Advances in Cognitive Systems 3 (2014) 67–86 Submitted 9/2013; published 8/2014

A Cognitive System for Understanding Human Manipulation Actions

Yezhou Yang YZYANG@CS.UMD.EDU
Anupam Guha AGUHA@CS.UMD.EDU
Cornelia Fermüller FER@UMIACS.UMD.EDU
Yiannis Aloimonos YIANNIS@CS.UMD.EDU
4470 A. V. Williams Building, University of Maryland, College Park, MD 20742 USA

Abstract

This paper describes the architecture of a cognitive system that interprets human manipulation actions from perceptual information (image and depth data) and that includes interacting modules for perception and reasoning. Our work contributes to two core problems at the heart of action understanding: (a) the grounding of relevant information about actions in perception (the perception-action integration problem), and (b) the organization of perceptual and high-level symbolic information for interpreting the actions (the sequencing problem). At the high level, actions are represented with the Manipulation Action Grammar, a context-free grammar that organizes actions as a sequence of sub events. Each sub event is described by the hand, movements, objects and tools involved, and the relevant information about these factors is obtained from biologically-inspired perception modules. These modules track the hands and objects, and they recognize the hand grasp, objects and actions using attention, segmentation, and feature description. Experiments on a new data set of manipulation actions show that our system extracts the relevant visual information and semantic representation. This representation could further be used by the cognitive agent for reasoning, prediction, and planning.

1. Introduction

Cognitive systems that interact with humans must be able to interpret actions. Here we are concerned with manipulation actions. These are actions performed by agents (humans or robots) on objects, which result in some physical change of the object. There has been much work recently on action recognition, with most studies considering short lived actions, where the beginning and end of the sequence is defined. Most efforts have focused on two problems of great interest to the study of perception: the recognition of movements and the recognition of associated objects. However, the more complex an action, the less reliable individual perceptual events are for the characterization of actions. Thus, the problem of interpreting manipulation actions involves many more challenges than simply recognizing movements and objects, due to the many ways that humans can perform them.

Since perceptual events do not suffice, how do we determine the beginning and end of action segments, and how do we combine the individual segments into longer ones corresponding to a manipulation action? An essential component in the description of manipulations is the underlying

goal. The goal of a manipulation action is the physical change induced on the object. To accomplish it, the hands must perform a sequence of sub actions on the object, such as when the hand grasps or releases the object, or when the hand changes the grasp type during a movement. Centered around this idea, we develop a grammatical formalism for parsing and interpreting action sequences, and we also develop vision modules to obtain from dynamic imagery the symbolic information used in the grammatical structure.

Our formalism for describing manipulation actions uses a structure similar to natural language. What do we gain from this formal description of action? This is equal to asking what one gains from a formal description of language. Chomsky's contribution to language was the formal description of language through his generative and transformational grammar (1957). This revolutionized language research, opened up new roads for its computational analysis and provided researchers with common, generative language structures and syntactic operations on which language analysis tools were built. Similarly, a grammar for action provides a common framework of the syntax and semantics of action, so that basic tools for action understanding can be built. Researchers can then use these tools when developing action interpretation systems, without having to start from scratch.

The input into our system for interpreting manipulation actions is perceptual data, specifically sequences of images and depth maps. Therefore, a crucial part of our system is the vision process, which obtains atomic symbols from perceptual data. In Section 3, we introduce an integrated vision system with attention, segmentation, hand tracking, grasp classification, and action recognition. The vision processes produce a set of symbols: the "Subject", "Action", and "Object" triplets, which serve as input to the reasoning module. At the core of our reasoning module is the *manipulation action context-free grammar* (MACFG). This grammar comes with a set of generative rules and a set of parsing algorithms. The parsing algorithms use two main operations – "construction" and "destruction" – to dynamically parse a sequence of tree (or forest) structures made up from the symbols provided by the vision module. The sequence of semantic tree structures could then be used by the cognitive system to perform reasoning and prediction. Figure 1 shows the flow chart of our cognitive system.

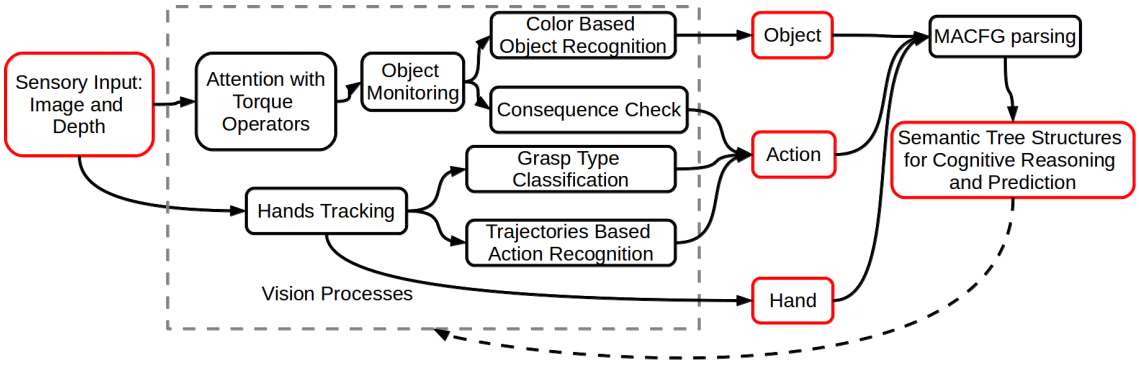


Figure 1. Overview of the manipulation action understanding system, including feedback loops within and between some of the modules. The feedback is denoted by the dotted arrow.

2. Related Work

The problem of human activity recognition and understanding has attracted considerable interest in computer vision in recent years. Both visual recognition methods and nonvisual methods using motion capture systems (Guerra-Filho, Fermüller, & Aloimonos, 2005; Li et al., 2010) have been used. Moeslund et al. (2006), Turaga et al. (2008), and Gavrila (1999) provide surveys of the former. There are many applications for this work in areas such as human computer interaction, biometrics, and video surveillance. Most visual recognition methods learn the visual signature of each action from many spatio-temporal feature points (e.g, Dollár et al., 2005; Laptev, 2005; Wang & Suter, 2007; Willems, Tuytelaars, & Van Gool, 2008). Work has focused on recognizing single human actions like walking, jumping, or running (Ben-Arie et al., 2002; Yilmaz & Shah, 2005). Approaches to more complex actions have employed parametric models such as hidden Markov models (Kale et al., 2004) to learn the transitions between image frames (e.g, Aksoy et al., 2011; Chaudhry et al., 2009; Hu et al., 2000; Saisan et al., 2001).

The problem of understanding manipulation actions is also of great interest in robotics, which focuses on execution. Much work has been devoted to learning from demonstration (Argall et al., 2009), such as the problem of a robot with hands learning to manipulate objects by mapping the trajectory observed for people performing the action to the robot body. These approaches have emphasized signal to signal mapping and lack the ability to generalize. More recently, within the domain of robot control research, Fainekos et al. (2005) have used temporal logic for hybrid controller design, and later Dantam and Stilman (2013) suggested a grammatical formal system to represent and verify robot control policies. Wörgötter et al. (2012) and Aein et al. (2013) created a library of manipulation actions through semantic object-action relations obtained from visual observation.

There have also been many syntactic approaches to human activity recognition that use the concept of context-free grammars, because they provide a sound theoretical basis for modeling structured processes. Brand (1996) used a grammar to recognize disassembly tasks that contain hand manipulations. Ryoo and Aggarwal (2006) used the context-free grammar formalism to recognize composite human activities and multi-person interactions. It was a two-level hierarchical approach in which the lower level was composed of hidden Markov models and Bayesian networks while the higher-level interactions were modeled by CFGs. More recent methods have used stochastic grammars to deal with errors from low-level processes such as tracking (Ivanov & Bobick, 2000; Moore & Essa, 2002). This work showed that grammar-based approaches can be practical in activity recognition systems and provided insight for understanding human manipulation actions. However, as mentioned, thinking about manipulation actions solely from the viewpoint of recognition has obvious limitations. In this work, we adopt principles from CFG-based activity recognition systems, with extensions to a minimalist grammar that accommodates not only the hierarchical structure of human activity, but also human-tool-object interactions. This approach lets the system serve as the core parsing engine for manipulation action interpretation.

Chomsky (1993) suggested that a minimalist generative structure, similar to the one in human language, also exists for action understanding. Pastra and Aloimonos (2012) introduced a minimalist grammar of action, which defines the set of terminals, features, non-terminals and production rules for the grammar in the sensorimotor domain. However, this was a purely theoretical description. The first implementation used only objects as sensory symbols (Summers-Stay et al., 2013).

Then Guha et al. (2013) proposed a minimalist set of atomic symbols to describe the movements in manipulation actions. In the field of natural language understanding, which traces back to the 1960s, Schank and Tesler proposed the Conceptual Dependency theory (1969) to represent content inferred from natural language input. In this theory, a sentence is represented as a series of diagrams representing both mental and physical actions that involve agents and objects. These actions are built from a set of primitive acts, which include atomic symbols like GRASP and MOVE. Manikonda et al. (1999) have also discussed the relationship between languages and motion control.

Here we extend the minimalist action grammar of Pastra and Aloimonos (2012) to dynamically parse the observations by providing a set of operations based on a set of context-free grammar rules. Then we provide a set of biologically inspired visual processes that compute from the low-level signals the symbols used as input to the grammar in the form of (Subject, Action, Object). By integrating the perception modules with the reasoning module, we obtain a cognitive system for human manipulation action understanding.

3. A Cognitive System For Understanding Human Manipulation Actions

In this section, we first describe the Manipulation Action Context-Free Grammar and the parsing algorithms based on it. Then we discuss the vision methods: the attention mechanism, the hand tracking and action recognition, the object monitoring and recognition, and the action consequence classification.

3.1 A Context-Free Manipulation Action Grammar

Our system includes vision modules that generate a sequence of "Subject" "Action" "Patient" triplets from the visual data, a reasoning module that takes in the sequence of triplets and builds them into semantic tree structures. The binary tree structure represents the parsing trees, in which leaf nodes are observations from the vision modules and the non-leaf nodes are non-terminal symbols. At any given stage of the process, the representation may have multiple tree structures. For implementation reasons, we use a DUMMY root node to combine multiple trees. Extracting semantic trees from observing manipulation actions is the target of the cognitive system.

Before introducing the parsing algorithms, we first introduce the core of our reasoning module: the Manipulation Action Context-Free Grammar (MACFG). In formal language theory, a context-free grammar is a formal grammar in which every production rule is of the form $V \rightarrow w$, where V is a single nonterminal symbol, and w is a string of terminals and/or nonterminals (w can be empty). The basic recursive structure of natural languages, the way in which clauses nest inside other clauses, and the way in which lists of adjectives and adverbs are followed by nouns and verbs, is described exactly by a context-free grammar.

Similarly for manipulation actions, every complex activity is built of smaller blocks. Using linguistics notation, a block consists of a "Subject", "Action" and "Patient" triplet. Here a "Subject" can be either a hand or an object, and the same holds for the "Patient". Furthermore, a complex activity also has a basic recursive structure, and can be decomposed into simpler actions. For example, the typical manipulation activity "sawing a plank" is described by the top-level triplet "handsaw saw plank", and has two lower-level triplets (which come before the top-level action in time), namely

Table 1. A Manipulation Action Context-Free Grammar.

AP	$\rightarrow$	$A\ O \mid A\ HP$	(1)
HP	$\rightarrow$	$H\ AP \mid HP\ AP$	(2)
H	$\rightarrow$	h	
A	$\rightarrow$	a	
O	$\rightarrow$	o	(3)

"hand grasp saw" and "hand grasp plank". Intuitively, the process of observing and interpreting manipulation actions is syntactically quite similar to natural language understanding. Thus, the Manipulation Grammar (Table 1) is presented to parse manipulation actions.

The nonterminals H, A, and O represent the hand, the action and the object (the tools and objects under manipulation) respectively, and the terminals h, a, and o are the observations. AP stands for Action Phrase and HP for Hand Phrase. They are proposed here as XPs following the X-bar theory, which is used to construct the logical form of the semantic structure (Jackendorff, 1977).

The design of this grammar is motivated by three observations: (i) Hands are the main driving force in manipulation actions, so a specialized nonterminal symbol H is used for their representation; (ii) An Action (A) can be applied to an Object (O) directly or to a Hand Phrase (HP), which in turn contains an Object (O), as encoded in Rule (1), which builds up an Action Phrase (AP); (iii) An Action Phrase (AP) can be combined either with the Hand (H) or a Hand Phrase, as encoded in rule (2), which recursively builds up the Hand Phrase. The rules discussed in Table 1 form the syntactic rules of the grammar used in the parsing algorithms.

3.2 Cognitive MACFG Parsing Algorithms

Our aim for this project is not only to provide a grammar for representing manipulation actions, but also to develop a set of operations that can automatically parse (create or dissolve) the semantic tree structures. This is crucial for practical purposes, since parsing a manipulation action is inherently an on-line process. The observations are obtained in a temporal sequence. Thus, the parsing algorithm for the grammar should be able to dynamically update the tree structures. At any point, the current leaves of the semantic forest structures represent the actions and objects involved so far. When a new triplet of ("Subject", "Action", "Patient") arrives, the parser updates the tree using the construction or destruction routine.

Theoretically, the non-regular context-free language defined in Table 1 can be recognized by a non-deterministic pushdown automaton. However, different from language input, the perception input is naturally a temporal sequence of observations. Thus, instead of simply building a non-deterministic pushdown automaton, it requires a special set of parsing operations.

Our parsing algorithm differentiates between constructive and destructive actions. Constructive actions are the movements that start with the hand (or a tool) coming in contact with an object and usually result in a certain physical change on the object (a consequence), e.g, "Grasp", "Cut", or "Saw". Destructive actions are movements at the end of physical change inducing actions, when the hand (or tool) separates from the object; some examples are "Ungrasp" or "FinishedCut". A

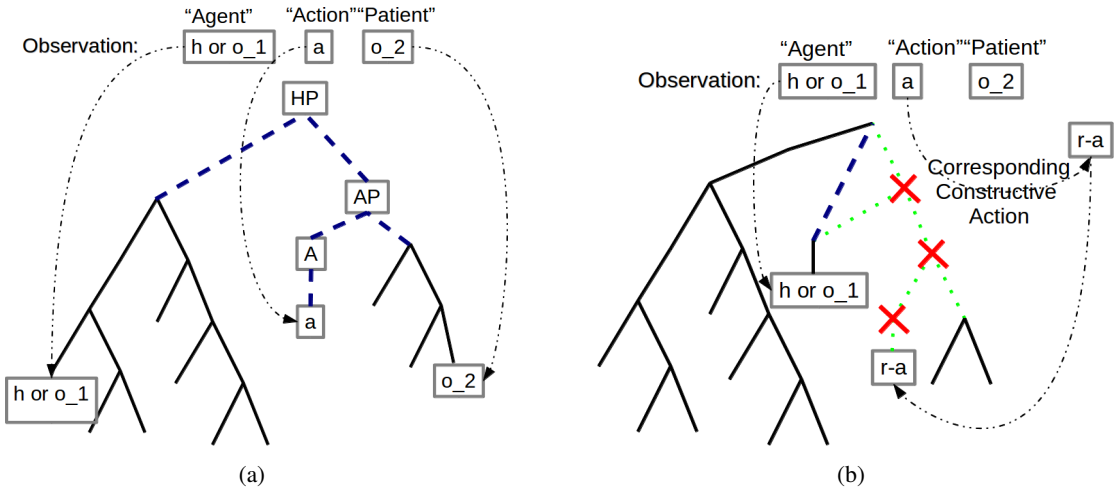


Figure 2. The (a) construction and (b) destruction operations. Fine dashed lines are newly added connections, crosses are node deletion, and fine dotted lines are connections to be deleted.

constructive action may or may not have a corresponding destructive action, but every destructive action must have a corresponding constructive action. Otherwise the parsing algorithm detects an error. In order to facilitate the action recognition, a look-up table that stores the constructive-destructive action pairs is used. This knowledge can be learned and further expanded.

The algorithm builds a tree structure T_s. This structure is updated as new observations are received. Once an observation triplet "Subject", "Action", and "Patient" arrives, the algorithm checks whether the "Action" is constructive or destructive and then follows one of two pathways. If the "Action" is constructive, a construction routine is used. Otherwise a destruction routine is used (Refer to Algorithm 1, Algorithm 2, and Algorithm 3 for details). The process continues till the last observation. Two illustrations in Figure 2 demonstrate how the construction and the destruction routines work. The parse operation amounts to a chart parser (Younger, 1967), which takes in the three nonterminals and performs bottom-up parsing following the context-free rules from Table 1.

Algorithm 1 Dynamic manipulation action tree parsing

Initialize an empty tree group (forest) T_s
while New observation (subject s, action a, patient p) **do**
  if a is a constructive action **then**
    construction(T_s, s, a, p)
  end if
  if a is a destructive action **then**
    destruction(T_s, s, a, p)
  end if
end while

Algorithm 2 construction(T_s, s, a, p)

Previous tree group (forest) T_s and new observation (subject s, action a and patient p)
if s is Hand h, and p is an object o **then**
  Find the highest subtrees T_h and T_o from T_s containing h and o. If h or o is not in the current forest, create new subtrees T_h and T_o, respectively.
  parse(T_h,a,T_o), attach it to update T_s.
end if
if s is an object o_1 and p is another object o_2 **then**
  Find the highest subtree T_o^1 and T_o^2 from T_s containing o_1 and o_2 respectively. If either o_1 or o_2 is not in the current forest, create new subtree T_o^1 or T_o^2. If both o_1 and o_2 are not in the current forest, raise an error.
  parse(T_o^1,a,T_o^2), attach it to update T_s.
end if

Figure 3 shows a typical manipulation action example. The parsing algorithm takes as input a sequence of key observations: "LeftHand Grasp Knife; RightHand Grasp Eggplant; Knife Cut Eggplant; Knife FinishedCut Eggplant; RightHand Ungrasp Eggplant; LeftHand Ungrasp Knife". Then a sequence of six tree structures is parsed up or dissolved along the time line. We provide more examples in Section 4, and a sample implementation of the parsing algorithm at `http://www.umiacs.umd.edu/~yzyang/MACFG`. For clarity, Figure 3 uses a dummy root node to create a tree structure from a forest and numbers the nonterminal nodes.

3.3 Attention Mechanism with the Torque Operator

It is essential for our cognitive system to have an effective attention mechanism, because the amount of information in real world images is vast. Visual attention, the process of driving an agent's attention to a certain area, is based on both bottom-up processes defined on low-level visual features and top-down processes influenced by the agent's previous experience and goals (Tsotsos, 1990). Recently, Nishigaki et al. (2012) have provided a vision tool, called the image torque, that captures

Algorithm 3 destruction(T_s, s, a, p)

Previous tree structures T_s and new observation (subject s, action a and patient p)
Find corresponding constructive action of a from the look-up table and denote it as a'
if There exists a lowest subtree T'_a that contains both s and a' **then**
  Remove every node on the path that starts from root of T'_a to a'.
  if T'_a has a parent node **then**
    Connect the highest subtree that contains s with T'_a's parent node.
  end if
  Leave all the remaining subtrees as individual trees.
end if
Set the rest of T_s as new T_s.

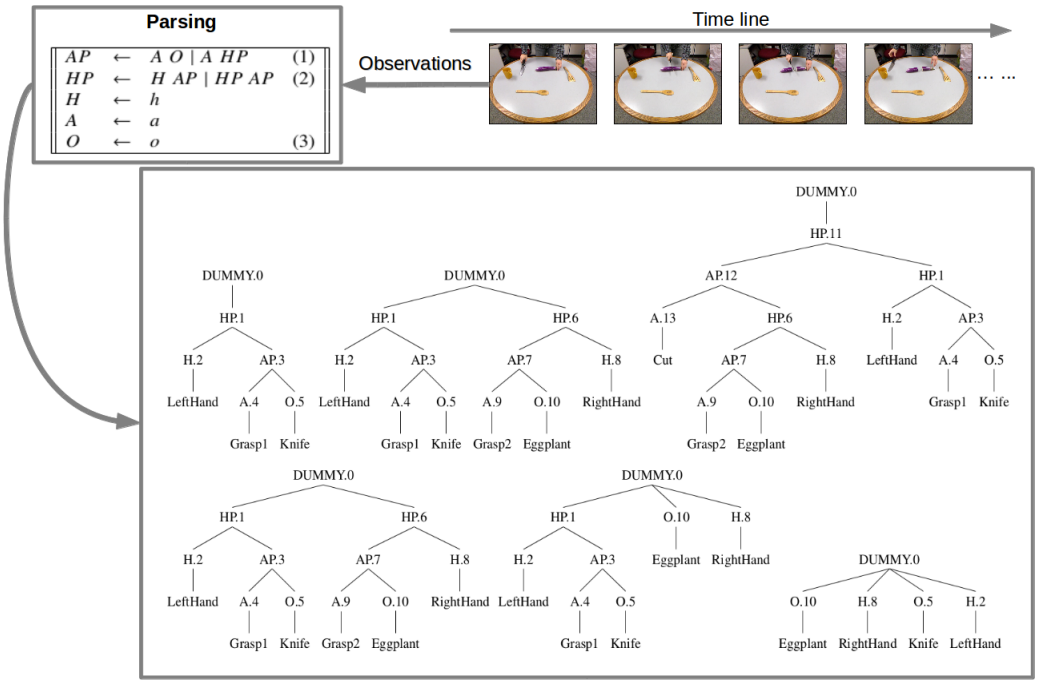


Figure 3. Here the system observes a typical manipulation action example, "Cut an eggplant", and builds a sequence of six trees to represent this action.

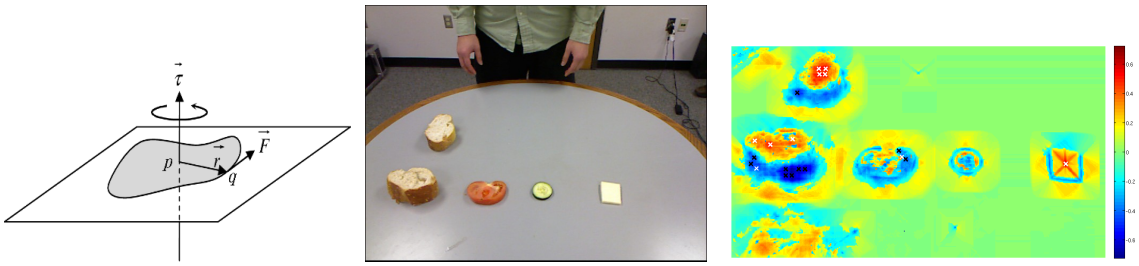
the concept of closed contours using bottom-up processes. Basically, the torque operator takes simple edges as input and computes, over regions of different sizes, a measure of how well the edges are aligned to form a closed, convex contour.

The underlying motivation is to find object-like regions by computing the "coherence" of the edges that support the object. Edge coherence is measured via the cross-product between the tangent vector of the edge pixel and a vector from a center point to the edge pixel, as shown in Figure 4(a). Formally, the value of torque, τ_{pq} of an edge pixel q within a discrete image patch with center p is defined as

$$\tau_{pq} = \|\vec{r}_{pq}\| sin\theta_{pq} \ , \tag{1}$$

where $\vec{r}_{pq}$ is the displacement vector from p to q, and θ_{pq} is the angle between $\vec{r}_{pq}$ and the tangent vector at q. The sign of τ_{pq} depends on the direction of the tangent vector and for this work, our system computes the direction based on the intensity contrast along the edge pixel. The torque of an image patch, P, is defined as the sum of the torque values of all edge pixels, $E(P)$, within the patch as

$$\tau_P = \frac{1}{2|P|} \sum_{q \in E(P)} \tau_{pq} \ . \tag{2}$$



(a) (b) (c)

Figure 4. (a) Torque for images, (b) a sample input frame, and (c) torque operator response. Crosses are the pixels with top extreme torque values that serve as the potential fixation points.

In this work, our system processes the testing sequences by applying the torque operators to obtain possible initial fixation points for the object monitoring process. Figure 4 shows an example of the application of the torque operator.

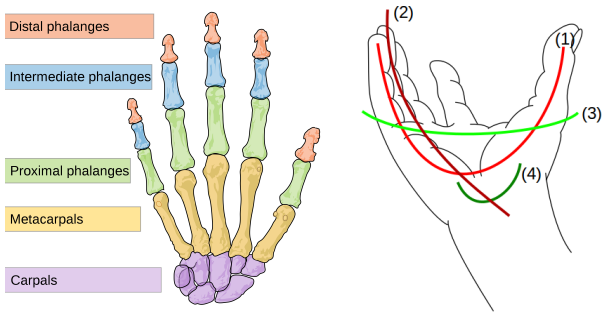
The system also employs a top-down attention mechanism; it uses the hand location to guide the attention. Here, we integrate the bottom-up torque operator output with hand tracking. Potential objects under manipulation are found when one of the hand regions intersects a region with high torque responses, after which the object monitoring system (Section 3.5) monitors it.

3.4 Hand Tracking, Grasp Classification and Action Recognition

With the recent development of a vision-based, markerless, fully articulated model-based human hand tracking system (Oikonomidis, Kyriazis, & Argyros, 2011) (`http://cvrlcode.ics.forth.gr/handtracking/`), the system is able to track a 26 degree of freedom model of hand. It is worth noting, however, that for a simple classification of movements into a small number of actions, the location of the hands and objects would be sufficient. Moreover, with the full hand model, a finer granularity of description can be achieved by classifying the tracked hand-model into different grasp types.

We collected training data from different actions, which then was processed. A set of bio-inspired features, following hand anatomy (Tubiana, Thomine, & Mackin, 1998), were extracted. Intuitively, the arches formed by the fingers are crucial to differentiate different grasp types. Figure 5 shows that the fixed and mobile parts of the hand adapt to various everyday tasks by forming bony arches: longitudinal arches (the rays formed by finger bones and associated metacarpal bones), and oblique arches (between the thumb and four fingers).

In each image frame, our system computed the oblique and longitudinal arches to obtain an eight parameter feature vector, as in Figure 6(a). We further reduced the dimensionality of the feature space by Principle Component Analysis and then applied k-means clustering to discover the underlying four general types of grasp, which are Rest, Firm Grasp, Delicate Grasp (Pinch) and Extension Grasp. To classify a given test sequence, the data was processed as described above and then the grasp type was computed using a naive Bayesian classifier. Figure 6(c) and (d) show examples of the classification result.



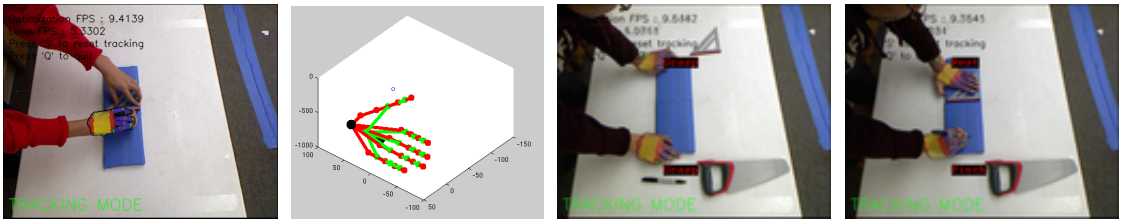
(a) (b)

Figure 5. (a) Bones of the human hand. (b) Arches of the hand: (1) one of the oblique arches; (2) one of the longitudinal arches of the digits; (3) transverse metacarpal arch; (4) transverse carpal arch. Source: Kapandji & Honoré, 2007.

The grasp classification is used to segment the image sequence in time and also serves as part of the action description. In addition, our system uses the trajectory of the mass center of the hands to classify the actions. The hand-tracking software provides the hand trajectories (of the given action sequence between the onset of grasp and release of the object), from which our system computed global features of the trajectory, including the frequency and velocity components. Frequency is encoded by the first four real coefficients of the Fourier transform in all the x, y and z directions, which gives a 24 dimensional vector over both hands. Velocity is encoded by averaging the difference in hand positions between two adjacent timestamps, which gives a six dimensional vector. These features are then combined to yield a 30 dimensional vector that the system uses for action recognition (Teo et al., 2012).

3.5 Object Monitoring and Recognition

Manipulation actions commonly involve objects. In order to obtain the information necessary to monitor the objects being worked on, our system applies a new method combining segmentation and tracking (Yang, Fermüller, & Aloimonos, 2013). This method combines stochastic tracking (Han



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

Figure 6. (a) One example of fully articulated hand model tracking, (b) a 3-D illustration of the tracked model, and (c-d) examples of grasp type recognition for both hands.

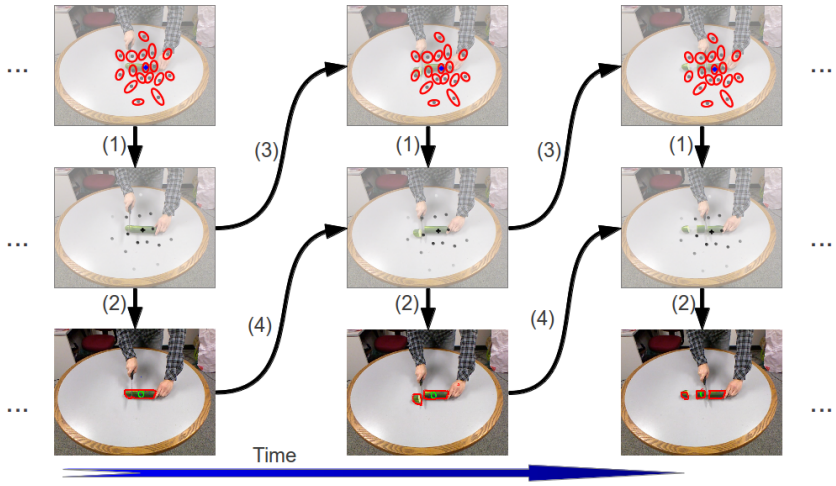


Figure 7. Flow chart of the active segmentation and tracking method for object monitoring: (1) Weighting; (2) weighted graph-cut segmentation; (3) points propagation and filtering; (4) update target tracking model.

et al., 2009) with a fixation-based active segmentation (Mishra, Fermüller, & Aloimonos, 2009). The tracking module provides a number of tracked points. The locations of these points define an area of interest and a fixation point for the segmentation. The data term of the segmentation module uses the color immediately surrounding the fixation points. The segmentation module segments the object and updates the appearance model for the tracker. Using this method, the system tracks objects as they deform and change topology (two objects can merge, or an object can be divided into parts.)

Figure 7 illustrates the method over time. The method used is a dynamic closed-loop process, where active segmentation provides the target model for the next tracking step and stochastic tracking provides the attention field for the active segmentation.

For object recognition, our system simply uses color information. The system uses a color distribution model to be invariant to various textures or patterns. A function $h(x_i)$ is defined to create a color histogram, which assigns one of the m-bins to a given color at location x_i. To make the algorithm less sensitive to lighting conditions, the system uses the Hue-Saturation-Value color space with less sensitivity in the V channel ($8 \times 8 \times 4$ bins). The color distribution for segment $s^{(n)}$ is denoted as

$$p(s^{(n)})^{(u)} = \gamma \sum_{i=1}^{I} k(\|y - x_i\|)\delta[h(x_i) - u] \,, \tag{3}$$

where $u = 1 \ldots m$, $\delta(.)$ is the Kronecker delta function and γ is the normalization term $\gamma = \frac{1}{\sum_{i=1}^{I} k(\|y-x_i\|)}$. $k(.)$ is a weighting function designed from the intuition that not all pixels in the sampling region are equally important for describing the color model. Specifically, pixels that are farther

away from the fixation point are assigned smaller weights,

$$k(r) = \begin{cases} 1 - r^2 & \text{if } r < a \\ 0 & \text{otherwise} \end{cases}, \tag{4}$$

where the parameter a is used to adapt the size of the region, and r is the distance from the fixation. By applying the weighting function, we increase the robustness of the distribution by weakening the influence of boundary pixels that may belong to the background or are occluded.

Since the objects in our experiments have distinct color profiles, the color distribution model used was sufficient to recognize the segmented objects. We manually labelled several examples from each object class as training data and used a nearest k-neighbours classifier. Figure 9 shows sample results.

3.6 Detection of Manipulation Action Consequences

Taking an object-centric point of view, manipulation actions can be classified into six categories according to how the manipulation transforms the object, or, in other words, what consequence the action has on the object. These categories are Divide, Assemble, Create, Consume, Transfer, and Deform.

To describe these action categories we need a formalism. We use the visual semantic graph (VSG) inspired by the work of Aksoy et al. (2011). This formalism takes as input computed object segments, their spatial relationship, and the temporal relationship over consecutive frames. An undirected graph $G(V, E, P)$ represents each frame. The vertex set $|V|$ represents the set of semantically meaningful segments, the edge set $|E|$ represents the spatial relations between any two segments. Two segments are connected when they share parts of their borders, or when one of the segments is contained in the other. If two nodes $v_1, v_2 \in V$ are connected, $E(v_1, v_2) = 1$, otherwise, $E(v_1, v_2) = 0$. In addition, every node $v \in V$ is associated with a set of properties $P(v)$ that describes the attributes of the segment. This set of properties provides additional information to discriminate between the different categories, and in principle many properties are possible. Here we use location, shape, and color.

The system needs to compute the changes in the object over time. Our formulation expresses this as the change in the VSG. At any time instance t, we consider two consecutive graphs, the graph at time $t-1$, denoted as $G_a(V_a, E_a, P_a)$ and the graph at time t, denoted as $G_z(V_z, E_z, P_z)$. We then use this formalism to represent four consequences, where $\leftarrow$ is used to denote the temporal correspondence between two vertices, $\nrightarrow$ is used to denote no correspondence:

- Divide: $\{\exists v_1 \in V_a; v_2, v_3 \in V_z | v_1 \leftarrow v_2, v_1 \leftarrow v_3)\}$ or
  $\{\exists v_1, v_2 \in V_a; v_3, v_4 \in V_z | E_a(v_1, v_2) = 1, E_z(v_3, v_4) = 0, v_1 \leftarrow v_3, v_2 \leftarrow v_4\}$ Condition (1)
- Assemble: $\{\exists v_1, v_2 \in V_a; v_3 \in V_z | v_1 \leftarrow v_3, v_2 \leftarrow v_3\}$ or
  $\{\exists v_1, v_2 \in V_a; v_3, v_4 \in V_z | E_a(v_1, v_2) = 0, E_z(v_3, v_4) = 1, v_1 \leftarrow v_3, v_2 \leftarrow v_4\}$ Condition (2)
- Create:$\{\forall v \in V_a; \exists v_1 \in V_z | v \nrightarrow v_1\}$ Condition (3)
- Consume:$\{\forall v \in V_z; \exists v_1 \in V_a | v \nrightarrow v_1\}$ Condition(4)

While the above actions can be defined purely on the basis of topological changes, there are no such changes for Transfer and Deform. Therefore, we must define them through changes in property:

- Transfer:$\{\exists v_1 \in V_a; v_2 \in V_z | P_a^L(v_1) \neq P_z^L(v_2)\}$ Condition (5)
- Deform: $\{\exists v_1 \in V_a; v_2 \in V_z | P_a^S(v_1) \neq P_z^S(v_2)\}$ Condition (6)

P^L represents properties of location, and P^S represents properties of appearance (e.g., shape or color). Figure 8 illustrates conditions (1) to (6). The active object monitoring process introduced in Section 3.5 to (1) finds correspondences ($\leftarrow$) between V_a and V_z; (2) monitors location property P^L and appearance property P^S in the VSG.

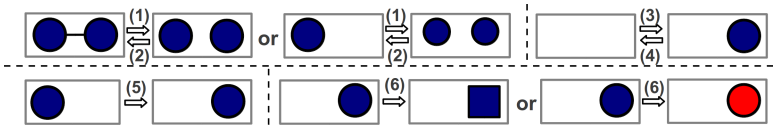


Figure 8. Illustration of the changes for condition (1) to (6).

Our system integrated visual modules in the following manner. Since hands are the most important components in manipulation actions, a state-of-the-art markerless hand tracking system obtains skeleton models of both hands. Using this data, our system classifies the manner in which the human grasps the objects into four primitive categories. On the basis of the grasp classification, our system finds the start and end points of action sequences. Our system then classifies the action from the hand trajectories, the hand grasp, and the consequence of the object (as explained above). To obtain objects, our system monitors the manipulated object using a process that combines stochastic tracking with active segmentation, then recognizes the segmented objects using color. Finally, based on the monitoring process, our system checks and classifies the effect on the object into one of four fundamental types of "consequences". The final output are sequences of "Subject" "Action" "Patient" triplets, and the manipulation grammar parser takes them as input to build up semantic structures.

4. Experiments

The theoretical framework we have presented suggests two hypotheses that deserve empirical tests: (a) manipulation actions performed by single human agents obey a manipulation action context-free grammar that includes manipulators, actions, and objects as terminal symbols; (b) a variant on chart parsing that includes both constructive and destructive actions, combined with methods for hand tracking and action and object recognition from RGBD data, can parse observed human manipulation actions.

To test the two hypotheses empirically, we need to define a set of performance variables and how they relate to our predicted results. The first hypothesis relates to representations, and we can empirically test if it is possible to manually generate target trees for each manipulation action in the test set. The second hypothesis has to do with processes, and can be tested by observing how many times our system builds a parse tree successfully from observed human manipulation

Table 2. "Hands", "Objects" and "Actions" involved in the experiments.

Category	*Hand*	*Object*	*Constructive Action*	*Destructive Action*
Kitchen	*LeftHand* *RightHand*	*Bread*, *Cheese*, *Tomato* *Eggplant*, *Cucumber*	*Grasp*, *Cut* *Assemble*	*Ungrasp*, *FinishedCut* *FinishedAssemble*
Manu *−facturing*	*LeftHand* *RightHand*	*Plank* *Saw*	*Grasp*, *Saw* *Assemble*	*Ungrasp*, *FinishedSaw* *FinishedAssemble*

actions. The theoretical framework for this consists of two parts: (1) a visual system to detect (subject, action, object) triplets from input sensory data and (2) a variant of the chart parsing system to transform the sequence of triplets into tree representations. Thus, we further separate the test for our second hypothesis into two sub-tasks: (1) we measure the precision and recall metrics by comparing the detected triplets from our visual system with human-labelled ground truth data and (2) given the ground truth triplets as input, we measure the success rate using our variant of chart parsing system by comparing it with the target trees for each action. We consider a parse successful if the generated tree is identical with the manually generated target parse. Therefore, we consider the second hypothesis supported when (1) our visual system achieves high precision and recall, and (2) our parsing system achieves a high success rate. We use the ground truth triplets as input instead of the detected ones because we cannot expect the visual system to generate the triplets with 100% precision and recall, due to occlusions, shadows, and unexpected events. As a complete system, we expect the visual module to have high precision and recall, thus the detected triplets can be used as input to the parsing module in practice.

We designed our experiments under the setting with one RGBD camera in a fixed location (we used a Kinect sensor). We asked human subjects to perform a set of manipulation actions in front of the camera while both objects and tools were presented within the view of the camera during the activity. We collected RGBD sequences of manipulation actions being performed by one human, and to ensure some diversity, we collected these from two domains, namely the kitchen and the manufacturing environments. The kitchen action set included "Making a sandwich", "Cutting an eggplant", and "Cutting bread", and the manufacturing action set included "Sawing a plank into two pieces" and "Assemble two pieces of the plank". To further diversify the data set, we adopted two different viewpoints for each action set. For the kitchen actions, we used a front view setting; for manufacturing actions, a side view setting. The five sequences have 32 human-labelled ground truth triplets. Table 2 gives a list of "Subjects", "Objects", and "Actions" involved in our experiments.

To evaluate the visual system, we applied the vision techniques introduced in Sections 3.3 to 3.6. To be specific, the grasp type classification module provides a "Grasp" signal when the hand status changes from "Rest" to one of the three other types, and an "Ungrasp" signal when it changes back to "Rest". At the same time, the object monitoring and the segmentation-based object recognition module provides the "Object" symbol when either of the hands touch an object. Also, the hand tracking module provides trajectory profiles that enable the trajectory-based action recognition module to produce "Action" symbols such as "Cut" and "Saw". The action "Assemble" did not have a distinctive trajectory profile, so we simply generated it when the "Cheese" merged with the "Bread" based on the object monitoring process. At the end of each recognized action, a

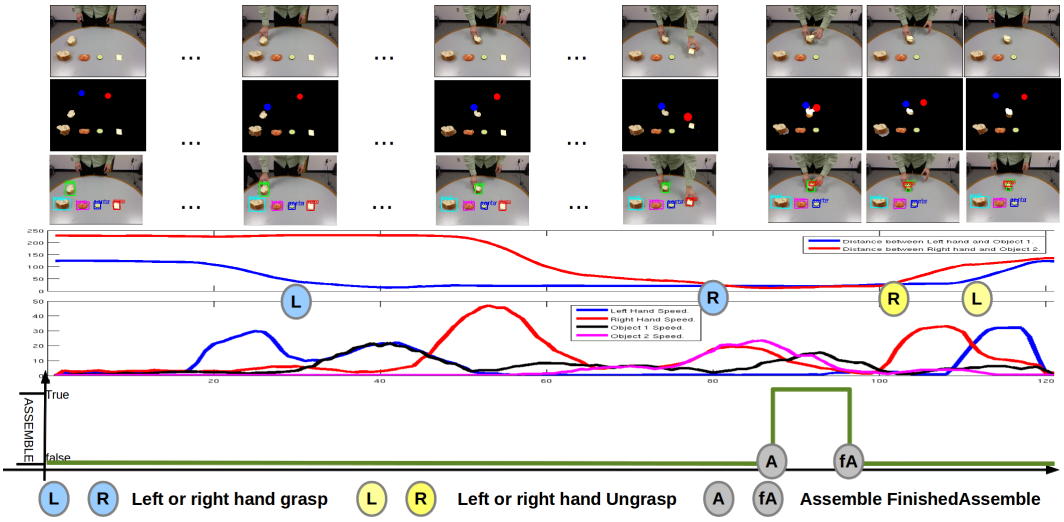


Figure 9. The second row shows the hand tracking and object monitoring. The third row shows the object recognition result, where each segmentation is labelled with an object name and a bounding box in different color. The fourth and fifth rows depict the hand speed profile and the Euclidean distances between hands and objects. The sixth row shows the consequence detection.

corresponding destructive symbol, as defined in Table 2, is produced, and the consequence checking module is called to confirm the action consequence. Figure 9 shows intermediate and final results of vision modules from a sequence of a person making a sandwich. In this scenario, our system reliably tracks, segments and recognizes both hands and objects, recognizes "grasp", "ungrasp" and "assemble" events, and generates a sequence of triplets along the time-line. To evaluate the parsing system, given the sequence of ground truth triplets as inputs, a sequence of trees (or forests) is created or dissolved dynamically using the manipulation action context free grammar parser (Section 3.1, 3.2).

Our experiments produced three results: (i) we were able to manually generate a sequence of target tree representations for each of the five sequences in our data set; (ii) our visual system detected 34 triplets, of which 29 were correct (compared with the 32 ground truth labels), and yielded a precision of 85.3% and a recall of 90.6%; (iii) given the sequence of ground truth triplets, our parsing system successfully parsed all five sequences in our data set into tree representations comparing with the target parses. Figure 10 shows the tree structures built from the sequence of triplets of the "Making a sandwich" sequence. More results for the rest of manipulation actions in the data set can be found at `http://www.umiacs.umd.edu/~yzyang/MACFG`. Overall, (i) supports our first hypothesis that human manipulation actions obey a manipulation action context-free grammar that includes manipulators, actions, and objects as terminal symbols, while (ii) and (iii) support our second hypothesis that the implementation of our cognitive system can parse observed human manipulation actions. [1]

1. As the experiments demonstrate, the system was robust enough to handle situations that involve hesitation, in which the human grasps a tool, finds that it is not the desired one, and ungrasps it (as in Figure 11).

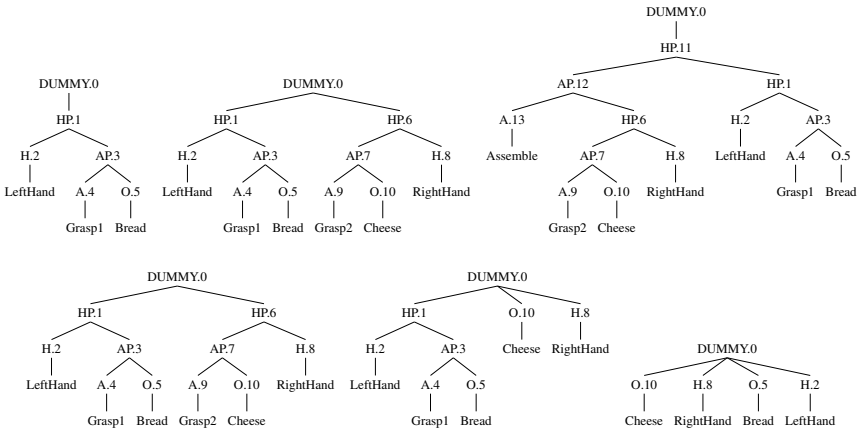


Figure 10. The tree structures generated from the "Make a Sandwich" sequence. Figure 9 depicts the corresponding visual processing. Since our system detected six triplets temporally from this sequence, it produced a set of six trees. The order of the six trees is from left to right.

The experimental results support our hypotheses, but we have not tested our system on a large data set with a variety of manipulation actions. We are currently testing the system on a larger set of kitchen actions and checking to see if our hypotheses are still supported.

5. Conclusion and Future Work

Manipulation actions are actions performed by agents (humans or robots) on objects that result in some physical change of the object. A cognitive system that interacts with humans should have the capability to interpret human manipulation actions. Our hypotheses are that (1) a minimalist generative structure exists for manipulation action understanding and (2) this generative structure organizes primitive semantic terminal symbols such as manipulators, actions, and objects into hierarchical and recursive representations. In this work, we presented a cognitive system for understanding human manipulation actions. The system integrates vision modules that ground semantically meaningful events in perceptual input with a reasoning module based on a context-free grammar and associated parsing algorithms, which dynamically build the sequence of structural representations. Experimental results showed that the cognitive system can extract the key events from the raw input and can interpret the observations by generating a sequence of tree structures.

In future work we will further generalize this framework. First, since the grammar is context-free, a direct extension is to make it probabilistic. Second, since the grammar does not assume constraints such as the number of operators, it can be further adapted to process scenarios with multiple agents doing complicated manipulation actions once the perception tools have been developed.

Moreover, we also plan to investigate operations that enable the system to reason during observation. After the system observes a significant number of manipulation actions, it can build a database of all sequences of trees. By querying this database, we expect the system to predict things such as which object will be manipulated next or which action will follow. Also, the action trees

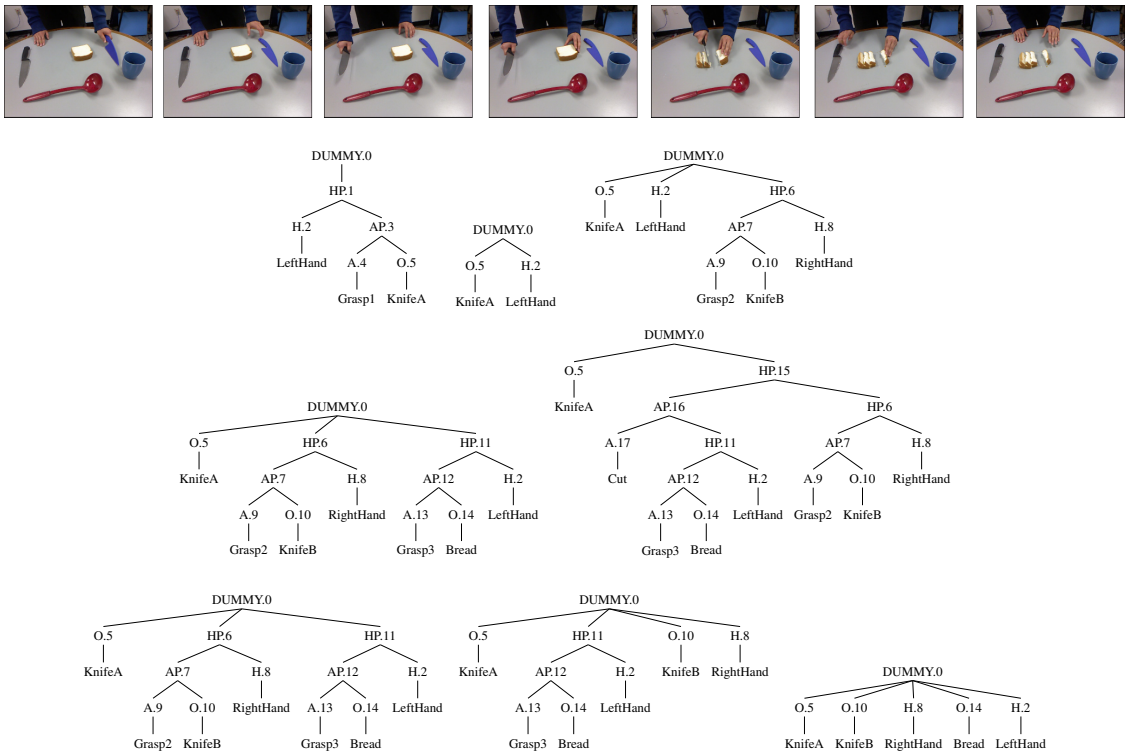


Figure 11. Example of the grammar that deals with hesitation. This figure shows key frames of the input visual data and the semantic tree structures.

could be learned not only from observation but also from language resources, such as dictionaries, recipes, manuals etc. This link to computational linguistics constitutes an interesting avenue for future research. Also, the manipulation action grammar introduced in this work is still a syntax grammar. We are currently investigating the possibility to couple manipulation action grammar rules with semantic rules using lambda expressions, through the formalism of combinatory categorial grammar developed by Steedman (2002).

6. Acknowledgements

An earlier version of this paper appeared in the Proceedings of the Second Annual Conference on Advances in Cognitive Systems in December, 2013. The authors thank the reviewers and editors for their insightful comments and suggestions. Various stages of the research were funded in part by the support of the European Union under the Cognitive Systems program (project POETICON++), the National Science Foundation under INSPIRE grant SMA 1248056, and support from the US Army, Grant W911NF-14-1-0384 under the Project: Shared Perception, Cognition and Reasoning for Autonomy. The first author was supported in part by a Qualcomm Innovation Fellowship.

References

Aein, J. M., Aksoy, E. E., Tamosiunaite, M., Papon, J., Ude, A., & Wörgötter, F. (2013). Toward a library of manipulation actions based on semantic object-action relations. *Proceedings of the 2013 EEE/RSJ International Conference on Intelligent Robots and Systems* (pp. 4555–4562). Tokyo: IEEE.

Aksoy, E., Abramov, A., Dörr, J., Ning, K., Dellen, B., & Wörgötter, F. (2011). Learning the semantics of object-action relations by observation. *The International Journal of Robotics Research*, *30*, 1229–1249.

Argall, B. D., Chernova, S., Veloso, M., & Browning, B. (2009). A survey of robot learning from demonstration. *Robotics and Autonomous Systems*, *57*, 469–483.

Ben-Arie, J., Wang, Z., Pandit, P., & Rajaram, S. (2002). Human activity recognition using multidimensional indexing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *24*, 1091–1104.

Brand, M. (1996). Understanding manipulation in video. *Proceedings of the Second International Conference on Automatic Face and Gesture Recognition* (pp. 94–99). Killington, VT: IEEE.

Chaudhry, R., Ravichandran, A., Hager, G., & Vidal, R. (2009). Histograms of oriented optical flow and Binet-Cauchy kernels on nonlinear dynamical systems for the recognition of human actions. *Proceedings of the 2009 IEEE International Conference on Computer Vision and Pattern Recognition* (pp. 1932–1939). Miami, FL: IEEE.

Chomsky, N. (1957). *Syntactic structures*. Berlin: Mouton de Gruyter.

Chomsky, N. (1993). *Lectures on government and binding: The Pisa lectures*. Berlin: Walter de Gruyter.

Dantam, N., & Stilman, M. (2013). The motion grammar: Analysis of a linguistic method for robot control. *Transactions on Robotics*, *29*, 704–718.

Dollár, P., Rabaud, V., Cottrell, G., & Belongie, S. (2005). Behavior recognition via sparse spatio-temporal features. *Proceedings of the Second Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance* (pp. 65–72). San Diego, CA: IEEE.

Fainekos, G. E., Kress-Gazit, H., & Pappas, G. J. (2005). Hybrid controllers for path planning: A temporal logic approach. *Proceedings of the Forty-fourth IEEE Conference on Decision and Control* (pp. 4885–4890). Philadelphia, PA: IEEE.

Gavrila, D. M. (1999). The visual analysis of human movement: A survey. *Computer vision and image understanding*, *73*, 82–98.

Guerra-Filho, G., Fermüller, C., & Aloimonos, Y. (2005). Discovering a language for human activity. *Proceedings of the AAAI 2005 Fall Symposium on Anticipatory Cognitive Embodied Systems*. Washington, DC: AAAI.

Guha, A., Yang, Y., Fermüller, C., & Aloimonos, Y. (2013). Minimalist plans for interpreting manipulation actions. *Proceedings of the 2013 International Conference on Intelligent Robots and Systems* (pp. 5908–5914). Tokyo: IEEE.

Han, B., Zhu, Y., Comaniciu, D., & Davis, L. (2009). Visual tracking by continuous density propagation in sequential Bayesian filtering framework. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *31*, 919–930.

Hu, C., Yu, Q., Li, Y., & Ma, S. (2000). Extraction of parametric human model for posture recognition using genetic algorithm. *Proceedings of the 2000 IEEE International Conference on Automatic Face and Gesture Recognition* (pp. 518–523). Grenoble, France: IEEE.

Ivanov, Y. A., & Bobick, A. F. (2000). Recognition of visual activities and interactions by stochastic parsing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *22*, 852–872.

Jackendorff, R. (1977). X-bar-syntax: A study of phrase structure. *Linguistic Inquiry Monograph*, *2*.

Kale, A., Sundaresan, A., Rajagopalan, A., Cuntoor, N., Roy-Chowdhury, A., Kruger, V., & Chellappa, R. (2004). Identification of humans using gait. *IEEE Transactions on Image Processing*, *13*, 1163–1173.

Kapandji, I. A., & Honoré, L. (2007). *The physiology of the joints*. Boca Raton, FL: Churchill Livingstone.

Laptev, I. (2005). On space-time interest points. *International Journal of Computer Vision*, *64*, 107–123.

Li, Y., Fermüller, C., Aloimonos, Y., & Ji, H. (2010). Learning shift-invariant sparse representation of actions. *Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition* (pp. 2630–2637). San Francisco, CA: IEEE.

Manikonda, V., Krishnaprasad, P. S., & Hendler, J. (1999). *Languages, behaviors, hybrid architectures, and motion control*. New York: Springer.

Mishra, A., Fermüller, C., & Aloimonos, Y. (2009). Active segmentation for robots. *Proceedings of the 20009 IEEE/RSJ International Conference on Intelligent Robots and Systems* (pp. 3133–3139). St. Louis, MO: IEEE.

Moeslund, T., Hilton, A., & Krüger, V. (2006). A survey of advances in vision-based human motion capture and analysis. *Computer Vision and Image Understanding*, *104*, 90–126.

Moore, D., & Essa, I. (2002). Recognizing multitasked activities from video using stochastic context-free grammar. *Proceedings of the National Conference on Artificial Intelligence* (pp. 770–776). Menlo Park, CA: AAAI.

Nishigaki, M., Fermüller, C., & DeMenthon, D. (2012). The image torque operator: A new tool for mid-level vision. *Proceedings of the 2012 IEEE International Conference on Computer Vision and Pattern Recogntion* (pp. 502–509). Providence, RI: IEEE.

Oikonomidis, I., Kyriazis, N., & Argyros, A. (2011). Efficient model-based 3D tracking of hand articulations using Kinect. *Proceedings of the 2011 British Machine Vision Conference* (pp. 1–11). Dundee, UK: BMVA.

Pastra, K., & Aloimonos, Y. (2012). The minimalist grammar of action. *Philosophical Transactions of the Royal Society: Biological Sciences*, *367*, 103–117.

Ryoo, M. S., & Aggarwal, J. K. (2006). Recognition of composite human activities through context-free grammar based representation. *Proceedings of the 2006 IEEE Conference on Computer Vision and Pattern Recognition* (pp. 1709–1718). New York: IEEE.

Saisan, P., Doretto, G., Wu, Y., & Soatto, S. (2001). Dynamic texture recognition. *Proceedings of the 2001 IEEE International Conference on Computer Vision and Pattern Recognition* (pp. 58–63). Kauai, HI: IEEE.

Schank, R. C., & Tesler, L. (1969). A conceptual dependency parser for natural language. *Proceedings of the 1969 Conference on Computational Linguistics* (pp. 1–3). Sanga-Saby, Sweden: Association for Computational Linguistics.

Steedman, M. (2002). Plans, affordances, and combinatory grammar. *Linguistics and Philosophy*, *25*, 723–753.

Summers-Stay, D., Teo, C., Yang, Y., Fermüller, C., & Aloimonos, Y. (2013). Using a minimal action grammar for activity understanding in the real world. *Proceedings of the 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems* (pp. 4104–4111). Vilamoura, Portugal: IEEE.

Teo, C., Yang, Y., Daume, H., Fermüller, C., & Aloimonos, Y. (2012). Towards a Watson that sees: Language-guided action recognition for robots. *Proceedings of the 2012 IEEE International Conference on Robotics and Automation* (pp. 374–381). Saint Paul, MN: IEEE.

Tsotsos, J. (1990). Analyzing vision at the complexity level. *Behavioral and Brain Sciences*, *13*, 423–469.

Tubiana, R., Thomine, J.-M., & Mackin, E. (1998). *Examination of the hand and the wrist*. Boca Raton, FL: CRC Press.

Turaga, P., Chellappa, R., Subrahmanian, V., & Udrea, O. (2008). Machine recognition of human activities: A survey. *IEEE Transactions on Circuits and Systems for Video Technology*, *18*, 1473–1488.

Wang, L., & Suter, D. (2007). Learning and matching of dynamic shape manifolds for human action recognition. *IEEE Transactions on Image Processing*, *16*, 1646–1661.

Willems, G., Tuytelaars, T., & Van Gool, L. (2008). An efficient dense and scale-invariant spatio-temporal interest point detector. *Proceedings of the 2008 IEEE European Conference on Computer Vision* (pp. 650–663). Marseille, France: Springer.

Wörgötter, F., Aksoy, E. E., Kruger, N., Piater, J., Ude, A., & Tamosiunaite, M. (2012). A simple ontology of manipulation actions based on hand-object relations. *IEEE Transactions on Autonomous Mental Development*, *1*, 117–134.

Yang, Y., Fermüller, C., & Aloimonos, Y. (2013). Detection of manipulation action consequences (MAC). *Proceedings of the 2013 IEEE Conference on Computer Vision and Pattern Recognition* (pp. 2563–2570). Portland, OR: IEEE.

Yilmaz, A., & Shah, M. (2005). Actions sketch: A novel action representation. *Proceedings of the 2005 IEEE International Conference on Computer Vision and Pattern Recognition* (pp. 984–989). San Diego, CA: IEEE.

Younger, D. H. (1967). Recognition and parsing of context-free languages in time n^3. *Information and Control*, *10*, 189–208.