Classification of Composite Actions involving Interaction with Objects

Rakesh Gupta and **Trupti Devdas Nayak**
Honda Research Institute USA, Inc.
800 California Street, Suite 300
Mountain View, CA 94041
rgupta@hra.com, td23@cse.buffalo.edu

Abstract

We have developed a framework to recognize composite actions by training from simple interactions with objects. We start with object tracking data for the hand object interaction. We collect training data for simple actions such as pull, push, lift, drop and put. The training data consists of object translations, scale as well as object hand connectivity information.

We apply transformations to merge these simple actions into composite actions such as put-push and push-lift. These composite actions are used to build a classifier using Singular Value Decomposition (SVD). We collect test data for these composite actions and use Euclidean distance for matching. Recognition performance is significantly improved using time warping techniques such as Correlation Optimized Warping (COW).

Introduction

In the past, NLP research has focussed mainly on nouns. Other than Levin (1993), verbs were largely ignored due to their inherent complexity and ambiguity, despite being the binding agents in a sentence. Recent projects like VerbNet (Kipper, Dang, & Palmer 2000) and VerbOcean (Chklovski & Pantel 2004) have successfully shifted linguistic focus to verbs. These projects have collected an impressive range of information about verbs, verb senses and relationships with other verbs. Our aim is to model these motion verbs and train a robot to recognize actions represented by these verbs.

The OpenMind Indoor Common Sense (OMICS) database (Gupta & Kochenderfer 2004) contains information about actions which can be performed on objects or actions that are used for interacting with objects. Each object is associated with this list of actions. For example: Actions which can be performed on a *cup* are *pick up, lift, throw, break, give*. Some of these actions such as *give* are composed of simpler pick and release actions.

Our interest is in recognition of composite interactions with objects in the near field (like on a table). These actions involve hand motions where the hand is in contact or in the proximity of the object. Our research focuses on developing an action model to classify such observed actions. We have developed a framework to recognize composite actions by obtaining the training data from simple interactions. A composite action consists of two simple actions, performed one after another. By analysing action data in OMICS for common household objects we selected five *simple actions*. These were Push, Pull, Lift, Drop and Put. A *composite action* is a combination of one or more simple actions (e.g. pull-lift).

Modeling composite actions and learning to recognize them presents a very challenging task. It would be very cumbersome to collect training data for each composite action. We have developed a technique to composite training data from simple actions by concatenating these sequences by application of transformations. In addition, we have improved recognition performance by application of time warping techniques such as Correlation Optimized Warping (COW) and Dynamic Time Warping (DTW).

A robot can use our framework to learn models for hand actions represented by these verbs. Using action data from the OMICS database, the robot can then determine the set of objects on which these actions can be performed. For example, once a robot learns the model for *pick up* and *lift*, it can use OMICS action data to associate these actions with objects like *cup, mug, toy* etc.

To our knowledge, the closest work in action recognition for near field interaction with objects is from Jeffrey Siskind (Fern, Givan, & Siskind 2002). He hardwired Newton's second law and Coulomb model of friction to interpret short video sequences in terms of support, contact and attachment relations between objects to recognize actions such as pick up and put down. Our technique is to not hard-code, but learn distinct contact and object motion trajectories.

In computer vision, Cen et. al.(Rao, Yilmaz, & Shah 2002) recognized complex actions (e.g. put an object in the cabinet) by representing speed and direction of the hand trajectory. However such techniques fail to recognize basic actions such as drop because there is very little hand motion and the object hand interaction is not modeled.

Intel Research and the University of Washington have been involved in research for learning and recognizing *Activities of Daily Living* (Philipose *et al.* 2004). Their research aims at helping care-givers for the elderly by providing and analyzing information about the elderly person's daily activities. The objects of interest are tagged with RFID tags and the users wear a glove fitted with an RFID tag reader. Their system PROACT represents activities as a probabilistic sequence of objects used. It also implements data mining techniques to learn and classify the action models and activities performed by the users based on these RFID readings.

This paper is organized as follows. In the next section we give an overview of the DTW and COW time alignment preprocessing techniques. We then describe our experimental procedure, and modeling and classification of complex action. We then describe Results and Conclusions.

Time Alignment Preprocessing Techniques

Dynamic Time Warping and Correlation Optimized Warping are both presented in literature as methods that can correct sample vectors towards a reference to match the relevant parts of the signals (Pravoda, Walczak, & Massart 2002). This preprocessing of the signal effectively reduces the rank, leading to more robust models. We give an overview of each of these techniques in this section.

Dynamic Time Warping For composite actions, the training instances were obtained by concatenating instances of simple actions. One potential disadvantage of this is that the concatenated instance does not accurately represent an actual composite action instance. The time axis for the two simple actions might not align, since the two simple actions were performed independently of each other. And considering the non-motion data at the beginning and end of each simple action instance, concatenating them results in unnecessary, useless non-motion data in the composite action feature vector. Extrapolating these feature vectors will further result in a very distorted instance, which cannot be used for training.

Dynamic Time Warping is a method that warps two sequences non-linearly, in order to find an optimal match between the two. Consider two time series P of length n, and Q of length m.

$$P = p_1, p_2, ...p_n$$
$$Q = q_1, q_2, ...q_m$$

In order to align two sequences using DTW, an n-by-m matrix is constructed. The (ith, jth) element of the matrix is typically the Euclidean distance $d(p_i, q_j)$, between the two points p_i and q_j. The alignment between the points p_i and q_j is mapped to the (ith, jth) elements of the matrix. The alignment is represented by the warping path, say W, which defines the mapping between the two sequences P and Q.

In the case of composite actions, a reference sequence is computed from the set of training instances. DTW is used to warp each sequence (of an instance) with the reference sequence.

Correlation Optimized Warping COW also provides a method for warping two sequences non-linearly, but it divides the sequence into segments instead of using all time points. The segments can be equal lengths or of user-defined lengths. The formula computes the sequence that has the highest sum of correlation measures after warping. A slack parameter determines the flexibility of warped segments, calculated by linear interpolation. The aim is to align a sample data vector with a reference vector by allowing changes in the segment lengths of the sample vector.

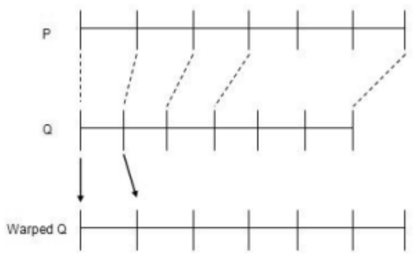

Figure 1: Correlation Optimized Warping

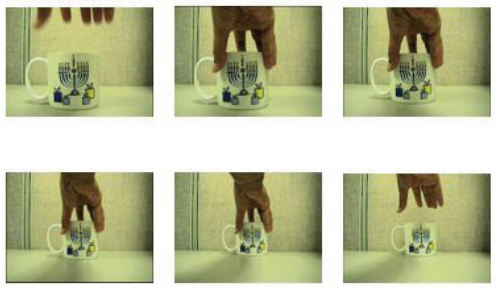

Figure 2: Frame sequence for simple action Push

A slack parameter controls the maximum length increase or decrease in a sample segment. If the number of time points in a sample and reference differs, the sample is linearly interpolated to create a segment of equal length. For reference vector r and sample vector s, the measure of correlation is given by:

$$p(n) = \frac{cov[r(n), s\{n\}]}{\sqrt{var}[r(n)]var(s\{n\})} \quad (1)$$

In COW, the segment lengths on the sample are warped to optimize the correlation between sample and reference. The optimization requires two parameters, the number of segment borders and the length of the slack area. We fixed the initial and final boundaries to ensure that the first and the last points in the sample and reference match.

Experimental Procedure

Data Collection

Simple everyday objects like a toy car, a cube, a mug and a toy robot were used for performing hand actions.

Single camera is used to detect movement of the object and the hand at 5-8 frames/second. Skin detection techniques enable detection of whether the hand is in the frame and if it is in touch with the object. This allows differentiation between subtly similar actions like drop and put. In the

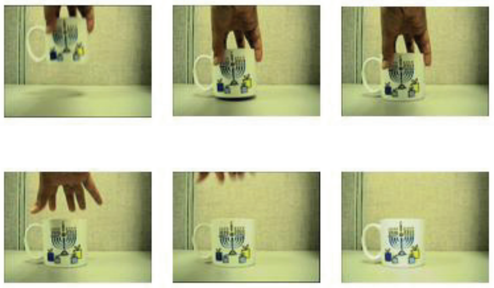
Figure 3: Frame sequence for simple action Put

case of drop, the robot's vision determines that the hand was not in contact with the object during the course of performing the 'drop', whereas in case of put-down, the hand was in touch with the object all the time.

Training data was collected by performing hand motions with small objects in front of the camera as shown in Fig 2 and 3. The vision system is based on a system developed by Lowe(2004). The object is placed in front of the camera. The vision software learns the object's shape and texture and recognizes it when it is moved. The movement of the object is continuously monitored by calculating the transformations the object is going through. Three transformations computed are Linear, Rotation and Scale.

Simple and Composite Actions

For each simple action, 20 instances each were recorded using the vision software. These simple actions are pull, push, drop, lift and put. Here we define pull to be motion of the object parallel to camera axis, away from the camera. Push is defined as motion of the object parallel to camera axis, towards the camera.

For composite actions, training instances were obtained by concatenating feature vectors of simple instances. For testing purposes single sequences with two simple actions were captured as shown in Fig 4. Five test samples were captured for each of these composite actions:

1. Pull Lift
2. Put Pull
3. Lift Put
4. Put Push
5. Pull Lift

Each action is represented as a feature vector which is obtained by concatenating the transformation values computed from the transformation matrix, namely translation and Scale values. Further, to incorporate information about skin and object intersection data, the skin detection flags are also concatenated with the action feature vector. We do not use rotation because none of our simple actions involve change of angles.

The number of frames per second varies from 5-8. The time required to complete every action is different from

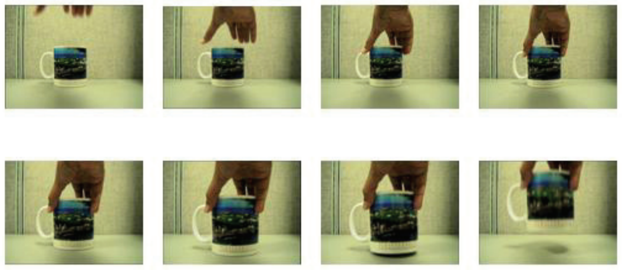
Figure 4: Frame sequence for composite action of Pull and Lift

other actions. Some actions like *Drop* take less time to perform and hence lesser number of frames, when compared to *Put down*. After concatenating the 4 parameters obtained for each action, the length of each instance for every action differs. It is therefore necessary to add non-motion data at the end to make the length of all the vectors equal.

Modeling and Classifying composite Actions

Each simple motion class has 20 instances, all of which are used for training. These simple actions were concatenated to form complex action training instances. Five test instances for each composite action were recorded.

Each action is represented as a feature vector, which is obtained by concatenating 4 parameters into one single vector. These 4 parameters are X and Y translation, scale obtained by decomposing the transformation matrix, and hand object connectivity flag. Non-motion data is added at the end to make the length of vector equal. Composite actions need preprocessing using Correlation Optimized Warping (COW) to warp the feature vector before matching. In COW, we divide the feature vector into segments and find a sequence with the highest sum of correlation measures after warping. We used a multiple of 4 sections and slack between 1 and 20.

Singular Value Decomposition (SVD) is used to identify and decompose an action into its principal components. Before SVD we normalize each transformation on the feature vector separately to have zero mean and unit variance. We used:
$A = USV^T$

Here matrix A is composed of all feature vectors of the training actions where each column corresponds to a training instance. The S and U matrices are computed. The first n diagonal elements of U are used for computing the eigenvalues and eigenvectors. For each test instance, the eigenvalues and eigenvectors are computed. Using the values of the U matrix obtained during training, a distance vector of the test instance and the n columns is calculated. The distance computed is the Euclidean distance between the two vectors. The minimum distance in the distance vector classifies the test instance as belonging to the corresponding class of

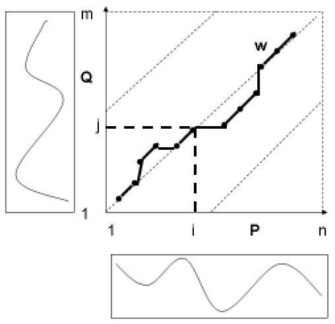

Figure 5: DTW and an example warping path W

Composite Action	Recognition without COW/DTW	Recognition with DTW processing	Recognition with COW processing
Lift Put	20%	20%	40%
Pull Lift	0%	0%	80%
Push Lift	0%	0%	20%
Put Pull	0%	0%	0%
Put Push	40%	40%	40%
Overall	12%	12%	36%

Table 1: Recognition of Composite Actions

actions.

Results

As shown in Table 1 without any warping the recognition rate is 12% which is similar to chance performance since we have a 10 class classification task. COW results were found to be significantly better than those with Dynamic Time Warping (DTW). With DTW, the recognition rate remains unchanged. However with COW, the recognition rate jumps to 36%.

Almost all mismatches were misclassifying the composite action as a simple actions so we need to work on a overall simple vs. composite action detector to improve performance.

Conclusions and Future work

We have developed a framework to classify composite actions involving interaction with objects. We concatenate simple action feature vectors to obtain training data for scalability. Given a set of simple actions, there can be many sequences of actions performed in different combinations. We can use the feature vectors of the simple actions to provide training data. Preprocessing warping of the data with Correlation Optimized Warping (COW) improved the recognition three fold from 12% to 36%. With preprocessing warping of the data with Dynamic Time Warping (DTW) the recognition rate was unchanged at 12%.

Our current computer vision software identifies an object in a given field of vision and tracks the object motion. In future object recognition can be used to identify the object. If the object is recognized as a *cup*, then the robot can use this information in conjunction with the action data from OMICS and know that the actions *pick up, lift, throw, break* etc can be performed on a *cup*. In future, we would also like to work towards extending this classification system to arbitrary sequences of actions.

Acknowledgments

This work was done while Trupti Devdas Nayak was a summer intern at Honda Research Institute USA, Inc. The vision system was implemented by Bernd Heisele. Thanks are also due to anonymous reviewers and the users of the OpenMind Indoor Common Sense website for their data and feedback.

References

Chklovski, T., and Pantel, P. 2004. Verbocean: Mining the web for fine-grained semantic verb relations. In *In Proceedings of 2004 Conference on Empirical Methods in Natural Language Processing (EMNLP - 2004)*. Barcelona, Spain.

Fern, A.; Givan, R.; and Siskind, J. 2002. Specific-to-general learning for temporal events with application to learning event definitions from video. *Journal of Artificial Intelligence Research* 17:379–449.

Gupta, R., and Kochenderfer, M. 2004. Common sense data acquisition for indoor mobile robots. In *Nineteenth National Conference on Artificial Intelligence (AAAI)*, 605–610.

Kipper, K.; Dang, H. T.; and Palmer, M. 2000. Class-based construction of a verb lexicon. In *Seventeenth National Conference on Artificial Intelligence (AAAI)*, 691–696.

Levin, B. 1993. *English Verb Classes and Alternations: A Preliminary Investigation*. Chicago: University of Chicago Press.

Lowe, D. G. 2004. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision* 60(2):91–110.

Philipose, M.; Fishkin, K. P.; Perkowitz, M.; Patterson, D. J.; Fox, D.; Kautz, H.; and Hhnel, D. 2004. Inferring activities from interactions with objects. *IEEE Pervasive Computing* 50–57.

Pravoda, V.; Walczak, B.; and Massart, D. L. 2002. A comparison of two algorithms for warping of analytical signals. *Anal Chim Acta* 456:77–92.

Rao, C.; Yilmaz, A.; and Shah, M. 2002. View-invariant representation and recognition of actions. *International Journal of Computer Vision* 50(2):203–226.