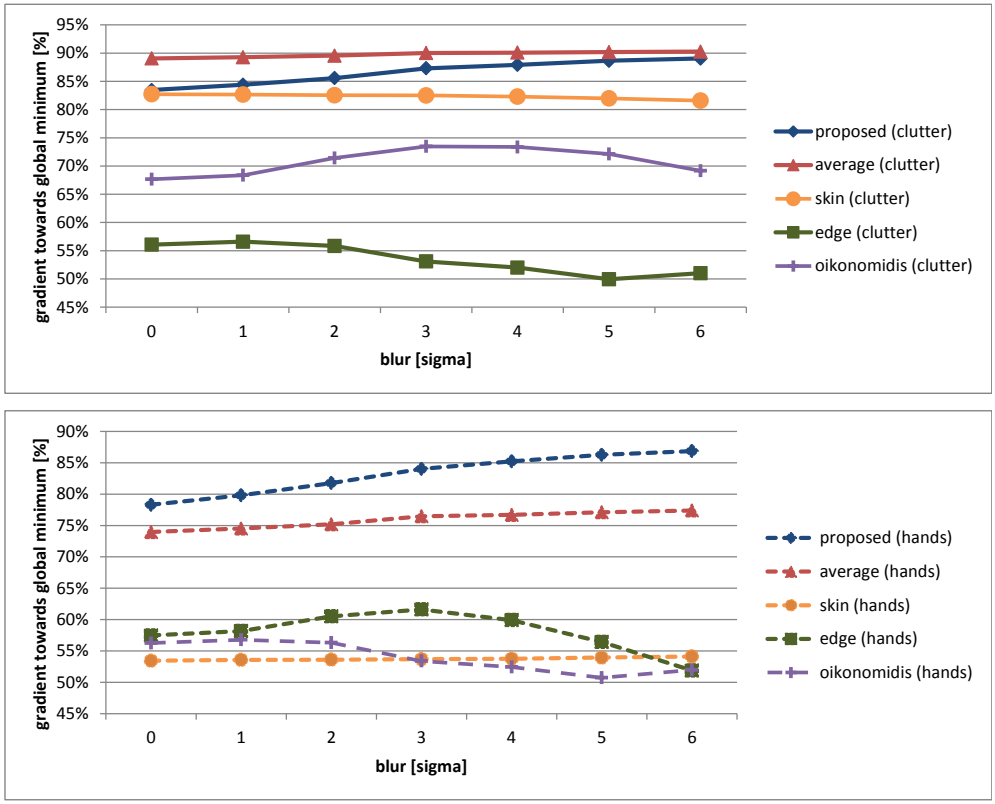


Figure 5.24: Percentage of gradients pointing towards global minimum sampled at 0.04° and averaged over the 2000 repetitions in the clutter (**left**) and hands (**right**) setup.

This is supported by our evaluation of the amount of local minima, shown in Figure 5.24. We counted how persistently each of the functions exhibits gradients that point towards the correct global minimum along the path from $-20°$ to $20°$. While our proposed method achieves an average ratio between 78% and 90% at all levels of blur in both setups, the next-leading state-of-the-art approach achieves 62% at best in the hands setup. While our proposed function is even slightly outperformed by the average pixel difference in the clutter setup, it clearly outperforms the simpler formulation in the hands setup.

After studying the function behaviors with respect to local optima, we also examined the principal correctness of the global optimum of each function. To that end, we examined the distance of the function's global optimum from the true location of the optimum according to ground truth.

The results are shown in Table 5.1 and in Figure 5.25. In the clutter setup, simple average

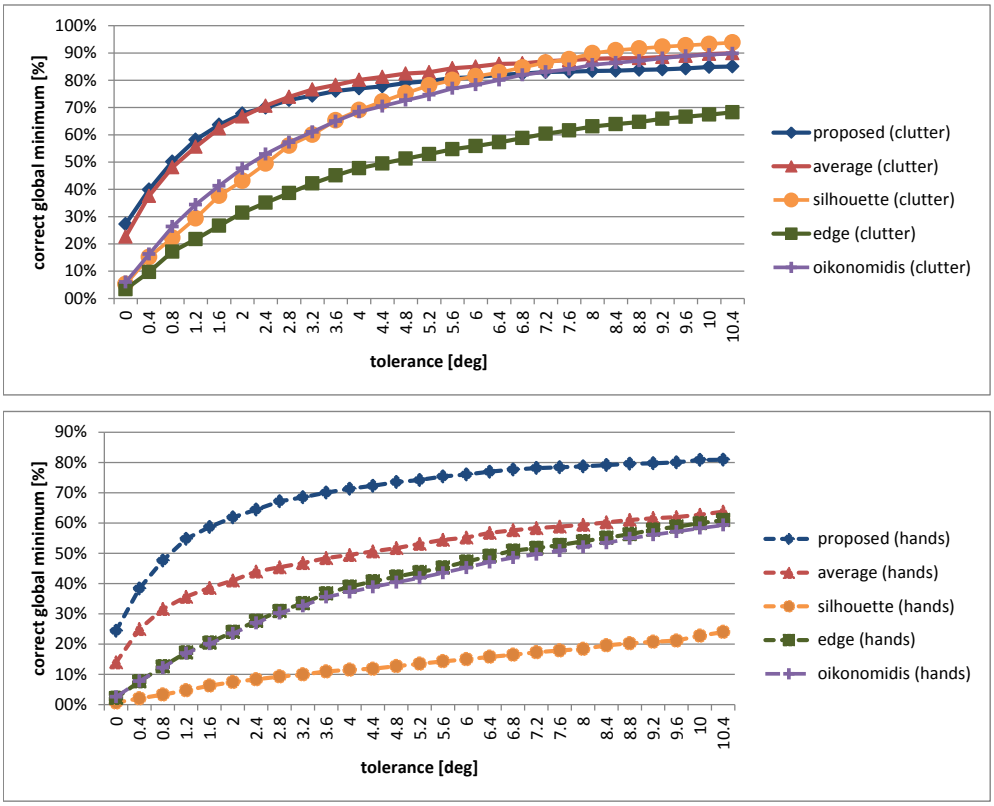


Figure 5.25: Percentage of repetitions with a correct global minimum with respect to an accepted tolerance for the clutter (**top**) and hands (**bottom**) setup.

and our proposed function both achieve a high accuracy. These functions result in an optimum within 2° of ground truth in about two thirds of the 2000 repetitions. For comparison, the next best approach (Oikonomidis) finds the optimum in less than 50% of the cases within he same tolerance. In the hands setup, our proposed function is only slightly worse with 61.9%, while simple average deteriorates to 41% and the remaining approaches are all below 25%.

5.3.3 Database tracking procedure

In this subsection, we want to study the database tracking approach before adapting to the observation, *i.e.,* only with synthetic entries in the database. Specifically, we want to demonstrate that it is able to reliably recover from tracking failures due to the broad spectrum of hypotheses that are tested in each iteration. This is a crucial prerequisite for the given problem, since the hands often disappear or get occluded during the course of a workflow. Note, that all results

		silhouette	edge	oikonomidis	average	**proposed**
clutter	% correct (=0°)	5.3%	3.4%	6.0%	22.7%	27.3%
	% correct ($\leq 2°$)	43.2%	31.5%	47.7%	66.7%	67.8%
	% outlier ($> 10°$)	6.7%	32.6%	10.7%	10.4%	15.1%
	std.dev	5.31°	11.13°	6.10°	6.23°	7.20°
hands	% correct (=0°)	0.7%	2.2%	2.6%	14.0%	24.4%
	% correct ($\leq 2°$)	7.5%	24.0%	23.5%	41.0%	61.9%
	% outlier ($> 10°$)	77.3%	40.1%	41.7%	37.3%	19.2%
	std.dev	16.55°	10.01°	11.41°	11.43°	8.10°

Table 5.1: Analysis of the global optima.

within this subsections are entirely synthetic and generated without incorporating any model adaptation. While the adaptation process will significantly improve the results, here we merely aim to demonstrate the general applicability of the underlying database approach.

We distinguish between observations of only dorsal views of the hand and observations containing arbitrary rotations of the hand, even ones that would not occur in first-person views of the user's own hands. The difference is that a dorsal view is generally sufficiently characteristic to fully estimate the hand pose from a single frame. Otherwise, single frame hand tracking is an ill-posed problem due to self-occlusion and ambiguous observation, *i.e.,* very different hand postures producing very similar feature vectors.

We prepared three test sets, each consisting of 40 completely random (within anatomical constraints) 9 DoF poses. These poses are interpolated to produce a 1950 frames long image sequence of a smoothly moving hand for each set, rendered as silhouette and contour edge model without shading. We used the same model to fill the database, however, with slightly different morphological parameters. Although the test sequence mostly comprises images with parameter vectors that have no exact match within the database, we additionally altered the output to create a more realistic test case. Each sequence was rendered with altered finger radii and the output image was scaled slightly larger, compared to the database content. All sequences were rendered on black background. We created the following three test setups:

Dorsal/dorsal: external parameters fixed to produce dorsal views only and the database constrained to dorsal views as well.

Dorsal/full: external parameters fixed to dorsal views but using the entire database.

Full/full: external parameters contain all 3 rotational DoFs and again using the entire database. Since some of the frames show frontal, fist-like poses, which are somewhat ambiguous for a single-frame method, this set shows if and how quickly the method can recover from tracking losses.

	matched	**best match**
dorsal/dorsal	92.5%	20.6%
dorsal/full	58.4%	9.6%
full/full	7.9%	0.9%

Table 5.2: Results on the synthetic test sets: The first column shows matching rate, i.e. the beam is centered on the correct tile. The second column shows the matching rate where the best hypothesis was the nearest neighbor entry in the database.

For each test set, we centered the search beam on the correct tile at the first frame and then processed the entire sequence with our proposed method. Table 5.2 shows matching rates for the three setups. In the dorsal/dorsal setup, where the data set and the database for tracking have been constrained to only comprise dorsal views, the system achieved a matching rate of 92.5%. So, almost all frames were correctly matched to the according cache tile, while on average every 5th frame was matched to the nearest neighbor in the database. On an equally constrained data set but using the full database (dorsal/full), these numbers drop to slightly below 60% resp. 10%. The reason for this is that the tracking system quite often erroneously matches a ”mirrored” template after tracking loss due to ambiguous or misclassified frames. In the entirely unconstrained setup, these numbers drop to approx. 8% and 1%. The reason for this is that the sequence contains views with very ambiguous projection appearance, see Figure 5.26 for examples. In the next subsection, we will show that the tracking performance drastically increases when adapting to the observed content.

The graph in Figure 5.27 shows histogram and cumulative distribution of angular tracking errors. The tracking results for 90% of the frames were below $\pm 30^{\circ}$ external pose deviation when applied to the quite descriptive dorsal views. For fully unconstrained movement 30% of the frames have been matched with this or less deviation. Table 5.3 contains a quantitative overview of the tracking mean, median, and maximum tracking error, separately for all frames and matched frames (*i.e.,* beam centered on correct tile) only.

Figure 5.26: Two examples of ambiguous views due to self-occlusion contained in the test set (*full/full*).

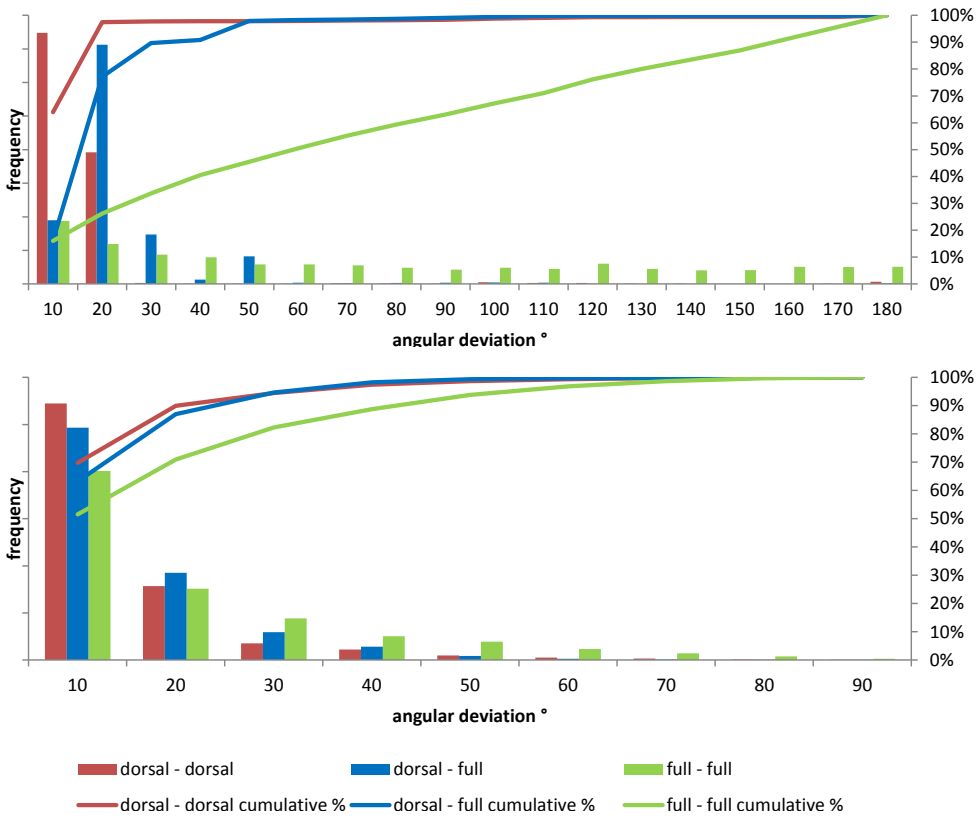


Figure 5.27: Histograms of the tracking error for external parameters (top) and internal parameters (bottom).

		mean	std.dev	median	max
dorsal/dorsal	external	8.7°	18.5°	3°	175°
	-matched	5.5°	7.2°	0°	15°
	internal	9.2°	10.9°	6°	96°
	-matched	5.6°	4.0°	4°	15°
dorsal/full	external	19.0°	15.2°	15°	175°
	-matched	14.9°	1.3°	15°	15°
	internal	10.0°	10.3°	7°	94°
	-matched	6.7°	8.7°	6°	81°
full/full	external	71.0°	56.0°	60°	179°
	-matched	9.2°	6.75°	8°	22°
	internal	16.2°	17.3°	10°	105°
	-matched	10.3°	10.3°	6°	90°

Table 5.3: Parameter errors between ground truth and best hypothesis: For each synthetic set the mean, std.dev., median, and maximum deviation are given separately for external and internal parameters and each separately for all frames resp. matched frames.

5.3.4 Adaptation and model-guided generalization

We will show how the tracking performance in terms of correctness is significantly improved with the image-based appearance model compared to just using silhouette and edge cues. Additionally, we will show that the explicit, model-guided generalization has a strong effect on the tracking performance on sequences recorded by different users.

For this experiment, we use three recordings of the first four steps of the ”Plugs & circuit board” workflow introduced in Section 4.4, performed by two different persons. Example frames are shown in Figure 5.28. To generate ground truth, we semi-automatically labeled the hand postures within the three sequences by using the method proposed within this chapter, manually verifying the results and correcting errors. We then used one of the two sequences that were performed by the same person as training material to sample hand view prototypes in six different ways:

Dense: Using all labeled frames with a visible hand as prototypes.

Sparse: Only using four key frames as prototypes, shown in Figure 5.29.

Generalized 5° / 10° / 15° / 20°: Generalizing the four sampled prototypes from the sparse set by different amounts. Within kinematic constraints, we randomly synthesized parameters with 5°, 10°, 15°, and 20° maximum deviation per joint, creating four independent database sets.

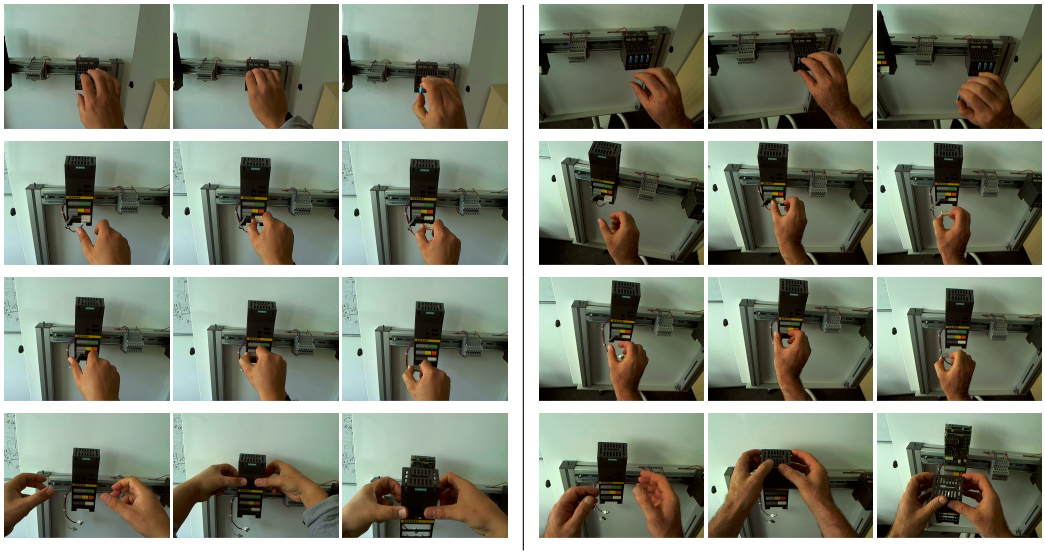


Figure 5.28: Example frames from the test sequences: Left side recorded by the same user, right side recorded by a different user.

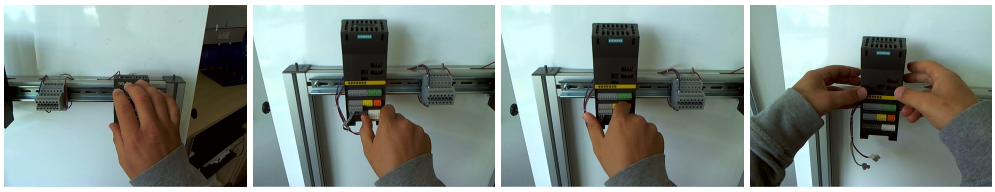


Figure 5.29: Prototype views used in the sparse set.

We then applied the six different databases to the two test sets shown in Figure 5.28. Hereby, only the database tracking approach as described in Section 5.2.2 was used, without further refinement of the weighted query results. The results of each adapted database were then compared to the verified ground truth parameters per frame. We then analysed the respective ratios of correctly recognized hand postures with respect to a tolerated deviation ranging from 0° to 30°.

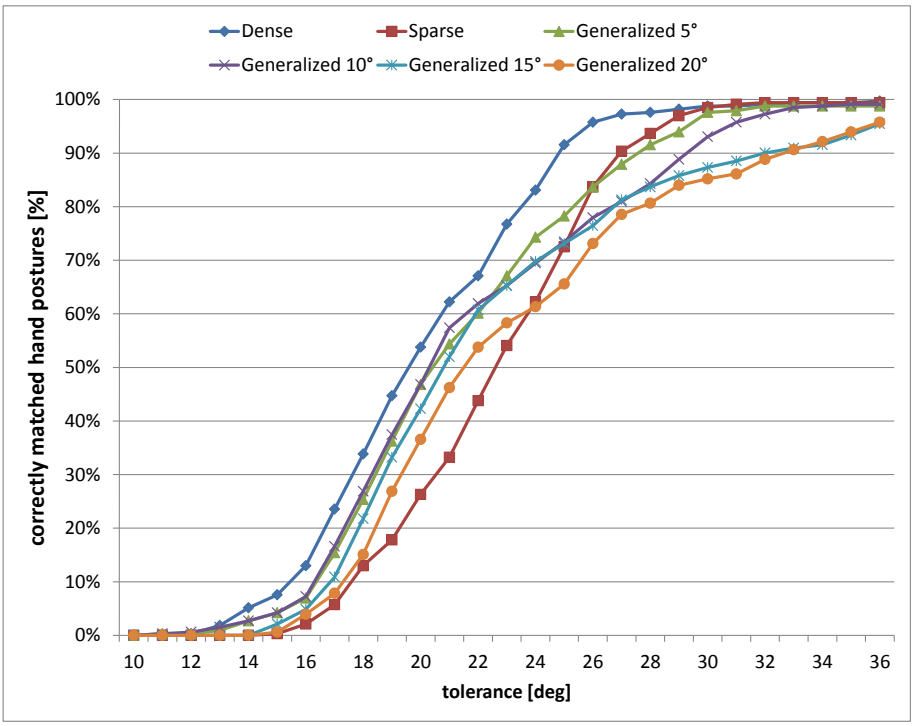


Figure 5.30: Rate of correct detections within allowed average deviation with training and target material recorded by the same user, compare Figure 5.33.

We first discuss the case, where training and test sequence are recorded by the same person. Figure 5.30 shows the resulting graphs and Figure 5.31 according example frames. Additionally, Figure 5.32 presents the achieved results after refinement in an offline optimization process based on PSO. As both performances strongly resemble each other and therefore most postures are already contained in the dense database, this database performs best on this test set. With respect to a tolerated deviation of 20° the correct recognition rate is double as high than that of the sparse database with 50% compared to 25%.

The fact that both curves meet at about 30° allowed deviation is an indicator for the maximum deviation of the user's hand posture while approaching and retracting from the respective key poses. Expectedly, the effects of generalization do not improve upon the dense sampling of all hand postures in the training material and produce results in between these two boundary conditions. Interestingly, the sparse database even outperforms the generalized at larger allowed deviations due to an increased likelihood of misclassification in the generalized models.

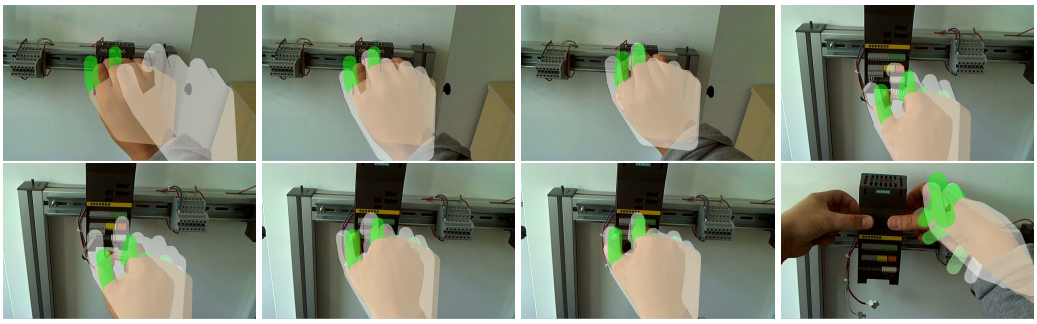


Figure 5.31: Tracking results for the test set from the same user: White overlay represents the database tracking results. Green/beige overlay represents the results after outlier rejection (using the objective function) and translational refinement.

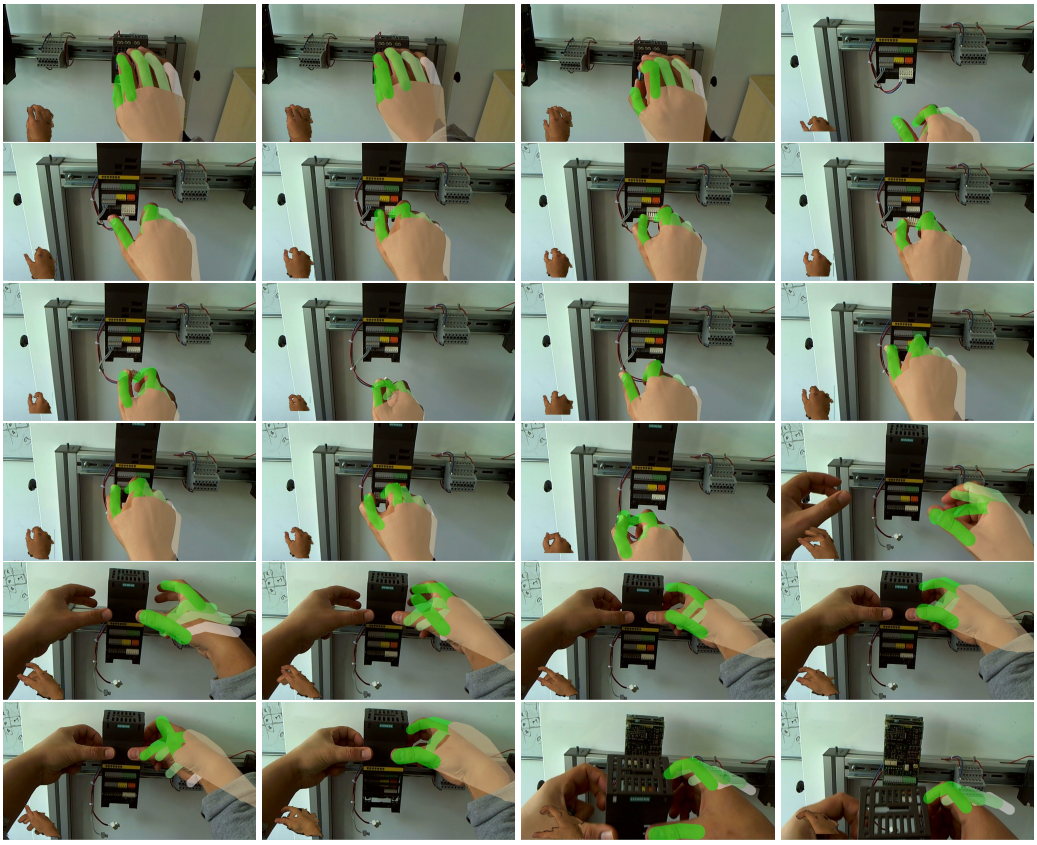


Figure 5.32: Tracking results on the same user test set after refinement.

The higher the generalization radii are, the earlier this break-even point is reached, beginning at 24° for the widest-spread generalization radius.

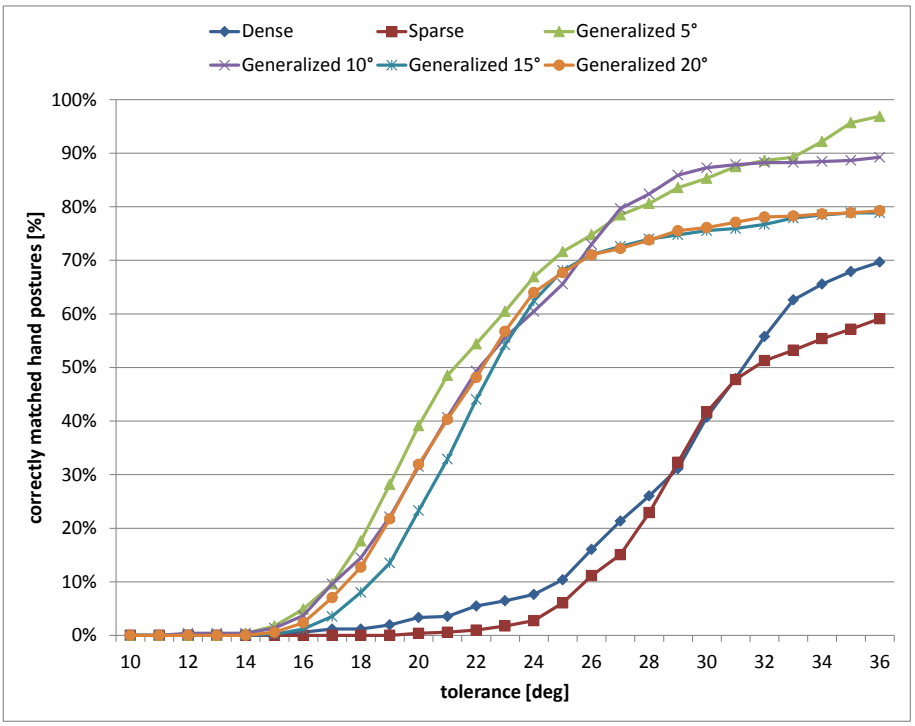


Figure 5.33: Rate of correct detections within allowed average deviation with training and target material recorded by different users, compare Figure 5.30.

When applying these databases to the test set, performed by a different user, the results are entirely different. The resulting graphs are shown in Figure 5.33, with Figure 5.34 showing example frames. The offline-refined results for this case are shown in Figure 5.35. As the hand's morphology and the modes of execution and even the viewpoint are slightly different, the two non-generalizing databases perform significantly worse than the generalized models. At an allowed deviation of 20°, the dense and sparse databases are virtually unable to match any of the frames of the test sequence, with 3% and below 1%, respectively. Compared to that, the best-performing generalized database is able to recover the hand pose in about 40% of the frames within this allowed tolerance.

One observation that is consistent with the same-user test set is that higher generalization radii lead to higher likelihood of classification errors. From the analyzed generalization radii,

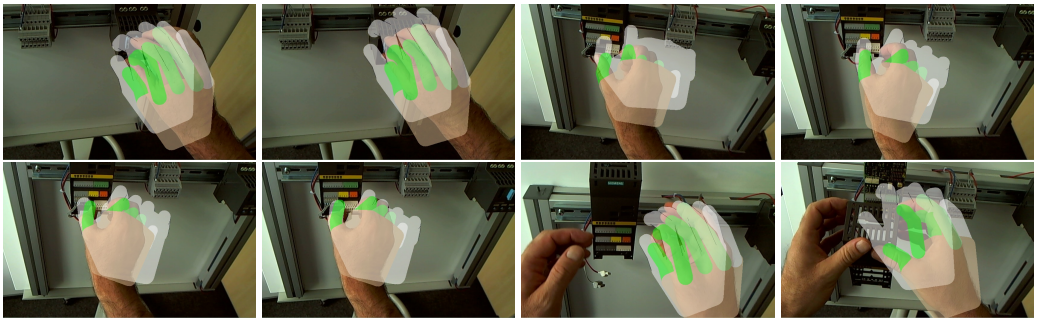


Figure 5.34: Tracking results for the test set from a different user: White overlay represents the database tracking results. Green/beige overlay represents the results after outlier rejection (using the objective function) and translational refinement.

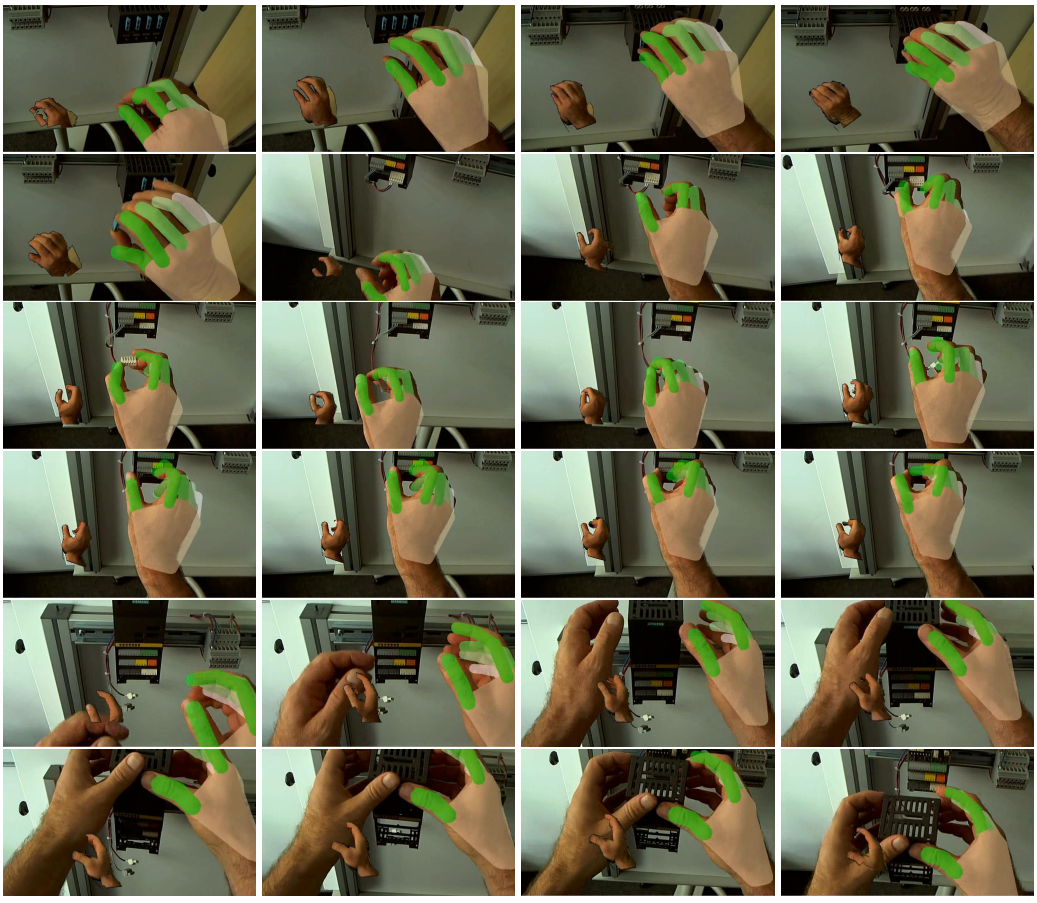


Figure 5.35: Tracking results on the different user test set after refinement.

5°, 10°, 15°, and 20°, the lowest radius of 5° performed best in both cases. This issue can be addressed with an outlier rejection. This is implicitly realized in the PSO-based refinement procedure (Section 5.2.2.1), as the exploration of the objective function starts simultaneously at all hypotheses, thus rejecting the outliers. However, for the use case of real-time adaptive hand tracking, the lower generalization radii clearly perform best.

As the tracking system operates on images from a monocular RGB camera, the error in estimating the depth can be expected to be higher than the error within the image plane. We therefore analyzed the average per-joint spatial deviation within the image plane and the depth component, separately. We hereby compare the best-performing non-generalizing database (dense) and the best-performing generalizing database (10°) in this respect. Since we are interested in accuracy, we explicitly removed erroneous outlier matches, *i.e.,* matches with an average spatial error of more than 3 cm. The results can be seen in Figure 5.36.

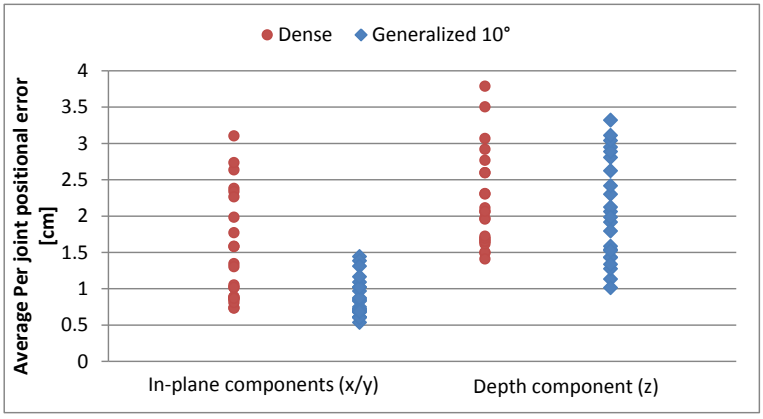


Figure 5.36: Scatter plot of all occurring spatial reprojection errors, separated by components within the image plane and depth.

The dense database leads to an average error-component of 1.5 cm parallel to the image-plane, compared to a slightly higher depth estimation error of 2.2 cm. Using the generalizing database, the depth estimation remains almost as high (2.1 cm), while the in-plane estimate is reduced to less than 0.9 cm.

6

Authoring and Presentation

To this point, we have covered all aspects that deal with the segmentation-, tracking-, and generalization-related aspects of the proposed workflow monitoring. So far, this has laid out the direction from observation to the tracking model as well as the technical aspects of real-time spatiotemporal tracking in order to present information adequately in time and position.

In this chapter we will describe the method to process the acquired information, in order to present it to a human recipient. These presentation-centric aspects of the authoring process have a crucial impact on the perceived quality of the task assistance. In the following sections, we first discuss concept-related difficulties within the AR context and how we address these in our framework. We then continue with presenting the single types of instructions and real-time feedback provided by our system and how we associate them with the corresponding work step.

In addition to the automatic authoring, we also intend to enable the user to manually add visual annotations. At the end of this chapter, we will briefly present our approach to manual authoring that does not require any knowledge in 3D computer graphics or computer vision from the user.

6.1 Concept-related challenges

While Augmented Reality (AR) has often been proven to have a positive impact on several key performance indicators, the concept suffers from known but often ignored difficulties, immanent to the AR principle. The authors of [52] have evaluated the use of an AR system for maintenance in a military scenario. Their results show significantly reduced head rotations

compared to using a mobile display with maintenance instructions and significantly reduced task localization times. For the overall task-completion time, however, the authors report a faster execution using the mobile display and explain this through the reduced comprehensiveness of instructional overlays in AR compared to the technical sketches on the mobile display. While their results are very promising and clearly demonstrate benefits in ergonomics and general muscoloskeletal strain, it also shows that the nature of the visual presentation is a key factor for the effectiveness.

Certainly, a large amount of this deterioration can also be attributed to ergonomic issues with head-mounted displays as often reported in according studies [4, 52]. Very recently, Marner *et al.*[225] have investigated the usage of AR for procedural task assistance without the deteriorating effects of an HMD. They are successfully using projection-based Spatial Augmented Reality (SAR) to improve procedural task performance. However, while SAR cannot be applied in all scenarios, there also exist fundamental difficulties directly related to the concept of first-person perspective, which are not addressed in their synthetic experimental setup:

The viewpoint from which objects that are involved in a procedural task are shown has a clear impact on understanding. For most objects, there exists a so-called *canonical view*, denoting the view from which an object is most characteristic, [226]. A simple illustrative example would be a coffee cup that is easiest to recognize by a human observer, when viewed from the side with the handle visible. The same concept clearly holds for procedural tasks and is trivially reflected in pictographic manuals, by depicting object parts and actions in easy to recognize poses. Canonical views are well studied in cognition research, also with respect to dynamic scenes [227]. However, the principle is not transferable to the presentation paradigm of Augmented Reality, where the viewpoint is constrained to the user's viewpoint. Therefore, AR and canonical views are in a way competing concepts for alleviating object identification for a human observer. In case of AR, unfortunately, this adds the cognitive burden of having to deal with occlusions or generally poor observability of important workspace parts:

While in technical manuals, this is often handled with explosion sketches, it is not entirely straight-forward to overcome this in AR. One approach is to incorporate explosion diagrams in AR [95]. However, the principle problem of obscured information from non-optimal viewpoints remains. Additionally, as humans tend to use occlusions as the dominant cue for depth ordering [228], the intuitive presentation of obscured information in AR is a broadly investigated but still challenging task [229, 230].

One possibility to overcome this is to drop the general AR presentation principle and use an untracked display showing optimal viewing angles for each work step. However, the performance of stationary or mobile display setups is provably much deteriorated as for example [4, 52, 69, 94, 225] have experimentally confirmed.

Another possibility is simply changing the user's viewpoint into a more adequate configuration. Several techniques have been developed for this, most notably attention funnels that allow to intuitively indicate viewpoint changes in all six degrees of freedom [92]. Attention funnels have been extensively evaluated in picking tasks [53], where directing the user is of particular importance.

While we cannot influence the viewpoint from which the reference sequence was recorded, we can however direct the user to that viewpoint, *e.g.,* using attention funnels. This does in fact mitigate the viewpoint problem, given that the person recording the reference sequence has deliberately chosen didactic viewpoints. Additionally, this is also a technical necessity as the video snippets that are extracted from the reference sequence (explained in Section 6.3.1) are obviously determined in their shown viewpoint.

Although this does not conclusively solve the issues with non-canonical views and occlusions, it by all means provides the possibility to solve these issues through a deliberate choice of viewpoints when recording the reference sequence. Therefore, incorporating a mechanism for location change directly into the presentation paradigm is a key prerequisite that is widely ignored by the state of the art.

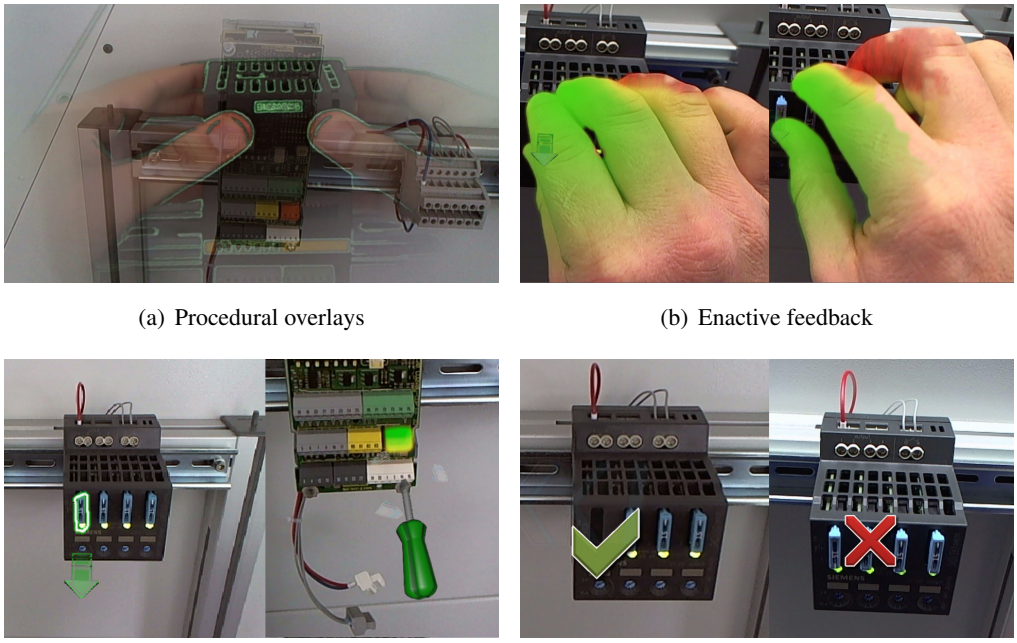
6.2 Visual representation

In this section, we present the different types of visual hints to guide the user through the workflow. The emphasis lies on their technical realization, *i.e.,* how they are assessed from the reference and the run-time observations.

We distinguish four types of visual overlays, see Figure 6.1 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.



(a) Procedural overlays

(b) Enactive feedback

(c) Annotational overlays

(d) Assessing overlays / optical validation

Figure 6.1: Visual feedback provided by the system.

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.

We will show how these overlays are generated automatically in Section 6.3 and manually in Section 6.4. In the remainder of this section, we show how the displayed information is scoped temporally and the user's view is guided spatially.

6.2.1 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,

Figure 6.2: Visual clutter due to procedural overlays interfering with the current appearance of the workspace.

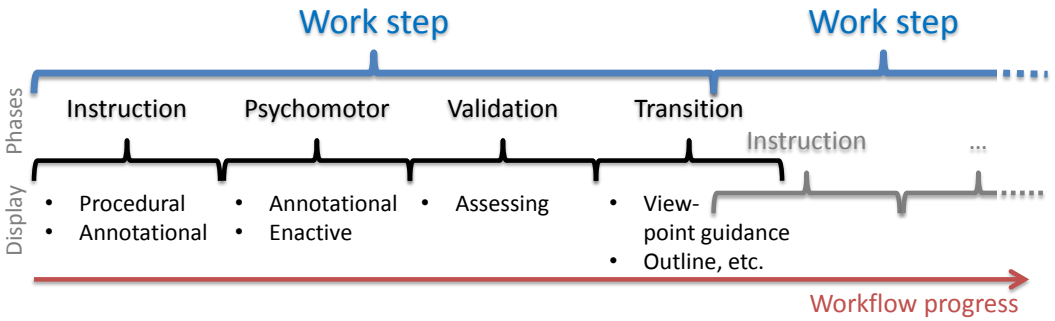


Figure 6.3: Illustration of the four partly overlapping phases distinguished within each work step and the respectively displayed information.

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. Figure 6.2 shows an example of visual clutter already occurring with a single type of visual overlay when not scoping the displayed information.

Figure 6.3 shows the temporal order of the distinguished phases, in particular the *instruc-*

tion 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-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, described in Section 4.2.2.

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 [109] also propose 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.

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.

6.2.2 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. In line with the considerations made in Section 6.1, 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 [92] to guide the user towards the target viewpoint. Figure 6.4 shows an example of the visual representation.

From the approach described in Section 4.2.1, we receive a homography relative to the reference material, which we can propagate frame-by-frame using the camera tracking approach from Section 3.2.1. We draw an axis-aligned rectangle representing the current view in the center of the screen. Let $\vec{p}_i, i = 1..4$ denote the four corner points of that rectangle and $\mathbf{H}$ be the

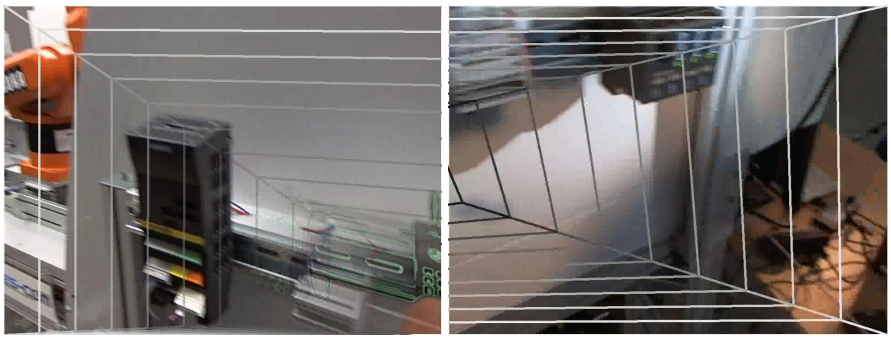


Figure 6.4: Example of the attention funnels used to guide the user to a target viewpoint.

target homography, relative to our viewpoint. We then repeatedly draw rectangles between the corner points $\vec{q}_i = (\alpha\mathbf{H} + (1-\alpha)\mathbf{I})\vec{p}_i$ with α ranging from 0 to 1 in steps of 0.1. This visual hint is hidden from the user, if the relative homography $\mathbf{H}$ is sufficiently small, as indicated by the measures described in Section 3.2.2.

6.3 Automatic overlay generation

Building upon the acquired information, we are able to automatically generate a rich set of visual overlays. The following subsections explain the processing steps for the technical realization of each type.

6.3.1 Procedural overlays

Since we use first-person view videos as input material, the straight-forward way to generate an instructional animation of a work step is to use the corresponding parts of the reference sequence, directly, as illustrated in Figure 6.5. 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

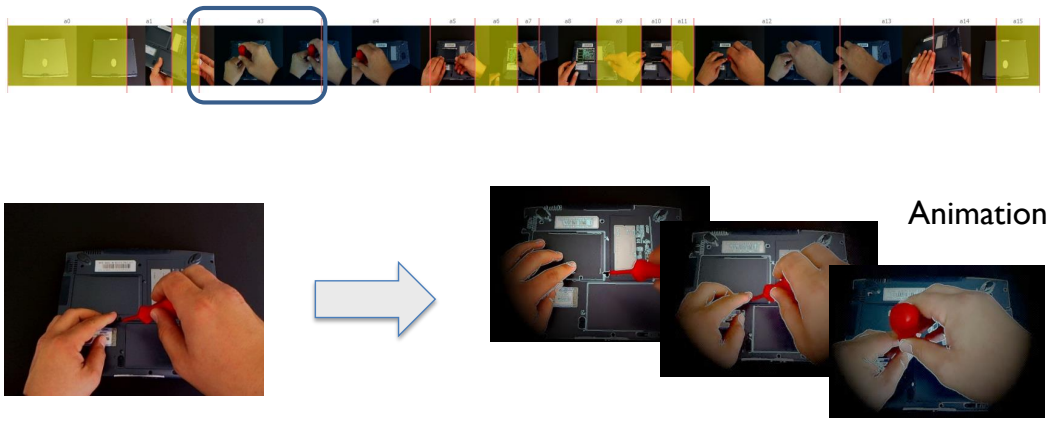


Figure 6.5: Illustration of our method to automatically generate procedural overlays based on the segmentation results.

tracking information, we would indeed be 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.

6.3.2 Enactive feedback

Through back projecting the color-coded location probability maps, described in Section 4.2.2 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.

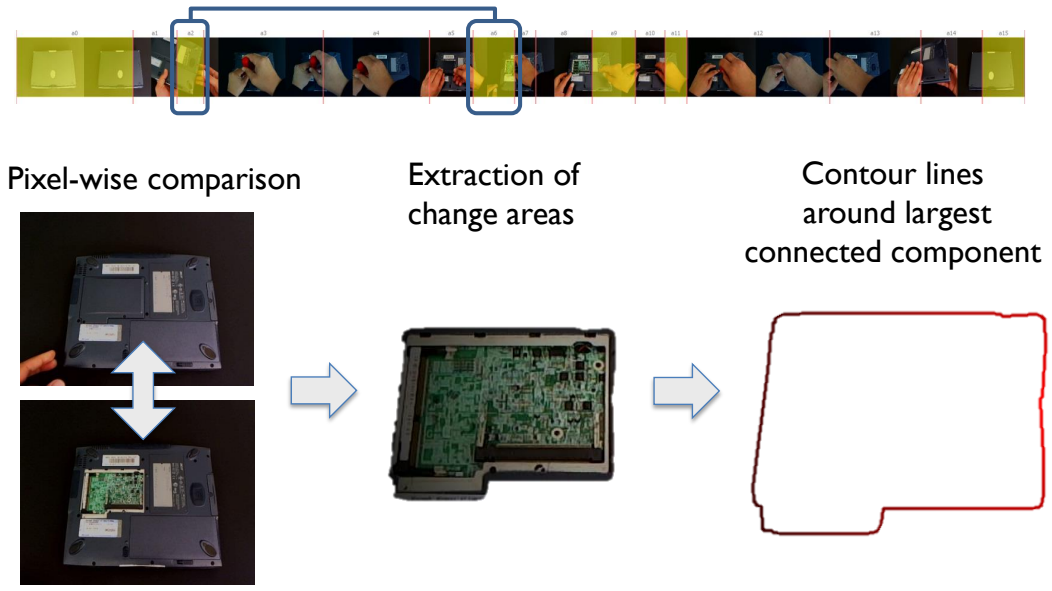


Figure 6.6: Illustration of our method to automatically generate annotational overlays indicating changed workspace areas based on the segmentation results.

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 6.1(b) and 4.5. In addition to indicating clearly incorrect hand positions, it also reassures the user of the ongoing support through the system.

6.3.3 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 Figure 6.6. This is achieved through registering the Relevance Planes for the preceding and subsequent static segments using the approach described in Section 4.3.2. 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 Figure 6.1(c).

6.3.4 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 translational ($+6$, 0, -6 pixels in x & y direction), rotational ($+5°$, $0°$, $-5°$), and scaling (90%, 100%, 111%) deviations 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 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 Figure 6.1(d).

6.4 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, discussed in Section 6.1, 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.

6.4.1 Structuring view

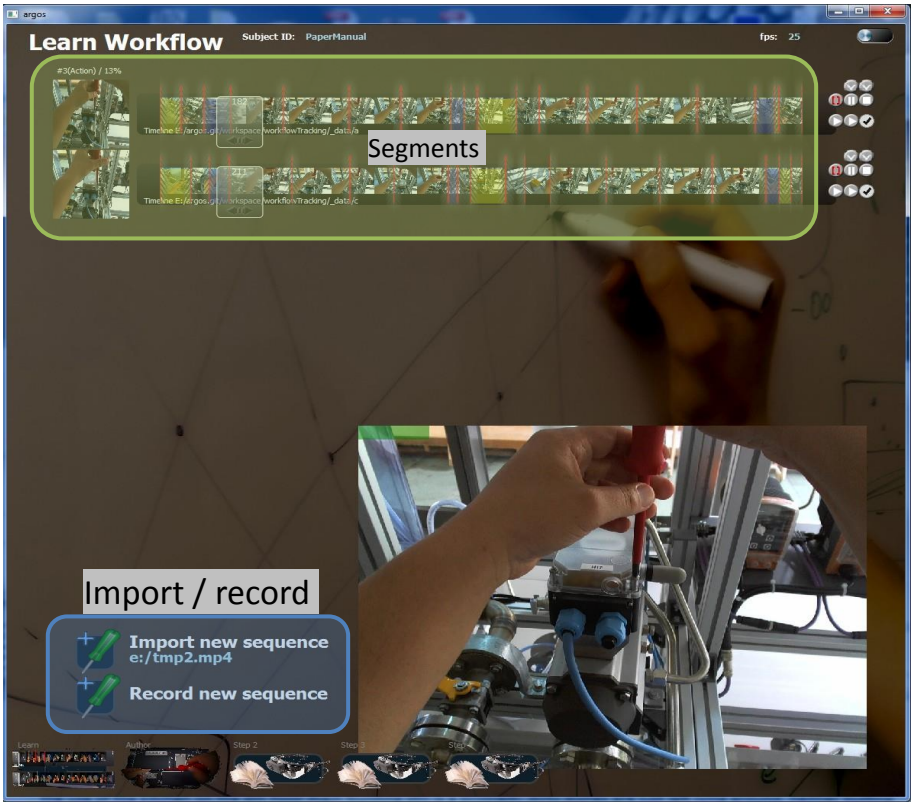


Figure 6.7: Labeled screenshot from the learning view of the authoring tool: The recognized actions within all available workflow recordings are presented to the user.

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 Figure 6.7. In particular, the user can add new recordings to the analyzed data

body (Figure 6.7, section ”Import / record”).

After adding a sequence, it is presented as a film strip with the segmentation result overlaid using different tints (Figure 6.7, 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.

6.4.2 Authoring view

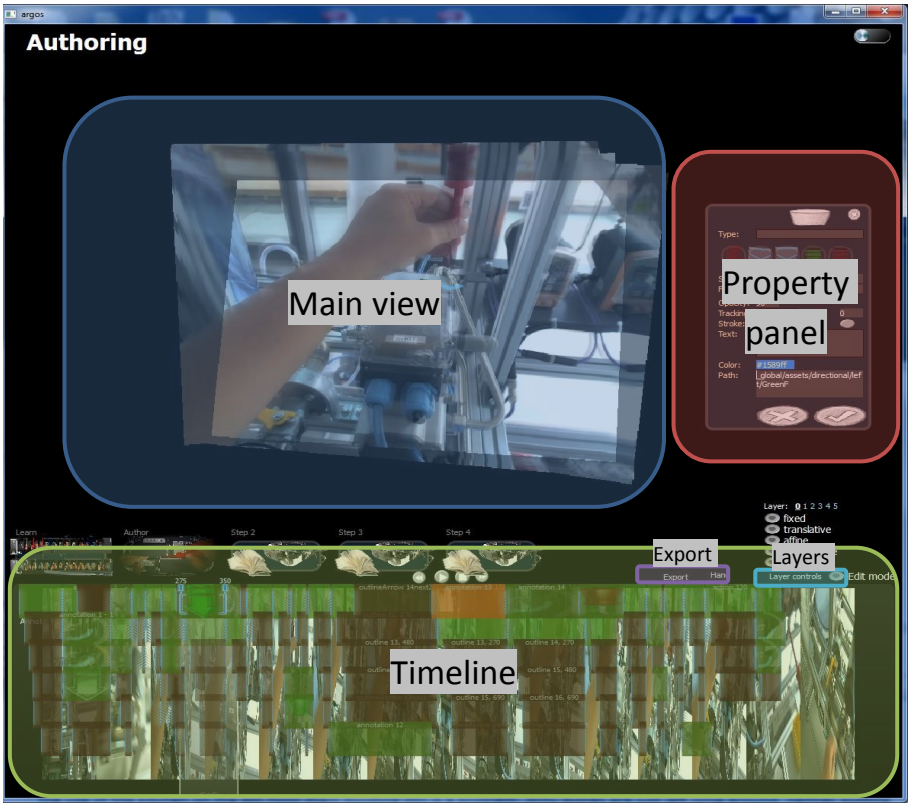


Figure 6.8: Labeled screenshot from the authoring view of the authoring tool: The current frame is projected onto the common frame to allow easy annotation within a stabilized frame of reference.

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 disjunct 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 view is shown in Figure 6.8. The editing takes place within the stabilized common frame of the according Relevance Plane (Figure 6.8, 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. Figure 6.9 shows the set of currently supported gestures and illustrates the procedure.

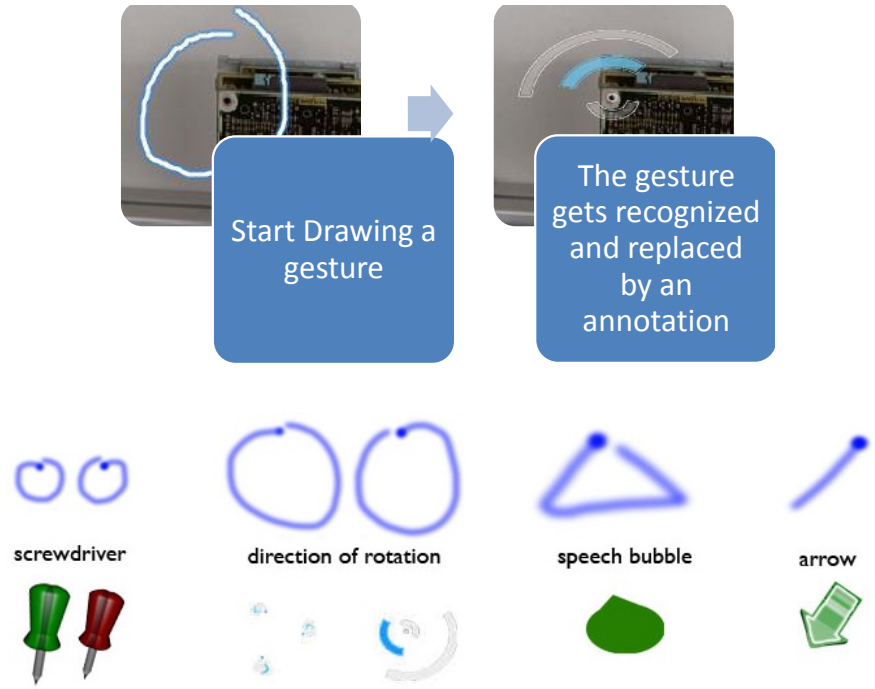


Figure 6.9: Illustration of the annotation procedure and selection of supported gestures.

7

Implementation

In this chapter, we present implementation details and benchmarking results of the system. A considerable amount of effort was invested in exploiting the streaming data nature of the underlying application. The outcome is a comprehensive programming model, specifically designed for auto-parallelizing stream processing. Since real-time performance is a crucial prerequisite for our application and AR in general, we will briefly present our approach. It is not limited to the application at hand and allows quick prototype development that immediately runs at speeds that would otherwise require elaborate fine-tuning of the implementation. Additionally, it facilitates the incorporation of multiple and heterogeneous physical machines within its parallelization strategy: We will show achieved results, considering the importance of tablets, smartphones, and even wearable mobile devices as target platforms.

7.1 Programming model

Developing efficient software for current multi-core and future many-core hardware is becoming an increasingly difficult task. While some constraints like the demand on memory efficiency could be relaxed to some extent due to cheap memory, a well parallelized implementation still remains an important requirement. This not only puts higher requirements on the algorithmic design and thus on the skills of the developer, but also leads to additional code issues that are very hard to discover.

There are several approaches to facilitate the writing of parallel code. There is the family of dedicated stream-processing languages, most notably nVidia CUDA [231] and OpenCL [232]. However, our proposed approach is closer to the class of declarative languages or language

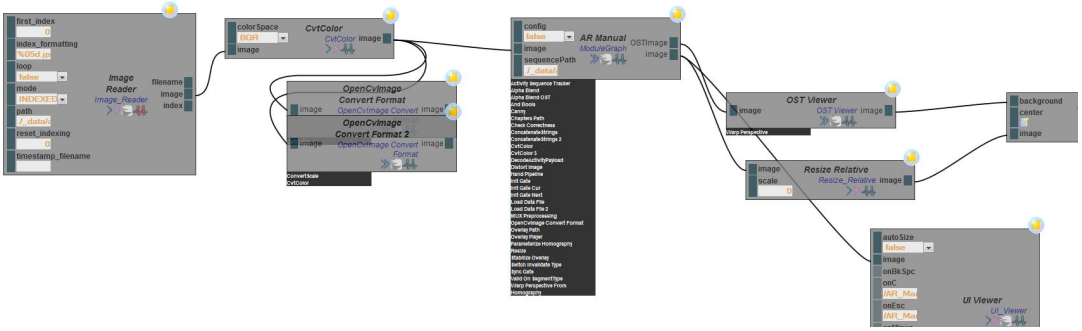


Figure 7.1: An example graph visualized in our web-based user interface: The components in the graph are cascadable. The ”AR Manual” component in the middle of the graph for example comprises further sub-components, which allows for internal parallelism.

extensions, *e.g.,* the Microsoft Accelerator concept [233], or OpenMP [234], where the parallel implementation is realized through merely declaring the data parallelism rather than explicitly creating and dispatching threads.

We have given careful thoughts of how to design a programming model that

- implicitly influences the developer to favor architectures that facilitate parallel execution,
- hides the complexity and the synchronization overhead of multi-threaded execution,
- is able to automatically parallelize and optimize parts of the provided algorithm,
- provides a strong separation between algorithm and implementation,
- facilitates the use of optional hardware like general purpose GPUs,
- can distribute code among heterogeneous physical machines.

The component paradigm, where code is organized between highly encapsulated entities that only declare their data interface, has several natural benefits for the purpose of parallelization. We require the developer to provide a component-graph that describes the data flow between single abstract component nodes. Figure 7.1 shows an example graph. The graph explicitly contains all data dependencies; either by ”wiring” a consumer to all associated data providers in the graph or by referencing constant default values. The component nodes in the graph are black-box placeholders for processing steps or algorithms. Even without knowledge about the actual implementations, this abstract definition already allows to statically distribute

parallel sub-graphs (*i.e.,* data-independent) among threads and physical machines. The component nodes only represent algorithms or even classes of algorithms that share a common data signature to which several implementations can be registered. This can be hardware-specific, like a CPU vs. a GPU implementation, but also specific to a problem size, *e.g.,* one that is fast for small-scale data, and one that suits large amounts of data. By delaying the association of an algorithm with a concrete implementation until after the system knows the exact target hardware, we are able to choose an optimal implementation for the given platform. Through a profile-guided optimization, we can further specify the set of implementations that optimize the resulting program in terms of execution time, memory footprint, or bandwidth consumption.

The developer needs to contribute the abstract, data-driven, graph-based definition of the algorithm, together with implementations for the missing building blocks. We call the combination of the data signature (*i.e.,* set of necessary inputs and outputs) and an implementation a *module*. The data signature of the according algorithm is automatically computed as the union of data inputs and the intersection of outputs of all modules registered to it. Algorithm 3 shows an example of a concrete module implementation.

Algorithm 3 Definition of an example module. The `ARGOS_WRAPx` macro builds a wrapper around the implementing function and registers it as an implementation of the according algorithm.

```
void Dilate(const argos::Image& input, argos::Image& output) {
    output.ensureFormat(input);
    cvDilate(input, output, 1);
} ARGOS_WRAP2(Dilate, const argos::Image, argos::Image, image; image)
```

Within this function, the module has exclusive writing rights on its output buffers and granted, synchronized reading rights on the input buffers. Additionally, the framework handles the memory management, creating multiple simultaneous buffers in the background as well as the related garbage collection. This vastly alleviates the implementation of multi-threaded applications, since there are no additional constraints compared to single threaded implementations within each module.

7.2 Scheduling and optimization

The set of associated implementations and the concrete, static dispatching to threads and physical machines is called a *schedule*. To create a schedule, the system pursues three strategies:

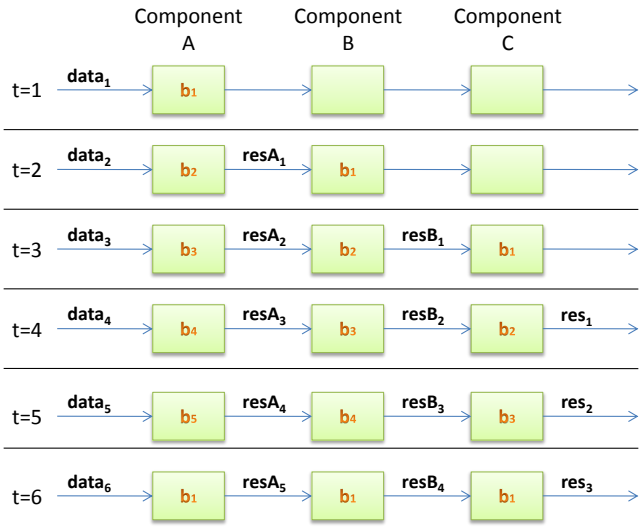


Figure 7.2: Illustration of pipelining, *i.e.,* stage-parallel execution: As soon as a component has finished processing a data item, the result is propagated and the component immediately continues with the next item.

- Choosing an optimal set of implementations for a given target hardware and a problem size.
- Graph-parallel execution, *i.e.,* dispatching sub-graphs to different threads and physical machines. The goal is to partition the graph into balanced fragments in terms of computational load with minimal bandwidth footprint in-between.
- Pipelining or stage-parallel execution, *i.e.,* accelerating sequential sub-graphs in presence of streaming data, see Figure 7.2 for an illustration. This technique, which is also used by most modern processors, allows to exploit hardware parallelism for algorithms only comprising sequential steps.

A profile-guided schedule requires information about the computational and bandwidth footprint of each module. While there are approaches using simulated performance data, here, the measurements are acquired through repeated instrumented runs in different configurations. The parameter search space for optimization is hereby reduced trough only examining modules above a certain computational footprint. Without a profiling run, this means only evaluating static graph analysis, the optimization potential is limited to exploiting graph- and stage-parallelism. This process can be performed in less than a second and is therefore always con-

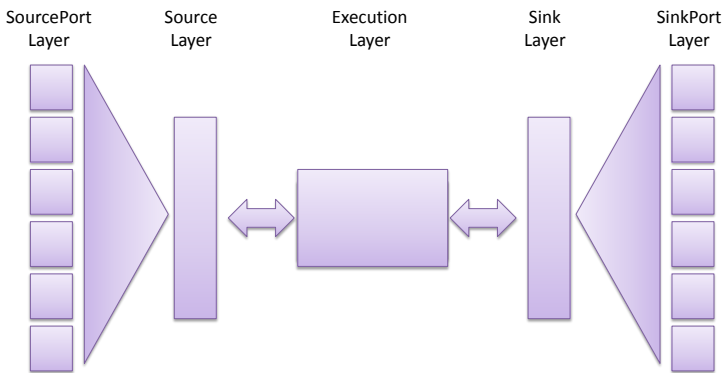


Figure 7.3: The internal structure of a module using 5 distinct layers.

ducted before executing a graph on a target platform. The abstract graph representation itself is invariant to a schedule and does not need to be changed or adapted for different platforms.

In the following two subsections, we will describe how we realize the automatic parallelization and schedules for heterogeneous target platforms comprising mobile devices.

7.2.1 Module parallelization

To allow for an automatic optimization of the resulting applications, most importantly parallelizing the execution, we require a thread-safe data handover between modules. As each implementation is hereby simultaneously invoked on different data items, this requires separating the implementation from the data buffers. A module is comprised of five different layers, see Figure 7.3. The *SourcePort* and *SinkPort* layers conduct the actual thread-safe handover. The *Source* and *Sink* layers accumulate the data from the connected ports and communicate with the so-called *execution* layer. The execution layer contains the actual implementation and is the only part that has to be provided by the developer, as seen in the code example in Algorithm 3.

In order to parallelize sequential branches of the graph in presence of streaming data, the framework uses thread-safe buffers maintained in the SinkPorts. Figure 7.4 illustrates a simplified sketch of the asynchronous data handover between two modules. Once a module is triggered, *i.e.,* all data that is necessary to run the implemented algorithm is available, the framework reserves exclusive output buffers for the run, sets the necessary read-locks, and invokes the implementation in the execution layer.

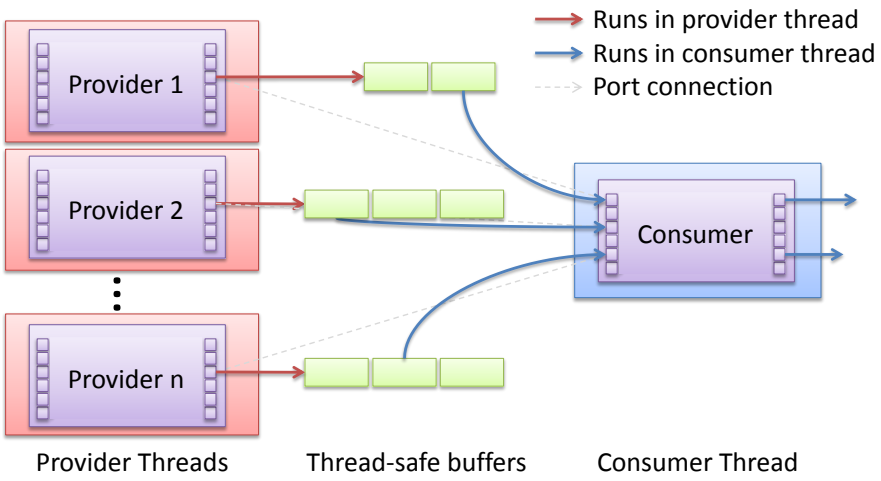


Figure 7.4: Asynchronous data handover between modules.

7.2.2 Schedules comprising mobile devices

Since mobile devices like smartphones, tablets, and most recently wearables like Google Glass are an important output device for AR task assistance, our implementation needs to accommodate the lower computational profile of these devices. In general, there are two possible approaches:

Remote execution: Using remote data processing which suffers from limited bandwidth and communication latencies and is therefore difficult to implement in real-time interactive applications.

Simplification: Adopting a faster but less powerful approach which results in a loss of accuracy or robustness.

Since our proposed framework affords the simple exchange of parts of the implementation and allows inserting network bridges at arbitrary points within the graph, we can very effectively investigate those two approaches.

Reducing the mobile computational load by remote execution is not a new strategy: The authors of [235] distinguish different subclasses of this approach like *primary functionality outsourcing*, *i.e.,* retaining simple components on the client and offloading computationally complex ones, or *background augmentation*, *i.e.,* offloading of a huge one-time task. Focusing on image processing, [236] differentiate between several client/server interaction types like

offloading of pose estimation, offloading of both pose estimation and classification or a thin-client.

Early work in mobile AR with remote execution includes [237, 238] both using the client solely as image source (thin-client) and performing all processing steps on the server. With the improvement of mobile hardware it became feasible to involve the client in the computation to reduce network load and overall processing time. The client in the system of [239] uses object tracking to minimize the number of requests to the object recognition server. Kumar *et al.*[240] propose a client, performing image tracking as well as feature extraction before sending a request but they do not aim for interactive frame rates.

Several frameworks have been proposed for enhancing mobile implementations by remote execution. CloneCloud [241] enables offloading by virtualization of the smartphone's operating system on a server. The client starts offloading by transmitting its complete processor state onto the remote system and receives the state resulting from the computations performed by the server. This enables switching between onboard and remote execution at any particular point in time.

In contrast to CloneCloud, μCloud [242] uses software decomposition. Viewing the whole application as a graph of black box components every node is weighted with its consumed time obtained during a previous run-time analysis. This graph is then split between client and server. However, in their proof-of-concept implementation they used the mobile client simply as an image source, computing all other steps exclusively in the cloud.

We will quantitatively analyze different workload balances ranging from extensive remote execution to pure onboard processing within our evaluation section. Whether remote execution improves the processing speed depends on the actual application and the execution context (mobile device, network quality, etc.). The performance behavior is therefore systematically analyzed under different network qualities and device capabilities.

7.3 Evaluation

Within this section, we provide an extensive performance evaluation of our approach in various configurations, also in combination with mobile devices.

7.3.1 Performance Evaluation

To evaluate the run-time behavior, we will first investigate the impact on run-time performance, when using our framework. After this analysis, we will present performance measures and profiling data for the aspects temporal segmentation, classifier training, and tracking.

7.3.1.1 Optimization gain

To give an impression of possible speed ups we evaluated morphological closing as a simple example case, *i.e.,* a sequence of dilation and erosion operations. We chose a total of four (two erode plus two dilate) steps to provide enough stages to saturate a quad-core processor. Additionally, we examine different implementations to demonstrate the importance of incorporating the choice of implementation in the scheduling strategy for a given problem size.

As baseline, we provided a simple sequential implementation using OpenCV [243]:

```
/* ... Creating input images ... */
for(int i=0; i<2000; i++) {
    cvDilate(input, dummy, 1);
    cvErode(dummy, output, 1);
    cvDilate(output, dummy, 1);
    cvErode(dummy, output, 1);
}
/* ... measuring performance */
```

We build the equivalent pipeline:

```
argos::ModuleGraph mg;
mg.add("Image Reader") >>
mg.add("Dilate") >> mg.add("Erode") >> mg.add("Dilate")
                                          >> mg.add("Erode");
mg.start();
```

The algorithm `Image Reader` hereby provides a stream of altering images. The reason for this is that creating a single image and feeding it repeatedly into the pipeline would be statically resolved by the system: The pipeline would then only be executed a single time, as repeating inputs to modules without internal state leads to simply repeating the last outputs without actually re-running the implementation. Aiming for a fair performance measurement, we have

worked around this optimization mechanism. As the framework possesses an integrated performance measurement and profiling, no further code is necessary.

We registered two implementations for the algorithms `Erode` and `Dilate`, each. One being a (trivial) OpenCV wrapper, analogous to Algorithm 3. One being a CUDA implementation adapted from the CUDA SDK material. The results of the experiment are shown in Figure 7.5.

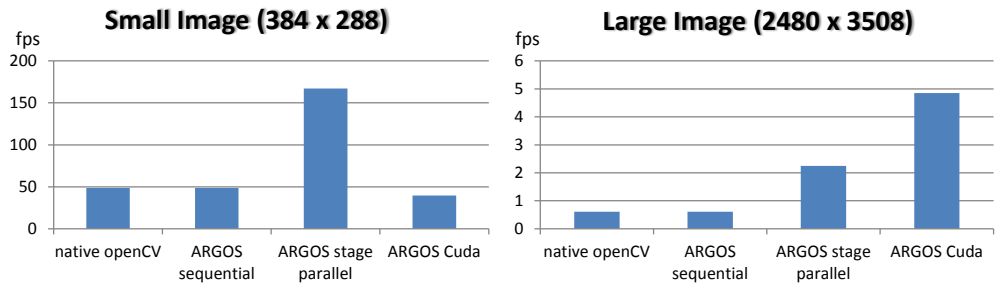


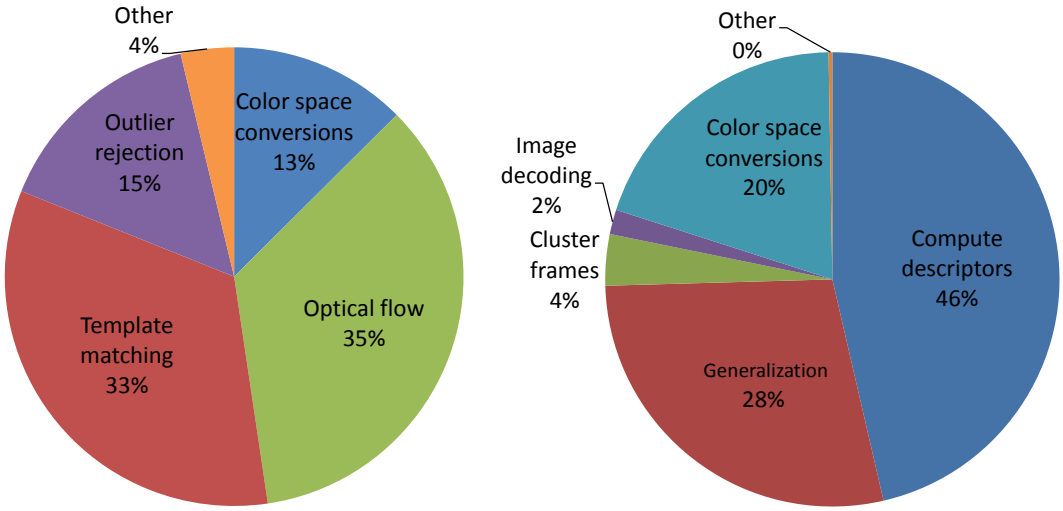
Figure 7.5: Performance comparison for small (left) and large (right) input cardinality: The experiment was performed on an Intel Core2Quad 2.5 GHz with nVidia 9800 GTX.

Both for small and large input images the framework overhead is below the measurement accuracy. When the framework was allowed to create a stage-parallel schedule, the application performed at about 3.3 times the speed of the single threaded execution resp. at 80% of the theoretically achievable maximum on a quad core. Forcing the scheduler to associate the CUDA implementation to `Erode` and `Dilate`, the performance was slightly worse on small input cardinality.

On bigger images, the CUDA implementation could exceed the CPU implementation. This underscores the utility of a loose association between algorithm and implementation. The optimal choice of algorithm is depending on the target platform as well as on the nature of the data to process. The possibility to delay and change the actual scheduling transparently without changing the definition of the algorithm is a beneficial property that we will further investigate in Section 7.3.2.

7.3.1.2 Application performance

We will show that the speed-ups described above are also possible for real applications. We will now present performance data for the three main aspects of our approach, temporal segmentation (see Chapter 3), classifier training (see Chapter 4), and workflow tracking (see Chapters 4



(a) Segmentation

(b) Classifier training

Figure 7.6: Performance profiling results for the offline components.

and 5). Without changing the algorithm graph, we ran the applications once multi-threaded and once using only a single thread on a Core i7 Quad W3520 at 2.66 GHz.

	Multi-thread	**Single thread**	**Factor**
Temporal segmentation	22.5	10.0	2.3
Classifier training	2.9	0.8	3.7
Workflow tracking	24.7	6.4	3.8
Tracking w/o display	38.6	10.0	3.9

Table 7.1: Frames per second and achieved speed-up factors.

The achieved speed-up factors are shown in Table 7.1. While both classifier training and tracking achieve high speed-ups of about factor 3.7, the very sequential segmentation task still achieves a factor of 2.3 without any change of the underlying algorithm.

To further analyze the performance behavior, we have profiled each of the algorithms in more detail. Figure 7.6(a) shows the ratios for the segmentation approach. The two largest computational chunks are determining optical flow and template matching, each consuming about one third of the computation time. These two algorithms are provided as single nodes

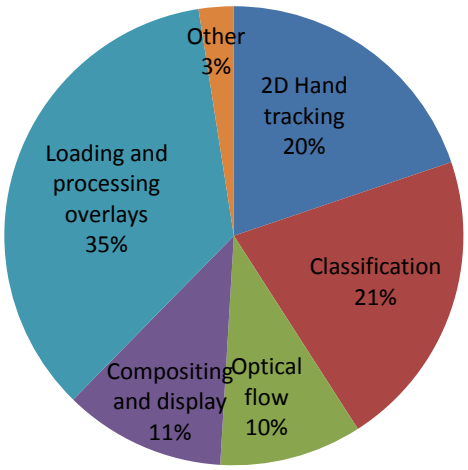


Figure 7.7: Performance profiling results for the tracking components.

within the graph and are internally single-threaded. This already caps the theoretically achievable speed-up factor to a value of 3, since two thirds of the computation time only can double in speed. By multi-threading, *i.e.,* sub-dividing these implementations, a higher performance is achievable.

The classifier training profile is shown in Figure 7.6(b). As expectable, the largest chunk is the descriptor computation with 46% of computation time. Since the comprised components can be fed with multiple frames simultaneously, this part is very well parallelizable.

Similar holds for the real-time workflow tracking, as the follow-up workflow states are evaluated by separate, parallelizable classifiers. The according results are shown in Figure 7.7. What is noteworthy, though, is that with over a third, the largest part of computation time is spent on loading and processing the video-based overlays and an additional 11% are spent on compositing and display. When providing the system as a back-end rendering node, these two parts would not be required and the system could achieve higher frame-rates. We measured this headless operation without any display functionality and received 38.6 frames multi-threaded and 10.0 frames single-threaded, respectively, which translates to a factor 1.5 speed-up. The detailed analysis of this mode of operation in combination with a mobile displaying device is presented in the following subsection.

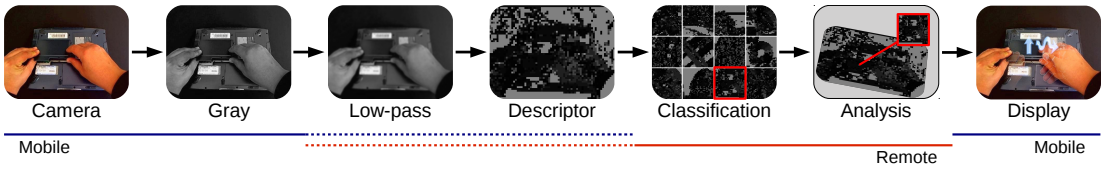


Figure 7.8: Illustration of the processing pipeline and its three possible handovers (gray, low-pass, gradients) for outsourcing computation to a remote server. The client then displays the visual instruction as a result.

7.3.2 Mobile devices and remote execution

We aim to investigate, when remote execution yields a benefit for our application in comparison to algorithm simplification, allowing to run an algorithm on the mobile device. To that end, we measure the time consumption with respect to different handovers between mobile device and remote server under varying device and network qualities.

Figure 7.8 shows a schematic overview of the workflow tracking approach, explained in Chapter 4. We provided equivalent implementations for all parts of the algorithm, except for the classification and analysis aspects. As these are the most time consuming aspects of the procedure, they would overburden the mobile devices in real-time and memory budget in unchanged form and are thus mandatory candidates for simplification.

Additionally, we provided two implementations for the UDP network bridge algorithm. One that compresses the data for every handover, using JPEG compression for images and run-length encoding for the DOT query descriptors and one that sends the data uncompressed, trading bandwidth for saved computation time. Hence, the system provides a total of six possible ways for subdividing the pipeline between client and remote system.

To implement the same approach using simplification, we exchange the (server-based) steps of classification, and analysis through a simplified mobile implementation based on bag of words classification using FAST+BRIEF features [172, 244]. The number of features was hereby limited to the best 50 keypoints. This prototypical implementation is solely intended to provide a repeatable frame of reference for comparative performance.

To have a controlled experimental setup, the tested devices were mounted onto a tripod facing a computer monitor, displaying a series of static images to classify (Figure 7.9).

We used a Samsung Galaxy S2 (SGS2) representing the class of faster mobile devices and a Nexus One (NX1) for the class of slower ones. The wireless LAN (WLAN) connection between client and server allowed a connection with around 40 Mbit/s and we measured a data

(a) Evaluation setup (b) Running application

Figure 7.9: Illustration of the synthetic evaluation setup and running application.

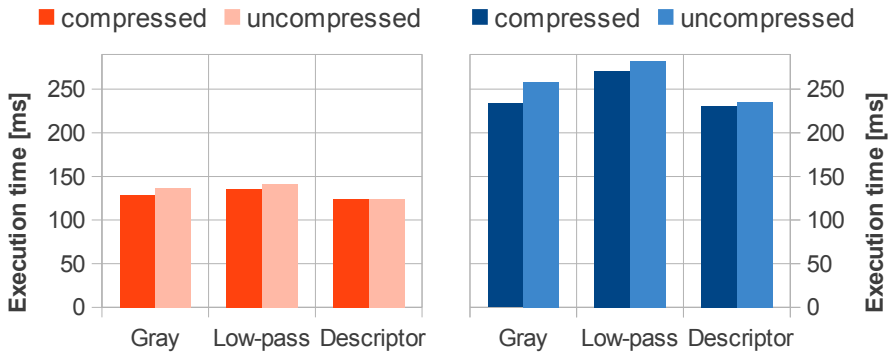
rate of around 0.3 Mbit/s for the mobile broadband (HSDPA). When not stated otherwise, the server received a 320×240 image and used 40 reference templates - one template per scene. The stated execution time is the time between the start of the scene classification and the arrival of the result averaged over a three to five minute run.

7.3.2.1 Analysis of the optimal handover position

To figure out which workload balance yields the best performance, we compare all six possible handovers: gray, low-pass, and descriptor, each with uncompressed and compressed bridge implementation. While the client's computational load rises with a late handover, the network load decreases. While the uncompressed size of the gray image is 75KB per frame, the size of the DOT query descriptor is only about 5% of that size.

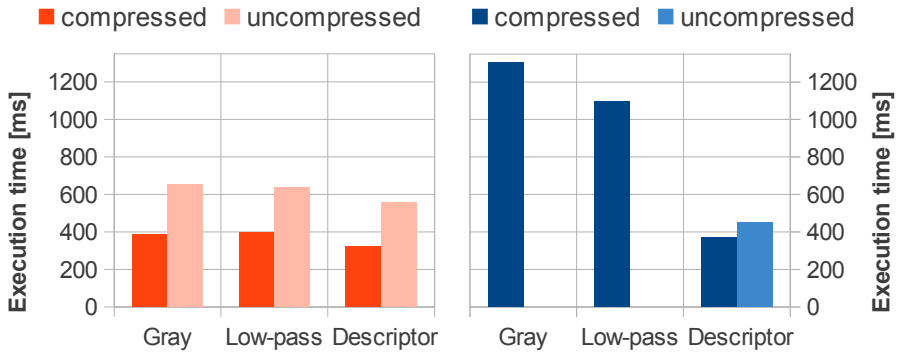
The effect of compression varies depending on the visual content. In our test cases we observed a data reduction by factor 5 for handover gray, factor 6 for handover low-pass (both lossy JPEG) and factor 3 for handover descriptor (lossless run-length encoding) compared to the respective uncompressed case.

The performance of the SGS2 in WLAN (Figure 7.10(a)) slightly increases with a late handover (descriptor) and benefits from compression. This behavior is more distinct with the NX1 (Figure 7.10(b)). This is surprising since one would assume that the higher workload and the additional compression would be disadvantageous for the slow device. However, the transmission time via the slower network interface seems to overcompensate the increased CPU load. The benefit of a late handover with compression is even more evident when using mobile broadband (Figure 7.11). However, choosing the low-pass as handover has no advantage. Since



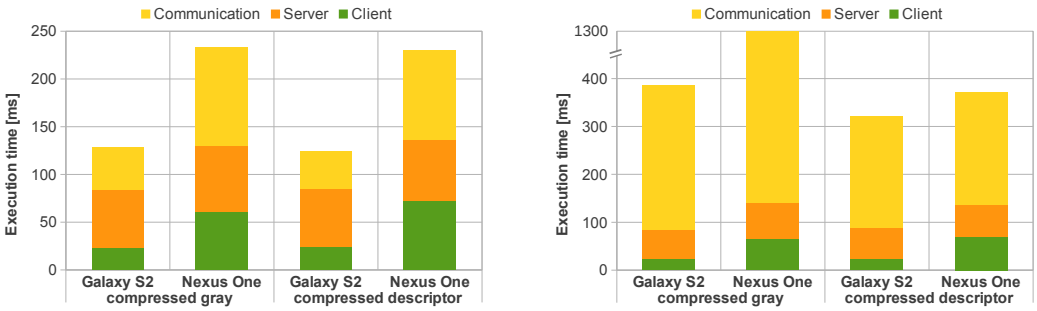
(a) Galaxy S2 - WLAN (b) Nexus One - WLAN

Figure 7.10: Evaluation of the six possible handovers using WLAN: The tests were conducted on two different mobile devices and show a slight advantage of compression in all cases. A late handover turns out to be the best choice.



(a) Galaxy S2 - mobile (b) Nexus One - mobile

Figure 7.11: Evaluation of the six possible handovers using mobile broadband: The tests were conducted on two different mobile devices and show a clear advantage of compression. A late handover (descriptor) turns out to be the best choice.



(a) WLAN (b) Mobile broadband

Figure 7.12: Breakdown of the thin client configuration (offloading the gray image) and the late handover (offloading the descriptor), both compressed: A higher mobile computational load comes with a decreased communication overhead which is especially advantageous when using mobile broadband.

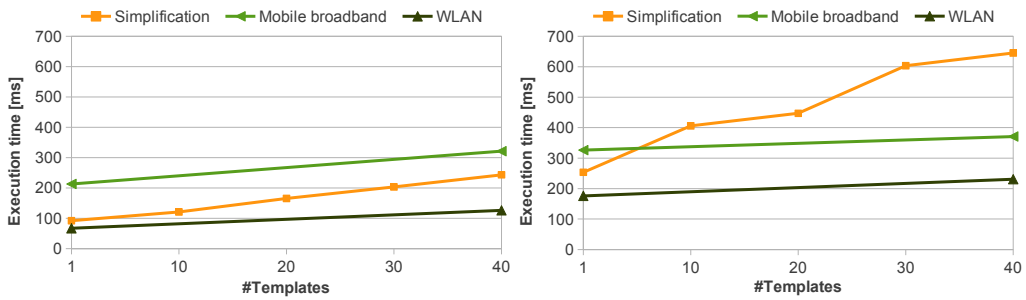
the smoothing kernel does not justify subsampling, the slightly higher JPEG compression rate of smoothed images is negligible when using WLAN.

To satisfy the real-time demand, we use UDP as transmission protocol, which means that lost datagrams are not sent again. Since the uncompressed gray image and the uncompressed blurred image have to be divided into many packets, the probability of one of those getting lost is very high leaving the server with an incomplete image. This occurred quite often when performing remote execution with the Nexus One via mobile broadband. Hence, we exclude those measurements in Figure 7.11(b) and conclude that compression also decreases the probability of datagram incompleteness.

Figure 7.12 illustrates the individual shares of the thin-client configuration (gray) and the late handover (descriptor), both compressed. It shows that a late handover comes with a higher computational load for the client but reduces communication load on the other hand. This client configuration is called ”non-trivial client” as opposed to a thin client and is particularly useful in mobile broadband.

7.3.2.2 Comparison of remote execution and simplification

Figure 7.13 illustrates the difference between simplification and remote execution. Remote execution was done with a late handover (compressed descriptor) since the previous experiments showed that this is the best choice. The measurements indicate that remote execution via WLAN is advantageous compared to running a simplified classification onboard.



(a) Galaxy S2 (b) Nexus One

Figure 7.13: Comparison of simplification (orange) and remote execution with compressed descriptors via mobile broadband (green) and WLAN (dark green): Remote execution turns out to be particularly useful for slower devices (right), strongly outperforming simplification. Simplification is preferable via mobile broadband, given that the amount of reference templates does not exceed a certain size.

Remote execution in mobile broadband becomes profitable when the template set exceeds a certain limit. This break-even point occurs very soon for the NX1 (less than 10 templates). For the SGS2, this will only occur with a considerably larger data set, roughly around 100 templates.

Our analysis of remote execution in the application context showed that offloading complex computations can indeed result in a higher processing speed. Reducing the network overhead increases this effect, which means that a simple thin-client configuration is typically not the best configuration. In our specific application, splitting the pipeline at a rather late point in time yields the highest performance increase.

Thus, we propose remote execution with non-trivial clients as an alternative to simplification when adopting computationally complex programs to mobile devices. This also has the advantage that already existing template databases can still be used and do not have to be recomputed for the simplified algorithm. Moreover, the remote system's database and even its implementation can easily be changed without the user having to update the application.

8

Conclusions

In the following sections, we will summarize our work and the conclusions drawn from it. This includes the identification of open research questions and directions for future work. In Section 8.2, we will briefly present the three short-term topics that are currently ongoing or planned.

8.1 Summary

We have presented the first comprehensive approach for the automatic creation of procedural task assistance using Augmented Reality. Over more than twenty years since the general concept's proposal [1], the technical approach has remained almost unchanged. There have been partial approaches for specific domains or with rigorous technical constraints and scenario prerequisites. However, this is the first work achieving the workflow structure assessment, content creation and during runtime the user monitoring and assistance with a markerless and domain-agnostic approach. Especially the contribution of this work regarding the first two aspects is cardinal, since it strongly alleviates the content creation issue and even conclusively solves it for a range of use cases. As the content creation problem is one of the major inhibitors for a wide-spread uptake of AR assistance systems in commercial markets, we expect a great commercial and scientific impact of our work.

During run-time, our approach allows to closely follow the user through the course of an activity sequence. For procedural assistance, this allows guiding the user during the psychomotor phase and providing feedback regarding the correctness of conduction. This has been achieved

for the first time without the use of optical tracking aids. In the following, we summarize and discuss the main technical contributions of our approach.

Segmentation: We have presented an online, live-stream capable task segmentation approach for video examples of manual workflows. The method provides the underlying workflow structure, which corresponds well with human interpretation. We can use this segmentation to automatically create a pictographic documentation of a workflow as well as to author animated overlays as part of Augmented Reality manuals.

In contrast to the typical approach of using high-level features such as recognized objects, we investigated the feasibility of whole-image properties that do not require a prior identification of image segments or any other semantics. Since the formulation is based on the learning rate of a nearest neighbor classifier, a beneficial side effect is that the segment boundaries facilitate k-NN classification during run-time.

The approach works very well on workflows that were recorded with a fixed camera. In this case, about 80% of the actions were segmented in strict compliance to what a human would manually determine as segment boundaries. It is important to note that no segmentation or any other prior identification of image parts needs to be carried out to achieve this performance (other than cropping the image by a fixed margin). When applied to recordings from a head-worn, moving camera, the approach has a higher failure potential. This is mainly due to parallax movement of the environment that cannot be distinguished from user activity by the proposed approach. Thus, in its current state, the approach is most effective on (mostly) stationary cameras. A dedicated recognition and handling of parallax motion is a promising candidate to improve the segmentation precision in the case of strongly moving cameras.

Beyond the use case of procedural assistance, there are further applications of the method. Since the clustering approach is real-time capable and can operate on online, live video input, it could be used as a preprocessing step for various real-time computer vision problems. Most prominently, it could be used to implement the idea of context-priors for problems with a prohibitively large search-space. We will further express this thought at the end of this section.

Modeling and markerless tracking of workflows: Using a set of classifiers trained on the segmented recording, we have presented an approach to precisely follow a user while

executing this workflow in real-time. This can be used to interactively display instructions, real-time feedback, and side-information without requiring the user to manually step through the single segments. To the best of our knowledge, this is the first time that this has been successfully implemented for AR and procedural assistance without the use of markers, special sensors, or additional tracking aids. Three major contributions lead to this. Besides hand tracking that will be discussed subsequently, these are:

- A novel representation of time-progressing 3D environments. Due to the fact that the camera motion during recording is unconstrained and not guaranteed to exhibit sufficient translation to estimate geometry, typical approaches like structure from motion or SLAM are infeasible for this application. Furthermore, all approaches that require prior knowledge (CAD-models) or special infrastructure (markers) were also deliberately excluded.

  Instead, we have proposed a piecewise homographic transform that we call *relevance plane 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 specific region of interest. This region gets determined for each temporal segment, independently, either through estimating the focus of attention or the focus of interaction using the hand tracking information. The transform then results in a piecewise two-dimensional spatiotemporal model of dynamic, changing environments that elegantly handles cases of incomplete observation.

  As the resulting 2D frames are spatially continuous, it is viable to directly apply 2D descriptors or to anchor 2D information associated spatially as well as temporally to the time-evolving 3D workspace. Through a robust two-step backprojection procedure, the descriptors or overlays can be robustly applied to a new recording or live stream. The limitation of the procedure is that the viewpoint during run-time is required to be similar to the one from the reference sequence. However, the extracted instructive video snippets would also be compromised in expressibility or even validity, when overlaid onto the workspace seen from a different angle. Therefore, the limitation is inherent to our approach in general. We have demonstrated the applicability by sampling 2D probability maps of the hand location from a moving camera that were used for classification and for providing live feedback.

- An image-based approach for the model-guided generalization of training data. As the originally resulting tracking model is quite user dependent, we have pursued several ideas to address this. We first showed how to include several reference examples into the same tracking model that allows accommodating user-related variations and differences in the point of view. Furthermore, this allows assessing structural variants of the workflow itself.

  Eventually, to allow the creation of a tracking model from a single reference video, we proposed an approach using model-guided generalization based on image-based rendering and hand tracking. On our data sets, provided by different users, the correct recognition rate on single frames was more than tenfold higher, when using this generalization scheme. We further showed how to seamlessly include this scheme into our framework through explicitly retouching the training data.

Hand tracking: In order to track the user's hands during the workflow, we have proposed a novel learning-based approach. The method combines a generative approach using an image-based appearance model and a discriminative approach based on queries in very large databases of hand views. The proposed database allows real-time queries for a rather densely sampled subset of the parameter space with millions of entries. This is possible due to a hierarchical structure that corresponds to the relative average joint velocities, in order to facilitate fast local beam-searches. Through exploiting associative symmetries, it allows the quick exchange of entries, which is the prerequisite to adapt to the observed content.

In order to populate the database, we have presented an image-based rendering approach. This method allows interpolating between observed prototype views by means of an extremely light-weight axis-aligned morphing scheme. Due to its efficiency, it is also feasible as hand appearance model for real-time hand tracking, in particular for refining results in presence of nearest neighbor hypotheses. Most importantly, this appearance model allows the formulation of a pixel-wise objective function that vastly outperforms skin color and edge based methods regarding robustness and the number of local optima. Our proposed objective function is provably very robust towards various challenging conditions including cluttered background, skin-colored background, skin-colored occlusions and strong blur. We have demonstrated that our method significantly outperforms the state of the art in hand tracking with a generative model. We could prove this

through unveiling systematic flaws in the most commonly used mathematical terms of the previous objective functions. Additionally, we have validated our findings through comparison with a concrete state of the art method [194].

To the best knowledge of the author, the resulting hand tracking approach is the only approach capable of tracking such challenging material as was used for evaluation without using markers and only using a monocular RGB camera. One could argue that the more fragile step of sampling the necessary prototypes compromises the validity of these findings. However, in a real application this could as well be handled by a short, dedicated preparation procedure, where the user gets prompted to perform a certain number of hand postures to robustly bootstrap the process.

We could further show that the generalization procedure that is key for the database population strategy has a paramount impact on the recognition rate. This eventually allows the inference of working tracking models from a single recording of the workflow.

Presentation and application: We have described our approach of extracting descriptive illustrations for each action from the provided reference sequence, automatically.

When displaying information during run-time, we exploit the ability to precisely track the user's progress to scope the visibility of each overlay. Additionally, we visualize several automatically assessed correctness indicators. Firstly, the system provides enactive feedback during the psychomotor phase. Secondly, it provides feedback after each work step by performing an optical validation of the step's outcome by comparing it to the desired target state from the reference material.

In order to allow augmenting the scene with further information, we have also presented a graphical user interface for manual authoring. Most notably, this interface entirely hides all 3D aspects in order to allow a domain expert, who is generally not knowledgeable in 3D graphics to operate the interface.

Implementation: We have briefly presented a component-based, data-driven programming model to design, implement, and execute algorithms on possibly heterogeneous hardware. The programming model particularly suits the requirements of computer vision and image processing by efficiently supporting the stream processing nature of the underlying applications. Thereby, the formulation allows the automatic parallelization of

the resulting algorithms. By abstracting from implementation details, the algorithms can quickly be adapted to accommodate specific target hardware.

We showed that the system is effectively able to achieve speed-ups, both in synthetic scenarios and for the actual application described in this work. Additionally, we have systematically evaluated different workload configurations between a remote system and a mobile client. We found that optimizing for maximum offload (thin-client configuration) is at a disadvantage compared to splitting the workload between remote system and client, even with a fast network connection. One of the major benefits of our approach is that it affords altering execution schedules without requiring any change of the user-provided program definition. This allows accommodating wide-ranging mobile hardware that alters the optimal workload configuration through an automatic process.

The presented work provides a fully functional technical approach to acquiring and transferring workflow knowledge using Augmented Reality. Outside of the immediate scope of procedural assistance, the proposed closed loop between sampling, modeling, and recognition of user context has further applications. Technically, this could be used to condition fragile processing steps on a prior context recognition. An example, where this has already been realized within this work is our proposed method for hand tracking that implicitly learns context-driven posture priors through a non-parametric model.

From a conceptual standpoint, this allows to generalize the central idea of Augmented Reality. While the classical definition is merely based on the spatial association between virtual information and real objects [8], we have presented a structured way of associating information with context, additionally.

8.2 Future work

Although the material presented in this thesis is self-contained, it opens up a large field of directions for future work. We will present three topics that we have already started to explore within the following subsections.

8.2.1 Study of performance indicators and human factors

While this work is concluded with the technical foundation for the automatic creation and provision of procedural assistance, there are many open questions regarding the acceptance

and the efficiency of the resulting system.

In an already published follow-up study of our work [127], we have validated the correctness of our premise to base the generation of instructions on prior event segmentation. The conducted study shows that the selection of temporal segments for creating instructions corresponds to a goal-agnostic segmentation of event boundaries.

In this regard, we intend to further study how understandable the automatically created visual instructions and their assigned temporal scope are for a human recipient. We expect that the automatic annotations effectively decrease the required time for understanding an instructed action, compared to watching an unprocessed reference sequence, directly. In case of a pictographic documentation, we even expect to increase the understandability of the pictorial representation. Further, we want to compare this to the effects of manually created annotations in the same setup.

Of particular interest is the impact of the proposed enactive feedback on objective and subjective factors. We expect the proposed feedback to possess a beneficial effect on the correctness of the conduction and, eventually, a lower error rate. More importantly, the feedback indicates the ongoing support from the system and will expectedly have a positive effect on the user's perceived quality of the assistance system.

Another important practical implication of these further studies is to help us to understand how to adapt the presentation to the user's mental state, for example, a state of confusion. While the purely technical "detection" of a state of confusion is one aspect, the more challenging and important question, however, is how to appropriately react to this state. Also generally, due to the fine-grained observation of the user's actions, we are able to scope and adapt the information that is provided by the system to the user and therefore need to understand the user's informational needs. For assembly, [245] identifies the task variables that influence perceived object assembly complexity. Their study investigates objective and quantifiable indicators (number of fastenings, component groups, or novel assemblies) that influence the perceived complexity. These indicators could be very useful to adapt the information offer by system to the subjective information need of the user.

8.2.2 Deriving procedural knowledge with hand tracking

An important technical direction of future work is to exploit the estimated hand and finger trajectories, in order to discover the modalities of the execution. We exemplify this on two concrete properties that can be directly estimated from the data:

Figure 8.1: Heat maps of thumb and index finger positions collected from a single sequence (left), averaged over three sequences (middle), and six sequences (right).

Grasp point detection: There is an observable reciprocal correlation between the required level of accuracy and the speed of execution when interacting with objects and environment. For example, when removing a plug, the relative velocity of reaching to the plug is typically higher than the velocity of grasping the plug to remove it from its socket. The same also holds for the subsequent retraction of the hand. Generally, relative velocity is one of the major statistical cues for interpreting manual activities [246] and we therefore have already incorporated it in the segmentation step described in Section 3.1.3. In contrast to the low-level observation features that we have used for segmentation, we are now able to extend this approach using the detailed hand trajectories.

We only use the positions of the index and thumb fingertips as surrogates for the hand interaction for several reasons. Firstly, while there exist several types of typical grips (*e.g.,* power grip, precision grip), almost all of them involve the index and thumb as the touching extremities. One easy example to illustrate why it is beneficial to restrict the observation to only the touching parts is the movement of the hand when trying to loosen up a plug. This often involves a kind of ”wobbling” hand movement. Since the touching extremities get fixated by the plug itself, the fingers are moving significantly less than the rest of the hand. Secondly and more importantly, the thumb and the index are the most consistently visible fingers when observed from a first-person viewpoint.

Figure 8.2: Heat maps of thumb and index finger trajectories collected from a single sequence (left), accumulated over three sequences (middle), and six sequences (right).

We estimate the grasp points by accumulating 2D positions of the index and thumb positions, projected into the common frame of the work step. We hereby process the thumb and index finger, jointly. This means that for each frame, we add two positions and therefore, for a segment with N frames $2N$ positions. We determine the grasp position using the first statistical moments, the mean and the covariance of the accumulated series. The mean then delivers the grasp center and the covariance matrix provides a robust statistical measure for the grasp size and orientation. These can be already estimated from a single sequence, examples of the resulting heat maps are shown in Figure 8.1.

Essential procedure: When several recordings of the same workflow are available, we are able to distinguish relevant from erratic activities simply by comparing whether an activity is present in all reference examples. Using the estimated hand trajectories, we are additionally able to identify the parts of each work step that are seemingly vital to reaching the goal.

The main approach is very similar to how we determine the grasp position in the previous subsection. Though, since we are not interested in relative frequency but aim to accumu-

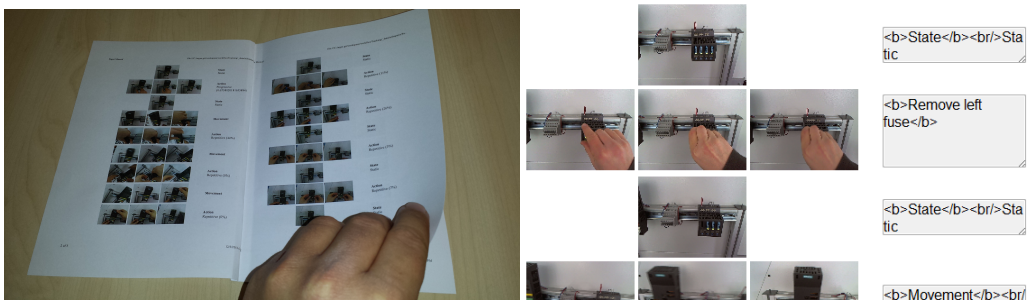


Figure 8.3: The automatically created pictographic paper manual.

late the fingertip coverage among several reference examples, we need to quantize and interpolate before calculating the statistical moments. Therefore, we also add linearly interpolated finger positions and the projected 2D positions are quantized into 2D bins. Each bin value is hereby incremented not more than once per sequence. Therefore, the bin value counts the number of sequences that project at least one point into the bin. The bins with a value equal (or close to) the number of examined sequences correspond to the motion that is seemingly essential due to the reoccurrence over all sequences. Again, the resulting example heat maps are shown in Figure 8.2.

8.2.3 Integration with paper-based workflows

As already mentioned, we can also automatically generate a pictographic paper manual, in addition to the interactive presentation. Figure 8.3 shows an example of the resulting documents. While the interactive representation clearly has several benefits towards the traditional paper-based documentation, it is not universally superior. In general, all user interfaces have inherent limitations in their affordances. While AR is great for intuitively associating virtual information with real, physical objects, it is a suboptimal or even clumsy interface for adding written or sketched annotations. Additionally, companies typically have a stock of documentation realized in paper form.

Quite recently, we have presented an approach called *Continuous Natural User Interface* (CNUI) [68], that intuitively connects physically distinct devices and in particular AR and a paper-based representation. Figure 8.4 shows the domain-continuity cycle described in the original publication.

The user can extract virtual pieces of paper from physical parts of the environment using a natural grab-and-pull hand gesture. The employed gesture resembles the real-world manipula-

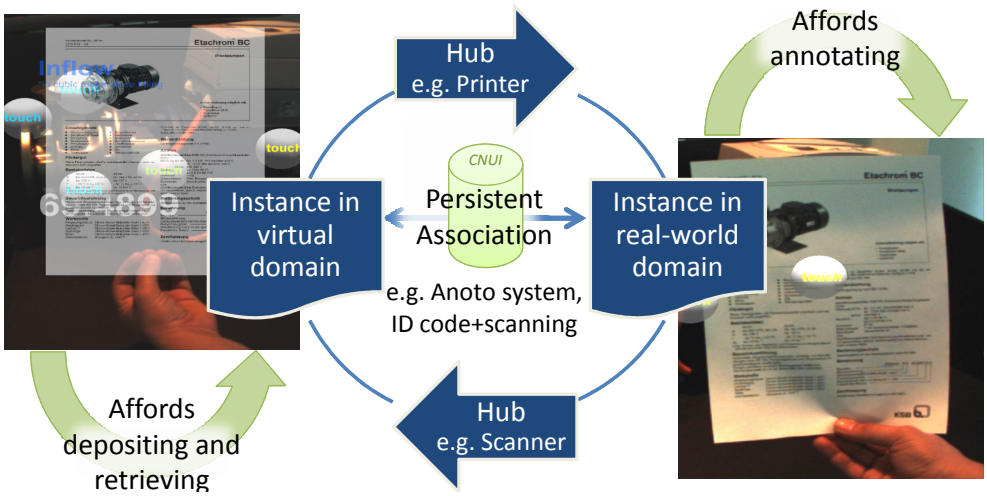


Figure 8.4: Illustration of a CNUI workflow cycle: This shows, how the user interface is continued between a virtual and real-world domain and makes use of their respective affordances: The two instances stay associated through the use of an Anoto system in this example. Taken from our previous publication [68].

Figure 8.5: Illustration of domain and interaction continuity between virtual (left) and real-world paper (right) in our demonstrator: Through an Anoto-pattern added during printing all hand-written annotations on the real paper are reflected on the virtual instance. Taken from our previous publication [68].

tion of paper, leading to a consistent look-and-feel between AR and paper, Figure 8.5 illustrates this consistency. Paper is a prominent example of a natural interface [247], as it offers a natural way, *e.g.,* for receiving handwritten annotations. To give the user the impression, that paper and AR are instances of the same continuous user interface, they need to be permanently associated with each other. This is realized using an Anoto pattern [248] that enables a corresponding optical pen to identify this piece of paper and precisely locate the pen's position on the paper during hand writing. All written annotations can thus be reassigned to the virtual instance, still deposited at the real-world object. Since the pen is equipped with wireless communication, this reassignment is instantaneous. This reinforces the impression of dealing with just another instance of the same content.

Within this line of future work, we will pursue a combination of the procedural task assistance as presented within this work with CNUI. The aim hereby is to crosswise augment interactive and paper-based representations in order to bridge between current and future means of documentation.

List of Figures

List of Tables

Glossary

AAM Active appearance model

AR Augmented Reality

BRIEF Method for point descriptor matching [172]

CAD Computer-aided design

CAI Computer-aided instructions

DoF Degree of freedom

DOT Method for region template matching [9]

DTW Dynamic time warping

FAST Method for interest point selection [244]

HMD Head-mounted display

HMM Hidden markov model

IBR Image-based rendering

Inpainting (Automatic) reconstructing of image areas

KLT Kanade-Lukas-Tomasi feature tracker [207, 249] - Point tracking method based on taylor series expansion within a small patch around each point

PSO Particle swarm optimization

Psychomotor phase Beginning of the actual execution of a work step, phase when motor skills are used

RANSAC Random sampling and consensus

RGB Red, green, blue

RGBD Red, green, blue, depth

RP Relevance plane

RPT Relevance plane transform

SAR Spatial Augmented Reality

SLAM Simultaneous localization and mapping

SSM Self-similarity matrix

References

[1] THOMAS P. CAUDELL AND DAVID W. MIZELL. **Augmented reality: an application of heads-up display technology to manual manufacturing processes**. In *the Proceedings of the Hawaii International Conference on System Sciences*, 1992. — *1, 17, 20, 159*

[2] K. M. BAIRD AND W. BARFIELD. **Evaluating the effectiveness of augmented reality displays for a manual assembly task**. *Virtual Reality*, **4**(4):250–259, December 1999. — *1, 3, 21, 23*

[3] ARTHUR TANG, CHARLES OWEN, FRANK BIOCCA, AND WEIMIN MOU. **Experimental Evaluation of Augmented Reality in Object Assembly task**. In *the Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR)*, 2002. — *1, 3*

[4] STEVEN HENDERSON AND STEVEN FEINER. **Exploring the Benefits of Augmented Reality Documentation for Maintenance and Repair.** *Transactions on Visualization and Computer Graphics*, **17**(10):1355–1368, October 2011. — *1, 3, 21, 22, 23, 130, 131*

[5] GOOGLE. **Project Glass**. http://www.google.com/glass/start/what-it-does/, Last visited on January 2014. — *2, 18*

[6] VUZIX. **STAR 1200**. http://www.vuzix.com/augmented-reality/products_star1200/, Last visited on June 2013. — *2*

[7] TAKEO KANADE AND MARTIAL HEBERT. **First-Person Vision**. *Proceedings of the IEEE*, **100**(8):2442–2453, August 2012. — *5*

[8] RONALD T. AZUMA. **A survey of augmented reality**. *Presence: Teleoperators and Virtual Environments*, **6**(4):355–385, 1997. — *6, 22, 164*

[9] STEFAN HINTERSTOISSER, VINCENT LEPETIT, SLOBODAN ILIC, PASCAL FUA, AND NASSIR NAVAB. **Dominant orientation templates for real-time detection of texture-less objects**. In *the Proceedings of Computer Vision and Pattern Recognition (CVPR)*, 2010. — *6, 31, 32, 33, 35, 43, 97, 100, 102, 105, 177*

[10] RENAUD OTT, DANIEL THALMANN, AND FRÉDÉRIC VEXO. **Haptic feedback in mixed-reality environment**. *The Visual Computer*, **23**:843–849, 2007. — *17*

[11] BORKO FURTH. *Handbook of Augmented Reality*. Springer, New York, 2011. — *17*

[12] VUZIX. **Vuzix Smart Glasses M100 - Hands Free Smartphone Display**. http://www.vuzix.com/consumer/ products_m100/, Last visited on January 2014. — *18*

[13] **ARVIKA**. http://www.arvika.de/www/e/home/home.htm, Last visited on January 2014. — *18*

[14] JENS WEIDENHAUSEN, CHRISTIAN KNÖPFLE, AND DIDIER STRICKER. **Lessons learned on the way to industrial augmented reality applications, a retrospective on ARVIKA**. *Computers & Graphics*, **27**(6):887–891, 2003. — *18*

[15] **ARTESAS**. http://www.artesas.de, Last visited on January 2014. — *18*

[16] **AVILUSplus**. http://www.avilusplus.de, Last visited on January 2014. — *18*

[17] WERNER SCHREIBER AND PETER ZIMMERMANN, editors. *Virtuelle Techniken im industriellen Umfeld: Das AVILUS-Projekt – Technologien und Anwendungen*. Springer, 2011. — *18*

[18] THOMAS ALT, WERNER SCHREIBER, WOLFGANG WOHLGEMUTH, AND PETER ZIMMERMANN. **Das Verbundprojekt AVILUS**. In WERNER SCHREIBER AND PETER ZIMMERMANN, editors, *Virtuelle Techniken im industriellen Umfeld*. Springer, 2011. — *18*

[19] **COGNITO**. http://www.ict-cognito.org, Last visited on January 2014. — *18, 20*

[20] **SKILLS**. http://www.skills-ip.eu, Last visited on January 2014. — *18*

[21] **Software-Cluster**. http://www.software-cluster.com/en/, Last visited on January 2014. — *18*

[22] WIKITUDE. **Wikitude**. http://www.wikitude.com, Last visited on January 2014. — *18*

[23] LAYAR. **Layar**. http://www.layar.com/what-is-layar/, Last visited on January 2014. — *18*

[24] METAIO. **Junaio**. http://www.junaio.com, Last visited on January 2014. — *18*

[25] MICHAEL WIGGINS. **Mobile Augmented Reality: Entertainment, LBS & Retail Strategies 2012-2017**. Technical report, Juniper Research, September 2012. — *18*

[26] BERND SCHWALD AND BLANDINE DE LAVAL. **An augmented reality system for training and assistance to maintenance in the industrial context**. *Journal of WSCG*, 2003. — *19*

[27] KATHARINA MURA, DOMINIC GORECKY, AND GERRIT MEIXNER. **Involving Users in the Design of Augmented Reality-Based Assistance in Industrial Assembly Tasks**. In *Applied Human Factors and Ergonomics*, 2012. — *19, 22*

[28] GABRIELE BLESER, DIMA DAMEN, ARDHENDU BEHERA, KATHARINA MURA, MARKUS MIEZAL, ANDREE GEE, GUSTAF HENDEBY, NILS PETERSEN, GUSTAVO MACC˜AES, HUGO DOMINGUES, DOMINIC GORECKY, LUIS ALMEIDA, WALTERIO MAYOL-CUEVAS, ANDREW CALWAY, ANTHONY G. COHN, DAVID C. HOGG, AND DIDIER STRICKER. **Cognitive Learning, Monitoring and Assistance of Industrial Workflows using Egocentric Sensor Networks**. *to appear in PLOS ONE*, 2014. — *19, 28, 29*

[29] OLIVER KORN. **Industrial Playgrounds. How Gamification helps to enrich work of elderly or impaired persons in production**. In *the Proceedings of the Symposium on Engineering Interactive Computer Systems*, 2012. — *20*

[30] BESHOY MORKOS, JOACHIM TAIBER, JOSHUA SUMMERS, LAINE MEARS, GEORGES FADEL, AND TORSTEN RILKA. **Mobile devices within manufacturing environments: a BMW applicability study**. *International Journal on Interactive Design and Manufacturing (IJIDeM)*, **6**(2):101–111, April 2012. — *20, 21*

[31] SONJA STORK AND ANNA SCHUBÖ. **Human cognition in manual assembly: Theories and applications**. *Advanced Engineering Informatics*, **24**(3):320–328, August 2010. — *20, 24*

[32] S. BENBELKACEM, N. ZENATI-HENDA, M. BELHOCINE, A. BELLARBI, M. TADJINE, AND S. MALEK. **Augmented Reality Platform for Solar Systems Maintenance Assistance**. In *the Proceedings of the International Symposium on Environment Friendly Energies in Electrical Applications (EFEEA)*, 2010. — *20, 21*

[33] JENNIFER J. OCKERMAN AND AMY R. PRITCHETT. **Preliminary investigation of wearable computers for task guidance in aircraft inspection**. In *the Proceedings of the International Symposium on Wearable Computers (ISWC)*, 1998. — *20, 21, 24*

[34] MIHRAN TUCERYAN AND NASSIR NAVAB. **Single point active alignment method (SPAAM) for optical see-through HMD calibration for AR**. In *the Proceedings of the International Symposium on Augmented Reality (ISAR)*, 2000. — *20*

[35] CHARLES B. OWEN, JI ZHOU, ARTHUR TANG, AND FAN XIAO. **Display-Relative Calibration for Optical See-Through Head-Mounted Displays**. In *the Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR)*, 2004. — *20*

[36] STUART J. GILSON, ANDREW W. FITZGIBBON, AND ANDREW GLENNERSTER. **Spatial calibration of an optical see-through head-mounted display**. *Journal of Neuroscience Methods*, **173**(1):140–146, 2008. — *20*

[37] SHANNON R. BOWLING, MOHAMMED T. KHASAWNEH, SITTICHAI KAEWKUEKOOL, XIAOCHUN JIANG, AND ANAND K. GRAMOPADHYE. **Evaluating the Effects of Virtual Training in an Aircraft Maintenance Task**. *International Journal of Aviation Psychology*, **18**(1):104–116, 2008. — *21*

[38] TOM HARITOS AND NICKOLAS D. MACCHIARELLA. **A mobile application of augmented reality for aerospace maintenance training**. In *the Proceedings of the Digital Avionics Systems Conference (DASC)*, 2005. — *21*

[39] JENNIFER J. OCKERMAN AND AMY R. PRITCHETT. **Preliminary Investigation of Wearable Computers for Task Guidance in Aircraft Inspection**. In *the Proceedings of the International Symposium on Wearable Computers (ISWC)*, 1998. — *21*

[40] NICKOLAS D. MACCHIARELLA AND DENNIS A. VINCENZI. **Augmented reality in a learning paradigm for flight aerospace maintenance training**. In *the Proceedings of the Digital Avionics Systems Conference (DASC)*, 2004. — *21*

[41] FRANCESCA DE CRESCENZIO, MASSIMILIANO FANTINI, FRANCO PERSIANI, LUIGI DI STEFANO, PIETRO AZZARI, AND SAMUELE SALTI. **Augmented Reality for Aircraft Maintenance Training and Operations Support**. *Computer Graphics and Applications*, **31**(1):96–101, 2011. — *21*

[42] LENNART MALMSKÖLD, ROLAND ÖRTENGREN, BLAIR E. CARLSON, AND LARS SVENSSON. **Virtual Training - Towards a Design Framework**. In *the Proceedings of the World Conference on E-Learning in Corporate, Government, Healthcare, and Higher Education (ELEARN)*, Chesapeake, VA, 2007. — *21, 23*

[43] FLORIAN ECHTLER, FABIAN STURM, KAY KINDERMANN, GUDRUN KLINKER, JOACHIM STILLA, JOERN TRILK, AND HESAM NAJAFI. **The intelligent welding gun: Augmented reality for experimental vehicle construction**. In *Virtual and augmented reality applications in manufacturing*, pages 333–360. Springer, 2004. — *21*

[44] S. DEFFEYES. **Mobile augmented reality in the data center**. *IBM Journal of Research and Development*, **55**:5:1–5:5, 2011. — *21*

[45] JÜRGEN ZAUNER, MICHAEL HALLER, ALEXANDER BRANDL, AND WERNER HARTMANN. **Authoring of a mixed reality assembly instructor for hierarchical structures**. In *the Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR)*, 2003. — *21, 25*

[46] M. ANDERSEN, R. ANDERSEN, C. LARSEN, T. MOESLUND, AND O. MADSEN. **Interactive Assembly Guide Using Augmented Reality**. *the Proceedings of the International Symposium on Advances in Visual Computing*, 2009. *— 21*

[47] OZAN CAKMAKCI, FRANCCOIS BÉRARD, AND JOËLLE COUTAZ. **An augmented reality based learning assistant for electric bass guitar**. In *the Proceedings of the International Conference on Human-Computer Interaction*, 2003. *— 21*

[48] JIM HAHN. **Mobile augmented reality applications for library services**. *New Library World*, **113**:429 – 438, 2012. *— 21*

[49] AARON KOTRANZA, D. SCOTT LIND, CARLA M. PUGH, AND BENJAMIN LOK. **Real-time in-situ visual feedback of task performance in mixed environments for learning joint psychomotor-cognitive tasks**. In *the Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR)*, 2009. *— 21, 24, 29*

[50] E. WIERINCK, V. PUTTEMANS, S. SWINNEN, AND D. VAN STEENBERGHE. **Effect of augmented visual feedback from a virtual reality simulation system on manual dexterity training**. *European Journal of Dental Education*, **10**(6), 2005. *— 21, 22*

[51] SUSANNA NILSSON AND BJORN JOHANSSON. **User experience and acceptance of a mixed reality system in a naturalistic setting: a case study**. In *the Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR)*, 2006. *— 21*

[52] STEVEN J. HENDERSON AND STEVEN FEINER. **Evaluating the benefits of augmented reality for task localization in maintenance of an armored personnel carrier turret**. In *the Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR)*, 2009. *— 21, 23, 129, 130, 131*

[53] BJÖRN SCHWERDTFEGER AND GUDRUN KLINKER. **Supporting Order Picking with Augmented Reality**. In *the Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR)*, 2008. *— 21, 23, 131*

[54] BASSEM BESBES, SYLVIE N. COLLETTE, MOHAMED TAMAAZOUSTI, AND STEVE BOURGEOIS. **An Interactive Augmented Reality System: a Prototype for Industrial Maintenance Training Applications**. In *the Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR)*, 2012. *— 21*

[55] HUGO ALVAREZ, IKER AGUINAGA, AND DIEGO BORRO. **Providing guidance for maintenance operations using automatic markerless Augmented Reality system**. In *the Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR)*, 2011. *— 21, 26*

[56] TAPIO SALONEN AND JUHA SÄÄSKI. **Dynamic and visual assembly instruction for configurable products using augmented reality techniques**. In *Advanced Design and Manufacture to Gain a Competitive Edge*, pages 23–32. Springer, 2008. *— 21*

[57] CHANGZHI KE, BO KANG, DONGYI CHEN, AND XINYU LI. **An augmented reality-based application for equipment maintenance**. *Affective Computing and Intelligent Interaction*, **3784**:836–841, 2005. *— 21*

[58] DOMINIC GORECKY, RICARDO CAMPOS, AND GERRIT MEIXNER. **Seamless Augmented Reality Support On The Shopfloor Based On Cyber-Physical-Systems**. In *the Proceedings of MobileCHI 2012*, 2012. *— 21*

[59] SABINE WEBEL, MARIO BECKER, DIDIER STRICKER, AND HARALD WUEST. **Identifying differences between CAD and physical mock-ups using AR**. In *the Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR)*, 2007. *— 21*

[60] DIRK REINERS, DIDIER STRICKER, GUDRUN KLINKER, AND STEFAN MÜLLER. **Augmented reality for construction tasks: doorlock assembly**. In *the Proceedings of the International Workshop on Augmented Reality (IWAR)*, 1998. *— 21*

[61] M. L. YUAN, S. K. ONG, AND A. Y. C. NEE. **Augmented reality for assembly guidance using a virtual interactive tool**. *International Journal of Production Research*, **46**(7):1745–1767, 2008. *— 21, 39*

[62] TOBIAS BLUM, TOBIAS SIELHORST, AND NASSIR NAVAB. **Advanced augmented reality feedback for teaching 3D tool manipulation**. In *New Technology Frontiers in Minimally Invasive Therapies*, chapter 25, pages 223—-236. Lupensis Biomedical, 2007. *— 21, 39*

[63] JURI PLATONOV, HAUKE HEIBEL, PETER MEIER, AND BERT GROLLMANN. **A mobile markerless AR system for maintenance and repair**. In *the Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR)*, 2006. *— 21*

[64] FABIAN DOIL, W. SCHREIBER, T. ALT, AND C. PATRON. **Augmented reality for manufacturing planning**. In *the Proceedings of the Workshop on Virtual Environments*, 2003. *— 21*

[65] ALEX OLWAL, JONNY GUSTAFSSON, AND CHRISTOFFER LINDFORS. **Spatial augmented reality on industrial CNC-machines**. *Proceedings of SPIE*, **6804**, 2008. *— 21*

[66] STEVEN J. HENDERSON AND STEVEN FEINER. **Opportunistic controls: leveraging natural affordances as tangible user interfaces for augmented reality**. In *the Proceedings of the Symposium on Virtual Reality Software and Technology (VRST)*, New York, NY, USA, 2008. ACM. *— 21*

[67] JONATHAN J. HULL, BERNA EROL, JAMEY GRAHAM, QIFA KE, HIDENOBU KISHI, JORGE MORALEDA, AND DANIEL G. VAN OLST. **Paper-Based Augmented Reality**. In *the Proceedings of the International Conference on Artificial Reality and Telexistence (ICAT)*, 2007. *— 21*

[68] NILS PETERSEN AND DIDIER STRICKER. **Continuous natural user interface: Reducing the gap between real and digital world**. In *the Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR)*, 2009. *— 21, 168, 169*

[69] ARTHUR TANG, CHARLES OWEN, FRANK BIOCCA, AND WEIMIN MOU. **Comparative Effectiveness of Augmented Reality in Object Assembly**. In *the Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2003. *— 21, 23, 131*

[70] DAVID J. HANIFF AND CHRIS BABER. **User evaluation of augmented reality systems**. In *the Proceedings of the International Conference on Information Visualization*, 2003. *— 21*

[71] JOHANNES TUMLER, RÜDIGER MECKE, MICHAEL SCHENK, ANKE HUCKAUF, FABIAN DOIL, GEORG PAUL, EBERHARD A. PFISTER, IRINA BOCKELMANN, AND ANJA ROGGENTIN. **Mobile Augmented Reality in industrial applications: Approaches for solution of user-related issues**. In *the Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR)*, 2008. *— 22*

[72] S. K. ONG AND A. Y. C. NEE. *Virtual Reality and Augmented Reality Applications in Manufacturing*. Springer, 2004. *— 22*

[73] S. K. ONG, M. L. YUAN, AND A. Y. C. NEE. **Augmented reality applications in manufacturing: a survey**. *International Journal of Production Research*, **46**(10):2707–2742, 2008. — *22*

[74] D.W.F. VAN KREVELEN AND R. POELMAN. **A survey of augmented reality technologies, applications and limitations**. *The International Journal of Virtual Reality (IJVR)*, **9**(2):1–20, 2010. — *22*

[75] O. HUGUES, P. FUCHS, AND O. NANNIPIERI. **New augmented reality taxonomy: Technologies and features of augmented environment**. *Handbook of Augmented Reality*, 2011. — *22*

[76] ROBERT W. LINDEMAN AND HARUO NOMA. **A classification scheme for multi-sensory augmented reality**. In *the Proceedings of the Symposium on Virtual Reality Software and Technology (VRST)*, 2007. — *22*

[77] JEAN-MARIE NORMAND, MYRIAM SERVIÈRES, AND GUILLAUME MOREAU. **A new typology of augmented reality applications**. In *the Proceedings of the Augmented Human International Conference*, 2012. — *22*

[78] NASSIR NAVAB. **Industrial augmented reality (IAR): challenges in design and commercialization of killer apps**. In *the Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR)*, 2003. — *22*

[79] NASSIR NAVAB. **Developing killer apps for industrial augmented reality**. *Computer Graphics and Applications*, **24**(3):16–20, 2004. — *22*

[80] RALPH SCHÖNFELDER AND DIETER SCHMALSTIEG. **Augmented reality for industrial building acceptance**. In *the Proceedings of the Virtual Reality Conference (VR)*, 2008. — *22*

[81] MATTHEW FRANKLIN. **The lessons learned in the application of Augmented Reality**. Technical report, DTIC Document, 2006. — *22*

[82] JENNIFER OCKERMAN AND AMY PRITCHETT. **A review and reappraisal of task guidance: Aiding workers in procedure following**. *International Journal of Cognitive Ergonomics*, **4**(3):191–212, 2000. — *22, 23*

[83] BARBARA TVERSKY, JULIE HEISER, PAUL LEE, AND MARIE-PAULE DANIEL. **Cognitive Design Principles for automated Generation of Visualizations**. In G ALLEN, editor, *Applied Spatial Cognition: From research to cognitive technology*, pages 53–74. Lawrence Erlbaum Associates, Inc., Hillsdale, NJ, 2006. — *22*

[84] XIANGYU WANG AND PHILLIP S. DUNSTON. **DESIGN , STRATEGIES , AND ISSUES TOWARDS AN AUGMENTED REALITY-BASED CONSTRUCTION TRAINING PLATFORM**. *ITcon Journal*, **12**:363–380, 2007. — *22*

[85] NIRIT GAVISH, TERESA GUTIERREZ, SABINE WEBEL, JORGE RODRIGUEZ, AND FRANCO TECCHIA. **Design Guidelines for the Development of Virtual Reality and Augmented Reality Training Systems for Maintenance and Assembly Tasks**. *BIO Web of Conferences*, **1**:1–4, 2011. — *22*

[86] SABINE WEBEL, ULI BOCKHOLT, AND JENS KEIL. **Design criteria for AR-based training of maintenance and assembly tasks**. In *Virtual and Mixed Reality-New Trends*, pages 123–132. Springer, 2011. — *22*

[87] MANEESH AGRAWALA, DOANTAM PHAN, JULIE HEISER, JOHN HAYMAKER, JEFF KLINGNER, PAT HANRAHAN, AND BARBARA TVERSKY. **Designing effective step-by-step assembly instructions**. *Transactions on Graphics (TOG)*, **22**(3):828–837, 2003. — *22*

[88] ELENA DRISKILL AND ELAINE COHEN. **Interactive design, analysis, and illustration of assemblies**. In *the Proceedings of the Symposium on Interactive 3D Graphics*, 1995. *— 22*

[89] MANEESH AGRAWALA, W. LI, AND F. BERTHOUZOZ. **Design principles for visual communication**. *Communications of the ACM*, 2011. *— 22*

[90] JULIE HEISER, DOANTAM PHAN, MANEESH AGRAWALA, BARBARA TVERSKY, AND PAT HANRAHAN. **Identification and validation of cognitive design principles for automated generation of assembly instructions**. *the Proceedings of the Working Conference on Advanced Visual Interfaces (AVI)*, 2004. *— 23*

[91] MICHIHIKO GOTO, YUKO UEMATSU, HIDEO SAITO, SHUJI SENDA, AND AKIHIKO IKETANI. **Task support system by displaying instructional video onto AR workspace**. In *the Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR)*, 2010. *— 23, 24, 26, 29*

[92] FRANK BIOCCA, ARTHUR TANG, CHARLES OWEN, FAN XIAO, AND EAST LANSING. **Attention funnel: omnidirectional 3D cursor for mobile augmented reality platforms**. In *the Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, CHI '06, 2006. *— 23, 131, 135*

[93] SABINE WEBEL, ULI BOCKHOLT, TIMO ENGELKE, NIRIT GAVISH, MANUEL OLBRICH, AND CARSTEN PREUSCHE. **An augmented reality training platform for assembly and maintenance skills**. *Robotics and Autonomous Systems*, **61**(4):398–403, April 2013. *— 23*

[94] CINDY M. ROBERTSON, BLAIR MACINTYRE, AND BRUCE N. WALKER. **An evaluation of graphical context when the graphics are outside of the task area**. In *the Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR)*, 2008. *— 23, 131*

[95] DENIS KALKOFEN, MARKUS TATZGERN, AND DIETER SCHMALSTIEG. **Explosion diagrams in augmented reality**. In *the Proceedings of the Virtual Reality Conference (VR)*, 2009. *— 23, 130*

[96] R. MOHAMMAD AND E. KROLL. **Automatic generation of exploded view by graph transformation**. In *the Proceedings of the Conference on Artificial Intelligence for Applications*, 1993. *— 23*

[97] WILMOT LI, MANEESH AGRAWALA, AND DAVID SALESIN. **Interactive image-based exploded view diagrams**. In *the Proceedings of Graphics Interface*, 2004. *— 23*

[98] R. E. MAYER. *The Cambridge Handbook of Multimedia Learning*. Cambridge University Press, Cambridge, 2005. *— 23*

[99] E. VAN GENUCHTEN, K. SCHEITER, AND A. SCHÜLER. **Examining learning from text and pictures for different task types: Does the multimedia effect differ for conceptual, causal, and procedural tasks?** *Computers in Human Behavior*, **28**:2209–2218, 2012. *— 23*

[100] BARBARA TVERSKY, JULIE BAUER MORRISON, AND MIREILLE BETRANCOURT. **Animation: can it facilitate?** *International Journal of HumanComputer Studies*, **57**:247–262, 2002. *— 24*

[101] RICHARD E. MAYER AND LAURA J. MASSA. **Three Facets of Visual and Verbal Learners: Cognitive Ability, Cognitive Style, and Learning Preference.** *Journal of Educational Psychology*, **95**(4):833–846, 2003. *— 24*

[102] HOLGER HORZ AND WOLFGANG SCHNOTZ. **Multimedia : How to Combine Language and Visuals**. *Language at work - Bridging theory and practice*, **3**(4):43–50, 2008. *— 24*

[103] MARKUS HUFF AND STEPHAN SCHWAN. **The verbal facilitation effect in learning to tie**

nautical knots. *Learning and Instruction*, **22**(5):376–385, 2012. — *24*

[104] HAROLD R. BOOHER. **Relative comprehensibility of pictorial information and printed words in proceduralized instructions.** *Human factors*, 1975. — *24*

[105] SUSAN PALMITER, JAY ELKERTON, AND PATRICIA BAGGETT. **Animated demonstrations vs written instructions for learning procedural tasks: a preliminary investigation**. *International Journal of Man-Machine Studies*, **34**(5):687–701, 1991. — *24*

[106] MARIE-PAULE DANIEL AND BARBARA TVERSKY. **How to put things together.** *Cognitive processing*, **13**(4):303–319, November 2012. — *24*

[107] JULIE HEISER AND BARBARA TVERSKY. **Arrows in comprehending and producing mechanical diagrams**. *Cognitive science*, **30**(3):581–592, 2006. — *24*

[108] BARBARA TVERSKY, JEFF ZACKS, PAUL LEE, AND JULIE HEISER. **Lines, Blobs, Crosses and Arrows: Diagrammatic Communication with Schematic Figures**. In M ANDERSON, P CHENG, AND V HAARSLEV, editors, *Diagrams 2000*, pages 221–230. Springer, Berlin, Heidelberg, 2000. — *24*

[109] STEVEN J. HENDERSON AND STEVEN K. FEINER. **Augmented reality in the psychomotor phase of a procedural task**. In *the Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR)*, **159**, 2011. — *24, 28, 29, 30, 134*

[110] M. LAND, N. MENNIE, AND J. RUSTED. **The roles of vision and eye movements in the control of activities of daily living**. *Perception*, **28**, 1999. — *24, 28*

[111] M. LAND. **Eye movements and the control of actions in everyday life**. *Progress in Retinal and Eye Research*, **25**:296–324, 2006. — *24, 28*

[112] CHRISTIAN KNÖPFLE, JENS WEIDENHAUSEN, LAURENT CHAUVIGNE, AND INGO STOCK. **Template Based Authoring for AR based Service Scenarios**. In *the Proceedings of the Virtual Reality Conference (VR)*, pages 237–240, 2005. — *25*

[113] ELIZABETH CARVALHO, HUGO DOMINGUES, GUSTAVO MACCÃES, AND LUÍS PAULO SANTOS. **Augmented Reality Visualization and Edition of Cognitive Workflow Capturing**. In *the Proceedings of the Experiment@ International Conference (exp.at)*, 2011. — *25*

[114] JÜRGEN ZAUNER AND MICHAEL HALLER. **Authoring of mixed reality applications including multi-marker calibration for mobile devices**. In *the Proceedings of the Eurographics conference on Virtual Environments*, 2004. — *25*

[115] DANIEL F. ABAWI, RALF DÖRNER, MICHAEL HALLER, AND JÜRGEN ZAUNER. **Efficient mixed reality application development**. In *the Proceedings of the Conference on Visual Media Production (CVMP)*, 2004. — *25*

[116] STEVEN FEINER. **APEX: An experiment in the automated creation of pictorial explanations**. *Computer Graphics and Applications*, **5**(11):29–37, 1985. — *25*

[117] WILMOT LI, MANEESH AGRAWALA, BRIAN CURLESS, AND DAVID SALESIN. **Automated generation of interactive 3D exploded view diagrams**. In *the Proceedings of SIGGRAPH*. ACM, 2008. — *25*

[118] L DA XU, CHENGEN WANG, ZHUMING BI, JIAPENG YU, AND LI DA XU. **AutoAssem: an automated assembly planning system for complex products**. *Transactions on Industrial Informatics*, **8**(3):669–678, 2012. — *25*

[119] JOHANNES BEHR, PATRICK DÄHNE, AND MARCUS ROTH. **Utilizing X3D for immersive environments**. In *the Proceedings of the International Conference on 3D Web Technology*, 2004. — *25*

[120] TIMO ENGELKE, MARIO BECKER, HARALD WUEST, JENS KEIL, AND ARJAN KUIJPER. **MobileAR Browser A generic architecture for rapid AR-multi-level development**. *Expert Systems with Applications*, **40**(7):2704–2714, 2013. — *25*

[121] JOHANNES BEHR, ULI BOCKHOLT, AND DIETER FELLNER. **Instantreality - a framework for industrial augmented and virtual reality applications**. In *Virtual Reality & Augmented Reality in Industry*, pages 91–99. Springer, 2009. — *25*

[122] FLORIAN LEDERMANN AND DIETER SCHMALSTIEG. **APRIL: a high-level framework for creating augmented reality presentations**. In *the Proceedings of the Virtual Reality Conference (VR)*, 2005. — *25*

[123] M. L. YUAN, S. K. ONG, AND A. Y. C. NEE. **Assembly guidance in augmented reality environments using a virtual interactive tool**. *Innovation in Manufacturing Systems and Technology*, 2005. — *25*

[124] JAKOB NIELSEN. **Finding usability problems through heuristic evaluation**. In *the Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, 1992. — *25*

[125] TIMO ENGELKE, SABINE WEBEL, ULI BOCKHOLT, HARALD WUEST, NIRIT GAVISH, FRANCO TECCHIA, AND CARSTEN PREUSCHE. **Towards automatic generation of multimodal AR-training applications and workflow descriptions**. In *the Proceedings of the International Symposium in Robot and Human Interactive Communication*, 2010. — *26*

[126] JEFFREY M. ZACKS AND BARBARA TVERSKY. **Structuring information interfaces for procedural learning.** *Journal of Experimental Psychology: Applied*, **9**(2):88–100, 2003. — *26*

[127] KATHARINA MURA, NILS PETERSEN, MARKUS HUFF, AND TANDRA GHOSE. **IBES: A Tool for Creating Instructions Based on Event Segmentation**. *Frontiers in Psychology*, **4**(994), 2013. — *26, 165*

[128] M. M. SAYLOR AND D. A. BALDWIN. **Infants' on-line segmentation of dynamic human action**. *Journal of Cognition and Development*, 2007. — *27, 46*

[129] DARE BALDWIN, ANNIKA ANDERSSON, JENNY SAFFRAN, AND MEREDITH MEYER. **Segmenting dynamic human action via statistical structure**. *Cognition*, 2008. — *27, 46*

[130] EKATERINA H. SPRIGGS, FERNANDO DE LA TORRE, AND MARTIAL HEBERT. **Temporal segmentation and activity classification from first-person sensing**. In *the Proceedings of the Computer Vision and Pattern Recognition Workshop (CVPRW)*, 2009. — *27, 28*

[131] ARDHENDU BEHERA, ANTHONY G. COHN, AND DAVID C. HOGG. **Workflow Activity Monitoring Using Dynamics of Pair-Wise Qualitative Spatial Relations**. *Advances in Multimedia Modeling*, 2012. — *27*

[132] FABIAN NATER, HELMUT GRABNER, AND LUC VAN GOOL. **Unsupervised workflow discovery in industrial environments**. In *the Proceedings of the Workshops of the International Conference on Computer Vision (ICCV Workshops)*, 2011. — *27*

[133] F VANDEWIELE AND C MOTAMED. **An unsupervised learning method for human activity recognition based on a temporal qualitative model**. In *the Proceedings of the International Workshop on Behaviour Analysis and Video Understanding*, 2011. — *27*

[134] OREN BOIMAN AND MICHAEL IRANI. **Detecting irregularities in images and in video**.

In *the Proceedings of the International Conference on Computer Vision (ICCV)*, pages 462–469 Vol. 1, 2005. — *27*

[135] RAMA CHELLAPPA, NARESH P. CUNTOOR, SEONG-WOOK JOO, V. S. SUBRAHMANIAN, AND PAVAN TURAGA. **Computational Vision Approaches for Event Modeling**. In T F SHIPLEY AND J M ZACKS, editors, *Understanding events - From Perception to Action*, pages 473–521. Oxford University Press, Oxford, 2008. — *27*

[136] MASAKAZU MATSUGU, MASAO YAMANAKA, AND MASASHI SUGIYAMA. **Detection of activities and events without explicit categorization**. In *the Proceedings of the Workshops of the International Conference on Computer Vision (ICCV Workshops)*, pages 1532–1539, November 2011. — *27*

[137] H. KATO AND M. BILLINGHURST. **Marker tracking and HMD calibration for a video-based augmented reality conferencing system**. In *the Proceedings of the International Workshop on Augmented Reality (IWAR)*, 1999. — *28*

[138] STEVE BOURGEOIS, HANNA MARTINSSON, QUOC-CUONG PHAM, AND SYLVIE NAUDET. **A practical guide to marker based and hybrid visual registration for AR industrial applications**. In *Computer Analysis of Images and Patterns Proceedings*, 2005. — *28*

[139] GABRIELE BLESER AND DIDIER STRICKER. **Advanced tracking through efficient image processing and visual-inertial sensor fusion**. *Computer & Graphics*, **33**:59–72, 2009. — *28*

[140] GABRIELE BLESER AND GUSTAF HENDEBY. **Using optical flow for filling the gaps in visual-inertial tracking**. In *European Signal Processing Conference (EUSIPCO)*, 2010. — *28*

[141] GEORG KLEIN AND DAVID MURRAY. **Parallel Tracking and Mapping for Small AR Workspaces**. In *the Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR)*, 2007. — *28, 31, 51*

[142] WEI TAN, LIU HAOMIN, ZILONG DONG, GUOFENG ZHANG, AND HUJUN BAO. **Robust Monocular SLAM in Dynamic Environments**. In *the Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR)*, 2013. — *28, 31*

[143] ALESSANDRO MULLONI, MAHESH RAMACHANDRAN, GERHARD REITMAYR, DANIEL WAGNER, RAPHAEL GRASSET, AND SERAFIN DIAZ. **User Friendly SLAM Initialization**. In *the Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR)*, 2013. — *28*

[144] CHRISTIAN PIRCHHEIM, DIETER SCHMALSTIEG, AND GERHARD REITMAYR. **Handling Pure Camera Rotation in Keyframe-Based SLAM**. In *the Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR)*, 2013. — *28, 32*

[145] RICHARD A. NEWCOMBE, SHAHRAM IZADI, OTMAR HILLIGES, DAVID MOLYNEAUX, DAVID KIM, ANDREW J. DAVISON, PUSHMEET KOHLI, JAMIE SHOTTON, STEVE HODGES, AND ANDREW FITZGIBBON. **KinectFusion: Real-Time Dense Surface Mapping and Tracking**. In *the Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR)*, 2011. — *28, 31*

[146] A. GEE AND W. MAYOL-CUEVAS. **6D Relocalisation for RGBD Cameras Using Synthetic View Regression**. In *the Proceedings of the British Machine Vision Conference (BMVC)*, 2012. — *28, 31*

[147] BEN GLOCKER, SHAHRAM IZADI, JAMIE SHOTTON, AND ANTONIO CRIMINISI. **Real-Time RGB-D Camera Relocalization**. In *the Proceedings of the International Symposium on*

Mixed and Augmented Reality (ISMAR), 2013. *— 28*

[148] RONALD POPPE. **A survey on vision-based human action recognition**. *Image and Vision Computing*, 2010. *— 28*

[149] Y. TSUBUKU, Y. NAKAMURA, AND Y. OHTA. **Object tracking and object change detection in desktop manipulation for video-based interactive manuals**. In *Advances in Multimedia Information Processing*, 2004. *— 28*

[150] D. DAMEN, P. BUNNUN, A. CALWAY, AND W. MAYOL-CUEVAS. **Real-time Learning and Detection of 3D Texture-less Objects: A Scalable Approach**. In *the Proceedings of the British Machine Vision Conference (BMVC)*, 2012. *— 28*

[151] D. DAMEN, A. GEE, W. MAYOL-CUEVAS, AND A. CALWAY. **Egocentric Real-time Workspace Monitoring using an RGB-D Camera**. In *the Proceedings of the International Conference on Intelligent Robots and Systems (IROS)*, 2012. *— 28*

[152] MARKUS MIEZAL, GABRIELE BLESER, AND DIDIER STRICKER. **Towards practical inside-out head tracking for mobile seating bucks**. In *the Proceedings of the Workshop on Tracking Methods and Application during the International Symposium on Mixed and Augmented Reality (ISMAR Workshops)*, 2012. *— 28, 29*

[153] NICOLAS VIGNAIS, MARKUS MIEZAL, GABRIELE BLESER, KATHARINA MURA, DOMINIC GORECKY, AND FRÉDÉRIC MARIN. **Innovative system for real-time ergonomic feedback in industrial manufacturing**. *Applied Ergonomics*, **44**:566–574, 2013. *— 28, 29*

[154] GABRIELE BLESER, GUSTAF HENDEBY, AND MARKUS MIEZAL. **Using Egocentric Vision to Achieve Robust Inertial Body Tracking under Magnetic Disturbances**. In *the Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR)*, 2011. *— 28, 29*

[155] HEDVIG KJELLSTRÖM, JAVIER ROMERO, DAVID MARTINEZ, AND DANICA KRAGIĆ. **Simultaneous Visual Recognition of Manipulation Actions and Manipulated Objects**. In *the Proceedings of the European Conference on Computer Vision (ECCV)*, 2008. *— 28, 29, 30*

[156] LI SUN, ULRICH KLANK, AND MICHAEL BEETZ. **EyeWatchMe - 3D Hand and Object Tracking for Inside Out Activity Analysis**. In *the Proceedings of the Computer Vision and Pattern Recognition Workshop (CVPRW)*, 2009. *— 28, 29*

[157] GALINA VERES, HELMUT GRABNER, LEE MIDDLETON, AND LUC VAN GOOL. **Automatic workflow monitoring in industrial environments**. In *the Proceedings of the Asian Conference on Computer Vision (ACCV)*, 2011. *— 28*

[158] ROSS CUTLER AND LARRY S. DAVIS. **Robust real-time periodic motion detection, analysis, and applications**. *Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2000. *— 28*

[159] IMRAN N. JUNEJO, EMILIE DEXTER, IVAN LAPTEV, AND PATRICK PÉREZ. **View-independent action recognition from temporal self-similarities**. *Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **33**(1):172–85, January 2011. *— 29, 72*

[160] Y. SATO, K. BERNARDIN, H. KIMURA, AND K. IKEUCHI. **Task analysis based on observing hands and objects by vision**. In *the Proceedings of the International Conference on Intelligent Robots and Systems (IROS)*, 2002. *— 29*

[161] ALIREZA FATHI, XIAOFENG REN, AND JAMES M. REHG. **Learning to Recognise Objects in Egocentric Activities**. In *the Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011. — *29*

[162] DANIEL ROETENBERG. *Inertial and Magnetic Sensing of Human Motion*. PhD thesis, University of Twente, 2006. — *29*

[163] T. SHIRATORI, H. S. PARK, L. SIGAL, Y. SHEIKH, AND J. K. HODGINS. **Motion Capture from Body-Mounted Cameras**. *Transactions on Graphics (TOG)*, **30**(4), 2011. — *29*

[164] T. B. MOESLUND, A. HILTON, AND V. KRUEGER. **A survey of advances in vision-based human motion capture and analysis**. *Computer Vision and Image Understanding*, **104**:90–126, 2006. — *29*

[165] ULRICHT NEUMANN AND ANTHONY MAJOROS. **Cognitive, performance, and systems issues for augmented reality applications in manufacturing and maintenance**. In *the Proceedings of the Virtual Reality Annual International Symposium*, 1998. — *30, 83*

[166] L. ZELNIK-MANOR AND M. IRANI. **Event-based analysis of video**. In *the Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2001. — *30*

[167] CLAUS LENZ, ALICE SOTZEK, THORSTEN RÖDER, MARKUS HUBER, AND STEFAN GLASAUER. **Human Workflow Analysis using 3D Occupancy Grid Hand Tracking in a Human-Robot Collaboration Scenario**. In *the Proceedings of the International Conference on Intelligent Robots and Systems (IROS)*, 2011. — *30*

[168] W. W. MAYOL AND D. W. MURRAY. **Wearable Hand Activity Recognition for Event Summarization**. In *the Proceedings of the International Symposium on Wearable Computers (ISWC)*, 2005. — *30*

[169] DIETER KOLLER, GUDRUN KLINKER, ERIC ROSE, DAVID BREEN, ROSS WHITAKER, AND MIHRAN TUCERYAN. **Real-time vision-based camera tracking for augmented reality applications**. In *the Proceedings of the Symposium on Virtual Reality Software and Technology (VRST)*, 1997. — *31*

[170] FOLKER WIENTAPPER, HARALD WUEST, AND ARJAN KUIJPER. **Reconstruction and Accurate Alignment of Feature Maps for Augmented Reality**. In *the Proceedings of the Conference on 3D Imaging, Modeling, Processing, Visualization and Transmission*, pages 140–147, 2011. — *31*

[171] ETHAN RUBLEE, VINCENT RABAUD, KURT KONOLIGE, AND GARY BRADSKI. **ORB: an efficient alternative to SIFT or SURF**. In *the Proceedings of the International Conference on Computer Vision (ICCV)*. IEEE, 2011. — *31, 69*

[172] MICHAEL CALONDER, VINCENT LEPETIT, CHRISTOPH STRECHA, AND PASCAL FUA. **BRIEF : Binary Robust Independent Elementary Features**. In *the Proceedings of the European Conference on Computer Vision (ECCV)*, Lecture Notes in Computer Science, 2010. — *31, 35, 154, 177*

[173] XIAOMENG WU, SHUNSUKE KAMIJO, WENLI ZHANG, AND MASAO SAKAUCHI. **Interactive Object Annotation for Construction of Video Information System**. In *the Proceedings of the International Symposium on Multimedia (ISM)*, 2006. — *31*

[174] D. DAMEN, A. GEE, A. CALWAY, AND W. MAYOL-CUEVAS. **Detecting and Localising Multiple 3D Objects: A Fast and Scalable Approach**. In *the Proceedings of the IROS Workshop on Active Semantic Perception and Object Search in the Real World (ASP-AVS)*, 2011. — *31*

[175] OLIVIER FAUGERAS AND FRANCIS LUSTMAN. **Motion and structure from motion in a**

piecewise planar environment. *International Journal of Pattern Recognition and Artificial Intelligence*, **02**(03), 1988. — *32*

[176] D. SANTOSH KUMAR AND C. V. JAWAHAR. **Robust Homography-Based Control for Camera Positioning in Piecewise Planar Environments**. In *the Proceedings of the Indian Conference on Computer Vision, Graphics and Image*, Lecture Notes in Computer Science, 2006. — *32*

[177] STEFFEN GAUGLITZ, CHRIS SWEENEY, JONATHAN VENTURA, MATTHEW TURK, AND TOBIAS HOLLERER. **Live tracking and mapping from both general and rotation-only camera motion**. In *the Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR)*, 2012. — *32, 51*

[178] NAVNEET DALAL, B. TRIGGS, AND WILLIAM TRIGGS. **Histograms of Oriented Gradients for Human Detection**. In *the Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2005. — *32, 33*

[179] CHRISTOPH H. LAMPERT, MATTHEW B. BLASCHKO, AND THOMAS HOFMANN. **Beyond sliding windows: Object localization by efficient subwindow search**. In *the Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008. — *32, 33, 34, 100*

[180] STEFAN HINTERSTOISSER, STEFAN HOLZER, CEDRIC CAGNIART, SLOBODAN ILIC, KURT KONOLIGE, NASSIR NAVAB, AND VINCENT LEPETIT. **Multimodal Templates for Real-Time Detection of Texture-Less Objects in Heavily Cluttered Scenes**. In *the Proceedings of the International Conference on Computer Vision (ICCV)*, 2011. — *33*

[181] A. GIONIS, P. INDYK, AND R. MOTWANI. **Similarity search in high dimensions via hashing**. In *the Proceedings of the International Conference on Very Large Data Bases (VLDB)*, 1999. — *33, 34*

[182] GREGORY SHAKHNAROVICH, PAUL VIOLA, AND TREVOR DARRELL. **Fast pose estimation with parameter-sensitive hashing**. In *the Proceedings of the International Conference on Computer Vision (ICCV)*, 2003. — *33, 34*

[183] C. F. OLSON AND D. P. HUTTENLOCHER. **Automatic target recognition by matching oriented edge pixels.** *Transactions on Image Processing*, **6**(1):103–113, January 1997. — *33*

[184] D. M. GAVRILA AND V. PHILOMIN. **Real-time object detection for "smart" vehicles**. In *the Proceedings of the International Conference on Computer Vision (ICCV)*, 1999. — *33*

[185] S. HOLZER, S. HINTERSTOISSER, S. ILIC, AND N. NAVAB. **Distance transform templates for object detection and pose estimation**. In *the Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009. — *33, 35*

[186] V. ATHITSOS AND S. SCLAROFF. **Estimating 3D hand pose from a cluttered image**. In *the Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2003. — *33, 34, 35, 36, 114*

[187] DANIEL MOHR AND GABRIEL ZACHMANN. **FAST: Fast Adaptive Silhouette Area based Template Matching**. In *the Proceedings of the British Machine Vision Conference (BMVC)*, Port d'Andratx, Mallorca, Spain, 2010. Springer Verlag. — *33, 34, 35, 36, 114*

[188] SIMON TAYLOR AND TOM DRUMMOND. **Multiple Target Localisation at over 100 FPS**. In *the Proceedings of the British Machine Vision Conference (BMVC)*, 2009. — *33*

[189] VINCENT LEPETIT AND PASCAL FUA. **Keypoint Recognition Using Randomized Trees**.

Transactions on Pattern Analysis and Machine Intelligence (PAMI), **28**(9):1465–1479, 2006. *— 35*

[190] LUCA BALLAN, APARNA TANEJA, JÜRGEN GALL, LUC VAN GOOL, MARC POLLEFEYS, AND TANEJA APARNA. **Motion Capture of Hands in Action Using Discriminative Salient Points**. In *the Proceedings of the European Conference on Computer Vision (ECCV)*, 2012. *— 35, 36, 114*

[191] ROBERT Y. WANG AND JOVAN POPOVIĆ. **Real-time hand-tracking with a color glove**. In *the Proceedings of SIGGRAPH*, SIGGRAPH '09, New York, New York, USA, 2009. *— 35, 95*

[192] A. B. R. GHERBI, S. G. J. RICHARDSON, AND D. TEIL. **Gesture-Based Communication in Human-Computer Interaction**. In ANTONIO CAMURRI AND GUALTIERO VOLPE, editors, *the Proceedings of the International Gesture Workshop*, Lecture Notes in Computer Science, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg. *— 35*

[193] PAUL VIOLA AND MICHAEL J. JONES. **Robust Real-time Object Detection**. *International Journal of Computer Vision (IJCV)*, **57**(2):137–154, 2001. *— 35*

[194] IASON OIKONOMIDIS, NIKOLAOS KYRIAZIS, AND ANTONIS ARGYROS. **Markerless and efficient 26-dof hand pose recovery**. In *the Proceedings of the Asian Conference on Computer Vision (ACCV)*, 2010. *— 36, 114, 163*

[195] JOHN LIN, YING WU, AND THOMAS S. HUANG. **Capturing human hand motion in image sequences**. In *the Proceedings of the Workshop on Motion and Video Computing*, 2002. *— 36, 37, 97*

[196] ALI EROL, GEORGE BEBIS, MIRCEA NICOLESCU, RICHARD D. BOYLE, AND XANDER TWOMBLY. **A review on vision-based full dof hand motion estimation**. In *the Proceedings of the Computer Vision and Pattern Recognition Workshops (CVPR Workshops)*, 2005. *— 36*

[197] B. STENGER, P. R. S. MENDONCCA, AND R. CIPOLLA. **Model-based 3D tracking of an articulated hand**. In *the Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2001. *— 36, 97, 114*

[198] J. KENNEDY AND R. EBERHART. **Particle swarm optimization**. In *the Proceedings of the International Conference on Neural Networks (ICNN)*, 1995. *— 36, 106, 107, 114*

[199] MARTIN DE LA GORCE, NIKOS PARAGIOS, AND DAVID J. FLEET. **Model-based hand tracking with texture, shading and self-occlusions**. In *the Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008. *— 36*

[200] IASON OIKONOMIDIS, NIKOLAOS KYRIAZIS, AND ANTONIS ARGYROS. **Efficient model-based 3D tracking of hand articulations using Kinect**. In *the Proceedings of the British Machine Vision Conference (BMVC)*, 2011. *— 37, 114*

[201] IASON OIKONOMIDIS, NIKOLAOS KYRIAZIS, AND ANTONIS ARGYROS. **Full dof tracking of a hand interacting with an object by modeling occlusions and physical constraints**. In *the Proceedings of the International Conference on Computer Vision (ICCV)*, 2011. *— 37, 106, 114*

[202] IASON OIKONOMIDIS, NIKOLAOS KYRIAZIS, AND ANTONIS ARGYROS. **Tracking the articulated motion of two strongly interacting hands**. In *the Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012. *— 37, 108, 114*

[203] MARCEL GERMANN, ALEXANDER HORNUNG, RICHARD KEISER, REMO ZIEGLER, STEPHAN WÜRMLIN, AND MARKUS GROSS.

Articulated Billboards for Video-based Rendering. In *the Proceedings of Eurographics*, 2010. — *37, 87*

[204] MICHAEL J. JONES AND TOMASO POGGIO. **Multidimensional morphable models: A framework for representing and matching object classes**. *International Journal of Computer Vision (IJCV)*, **29**(2):107–131, 1998. — *37*

[205] T. COOTES, G. EDWARDS, AND C. TAYLOR. **Active appearance models**. In *the Proceedings of the European Conference on Computer Vision (ECCV)*, 1998. — *37*

[206] HAE JONG SEO AND PEYMAN MILANFAR. **Detection of human actions from a single example**. In *the Proceedings of the International Conference on Computer Vision (ICCV)*, pages 1965–1970, 2009. — *39*

[207] JIANBO SHI AND CARLO TOMASI. **Good features to track**. In *the Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 1994. — *50, 177*

[208] GILLES SIMON, ANDREW W. FITZGIBBON, AND ANDREW ZISSERMAN. **Markerless tracking using planar structures in the scene**. In *the Proceedings of the International Symposium on Augmented Reality (ISAR)*, 2000. — *63*

[209] NILS PETERSEN AND DIDIER STRICKER. **Fast hand detection using posture invariant constraints**. In *the Proceedings of the KI Conference*, Lecture Notes in Artificial Intelligence, 2009. — *64*

[210] OREN BOIMAN, ELI SHECHTMAN, AND MICHAL IRANI. **In defense of Nearest-Neighbor based image classification**. In *the Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008. — *67*

[211] ANDREAS HOFHAUSER, CARSTEN STEGER, AND NASSIR NAVAB. **Edge-Based Template Matching and Tracking for Perspectively Distorted Planar Objects**. In *the Proceedings of the International Symposium on Advances in Visual Computing*, ISVC '08, Berlin, Heidelberg, 2008. — *68*

[212] CHENG LI AND KRIS M. KITANI. **Pixel-level Hand Detection in Ego-Centric Videos**. In *the Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013. — *69*

[213] IRA KEMELMACHER-SHLIZERMAN, ELI SHECHTMAN, RAHUL GARG, AND STEVEN M. SEITZ. **Exploring Photobios**. In *the Proceedings of SIGGRAPH*, 2011. — *87, 92*

[214] A. JACOBSON, I. BARAN, J. POPOVIC, AND O. SORKINE. **Bounded biharmonic weights for real-time deformation**. In *the Proceedings of SIGGRAPH*, 2011. — *92*

[215] YING WU, JOHN Y. LIN, AND THOMAS S. HUANG. **Capturing natural hand articulation**. In *the Proceedings of the International Conference on Computer Vision (ICCV)*, **2**, pages 426–432, 2001. — *97*

[216] RICCARDO POLI. **Analysis of the Publications on the Applications of Particle Swarm Optimisation**. *Journal of Artificial Evolution and Applications*, **2008**:1–10, January 2008. — *106*

[217] RICCARDO POLI, JAMES KENNEDY, AND TIM BLACKWELL. **Particle swarm optimization**. *Swarm Intelligence*, **1**(1):33–57, August 2007. — *106*

[218] SMITHMICRO. **Poser 3D Animation & Character Creation Software**. http://poser.smithmicro.com/, Last visited on January 2014. — *108, 111*

[219] ALEXANDRU TELEA. **An Image Inpainting Technique Based on the Fast Marching**

Method. *Journal of Graphics Tools*, **9**(1):23–34, 2004. — *110*

[220] SHAN LU, GANG HUANG, DIMITRIS SAMARAS, AND DIMITRIS METAXAS. **Model-based integration of visual cues for hand tracking**. In *the Proceedings of the Workshop on Motion and Video Computing*, 2002. — *114*

[221] B. STENGER, P. R. S. MENDONCCA, AND R. CIPOLLA. **Model-based hand tracking using an unscented kalman filter**. In *the Proceedings of the British Machine Vision Conference (BMVC)*, **1**, pages 63–72, 2001. — *114*

[222] BJÖRN STENGER, ARASANATHAN THAYANANTHAN, PHILIP H. S. TORR, AND ROBERTO CIPOLLA. **Model-based hand tracking using a hierarchical Bayesian filter.** *Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **28**(9):1372–1384, September 2006. — *114*

[223] DANIEL MOHR AND GABRIEL ZACHMANN. **Silhouette Area Based Similarity Measure for Template Matching in Constant Time**. In *the Proceedings of the International Conference of Articulated Motion and Deformable Objects*, Port d'Andratx, Mallorca, Spain, 2010. Springer Verlag. — *114*

[224] DANIEL MOHR AND GABRIEL ZACHMANN. **Segmentation-free, area-based articulated object tracking**. *Advances in Visual Computing*, 2011. — *114*

[225] MICHAEL R. MARNER, ANDREW IRLITTI, AND BRUCE H. THOMAS. **Improving Procedural Task Performance with Augmented Reality Annotations**. In *the Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR)*, 2013. — *130, 131*

[226] VOLKER BLANZ, MICHAEL J. TARR, AND HEINRICH H. BÜLTHOFF. **What object attributes determine canonical views?** *Perception*, **28**(5):575–600, 1999. — *130*

[227] BÄRBEL GARSOFFKY, STEPHAN SCHWAN, AND MARKUS HUFF. **Canonical views of dynamic scenes.** *Journal of experimental psychology. Human perception and performance*, **35**(1):17–27, February 2009. — *130*

[228] DAVID DRASCIC AND PAUL MILGRAM. **Perceptual issues in augmented reality**. In *the Proceedings of SPIE: Stereoscopic Displays and Virtual Reality Systems*, 1996. — *130*

[229] CHRIS FURMANSKI, RONALD AZUMA, AND MIKE DAILY. **Augmented-reality visualizations guided by cognition: Perceptual heuristics for combining visible and obscured information**. In *the Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR)*, 2002. — *130*

[230] DENIS KALKOFEN, EDUARDO VEAS, STEFANIE ZOLLMANN, MARKUS STEINBERGER, AND DIETER SCHMALSTIEG. **Adaptive Ghosted Views for Augmented Reality**. In *the Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR)*, 2013. — *130*

[231] NVIDIA. **CUDA**. http://www.nvidia.com/object/cuda_home_new.html. — *143*

[232] J. E. STONE, D. GOHARA, AND G. SHI. **OpenCL: A parallel programming standard for heterogeneous computing systems**. *Computing in Science and Engineering*, **12**:66–73, 2010. — *143*

[233] D. TARDITI, S. PURI, AND J. OGLESBY. **Accelerator: using data-parallelsim to program GPUs for genral purpose uses**. In *ASPLOS*, 2006. — *144*

[234] L. DAGUM AND R. MENON. **OpenMP: an industry standard API for shared-memory programming**. *Computational Science Engineering, IEEE*, **5**(1):46–55, 1998. — *144*

[235] BYUNG-GON CHUN AND PETROS MANIATIS. **Augmented smartphone applications through clone cloud execution**. In *HotOS*, 2009. — *148*

[236] DANIEL WAGNER AND DIETER SCHMALSTIEG. **First steps towards handheld augmented reality**. In *the Proceedings of the International Symposium on Wearable Computers (ISWC)*, 2003. — *148*

[237] H. T. REGENBRECHT AND R. SPECHT. **A mobile Passive Augmented Reality Device - mPARD**. In *the Proceedings of the International Symposium on Augmented Reality (ISAR)*, 2000. — *149*

[238] JUERGEN GAUSEMEIER, JUERGEN FRUEND, CARSTEN MATYSZCZOK, BEAT BRUEDERLIN, AND DAVID BEIER. **Development of a real time image based object recognition method for mobile AR-devices**. In *AFRIGRAPH*, 2003. — *149*

[239] STEPHAN GAMMETER, ALEXANDER GASSMANN, LUKAS BOSSARD, TILL QUACK, AND LUC VAN GOOL. **Server-side object recognition and client-side object tracking for mobile augmented reality**. In *the Proceedings of the Computer Vision and Pattern Recognition Workshop (CVPRW)*, 2010. — *149*

[240] S. S. KUMAR, MIN SUN, AND S. SAVARESE. **Mobile object detection through client-server based vote transfer**. In *the Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012. — *149*

[241] BYUNG-GON CHUN, SUNGHWAN IHM, PETROS MANIATIS, MAYUR NAIK, AND ASHWIN PATTI. **CloneCloud: elastic execution between mobile device and cloud**. In *EuroSys*, 2011. — *149*

[242] VERDI MARCH, YAN GU, ERWIN LEONARDI, GEORGE GOH, MARKUS KIRCHBERG, AND BU SUNG LEE. **μCloud: Towards a New Paradigm of Rich Mobile Applications**. *Procedia Computer Science*, **5**:618–624, 2011. — *149*

[243] **OpenCV**. http://opencv.org, Last visited on January 2014. — *150*

[244] EDWARD ROSTEN AND TOM DRUMMOND. **Machine Learning for High-Speed Corner Detection**. In *the Proceedings of the European Conference on Computer Vision (ECCV)*, 2006. — *154, 177*

[245] MILES RICHARDSON, GARY JONES, AND MARK TORRANCE. **Identifying the task variables that influence perceived object assembly complexity.** *Ergonomics*, **47**(9):945–964, 2004. — *165*

[246] JEFFREY M. ZACKS, SHAWN KUMAR, RICHARD A. ABRAMS, AND RITESH MEHTA. **Using movement and intentions to understand human activity.** *Cognition*, **112**(2):201–16, August 2009. — *166*

[247] ABIGAIL J. SELLEN AND RICHARD HARPER. *The myth of the paperless office*. Cambridge, Mass. : MIT Press, 2001. — *170*

[248] ANOTO GROUP. **Anoto Digital Pen**. http://www.anoto.com, Last visited on January 2014. — *170*

[249] BRUCE D LUCAS AND TAKEO KANADE. **An iterative image registration technique with an application to stereo vision**. In *the Proceedings of the International Joint Conference on Artificial Intelligence*, 1981. — *177*

Curriculum Vitae

Nils Daniel Petersen
Erlenstr. 24
D - 67655 Kaiserslautern
Phone: +49 631 205 75 3540
Cell: +49 177 6041658
E-mail: nils.petersen@dfki.de

Education

10 / 2001 - 04 / 2008 — KIT / University of Karlsruhe (TH)

- Study of Computer Science, final grade: 1.5
- Majors: Cognitive Systems and Anthropomatics
- Computer Science I + II passed in 1999 within the educational project ”Schüler-Studenten”

01 / 2006 - 06 / 2006 — University of Edinburgh, Scotland

- College of Science and Engineering
- Abroad semester

09 / 1991 - 07 / 2000 — Humboldt-Gymnasium, Karlsruhe

Work experience

09 / 2008 - today — Deutsches Forschungszentrum für Künstliche Intelligenz (DFKI) GmbH

- Researcher at the research department *Augmented Vision*
- Participation in three large-scale research projects
  - *AVILUSplus*, work on hand tracking in video sequences
  - *Cognito*, work on image-based user monitoring in manual workflows
  - *Software-Cluster*, work on workflow analysis and assistance systems
- Project lead in several industrially funded projects focused on Augmented Reality

01 / 2004 - 03 / 2008 — KIT / University of Karlsruhe (TH)

- Student scientific assistant at the Institute for Cognitive Systems (IAKS)
- Development of a stereoscopic volume renderer for CT- / MR- image sets
- Implementation of a 3D-editor to create and calibrate kinematic body models for motion analysis

10 / 2000 - 09 / 2001 — Deutscher Paritätischer Wohlfahrtsverband (DPWV)

- Civil service

07 / 2000 - 09 / 2001 — Brainiac GmbH

- Customizing for enterprise resource planning software Sage Office Line
- Database-centric development in large existing code base (VBA, SQL)

Awards

Best paper award for the conference paper:

Nils Petersen and Didier Stricker, *Morphing Billboards for Accurate Reproduction of Shape and Shading of Articulated Objects with an Application to Real-time Hand Tracking*, in the Proceedings of Computational Modeling of Objects presented in Images (CompImage), 2012

Winner of the start-up idea contest, organized by the start-up office, Kaiserslautern, December, 2013

List of Publications

Journal articles

Nils Petersen and Didier Stricker, *Morphing Billboards - An Image Based Appearance Model for Hand Tracking*, In Computer Methods in Biomechanics and Biomedical Engineering: Imaging & Visualization (accepted), 2014

Katharina Mura, Nils Petersen, Markus Huff, and Tandra Ghose, *IBES: A Tool for Creating Instructions Based on Event Segmentation*, In Frontiers in Psychology, 4(994)

G. Bleser, D. Damen, A. Behera, K. Mura, M. Miezal, A. Gee, G. Hendeby, N. Petersen, G. Maães, H. Domingues, D. Gorecky, L. Almeida, W. Mayol-Cuevas, A. Calway, A. Cohn, D. Hogg, and D. Stricker, *Cognitive Learning, Monitoring and Assistance of Industrial Workflows using Egocentric Sensor Networks*, In PLOS ONE (to appear), 2014

Conference papers

Philipp Hasper, Nils Petersen, and Didier Stricker, *Remote Execution vs. Simplification for Mobile Real-time Computer Vision*, in the Proceedings of the International Conference on Computer Vision Theory and Applications, 2014

Nils Petersen, Alain Pagani, and Didier Stricker, *Real-time Modeling and Tracking Manual Workflows from First-Person Vision*, in the Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR), 2013

Christian Bartolein, Nils Petersen and Arnold A. Taube, *Augmented Reality on Mobile Devices to Enhance Training and Service Capabilities*, in the Proceedings of the International Conference Agricultural Engineering, 2013

Nils Petersen and Didier Stricker, *Learning Task Structure from Video Examples for Workflow Tracking and Authoring*, in the Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR), 2012

Nils Petersen and Didier Stricker, *Morphing Billboards for Accurate Reproduction of Shape and Shading of Articulated Objects with an Application to Real-time Hand Tracking*, in the Proceedings of Computational Modeling of Objects presented in Images (CompImage), 2012

Bjrn Forcher, Nils Petersen, and A Dengel, *Visualization of Intuitive Explanations Using Koios++*, in the Proceedings of the International Workshop on Explanation-aware Computing (ExaCt-2012), located at ECAI, 2012

Nils Petersen and Didier Stricker, *Adaptive Search Tree Database Indexing for Hand Tracking*, in the Proceedings of Computer Graphics, Visualization, Computer Vision and Image Processing (CGVCVIP), 2012

Nils Petersen, Julian Pastarmov, and Didier Stricker, *ARGOS - a Software Framework to Facilitate User Transparent Multi-threading*, in the Proceedings of the MARC Symposium, 2011

Gerrit Meixner, Nils Petersen, and Holger Koessling, *User Interaction Evolution in the SmartFactoryKL*, in the Proceedings of the BCS Interaction Specialist Group Conference, 2010

Nils Petersen and Didier Stricker, *Continuous Natural User Interface: Reducing the Gap Between Real and Digital World*, in the Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR), 2009

Nils Petersen and Didier Stricker, *Fast Hand Detection Using Posture Invariant Constraints*, in the Proceedings of Advances in Artificial Intelligence (KI), 2009

Talks

Digitalisierte Expertise und Augmented Reality Assistenzsysteme, *IFF Gastvortragsreihe*, Magdeburg, December 4, 2013

Augmented Reality in Marketing & Service, *Webciety panel discussion at the CeBIT*, Hannover, March 8, 2013

Lernen im Kontext, *Lab talk at the CeBIT*, Hannover, March 6, 2013

Neuartige Verfahren bei der Wartung komplexer Anlagen Serviceunterstützung durch neuartige IT -Verfahren, *Conges Kundentage*, St. Wendel, June 19, 2013

Augmented Reality auf mobilen Endgeräten, *[vdav] Branchentreff*, Berlin, June 9, 2010

Gestengesteuerte Visualisierung von Prozess- und Produktdaten, *Innovationstag Smartfactory*, Kaiserslautern-Siegelbach, October 1, 2009

Declaration

I herewith declare that I have produced this thesis without the prohibited assistance of third parties and without making use of aids other than those specified; notions taken over directly or indirectly from other sources have been identified as such. This thesis has not previously been presented in identical or similar form to any other German or foreign examination board.

The thesis work was conducted from September 2008 to January 2014 under the supervision of Prof. Dr. Didier Stricker at research Department Augmented Vision within the German Research Center for Artifical Intelligence (DFKI).

Kaiserslautern, January 30, 2014