Action-Based Learning: A Proposal for Robust Gesture & Object Recognition using Spatio-Temporal Domain Models

Matthew Devlin ***Yiannis Aloimonos***

Abstract

In recent years, there has been a significant amount of effort into object recognition techniques as well as vision-based learning algorithms. But view invariance, complex gesture-object interactions, and the lack of reasoning in vision systems continues to make this an incredibly challenging problem. In this paper, we propose an Action-Based Learning model in which the AI learns gestures, objects, and their relationships by observing videos of humans completing a task by performing gestures on objects present in the scene.

Keywords: *3D Object Recognition, 3D Action & Gesture Recognition, Semi-Supervised Machine Learning, Spatio-Temporal Domain Models.*

1. Introduction

Learning through observation is a fundamental aspect of human learning, as it enables us to learn how to recognize new objects and perform new actions with them. The goal of this paper is to propose a new model for an intelligent vision-based learning system, capable of utilizing domain knowledge and reasoning about objects in order to improve accuracy and performance.

Problem: Given a video of a human completing a task, generate a list of instructions for another human to complete the same task, including the objects and interactions.

From the videos, we are trying to learn a sequence of ***Actions*** that are performed in order to complete a high-level ***Task***. An action is composed of a ***Gesture*** performed on ***Objects***.

For this project, we focus on processing an entire continuous video that consists of a human completing some task, rather than processing individual images. This provides several advantages for recognizing objects and gestures, because we are able to use a number of techniques that are not as effective when working with images, such as motion, accurate depth data, full skeletal tracking, and easier tracking of humans and objects present in the scene**.** Another benefit of using a continuous video is that we are able to analyze the scene at any point in the time range, which can be used to (a) improved recognition of occluded objects by searching for a time **t** where the object is more visible, (b) understand the effects an action has on the environment by examining the state of objects before and after an action occurs, (c) sequential action recognition using rule based learning to categorize what actions the human is performing and learn new actions based on existing ones.

Another benefit most papers don't look at: We are assuming we have a human in the scene interacting with these objects. Object affordance + gesture recognition can be combined to improve accuracy. Combined with domain knowledge involving actions (and their preconditions/typical sequentially performed order), we can create an intelligent system without creating a general recognition system.

Existing object recognition algorithms have fairly high accuracy rates, depending on the training and testing datasets used. Object recognition remains a challenging problem due to large intra-class variability and inter-class similarity, which makes accurate recognition relying only on RGB data a difficult challenge [21]. In addition to processing RGB with depth data, we have access to a video feed consisting of time frames which may also contain motion which is called a Spatio-temporal relationship [19]. We can leverage this to improve the performance of our algorithm, such as identifying occluded objects by finding a time **t**, where the object is not occluded. Processing video footage also means that we can analyze the effects of an action on the environment by looking at the scene before and after the action occurs.

Another advantage of this system is that we can use the human in the scene to determine which objects are

relevant to the actions being performed. Segmenting and recognizing all the objects in a given scene is a far more difficult problem than recognizing a single object being manipulated by a human. When a human interacts with an object, we usually gain additional views of the object from the camera's perspective, such as when an object is physically moved or rotated. Some actions may not provide any useful additional perspectives of the object, such as flipping a light switch, but the significance is the change in state of the light switch.

2. Related Work

There has been extensive research into 3D object and gesture recognition, and many frameworks are available to process the RGB-D footage gathered from sensors like the Kinect. In addition to depth data, the Kinect also provides recognition and tracking for human body masks as well as individual body joints. These features are especially useful for recognizing gestures involving large amounts of movement between distinguishable joints. Recognizing smaller, more detailed actions such as gestures involving only the hands can be much more difficult but there have been several successful projects to recognize these gestures including [8] and [9].

Since our definition of an action is a sequence of gestures performed on at least one object, we can limit our search for potential actions to time intervals where an object interaction is possible. Locality is an obvious restriction to impose on object interactions, meaning the human must be near an object for it to be considered the target of an action. Sections 3.1 and 3.2 focus on the vision aspects of this system, for recognizing objects and gestures using RGB-D and skeleton data.

In Section 3.4, the results of our vision system (consisting of object and gesture recognition) is processed using a Spatio-temporal model and knowledge about relationships between objects and gestures, gathered from the “cognitive dialog” described in Section 3.3. These relationships can be manually programmed into a system, but the goal is to develop an intelligent system that learns to ask complex questions based on the vision system’s answers to simple questions to formulate more complex questions based on domain knowledge.

In recent years, there has been a focus on Spatio-temporal structures that focus on footage from RGB-D videos consisting of a human performing 3D motions over discrete time frames that aim to learn human actions using conditional random fields (CRF) and probabilistic models to formulate better predictions based on past actions and potential future actions [16, 17].

3. Approach

3.1 Notable Object Tracking & Recognition

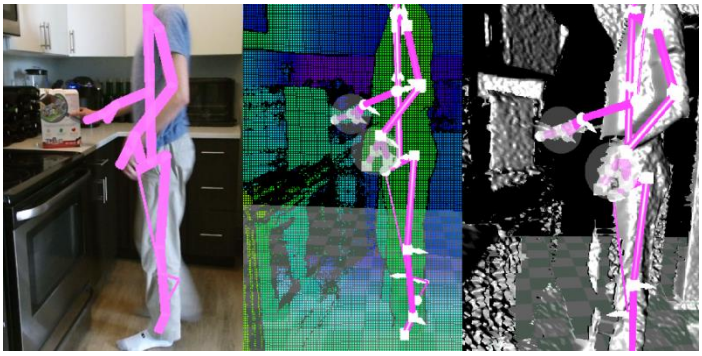


Figure A: In the above image, a human is reaching for a cereal box, but due to the camera angle and distance, it is difficult for the system to make an accurate prediction about the cereal box, until it is brought into a better view as shown in ***Figure B***.

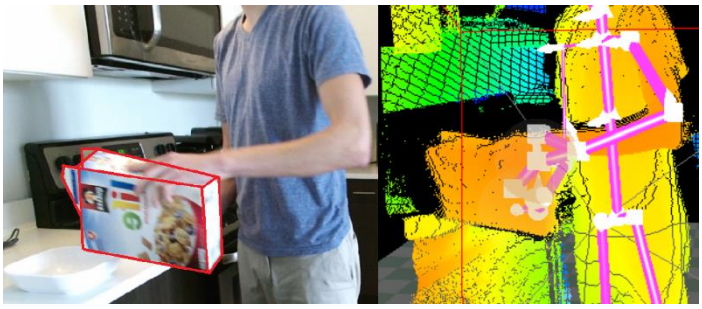


Figure B: Left shows a much clearer RGB image that provides much better views for texture-based recognition. Right shows depth point cloud which is much more accurate from this view.

While accurate 3D object recognition remains a difficult problem, many algorithms have been developed to address this problem. To process the RGB-D video footage, we are using the Point Cloud Library (PCL) which contains implementations for most of the techniques discussed below [2]. PCL

contains a multitude of useful functions, including object segmentation, numerous implementations for 3D object recognition, and 6-DOF pose estimations for an object.

For this project, object recognition does not need to be performed on the entire scene, since we are only interested in objects that are the target of an action. This allows us to limit our search space to objects located near a human when a gesture is being performed. To quickly determine if an object exists near the human's location, we can use a Hough Voting approach adopted for 3D space [1].

To perform object recognition, there exists two popular types of algorithms: geometric feature-based recognition and pattern-based recognition [7]. In geometric feature-based learning, we extract *descriptors* from a 3D object that describe the important visual features of an object such as its 3D shape, the textures at particular points, depth data, and other distinctive attributes [5]. There exist a large number of descriptors that focus on different features, but we can divide descriptors into two types: local and global. A local surface feature describes the attributes for a particular point on an object, and local descriptors tend to be more robust to occlusion and clutter, which are common in real-world environments [3]. A global descriptor provides a high-dimensional representation of an object's geometry and is used for recognition, geometric categorization, and shape retrieval.

PCL provides many state-of-the-art implementations for both local and global descriptors which are frequently used for 3D object recognition including Hierarchal Matching Pursuit (HMP) [13], Histograms of Orientations (HOG), Point Feature Histograms (PFH), Spin Images (SIs) [14], and Clustered Viewpoint Feature Histogram (CVFH) [15]. Once computed, these descriptors can be compared against the feature points of an object in a scene to perform object recognition and classification using algorithms like correspondence grouping, hypothesis verification, and LINEMOD. To learn new objects, we add "templates" to our object database, which describe the shape and texture of an object from a particular camera angle. These recognition algorithms typically have accuracies ranging between ~50% and ~85% on most RGB-D image datasets, but we can improve this accuracy by incorporating an Action-Based model that incorporates domain knowledge about objects and gestures. This is implemented in the form of a reasoning executive and domain model, which are described in Section 3.3.

Another interesting problem pertaining to object recognition is understanding an object's functionality based on its geometric features. The potential functions of an object are generally referred to as its *affordances*. Myers et al. propose an approach to detect object affordance based on superpixel HMP and random structured forests [11]. These affordances can be used in our domain model to represent likely interactions between certain classes of objects and classes of gestures. There are also several affordance learning approaches consisting of both supervised [12] and unsupervised [13] learning algorithms using 6-DOF pose estimation and HMP to learn affordances. These algorithms have been studied in Spatio-temporal models, based on RGB-D footage of humans performing actions which involves both 3D motions and a temporal component [14].

3.2 Gesture Recognition

Gesture recognition remains a difficult problem due to the number of potential joint orientations and the Spatio-Temporal aspects of gesture recognition. Since a gesture is composed of gesture-parts over time, the same approaches used in object recognition are not applicable. Fortunately, gesture recognition is a heavily researched area and there exist numerous algorithms that analyze the RGB-D or skeleton joint data over time to recognize gestures. Xia et al. developed an algorithm that uses histograms of 3D joint locations (HOJ3D) combined with a Hidden Markov Model to represent the temporal evolution of joints over time [4]. A major advantage of this algorithm is that it is view invariant, meaning it will recognize the gestures regardless of the human's orientation with respect to the camera. Amor et al. proposed a similar gesture recognition system that uses rate-invariant analysis of the skeletal shape trajectories. This system focuses on temporal registration which makes gesture recognition more robust by recognizing the same gesture regardless of

how fast it is performed [22]. There also exist gesture recognition algorithms that use Hidden Markov Models to provide more accurate skeletal tracking during actions in a Spatio-temporal domain, which is based on the calculation of spherical angles between selected joints, which is also invariant to viewpoints [31].

Another popular approach is training a multi-class Support Vector Machines (SVM) on the joint differences between subsequent frames. We can leverage the Kinect's advanced joint tracking to get an accurate motion profile for each joint, which can be used to efficiently perform gesture recognition using the trained SVM [10]. However, while the SVMs used are efficiently computed they tend to be less accurate than more robust methods utilize additional properties at the cost of some efficiency, which is not a problem for our *offline* system. Also, we are only concerned with gestures that involve interactions with objects such as using tools, meaning we can ignore gestures that do not involve a nearby or held object [28].

An even more difficult problem is correctly recognizing Hand-Gestures since hands are much smaller compared to the rest of our body, and have more complex articulations that make them prone to segmentation errors [6]. To address this, there exist many algorithms that focus specifically on hand localization and recognition, that use techniques capable of handling noisy shapes, such as Finger-Earth Mover's Distance (FEMD) to measure differences between two hand shapes [9]. By using techniques specifically for hands, we can achieve much higher accuracy for recognizing small Hand-Gestures, with accuracy of around 90%. Additionally, these techniques can be used to identify the grip of a hand on an object, even when it is occluded.

3.3 Reasoning Executive & Domain Model

Using the algorithms described in Sections 3.1 and 3.2, we can develop a robust vision framework for recognizing 3D objects and gestures, if we are given enough training data for all the desired objects and gestures that we want to learn. Requiring manual training in our framework would severely limit the usefulness of our proposed system. Creating a good model for a 3D object or gesture is time consuming, and whenever the system needs to learn a new object or gesture, the training data must be manually added to the system's knowledge.

A much better approach is to design an intelligent system that incorporates a domain model and a reasoning executive, in addition to our described vision system. The goal of the reasoning executive is to model domain knowledge about objects and gestures, and communicate with the vision system. We call this communication between the vision system and the reasoning executive "cognitive dialog". Through this dialog, the vision system can ask the reasoning executive if a particular object or gesture prediction makes sense in the context of the domain model given the state of the human and the other objects in the scene, which can help us correctly recognize known or new objects. Since this system is not being run *online,* we can spare some speed efficiency for higher accuracy, and use this to develop a learning system based on semantic representation [29].

The reasoning executive utilizes the same three methods of reasoning used by humans: (1) deduction, (2) induction, and (3) abduction, to evaluate semantic properties and expand on our domain knowledge.

(1) Through deduction, we can enable a rule based learning approach for our system to form conclusions on its own based on previously known "facts". For example, if our reasoning system knows that knives are associated with the cutting gesture and our vision system detects the cut gesture with a different but similar shaped knife, we can conclude that there exists a relationship between the two knife objects [27]
(2) Induction is forming generalizations based on observed instances. This is harder to formulate in an intelligent system than deduction because it attempts to generalize relationships based on what we've seen so far. When we have a large sample size of objects and gestures that all hold true for some statement, it is likely safe to form generalizations. We only perform induction when the size of samples is large enough to minimize error the

chances of overgeneralizing objects through inductive reasoning

(3) Abduction is when the major premise is evident or observable, but the minor premise and therefore conclusion is only probable. We can use statistical models to perform abduction by choosing the most likely conclusion based on past observed situations

Formal Definitions:

- Task([Action]): A task is a sequence of Actions ordered by the time they occur
- Action([Gesture]): An action is a sequence of Gestures performed on one or more objects
- Gesture([GesturePart], Object): In this Action-Based learning model, a gesture is defined as a sequence of upper-body motions performed either (a) on an object or (b) a movement towards an object or location
- GesturePart(SkeletonData): A part of a Gesture that includes skeleton data of the gesture segment being performed. Generated from Kinect depth sensor to track body motions (emphasis on hands and arms)
- SceneObject(XYZ, 3DPointCloud): A SceneObject is an Object that is being tracked during video processing. SceneObjects have a 3D vector representing its (X,Y,Z) position as it is tracked in the scene. SceneObjects also have a 3DPointCloud gathered by a depth sensor camera like the Kinect
- ObjectTemplate({3DPointCloud_Views}, name, {ObjectStates}, {Classes}): An object model learned by our system, stored in the database The 3DPointCloud is either provided or generated by tracking SceneObjects. The set of ObjectStates represents the possible states of the object, including 3D models representing the change in appearance, if applicable. The set of classifiers are either strong (highly likely) or weak (possible class relationship through observation).
- ObjectState(Type, 3DPointCloud, Value): A 3DPointCloud of the object when it is in this state. Type refers to the type of state this object possess such as Binary (switch), Radial (Rotate by a Vector, Turn to a Value), Pressable (Keyboard, Buttons).

3.4 Action Recognition

Different classes of objects are interacted with through different types of gesture/manipulations. The result is an action, which has some observable effect on the state of the objects in the scene. The effects of humans on a scene has been studied using a number of deep learning techniques including CRFs [19], unsupervised learning [20], CNNs [21], RNNs [23], and radius-margin bounded neural networks [24].

In humans, memory formation depends heavily on changes [25]:

- When humans perform actions, they actively change the environment and the states of objects
- We are only interested ***deliberate actions*** that involve object interactions changing its state. So instead of trying to solve general gesture recognition, we only need to examine gestures that directly involve objects. This can be used to significantly reduce false positives during gesture recognition, by filtering out gestures that do not involve objects
- Some actions may be more recognizable by their state than from observing a gesture being performed (e.g. when there is a lot of occluded motion). For example, an action as simple as flipping a light switch on, there are many different ways to position your hand to flip the switch (grasp with two fingers, pull down with entire hand, or even use your wrist). For this type of action, we are more concerned about the object's state after the action occurs than we are about the specific hand motion or how they grasp the switch. WE can introduce Object-States to the system's domain knowledge to allow for more advanced reasoning about objects by analyzing their state's over the time period of the video as a human interacts with it.

3.5 Handling Special Cases: Occlusion, Symmetry

- **Occlusion**: During a gesture, parts of an object (or the entirety of it) may become occluded, causing difficulty with recognition. In a Spatio-temporal domain, we can solve this using several techniques:
  - (a) domain knowledge of an object's possible states
    - E.g. learn to understand the state and possible interactions of an object and learn to recognize differences between states
  - (b) grip position and orientation to approximate object shape based on visible hand or object parts
  - (c) infer object symmetry based on grip for partially occluded objects to generate a more accurate model

4. Implementation Details

4.1 Actions & Relationship Graph

Goal: Model high-level relationships between Actions, Objects, & Gestures and learn new relationships from this model.

- Object ←→ Object
  - Identify similar geometric shapes or feature descriptors
  - Object States
- Gesture ←→ Object
  - Gestures performed on an object (conditions that must hold true to perform action)
- Action ←→ Object
  - Object state before and after an Action
- Classification Relationships
- State Logic Rules

Suppose in our Kinect footage, the Task being performed by the human is making a bowl of cereal and milk. We can look at this simple task as three High-Level Actions (which are composed of a sequence of Low-Level Actions). The three Actions can be modeled as:

Action A: **Put** [Bowl] *on* [Counter]

Action B: **Pour** [Cereal] *into* [Bowl]

Action C: **Pour** [Milk] *into* [Bowl]

We may observe other High-Level Actions being performed in the footage, such as putting the cereal and milk back into their original locations (i.e. cabinet for cereal, fridge for milk). These are called "cleanup" actions because they revert the effects of a past action and restore objects in the environment to their original state. For this Task, let's assume we observed two "cleanup" actions for Actions B and C:

Action B': **Put** [Cereal] *inside* [Cabinet]

Action C': **Put** [Milk] *inside* [Fridge]

Additionally, tasks may or may not depend on a strict ordering of actions: Actions B and C are dependent on the [Bowl] object retrieved in Action A, so we can infer that A must come before these actions every time the task is performed. The same applies to the "cleanup" actions: B must come before B' and C must come before C', otherwise the objects will not be in valid states. We can provide formal definitions to model an action's preconditions (based on temporal state variables) and its effects on the human, objects, or environment. For example, here is the definition for the "Pour" Action:

Action: Pour

Notation*: **Pour*** [Primary Object] into [Secondary Object] with *{Hand}*

Preconditions

P1. Involves Two Objects:

- [Primary Object]: The object being poured
- [Secondary Object]: The object being poured into

P2. [Primary Object] is currently held in *{Hand}*

P3. [Primary Object] *is located above* [Secondary Object]

P4. {Primary Hand} *is rotating towards* [Secondary Object]

Effects

E1. Transfer Contents of [Primary Object] into [Secondary Object]

Observed Relationships

O1. [Primary Object] is typically of class "Container"

O2. [Secondary Object] is typically of class "Container"

4.2 Action Graphs

Note on Processing Temporal Footage: Our Kinect footage is a sequence of time frames that can be labelled $[t_1, \ldots, t_n]$ where t_1 is the first frame and t_n is the last frame, and t_i denotes a time frame such that $t_1 <= t_i <= t_n$. Some preconditions can be evaluated for every individual time, such as determining if the human is "*Near*" an object [o] at time t_i. This just requires a check to see if Human.Near([o], t_i) is true, which we can evaluate using the skeletal tracking provided in the time frame and object tracking system discussed in Section 3.1 of the paper. Some conditions need to be evaluated over a sequence of intervals, such as gestures which are spatio-temporal and need to be evaluated with respect to both space and time. For example, to determine if a hand is rotating an object we need a time interval t_i to t_j, where $t_i < t_j$, to analyze so we can track the motion of body joints and objects over the time interval and perform gesture recognition. This is done using a combination of techniques described in Section 3.2, including temporal registration and part-based hand gesture recognition.

- **Actions (Diamond Nodes):** The definition of an action, which consists of a name and a list of parameters that are used to perform the Action. These parameters can be objects, positional or rotational vectors, or human state variables (i.e. the states and locations of hands/body joints).
- **Preconditions:** A conditional that operates on parameters to determine if the action can be currently performed based on the states and values of the parameters. If any predicate node predicts that it is unlikely for the condition to be true at that time, we can determine that the Action is likely not a match. To represent preconditions in our graph, we divide them into two categories:
    - **Instantaneous Predicates (Circle Nodes):** Represent predicates that can be evaluated at individual time frames.
    - **Temporal Predicates (Hexagon Nodes):** Represent predicates that operate on a sequence of time intervals such as recognizing gestures and tracking object motion (i.e. spatio-temporal)
- **Effects (Rectangle Nodes):** The effect of the action on the environment, which can only be performed when all the conditions are likely to be true. Effects change the state of objects and the environment.

4.3 Integrating Domain Knowledge

Using *domain knowledge*, we can improve our accuracy when performing object and gesture recognition using the state-based predicate system shown in the Action Graphs. Instead of relying only on vision, we incorporate the state of the environment and the actions being performed using the object. If we are able to identify the action being performed, we can use that knowledge to improve 3D object recognition accuracy by searching known objects of the class "Container". After observing the entire footage and identifying all the performed actions, the result can be modeled as a Task Graph (example on next page), which shows the High-Level and Low-Level Actions that make up the Task. Note that the Task Graph doesn't show all the knowledge learned such as object models, gestures, or object classes.

Handling Occlusion: During an Action, it is very likely for objects and body parts to become occluded, making it impossible to determine which gesture was performed or the object that was interacted with. This is where tracking the effects of an action on the environment's state is useful, because we can predict which action was performed in the past, based on its effect. For example, suppose that when picking the milk up from the fridge, the gesture and milk cartoon are occluded during the duration of the action. We can still determine that the PickUp action was performed on [Milk], by examining changes in the scene to predict which action was performed based on the state of the environment before and after the action. For example, during Action C when the hand and object become visible while pouring the milk, we can see an object is now being held and perform 3D OR to recognize the held object as [Milk]. We can determine that this was the result of a past action and use domain knowledge to predict which action was performed and when, even if we cannot perform accurate recognition during the action itself.

Figure C: Action Graphs for Pour, WalkTo, PickUp, and PutDown

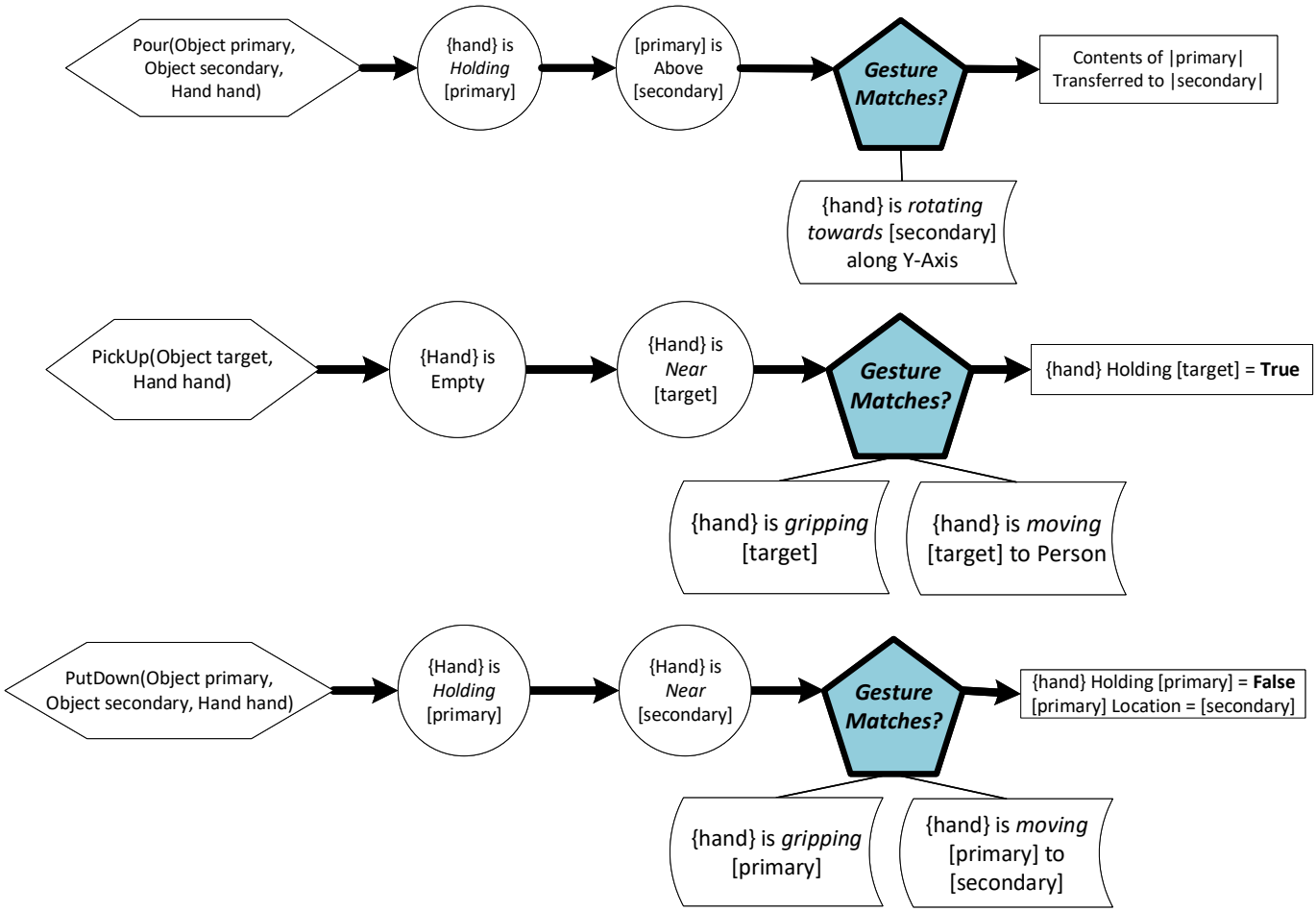
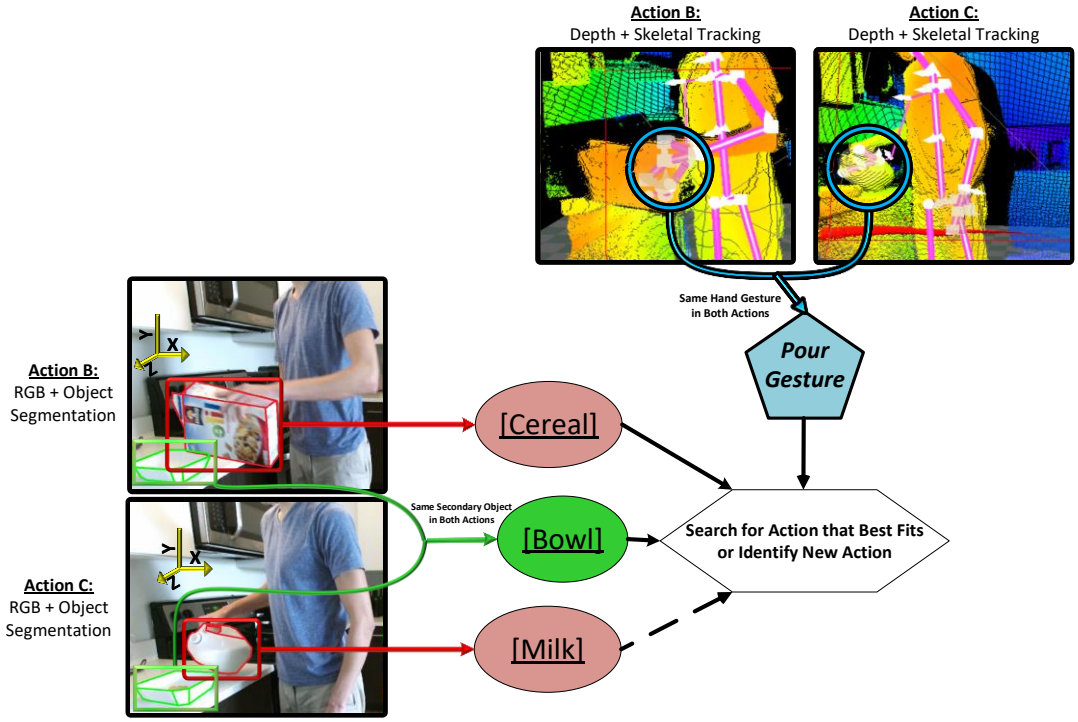
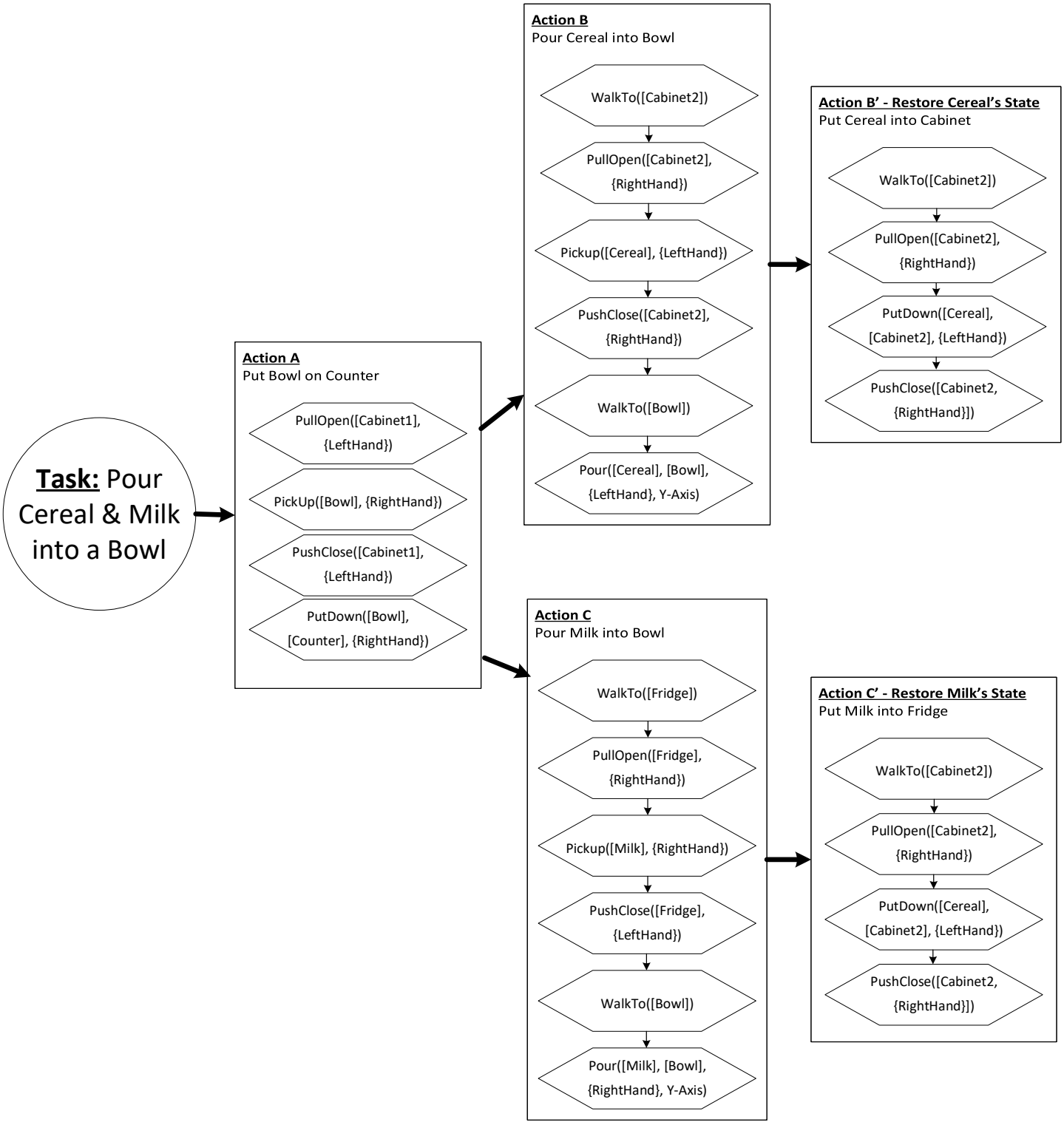


Figure D: Actions B and C being performed with Object and Gesture Recognition, to find the Action that best matches based on domain knowledge and the current state of the environment.



4.4 Task Graph

Below is a diagram of the Task which shows the High-Level Actions (A, B, C), the Cleanup Actions (B', C'), and the Low-Level Actions they are composed of (Diamonds). We can determine the High-Level Action by looking at the effects of an Action minus its Cleanup Action (if one exists). The set of effects in B – B', is just the Pour effect, because B' restored the cereal to its original state, leaving Pour as the only action that resulted in an observable effect to complete the Task.



5. Conclusion

This paper researched state-of-the-art techniques for handling object recognition, gesture recognition, Spatio-temporal data processing, and a proposed system which incorporates a reasoning executive which improve accuracy over just using vision data to classify and learn objects.

References

[1] Tombari, F., & Di Stefano, L. (2010, November). Object recognition in 3D scenes with occlusions and clutter by Hough voting. In *Image and Video Technology (PSIVT), 2010 Fourth Pacific-Rim Symposium on* (pp. 349-355). IEEE.

[2] Aldoma, A., Marton, Z. C., Tombari, F., Wohlkinger, W., Potthast, C., Zeisl, B., & Vincze, M. (2012). Tutorial: Point cloud library: Three-dimensional object recognition and 6 dof pose estimation. *IEEE Robotics & Automation Magazine*, *19*(3), 80-91.

[3] Guo, Y., Bennamoun, M., Sohel, F., Lu, M., & Wan, J. (2014). 3D Object Recognition in Cluttered Scenes with Local Surface Features: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *36*(11), 2270-2287.

[4] Xia, L., Chen, C. C., & Aggarwal, J. K. (2012, June). View invariant human action recognition using histograms of 3d joints. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2012 IEEE Computer Society Conference on* (pp. 20-27). IEEE.

[5] Llarena, A., Boldt, J. F., Steinke, N. S., Engelmeyer, H., & Rojas, R. BerlinUnited@ Home 2013 Team Description Paper.

[6] Suarez, J., & Murphy, R. R. (2012, September). Hand gesture recognition with depth images: A review. In *RO-MAN, 2012 IEEE* (pp. 411-417). IEEE.

[7] Janoch, A., Karayev, S., Jia, Y., Barron, J. T., Fritz, M., Saenko, K., & Darrell, T. (2013). A category-level 3d object dataset: Putting the Kinect to work. In *Consumer Depth Cameras for Computer Vision* (pp. 141-165). Springer London.

[8] Lv, X., Jiang, S. Q., Herranz, L., & Wang, S. (2015). RGB-D hand-held object recognition based on heterogeneous feature fusion. *Journal of Computer Science and Technology*, *30*(2), 340.

[9] Ren, Z., Yuan, J., Meng, J., & Zhang, Z. (2013). Robust part-based hand gesture recognition using Kinect sensor. *IEEE transactions on multimedia*, *15*(5), 1110-1120.

[10] Biswas, K. K., & Basu, S. K. (2011, December). Gesture recognition using Microsoft Kinect®. In *Automation, Robotics and Applications (ICARA), 2011 5th International Conference on* (pp. 100-103). IEEE.

[11] Myers, A., Teo, C. L., Fermüller, C., & Aloimonos, Y. (2015, May). Affordance detection of tool parts from geometric features. In *Robotics and Automation (ICRA), 2015 IEEE International Conference on* (pp. 1374-1381). IEEE.

[12] Aldoma, A., Tombari, F., & Vincze, M. (2012, May). Supervised learning of hidden and non-hidden 0-order affordances and detection in real scenes. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on* (pp. 1732-1739). IEEE.

[13] Bo, L., Ren, X., & Fox, D. (2013). Unsupervised feature learning for RGB-D based object recognition. In *Experimental Robotics* (pp. 387-402). Springer International Publishing.

[14] Koppula, H. S., Gupta, R., & Saxena, A. (2013). Learning human activities and object affordances from RGB-D videos. *The International Journal of Robotics Research*, *32*(8), 951-970.

[15] Koppula, H. S., & Saxena, A. (2013, June). Learning Spatio-Temporal Structure from RGB-D Videos for Human Activity Detection and Anticipation. In *ICML (3)* (pp. 792-800).

[16] Koppula, H. S., & Saxena, A. (2014, September). Physically grounded spatio-temporal object affordances. In *European Conference on Computer Vision* (pp. 831-847). Springer International Publishing.

[17] Koppula, H. S., & Saxena, A. (2016). Anticipating human activities using object affordances for reactive robotic response. *IEEE transactions on pattern analysis and machine intelligence*, *38*(1), 14-29.

[18] Wang, K., Wang, X., Lin, L., Wang, M., & Zuo, W. (2014, November). 3D human activity recognition with reconfigurable convolutional neural networks. In *Proceedings of the 22nd ACM international conference on Multimedia* (pp. 97-106). ACM.

[19] Jiang, Y., & Saxena, A. (2013, June). Infinite Latent Conditional Random Fields for Modeling Environments through Humans. In *Robotics: Science and Systems* (pp. 1-8).

[20] Wu, C., Zhang, J., Savarese, S., & Saxena, A. (2015). Watch-n-patch: Unsupervised understanding of actions and relations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 4362-4370).

[21] Han, F., Reily, B., Hoff, W., & Zhang, H. (2017). Space-time representation of people based on 3D skeletal data: A review. *Computer Vision and Image Understanding*.

[22] Amor, B. B., Su, J., & Srivastava, A. (2016). Action recognition using rate-invariant analysis of skeletal shape trajectories. *IEEE transactions on pattern analysis and machine intelligence*, *38*(1), 1-13.

[23] Jain, A., Zamir, A. R., Savarese, S., & Saxena, A. (2016). Structural-RNN: Deep learning on spatio-temporal graphs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 5308-5317).

[24] Lin, L., Wang, K., Zuo, W., Wang, M., Luo, J., & Zhang, L. (2016). A deep structured model with radius–margin bound for 3D human activity recognition. *International Journal of Computer Vision*, *118*(2), 256-273.

[25] Tayyub, J., Tavanai, A., Gatsoulis, Y., Cohn, A. G., & Hogg, D. C. (2014, November). Qualitative and quantitative spatio-temporal relations in daily living activity recognition. In *Asian Conference on Computer Vision* (pp. 115-130). Springer International Publishing.

[26] Ramirez-Amaro, K., Beetz, M., & Cheng, G. (2015). Transferring skills to humanoid robots by extracting semantic representations from observations of human activities. *Artificial Intelligence*.

[27] Bütepage, J., Black, M., Kragic, D., & Kjellström, H. (2017). Deep representation learning for human motion prediction and classification. *arXiv:1702.07486*.

[28] Jain, A. (2016). *Learning From Natural Human Interactions For Assistive Robots* (Doctoral dissertation, Cornell University).

[29] Park, C. W., Choi, J., & Lee, S. Analysis on Recurrent Neural Network using Human Action Recognition Problem. *neural networks*, *6*(7), 8.

[30] Huang, Z., Wan, C., Probst, T., & Van Gool, L. (2016). Deep Learning on Lie Groups for Skeleton-based Action Recognition. *arXiv:1612.05877*.

[31] Papadopoulos, G. T., Axenopoulos, A., & Daras, P. (2014, January). Real-time skeleton-tracking-based human action recognition using Kinect data. In *International Conference on Multimedia Modeling* (pp. 473-483). Springer International Publishing.

[32] Parisi, G. I., Weber, C., & Wermter, S. (2015). Self-organizing neural integration of pose-motion features for human action recognition. *Frontiers in neurorobotics*, *9*, 3.