Learning People Movement Model from Multiple Cameras for Behaviour Recognition

Nam T. Nguyen[1], Svetha Venkatesh[1], Geoff A.W. West[1], and Hung H. Bui[2]

[1] Department of Computing, Curtin University of Technology
GPO Box U1987 Perth, 6845 Western Australia
{nguyentn,svetha,geoff}@cs.curtin.edu.au
[2] Artificial Intelligence Center, SRI International
333 Ravenswood Ave, Menlo Park, CA 94025, USA
bui@ai.sri.com

Abstract. In surveillance systems for monitoring people behaviours, it is important to build systems that can adapt to the signatures of people's tasks and movements in the environment. At the same time, it is important to cope with noisy observations produced by a set of cameras with possibly different characteristics. In previous work, we have implemented a distributed surveillance system designed for complex indoor environments [1]. The system uses the Abstract Hidden Markov mEmory Model (AHMEM) for modelling and specifying complex human behaviours that can take place in the environment. Given a sequence of observations from a set of cameras, the system employs approximate probabilistic inference to compute the likelihood of different possible behaviours in real-time. This paper describes the techniques that can be used to learn the different camera noise models and the human movement models to be used in this system. The system is able to monitor and classify people behaviours as data is being gathered, and we provide classification results showing the system is able to identify behaviours of people from their movement signatures.

1 Introduction

Monitoring people behaviours in large and complex environments using multiple cameras for automated surveillance is an important research area. Approaches to this problem usually divide the solution into two layers of processing: a low-level tracking component processes low-level visual data from the cameras and produces a stream of events which are then interpreted by a recognition module to produce high-level description of the people activities in the environment [2–5]. Oliver *et al* [2] propose a Layered Hidden Markov Model (LHMM), where the classification results of the lower layer are used as inputs to the higher layer. Ivanov and Bobick [3] proposed a two-stage strategy to recognise the interactions of humans and vehicles. At the lowest level, the system recognises simple events, which are used as inputs for a stochastic context-free grammar parsing mechanism to recognise multi-object interactions at the higher level.

It is well-known that current low-level tracking techniques are not robust, and their outputs are inherently noisy due to a variety of environmental and processing conditions. Thus, it is up to the "high-level" module to deal with the imperfect output pro-

A. Fred et al. (Eds.): SSPR&SPR 2004, LNCS 3138, pp. 315–324, 2004.

duced by low-level processing to produce robust and accurate descriptions of the observed activities. We thus argue that the high-level behaviour recognition module needs to be based on a framework that facilitates the modelling of and reasoning with uncertainty. Previously, we proposed the use of the Abstract Hidden Markov mEmory Model (AHMEM) for this purpose [1, 6]. In the AHMEM, behaviours are organized into a stochastic hierarchy. Each behaviour can be refined into a sequence of more simple behaviours at lower levels. In addition, the rules for refinement can be made non-deterministic or stochastic. The model is as expressive as other grammar-based models such as the Probabilistic Context Free Grammar (PCFG) [7], and can model state-dependent goal-directed behavours. At the same time, it supports online, and efficient probabilistic inference of high-level behaviours from low-level data. Furthermore, the hierarchical nature of the model makes it suitable for the natural hierarchy existing in spatial regions, making it scalable to larger and more complex environments.

The AHMEM framework also provides the flexibility for integrating with a noisy low-level tracking module via a HMM-like model at the bottom level of the behaviour hierarchy. This acts like an interface between the AHMEM and the low-level tracking module. This paper addresses the problem of learning the necessary parameters for building this interface between the high-level and low-level tracking module. We provide techniques for estimating the obsevation models of the cameras, and to estimate the movement models of people on one floor of a building. We then describe a complete distributed surveillance system combining a low-level tracking module with the AHMEM for behaviour recognition. We provide experimental results showing that the system is able to monitor and robustly classify complex human behaviours in an indoor environment.

The paper is organised as follows. An overview of the surveillance system is provided in Section 2. The techniques to learn observation models and movement models for the surveillance system are presented in Sections 3 and 4, respectively. The system implementation is described in Section 5. Finally, Section 6 presents the experimental results of the implemented system in a real office-like environment.

2 Overview of the Surveillance System

The surveillance system has two major components: the distributed low-level tracking module and the high-level behaviour recognition module. The distributed tracking module extracts people trajectories using multiple static cameras. The trajectories are inputs for the high-level behaviour recognition module. The implementation of the surveillance system is described in [1].

In the behaviour recognition module, the AHMEM and its parameters define a conditional distribution over the observation sequences given a behaviour: $\Pr(\tilde{o}|\pi^k)$. In recognising the behaviour of a person in the scene, we are given a sequence of observations from the low-level tracking module: $\tilde{o}_{t-1} = (o_1, \ldots, o_{t-1})$ up to the current time t, and need to compute the probability $\Pr(\pi_t^k|\tilde{o}_{t-1})$, where π_t^k represents the policy being executed at level k and time t. This provides the distribution of the possible behaviours that might be currently executed at level k in the hierarchy. The computation needs to be done at every time instance t when a new observation o_t arrives. The problem is termed *policy recognition* [8], and is equivalent to the on-line inference (filtering) problem in the AHMEM. An efficient approximate inference algorithm based on the Rao-Blackwellised Particle Filter (RBPF) [9] for computing the probabilities is given in [10, 8]

The necessary parameters for building the interface between the high-level and low-level tracking modules are the observation models of cameras and the movement models of people on one floor of a building [1]. In the following sections, we will describe the techniques to learn these parameters in detail.

3 Learning Observation Models

The observation model for a camera C is defined as $B = \Pr(o|s, C)$, where s is the state of a person and o is the observation. Usually, there are a large number of states in the field of view (FOV) of the camera C. Thus, we have difficulty in creating enough sample video sequences to learn B. Assume that the observation o is in one of the cells within the neighbourhood of the state s (including s), and $\Pr(o|s, C)$ is unchanged for all states s in the FOV of camera C, i.e. the statistics are spatially invariant. We can compress the observation model B to a compressed observation model B^c, which is a 3×3 matrix given as:

$$B^c = \begin{bmatrix} \Pr(o_{northwest}|C) & \Pr(o_{north}|C) & \Pr(o_{northeast}|C) \\ \Pr(o_{west}|C) & \Pr(o_{center}|C) & \Pr(o_{east}|C) \\ \Pr(o_{southwest}|C) & \Pr(o_{south}|C) & \Pr(o_{southeast}|C) \end{bmatrix}$$

where o_{north}, $o_{northeast}$, o_{east}, $o_{southeast}$, o_{south}, $o_{southwest}$, o_{west}, $o_{northwest}$ and o_{center} are nine possible observations of a true state s (see Fig. 1). Instead of learning the observation model B, we can learn the compressed observation model B^c from a set of sample video sequences.

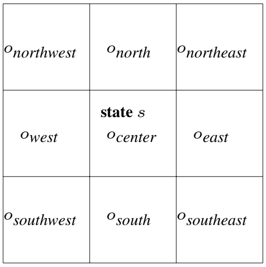


Fig. 1. The possible observations of a state s.

We learn the compressed observation model B^c for a camera C from sample video sequences, which are created by recording people in the environment from the views of all cameras. We run the tracking system to obtain the real world coordinates of people in the sample video sequences. Among these coordinates, we randomly choose N

coordinates $(x_1,y_1),\ldots,(x_N,y_N)$ that are originally generated from camera C. We consider these coordinates as the observations of the people. We then manually extract the corresponding person's true positions: $(x_1^t, y_1^t),\ldots,(x_N^t, y_N^t)$. These coordinates are mapped to the cells (states) in the environment, i. e. $(x_1, y_1),\ldots,(x_N, y_N)$ are mapped to $o_1,\ldots,o_N$ and $(x_1^t, y_1^t),\ldots,(x_N^t, y_N^t)$ are mapped to $s_1,\ldots,s_N$.

We estimate the compressed observation model B^c from the N observations $o_1, \ldots, o_N$ and the N corresponding states $s_1,\ldots,s_N$. Note that $B^c = \Pr(o|C)$, where $o \in \{o_{north}, o_{northeast}, o_{east}, o_{southeast}, o_{south}, o_{southwest}, o_{west}, o_{northwest}, o_{center}\}$ (see Fig. 1). To estimate $\Pr(o_{north}|C)$, we count the number of times that o_i is the northern neighbouring state of s_i from the N observations and the N corresponding states. $\Pr(o_{north}|C)$ then equals the frequency that o_i is the northern neighbouring state of s_i. The remaining probabilities of B^c are estimated in a similar manner.

4 Learning Movement Models

For a bottom level behaviour π in a region R, we need to learn the movement model $A = \Pr(s' \mid s, \pi)$, which is defined for all states s in R and for all neighbouring states s' of s [1].

4.1 Dealing with Large Transition Models

For the case in which region R is small, we can learn A from a number of training video sequences. However, when region R is large and has many states, the number of training video sequences required to learn A is large due to the size of the state space. Therefore, instead of learning the complete movement model A, which is a difficult task, we compress A to a compressed movement model A^c and learn A^c.

In large regions, we are only interested in behaviours representing the action of a person going to a destination such as going to a printer, going to a computer, and so on. Thus, we can assume that each behaviour defined in a large region has a destination. The compressed movement model A^c is defined as a 3×3 matrix specifying the probabilities that a person moves to the next state, assuming that the direction to a destination is the *up-front* vector. The compressed movement model A^c is given as:

$$A^c = \begin{bmatrix} p_{northwest} & p_{north} & p_{northeast} \\ p_{west} & p_{center} & p_{east} \\ p_{southwest} & p_{south} & p_{southeast} \end{bmatrix} \quad (1)$$

Given a direction to reach the destination of π which is say East, A^c can be rotated to apply the probabilities.

4.2 Compressing the Movement Model

The compressed movement model A^c can be computed from the movement model A. We compute the probability p_{south} of A^c as:

$$p_{south} = \frac{\sum_{s,s',\text{ where } s'=south(s,\pi)} \Pr(s'|s,\pi)}{\sum_{s,s'} \Pr(s'|s,\pi)}$$

where the state *south*(s, π) is computed as follows: The set of directions {*North*, *Northeast*, *East*, *Southeast*, *South*, *Southwest*, *West*, *Northwest* } is rotated such that the *up-front* vector (North) becomes the direction to reach to the destination of π from s. Then, *south*(s, π) is the neighbouring state of the state s in the new *South*. In a similar way, we can define *north*(s, π), *northeast*(s, π) and so on. The other probabilities of A^c are computed in a similar manner.

4.3 Expanding Compressed Movement Models

The movement model A can be computed from the compressed movement model A^c as follows: Note that $A = \Pr(s'|s, \pi)$, where s' is a neighbouring state of s (including s), and A^c is shown in Eq. 1. We first determine the relation among s, s' and π. If $s' = \mathit{north}(s, \pi)$, then $\Pr(s'|s, \pi) = p_{\mathit{north}}$, if $s' = \mathit{northeast}(s, \pi)$, $\Pr(s'|s, \pi) = p_{\mathit{northeast}}$, and so on.

4.4 Learning the Compressed Transition Models

The movement model of the behaviour π, i.e. $A = \Pr(s' \mid s, \pi)$, and the observation of each camera C, i.e. $B = \Pr(o|s, C)$, form a Hidden Markov Model. We propose an algorithm based on the expectation maximisation (EM) algorithm for the Hidden Markov Model (HMM) to learn the compressed movement model A^c. We term this algorithm the EM algorithm for the HMM with compressed parameters (Algorithm 1.1).

The inputs for Algorithm 1.1 are the compressed observation models for the cameras and a set of training sequences. The compressed observation models of the cameras are learned as in Section 3. They remain unchanged throughout the algorithm. To generate the required training data for the algorithm, we determine all cameras that can view the execution of the behaviour π, and record a set of video sequences of people executing π using these cameras. We take each video sequence as input to the tracking system to extract the person's trajectory. The trajectory is converted to a sequence of cells or observations. As a result, we have a set of observation sequences for the behaviour π.

In the beginning, the algorithm initialises the initial state probability π and the compressed movement model A^c. It also expands the compressed observation model $B^c(C_k)$ to the observation model $B(C_k)$ $(k = 0, \ldots, \mathit{no\_camera} - 1)$. In the main loop, for each observation sequence o_1^j, o_2^j,...,$o_{m_j}^j$ $(j = 1, 2, \ldots, \mathit{no\_seq})$, the algorithm finds the camera that generates this observation sequence. Assume that camera C_k is found. The algorithm expands the compressed movement model A^c to the full movement model A. Then, it computes the sufficient statistics, which are the expected frequency (number of times) in a state s at time $t = 1$, i.e. $\bar{\pi}_j$, and the expected number of transitions from a state s to a state s', i.e. $\bar{A}_j$. $\bar{A}_j$ is compressed to $\bar{A}_j^c$. After obtaining the expected sufficient statistics for all observation sequences, we estimate the parameters of the HMM for the next iteration as $\pi = \mathit{normalise}(\sum_{j=1}^{\mathit{no\_seq}} \bar{\pi}_j)$ and $A^c = \mathit{normalise}(\sum_{j=1}^{\mathit{no\_seq}} \bar{A}_j^c)$. The algorithm terminates when the likelihood score has reached a local minimum, and we obtain the compressed movement model A^c.

Algorithm 1.1 The EM algorithm for the HMM with compressed parameters.

```
input
  Obs. sequences $o^j_1, o^j_2, \ldots, o^j_{m_j}$ ($j = 1, \ldots, no\_seq$)
  $B^c(C_k)$, $k = 0, \ldots, no\_camera - 1$)
output
  Compressed movement model $A^c$
begin
  Initialise $\pi^{(1)}$, $A^{c(1)}$
  Expand $B^c(C_k) \rightarrow B(C_k), \forall k = 0, \ldots, no\_camera - 1$
  for $i$=1 to $N$
    for $j$=1 to $no\_seq$
      Get camera $C_k$ which generates $o^j_1, o^j_2, \ldots, o^j_{m_j}$
      Expand $\rightarrow A^{(i)}$
      From $\pi^{(i)}$, $A^{(i)}$, $B(C_k)$ and $o^j_1, o^j_2, \ldots, o^j_{m_j}$, compute:
        $\bar{\pi}^{(i)}_j$: expected frequency in state $s$ at time $t = 1$
        $\bar{A}^{(i)}_j$: expected number of transitions from $s$ to $s'$
        Compress $\bar{A}^{(i)}_j \rightarrow \bar{A}^{c(i)}_j$
    end
    Compute $\pi^{(i+1)}$, $A^{c(i+1)}$ as:
      $\pi^{(i+1)} = normalise(\sum_{j=1}^{no\_seq} \bar{\pi}^{(i)}_j)$
      $A^{c(i+1)} = normalise(\sum_{j=1}^{no\_seq} \bar{A}^{c(i)}_j)$
  end
  return $A^c = A^{c(N+1)}$
end
```

5 System Implementation in Real Environments

The implementation of the surveillance system in an office-like environment is described in [1]. The environment has a Corridor, a Staff room and a Vision lab. The surveillance system has six static cameras, in which two are in the Corridor, one in the Staff room and the last three in Vision lab.

A three-level behaviour hierarchy is defined in the system (Fig. 2). The behaviours at the bottom level represent the movement of a person within a single region (Corridor, Staff room or Vision lab). The behaviours at the middle level represent the movement of a person in the whole environment. The top level behaviours represent the different tasks that a person might perform during the entire interval that the person stays in the environment, i.e. printing the documents, using the library or an unclassified task.

The set of parameters of the behaviour hierarchy are described in [1]. The parameters of the middle level and high level behaviours are defined manually, but they can be learned easily by observing many real scenarios. The movement models of the bottom level behaviours and the camera observation models are specified as follows:

5.1 Specifying the Compressed Observation Models for the Cameras

We learn the compressed observation models for the six cameras $C_0, \ldots, C_5$ as in Section 3. We let people walk in the environment and record a set of video sequences seen

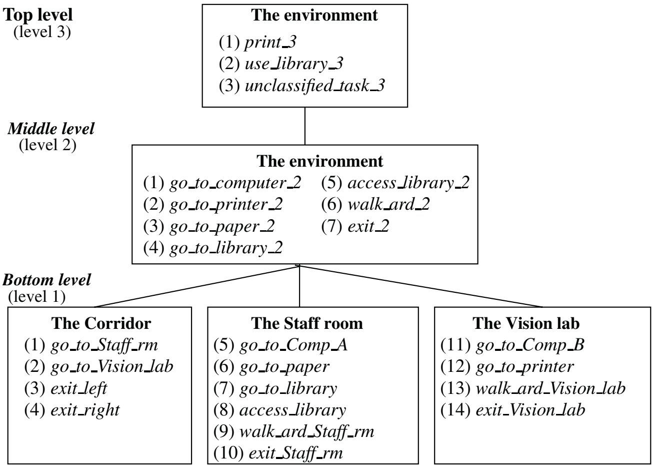


Fig. 2. The behaviour hierarchy.

from the six cameras. With each camera, we obtain 100 coordinates of the people returned from the tracking system and manually get the corresponding true coordinates. From these coordinates, we estimate the compressed observation model for the camera.

Fig. 3(a)-(f) show the compressed observation models learned for the six cameras. Note that a coordinate of a person returned from the tracking system (the person's observation) is the centre of the bottom edge of the person's bounding box. Therefore, the observation of a person is usually nearer the camera than the person's true position. This explains why in the compressed observation model for camera C_0, probabilities $\Pr(o_{north}|C_0)$ and $\Pr(o_{east}|C_0)$ are quite high (see Fig. 3(a)). The probabilities of the compressed observation models for the other cameras show the same property.

5.2 Specifying the Movement Models for Bottom Level Behaviours

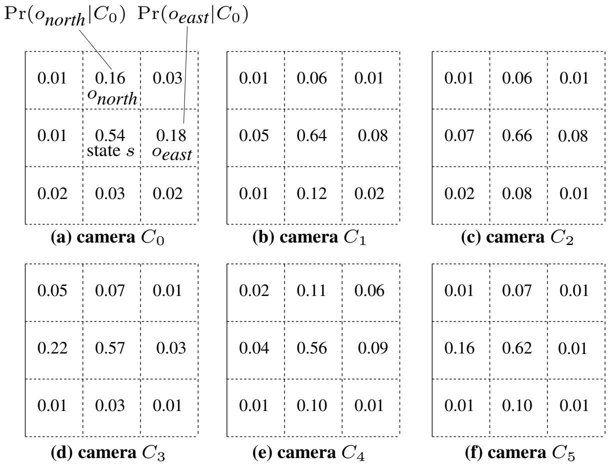
The bottom level behaviours, which translate to the full or compressed movement models are learned as described in Section 4.

We learn the compressed movement model of behaviour *go_to_printer* as follows: Behaviour *go_to_printer* can be viewed from cameras C_0, C_1 and C_5. We record 30 video sequences of people executing behaviour *go_to_printer* from cameras C_0, C_1 and C_5. Then, we use Algorithm 1.1 to obtain the compressed movement model of behaviour *go_to_printer* (see Fig. 4(a)).

The movement models of other bottom level behaviours are learned in a similar way. For example, the results of the learning steps for behaviours *go_to_Linux*, *exit_Vision*, *go_to_paper*, *go_to_library* and *exit_Staff* are shown in Fig. 4(b)-(f).

6 Experiments and Results

To demonstrate that the parameters specified in Sections 5.1 and 5.2 allow the surveillance system to recognise and monitor people behaviour reliably, we tested the system



0.01	0.16 o_{north}	0.03
0.01	0.54 state s	0.18 o_{east}
0.02	0.03	0.02

(a) camera C_0

0.01	0.06	0.01
0.05	0.64	0.08
0.01	0.12	0.02

(b) camera C_1

0.01	0.06	0.01
0.07	0.66	0.08
0.02	0.08	0.01

(c) camera C_2

0.05	0.07	0.01
0.22	0.57	0.03
0.01	0.03	0.01

(d) camera C_3

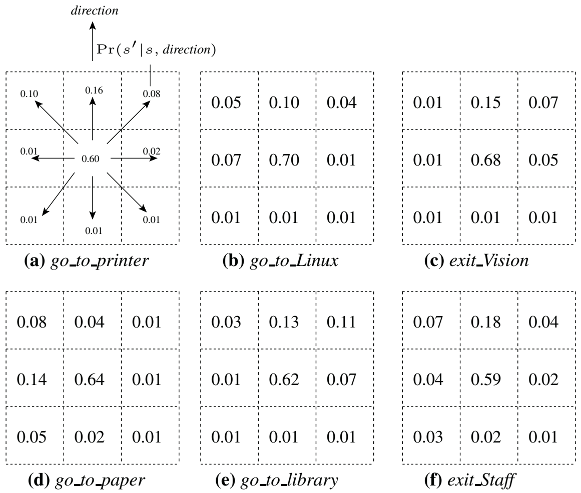
0.02	0.11	0.06
0.04	0.56	0.09
0.01	0.10	0.01

(e) camera C_4

0.01	0.07	0.01
0.16	0.62	0.01
0.01	0.10	0.01

(f) camera C_5

Fig. 3. The observation models for the six cameras C_0,...,C_5.



0.10	0.16	0.08
0.01	0.60	0.02
0.01	0.01	0.01

(a) *go_to_printer*

0.05	0.10	0.04
0.07	0.70	0.01
0.01	0.01	0.01

(b) *go_to_Linux*

0.01	0.15	0.07
0.01	0.68	0.05
0.01	0.01	0.01

(c) *exit_Vision*

0.08	0.04	0.01
0.14	0.64	0.01
0.05	0.02	0.01

(d) *go_to_paper*

0.03	0.13	0.11
0.01	0.62	0.07
0.01	0.01	0.01

(e) *go_to_library*

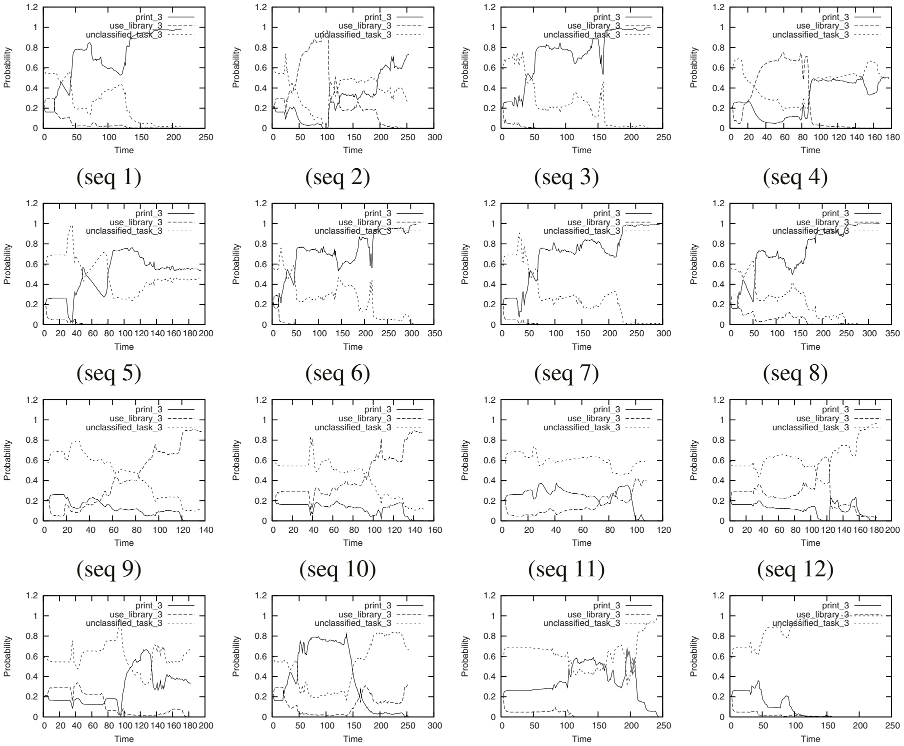
0.07	0.18	0.04
0.04	0.59	0.02
0.03	0.02	0.01

(f) *exit_Staff*

Fig. 4. The compressed movement models of behaviours *go_to_printer*, *go_to_Linux*, *exit_Vision*, *go_to_paper*, *go_to_library* and *exit_Staff*.

with 16 video sequences. In each video sequence, a person performs a task of printing, using the library or an unclassified task. The results of recognising these behaviours over time are shown in Fig. 5. As in the figure, with each video sequence and at each time slice, the system can recognise the most likely behaviour being executed by the person. The *winning* top level behaviour is available only at the end of the corresponding video sequence.

We compare the *winning* top level behaviours recognised by the system in the 16 video sequences with the groundtruth. The system correctly recognises the top level behaviours in 15 video sequences and misclassifies the top level behaviour in video sequence 11 (see Table 1). In video sequence 11, a person is executing behaviour *use_library_3*, but the system recognises *unclassified_task_3* as the *winning* behaviour (see Fig. 5, seq 11). This is because the person changes direction suddenly just before leaving the environment. A re-examination of the diagram in Fig. 5 (seq 11) does show that $p_{use\_library\_3}$ is approximately 0.4, and is significantly better than p_{print_3}. These results show that the system is able to robustly recognise people activities.



(seq 1) (seq 2) (seq 3) (seq 4)

(seq 5) (seq 6) (seq 7) (seq 8)

(seq 9) (seq 10) (seq 11) (seq 12)

(seq 13) (seq 14) (seq 15) (seq 16)

Fig. 5. Querying the top level behaviour in the 16 video sequences.

7 Conclusion

We have presented the techniques that can be used to learn camera observation models and human movement models. These techniques are used in a surveillance system for recognising and monitoring high-level human behaviours from multi-camera surveillance data. The system can query the high-level behaviours executed by a person over time. Behaviour classification results in a real environment demonstrate the ability of the system to provide real-time monitoring of high level behaviours in complex spatial environments with large state spaces.

Table 1. The results of recognising the top level behaviour in the 16 video sequences.

Seq.	*Winning* behaviour	Time periods that the *winning* behaviour has the highest probability	Compared with groundtruth
1	*print_3*	26-27, 42-213=END	correct
2	*print_3*	194-253=END	correct
3	*print_3*	38-45, 47-231=END	correct
4	*print_3*	89-93, 145-146, 170-170, 172-172, 174-176=END	correct
5	*print_3*	48-51, 79-194=END	correct
6	*print_3*	30-36, 48-311=END	correct
7	*print_3*	51-51, 53-57, 68-294=END	correct
8	*print_3*	54-323=END	correct
9	*use_library_3*	82-136=END	correct
10	*use_library_3*	95-149=END	correct
11	*unclassified_task_3*	1-107=END	wrong
12	*unclassified_task_3*	1-106, 123-181=END	correct
13	*unclassified_task_3*	1-106, 133-181=END	correct
14	*unclassified_task_3*	1-46, 149-252=END	correct
15	*unclassified_task_3*	1-108, 158-159, 162-191, 201-203, 207-246=END	correct
16	*unclassified_task_3*	1-239=END	correct

References

1. Nguyen, N.T., Bui, H.H., Venkatesh, S., West, G.: Recognising and monitoring high-level behaviours in complex spatial environments. In: IEEE Conference on Computer Vision and Pattern Recognition, Madison, Wisconsin (2003) 620–625
2. Oliver, N., Horvitz, E., Garg, A.: Layered representations for human activity recognition. In: Proceedings of the Fourth IEEE International Conference on Multimodal Interfaces. (2002) 3–8
3. Ivanov, Y., Bobick, A.: Recognition of visual activities and interactions by stochastic parsing. IEEE Transactions on Pattern Recognition and Machine Intelligence **22** (2000) 852–872
4. Galata, A., Johnson, N., Hogg, D.: Learning variable length Markov models of behaviour. Intenational Journal of Computer Vision and Image Understanding **81** (2001) 398–413
5. Hoey, J.: Hierarchical unsupervised learning of event categories. In: IEEE Workshop on Detection and Recognition of Events in Video, Vancouver, Canada (2001) 99–106
6. Bui, H.H.: A general model for online probabilistic plan recognition. In: The 18th International Joint Conference on Artificial Intelligence (IJCAI 2003), Acapulco, Mexico (2003)
7. Pynadath, D.V., Wellman, M.P.: Generalized queries on probabilistic context-free grammars. IEEE Transactions on Pattern Analysis and Machine Intelligence **20** (1998) 65–77
8. Bui, H.H., Venkatesh, S., West, G.: Policy recognition in the Abstract Hidden Markov Model. Journal of Artificial Intelligence Research **17** (2002) 451–499
9. Doucet, A., de Freitas, N., Murphy, K., Russell, S.: Rao-Blackwellised particle filtering for dynamic Bayesian networks. In: Proceedings of the Sixteenth Annual Conference on Uncertainty in Artificial Intelligence, Stanford, California, AAAI Press (2000) 176–183
10. Bui, H.H.: Efficient approximate inference for online probabilistic plan recognition. In: AAAI Fall Symposium on Intent Inference for Users, Teams and Adversaries, Falmouth, Massachusetts (2002)