Learning Situation Models in a Smart Home

Oliver Brdiczka, James L. Crowley, Patrick Reignier

Learning Situation Models in a Smart Home

Oliver Brdiczka, James L. Crowley, and Patrick Reignier

***Abstract*—This article addresses the problem of learning situation models for providing context-aware services. Context for modeling human behavior in a smart environment is represented by a situation model describing environment, users and their activities. A framework for acquiring and evolving different layers of a situation model in a smart environment is proposed. Different learning methods are presented as part of this framework: role detection per entity, unsupervised extraction of situations from multimodal data, supervised learning of situation representations, and the evolution of a predefined situation model with feedback. The situation model serves as frame and support for the different methods, permitting to stay in an intuitive declarative framework. The proposed methods have been integrated into a whole system for smart home environment. The implementation is detailed and two evaluations are conducted in the smart home environment. The obtained results validate the proposed approach.**

***Index Terms*—Human-centered computing, Context-Awareness, Situation modeling, Machine learning, Situation split.**

I. Introduction

Smart environments have enabled the computer observation of human (inter)action within the environment. Computerized spaces and their devices require situational information, i.e. context [1], to respond correctly to human activity. In order to become context-aware, computer systems must thus maintain a model describing the environment, its occupants and their activities. Situations are semantic abstractions from low-level contextual cues that can be used for constructing such a model of the scene. The situation model [2] and the underlying concepts are motivated by models of human perception of behavior in the environment. Human behavior is described by a finite number of states, called *situations*. These situations are characterized by entities playing particular *roles* and being in *relation* within the environment. Perceptual information from the different sensors in the environment is associated to the situations, roles and relations. The different situations are connected within a network. A path in this network (called *script*) describes behavior in the scene. System services to provide are associated to the different situations in the network. Figure 1 gives a simple example of a situation model for a lecture room. The situations "empty", "lecture" and "audience" are characterized by the roles "lecturer" and "audience" as well as the relation "notSameAs". System services can then be associated to the situations (e.g. service "switch on projector" to situation "lecture").

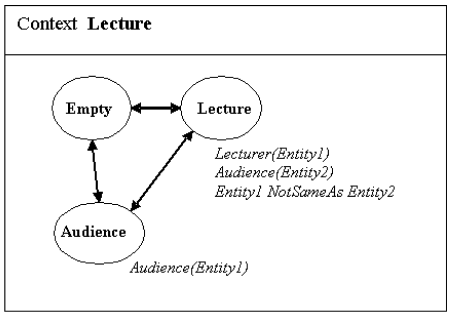

Fig. 1. Example of a simple situation model for a lecture room. **Empty**, **Audience** and **Lecture** are the available situations. *Lecturer*, *Audience* are the available roles and *NotSameAs* the available relation

Human behavior and needs evolve over time. A context model representing behavior and needs of the users must hence also evolve. Machine learning methods are necessary to acquire such a model from observation data and to adapt it according to changing behavior and needs. System reasoning and behavior must, however, be kept transparent for the users. It is hence essential to operate on a human understandable context model like the situation model, representing user behavior and needs as well as system service execution.

This article proposes a framework for learning situation models. The objective is to build up and evolve a context model for providing context-aware services in a smart environment. The proposed framework consists of different machine learning methods that acquire and adapt a situation model with different levels of supervision. The approach has been implemented and evaluated in the smart home environment of the PRIMA research group.

Oliver Brdiczka is with Telecooperation Group at TU Darmstadt, Germany. Email: brdiczka@tk.informatik.tu-darmstadt.de

James L. Crowley and Patrick Reignier are with PRIMA research group, INRIA Rhône-Alpes, France.

Manuscript received April 13, 2007; revised September 26, 2007.

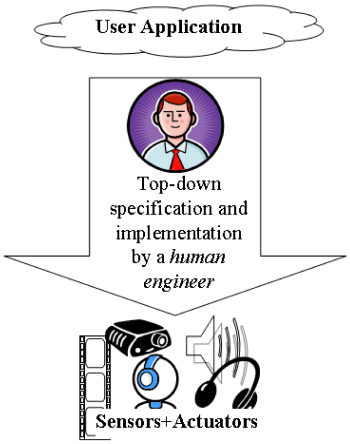

Fig. 2. Top-down manual specification and implementation of a context model

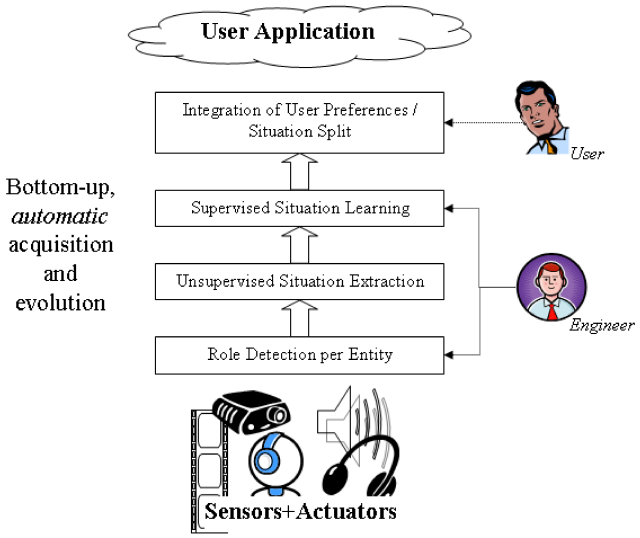

Fig. 3. Bottom-up acquisition and evolution of a context model using an automatic framework

II. Problem Definition and Approach

Experts normally define and implement context models according to the needs of users and application (Figure 2). Based on user needs and envisaged application, a human engineer specifies and implements the context model. Sensor perceptions, context model and system services to be provided are associated manually.

Human behavior evolves over time. New activities and scenarios emerge in a smart environment, others disappear. New services must be integrated into the environment, while obsolete services should be deleted. A fixed context model is thus not sufficient. Experts can construct and adapt context models according to changing needs of users and application. However, experts are expensive and not always available. Moreover, the environment's intelligence lies in its ability to adapt its operation to accommodate the users. The research challenge is thus to develop machine learning methods for this process, making it possible to automatically acquire and evolve context models reflecting user behavior and needs in a smart environment (Figure 3). Intelligibility [3] of the employed context model and the reasoning process is essential in order to permit the users to trust the system.

The proposed approach addresses the problem by providing an intelligible framework for acquiring and evolving an intuitive, comprehensible context model of the scene, the *situation model*. The methods proposed as part of this framework acquire different layers of the situation model, with different levels of supervision (Figure 3). The situation model serves as frame and support for the different learning methods, permitting to stay in an intuitive declarative framework. First, roles are learned and detected using support vector machines based on collected data labeled by an expert [4]. Situations are then extracted in an unsupervised manner from observation data using the Jeffrey divergence between sliding histograms [6]. The extracted situation segments can then be used to learn situation labels with user or expert input [5]. The resulting situation model can finally be evolved according to user feedback using the situation split [7].

III. Implementation

In this section, we describe our current implementation. The implementation is based on a 3D tracking system that creates and tracks targets in our smart home environment. The extracted target are used to detect individual roles per entity (subsection III-B). Using the role values of several entities, observations are generated that are the input for unsupervised situation extraction. The results of the extraction are used for supervised situation learning. The learned situation model is then the basis for the integration of user preferences, i.e. associating and changing services.

A. Smart Home Environment: 3D tracker, microphone array and head set microphones

The experiments described in the following sections are performed in our laboratory mockup of a living room environment in a smart home. The smart room is equipped with a wide-angle camera (Figure 4) plus two other normal cameras mounted in the corners of the room. A microphone array mounted against the wall of the smart environment is used for noise detection. The

speech of people in the scene is recorded using head set microphones.

Fig. 4. The Smart Room environment as seen by the wide angle camera

A 3D video tracking system [8] detects and tracks entities (people) in the scene in real-time using multiple cameras (Figure 5). The tracker itself is an instance of basic Bayesian reasoning combining tracking results from several 2D trackers [9] running on the video images of each camera. Each couple camera-detector is running on a dedicated processor. All interprocess communication is managed with an object oriented middleware for service connection [10].

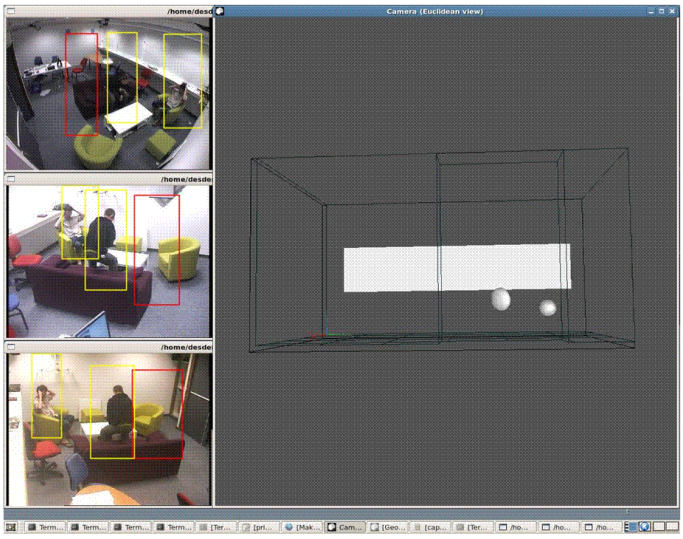

Fig. 5. 3D video tracking system fusing information of 3 2D trackers to a 3D representation

The output of the 3D tracker are the position (x, y, z) of each detected target as well as the corresponding covariance matrix (3x3 matrix describing the form of the bounding ellipsoid of the target). Additionally, a velocity vector $\vec{v}$ can be calculated for each target.

The microphone array is used for noise detection. Based on the energy of the audio streams, we determine whether there is noise in the environment or not (e.g. movement of objects on the table). A real-time speech activity detector [6] analyzes the audio stream of each head set microphone and determines whether the corresponding person speaks or not.

The association of the audio streams (microphone number) to the corresponding entity (target) generated by the 3D tracker is done at the beginning of each recording by a supervisor.

Ambient sound, speech detection and 3D tracking are synchronized. As the audio events have a much higher frame rate (62.5 Hz) than video (up to 25 Hz), we add sound events (no sound, speech, noise) to each video frame (of each entity).

B. Role Detection per Entity

Role detection is conducted per entity (person) and for each observation frame. The input are the extracted properties of each target (position (x, y, z), 3x3 covariance matrix and speed $|\vec{v}|$) provided by the 3D tracking system. The output is one of the role labels (Figure 8 bottom).

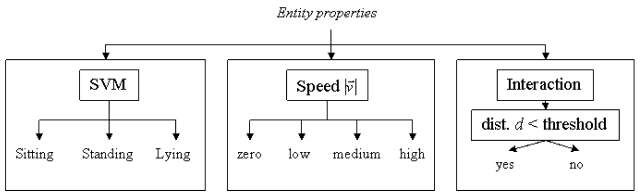

Fig. 6. Role detection process: SVMs (left), Target Speed (middle), Distance to Interaction Object (right)

The role detection process consists of 3 parts (Figure 6). The first part is based on support vector machines (SVMs). A first approach used only SVMs as a black box learning method, without considering specific target properties. From first results obtained in our smart home environment [4], we concluded that, in order to optimize role recognition, we need to reduce the number of classes as well as the target properties used for classification. Additional classes are determined by using specific target properties (speed, interaction distance) and expert knowledge (see parts 2 and 3 of the role detection process).

The first part of the process (Figure 6 left) takes the covariance matrix values of each target as input. Trained SVMs detect, based on these covariance values, the basic individual roles “sitting”, “standing” and “lying down“ (Figure 7).

Fig. 7. Basic individual roles "standing", "lying down" and "sitting" detected by the SVMs

The second part of the process (Figure 6 middle) uses the speed value $\mid \overrightarrow{v} \mid$ of each target. Based on empirical values in our smart environment, we can then determine whether the speed of the target is $zero$, low, $medium$ or $high$.

The third part of the process (Figure 6 right) uses the position (x, y, z) of each target to calculate the distance to an interaction object. In our smart environment, we are interested in the interaction with a table at a known position (white table in Figure 4). So we calculate the distance d between the target and the table in the environment. If this distance is approaching zero (or below zero), the target is interacting with the table.

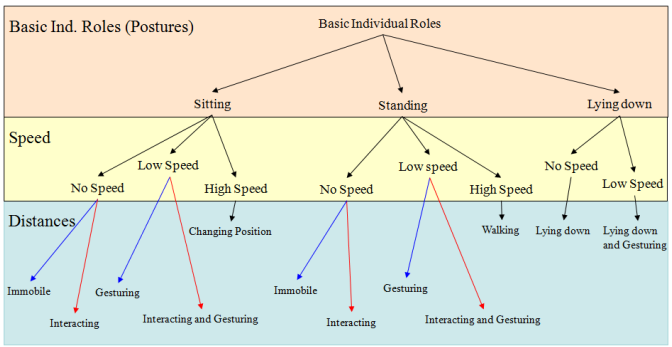

Fig. 8. Schema describing the combination of basic individual role, speed and distance values to roles (blue arrows refer to "no interaction distance with table", red arrows refer to "interaction distance with table")

The results of the different parts of the detection process are combined to roles following the schema in Figure 8.

Based on role detection results as well as the ambient sound and speech detection, we derive multimodal observation codes for each entity created and tracked by the 3D tracking system. 12 individual role values (Figure 8 bottom) are derived for each entity by the role detection process. Further, the ambient sound detector indicates whether there is noise in the environment or not. The speech activity detector determines whether the concerned entity is speaking or not. This multimodal information is fused to 53 observation codes for each entity. Codes 1-13 (12 role values + 1 error code) are based on the role detection process. These 13 codes are combined with ambient sound detection (codes 27-39 and 40-52) and speech detection per entity (codes 14-26 and 40-52). As ambient sound and speech detection return binary values, $2^2 * 13 = 52$ different code values are necessary to represent role, ambient sound and speech detection. If we add an observation code value for a non-existing entity (code 0), we get 53 different observation code values.

Fusion algorithm

Input: (a, b), $0 \leq a, b \leq max_{code}$

Step 1: if $(a > b)$ $\{\text{exchange}(a, b)\}$,

Step 2: $code = \sum_{i=0}^{a-1}\{(max_{code} + 1) - i\} + (b - a)$.

Fig. 9. Fusion algorithm combining the multimodal observation values (a, b) of two entities. For $max_{code} = 52$, the resulting codes are between 0 and 1430

As we can have several persons involved in a situation, multimodal observations of several entities are fused by combining the individual observation codes (Figure 9). The idea is to attribute a code to the combination of two multimodal entity observation codes (without considering their order). The resulting observation code fuses the observation codes of two (or more) entities. In order to fuse the observation codes of more than two entities, the fusion can be applied several times, fusing successively all entities.

IV. Experimental Evaluation and Results

In this section, we detail the experimental evaluations that have been conducted as well as the obtained results. We did 2 different evaluations (Figure 10).

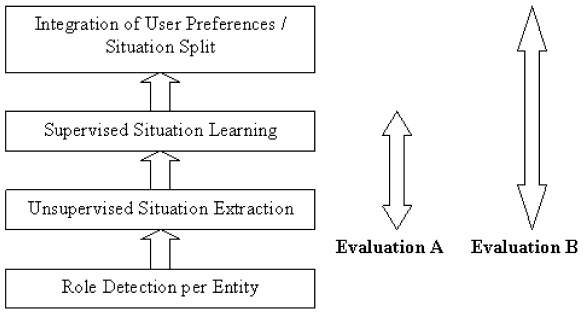

Fig. 10. Different parts of the implementation and their evaluation: role detection per entity, unsupervised situation extraction, supervised situation learning and integration of user preferences

The aim of Evaluation A was to investigate the quality of one-person and multi-person situation learning and

recognition using the proposed framework. Therefore, we recorded several small scenarios showing different situations like “presentation” or “siesta”. The recordings have been segmented automatically and the situations have been learned using the methods of the framework. We evaluate the recognition of the one-person and multi-person situations in the scenarios with and without automatic presegmention. The aim of Evaluation B was to show and validate the combination of the three methods: unsupervised situation extraction, supervised situation learning and integration of user preferences. Therefore, we recorded 3 long scenarios showing several situations like “aperitif”, “playing game” or “presentation”. The recordings have first been automatically segmented. Then, the extracted segments have been labeled and the situations have been learned. Finally, the learned situation model has been evolved with user feedback. We evaluate the recognition of the labeled as well as the added situation (via situation split).

A. Evaluation A

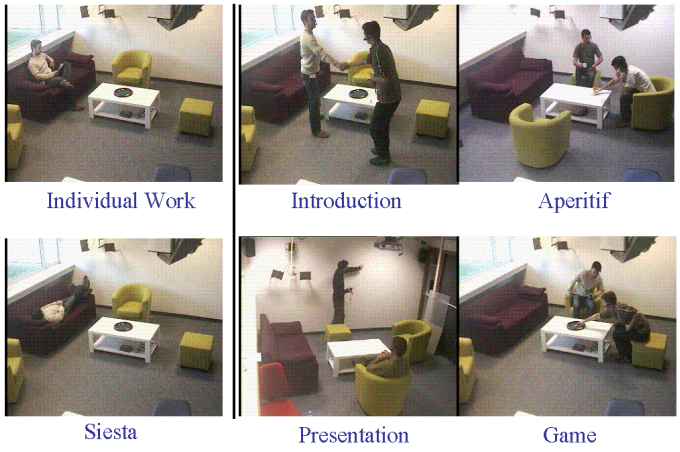

Fig. 11. One person situations ”individual work“ and ”siesta“ (left side) and multi-person situations ”introduction“, ”aperitif“, ”presentation“ and ”game“

In this subsection, we aim at investigating the quality of one-person and multi-person situation learning and recognition. Therefore, we made three different recordings of each of the following situations: ”siesta“, ”an individual working“, ”aperitif“, ”introduction/address of welcome“, ”presentation“, and ”playing a game“. ”Introduction/address of welcome“, ”aperitif“, ”presentation“ and ”playing a game“ involved two persons, while ”siesta“ and ”individual work“ concerned only one person. The role detection values have been generated as described in subsection III-B. The sequences designated for learning are presegmented, i.e. only the segment containing the pure situation is used for learning. This means that, for recordings containing only one situation, disturbances at the beginning and at the end of the recording are automatically removed (see Figure 12 for an example). The supervised learning scheme [5] is then used for learning the situation representations from the sequences. We adopt left-right hidden Markov models as unique learner class for the situations.

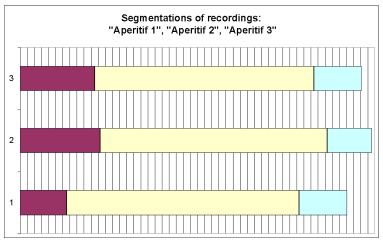

Fig. 12. Extracted segments for situation recordings ”Aperitif 1”, ”Aperitif 2”, ”Aperitif 3”. Segments at the beginning and at the end of the recordings will be removed automatically

First, we did an evaluation on the situation detection for one person only (role detection value between 0 and 52). Situation recordings involving 2 people gave thus two one-person sequences. We did a 3-fold cross-validation, taking two third of the sequences as input for supervised learning and the remaining third of the sequences as basis for recognition. Table I shows the results. The presegmentation improves the recognition results for the one-person recording sequences. In particular, ”aperitif“ and ”game“ can correctly be distinguished, while some wrong detections between ”introduction“ and ”presenter“ persist.

Additionally, we did an evaluation on the situation detection for two-person situations. Therefore, multi-person observation codes have been generated from the individual role detection values. We did again a 3-fold cross-validation on the situation recognition after supervised situation learning of the given observation sequences. Table II shows the results. The presegmentation also improves the recognition results for the two-person recordings. As for the one-person situation detection, situations ”aperitif“ and ”game” can correctly be distinguished with presegmentation. The two-person observation fusion further eliminates wrong detections between “aperitif” and “game”, resulting in a correct situation recognition rate of 100 % (Table II). The obtained results indicate that multi-person observation and presegmentation of observation streams is beneficial when learning and recognizing situations.

B. Evaluation B

In this subsection, we intend to show and validate the combination of the three methods: unsupervised

	Preseg.	Siesta	Ind. Work	Aper.	Intro.	Pres.	Game	Aud.
Siesta	No	1	0	0	0	0	0	0
	Yes	1	0	0	0	0	0	0
Ind. Work	No	0	1	0	0	0	0	0
	Yes	0	1	0	0	0	0	0
Aperitif	No	0	0	0.83	0	0	0.17	0
	Yes	0	0	1	0	0	0	0
Introduc.	No	0	0	0	0.83	0.17	0	0
	Yes	0	0	0	0.83	0.17	0	0
Presenter	No	0	0	0	0	1	0	0
	Yes	0	0	0	0	1	0	0
Game	No	0	0	0	0	0	1	0
	Yes	0	0	0	0	0	1	0
Audience	No	0	0	0	0	0	0	1
	Yes	0	0	0	0	0	0	1

Class	Preseg.	TP rate	FP rate	Precision	Recall	F-measure
Siesta	No	1	0	1	1	1
	Yes	1	0	1	1	1
Ind. Work	No	1	0	1	1	1
	Yes	1	0	1	1	1
Aperitif	No	0.83	0	1	0.83	0.89
	Yes	1	0	1	1	1
Introduc.	No	0.83	0	1	0.83	0.89
	Yes	0.83	0	1	0.83	0.89
Presenter	No	1	0.04	0.83	1	0.89
	Yes	1	0.04	0.83	1	0.89
Game	No	1	0.04	0.89	1	0.93
	Yes	1	0	1	1	1
Audience	No	1	0	1	1	1
	Yes	1	0	1	1	1
Total	No	0.95	0.01	0.96	0.95	0.94
	Yes	0.98	0.01	0.98	0.98	0.97

TABLE I

CONFUSION MATRIX AND INFORMATION RETRIEVAL STATISTICS FOR ONE-PERSON SITUATION DETECTION WITHOUT AND WITH PRESEGMENTATION. THE TOTAL RECOGNITION RATE IS 93.33 % (WITHOUT PRESEG.) AND 96.67 % (WITH PRESEG.).

	Preseg.	Aperitif	Introduc.	Presentation	Game
Aperitif	No	0.67	0	0.33	0
	Yes	1	0	0	0
Introduc.	No	0	1	0	0
	Yes	0	1	0	0
Presentation	No	0	0	1	0
	Yes	0	0	1	0
Game	No	0	0	0	1
	Yes	0	0	0	1

Class	Preseg.	TP rate	FP rate	Precision	Recall	F-measure
Aperitif	No	0.67	0	0.67	0.67	0.67
	Yes	1	0	1	1	1
Introduc.	No	1	0	1	1	1
	Yes	1	0	1	1	1
Presentation	No	1	0.11	0.83	1	0.89
	Yes	1	0	1	1	1
Game	No	1	0	1	1	1
	Yes	1	0	1	1	1
Total	No	0.92	0.03	0.88	0.92	0.89
	Yes	1	0	1	1	1

TABLE II

CONFUSION MATRIX AND INFORMATION RETRIEVAL STATISTICS FOR TWO-PERSON SITUATION DETECTION WITH AND WITHOUT PRESEGMENTATION. THE TOTAL RECOGNITION RATE IS 91.67 % (WITHOUT PRESEG.) AND 100.00 % (WITH PRESEG.).

situation extraction, supervised situation learning and integration of user preferences. Therefore, we evaluated the integral approach on 3 scenarios recorded in our smart home environment. The scenarios involved up to 2 persons doing different activities (situations: "introduction/address of welcome", "presentation", "aperitif", "playing a game", "siesta"{1 person}) in the environment. The role detection values have been generated as described in subsection III-B using 3D tracker as well as noise and speech detection (head set microphones). The role detection values have then been fused to multi-person observations.

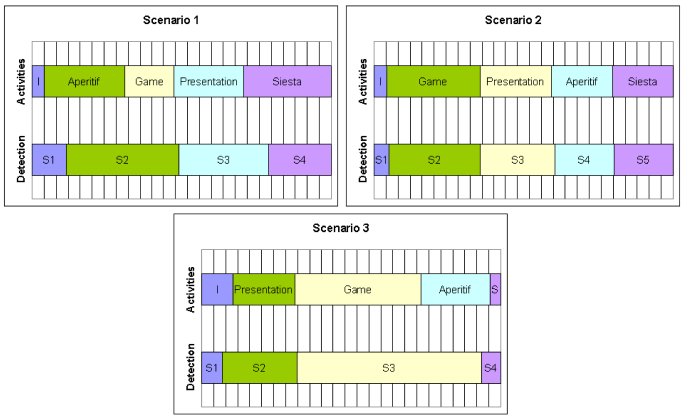

Fig. 13. Extracted situation segments and the corresponding *ground truth* for scenario 1 (Q = 0.68), scenario 2 (Q = 0.95), scenario 3 (Q = 0.74)

The first step of our proposed approach is to create the initial situation model. We extract the situations from the sensor perceptions, i.e. the observations generated for the targets in the scene using our automatic segmentor [6]. The automatically extracted segments and the *ground truth* for the scenarios are depicted in Figure 13. The overall segmentation exactitude Q [11] is best for scenario 2. This can be explained by the fact that the algorithm has difficulties to distinguish ground truth segments "game" and "aperitif". In scenario 1 and scenario 3, "game" and "aperitif" are detected as one segment. Because in scenario 2, "playing game" and "aperitif" are separated by "presentation", these segments can be correctly detected.

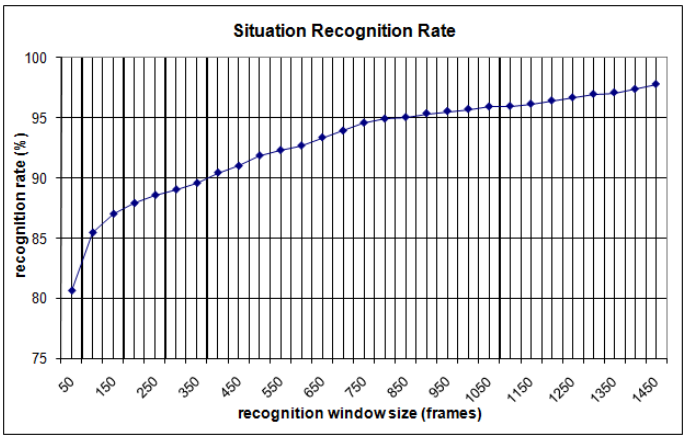

Fig. 14. Recognition rate of situations "introduction", "presentation", "group activity" (="aperitif" or "game") and "siesta" for different recognition window sizes

The supervised learning scheme [5] is applied on the detected segments. As expert knowledge, we inject the

situation labels: “introduction”, “presentation”, “group activity” (=“aperitif” or “game”), “siesta”. We will adopt left-right hidden Markov models as unique learner class for the situations. To evaluate, we did 3-fold cross-validation, taking the detected segments + expert labels of 2 scenarios as input for learning and the third scenario as basis for recognition. As our system should be as responsive as possible, we evaluated different window sizes used for recognition. The obtained situation recognition rates are depicted in Figure 14. If we limit the observation time provided for recognition to 10 seconds (i.e. 250 frames with a frame rate of 25 frames/sec), we get a recognition rate of 88.58 % (Table III). The recognition rate of “siesta” is poor due to the fact that in two of the three scenario recordings wrong targets have been created and detected when a person lay down on the couch, resulting in a disturbance of the existing target properties.

	Introduction	Group Activity	Presentation	Siesta
Introduction	0.98	0	0	0.02
Group Activity	0	1	0	0
Presentation	0.03	0.03	0.84	0.10
Siesta	0.22	0.01	0.32	0.45

Class	TP rate	FP rate	Precision	Recall	F-measure
Introduction	0.98	0.06	0.64	0.98	0.71
Group Activity	1	0.02	0.99	1	0.99
Presentation	0.84	0.02	0.96	0.84	0.88
Siesta	0.44	0.04	0.81	0.45	0.44
Total	0.82	0.03	0.85	0.82	0.76

TABLE III
CONFUSION MATRIX AND INFORMATION RETRIEVAL STATISTICS FOR EACH SITUATION (OBSERVATION WINDOW SIZE=250). THE OVERALL SITUATION RECOGNITION RATE IS 88.58 %

We have now learned an initial situation model with the situations “introduction”, “group activity”, “presentation” and “siesta”. In order to integrate user preferences into this model, a user can give feedback to our system. The feedback is recorded and associated to the particular frame when it has been given. The initially learned model is then adapted according to this feedback. For our scenarios, we want to integrate the following services:

- S1: Introduction ⇒ normal light and no music
- S2: Aperitif ⇒ dimmed light and jazz music
- S3: Game ⇒ normal light and pop music
- S4: Presentation ⇒ dimmed light and no music
- S5: Siesta ⇒ dimmed light and yoga music

The user gives one feedback indicating the corresponding service during each situation. As the initial situation model does not contain any situation-service associations, S1, S4 and S5 can then be simply associated to the corresponding situations. For S2 and S3, there is only one situation “group activity” which is too general in order to associate both distinct services. This situation needs thus to be split into sub-situations (following the situation split scheme of [7]). The learned situation representation for “group activity” (here: a HMM) is erased and two distinct situation representations (here: HMMs) for “aperitif” and “game” are learned. The observations necessary to learn these situations are taken around the time points when the user gave the corresponding feedback. The size of the observation window used for learning the new sub-situations can be varied. The situation recognition rates for different learning window sizes are depicted in Figure 15. We used a window size of 250 observations for recognition (i.e. 10 seconds of observation time with a frame rate of 250 frames/sec). The curve indicates that a larger learning window size does not always result in a better recognition rate. The total situation recognition rate can even drop with a larger learning window size. This is due to the fact that the best recognition results are obtained when the learning window contains a maximum of observation data being characteristic for the concerned situation and a minimum of “foreign“ observations, i.e. wrong detections or observations corresponding to other situations. The resulting situation recognition curve tends upwards, but it contains local peaks corresponding to a learning window size with a good tradeoff between characteristic and foreign observations. For our scenario recordings, such a local peak is at a learning window size of 400, i.e. 400 observations around the feedback time points to learn “aperitif” and “game”. The obtained results of the 3-fold cross validation for recognition window size 250 are detailed in Table IV.

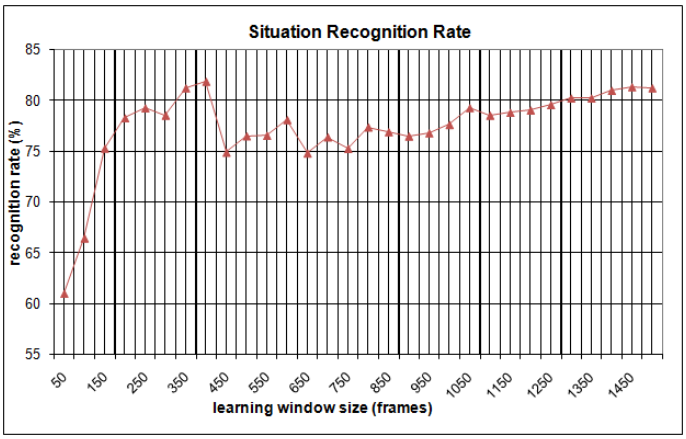

Fig. 15. Recognition rate of situations “introduction”, “presentation”, “aperitif”, “game” (after split) and “siesta” for different learning window sizes. The curve is for 250 observations (recognition window size)

V. CONCLUSIONS

Although the obtained results are encouraging, the realization of a smart home anticipating the needs and

	Introduction	Aperitif	Game	Presentation	Siesta
Introduction	0.97	0	0	0	0.03
Aperitif	0	0.70	0.30	0	0.01
Game	0	0.01	0.99	0	0
Presentation	0.04	0	0.03	0.84	0.10
Siesta	0.22	0	0	0.33	0.45

Class	TP rate	FP rate	Precision	Recall	F-measure
Introduction	0.97	0.06	0.63	0.97	0.70
Aperitif	0.70	0.00	0.98	0.70	0.80
Game	0.99	0.088	0.81	0.99	0.88
Presentation	0.84	0.02	0.96	0.84	0.88
Siesta	0.45	0.04	0.80	0.45	0.44
Total	0.79	0.04	0.83	0.79	0.74

TABLE IV

CONFUSION MATRIX AND INFORMATION RETRIEVAL STATISTICS FOR EACH SITUATION (OBSERVATION WINDOW SIZE=250) AFTER THE SPLIT OF "GROUP ACTIVITY". THE WINDOW SIZE FOR LEARNING THE NEW SUB-SITUATIONS IS 400. THE OVERALL SITUATION RECOGNITION RATE IS 81.86 %

preferences of the user is still far away. First products that a user could buy in his local computer store and install himself are not mature enough. First, the sensors necessary for a reliable sensing of user activities are still too invasive. Multiple cameras, microphones or other sensors must be installed and calibrated in the home. These are still not auto-installing and not easy to use. Second, even though our results are encouraging, the error rates are still too high. Further improvements in detection and learning algorithms are necessary in order to provide a reliable system that could be accepted by a user in his daily life. One way to alleviate this is to provide explanations. When errors occur (and corresponding system explanations are good), the user could understand and correct wrong system perceptions and reasoning himself.

REFERENCES

[1] A. K. Dey, "Understanding and using context," *Personal and Ubiquitous Computing*, vol. 5, no. 1, pp. 4–7, 2001.

[2] J. L. Crowley, J. Coutaz, G. Rey, and P. Reignier, "Perceptual components for context aware computing," in *Proceedings of UbiComp*, 2002, pp. 117–134.

[3] V. Bellotti and K. Edwards, "Intelligibility and accountability: Human considerations in context-aware systems," *Human-Computer Interaction*, vol. 16, pp. 193–212, 2001.

[4] O. Brdiczka, J. Maisonnasse, P. Reignier, and J. Crowley, "Learning individual roles from video in a smart home," in *Proceedings of 2nd IET International Conference on Intelligent Environments*, vol. 1, July 2006, pp. 61–69.

[5] O. Brdiczka, P. Yuen, S. Zaidenberg, P. Reignier, and J. Crowley, "Automatic acquisition of context models and its application to video surveillance," in *Proceedings of ICPR*, 2006, pp. 1175–1178.

[6] O. Brdiczka, D. Vaufreydaz, J. Maisonnasse, and P. Reignier, "Unsupervised segmentation of meeting configurations and activities using speech activity detection," in *Proceedings of 3rd IFIP AIAI Conference*, June 2006, pp. 195–203.

[7] O. Brdiczka, P. Reignier, and J. L. Crowley, "Supervised learning of an abstract context model for an intelligent environment," in *Proceedings of sOc-EUSAI Conference*, October 2005, pp. 259–264.

[8] A. Biosca-Ferrer and A. Lux, "A visual service for distributed environments: a bayesian 3d person tracker," , PRIMA internal Report, 2007.

[9] A. Caporossi, D. Hall, P. Reignier, and J. L. Crowley, "Robust visual tracking from dynamic control of processing," in *Sixth IEEE PETS Workshop*, May 2004.

[10] R. Emonet, D. Vaufreydaz, P. Reignier, and J. Letessier, "O3miscid: an object oriented opensource middleware for service connection, introspection and discovery," in *Proceedings of 1st IEEE International Workshop on Services Integration in Pervasive Environments*, June 2006.

[11] D. Zhang, D. Gatica-Perez, S. Bengio, I. McCowan, and G. Lathoud, "Multimodal group action clustering in meetings," in *Proceedings of 2nd VSSN workshop*, 2004, pp. 54–62.

Oliver Brdiczka received the diploma degree in computer science from the University of Karlsruhe (TH) and the engineer's degree from ENSIMAG (Ecole National Superieure d'Informatique et de Mathematiques Appliqués). He further holds a PhD degree from Institut National Polytechnique de Grenoble (INPG). Dr. Brdiczka pursued his PhD research with PRIMA research group at the INRIA Rhône-Alpes research center. He currently leads the Ambient Collaborative Learning Group which is part of the Telecooperation Group at TU Darmstadt, Germany. His research interests include multimodal meeting processing, context modeling and machine learning.

James L. Crowley leads the PRIMA research group at the INRIA Rhône-Alpes research center in Montbonnot (near Grenoble), France. He holds the post of Professor at the Institut National Polytechnique de Grenoble (INPG), where he teaches courses in Computer Vision, Signal Processing, Pattern Recognition and Artificial Intelligence at ENSIMAG (Ecole National Superieure d'Informatique et de Mathematiques Appliqués). Professor Crowley has edited two books, five special issues of journals, and authored over 180 articles on computer vision and mobile robotics. He ranks number 1473 in the CiteSeers most cited authors in Computer Science (August 2006).

Patrick Reignier is a researcher at the LIG laboratory at the INRIA Rhône-Alpes research center in Montbonnot (near Grenoble), France. He holds the post of Assistant Professor at the University Joseph Fourier (UJF) in Grenoble.