A Simple Ontology of Manipulation Actions Based on Hand-Object Relations

Florentin Wörgötter, Eren Erdal Aksoy, Norbert Krüger, Justus Piater, Ales Ude, and Minija Tamosiunaite

***Abstract*—Humans can perform a multitude of different actions with their hands (manipulations). In spite of this, so far there have been only a few attempts to represent manipulation types trying to understand the underlying principles. Here we first discuss how manipulation actions are structured in space and time. For this we use as temporal anchor points those moments where two objects (or hand and object) touch or un-touch each other during a manipulation. We show that by this one can define a relatively small tree-like manipulation ontology. We find less than 30 fundamental manipulations. The temporal anchors also provide us with information about when to pay attention to additional important information, for example when to consider trajectory shapes and relative poses between objects. As a consequence a highly condensed representation emerges by which different manipulations can be recognized and encoded. Examples of manipulations recognition and execution by a robot based on this representation are given at the end of this study.**

***Index Terms*—Manipulation action, manipulation ontology, scene graph, semantic event chain.**

I. Introduction

OBJECT manipulation is certainly one of the major germs of human cognition. Human hands can be used in a highly targeted way to alter the relations between several objects (e.g., putting two objects together by pick&place actions) or to modify the structure of an object, here many times even without a grasp (e.g., boring a hole into a soft surface).

In this context the question arises, which types of manipulations exist? And usually this has been phrased by asking: What can you do with "all the things in the world" (or with those recognized in a complex visual scenery)? An experienced human could indeed perform a mental simulation of the scenery and come up with many possible action plans. However, from a bottom-up, purely data-driven perspective, there is no answer to this question as things in a complex scene contain too many feature-combinations and there are far too many interpretations possible about the meaning of the different things in various action contexts. Thus, for an inexperienced agent a bootstrapping process is needed on which its experience (object-action memory) can be grounded. This, however, is very difficult, when considering "all things."

Possibly, not the question: "What can you do with all things?" but rather the simpler one: "What can you do with your hands?" underlies the process which has bootstrapped cognition along the phylogeny of our species and which is still bootstrapping ever baby's development into an experienced adult (for a very detailed review see [1]). As compared to the almost infinite number of possible feature combinations in the object domain, there are far fewer possible basic hand-shapes existing. As a consequence, ordering the space of manipulations starting from "the hand" is easier than when starting from "all objects". Thus, while the importance of hands in development is not a novel statement and rather widely accepted, it is quite amazing that very little has been done to arrive at a systematic analysis about "hand-actions."

This study will analyze manipulations from an abstract point of view and introduce a manipulation ontology tree based on graph sequences, where each graph represents the touching relations of the different manipulated objects. The manipulations as presented by the ontology tree can also be used as a powerful abstract data-representation of manipulations (called the Semantic Event Chain) for robot applications (as shown in some recent studies in [2] and [3]). Based on the first, theoretical part we will then indeed show that one can define a metric where manipulations of the same class (based on the introduced ontology) appear closer together than manipulations from different classes. At the end of this paper we summarize some older experiments to provide examples demonstrating how this representation allows the recognition of human-performed manipulations. Finally we also show some robotic experiments, where a robot actually performs a simple manipulation based on the above introduced graph-like representation. These two sets of different experiments show that the apparently rather abstract way, in which we define the manipulation tree, does indeed lead to a useful representation.

A. State of the Art

As explained in more detail at the beginning of Section II, we will, in the context of this paper, centrally focus on hand-actions that involve objects (not gestures, etc.). We would ask

Manuscript received April 10, 2012; revised September 28, 2012; accepted November 23, 2012. Date of publication April 03, 2013; date of current version June 08, 2013. This work was supported by the European Community's Seventh Framework Programme FP7/2007–2013 (Specific Programme Cooperation, Theme 3, Information and Communication Technologies) under Xperience Grant 270273 and IntellAct Grant 269959.

F. Wörgötter, E. E. Aksoy, and M. Tamosiunaite are with the Computational Neuroscience Department, Georg-August-University, Göttingen 37073, Germany (e-mail: worgott@physik3.gwdg.de; eaksoye@physik3.gwdg.de; minija@physik3.gwdg.de).

N. Krüger is with the Cognitive Vision Laboratory, University of Southern Denmark, Odense DK-5230, Denmark (e-mail: norbert@mip.sdu.dk).

J. Piater is with the Institut für Informatik, University of Innsbruck, Innsbruck 6020, Austria (e-mail: justus.piater@uibk.ac.at).

A. Ude is with the Jozef Stefan Institute, Department of Automatics, Biocybernetics, and Robotics, Ljubljana 1000, Slovenia (e-mail: ales.ude@ijs.si).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TAMD.2012.2232291

our readers to keep this in mind, when we use the word action in the following.

Reasoning about action has a long tradition in philosophy (see [4] for a detailed account on action theories) and from this a diverse picture has emerged, where we can cite Aune directly to point out the major viewpoints [5, pg. 195].

> "Perhaps the most controversial aspect of so called action theory is its subject matter. This subject matter is generally said to be (or to concern) actions, but different philosophers conceive of actions in radically different ways. For some philosophers actions are abstract entities—states of affairs, propositions, sets, or even ordered pairs of some kind. For others, actions are distinctively concrete entities located in space and time. Another group of philosophers, among whom I include myself, have even denied that actions are required for a reasonable action theory, insisting that agents or actors will suffice as the theory's sole objects."

Following the mid-1980s, the interest on philosophical as well as cognitive aspects of action had a little bit died down and only more recently again there exist more large-scale attempts to formulate theories of action (e.g., [6]–[8]). A general agreement seems to prevail in the field that intentional, cognitive action (hence, not reflexes, etc.) requires "agency," much discussed already by Aune [4]. Intension based control of actions has, as a consequence, been implemented at least for the higher layers in a multitude of multilayer cognitive architectures for differently complex action-control (some examples are: [9]–[12]). Links to the possible cognitive architecture in humans have also been drawn [7]. Hence, with respect to cognitive actions, agency and intentionality have many times been (almost) equated. This manifests itself in the fact that a cognitive agent has to possess a deliberate first-person perspective against the objects in the world including other agents. Importantly, a relational view emerges from this in a natural way setting agent against objects and putting much emphasis on the interaction between both [see, e.g., 13]).

Originally this interaction has been considered as object-driven, where the structure and the shape of an object that determines what you can do with it (affordance principle, [14], [15]). Agency (or intentionality), on the other hand, suggest that—possibly even more importantly—it is the intended, the planned action, which lets you seek for suitable objects. Thus, it seems that, for a cognitive agent, objects and actions are inseparably intertwined [6], [16], which has been formalized by the concept of "object-action complexes" (OACs,[1] [17]–[19]). Turchin states [6, p.23].

> "Actions and, in particular, sensations are intimately tied to the agent, the subject of knowledge. An object is a transformation and prediction of actions."

Thus, modern accounts are now making specific use of the agency idea by more strongly emphasizing that actions, resulting from intentions, lead to specific relations between things, that the sequence of changing relations defines the use of the "thing," and that this ways it becomes an object. This is compatible with Turchin's claim that for a cognitive agent a (physical) *thing* in the world *only* becomes a meaningful *object* by its—planned and predictable—use [18], [19]. This tight link between actions and objects has been employed by some recent approaches as useful for the recognition of affordances in an action context [20], [21]. We will show that the manipulation ontology presented in this paper fundamentally relies on the same relational way of thinking. This has two consequences:

1) action components (movement primitives) are defined in natural way between those time-points where relations between objects change. This way, movement primitives are naturally constrained;
2) and, quite unexpectedly we can—for a long time—even drop the notion of specific objects entirely (above we had briefly mentioned that our "objects" will be just graph nodes) and derive the basic ontology merely from this relational point of view.

Graph nodes have also been used in other works to represent objects in an action context [22]–[24]. For example Griffith *et al.* [22] used scene graphs to analyze comovement relationships between the robot arm (manipulator) and (manipulated) objects. The graph nodes represent the tracked features of manipulator and manipulated objects. The edges are created when manipulator and manipulated object perform the same movements. As the arm manipulates objects, the graph structure changes with the movement patterns of the tracked features. Some ideas of this work will also play a role in our contribution.

Ontologies are heavily used in artificial intelligence research, computational linguistics, interface design, and many other fields. In general there are several different types of ontologies. Those that focus on objects and their relations [25], [26], while others more strongly emphasize actions [27], [28].

We will not attempt to discuss the wide field of general object-ontologies. In the context of this study only those are of interest, which describe objects for manipulations. Such ontologies have been mostly exploited in the field of designing virtual realities and multimodal interfaces [29],[30], [31], but only recently object ontologies started to be more strongly used in robotics [32]–[34].

With respect to an action-focus, a few general ontologies have been formulated covering the whole body [35], [36], and some subsymbolic approaches based on observation and clustering also exist [37]. To allow for sequencing, action grammars have been designed [38]–[40] and for manipulation description, different grasp-types have been classified [41]–[44]. Recently an interesting contribution has been made by Pastra and Aloimonos [45] who present a serious effort to provide a generative grammar of actions adhering to a minimalist approach. We cite from their paper (end of Abstract): Where…

> "… action terminals combine hierarchically into temporal sequences of actions of increasing complexity; the

[1] Going beyond Gibson's notion on Object-Affordances, the OAC concept puts a much stronger emphasis on the planned action, which "defines" an object. For example, a hollow cylinder with a solid bottom becomes a "beaker" when drinking from it. When you want to use it as a support, you can turn it around and it becomes a "pedestal."

> actions are bound with the involved tools and affected objects and are governed by certain goals. We [Pastra and Aloimonos] show, how the tool-role and the affected-object role of an entity within an action drives the derivation of the action syntax in this grammar and controls recursion, merge and move, the latter being mechanisms that manifest themselves not only in human language, but in human action too."

As shown later, the contribution of our paper will indeed relate to these ideas.

Thus, while some attempts to get a better understanding of the ontology of manipulation actions have been made [32], [33], [45]–[47], little has been done to try to find the basic structuring principles by which such an ontology could be bootstrapped.

To define these principles (if existing) would lead to a somewhat better grounded ontology and would be helpful to define: 1) which aspects of a manipulation are important; 2) which temporal phases of the action need to be accurate; and 3) which relations between the hand and the object are decisive for the outcome of the whole process.

II. General Properties of Manipulations

There are many actions that you can perform with your hand and we will restrict ourselves to those manipulations where hand and object interact so as to induce a change at the object (or at the object's relation to other objects). This, for example, excludes so-called *sensing actions* (feeling with your finger whether a liquid is hot) and some others (like *gestures*, etc.).

Also, we will only deal with the manipulations performed with one hand. This is due to the fact that many times bimanual manipulations can be performed one after the other with one hand or they fall into the category of support-actions, where the one hand is used to stabilize an object, while the other hand performs the manipulation. Genuine bimanual manipulations are much rarer and are often quite complex. The systematic understanding of single-hand actions presented in the following might, however, help to also understand those.

Furthermore, we will not talk about complex manipulations, for example the handling of a power drill, where the tool has its own actuator, or similar manipulations, but we will include a discussion about the handling of simple, primary tools. Clearly the transition from simple to complex tool handling is gradual, but this distinction will make sense in order to better understand manipulation ontologies. Early on we define here that *primary tools* are those that extend the functionality of the hand (e.g., hammer instead of fist) in a quantitative way. The implications of this will be discussed in detail below.

Also, we will only deal with simple objects, i.e., objects that do not have separate movable parts and are not deformable. Manipulating objects with movable parts would require a more complex ontology, as relations between object *parts* then can change (e.g., opening a book or opening a door). In this case we would find that complete objects are represented by more than one graph node, where the multiple nodes will have to stand for the object parts. Now two things can happen: Either the relation between parts changes (and—as a consequence—the corresponding subgraph will change) or the relation of this subgraph to other nodes (or subgraphs) can change. This might allow conclusions about object-part relations within an action context, where parts will take the role of objects. The complexity of the analysis, however, increases and we would, thus, like to leave this aspect for future investigations.

Another very important notion is that we will initially treat all manipulation as if one could perform them on any object. In our diagrams objects will be represented by little colored disks and we will define the manipulation ontology purely from the viewpoint of how an action changes the relations between objects. Only much later, when talking about our experimental work we will come back to the issue of objects.

A. Manipulation Types and Goal Categories

To structure the space of single hand manipulations we will make a few assumptions (rules), which are:

1) before the manipulation the hand is free (not touching anything);
2) at the start of the manipulation the hand touches only one object and the hand can—during the manipulation—not purposefully touch another object;
3) the manipulation can take place at the touched object itself or one other object can be a target, with which the first one is combined;
4) after the manipulation the hand is free again, this also defines a natural endpoint of the manipulation.

Thus, we have two or three entities, hand plus maximally two objects, which interact. We can now encode all entities by graph nodes and the fact that two entities touch each other by drawing an edge between these nodes. As objects can combine (or split—e.g., think of a cutting action), during a manipulation, object nodes can disappear (or new ones appear).

With this we can emulate manipulations in an abstract way by drawing sequences of such graphs, where between two graphs the touching patterns between objects (edges) or the objects themselves (nodes) have changed. These are the so-called *Key Events*, hence those moments in time where edges in the graphs are formed or deleted or new nodes emerge or disappear. Without even thinking about concrete manipulation examples, we can now construct all possible graph sequences without trivial loops or inversions (see Fig. 1) that obey the four rules from above, using maximally three nodes. While this looks terrifically abstract, we will soon see that all single hand manipulations are encoded by six graphs representing four *graph types* only, which can also be understood in more common terms.

This representation corresponds to a relational view onto manipulations. As shown in an earlier study, those few key events, where touching or untouching happens, are highly characteristic for a manipulation action and can be used to categorize manipulations in a model-free way [2], [3]. Hence, absolute space and time are often meaningless for a manipulation. For many manipulations it does not matter how fast they are performed or how much space is in between objects before they recombine with each other.

Fig. 1, top, shows that manipulations can be performed on one other object, where the hand interacts with it and then withdraws from it without involving another object. Stirring in a cup,

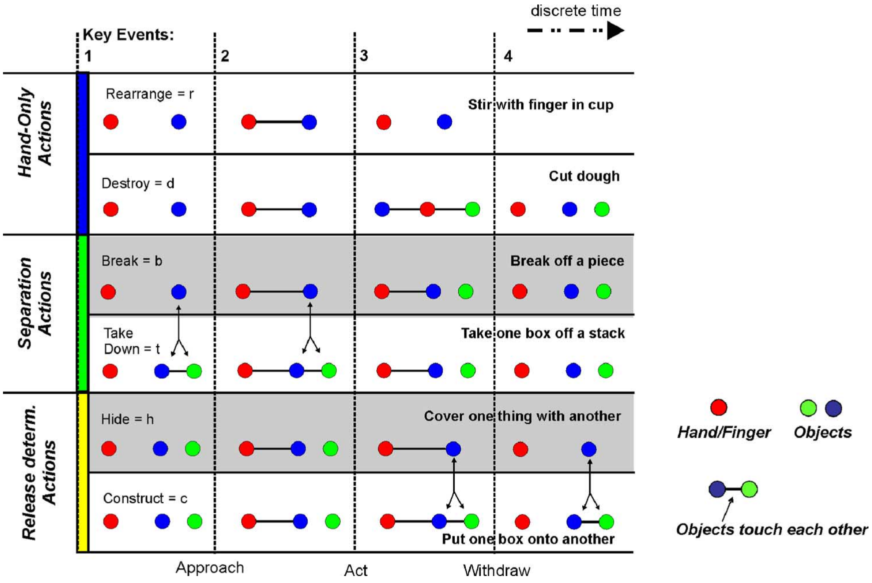


Fig. 1. Manipulation goal categories represented by graph sequences. Manipulation types indicated by the colored side-bars.

punching a ball, or cutting dough with your hand, are such examples. The first two graph sequences represent these manipulation types called "*Hand-Only Actions*" (blue side bar). Take, for example, the first line in this Figure. Here the finger moves towards a cup, then touches the liquid and performs the stirring until it is finally withdrawn again. Note, while nodes of all graphs in this paper are being annotated (colored), this is done only for graphical reasons. Distinctions between graphs arise entirely from their structural differences (nodes & edges). *Node or edge annotations are not needed!*

Evidently, with these graphs we are only representing the most fundamental relation between objects ("touching") and do not yet consider other important information like movement trajectories (stirring means to move your finger or a spoon in a circular way) or pose (the finger/spoon should be place orthogonal to the plane given by the cup opening). We will address these points below.

Many times these Hand-Only Actions are done by first grasping a primary tool and then performing the manipulation. The handling of a primary tool, however, does not alter the outcome of a manipulation in a qualitative way. It only introduces sometimes very strong quantitative changes (see section on tool handling). Stirring with a spoon is not fundamentally different from stirring with your finger.

Let us continue analyzing the other manipulations in Fig. 1: You can interact with one object and thereby create or set free another (part of an) object, for example ripping a piece off one thing, or uncovering a hidden object. In the same way you can also take off a box from a stack thereby freeing the box underneath. Fundamentally here you are separating graph nodes from each other, either by splitting a single node or by breaking an edge. The middle two graph sequences represent these manipulations, called "*Separation Actions*" (green side bar).

The opposite is true for the last two graph sequences. Here nodes are combined or edges formed. For reasons that will become clear later, we will call these actions "Release Determined Actions" (yellow side bar). The classical pick&place action belongs here.

Note, at the level of these graph sequence, separation actions are the time inverted versions of release determined actions. This will to some degree change when considering trajectories, too.

There are many ways to combine these manipulation types to create complex manipulation sequences. Thus, one should be careful to only consider manipulation primitives and break sequences up into the smallest reasonable manipulations types (that are being sequenced). Note, when performing a grasp (for example for putting two objects together) the grasp is not a manipulation-as-such. It is just the generation of a nonpermanent binding between hand and object: It is the building of a hand-object complex and the hand-object complex (as long as the grasp persists) now takes the role of the acting entity while some other objects in the world are possible target objects. Thus, we can remove the grasping problem (which is technologically a very hard one) from the analysis of our manipulation ontology. Later we will, however, come back to the grasping problem and to the often essential role of a grasp for the preparation of the most appropriate way to perform a manipulation (including the problem of tool-handling).

Furthermore, manipulations are performed to achieve certain goals. Essentially one finds six categories: Manipulations are used to Rearrange or to Destroy objects. There are also manipulations for Breaking and Taking-Down, as well as some for Hiding or covering objects or for Constructing something. These categories are therefore called manipulation goal categories (short: *goal categories*). Table I summarizes these considerations.

Triple arrows in Fig. 1 show that some graphs are pair-wise topologically equivalent. For example, the single blue graph node in the Breaking action (left side) is equivalent to the blue and green nodes, which are connected by an edge, in the Take-Down action below. This is interesting from a cognitive perspective: Breaking off a piece of something is very much the same as taking one thing off another thing. Unity versus duality is many times "in the eyes of the beholder". Think of two

TABLE I
MANIPULATION TYPES AND GOAL CATEGORIES

Type	Name	Description	Goal Category	Abbrev.	Examples
1	Hand-Only Actions	Hand alone acts on target object or Hand grasps tool and acts on target object	Rearranging	r	E.g. stir sugar into coffee, or push an object to free a space
			Destroying	d	Cut dough into pieces Chop wood
2	Separation Actions	Hand manipulates one object to either destroy it or to remove it from another object.	Break	b	Break off, rip-off, but also uncover
			Take-Down	t	Take down an object from another one
3	Release determined actions	Hand manipulates object and combines it with second, target object. (Many times this requires a grasp.) Combination of the objects is determined by the way the hand releases the grasped object.	Hide	h	Cover an object with another one
			Construct	c	Build a tower

tightly connected red Lego blocks. You can consider them as one entity and break one block off, or—if your vision is good enough—you could equally consider them as two blocks and take one down. Hiding behind these considerations is the famous "Binding Problem" (see, e.g., [48]). It is often a nontrivial question whether or not two connected entities represent one or two objects and the decision about unity versus duality often rests on tasks and goals rather than on physical facts.

Due to this topological equivalence and to make the now coming analysis easier we will sometimes neglect the grey panels, and focus on the other four main graph types. The examples covered by the grey panels are indeed also executed less often (by humans) as compared to their equivalent white partners, which justifies this step, too. The next following statement makes a somewhat astounding claim:

We believe that the manipulation ontology as well as the resulting data representations can be basically structured by using only those six (rather four!) fundamental graph sequences from Fig. 1.

It is not possible to arrive at rigorous proof of this claim, but we think it will be hard, if not impossible, to find a fundamental (not composed or chained!) manipulation which cannot be represented by one of these graphs.

Two questions, however, immediately arise: For which manipulations will space and time play a more concrete role? This will be discussed at the end of the paper.

And: Does the representation of "all manipulations" by just six graphs not just lead to a trivial concatenation, because many, very different manipulations will now look totally the same?

This problem is largely mitigated as soon as one considers these graphs in a more realistic context. Fig. 1 represents object relations as if the objects were floating in the air. Normally we deal with objects that are at least connected to "the ground," or often also to other objects (e.g., forming a pile, etc.). In real scenarios, this does indeed introduce enough "second-order" relations, which allow recognizing different manipulations with quite a high degree of reliability by their graph sequences (see [2] and [3]). Some examples will be discussed in Section VII, below. First, here—in the theoretical part of the paper—we will introduce the relation to the ground for some examples (background, Fig. 2), to show that more structure emerges.[2] We only discuss the four essential graphs. Objects (before lifting) touch the ground. The first two graph sequences (rearranging $= \mathrm{r}$ and destroying $= \mathrm{d}$) do not show any fundamental differences as compared to Fig. 1. Also constructing (c) and taking down (t) remain inverse to each other, but now there is an "air-phase" where the hand lifts one object off the ground (or off the pile). We will soon see that the constructing action (and vice versa for taking-down) this way subdivides into three subcategories, which are distinguishable by their graphs by either having an "air-phase" or not.

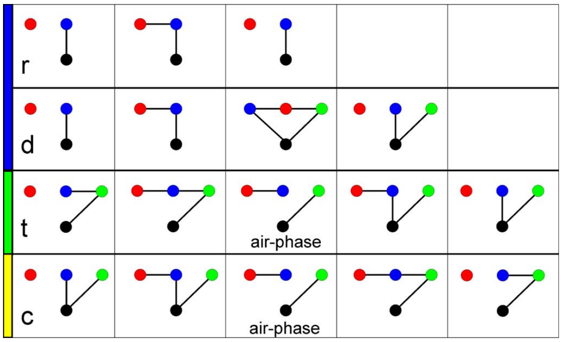


Fig. 2. Manipulation goal categories represented by graph sequences with background (black dots). Manipulation types are indicated by blue, green, and yellow side-bars.

Including the background is also interesting from a more subtle, theoretical point of view. If one wants to apply a very purist attitude, one could now say that building at tower is really composed of one taking-down action (picking a block up from the ground) and one putting down action (putting the

[2] Note, the label "background" is introduced here explicitly for the sake of theoretical argumentation (to distinguish reasoning with background versus reasoning without background), and for making the diagrams for the reader easier to follow. In actual manipulation recognition procedures, we are treating the background just as any other object. Hence background does not differ from the other objects to be manipulated or even the hand. Differences emerge only due to the different temporal touching patterns of the different objects (including the background).

TABLE II

DIFFERENT MANIPULATION INSTANTIATIONS. ABBREVIATIONS: D: DESTROYING, R: REARRANGING (CAN BE DISPLACEMENT, TOO, WHICH IS A FORM OF REARRANGEMENT), C: CONSTRUCTING, T: TAKING-DOWN, H: HIDING, B: BREAKING. BLUE, GREEN, AND YELLOW MARK THE DIFFERENT MANIPULATION TYPES. SOME MANIPULATIONS ARE QUITE DIFFICULT AND CAN TAKE DIFFERENT SHAPES. FOR EXAMPLE SCRATCHING AND SQUASHING/SQUEEZING CAN LEAD TO A NEW GRAPH NODE (FOR EXAMPLE SQUEEZING AN OBJECT TO THE DEGREE THAT IT FALLS APART) OR LEAD TO A SITUATION WHERE THE EXISTING OBJECT IS JUST DEFORMED (ONE GRAPH NODE REMAINS)

No.	Manip. Type	Goal Category	Instantiation	Comment
1	1	r	Punch / hit	with your hand an object
2	1	r	Flick	with your finger nail, quickly
3	1	r	Poke	with your finger tip, slowly
4	1	d	Chop	quickly, with the edge of your hand
5	1	r	Turn = bore (rotate wrist x)	a hole with your finger or your hand
6	1	d	Cut	slowly, with the edge of your hand
7	1	d/r	Scratch	with your finger nail
8	1	d	Scissor-cut / pinch	between your fingers
9	1	d/r	Squash, squeeze	inside your fist
10	1	d	Draw	with finger in sand
11	1	r	Push / pull-without-grasp	regular push, hook-pull, adduct
12	1	r	Stir	with finger
13	1	r	Knead	kneading dough, etc.
14	1	r	Rub / massage	with your hand someone else's body
15	1	r	Lever (rotate wrist y)	e.g. break open a hole
16	1	d	Scoop / ladle	fill your hand with liquid
17	2	t	**Take Down** or Pick apart	one block from a laterally connected group or a pile by pick&place
18	2	t	Push down or push apart	one block from a laterally connected group or a pile by pushing
19	2	b	Rip off	Rip a piece off an object
20	2	b	Break off	Break a piece off an object
21	2	b	Uncover by pick&place	Pick off an object to uncover another object
22	2	b	Uncover by pushing	Push off an object to uncover another object
23	3	c	**Put on top** or Put together	two blocks on top of each other or side by side by pick&place
24	3	c	Push on top or push together	two blocks on top of each other or side by side by pushing
25	3	h	Put over	Put one object above another one to cover it completely
26	3	h	Push over	Push one object above another one to cover it completely
27++	2,3	t,b,c,h	Some dynamic versions of 17-26	f. e. throw-in, nudge-together, flick-off etc.

block on another block).[3] Strictly this is true, but as the hand continues to hold the block until final put-down, we would argue that the two subcomponents do not really exist in their own right as "manipulations." Above we had defined the end of a manipulation as the moment where the hand gets free, which does only happen at the end of the two subcomponents. Thus, a further subdivision does not make sense from the viewpoint of our manipulation ontology.

B. Manipulation Instantiations

As discussed above, we can structure the analysis by the manipulation types as well as the resulting goal categories and furthermore ask: Which instantiations of Type 1–3 actions exist. Table II lists many such manipulations and hopefully this list is fairly complete, but—as it has been compiled just from our experience—it is open to additions. For Type 1 actions, it is ordered from top to bottom by the increasing contact duration of hand with object. On the right some explanatory notes are given, sometimes by ways of an example.

We note that in daily life combinations are possible where the underlying movement trajectories are fluently merged (like bore-and-push, etc.), but as we want to deal with primitives only we would like to keep these manipulations separate knowing, however, that smooth motion will require trajectory merging in such cases. Furthermore, we note that the manipulations marked with large letters are far more common as compared to the other possible version of the same action. Especially dynamic versions of these $(27++)$ are quite rare (e.g., throwing one object against another one is not very often observed if not when playing Péntaque). Thus these dynamic versions lead over to sports and games and will not be considered any further for our manipulation ontology. This is different for the dynamic versions of rearranging "r" and destroying "d," here we will find (see Table IV) that there are important manipulations existing which are performed rather often in every day life.

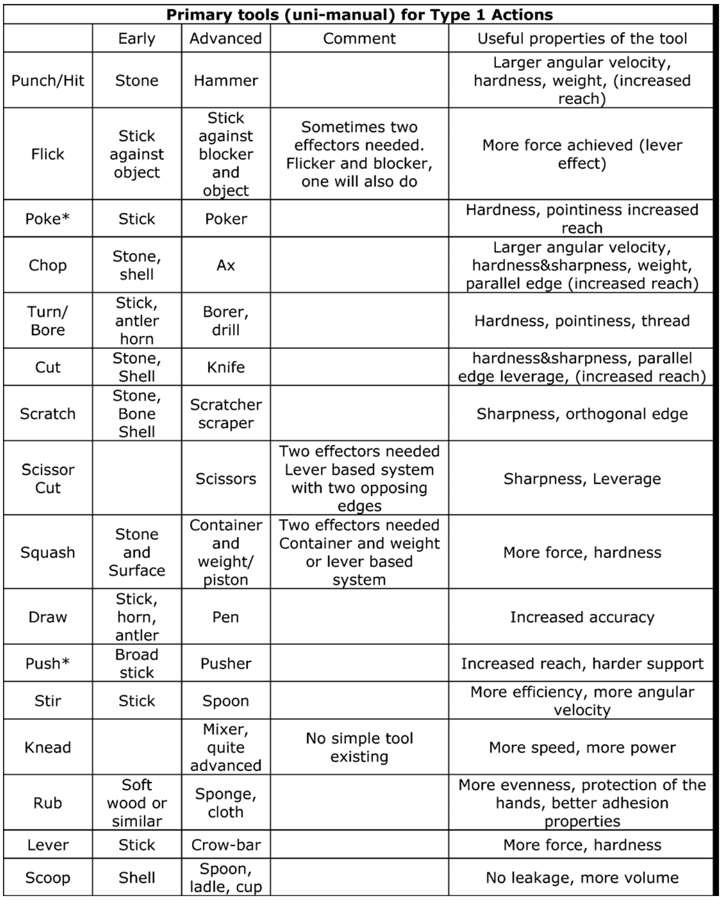
An important statement immediately arises. The different manipulations shown here are fundamentally determined by their graphs but, especially for manipulation type 1 (hand-only actions), also by the way the hand/finger(s) touches the object (relative pose) for some short time and by the movement trajectory for interacting with the object.

This information has to be added to the goal categories (graph sequences) introduced above (Fig. 1) and will require further structural analysis (see below).

It is interesting to note that very many actions fall into goal categories r or d: Almost everything we do is only done to rearrange or destroy entities.

[3] This is reminiscent to the second law of thermodynamics, where entropy has to increase somewhere if one wants to decrease it somewhere else.

TABLE III
HAND-TOOL ACTIONS ARE RELATED TO HAND-ONLY ACTIONS (TYPE 1) FROM TABLE I.
* PUSH IS USUALLY USING A FLAT SURFACE; POKE A POINTY SURFACE OF THE ACTOR



Primary tools (uni-manual) for Type 1 Actions				
	Early	Advanced	Comment	Useful properties of the tool
Punch/Hit	Stone	Hammer		Larger angular velocity, hardness, weight, (increased reach)
Flick	Stick against object	Stick against blocker and object	Sometimes two effectors needed. Flicker and blocker, one will also do	More force achieved (lever effect)
Poke*	Stick	Poker		Hardness, pointiness increased reach
Chop	Stone, shell	Ax		Larger angular velocity, hardness&sharpness, weight, parallel edge (increased reach)
Turn/ Bore	Stick, antler horn	Borer, drill		Hardness, pointiness, thread
Cut	Stone, Shell	Knife		hardness&sharpness, parallel edge leverage, (increased reach)
Scratch	Stone, Bone Shell	Scratcher scraper		Sharpness, orthogonal edge
Scissor Cut		Scissors	Two effectors needed Lever based system with two opposing edges	Sharpness, Leverage
Squash	Stone and Surface	Container and weight/ piston	Two effectors needed Container and weight or lever based system	More force, hardness
Draw	Stick, horn, antler	Pen		Increased accuracy
Push*	Broad stick	Pusher		Increased reach, harder support
Stir	Stick	Spoon		More efficiency, more angular velocity
Knead		Mixer, quite advanced	No simple tool existing	More speed, more power
Rub	Soft wood or similar	Sponge, cloth		More evenness, protection of the hands, better adhesion properties
Lever	Stick	Crow-bar		More force, hardness
Scoop	Shell	Spoon, ladle, cup		No leakage, more volume

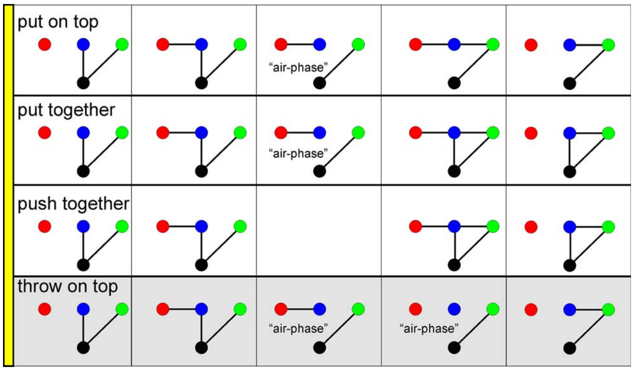


Fig. 3. Manipulation instantiations for the goal category "constructing" (Entries 23,24 in Table II.) All these manipulations belong to type 3 (yellow). In gray there is the dynamic action of throw-on-top related to put-on-top, above.

Fig. 3 shows the graphs for some different cases of constructing (entries 23, 24 in Table II). Taking-down is just the inverse. Only when considering the background these cases are graph-distinguishable! Otherwise they are all corresponding to panel "construct" in Fig. 1. Note, for "push-together" all objects remain constantly on the ground and there is no air-phase. Furthermore, as mentioned above, there are many times kinematic as well as dynamic versions of similar Type-3 actions existing. "Throw-on-top" (more common is "throw-in") is the dynamic version of "put-on-top." In this case the hand lets go of object one before it touches object 2 and a second air-phase emerges. But these rare actions shall not be considered any further.

C. *The Role of Primary Tools*

Furthermore, note that for all of the above Type 1 actions *primary tools* exist, with which the same action can be performed (in a more efficient way) now using not only the hand-on-its-own but, instead, a hand-tool complex. There is, however, no fundamental difference between such hand-only and the corresponding hand-tool actions. Use your hand to scoop something up or hold a ladle and use this to scoop; use your finger to bore a hole or hold or use a drill, and so on. Thus primary tools offer a change of quantity not quality. The following table (see Table III) summarizes some early and more advanced primary tools that have been invented to extend and improve the hand-only actions from Table II. Note also Table III is not necessarily complete.

D. *More About Grasping and Construction Actions*

Grasping has been much linked to cognitive development (e.g., [1] and [49]). While this is generally agreed in the context of our article grasping needs some different considerations, too, asking: What does a grasp do?

TABLE IV
DIFFERENT DEMANDS ON TRAJECTORIES, POSES AND HAND-SHAPES FOR THE ABOVE INTRODUCED MANIPULATION ACTIONS. HAND SHAPES CAN BE FLAT (FLAT HAND), POINT (FINGER TIP), POINT, NAIL (FINGER NAIL USED WHEN FLICKING), EDGE (EDGE OF HAND OR FINGERS), FIST (FIST), HOOK (HOOK WITH ONE OR MORE FINGERS), HOLLOW (LIKE A LADLE) OR THE HAND CAN GRASP AN OBJECT (GRASP). ONLY THE MOST GENERIC HAND SHAPES ARE CONSIDERED. HUMAN CAN SOMETIMES DO THINGS DIFFERENTLY, THOUGH. GRASPING AND RELEASING CAN SOMETIMES BE BALLISTIC/DYNAMIC, BUT THIS IS NORMALLY NOT EXACTLY ASSOCIATED TO A BUILDING-ACTION AND JUST LISTED FOR THE SAKE OF COMPLETENESS HERE, TOO

Type	Goal Category	Instantiation	Hand-Shape	Trajectory			Pose
				Approach	During	Withdraw	
Type 1 Hand-Only Actions	REARRANGE	Hit / Punch	Flat/Fist	Dynamic	Fleeting	Trajectory largely irrelevant	Relevant between hand and target object on touching
		Flick	Point, Nail				
		Poke	Point				
		Turn = bore	Point	Kinematic	Trajectory critical		
		Push/pull w.o. grasp	Point/ Flat/ Fist/ Hook				
		Stir	Point				
		Knead	Fist				
		Rub / massage	Flat				
		Lever	Flat				
	DESTROY	Chop	Flat/Edge	Dynamic	Fleeting		
		Cut	Flat/Edge	Kinematic	Trajectory critical		
		Scratch	Point				
		Scissor-cut Pinch	Edge				
		Squash Squeeze	Flat/Fist				
		Draw	Point				
		Scoop / ladle	Hollow				
Type 2 Separation Actions	TAKE-DOWN	**Take down** or pick apart	Grasp	Kinematic	Trajectory non-critical	Trajectory irrelevant	Largely irrelevant
		Push apart or pushdown	Flat/Point	Kinematic or rarer dynamic (e.g. flick)			
	BREAK	Rip off	Grasp	Kinematic	Trajectory critical		Grasp det-ermined
		Break off					
		Uncover by pick& place	Grasp	Kinematic	Trajectory non-critical		Largely irrelevant
		Uncover by pushing	Flat/Point	Kinematic or rarer dynamic (e.g. flick)			
Type 3 Release Determined Actions	CONSTRUCT	**Put on top** or Put-together	Grasp	Kinematic or rarer dynamic (e.g. throw)	Trajectory relevant just before release	Trajectory relevant just before release	Relevant between target objects on release
		Push-together	Flat/Point	as above			
	HIDE	Put Over	Grasp	Kinematic or rarer dynamic (e.g. throw)			
		Push Over	Flat/Point	as above			

Essentially there are only three goals for a grasp (called *grasp goals*):

1) a grasp can lead to the replacement of a hand-only action from Table II with a hand-tool action in Table III above. (After you have grasped a tool you can use it now);
2) it can be a grasp for performing a separation action (Type-2) of Table II. (e.g., grasp and rip, or grasp and turn);
3) it can be a grasp fundamentally performed to be leading to a release of the grasped object;

The release can take two forms:

a) put-on (guided release, largely kinematic);
b) drop, throw (ballistic release, largely dynamic).

Arguably nothing else can happen with a grasp as far as it concerns one hand manipulations analyzed here!

Grasp goal 1 is fully captured by the specification demands from Type 1 actions in Table II, above. As discussed above grasping a primary tool does not alter the corresponding hand-only actions in a qualitative way.

In grasp goal 2, the grasp takes a preparatory role for the subsequently happening action, where this action is again taken from the actions in Table II (mostly Type 2).

In general, however, this shows that grasp goals 1 and 2 are much related to the execution of hand-only actions or separation actions. Thus, the performed grasp does not induce anything conceptually different from what we had discussed in conjunction with the Type 1 and 2 actions from Table II.

This leaves us with grasp goal 3 and brings us to the last set of actions (23–26) in Table II. If we have not overlooked something, this fundamentally shows that a put-on (and its pushing-variants) are the only manipulation actions that lead to a long lasting constructive basic-relation between objects. All other constructive relations are specifications (for example of "pose") of the resulting touching relation between the two objects involved, which follows from a put-on action.

Ultimately what counts is that the two objects, with which the action is concerned, connect in the right way. This, however, brings us back to the question which spatial and tem-

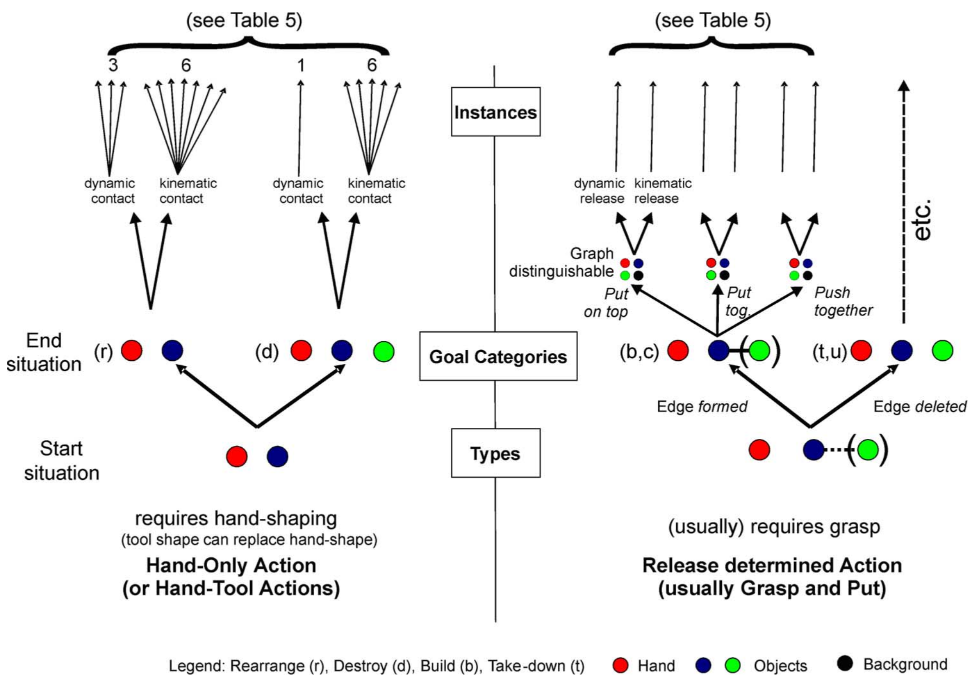


Fig. 4. Manipulation Ontology Tree.

poral aspects of a constructive manipulation are required. We observe that fundamentally the 3D-Pose of the two objects *relative to each other* is important, which usually results from a certain way (trajectory) of combining the one with the other. But for which moments in time is this information required? Naively, one would think: When grasping! But this is not true. The essence of a put-on manipulation results from the action phase immediately before/at release of the grasped object. The grasp only influences the outcome of a construction action indirectly or sometimes not at all.

This explains in more detail, why we had called (see Table I) construction a *release-determined action* (see Table I). It is fundamentally always an *ungrasp*. A release determined action is most often a pure ungrasp which follows a kinematic movement, but it can contain dynamic aspects, for example when throwing one object onto another one or resulting from hand-only-like component, e.g., when clicking something on or when screwing something on. The latter cases again lead over to action sequences. Hence, we will not dig deeper into this.

III. Manipulation Ontology Tree: A Summary

Fig. 4 summarizes the above discussion on how one can subdivide the manipulation action space. A structured but very simple tree with only a few branches at the different levels has emerged, where the color coding is like in Fig. 1. If desired, one could consider the right side of this diagram (release determined actions) as the inverse of the middle (separation actions).

We found that:

1) There are only three fundamentally different manipulation types existing (bottom level of tree):
   i) hand-only-actions $(= \mathrm{hand\text{-}tool\text{-}actions})$;
   ii) separation actions;
   iii) release determined actions (e.g., grasping and putting down).
2) There are only six manipulation goal categories (next level of tree) and those are much related to the manipulation types, where Type-1 manipulations (hand-only actions) correspond to goal categories *rearrange* and *destroy*, whereas Type-2 (separation actions) corresponds to *break* and *take-down* and Type-3 (release determined actions) to *construct* and *hide*.
3) Manipulation Types and Goal Categories are distinguishable by their fundamental graphs. (When considering the background, more actions can be graph-distinguished, too, see Fig. 3).
4) Manipulations are determined by continuous and discontinuous phases:
   i) discontinuity comes from the temporary formation or deletion of contacts between objects (or hand and object);
   ii) the final discontinuity, when the hand gets free again, defines the end of the manipulation;
   iii) continuity is due to continuous (or repetitive [e.g., chopping]) contact between hand and objects. Here trajectory and pose information will be needed (discussed next).
5) Putting together (pushing together, etc.) of two objects are the only existing long-lasting constructive actions.
6) All other actions are short-lived and used for rearrangement, destruction or to separate entities.
7) There are dynamic as well as kinematic actions possible.
8) Primary tools do not play a special rule.

IV. Special Cases: on Topological Equivalence

Fig. 5 discusses some special cases when more than two objects are present in the environment (see case description on

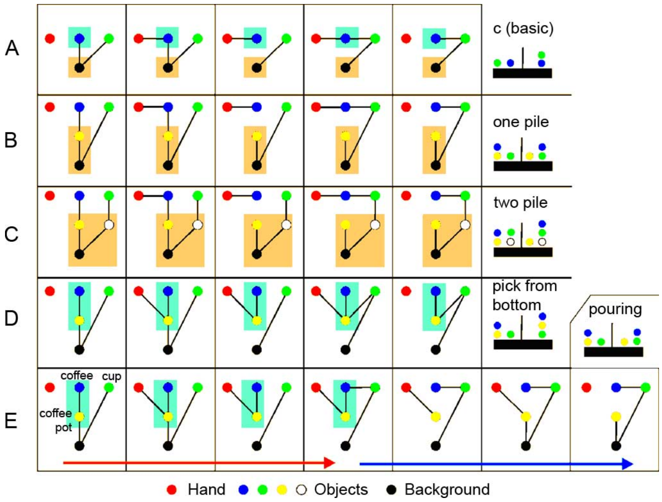


Fig. 5. Special cases as named on the right side of the graphs. Little sketches on the right show initial and end configuration of the manipulations. Orange and light-blue blocks depict topologically identical situations.

the right) to show that on the one hand there is a fundamental topological equivalence of all these cases to the basic graph sequences in Fig. 1, while—on the other hand—the actual graph sequences remain distinguishable from each other, which is important from an implementation point of view. Again we emphasize that it is not necessary to annotate the nodes. Graph sequences are distinguishable just by their node&edge structure.

Panel A repeats the basic graph sequence for "constructing" from Fig. 2. Colored boxes are used to depict the topologically equivalent structures. Here we see that panels B and C are equivalent to A by the orange box. Along a given sequence subgraphs inside the orange box do not change and can be replaced by one entity (e.g., the black node) from which one immediately sees that—after such a replacement—panels B and C are identical to A.

A similar conjecture holds for panel D and A. Here the light blue boxes depict similarity.

Case E, pouring liquid from one container into another container, is intriguing. The outcome of this manipulation is structurally identical to B, the "one-pile" case. The start of the sequence (first four graphs, red arrow) is topological equivalent to the first four panels in A (in fact the first three graphs of D and E are identical). To understand the remaining graphs (blue arrow), we need to consider that the coffee-pot is now put back on the table. This, however, is a "take-down" action. A close look at the take-down action shown in Fig. 2 reveals that indeed the last three panels of E are topologically equivalent to taking-down. Hence, "pouring" is really not a fundamental manipulation, but is better understood as a sequence of two.

These results indeed suggest that more complex situations seem to fall into the basic manipulation goal categories defined above. At least we have so far not found any exceptions.

V. The Role of Trajectory, Relative Object Poses, and Hand (Tool)-Shapes

In this analysis we have several times come across the question when and to what degree trajectory and pose information is required for understanding and executing a manipulation. Note, when using the word trajectory, we mean position, velocity and acceleration, each of which may or may not play a role for a given manipulation.

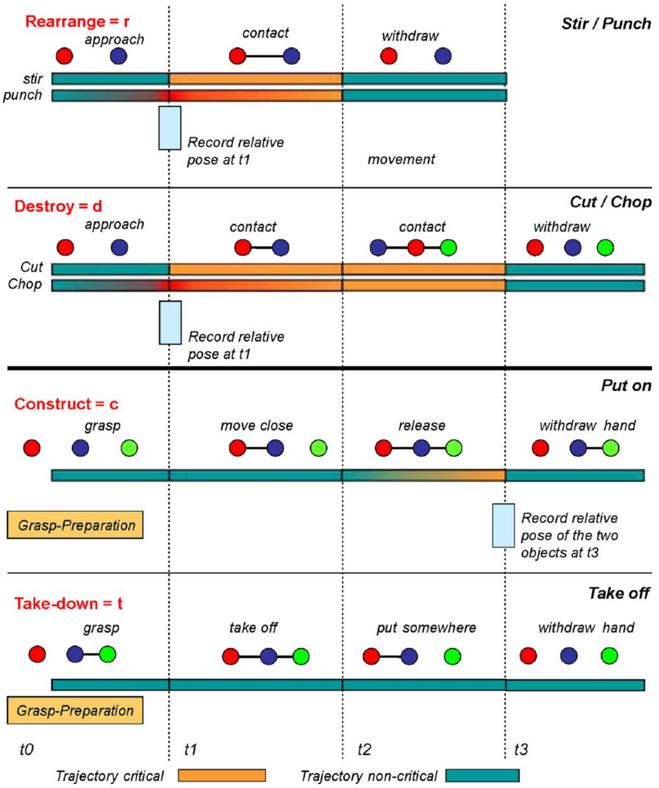


Fig. 6. Time bars for trajectory and pose information. Red tinge indicates that this trajectory contains a dynamic component.

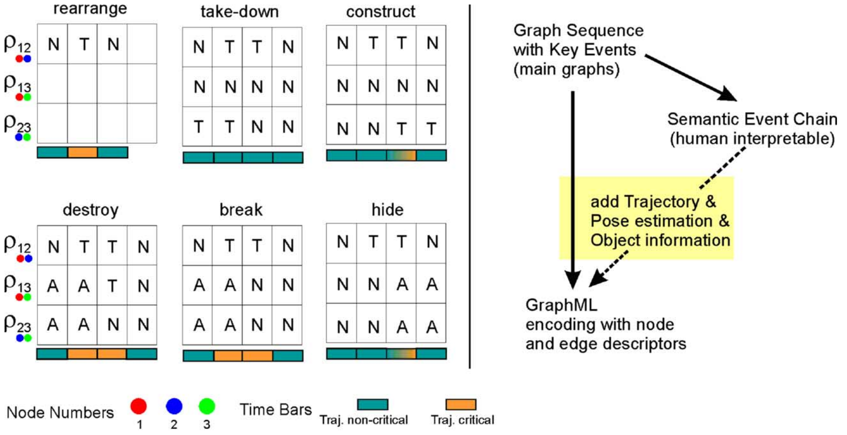
Evidently the example in the top row of Fig. 1 cannot be understood as "stirring" unless we also consider the movement of the finger (or spoon) in the cup. The graph sequence just tells us that hand and object have touched and later untouched each other. These moments provide us with the temporal anchor points for pose and trajectory analysis.

Thus, to understand the role of trajectories and (relative) poses we combine the information from Table II with that presented in Fig. 1 and add time-bars to the Key-Sequences showing the results in Fig. 6. We use the four main graph sequences from Fig. 1 and discuss the same examples as above.

We note at first that the different segments (intervals between Key Events) are not of equal length. For stirring the second segment will last for quite some time, for punching it will be (infinitely) short.

Furthermore we can see that "constructing" and "taking-down" produce differently structured time-bars as compared to "stirring/punching" and "cutting/chopping."

The latter contain long trajectory-critical phases (orange) and the pose between hand (or tool) and object has to be known at the moment when it touches the object (indicated by the position of the light-blue boxes). For a punching movement it is necessary to hit the target object hard, whereas for a stirring movement the way of approach does not matter. The same holds true for chopping versus cutting. Thus, the red tinge at the time bars indicates that some of these manipulations need a dynamic component.



(a) (b)

Fig. 7. (a) Semantic Event Chains belonging to the six graphs from Fig. 1 (b) Representation of the different algorithmic stages.

On the contrary, and as mentioned already above for "constructing" the trajectory is critical just before and at release. Here we will discuss on the commonly performed kinematic versions of these actions (thus, time bars do not have any red tinge here). Furthermore, the pose between hand and grasped object is not very important (if not for preparatory reasons), but the relative pose between both combined objects needs to be correct at the moment of release. For taking-down neither trajectory nor poses are important (if just putting the taken-off object down somehow without considering it further). Note, in those two cases—constructing and taking-down—we are fundamentally also faced with the grasping problem, which is, however, not part of the manipulation analysis.

Table IV subsumes the characteristics of pose and trajectory for the different cases introduced in Table II and in Fig. 6. In addition one column is added that shows how the hand should roughly be shaped when performing a given manipulation. Alternatively one could use a tool, which emulates the required hand-shape.

VI. Data Structures for Manipulation Recognition and Execution

As suggested above we can structure a manipulation representation by the sequence of key events that describe which objects touch each other (see Fig. 1) and add to this the additionally required information on pose and trajectory (see Fig. 6).

A. Introducing Semantic Event Chains

To this end we had in an earlier study introduced the so-called semantic event chain (SEC) representation [2], [3]. This is a matrix where every column corresponds to a key event and the rows represent the relations between pairs of graph nodes, which are the edges of the graph. We have three fundamental relations: A = Absent (node does not exist in this key frame), N = no-touching between nodes (no edge between nodes), T = nodes touch each other (there is an edge between nodes). In the older study we had also defined an overlapping relation, which is not relevant here when analyzing 3-D relations.

For the six graphs in Fig. 1 we get the following SECs [see Fig. 7(a)]. Most consist of four Key Events (columns) and three rows, due to having three graph nodes. Only the first, simplest one—rearrange – is different. Simplified time bars from Fig. 6 are shown on the bottom of each SEC. This demonstrates that the key frames provide us with temporal anchor points for recording additional information such as trajectory shapes and poses. This is an important notion, because it allows focusing additional analysis only onto certain moments in time.

SECs, thus, represent a human-interpretable representation of a manipulations action [see Fig. 7(b)]. This representation is very useful because the tabular view created by a SEC makes it possible to essentially "see how a manipulation basically looks like." The naked SECs however, do not contain information about pose and trajectory and they are not directly machine-readable. Please see the APPENDIX for the complete representation, which is then a machine readable format of the SEC including pose, trajectory and object information at the temporal anchor points. GraphML code, mentioned in Fig. 7(b).

B. Manipulation Similarity Measurement

Here we show how one can measure the similarity of semantic event chains and thus recognize different manipulations from observation up to the goal category (as given in Fig. 4).

To measure similarity two semantic event chains are compared with each other by analyzing their row-column relations using simple substring search and counting algorithms. We are searching for correspondences between the two event chains by first comparing rows. Here we allow shuffling and select for all rows in the first SEC their most similar counterpart in Section II. Then the search process counts equal entries in the corresponding rows. Here we use a standard substring search which does not rely on the dimensions of the SECs and, thus, allows comparing arbitrarily long manipulation actions by giving punishments for insertions and deletions. Then in the temporal domain the sequence of the columns is examined in a similar way to get the final similarity-measure result. For details see [3]. Note, in all real experiments [2], [3] we are using a similarity threshold of 65% and consider actions to be the same only if this threshold is passed.

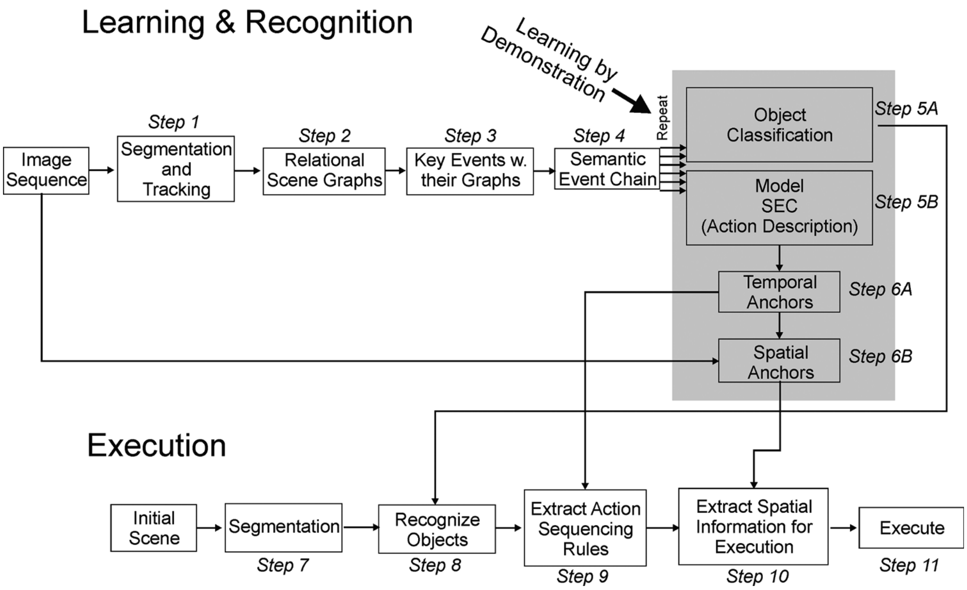
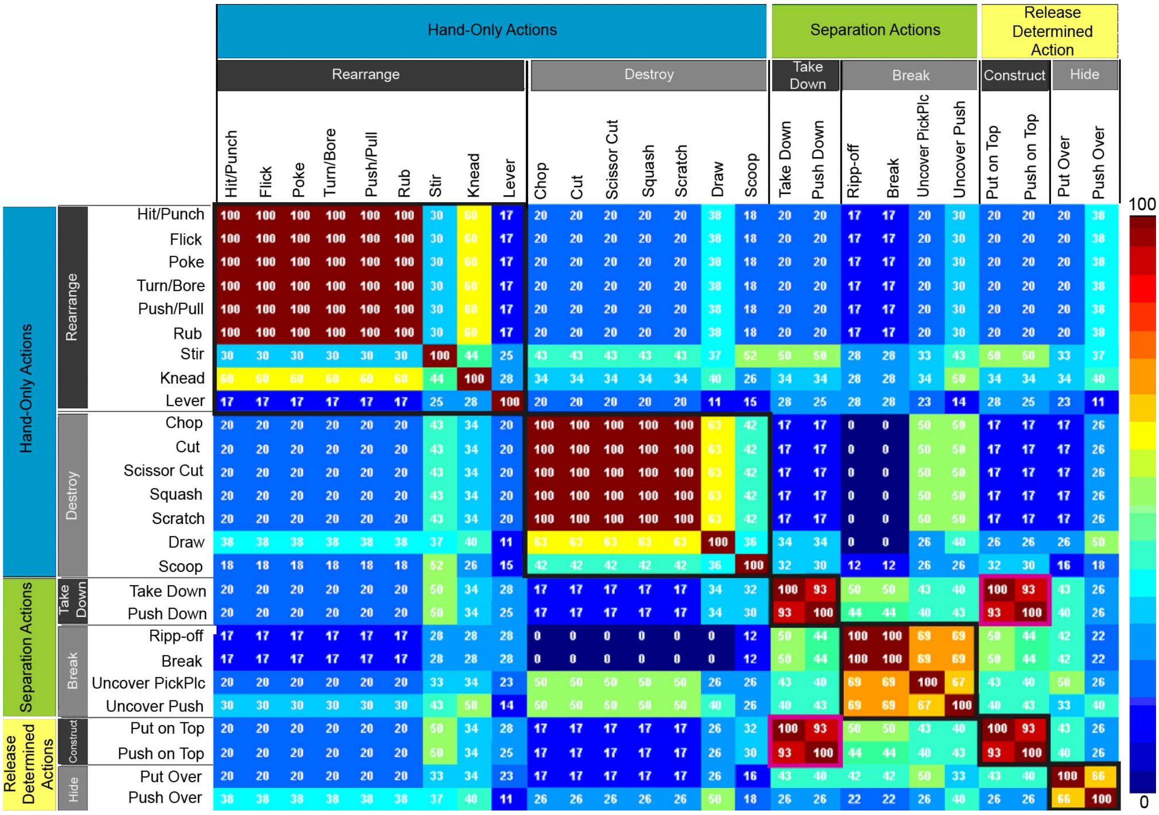


Fig. 8. Algorithmic procedure for manipulation learning, recognition, and execution.



			Hand-Only Actions																Separation Actions						Release Determined Action			
			Rearrange									Destroy							Take Down		Break				Construct		Hide	
			Hit/Punch	Flick	Poke	Turn/Bore	Push/Pull	Rub	Stir	Knead	Lever	Chop	Cut	Scissor Cut	Squash	Scratch	Draw	Scoop	Take Down	Push Down	Ripp-off	Break	Uncover PickPlc	Uncover Push	Put on Top	Push on Top	Put Over	Push Over
Hand-Only Actions	Rearrange	Hit/Punch	100	100	100	100	100	100	30	60	17	20	20	20	20	20	38	18	20	20	17	17	20	30	20	20	20	38
		Flick	100	100	100	100	100	100	30	60	17	20	20	20	20	20	38	18	20	20	17	17	20	30	20	20	20	38
		Poke	100	100	100	100	100	100	30	60	17	20	20	20	20	20	38	18	20	20	17	17	20	30	20	20	20	38
		Turn/Bore	100	100	100	100	100	100	30	60	17	20	20	20	20	20	38	18	20	20	17	17	20	30	20	20	20	38
		Push/Pull	100	100	100	100	100	100	30	60	17	20	20	20	20	20	38	18	20	20	17	17	20	30	20	20	20	38
		Rub	100	100	100	100	100	100	30	60	17	20	20	20	20	20	38	18	20	20	17	17	20	30	20	20	20	38
		Stir	30	30	30	30	30	30	100	44	25	43	43	43	43	43	37	52	50	50	28	28	33	43	50	50	33	37
		Knead	60	60	60	60	60	60	44	100	28	34	34	34	34	34	40	26	34	34	28	28	34	50	34	34	34	40
		Lever	17	17	17	17	17	17	25	28	100	20	20	20	20	20	11	15	28	25	28	28	23	14	28	25	23	11
	Destroy	Chop	20	20	20	20	20	20	43	34	20	100	100	100	100	100	63	42	17	17	0	0	50	50	17	17	17	26
		Cut	20	20	20	20	20	20	43	34	20	100	100	100	100	100	63	42	17	17	0	0	50	50	17	17	17	26
		Scissor Cut	20	20	20	20	20	20	43	34	20	100	100	100	100	100	63	42	17	17	0	0	50	50	17	17	17	26
		Squash	20	20	20	20	20	20	43	34	20	100	100	100	100	100	63	42	17	17	0	0	50	50	17	17	17	26
		Scratch	20	20	20	20	20	20	43	34	20	100	100	100	100	100	63	42	17	17	0	0	50	50	17	17	17	26
		Draw	38	38	38	38	38	38	37	40	11	63	63	63	63	63	100	36	34	34	0	0	26	40	26	26	26	50
		Scoop	18	18	18	18	18	18	52	26	15	42	42	42	42	42	36	100	32	30	12	12	26	26	32	30	16	18
Separation Actions	Take Down	Take Down	20	20	20	20	20	20	50	34	28	17	17	17	17	17	34	32	100	93	50	50	43	40	100	93	43	26
		Push Down	20	20	20	20	20	20	50	34	25	17	17	17	17	17	34	30	93	100	44	44	40	43	93	100	40	26
	Break	Ripp-off	17	17	17	17	17	17	28	28	28	0	0	0	0	0	0	12	50	44	100	100	69	69	50	44	42	22
		Break	17	17	17	17	17	17	28	28	28	0	0	0	0	0	0	12	50	44	100	100	69	69	50	44	42	22
		Uncover PickPlc	20	20	20	20	20	20	33	34	23	50	50	50	50	50	26	26	43	40	69	69	100	67	43	40	50	26
		Uncover Push	30	30	30	30	30	30	43	50	14	50	50	50	50	50	40	26	40	43	69	69	67	100	40	43	33	40
Release Determined Actions	Construct	Put on Top	20	20	20	20	20	20	50	34	28	17	17	17	17	17	26	32	100	93	50	50	43	40	100	93	43	26
		Push on Top	20	20	20	20	20	20	50	34	25	17	17	17	17	17	26	30	93	100	44	44	40	43	93	100	40	26
	Hide	Put Over	20	20	20	20	20	20	33	34	23	17	17	17	17	17	26	16	43	40	42	42	50	33	43	40	100	66
		Push Over	38	38	38	38	38	38	37	40	11	26	26	26	26	26	50	18	26	26	22	22	26	40	26	26	66	100

Fig. 9. Confusion matrix for action similarity. Similarity is measured in percents (see numbers inside each small square), red color stands for high similarity, blue color stands for low similarity. Blocks emphasized with black boxes show high similarity within the same action group, blocks emphasized in pink show similarity between "Put on top" and "Take down" actions.

C. Semantic Event Chains in Real Applications—Towards a Full Manipulation Description

In our real applications we do not use abstract objects ("colored disks") for the graph nodes, but image segments obtained from computer vision methods, where several image segments usually represent one object. As a consequence we get more nodes and bigger event chains [see Fig. 10(d)–10(f)]. Here certain subgraphs represent one object.

As an important finding we have observed that the touching and untouching behavior of these subgraphs relative to some other subgraphs is quite invariant against changes *within* a subgraph. What does this mean? Essentially this means that one can replace one suitable object (in a given manipulation

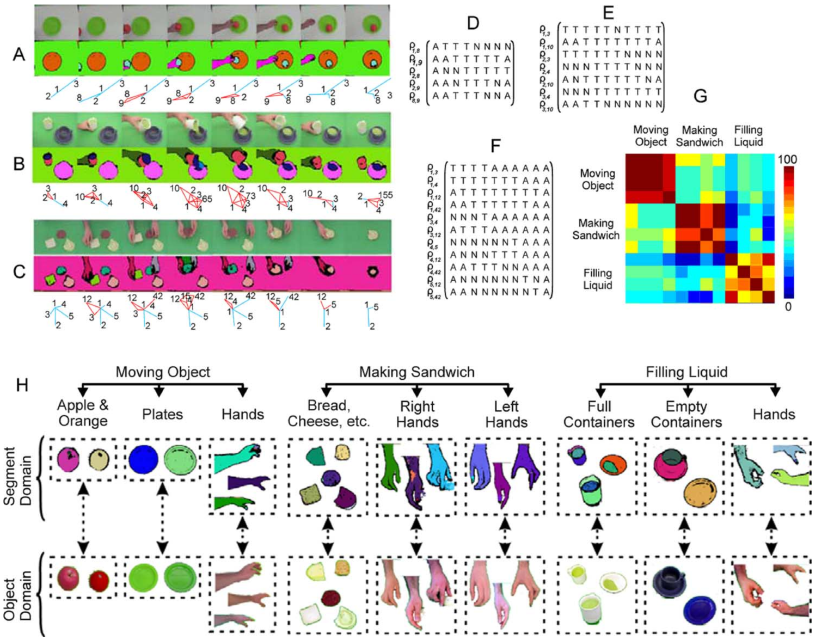


Fig. 10. Recognition Example. (A)–(C) Frames from example movies with segmentation results and graphs, (D)–(F) SECs belonging to these examples, (G) confusion matrix from a total of 12 different examples. (H) Image segment groups recognized as performing the same role in the manipulation and association to the actual images.

context) by any other suitable object (which may produce a much different image segmentation result and hence a much different subgraph). Such a replacement does not affect the essential structure of the SEC, which is given by the relational changes between the decisive nodes. This observation is expected from the theory outlined above. Complex event chains should ultimately be topologically equivalent to simpler ones. Thus, similar manipulations can be recognized by defining a metric similarity measure that hinges not on the details, but on the essential structure of the SEC.

This observation, however, also has a second important and very useful consequence. By observing the subgraphs, it is possible to actually recognize the objects that take a specific role in a manipulation. This addresses a so far missing vital aspect because it allows for the actual association of objects to a manipulation in a model-free way.

Fig. 8 describes the complete framework for Object Classification and Action Description (top part), where the system observes several movies showing the same manipulation (but possible with much different objects) and extracts from these repetitions the archetypical SEC (model-SEC, Step 5B) and the archetypical subgraphs (Object Classification, Step 5A, [2], [3]). This step is related to "learning by demonstration" ([50]–[54]), commonly used in robotics, only here we specifically focus on the extraction and the learning of a model SEC.

To arrive at a model SEC, one has to observe the same manipulation several times. The learning procedure is initiated by assigning zero weight values to all rows and to all columns of the first observation (first observed event chain). When observing the next manipulation, the event chains are compared by measuring their similarity as described above. For all similar entries, weights of the corresponding rows and columns are incremented by a small amount. If the new chain has additional or unobserved rows or columns the model is extended by these rows/columns, which start with the initial zero weight value. This is repeated for all observations. After this, weights are thresholded, deleting all rows and columns which are subthreshold and returning the resulting model event chain for this manipulation type.

The model-SEC defines the temporal anchor points (Step 6A), which are those moments in time when a discontinuous change happens (edge forming or deletion, [dis-] appearance of nodes). These temporal anchor points are—as discussed in Fig. 6–also the moments where trajectory information, including the belonging start and end-points of the movement, should be stored and also relative poses as required. Trajectory and pose information render the spatial anchor points (Step 6B) needed for execution and to obtain them one needs to extract some information directly from the visual scene (e.g., by means of trajectory and pose estimation algorithms, not discussed in this paper).

Execution—for example with a robot—proceeds along the lower part of the diagram and extracts objects which could potentially be used for a given manipulation. This requires object recognition and matching to the stored object list from step 5A, which is another difficult problem and will not be further discussed here. Action sequencing arises then from the N-T or T-N transitions in the model-SEC and spatial information is extracted from the spatial anchor points defined above. This information must then be submitted to the appropriate low-level robot execution protocols (controllers, etc.) by which the actual execution will be driven.

This basic algorithmic procedure described here is quite simple, complexity arises from some of the required submodules, for example, pose estimation, trajectory learning, object recognition, and robot control. This is to some degree good news as it allows improving submodules without affecting the basic algorithmic procedure. We have recently suggested data structures for storing object, pose, and trajectory information in

[55] and defined SEC-based robot control procedures in [56]. Some results, where simple submodules have been used for trajectory and object handling, are presented in Figs. 10 and 11, next, in order to raise confidence in the proposed algorithmic framework.

VII. Experiments

A. Analyzing Basic Manipulations

We have analyzed the 26 actions listed in Table II first providing the graph sequences and then creating a confusion matrix (Fig. 9). With a few exceptions, it can be seen that within each of the six manipulation goal categories, actions have high similarity with each other but are rather dissimilar from actions of different goal categories.[4] Note, in all real experiments [2], [3] we are using a similarity threshold of 65% and consider actions to be the same only if this threshold is passed. Exceptions are observed mainly for the category "Rearrange," where actions Stir, Knead, and Lever have less similarity as compared to other actions in the same category.

Another interesting result seen in this confusion matrix is that "Take Down" and "Construct" actions have high similarity with each other (see the pink frames). This is because both action groups include two half actions: first "Pick up" and then "Put down", which can be temporally inverted to obtain the one or the other action. This has already been discussed in the last paragraph of Section II-A. The higher scores all across the lower right part of the matrix point to such temporal inversions.

"Stir", is also to some degree (but subthreshold) similar to "TakeDown/Push/Down" as well as "PushOnTop/PutOnTop," where the same temporal inversion explains the dual-similarity. The fact that "Stir" looks a bit like "TakeDown/Push/Down" is because both actions start with a "stack" of objects (background + cup + liquid or background + stacked object1 + stacked object 2) where the hand touches the stack (finger in liquid versus hand grasps stacked object 2) and where the action end with its release. This leads to 50% similarity of the event chains. Another exception is the scooping action, which has low similarities within its own category… This is also expected as scooping is another action performed on liquid substances and requires a container for holding it (an additional node). We also find that the two "Uncover" actions are also related to the "Destroy" category. Here we realize that uncovering uncovers a so-far unseen object. This is similar to the destroying action by which also new objects (rather object parts, hence new nodes) are being created.

This confusion matrix, thus, confirms that the theoretically discussed differences between actions can actually be measured in a generic way. Interestingly, this analysis has also revealed several unexpected additional cognitive cross-links which are not immediately obvious. Note, in order to hook up to the theoretical considerations from above the complete analysis took place using unlabeled graphs. For all practical purposes it will, however, certainly make sense to label the "hand" and the "background" and calculate similarities on such labeled graphs. This way several false positive will be eliminated (for example the Stir-TakeDown/Push/Down similarity will vanish).

B. Human Manipulation Recognition

The following results are recompiled from our older publications and kept short. The presented experiments are to illustrate the use of semantic event chains but do not use the ontology from above directly.

Fig. 10 shows one example each from the following three manipulation actions: 1) pick and place; 2) pour liquid; and 3) make sandwich. The first two have been discussed above (see Fig. 5); the third one is a sequence with two hands, showing that the framework can also capture bimanual manipulations. For these different examples four instantiations each have been recorded (12 movies in total). Movies contained several 100 frames; some are shown together with their image segmentation (see [57]) and the corresponding graphs in Fig. 10(a)–10(c). From the main graphs, SECs have been derived [see example in Fig. 10(d)–10(f)] and their metric similarity has been measured (see [3] for algorithmic details). The resulting confusion matrix is shown in panel G and gives percent similarities coded in color. This demonstrates that the different manipulations are recognized with quite some reliability when using SECs. The bottom part of the figure (panel H) shows the results obtained from object classification, where the unchanging role of a certain subgraph in a given manipulation is indicative of an object (in its action context). If desired one can, thus, obtain a list of eligible objects from many observations of the same manipulation.

As explained above, for executing a manipulation the model SEC can be used, too, as it provides in the first place the temporal anchor points (see Fig. 8, Step 6A), which define when an N-T or T-N transition should happen. At these time moments, descriptors for pose and trajectory are attached (spatial anchors) as suggested by the time bars in Fig. 7 (see Fig. 8, Step 6B). Thus, the model-SEC, allows restricts manipulation definition to just a few anchor points in time and space.

C. Manipulation Execution With a Robot

For demonstrating that the same framework (SECs) can also be employed for execution we have used a simulation setup (WebBots with the Katana Robot Arm). A detailed description of the complete execution example is given in [58], here we only provide a summary.

In the first stage we have manually programmed the robot arm to perform 10 different pushing actions (demonstration phase for learning), where the arm pushes a red box to a green box. Using the robot already at this stage was for practical purposes only and demonstration could have also be performed by a human as the recognition and model-SEC learning steps (top parts of Fig. 8) do not differ. The blue area in Fig. 11 (top left) shows some frames from one example from such a demonstration together with image segmentation and graphs. A magnified view is given at the bottom. After 10 repetitions we received the model-SEC shown in the yellow area (top right) together with the confidence values ω of rows and columns normalized to one. Entries "A" in the SEC mean that this node is absent during that particular temporal interval. We note that

[4] Here also some of the stranger conjectures find an explanation. For example, we put "Draw" into the "Destroy" category. This is intriguing but can be easily explained by realizing that "Draw" generates a new object (a line, by a creative process) and this way it becomes similar to the results of destroying, where also new objects are created albeit here by destructive processes.

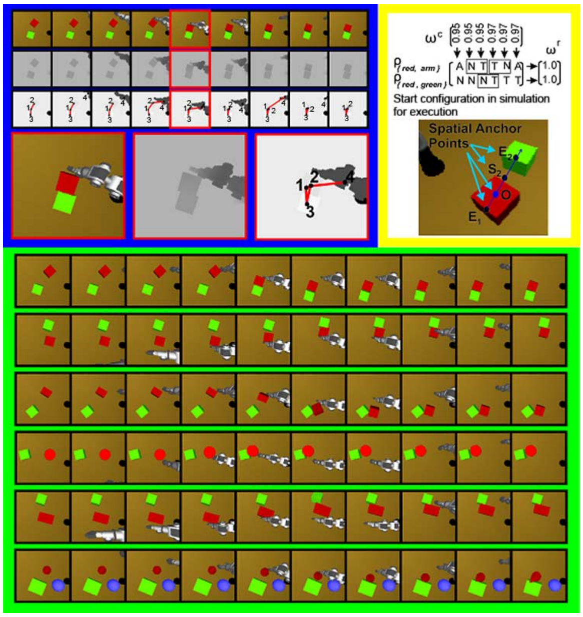


Fig. 11. Execution example. (Blue) Example of a demonstrated pushing action, (Yellow) Learned model-SEC and definition of motion parameters, (Green) ten successful executions of pushing with different initial configurations.

"pushing two objects together" corresponds to entry 24 in Table II. When comparing the SEC for "constructing" in Fig. 7 with the one shown here one can see that they are essentially identical, where here we have left out the meaningless "always-N" row which would describe the relation between robot arm and green object (they never touch). Robot arm and boxes are the objects that "play a certain role" in the manipulation as defined by the model-SEC. They are recognized by their color.

The N-T or T-N transitions (nontouching to touching and vice versa) provide the temporal anchor points (Fig. 8) on which the trajectory information, required for pushing, is being attached. Thus, for this case we did indeed perform the required trajectory analysis procedures at the temporal anchor points. Note, detailed pose information is not needed as the push-together action was performed in a simple, pose-independent way. Please see the APPENDIX, which provides the complete data structure which captures the SEC and includes the additionally required pieces of information (e.g., objects, poses and trajectories).

Some spatial anchor points are shown in the Start Configuration Diagram (yellow area) marked O, S, and E. It is important to note that this process of extracting "execution commands and parameters for pushing" (Steps 9 and 10 in Fig. 8) is not proprietary; instead it relies on general properties of all building manipulation actions. By this we can generically define a coordinate origin, which is always the center of the first touched object (blue dot O in "start configuration" yellow area, bottom) and the main direction of motion (blue arrow), which is always from origin to center of the second touched object, as well as start and endpoints for the different motion segments (S, E, not all relevant spatial anchor points are shown in this diagram).

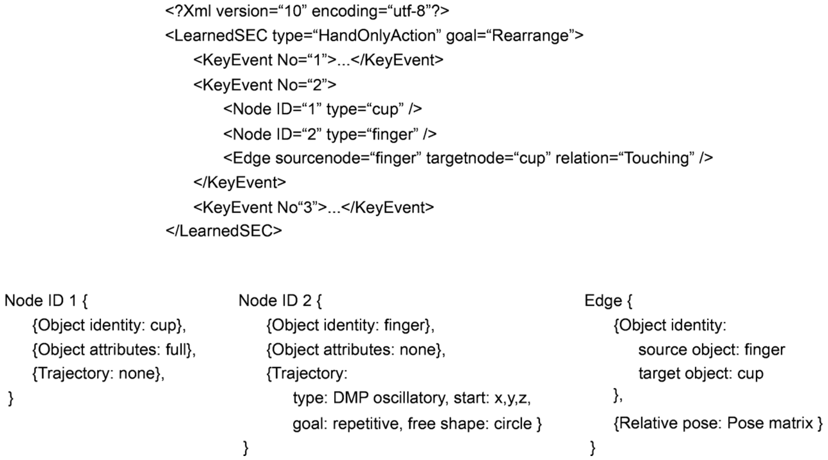
Applying these definitions in any new scene with two objects and a robot arm allows us now to perform the desired push-together action. Six examples with quite different initial configurations are shown in the green area (bottom of Fig. 11).

VIII. Conclusion

Structuring the world for a cognitive agent has proven to be difficult, especially when trying to do this from the perspective of objects. This is due to the fact that the same physical thing can take different roles depending on the actions performed with it. Hence different "objects" emerge from the same thing [6], [18], [19]. It seems, thus, more promising trying to provide structure to an agent by considering actions (here manipulations) instead of things and objects. As shown in this study (by ways of 447 small, colored disks), the manipulation space is much more restricted and a rather simple tree-like structure emerges for most if not all possible manipulations. Thus, a somehow remarkable fact of the introduced ontology is its "simplicity." Less than 30 basic actions with three associated manipulation types and six goal categories have been found. This fact gives quite some hope that the actual problem of establishing systems realizing tool-use behaviors on robots can be based on a rather constrained number of basic skills, where complexity arises only at the control-level. Our claim of simplicity also gets support from at least three additional sources.

- When we compare the dimensions of the space of perceptive information with the dimensions of the actuator space we realize that the action space is of much lower dimension (26 DoF for one hand and 7 DoF for one human arm, i.e., 66 DoF for both hands and arms) versus a dimension of 10^6 to 10^7 for sensorial information (assuming a 1000×1000 pixel image stream a space of dimension 10^6 is spanned not counting audio, tactile or proprioceptive information).
- Looking at a quantification of verbs and nouns it becomes obvious that there are much fewer verbs (in particular tool-use related verbs) than nouns. The group of Michael Beetz has performed an interesting analysis for the problem of cooking and they find that only a very limited number of relevant action (relevant verbs) exist for a given scenario (they report about 15 verbs, Nyga and Beetz, personal communication). Note also that Biederman [59] suggested that every person is familiar with around 30 000 objects compared with a rather restricted set of basic skills.
- Looking at the development of human tool use abilities [1] there is evidence that—starting with a small set of around five tool-use related repetitive action patterns, which are probably to a large degree innately coded, tool use competences maturate in the developmental process by interacting with objects in the world. This developmental process is guided by a number of basic mechanisms such as *repetition*, *variation*, and *selection* and in our context most importantly *composition*, i.e., the combination of lower level skills to higher level skills.

Furthermore, as discussed above, the ontology introduced in this paper represents an abstraction of actions and the establishment of these actions in real systems requires to add more information such as trajectory, pose, etc. but also—and this has not yet been mentioned—ultimately one needs to include other perceptive information (e.g., haptic or proprioceptive signals) to find the optimal trajectory as well as the best contact with the object.



```
<?Xml version=“10” encoding=“utf-8”?>
<LearnedSEC type=“HandOnlyAction” goal=“Rearrange”>
    <KeyEvent No=“1”>...</KeyEvent>
    <KeyEvent No=“2”>
        <Node ID=“1” type=“cup” />
        <Node ID=“2” type=“finger” />
        <Edge sourcenode=“finger” targetnode=“cup” relation=“Touching” />
    </KeyEvent>
    <KeyEvent No“3”>...</KeyEvent>
</LearnedSEC>
```

```
Node ID 1 {                      Node ID 2 {                                 Edge {
    {Object identity: cup},          {Object identity: finger},                  {Object identity:
    {Object attributes: full},       {Object attributes: none},                      source object: finger
    {Trajectory: none},              {Trajectory:                                    target object: cup
}                                        type: DMP oscillatory, start: x,y,z,    },
                                         goal: repetitive, free shape: circle }  {Relative pose: Pose matrix }
                                 }                                           }
```

Fig. 12. GraphML code for Key Event 2 in the “rearrange” Manipulation (first one in Fig. 1, 6). The GraphML code contains node and edge descriptors which define the required poses and trajectories for the different objects. Note, we use a human readable form here for encoding subentities. For example the cup-description is called “cup”. This is not necessarily the case in an actual implementation. Some details, like the actual start positions, x,y,z and the pose-matrix are also left unspecified.

Finally, we have shown that this manipulation ontology finds its use for robotics by ways of the semantic event chains which provide temporal anchor points also for the attachment of pose, trajectory and object information. This allows—at least in principle—now a fairly complete encoding of a manipulation. Many steps will have to be performed to make this framework rock-solid and robust against noise (in the sensor as well as motor domain) and other contingencies. This none withstanding, we think the manipulation ontology presented here together with some useful data structures may indeed help to better understand and execute manipulations in the future.

Appendix
GraphML Code

SECs are human interpretable but not directly machine readable and they do not encode object, trajectory and pose information. To address these aspects, we choose a different, second form of representation based on GraphML code [60]. The basis of this is the SEC, where *every* key event is encoded in *one* GraphML data sheet. This representation is, thus, quite detailed and difficult to read for a human. On the other hand, the GraphML code is machine compatible and can be used for automatic, computer-based manipulation recognition as well as execution. Essentially it is identical to a SEC, just using a different type of encoding, but now it is possible to supplement this GraphML code with object, pose and trajectory information. Fig. 12 shows one example of the GraphML code for the second Key Event (the event marked with “T,” when both touch) of the “Rearrange”-Graph in Fig. 7, hence that Key Event where the two nodes touch (spoon enters cup and begins stirring). A similar GraphML entry must be created for every Key Event in a Graph Sequence (hence in a SEC). As the figure demonstrates, nodes and edges are being supplemented with trajectory and pose information.

Acknowledgment

The authors would like to thank T. Asfour for valuable discussions and very helpful criticism along the way.

References

[1] F. Guerin, N. Krüger, and D. Kraft, “A survey of the ontogeny of tool use: from sensorimotor experience to planning,” *IEEE Trans. Autonom. Mental Develop.*, vol. 5, no. 1, pp. 18–45, Mar. 2013.

[2] E. E. Aksoy, A. Abramov, F. Wörgötter, and B. Dellen, “Categorizing object-action relations from semantic scene graphs,” presented at the IEEE Int. Conf. Robot. Autom. (ICRA), 2010.

[3] E. E. Aksoy, A. Abramov, J. Dörr, N. Kejun, B. Dellen, and W. Florentin, “Learning the semantics of object-action relations by observation,” *Int. J. Robot. Res. (IJRR)*, vol. 30, no. 10, pp. 1229–1249, 2011.

[4] B. Aune, *Reason and Action*. Boston, MA, USA: D. Reidel Publishing Comp, 1978.

[5] B. Aune, “Action and Ontology,” *Philosoph. Studies*, vol. 54, pp. 195–213, 1988.

[6] V. F. Turchin, “The cybernetic ontology of action,” *Kybernetes*, vol. 22, pp. 10–30, 1993.

[7] E. Pacherie, “The phenomenology of action: A conceptual framework,” *Cognition*, vol. 107, pp. 179–217, 2008.

[8] T. Schack and H. Ritter, “The cognitive nature of action—Functional links between cognitive psychology, movement science, and robotics,” *Progress. Brain Res.*, vol. 174, pp. 231–251, 2009.

[9] A. Sloman, Progress Report on the Cognition and Affect Project: Architectures, Architecture-Schemas, and the New Science of Mind 2008.

[10] D. Vernon, G. Metta, and G. Sandini, “The iCub cognitive architecture: interactive development in a humanoid robot,” in *IEEE Int. Conf. Develop. Learn. Imperial College*, London, U.K., 2007.

[11] D. Vernon, C. Hofsten, and L. Fadiga, “A roadmap for cognitive development in humanoid robots,” in *Cognitive Syst. Monographs*. Berlin, Germany: Springer-Verlag, 2010, vol. 11, pp. 121–153.

[12] D. Kraft, E. Baseski, M. Popovic, A. M. Batog, A. Kjær-Nielsen, N. Krüger, R. Petrick, C. Geib, N. Pugeault, M. Steedman, T. Asfour, R. Dillmann, S. Kalkan, F. Wörgötter, B. Hommel, R. Detry, and J. Piater, “Exploration and planning in a three-level cognitive architecture,” presented at the Int. Conf. Cognitive Syst. (CogSys), 2008.

[13] T. Metzinger and V. Gallese, “The emergence of a shared action ontology: Building blocks for a theory,” *Consciousness Cogn.*, vol. 12, pp. 549–571, 2003.

[14] J. J. Gibson, *The Theory of Affordances. In Perceiving, Acting, and Knowing*, R. Shaw and J. Bransford, Eds. Hillsdale, NJ, USA: Lawrence Erlbaum Associates, 1977.

[15] J. J. Gibson, *The Ecological Approach to Visual Perception*. Boston, MA, USA: Houghton Mifflin, 1979.

[16] B. Hommel, J. Müsseler, G. Aschersleben, and W. Prinz, “The theory of event coding (TEC): A framework for perception and action planning,” *Behav. Brain Sci.*, vol. 24, pp. 849–87, 2001.

[17] J. Piaget, *The Child's Construction of Reality*. London, U.K.: Routledge & Kegan Paul, 1955.

[18] F. Wörgötter, A. Agostini, N. Krüger, N. Shylo, and B. Porr, "Cognitive agents — a procedural perspective relying on predictability of object-action complexes (OACs)," *Robot. Auton. Syst.*, vol. 57, pp. 420–432, 2009.

[19] N. Krüger, J. Piater, C. Geib, R. Petrick, M. Steedman, F. Wörgötter, A. Ude, T. Asfour, D. Kraft, D. Omrcen, A. Agostini, and R. Dillmann, "Object-action complexes: Grounded abstractions of sensorimotor processes," *Robot. Autonom. Syst. (RAS)*, vol. 59, pp. 740–757, 2011.

[20] H. Kjellström, J. Romero, and D. Kragic, "Visual object-action recognition: Inferring object affordances from human demonstration," *Comput. Vis. Image Understand.*, vol. 115, pp. 81–90, 2011.

[21] H. Kjellström, J. Romero, D. Martínez, and D. Kragic, "Simultaneous visual recognition of manipulation actions and manipulated objects," in *Eur. Conf. Comput. Vision*, 2008.

[22] S. Griffith, V. Sukhoy, and A. Stoytchev, "Using sequences of movement dependency graphs to form object categories," presented at the IEEE Conf. Humanoids, Bled, Slovenia, 2011.

[23] M. Sridhar, A. Cohn, and D. Hogg, "Learning functional object-categories from a relational spatio-temporal representation," presented at the 18th Eur. Conf. Artif. Intell., 2008.

[24] M. Sridhar, A. Cohn, and D. Hogg, "Unsupervised learning of event classes from video," presented at the 24th AAAI Conf. Artif. Intell., 2010.

[25] A. Gangemi, R. Navigli, and P. Velardi, "The ontowordnet project: Extension and axiomatization of conceptual relations in wordnet," *Proc. CoopIS/DOA/ODBASE'03*, 2003.

[26] H. Liu and P. Singh, "Conceptnet: A practical commonsense reasoning toolkit," *BT Technol. J.*, vol. 22, pp. 221–226, 2004.

[27] C. Kemke, "An action ontology framework for natural language interfaces to agent systems," in *Artif. Intell. Rev.*   : Springer, 2007.

[28] R. Trypuz, "Formal Ontology of Action: A Unifying Approach," Ph.D. dissertation, Univ. Trento, Trento, Italy, 2007.

[29] M. Gutierrez, F. Vexo, and D. Thalman, "Semantic-based representation of virtual environments," *Int. J. Comput. Appl. Technol.*, pp. 229–238, 2005.

[30] S. Irawati, D. Calderón, and H. Ko, "Spatial ontology for semantic integration in 3D multimodal interaction framework," presented at the ACM Int. Conf. Virtual Reality Contin. Appl., Hongkong, 2006.

[31] J. Lugrin and M. Cavazza, "Making sense of virtual environments: Action representation, grounding and common sense," presented at the 2007 Int. Conf. Intell. User Interfaces, IUI, Honolulu, Hawaii, 2007.

[32] M. Tenorth, D. Nyga, and M. Beetz, "Understanding and executing instructions for everyday manipulation tasks from the world wide web," presented at the IEEE Int. Conf. Robot. Autom., 2010.

[33] K. Kunze, M. E. Dolha, E. Guzman, and M. Beetz, "Simulation-based temporal projection of everyday robot object manipulation," presented at the 10th Int. Conf. on Auton. Agents Multiagent Syst., 2011.

[34] M. Soury, P. Hède, P. Morignot, C. Leroux, J. Savary, and J. Canou, "Cognitive and physical stimulation services with a robotic agent in a domestic setting," presented at the 11th Conf. Eur. Assoc. Adv. Assistive Technol. Eur., 2011.

[35] H. S. Chung, J. M. Kim, Y. C. Byun, and S. Y. Byun, "Retrieving and exploring ontology-based human motion sequences," presented at the Comput. Sci. Appl. — ICCSA, .

[36] S. Kobayashi, S. Tamagawa, T. Morita, and T. Yamaguchi, "Intelligent humanoid robot with japanese Wikipedia ontology and robot action ontology," presented at the HRI'2011, .

[37] D. Kulic, W. Takano, and Y. Nakamura, "On-line segmentation and clustering from continuous observation of whole body motions," *IEEE Trans. Robot.*, vol. 25, pp. 1158–1166, 2009.

[38] M. Yamamoto, H. Mitomi, F. Fujiwara, and T. Sato, "Bayesian classification of task-oriented actions based on stochastic context-free grammar," presented at the Int. Conf. Automat. Face Gesture Recognit., Southampton, 2006.

[39] V. Krüger, D. Herzog, S. Baby, A. Ude, and D. Kragic, "Learning actions from observations: Primitive-based modeling and grammar," *IEEE Robot. Autom. Mag.*, vol. 17, no. 2, pp. 30–43, Apr. 2010.

[40] N. Dantam and M. Stilman, "The motion grammar: Linguistic perception, planning, and control," *Robotics: Sci. and Syst., RSS*, 2011.

[41] M. R. Cutkosky and R. D. Howe, "Dextrous robot hands," in *Human Grasp Choice and Robotic Grasp Anal.*   Berlin, Germany: Springer-Verlag, 1990, pp. 5–31.

[42] S. Kang and K. Ikeuchi, "A grasp abstraction hierarchy for recognition of grasping tasks from observation," in *Proc. IEEE/RSJ Int'l Conf. Intell. Robots Syst.*, Yokohama, Japan, 1993.

[43] T. Feix, R. Pawlik, H. Schmiedmayer, J. Romero, and D. Kragic, "A comprehensive grasp taxonomy," presented at the Robotics, Sci. Syst.: Workshop Understanding the Human Hand for Adv. Robotic Manipulation, 2009.

[44] M. Do, J. Romero, H. Kjellstrom, P. Azad, T. Asfour, D. Kragic, and R. Dillmann, "Grasp recognition and mapping on humanoid robots," presented at the IEEE-RAS Int. Conf. Humanoid Robots, 2009.

[45] K. Pastra and Y. Aloimonos, "The minimalist grammar of action," *Philosophical Trans. Roy. Soc. B*, vol. 367, pp. 103–117, 2012.

[46] M. Beetz, D. Jain, L. Mösenlechner, and M. Tenorth, "Towards performing everyday manipulation activities," *Robot. Autonom. Syst.*, vol. 58, no. 9, pp. 1085–1095, 2010.

[47] M. Beetz, M. Tenorth, D. Jain, and J. Bandouch, "Towards automated models of activities of daily life," *Technol. Disability*, vol. 22, pp. 27–40, 2010.

[48] A. Revonsuo, "Binding and the phenomenal unity of consciousness," *Conscious Cogn.*, vol. 8, pp. 173–85, 1999.

[49] H. Ritter, R. Haschke, and J. J. Steil, "A dual interaction perspective for robot cognition: Grasping as a "Rosetta stone"," *Perspectives of Neural-Symbolic Integr.*, pp. 159–178, 2007.

[50] C. Breazeal and B. Scassellati, "Robots that imitate humans," *Trends Cogn. Sci.*, vol. 6, pp. 481–487, 2002.

[51] R. Dillmann, "Robot learning from demonstration," *Robot. Autonom. Syst. (RAS)*, vol. 47, pp. 109–116, 2004.

[52] R. Zöllner, T. Asfour, and R. Dillmann, "Programming by demonstration: Dual-arm manipulation tasks for humanoid," presented at the IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Sendai, Japan, 2004.

[53] E. Billing and T. Hellström, "A formalism for learning from demonstration," *Paladyn J. Behav. Robot.*, vol. 1, pp. 1–13, 2010.

[54] A. Billard and D. Grollman, Robot Learning by Demonstration (Under Review).   Scholarpedia, 2011 [Online]. Available: http://www.scholarpedia.org/article/Robot_learning_by_demonstration

[55] E. E. Aksoy, M. Tamosiunaite, R. Vuga, A. Ude, C. Geib, M. Steedman, and F. Wörgötter, *Structural bootstrapping at the sensorimotor level for the fast acquisition of action knowledge for cognitive robots*, 2013, submitted for publication.

[56] M. J. Aein, E. E. Aksoy, M. Tamosiunaite, J. Papon, A. Ude, and F. Wörgötter, *Toward a library of actions based on semantic object-action relations*, 2013, submitted for publication.

[57] A. Abramov, E. E. Aksoy, J. Dörr, F. Wörgötter, and B. Dellen, "3D semantic representation of actions from efficient stereo-image-sequence segmentation on GPUs," presented at the 5th Int. Symp. 3D Data Process., Visualiz., Trans., 2010.

[58] E. E. Aksoy, B. Dellen, M. Tamosiunaite, and F. Wörgötter, "Execution of a dual-object (pushing) action with semantic event chains," presented at the IEEE-RAS Int. Conf. Humanoid Robots, 2011.

[59] I. Biederman, "Recognition-by-components: A theory of human image understanding," *Psychol. Rev.*, pp. 115–47, 1987.

[60] U. Brandes, M. Eiglsperger, J. Lerner, and C. Pich, "Graph markup language (GraphML)," *Handbook of Graph Drawing Visualiz.*, 2010.

Florentin Wörgötter has studied biology and mathematics at the University of Düsseldorf, Germany. He received the Ph.D. degree for work on the visual cortex from the University of Essen, Essen, Germany, in 1988.

From 1988 to 1990, he was engaged in computational studies with the California Institute of Technology, Pasadena, CA, USA. Between 1990 and 2000, he was a Researcher at the University of Bochum, Bochum, Germany, where he was investigating the experimental and computational neuroscience of the visual system. From 2000 to 2005, he was a Professor of computational neuroscience with the Psychology Department, University of Stirling, U.K., where his interests strongly turned towards "Learning in Neurons". Since July 2005, he has been the Head of the Computational Neuroscience Department at the Bernstein Center for Computational Neuroscience, Inst. Physics 3, University of Göttingen, Germany. His current research interests include information processing in closed-loop perception-action systems, sensory processing, motor control, and learning/plasticity, which are tested in different robotic implementations. His group has also developed the RunBot, which is a fast and adaptive biped-walking robot.

Eren Erdal Aksoy received the M.Sc. degree in mechatronics from the University of Siegen, Siegen, Germany, in 2008, and the Ph.D. degree in computer science from the University of Göttingen, Göttingen, Germany, in 2012.

He is currently employed as a research assistant at the Bernstein Center at the University of Göttingen. His research interests include semantic analysis of manipulation actions, computer vision, and imitation learning.

Norbert Krüger is a Professor at the Mærsk McKinney Møller Institute, University of Southern Denmark. He holds a M.Sc. degree from the Ruhr-Universitat Bochum, Germany and his Ph.D. degree from the University of Bielefeld. He leads the Cognitive Vision Lab which focuses on computer vision and cognitive systems, in particular the learning of object representations in the context of grasping.

Justus Piater received the Ph.D. degree from the University of Massachusetts, Amherst, MA, USA, where he held a Fulbright Graduate Student Fellowship.

After a European Marie-Curie Individual Fellowship at INRIA Rhone-Alpes, France, he was a professor at the University of Liege, Belgium, and a Visiting Research Scientist at the Max Planck Institute for Biological Cybernetics in Tübingen, Germany. He is a Professor of Computer Science at the University of Innsbruck, Austria. His research in computer vision and machine learning is motivated by intelligent and interactive systems, where he focuses on visual learning, closed-loop interaction of sensorimotor systems, and video analysis.

Aleš Ude studied applied mathematics at the University of Ljubljana, Ljubljana, Slovenia, and received the doctoral degree from the Faculty of Informatics, University of Karlsruhe, Karlsruhe, Germany.

He has been a Visiting Researcher at ATR Computational Neuroscience Laboratories, Kyoto, Japan, for a number of years and is still associated with this group. Currently, he is the Head of the Humanoid and Cognitive Robotics Lab at Jozef Stefan Institute, Ljubljana, Slovenia. His research focuses on imitation and action learning, perception of human activity, humanoid robot vision, and humanoid cognition.

Dr. Ude was awarded the STA fellowship for postdoctoral studies in ERATO Kawato Dynamic Brain Project.

Minija Tamosiunaite received the Ph.D. in informatics from Vytautas Magnus University, Kaunas, Lithuania, in 1997.

Currently, she works as a Senior Researcher at the Bernstein Center for Computational Neuroscience, University of Göttingen, Göttingen, Germany. Her research interests include machine learning, biological signal analysis, and application of learning methods in robotics.