High Level Activity Recognition using Low Resolution Wearable Vision

Sudeep Sundaram, Walterio W. Mayol Cuevas
Department of Computer Science
University of Bristol, UK
sundaram@cs.bris.ac.uk, wmayol@cs.bris.ac.uk

Abstract

This paper presents a system aimed to serve as the enabling platform for a wearable assistant. The method observes manipulations from a wearable camera and classifies activities from roughly stabilized low resolution images (160x120 pixels) with the help of a 3-level Dynamic Bayesian Network and adapted temporal templates. Our motivation is to explore robust but computationally inexpensive visual methods to perform as much activity inference as possible without resorting to more complex object or hand detectors. The description of the method and results obtained are presented, as well as the motivation for further work in the area of wearable visual sensing.

1. Introduction

Activity recognition and event classification are of prime relevance to any intelligent system designed to assist on the move. There have been several systems aimed at the capturing of signals from a wearable computer with the aim of establishing a relationship between what is being perceived *now* and what *should* be happening. Assisting people is indeed one of the main championed potentials of wearable computing and therefore of significant research interest for a range of applications.

Most of the successful systems for wearable activity recognition to date, have relied on the instrumentation of limbs or the immediate environment (e.g. [15],[21], [8], [16]). In these cases the use of "low-level" sensors such as flexion sensors, accelerometers or passive microphones offer a degree of robustness that is important when performing complex manual tasks, and which other more advanced sensors such as cameras have not been able to fully deliver. It is interesting however to note that some of the very first approaches related to activity recognition from a wearable were based on visual sensing. In [5] and [1], for example, low-resolution vision is used to recognize previous locations which have been hand-labeled by the user using rough positioning. And in [12] an active wearable camera observes and classifies basic hand manipulations from skin images. In [20] a different approach altogether is used in that a simulated VR environment serves as the testbed for the recognition of activity.

These latter systems show how using reduced visual processing it is possible to achieve a degree of robustness to tolerate some of the challenges of wearable vision. Having a system that uses visual signals remains highly appealing as cameras are small, affordable and powerful sensors, able to detect an array of features from wearable environments that range from untagged object identities, wearer's manipulation events and social interactions.

In this paper, we present a framework for higher-level activity recognition that processes very low resolution motion images (160x120 pixels) to classify user manipulations. For this work, we base our test environment on supervised learning of the user's behavior from video sequences. The system observes interaction between the user's hands and various objects, in various locations of the environment from a wide angle shoulder-worn camera. The objects being interacted with are indirectly deduced on the fly from the manipulation motions. Using this low-level visual information user activity is classified as one from a set of previously learned classes.

The system is broadly divided into two sections - (1) the vision section, which recognizes manipulation motion, and (2) a Dynamic Bayesian Network that infers locations, objects and activities from a sequence of actions. An adapted version of temporal templates [4] to the wearable non-static camera domain is used to classify manipulation motion, while the inference model consists of a 3-level Dynamic Bayesian Network (DBN), which treats each activity as a hierarchy, and inference is performed at all three levels.

We present a detailed analysis of the vision algorithms stage, as well as our initial results on the higher-level activity recognition.

2. Related Work

Understanding hand motion has been an area that has received significant attention, primarily in Human-Computer

978-1-4244-3993-5/09/$25.00 ©2009 IEEE

Interaction. Vision-based gesture recognition systems can be used for various applications, such as visual control of a mouse [2], sign language recognition [6][19] and control of music applications [10] to name a few. Methods have focused on minimal restrictions on the user's hands, and have proved to some extent that the use of a single visual sensor is sufficient to recognize complex articulated motion.

Action recognition, dealing with human motion as a whole, has been also an active area of research in applications such as behavior analysis and surveillance. In recent years, spatio-temporal modeling of human actions has been a common approach for recognizing actions. Gorelick *et al*. [3], Ke *et al*. [9] and Yilmaz and Shah [23] model actions as space-time shapes in 3D, which can be matched with learned data. Similar work, but in 2D, is to produce temporal templates [4] to represent action classes. All of the spatio-temporal action modeling systems use a static camera strategically placed to observe the user's actions. Our work differs on this front in that we use a body-worn camera to observe actions, which poses a host of new challenges.

In terms of frameworks for modeling a succession of events, the most common approaches consider Hidden Markov Models [17] and Dynamic Bayesian Networks (DBN) [13]. And in terms of the kind of sensors, many wearable activity recognition systems, such as Patterson *et al*. [15], Ward [21] and Huynh [8], make use of non-visual wearable sensors to monitor user activity, while Clarkson *et al*. [5] use a combination of visual and non-visual sensors. RFID has also been used to tag objects in order to observe the user's interaction with them in work by Wu *et al*. [22] and Park *et al*. [14].

Our work on activity recognition is close to that carried out by Liao *et al*. in finding transportation routines [11], where a single low level sensor (GPS) is used to infer transport routines using a DBN, and to that of Sridhar *et al*. [18] use a visual sensor to infer object categories by modeling a spatio-temporal activity graph.

3. Manipulation Recognition

Related to our efforts, there has been significant attempts to develop systems that can detect and recognize hand poses or activity. for a recent overview of hand detection and recognition from visual methods see [12]. Detecting hands is important as a cue to what the person is doing *and* what s/he plans to do. However, hand detection and recognition is challenging because the higher number of degrees of freedom involved. From the point of view of a conventional camera this is also challenging as frame rates are low compared to other sensing strategies such as accelerometers (25-30 Hz vs 100s Hz). In this work we avoid the direct, constant, detection of hand pose or gestures and concentrate instead on developing methods that indirectly tell us what the person is doing at a rougher level of description. This rough hand activity is what we will refer to in this paper as manipulation and that serves as the main input to our recognition algorithm.

Manipulation recognition is performed on low resolution images using temporal templates - first proposed by Bobick and Davis [4]. Temporal templates (or motion history images) capture any motion detected in the video, with weights inversely proportional to the temporal distance from the frame in which the motion was detected, to the current frame. This discussed further in Section 3.2.

3.1. Image Registration

Given that the system's setup consists of a body-worn camera, actions performed by the user will result in temporal templates that consist not only of his/her hand actions, but also the relative motion of the environment with respect to the camera. This results in a noisy template with minimal information about hand actions.

In order to address this problem, the camera motion needs to be compensated in each frame before computing temporal templates. We have achieved reasonable success by using a dictionary of affine transformations as an approximation to compensate camera motion between frames. We assume that, relative to the background, the hands take up a less significant portion of the frame.

Let I_k be the current frame being processed, and I_{k+1} be the next incoming frame. Let $\phi(I, \mu)$ denote the result obtained when an image I is affine transformed by parameter vector μ. We approximate the compensating of camera motion by finding the affine transformation parameter vector μ_k such that the difference between $\phi(I_k, \mu_k)$, and I_{k+1} is minimized.

μ_k can be estimated as

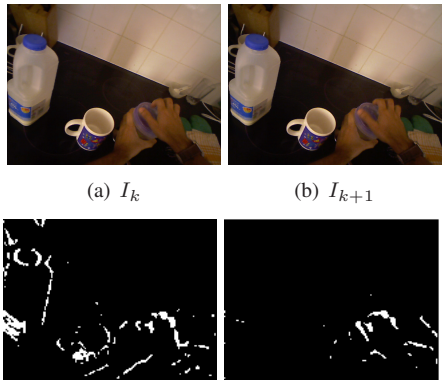
$$\mu_k = argmin_{\mu}(I_{k+1} - \phi(I_k, \mu)) \tag{1}$$

The difference image D_k is given by

$$D_k = I_{k+1} - \phi(I_k, \mu_k) \tag{2}$$

Any remanent background noise is removed using a simple 2x2 averaging filter on D_k, after which the image is thresholded to obtain outlines of objects that have moved in the frame.

The composition of μ can be determined based on the expected camera motion in the environment. In our case, we include parameters related to scale, rotation and translation. The dictionary of μ is created by uniformly sampling parameters of scale, rotation and translation over a range of pre-defined values. The dictionary consists of variations of scale between 0.95 and 1.05, rotation between -2^o and 2^o, X-translation between -4 and 4 pixels, and Y-translation



(a) I_k (b) I_{k+1}

(c) $I_{k+1} - I_k$ (d) D_k

Figure 1. Approximate Image Registration

between -2 and 2 pixels. This results in a total of approximately 1350 affine transformations, which the system processes at about 20 frames per second.

3.2. Temporal Templates

Based on the assumption that the background takes up a significant portion of the frame, D_k now contains information of any motion (including hand motion) that has occurred in the workspace visible to the camera. This motion information is collated over a sequence of images, with the intensity of each varying as a function of time in order to obtain motion history. A pixel that has experienced motion n frames prior to the current frame, is assigned an intensity that is an n^{-1} factor of the intensity assigned to pixels that have experienced motion in the current frame.

As a measure to enhance the response of the hand detection, and differentiate it further from the background, we use the chroma red component of the image for computing the temporal templates.

It can be seen intuitively that the length L of the sequence of frames encoded in a temporal template is equal to the number of intensity quantization levels used to create the temporal template. If we consider the maximum image intensity as P_{max}, then the temporal template can be described as

$$\eta_k = \sum_{i=1}^{L} (\frac{i}{L} P_{max}) D_{k-i} \tag{3}$$

Figure 2 shows some samples captured at equal intervals from a video sequence of the use removing a bottle cap, and the corresponding temporal templates. This clearly demonstrates how the action's temporal information is captured in the template.

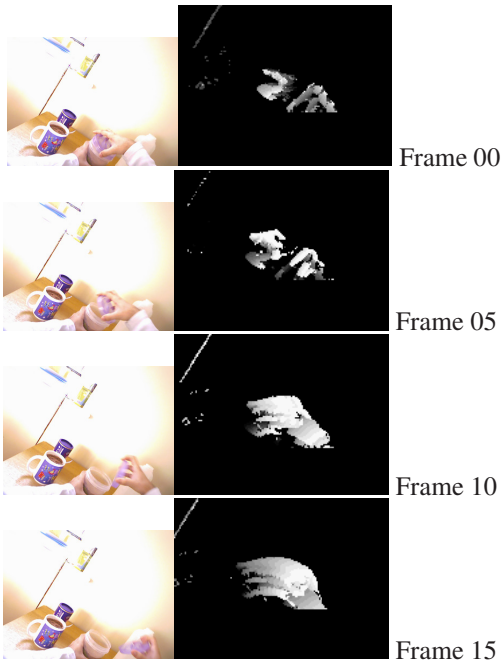


Figure 2. Temporal Templates corresponding to sampled frames

3.3. Temporal Template Matching

Recognition of manipulations was carried out by matching temporal templates to learned priors. In the training stage, samples that contributed less were filtered out using RANSAC, and mean templates were computed for each action class. Simple image matching techniques were employed to match temporal templates obtained from the sequences to the means. Experiments were carried out on three image matching techniques -

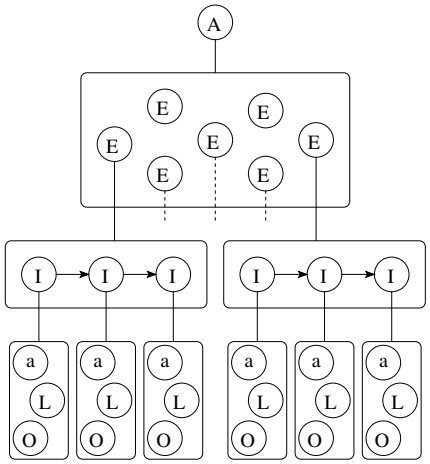
- Using sums of row and column intensities as features
- Hu's set of scale, translation and rotation invariant moments [7]
- Normalized cross-correlation

Based on tests carried out using 141 action templates, normalized cross-correlation was found to be the most suitable technique. Experimental results for the same are presented and discussed in Section 6.1

4. Activity Recognition

4.1. Activity Hierarchy

In order to complete an *activity*, the user performs a set of *manipulations* in a conducive environment, *objects* within which are used to accomplish tasks. The environment may be a complex one, where the objects are distributed in various places, which gives rise to the possibility of the user



a:Action; O:Object; L:Location; I:Interaction E:Event; A:Activity

Figure 3. Activity defined as a bag of events. Events are sequences of interactions, which in turn are determined by the current manipulation, object and location

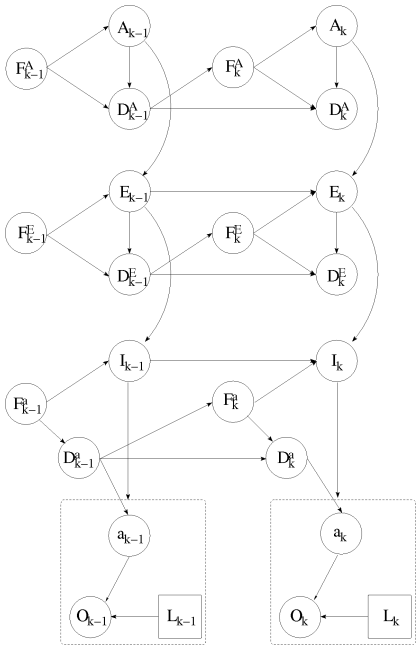
having to move to different *locations* within the environment to complete the task on hand.

Based on the above scenario, we can now define an *interaction* as the manipulation performed by the user on an object within a definite location in the given environment. Further, we define an *event* as a fixed sequence of interactions. For example, the event of pouring water from a jug into a cup effectively involves two interactions - one to pour from the jug and the other to place the jug back in its original position.

Putting a set of events together, we now define an *activity* as a collection of events, with or without the constraint that the events are to take place in a defined sequence. For example, while setting up a dining table, it does not matter whether we first place the spoons or the forks. However, we need to make sure that the dishes are placed before serving food! This gives rise to the requirement of a model that provides a restricted measure of freedom to the order of events. The hierarchy of such a model is shown in Figure 3.

4.2. Hierarchical DBN for Inferring Activity

Figure 4 shows a three level Dynamic Bayesian Network used to recognize the user's activities. The highest level infers user activity based on past events, which in turn are recognized in the second level based on past interactions. Interactions are determined based on the current manipulation, and estimates of the object and location. Location is inferred by analyzing interactions of the user with doors, and sometimes other objects in the environment. The object being interacted with is inferred based on the current manipulation and the likelihood distribution of where the object is located on the map.



a_k Recognized manipulation at frame k
D_k^a Delay state for manipulation a_k
O_k Estimated object being interacted with at frame k
L_k Estimated location of user at frame k
F_k^a Action switch for at frame k
I_k Estimated interaction at frame k
E_k Estimated event at frame k
D_k^E Delay state for event E_k
F_k^E Event switch at frame k
A_k Estimated activity at frame k
D_k^A Delay state for activity A_k
F_k^A Activity switch at frame k

Figure 4. Two-frame slice of Activity Recognition Model

4.2.1 Inferring Objects

Based on sequences collected during the training stage, the system learns a likelihood distribution of objects with respect to locations. For instance, considering a map of an apartment, a kettle is more likely to be found in the kitchen, less likely to be found in the bedroom and very unlikely to be found in the bathroom. Also, the system learns a likelihood distribution of objects being used with respect to the user's manipulations. The action of unscrewing the lid of a bottle is likely to occur on a milk bottle and rather unlikely on a door. Recall that we do not detect objects directly, but through their interaction with the user and the associated manipulations.

We can thus define the object model $p(O_k|L_k, a_k)$, which computes the probability of an object being used, given the current location and hand action.

4.2.2 Inferring Location

In these experiments we assume that different locations on the map are connected by doors, thus enforcing the rule that a change in location happens only when a door is opened, and also that when a door is opened, the location of the user has changed. The manipulation recognition system learns temporal templates obtained on opening doors, and this facilitates availability of location information at the lowest level in the hierarchy.

Change in location is straightforward to deduce when the current location consists of only a single door, thus restricting the location that the user enters after opening the door. In some cases, however, a location on the map may contain more than one door. When the user opens a door from such a location, the new location cannot be determined right away, and the system switches to the 'lost' state.

Also, at any location and time in the sequence, the system might sometimes fail to recognize that a door has been opened, and has a chance of erroneously persisting with the previous location. Detection of the latter case is done by analyzing the set of most recent actions and the corresponding inferred objects. If manipulations are being recognized with high confidences, and there is very little likelihood that these manipulations are performed with objects in the current location, then the system is again said to be 'lost'.

In the 'lost' state, the system switches to a flat object likelihood distribution i.e. all objects are equally likely to be found in the current location. This enables the system to continue inferring activity without depending on knowledge of location. In the next l frames, the user's manipulations are used to determine the location by considering the conditional probabilities $p(l_i|a_i)$. Though conditional probabilities are considered for each manipulation, location inference is not carried out using a Bayesian approach, in order to avoid the use of an l^{th} order DBN. Instead, we determine the location merely as the one that produces the highest product of probabilities $p(l_i|a_i)$ for i between $l-k$ and k.

4.2.3 Delay States and Switches

The use of temporal templates for manipulation recognition gives room for the possibility of obtaining high confidence values for a manipulation within a temporal window of n frames on both sides of the current frame, since the generated template does not change much in a few time steps. If the recognized outputs from more than one frame within this window are input to the hierarchical model, they will be seen as two separate actions performed by the user. This brings about the need for the *delay state* D_k^a. The model learns the minimum time (number of frames) $\tau(a)$ taken up by the user to complete a given manipulation. When a manipulation is recognized with high confidence, the delay state acts as a count-down timer, and is set to $\tau(a)$. For the next $\tau(a)$ frames, the delay state blocks any update in the model. This is facilitated by the presence of the switch F_k^a, which is reset to 0 when the delay state is active, and set to 1 when the count reaches 0.

Switches and delay states for the lowest level are modeled in Equations 4 and 5. Delay states and switches are also present for events and activities at the middle and top levels of the model. At the middle level, delay state D_k^E, is set to the number of component interactions in the candidate event when the event starts. At the top level, delay state D_k^A is set to the number of component events in the candidate activity once the activity starts. Models for delay states and switches corresponding to events and activities can be derived in a similar manner to Equations 4 and 5.

4.2.4 Modeling Interactions and Events

As mentioned earlier, interactions are built using three basic elements - manipulation, object and location. The confidence of a recognized interaction thus depends on the confidence of these components. An interaction can be recognized only when the action switch F_k^a is active, else it remains unrecognized. The interaction model has been defined in Equation 6.

Since an event is defined as a fixed sequence of interactions, the model consists of a window whose size is equal to the number of interactions in the sequence. The likelihood of the event can be computed from the likelihoods of the component interactions within the window. If l_{E_k} denotes the length of the interaction sequence for E_k, we can describe it as $p(E_k|I_{k-l_{E_k}:k})$.

4.2.5 Modeling Activities

Keeping in mind that an activity has been defined as a bag of events that may or may not be performed in the same order, training the activity recognition level in the DBN becomes a challenging task. If one analyzes the available data, we have on hand a finite number of ways in which the activity can be accomplished - each essentially a unique order of component events. From this, we can compute the probability distribution of the event that occurs first. Similarly, transition probabilities between component events can be computed statistically. These distributions represent the π and A models of an HMM, which is simpler to train and use in this scenario. The distribution B is computed directly for each event using its component interactions, as described in Section 4.2.4. We can thus use HMMs to model the highest level of the DBN, in order to recognize activities.

The event delay states and switches play a vital role in the recognition of activities, since these define when an event transition can take place. An activity is said to be

$$P(F_k^a | D_{k-1}^a = d) = \delta(d, 0) \tag{4}$$

$$P(D_k^a = d' | D_{k-1}^a = d, a_k = \alpha) = \begin{cases} \delta(d', d-1) & d > 0, \quad F_{k-1}^a = 0 \\ undefined & d = 0, \quad F_{k-1}^a = 0 \\ \delta(d', \tau(\alpha)) & d = any, \quad F_{k-1}^a = 1 \end{cases} \tag{5}$$

$$P(I_k | F_{k-1}^a) = \begin{cases} \delta(I_{k-1}, I_k) & f = 0 \text{ (remain in same state)} \\ p(I_k | L_k, a_k, O_k) & f = 1 \text{ (move to new state)} \end{cases} \tag{6}$$

(a) User (b) Map

Figure 5. Experimental Setup

complete when the activity switch F_k^A is set. Event transitions are modeled using Equation 7.

If we consider the number of events in the 'bag' corresponding to activity A_i as l_{A_i}, then Equation 8 gives the likelihood that A_i has been completed in frame k.

5. Experimental Setup

Daily morning routine sequences were captured from a single user over a period of two weeks. Many of the activities were centered around kitchen routines. This section describes the setup and experiments in detail.

5.1. User

The hardware consists of a USB camera, connected to a notebook, which is placed in a backpack. The camera is strapped on to the user's shoulder, appropriately placed to capture motion of both hands. The field of view of the camera is a vital feature that is to be taken into account. A narrow angled lens will not cover the space of motion of the hand, unless restrictions are placed on the user. The camera used for our experiments has an 80° diagonal FOV, which although not entirely satisfactory is a good compromise for the current activities considered. A snapshot of the setup is shown in Figure 5.

5.2. Map

Experiments were carried out in an apartment, whose layout is shown in Figure 5. The grey sectors indicate swinging doors. Note that the *Out*, *Bedroom*, *Bathroom* and *Kitchen* locations each connect to a single door, so there is no ambiguity with respect to updating the user's location when a door is opened from one of these locations. However, the *Corridor* connects to 4 doors, and once the user opens a door from the corridor, the system depends on the DBN to resolve the location based on the user's manipulations.

5.3. Manipulations, Objects and Activities

The system was trained to recognize a total of 12 primitive manipulations:

0 Screw/Unscrew Lid	1 Remove Lid	2 Eat/Drink
3 Open Door (Enter)	4 Open Door (Exit)	5 Stir
6 Wipe (circular)	7 Wipe (to and fro)	8 Scoop
9 Drop 'A' into 'B'	10 Pour	11 Place Back

Objects that the user interacts with in the environment include:

Cup	Hot Chocolate Bottle	Kettle
Tea Bottle	Coffee Bottle	Sugar Bottle
Milk Bottle	Spoon	Door

Activities captured during morning routines include:

Making a cup of coffee
Making a cup of tea
Making a cup of hot chocolate
Wash a dish
Wipe kitchen platform

6. Results

Work is currently in progress to verify the performance of the activity recognition system for very long periods of operation, though initial results on a reduced number of sequences are encouraging. In this section, we go over the available results at lowest and highest levels of the model.

6.1. Manipulation Recognition

As mentioned in Section 3.3, three image matching techniques were used to match temporal templates. Confusion matrices were generated using a subset of the listed actions on a limited dataset. Normalized cross-correlation proved to perform the best among the three methods with 60.99% accuracy, while the Image Moments and Intensity Vectors methods were respectively 42.33% and 54.62% accurate.

$$P(E_k = j|E_{k-1} = i, F_{k-1} = f) = \begin{cases} \delta(i,j) & f = 0 \text{ (remain in the same state)} \\ A(i,j) & f = 1 \text{ (move to state j)} \end{cases} \tag{7}$$

$$P(A_k = A_i) = \pi(E_{k-l_{A_i}})p(E_{k-l_{A_i}}) \sum_{j=k-l_{A_i}+1}^{k} A(j-1,j)p(E_j) \tag{8}$$

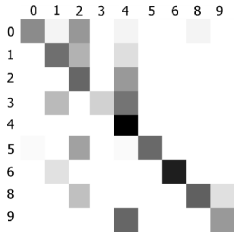


Figure 6. Confusion Matrix for matching a subset of manipulation classes using Normalized Cross-Correlation. Manipulation IDs correspond to those in Section 5.3

The confusion matrix for manipulation recognition using normalized cross-correlation is shown in Figure 6. This lack of accuracy at the lowest level is handled by higher levels in the activity recognition system.

6.2. Activity Recognition

Recognition confidences of all manipulation classes are fed into the lowest level of the model at each update. Events and activities are recognized using the procedure described in Section 4.2. To illustrate the process, we have considered two sequences, the first being of the user washing a dish, and the second of the making of a cup of hot chocolate. Figures 7 and 8 show the variation of recognition confidences for all activities plotted against the interaction number in the input sequences. Note that activities remain low on confidence until enough interactions have taken place for their respective completion. In 7, the confidences of 3 of the 5 activities remain low throughout the sequence since they all need more than 4 interactions to be completed. The horizontal line shows the confidence threshold for declaring a valid recognition. In 8 snapshots of the sequence at different frames are shown. Note that several of the activities tested are challenging to differentiate as they all are related to making drinks with similar objects and steps.

6.2.1 Sensitivity Tests

In this section, we present results of a simulation carried out to test the activity recognition's sensitivity to misclassified manipulations at the lowest level of the model.

Consider an activity A that consists of N_A manipulations. As part of the simulation, we start by providing correct inputs for all N_A manipulations, and start introducing errors step by step. In the i^{th} step, we introduce $i-1$ errors. Errors are introduced by randomly picking one of the component manipulations a_o and replacing it with another randomly picked manipulation a_r (a misclassification). We now have an updated sequence of manipulations, which we will call M, of length N_A. The model is then sequentially updated with each manipulation.

For each update j, we choose manipulation $M[j]$ as the mean and fit a normal probability distribution over the space of all manipulations, which are fed into the lowest level of the model. The standard deviation of the normal distribution is varied over 200 steps in order to carry out tests over a

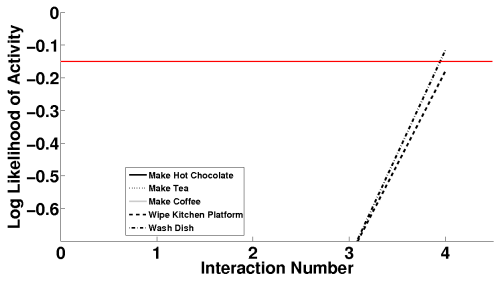


Figure 7. Activity Recognition Confidences against Interactions for a sequence of "Wash Dish". The red line indicates the recognition threshold.

(a) I-1 (b) I-4 (c) I-9 (d) I-11 (e) I-13

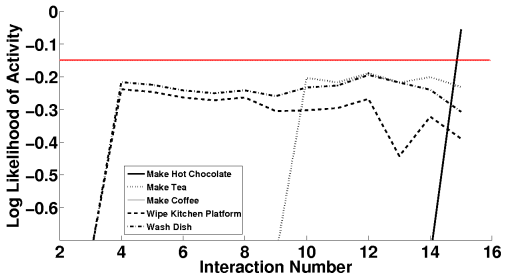


Figure 8. Activity Recognition Confidences against Interactions for a sequence of "Make Hot Chocolate". The red line indicates the recognition threshold.

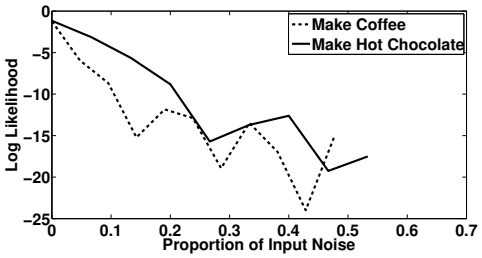


Figure 9. Noise Sensitivity of Activity Recognition Model for two activities ("Make Coffee" and "Make Hot Chocolate").

large range of values. Recognition rates are determined by finding the average likelihood over all standard deviations for a given percentage of input noise.

Figure 9 shows the sensitivity of the model in recognition of two chosen activities. It can be seen that the model performs relatively well for inputs that are 20% or less noisy. The simulations indicate that ss long as manipulations occur in the right order, misclassification of activities is unlikely.

7. Discussions

We have proposed a framework to recognize activities by observing actions and manipulations performed by the user with non-adorned hands and using a wearable camera. We have shown the inference of high level activity from roughly registered low resolution images. A Dynamic Bayesian Network has been proposed to infer location, objects, interactions, events and activities, and we have presented results from a simulation to show that this model is robust to reasonable levels of misclassification.

Further work includes collection of a large and more exhaustive set of daily routine sequences of day-to-day activities, and testing of the model with this data. The accuracy of temporal template matching can also be improved in order to ensure better performance.

References

[1] H. Aoki, B. Schiele, and A. Pentland. Realtime personal positioning system for wearable computers. In *Proc. International Symp. on Wearable Computing*, 1999.

[2] A. A. Argyros and M. I. A. Lourakis. Vision-based interpretation of hand gestures for remote control of a computer mouse. In *ECCV Workshop on HCI*, pages 40–51, 2006.

[3] M. Blank, L. Gorelick, E. Shechtman, M. Irani, and R. Basri. Actions as space-time shapes. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 2, pages 1395–1402 Vol. 2, Oct. 2005.

[4] A. Bobick and J. Davis. The recognition of human movement using temporal templates. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 23(3):257–267, Mar 2001.

[5] B. Clarkson, K. Mase, and A. Pentland. Recognizing user context via wearable sensors. In *Wearable Computers, 2000. The Fourth International Symposium on*, pages 69–75, 2000.

[6] K. G. Derpanis, R. P. Wildes, and J. K. Tsotsos. Hand gesture recognition within a linguistics-based framework. In *Computer Vision ECCV 2004*, pages 282–296, 2004.

[7] M.-K. Hu. Visual pattern recognition by moment invariants. *Information Theory, IRE Transactions on*, 8(2):179–187, February 1962.

[8] D. T. G. Huynh. *Human Activity Recognition with Wearable Sensors*. PhD thesis, TU Darmstadt, 2008.

[9] Y. Ke, R. Sukthankar, and M. Hebert. Spatio-temporal shape and flow correlation for action recognition. In *Computer Vision and Pattern Recognition 2007*, pages 1–8, June 2007.

[10] M. Kolsch and M. Turk. Fast 2d hand tracking with flocks of features and multi-cue integration. In *CVPRW '04: Proceedings of the 2004 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'04) Volume 10*, page 158, Washington, DC, USA, 2004. IEEE Computer Society.

[11] L. Liao, D. J. Patterson, D. Fox, and H. Kautz. Learning and inferring transportation routines. *Artificial Intelligence*, 171(5-6):311 – 331, 2007.

[12] W. Mayol and D. Murray. Wearable hand activity recognition for event summarization. In *Proc. IEEE Int. Symposium on Wearable Computers*, 2005.

[13] K. P. Murphy. Dynamic bayesian networks. In *Probabilistic Graphical Models*, 2002.

[14] S. Park and H. Kautz. Hierarchical recognition of activities of daily living using multi-scale, multi-perspective vision and rfid. In *Intelligent Environments, 2008 IET 4th International Conference on*, pages 1–4, July 2008.

[15] D. Patterson, D. Fox, H. Kautz, and M. Philipose. Fine-grained activity recognition by aggregating abstract object usage. In *Wearable Computers, 2005. Proceedings. Ninth IEEE International Symposium on*, pages 44–51, Oct. 2005.

[16] B. Paulson and T. Hammond. Office activity recognition using hand posture cues. In *HCI2008*, United Kingdom, 2008.

[17] L. Rabiner and B. Juang. An introduction to hidden markov models. *ASSP Magazine, IEEE [see also IEEE Signal Processing Magazine]*, 3(1):4–16, 1986.

[18] M. Sridhar, A. G. Cohn, and D. C. Hogg. Learning functional object-categories from a relational spatio-temporal representation. In M. Ghallab, C. D. Spyropoulos, N. Fakotakis, and N. M. Avouris, editors, *ECAI*, volume 178 of *Frontiers in Artificial Intelligence and Applications*, pages 606–610. IOS Press, 2008.

[19] T. Starner, A. Pentland, and J. Weaver. Real-time american sign language recognition using desk and wearable computer based video. *IEEE Trans. Pattern Anal. Mach. Intell.*, 20(12):1371–1375, 1998.

[20] D. Surie, F. Lagriffoul, T. Pederson, and D. Sjolie. Activity recognition based on intra and extra manipulation of everyday objects. In *4th International Symposium of Ubiquitous Computing Systems*, Tokyo, Japan, 2007.

[21] J. A. Ward. *Activity Monitoring: Continuous Recognition and Performance Evaluation*. PhD thesis, Swiss Federal Institute of Technology, 2006.

[22] J. Wu, A. Osuntogun, T. Choudhury, M. Philipose, and J. Rehg. A scalable approach to activity recognition based on object use. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8, Oct. 2007.

[23] A. Yilmaz and M. Shah. Actions sketch: a novel action representation. *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, 1:984–989 vol. 1, June 2005.