



**HAL**  
open science

## Human and Workcell Event Recognition and Its Application Areas in Industrial Assembly

Matthias Propst, Sharath Chandra Akkaladevi, Michael Hofmann, Christian Wögerer, Andreas Pichler

► **To cite this version:**

Matthias Propst, Sharath Chandra Akkaladevi, Michael Hofmann, Christian Wögerer, Andreas Pichler. Human and Workcell Event Recognition and Its Application Areas in Industrial Assembly. 9th International Precision Assembly Seminar (IPAS), Dec 2020, Held virtually, Unknown Region. pp.260-275, 10.1007/978-3-030-72632-4\_19 . hal-03520410

**HAL Id: hal-03520410**

**<https://inria.hal.science/hal-03520410>**

Submitted on 11 Jan 2022





**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Distributed under a Creative Commons Attribution 4.0 International License

# Human and Workcell Event Recognition and its Application Areas in Industrial Assembly

Matthias Propst<sup>1</sup> , Sharath Chandra Akkaladevi<sup>1</sup> , Michael Hofmann<sup>1</sup> ,  
Christian Wögerer<sup>1</sup> , and Andreas Pichler<sup>1</sup>

Profactor Gmbh, Austria  
firstname.lastname@profactor.at,  
WWW home page: <http://profactor.at>

**Abstract.** Assistance systems in production gain increased importance for industry, to tackle the challenges of mass customization and the demographic change. Common to these systems is the need for context awareness and understanding of the state of an assembly process. This paper presents an approach towards Event Recognition in manual assembly settings and introduces concepts to apply this key technology to the application areas of Quality Assurance, Worker Assistance, and Process Teaching.

**Keywords:** event recognition, manual assembly processes, knowledge representation

## 1 Introduction

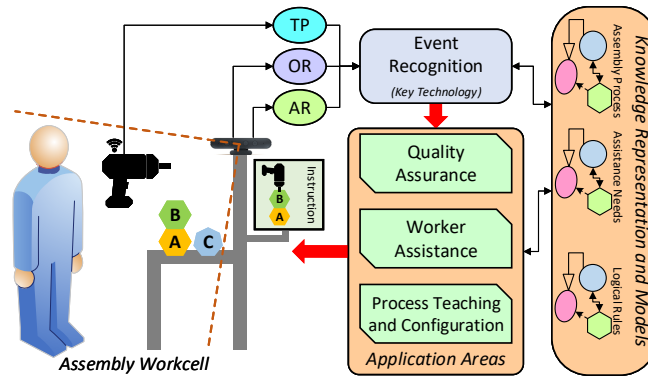
Today's production industry is facing an ongoing paradigm shift, from mass production towards mass customization and tailoring of highly individualized products. Moreover, production systems and work cells are equipped with sensor systems and interconnected to realize efficient information transport. Due to these facts, the information density and complexity is increasing, which means higher mental workload and stress to human workers in the loop. There is a demand for cognitive assistance provided to human workers in order to reduce stress, the probability of mistakes, and to keep motivation and concentration at a high level. Such assistance systems need to be aware of the production context as well as of the state of the environment and the behavior of the human worker. This paper focuses on description of an approach to classify events performed by a human worker, including smart tools, in a manual assembly setting. Moreover, three potential application fields of event recognition, as an enabling technology, will be introduced and discussed. The paper concludes with a description of the current state of development and future work items.

## 2 Problem Statement

Perceiving the environment of an assembly workplace and making a meaning of the current assembly status and activities is a very challenging task. The

first challenge is to perceive all relevant information from the environment, and combine this data to meaningful events. This requires a clear definition of the way events are characterized, which has to correlate with the representation of the domain knowledge of Assembly Processes (henceforth: AP). Additionally to the abstraction of events and AP knowledge, algorithms and rules are required to store, retrieve and reason about the exiting knowledge. This publication discusses methods and implementation to create a framework for classifying events in an AP setting. The proposed framework shall serve as enabling technology to realize domain relevant applications, including a) Quality Assurance, b) Worker Assistance, and c) Process Teaching and Configuration.

A manual assembly setup/workplace, as schematically depicted in figure 1, is proposed as platform for implementation and verification of the approaches and methods. It is assumed that the assembly workplace is realized as a work table, including parts or objects (see objects A, B, C), and smart tools (e.g. wireless screwing devices) the human worker can use. Smart tools provide interfaces to communicate tool status and configuration data e.g. received from Manufacturing Execution Systems (MES). An information screen is available to provide instructions and hints to the worker in order to reduce probability of assembly mistakes. In order to perceive the environment, 2D and 3D vision sensors are



**Fig. 1.** Event Recognition and its applications in assembly settings.

applied to observe the work table including the human worker, objects and the smart tools. The sensor data needs to be evaluated accordingly using perception modules, including *Tool Proprioception (TP)*, *Action Recognition (AR)*, and *Object Recognition (OR)*. These subordinate perception modules provide input to the *Event Recognition* system. As depicted in figure 1 above, event recognition also requires domain knowledge and rules to make a meaning of the perception data. The goal of this publication is to present how event recognition can be realized in assembly settings and how it can contribute in solving the functional challenges of potential application fields as described below.

## 2.1 Quality Assurance

Manual APs include steps that are very critical to ensure high quality of the product. In some cases, these steps include actions that involve non-rigid parts or there is no application of tools required but the worker's hands only. Missing such process steps yields the need of disassembling parts of the product again. It is a challenge to realize a sensor based detection, without requiring manual confirmation of each process step through the worker. Event recognition can help here, to monitor critical process steps during manual assembly.

## 2.2 Worker Assistance

This application area includes modular assistance systems, that can be tailored and configured based on the given AP requirements and the current process context. A major challenge in this field is given through the required collaboration between assistance system and human worker. The interaction needs to happen intuitively and seamlessly to ensure worker acceptance on the one hand, and to keep performance on a high level. Assistance systems need to interpret and understand what is happening in the environment. In this context event recognition can provide valuable information inputs to understand the situation of environment and the human worker.

## 2.3 Process Teaching and Configuration

High product variation and flexibility require a significant reduction of setup and ramp-up times for production systems. However, today's automation systems, especially when including robot technology, require a substantial programming effort to adapt to changes. The challenging problem of reducing the programming effort for complex tasks, by using modes that are natural to humans, is an open topic in R&D. This paper will discuss an approach, where event recognition is applied to enable programming of robots based on demonstration of steps through human workers.

# 3 State of the Art

## 3.1 Event Recognition

Event recognition in relation to an AP in an industrial setting could be defined as identifying the activities or interactions of agents in the environment. The terms gestures, actions, activities, and events have been used interchangeably [1]. An action, denoting a sequence of movements, that involves an agent interacting with an object is defined as an *Event*. Events are high-level activities which are derived not directly from the sensor data, but from the interpreted information. In literature, to recognize complex activities several hierarchical approaches are proposed and can be broadly divided into three categories depending on the

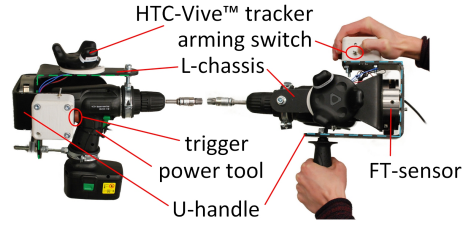
recognition process involved. They are: **Statistical approaches** use statistical state-based models to recognize activities (usually two layers) e.g. Hidden Markov Models (HMM) and Bayesian Networks (DBN) [2] [3]. For each model, the probability of the model generating a sequence of observations (i.e., actions) is calculated. This step is followed by either a maximum likelihood estimation (MLE) or the maximum a posteriori probability (MAP) classifier to classify the complex activity. **Syntactic approaches** model human activities as a string of symbols, where each symbol corresponds to a low-level action. These approaches generally use a grammar syntax such as stochastic context free grammar (SCFG) to model activities [4]. **Description-based approaches** maintain the temporal, spatial and logical relationships between low-level actions to describe complex activities. In other words, a complex activity is described as an occurrence of its composed low level actions which satisfy a certain relation (logical, temporal, spatial) [5]. Event recognition in industrial scenarios has not yet been widely researched. There are few approaches that show promise in this direction. An extensive review of event recognition approaches is given in [6]. Most approaches with task driven goals only consider object localization for manipulation (often with simple structure). Thereby, they do not consider the problems associated with dynamic environments, such as tracking and manipulation of real world objects (exceptions include [7] [8]).

### 3.2 Object Recognition and Tracking

Localization of objects in the assembly workcell is required to understand the status of APs and to classify events. 3D object recognition systems return the positions and orientations of objects of interest. Algorithms of common approaches are based on depth data and CAD models of the objects of interest (see e.g. [9], [10] or [11]). Object tracking systems ([12][10]) can help to increase performance and stability of detections in highly dynamical environments. A further performance boost can be achieved by adding process context information to object localization and tracking systems [13][14]. In [15] earlier work on 3D object tracking is extended, by combining a object tracking system with a cognitive architecture in a dynamic Human Robot Collaboration (HRC) assembly scenario. Such an architecture also supports reasoning about the current state of an AP.

### 3.3 Smart Tools in Assembly Settings

Today's industrial manual assembly workcells are often equipped with *Smart Tools*, which include communication interfaces to local control systems (e.g. screwing controllers [16][17]) and superordinate control systems for configuration and process and status data exchange. The authors in [18] present a real-time capable, event-driven architecture to efficiently integrate smart tools within a modern factory. A special case of smart tools is given by *Instrumented Tools* as introduced in [19]. These tools are equipped with sensors to measure process data, and a wireless tracking system in order to enable recording of the tools motion trajectories. An example outline of such a tool is depicted in figure 2.



**Fig. 2.** Hardware setup of the Instrumented Tool, including tracking and force/torque sensors [19].

### 3.4 Semantic Knowledge Representation and Processing

Databases like MongoDB [20] or CouchDB [21] make use of W3C standards, e.g. JSON or JSON-LD, to organize contents using document based management systems. Standard features include free-text based searching, distribution of data, and built-in data analysis algorithms. AllegroGraph [22] combines document based approaches with graph databases, is compliant with W3C/ISO standards including JSON, RDF(S), OWL [23] for expression of knowledge, and provides built-in OWL and RDFS reasoning engines. A similar technology combination is supported by Neo4j [24] and GraphScale [25]. GraphScale enhances the graph-database Neo4j with OWL reasoning [26]. Within the domain of robotics, Knowrob [8] builds on a OWL/RDF knowledge representation, combined with enhanced reasoning and inference capabilities based on Prolog rules. OWL and RDF graph based knowledge management systems [26] require thorough understanding or modelling techniques. GraknAI [27] uses higher level schema to express knowledge based on hyper-graphs. The developers explain [28] that this modeling schema is more intuitive and scalable for DB design. At the same time, built in algorithms enable similar reasoning features than possible with OWL/RDF [29].

## 4 Methods

This section deals with the development of a methodology to implement an event recognition framework, applicable to the application fields stated in Section 2.

### 4.1 Events and Semantic Representation of Domain Knowledge

As mentioned in Section 3.1 an Event describes an interaction of an agent, based on a detected *Action*, with a physical object. In previous work [7], the definition was extended to also cover interactions with other agents (e.g. human or robot). This modification was necessary in order to describe the potential of events affecting the environment more clearly. Moreover, this adaptation include the possibility to associate a temporal-condition with the event-related action, which

needs to be fulfilled to satisfy the event definition. For example an event-related human action "pick" needs to be executed for a duration of ten seconds. We propose to describe temporal-conditions using Allen's Interval Algebra [30], due to their generalized structure and simplicity. By using a representation according to [7], events can be considered as "*semantic entities*" that relate individual states of an *Assembly Process*.

In its simplest form, APs can be described as a series of (process) *States*, a set of *Events*, and a set of *Relations*. A *State* describes a certain, stable stage of an AP including the status of smart tools, and objects (spatial relations). A *Relation* describes how an event drives the progress of an AP further, by semantically relating two AP states with a corresponding event (see figure 3). A formal description of APs is provided in previous work [31][19]. Referring to the event description in [7], events issued through smart tools cannot be abstracted. However, we argue that similar to human actions (see definition in Section 3.1), smart tool actions can be defined. These actions include for example screwing (screwing device), or movement along a trajectory (trackable lifting assistance tool). Similarly, smart tools can interact with objects. Therefore, the definition of events is still sound, also if applied to smart tools.

As events affect the AP an abstraction of these effects, that conforms with the data structures of an AP, is necessary. We introduce the concept of *Consequences* to describe changes in spatial-configurations of objects (e.g. "*on-top-of*", "*inside-of*", or displacement), and changes of the "internal" state of tools or agents (see *csqSpatial* and *csqActStateChange* in figure 3). It is assumed that an event can generate an arbitrary number of consequences. Consequences "physically" advance the AP to a subsequent state. Finally, we introduce *logical Rules* to define in which "logical sequence" the generation of consequences will happen.

We summarize: An event is characterized through a) an related human or tool action including an optional temporal-condition to be satisfied, b) consequences it generates, and c) logical rules describing the sequence of consequence generation. Figure 3 depicts a simplified schema for event definition as explained and referred to above, modelled using GraknAI [27].

## 4.2 Conception of the Event Recognition System

Referring to the overview in figure 1, and to the event definition concepts in Section 4.1, event recognition relies on the perception data of the perception systems *Tool Proprioception (TP)*, *Object Recognition (OR)* and *(Human) Action Recognition (AR)*. Moreover, the definitions of events (offline defined) are acquired from the available AP domain knowledge. Figure 4 depicts a schematic overview of the *Event Recognition (ER) System*. The main functionality of the ER system is to continuously observe the perception data streams, and to evaluate the Logical Rules describing an event. A common problem of such recognition systems is to determine the correct point in time, when to start an event evaluation. In our previous work [31] the assumption as follows was made to overcome this problem: Both AR and TP perception sources serve as triggers for the evaluation procedure. We emphasize that events are related to the execution of actions, which

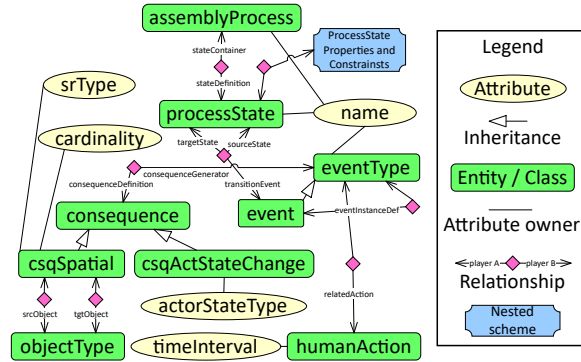


Fig. 3. Knowledge representation for event definitions, consequences and the AP.

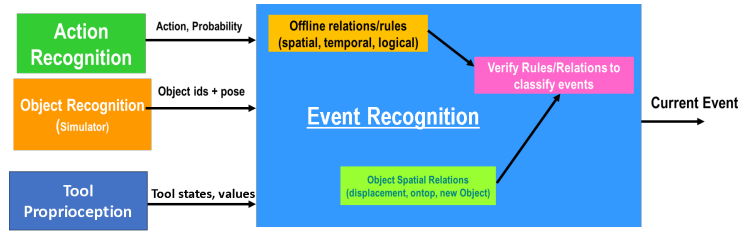


Fig. 4. Outline of the Event Recognition System.

in turn justifies this assumption. Therefore the beginning/ending of a detected human or smart tool action will activate evaluation procedures as summarized below.

- 1) **Loading of Event Definitions:** The event definitions (according to Section 4.1), are loaded from the AP knowledge.
- 2) **Reference Data Acquisition:** An initial detection of the start of an action, advances the event recognition to request the current depth image frame from the OR. Similarly, the TP and AR data will be stored for reference including timestamps. This data serves as *reference data*.
- 3) **Comparison Data Acquisition:** Upon detection of the end of the current action (i.e. different action detected), the same type of data will be acquired serving as *comparison data*.
- 4) **Evaluation of Logical Rules:** All logical rules for the given event definitions are evaluated in the defined sequence. The event definition having all rules fulfilled will be returned as classified.
- 5) **Generation of parameterized event:** Based on the classified event, an event instance configuration is created and returned, including parameterization. The event recognition process is continued at step 2 (repetition).



**Evaluation of Logical Rules:** The evaluation of logical rules (see Section 4.1) in order to classify offline defined events, is performed on based on comparison operations. While matching event-related human or tool action type requires a simple comparison of action type identifiers, the evaluation of spatial-consequences and temporal-conditions is more complex. This is because recorded *reference data* and *comparison data* needs to be considered. Given the case of spatial-relations, the exact POSES (positions and orientations) of the relevant objects of interests are required. These are obtained by triggering the OR to recognize objects in both 3D depth frames of the reference and comparison data set. The returned POSE results are then used to compute spatial-relations for pairs of objects based on euclidean operations. A spatial-consequence is then derived by e.g. comparing the spatial-relation of "Object A" towards "Object B" for the reference and comparison data set. Temporal-conditions are evaluated by comparing the timestamps of action beginnings and endings, again obtained from the reference and comparison data sets.

## 5 Applying Event Recognition to Selected Application Areas

In Section 2, potential application areas of industrial assembly were identified, where event recognition is considered a valuable technology, to solve challenging realization problems. Common to these application is the major goal of providing cognitive assistance and support to the human in the loop. Each application poses different requirements to event recognition, and variations in the system architecture. This section focuses on the specificities of applying event recognition to each application area.

### 5.1 Quality Assurance

The purpose of this application is to track whether certain steps during an AP did happen, in order to ensure high quality of the product and to avoid necessary disassembling actions at a later time. All steps necessary to be tracked can be represented as events to be modelled in the domain knowledge base, following the event definition methods. Based on the defined events, the main idea is to present the relevant events to the human worker, using a graphical user interface (GUI). Figure 5 depicts a mock-up of the visualization and shows the application concept. In general the visualization presents a configurable list of steps (i.e. events) that need to happen during the AP. One can distinguish among two phases. During the first phase, "Continuous Event Detection" the user can query detailed information about the steps. All steps are highlighted in yellow colour in the beginning, meaning that the related events are not yet detected. During the process execution, the event recognition system tries to observe relevant events. Upon detection, the respective step is coloured green. At the end of the AP, during the "Process Review" phase, the human worker gets notified about events that were not detected. After checking, the user needs to manually confirm remaining events.

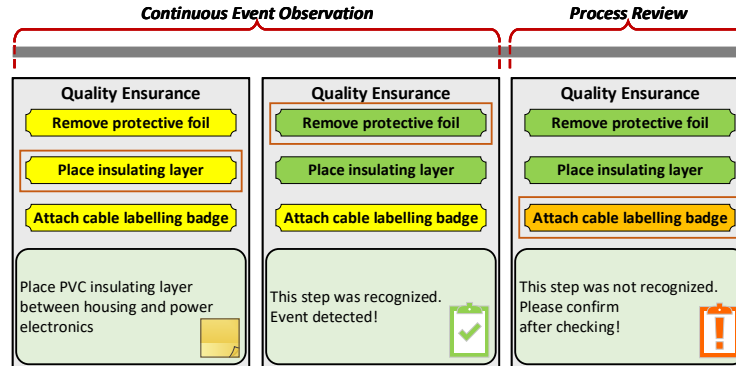


Fig. 5. Visualization mock-up of the Quality Assurance system.

**Specifics of the System Architecture:** In terms of process knowledge, offline event definitions are required. The event recognition system observes all possible perception sources as given in figure 1. A GUI frontend is necessary to visualize the status intuitively for the user.

**Requirements:** Strict avoidance of "false-positive" detections, i.e. marking as "green" events, which did not happen, is considered the worst case. A low number of "true-negative" detections is preferred as asking the human to confirm undetected events impacts efficiency of the entire process. Process steps of interest often include **non-rigid** objects (e.g. cables, foils), thus resulting in a major challenge for OR systems.

**Perception Sources:** The required perception sources for event recognition include primarily AR and OR, but also TP systems. Having an AR system with high detection quality available can compensate OR detections with low confidence. All process steps including smart tools, profit from generally stable tool status detections.

## 5.2 Worker Assistance

This application deals with modular assistance systems, configurable based on the given AP requirements context. Such an assistance system requires a cognitive architecture that is capable of perceiving, reasoning and to plan assistance actions [31]. Figure 6 shows the outline of the proposed architecture.

**Specifics of the System Architecture:** The core of this cognitive system architecture is formed through an *Situation Recognition* system, that reasons about the current state of the AP and the environment. The main goal is to identify situations, where the human worker needs support to complete the common goal,

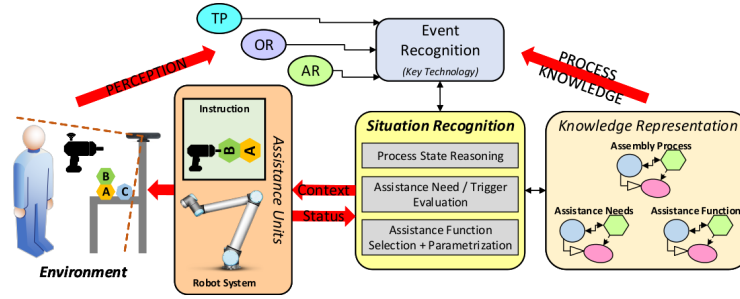


Fig. 6. Specific system architecture for the Worker Assistance application.

i.e. the AP. In order to achieve such a behavior additional domain knowledge, especially the description of (*Human*) *Assistance Needs* and *Assistance Functions* are required. The identification of assistance needs and functions is one of the main research questions targeted in the project MMAssist II [32]. Assistance functions refer to configurable capabilities of *Assistance Units* that are used to fill cognitive or physical gaps of the human operator in order to complete the task.

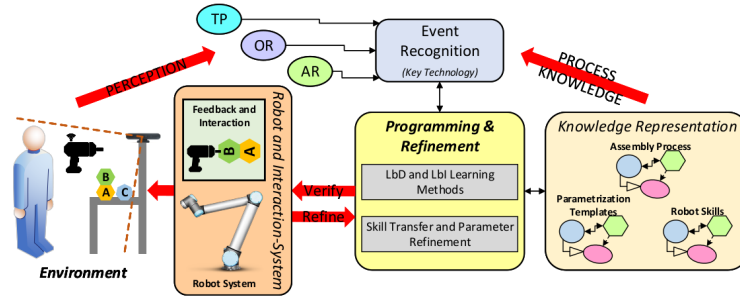
**Requirements:** The main requirements include extensive AP knowledge, to enable reasoning a) about the current process state, b) capabilities of the human, c) capabilities of the assistance units and d) effects of assistance functions. Reasoning about the current process state, requires stable perception systems in order generate proper hypotheses. Process knowledge can be used to provide context information to perception systems, e.g. to OR systems to locate objects that are of current interest in a given process state. Approaches are introduced by [13] and [14]. Our previous work [15] describes a method to enable context enhanced tracking of object in a cognitive architecture for AP reasoning.

**Perception Sources:** All introduced perception sources are of interest in this application area. The application of context information for enhanced parametrization is proven to increase confidence and stability of vision based perception systems.

### 5.3 Process Teaching and Configuration

Teaching an AP to robot system is a tedious task. The aim of this application is to combine aspects of *Learning by Demonstration (LbD)* and *Learning by Interaction (LbI)*, as demonstrated in our recent work [19], to significantly reduce programming effort. Figure 7 shows the outline of the proposed architecture.

**Specifics of the System Architecture:** The teaching process founds on two approaches as described in [19]: a) LbD using smart tools (TP), that are instru-



**Fig. 7.** Specific system architecture for the Process Teaching and Configuration application.

mented with sensors to capture trajectories and process data (e.g force / torque), and b) LbI by applying visual event recognition (OR, AR) to capture coarse process knowledge. An important concept for the second approach is *Skill Transfer and Parameter Refinement*. Skill transfer refers to the problem of "mapping" a sequence of detected and parametrized events (see step 5 in 4.2) to equivalent robot skills. This requires extended process knowledge of robot skills that are similarly described as events, i.e. based on Consequences (see section 4.1). The parametrization of the robot skills is initially derived from the parametrized events, by mapping equally typed parameters. In this context, parameter refinement refers to fine-tuning of a) initially derived robot skill parameters or b) additional process parameters (e.g. gripping force) required. This adaptation step is carried out based on GUI interaction with the human actor [19].

**Requirements:** The level of confidence and accuracy of visual event recognition is less critical than in the other applications (see Sections 5.1 and 5.2). Due to the interactive behavior of the proposed process, the LbI system can query the human to resolve ambiguities of detected events. Robot skill definitions as well as parameterization templates for these skills are additionally required to be defined in the domain knowledge.

**Perception Sources:** All introduced perception sources are of interest in this application area in order to realized both learning aspects: LbD and LbI. Specialized, trackable smart tools enable further possibilities of perception, such as physical parameters and motion trajectories. An implementation of such a tool is described in our recent work [19].

## 6 Implementation and Results

This section provides brief details on the implementation of relevant concepts and systems that were introduced in Section 4, and refers to the authors previous work.

## 6.1 Abstraction of Domain Knowledge

As mentioned earlier, abstraction of domain knowledge is a challenging and tedious task. In previous work [31] the Knowrob knowledge processing framework [8] was applied to abstract AP knowledge for a HRC assembly setting. Moreover, the built in reasoning infrastructure was extended and utilized to realize an architecture similar to figure 6. In recent work [19], GraknAI [27] is applied to abstract domain knowledge and to implement reasoning functionalities for human to robot skill mapping.

## 6.2 Action Recognition (AR)

Action recognition plays an important role to classify human events in an industrial assembly setting. The approach used in this work is based on Deep Learning and applies 3-dimensional DenseNets in Pytorch [33]. Densely Connected Convolutional Networks (DenseNet [34]) are extended into 3D DenseNet by adding a temporal dimension to all internal convolutional and pooling layers. For each layer, the feature maps of all previous layers are used as input, and custom feature maps are used as input to all subsequent layers. Feature extraction is realized using OpenPose [35], that enables real-time person-tracking and POSE estimation based on 2D image data. A total of five action classes (Pick, Place, Insert, Screw and Idle) were defined and classified with the AR module. In the training phase, a record is created by capturing multiple videos of the assembly covering the action classes. The videos are then "labeled" and split accordingly. The 3D DenseNet Model Network implemented in Pytorch will then be trained with the recorded data set. Once the model is trained, it can be used to predict the actions taken during a live stream or recorded video. Figure 8 shows an example result for an "Insertion" action of one object into another (see red circle in image). Test results show that the network is able to predict the action classes from the trained data. The maximum achievable accuracy is about 97%.



**Fig. 8.** Example output of the Action Recognition system for object insertion (see left). The right-hand side of the figure shows the feature extraction based on OpenPose and the classified result.

### 6.3 Event Recognition System

The Event Recognition System is implemented as a Python based module in ROS (Robot Operating System), reflecting the architecture introduced in Section 4.2. ROS message filters are applied in order to synchronize the perception data inputs originating from the AR, OR [15], and TP [19] modules. During startup of the application, a connection to the AP knowledge base is established in order to retrieve the available event definitions. Therefore, the event recognition is online configurable based on the current contents of the knowledge base. An open topic is the application of AP context (current state) to improve the confidence and stability of event detections.

## 7 Conclusion and Future Work

Reasoning about the current status of an industrial AP, is a true challenge and an important issue when researching context aware assistance systems for the production industry. This work discusses methods to realize event recognition in industrial assembly settings, and identified application areas that potentially benefit from this key technology. We can conclude that, despite the different application targets, there are only little differences in the proposed architectures to realize the considered applications. The existing results of the core technologies presented, encourage further development and improvement. The presented suggested approaches to apply event recognition in the considered applications are currently at a concept level, and implementation is part of present work. Therefore, the validation of these approaches and evaluation based on user studies, is a logical next step.

## 8 Acknowledgements

This research is funded by the projects MMAssist II (FFG, 858623), ASKRob (Austrian Institute of Technology), Bridge-Human Robot Interaction (Teach-Bots), Smart Factory Lab and DigiManu (funded by the State of Upper Austria).

## References

1. Ronald Poppe. A survey on vision-based human action recognition. *Image and vision computing*, 28(6):976–990, 2010.
2. Ahmad Jalal, Yeon-Ho Kim, Yong-Joong Kim, Shaharyar Kamal, and Daijin Kim. Robust human activity recognition from depth video using spatiotemporal multi-fused features. *Pattern recognition*, 61:295–308, 2017.
3. Natraj Raman and Stephen J Maybank. Activity recognition using a supervised non-parametric hierarchical hmm. *Neurocomputing*, 199:163–177, 2016.
4. Mingtao Pei, Yunde Jia, and Song-Chun Zhu. Parsing video events with goal inference and intent prediction. In *2011 International Conference on Computer Vision*, pages 487–494. IEEE, 2011.

5. Yingying Zhu, Nandita M Nayak, and Amit K Roy-Chowdhury. Context-aware activity recognition and anomaly detection in video. *IEEE Journal of Selected Topics in Signal Processing*, 7(1):91–101, 2012.
6. Karinne Ramirez-Amaro, Yezhou Yang, and Gordon Cheng. A survey on semantic-based methods for the understanding of human movements. *Robotics and Autonomous Systems*, 2019.
7. Sharath Chandra Akkaladevi, Matthias Plasch, Andreas Pichler, and Bernhard Rinner. Human robot collaboration to reach a common goal in an assembly process. In *STAIRS*, pages 3–14, 2016.
8. Moritz Tenorth and Michael Beetz. Representations for robot knowledge in the knowrob framework. *Artificial Intelligence*, 247:151–169, 2017.
9. Stefan Thalhammer, Timothy Patten, and Markus Vincze. Sydpose: Object detection and pose estimation in cluttered real-world depth images trained using only synthetic data. In *2019 International Conference on 3D Vision (3DV)*, pages 106–115. IEEE, 2019.
10. Sharath Akkaladevi, Martin Ankerl, Christoph Heindl, and Andreas Pichler. Tracking multiple rigid symmetric and non-symmetric objects in real-time using depth data. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5644–5649. IEEE, 2016.
11. Cheng-Hei Wu, Sin-Yi Jiang, and Kai-Tai Song. Cad-based pose estimation for random bin-picking of multiple objects using a rgb-d camera. In *2015 15th International Conference on Control, Automation and Systems (ICCAS)*, pages 1645–1649. IEEE, 2015.
12. David Joseph Tan, Federico Tombari, Slobodan Ilic, and Nassir Navab. A versatile learning-based 3d temporal tracker: Scalable, robust, online. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 693–701, 2015.
13. Jaume Amores, Nicu Sebe, and Petia Radeva. Context-based object-class recognition and retrieval by generalized correlograms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(10):1818–1833, 2007.
14. Thang Ba Dinh, Nam Vo, and Gérard Medioni. Context tracker: Exploring supporters and distracters in unconstrained environments. In *CVPR 2011*, pages 1177–1184. IEEE, 2011.
15. Sharath Chandra Akkaladevi, Matthias Plasch, Christian Eitzinger, Andreas Pichler, and Bernhard Rinner. Towards a context enhanced framework for multi object tracking in human robot collaboration. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 168–173. IEEE, 2018.
16. Deprag Schulz. Screwdriving controllers. <https://www.deprag.com/en/screwdriving-technology/products/control-technology/>, 2019.
17. Atlas Copco. Atlas copco power focus. <https://www.atlascopco.com/en-uk/itba/products/assembly-solutions/electric-assembly-systems/powerfocus-6000>, 2019.
18. Muhammad Umer, Bhargav Mahesh, Lars Hanson, Mahmood Reza Khabbazi, and Mauro Onori. Smart power tools: An industrial event-driven architecture implementation. *Procedia CIRP*, 72:1357–1361, 2018.
19. Sharath Chandra Akkaladevi, Andreas Pichler, Matthias Plasch, Markus Ikeda, and Michael Hofmann. Skill-based programming of complex robotic assembly tasks for industrial application. *e & i Elektrotechnik und Informationstechnik*, pages 1–8, 2019.
20. Kristina Chodorow. *MongoDB: the definitive guide: powerful and scalable data storage.* ” O’Reilly Media, Inc.”, 2013.
21. CouchDB Team. Couchdb 2.0 reference manual. 2015.

22. Inc Franz. Allegrograph: Semantic graph database. <https://allegrograph.com/products/allegrograph/>, 2017.
23. OWL Working Group et al. Owl 2 web ontology language document overview: W3c recommendation 27 october 2009. 2009.
24. Neo4J Developers. Neo4j. *Graph NoSQL Database [online]*, 2012.
25. Thorsten Liebig, Vincent Vialard, Michael Opitz, and Sandra Metzl. Graphscale: Adding expressive reasoning to semantic data stores. In *International Semantic Web Conference (Posters & Demos)*, 2015.
26. Lisa Ehrlinger and Wolfram Wöb. Towards a definition of knowledge graphs. *SEMANTiCS (Posters, Demos, SuCCESS)*, 48, 2016.
27. Enterprise Grakn Labs. Graknai: A knowledge graph. <https://grakn.ai/about>, 2019.
28. Haikal Pribadi. The grakn.ai ontology: Simplicity and maintainability. <https://blog.grakn.ai/the-grakn-ai-ontology-simplicity-and-maintainability-ab78340f5ff6>, 2017.
29. Szymon Klarman. A closer look at grakn.ai’s hypergraph datamodel. <https://blog.grakn.ai/modelling-data-with-hypergraphs-edff1e12edf0>, 2017.
30. James F Allen. Maintaining knowledge about temporal intervals. In *Readings in qualitative reasoning about physical systems*, pages 361–372. Elsevier, 1990.
31. Sharath Chandra Akkaladevi, Matthias Plasch, Srinivas Maddukuri, Christian Eitzinger, Andreas Pichler, and Bernhard Rinner. Toward an interactive reinforcement based learning framework for human robot collaborative assembly processes. *FRONTIERS IN ROBOTICS AND AI*, 5, 2018.
32. Christian Wögerer, Matthias Plasch, Manfred Tscheligi, and Sebastian Egger Lampl. Industrial assistance as an i4.0 topic—mmassist: Assistance in production in the context of human–machine cooperation. In *International Conference on Sustainable Design and Manufacturing*, pages 399–410. Springer, 2019.
33. Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.
34. Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.
35. Zhe Cao, Gines Hidalgo, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Openpose: realtime multi-person 2d pose estimation using part affinity fields. *arXiv preprint arXiv:1812.08008*, 2018.