AUTOMATIC MULTI-MODAL MEETING CAMERA SELECTION FOR VIDEO-CONFERENCES AND MEETING BROWSERS

Marc Al-Hames, Benedikt Hörnler, Ronald Müller, Joachim Schenk, and Gerhard Rigoll

Technische Universität München
Institute for Human-Machine Communication
Arcisstrasse 21, 80333 München, Germany
{alh, hbe, mur, joa, rigoll}@mmk.ei.tum.de

ABSTRACT

In a video-conference the participants usually see the video of the speaker. However if somebody reacts (e. g. nodding) the system should switch to his video. Current systems do not support this. We formulate this camera selection as a pattern recognition problem. Then we apply HMMs to learn this behaviour. Thus our system can easily be adapted to different meeting scenarios. Furthermore, while current systems stay on the speaker, our system will switch if somebody reacts. In an experimental section we show that – compared to a desired output – a current system shows the wrong camera more than half of the time (frame error rate 53%), where our system selects the wrong camera in only a quarter of the time (FER 27%).

1. INTRODUCTION

A lot of people think that most meetings are just a waste of time [1]. On the other hand they are often mandatory for many of us and can consume a large part of our working days. Projects like the ICSI meeting project [2], Computers in the Human Interaction Loop [3], or Augmented Multi-party Interaction [4] therefore investigate how computers can be used to make meetings and lectures more effective. In this work we address a problem that occurs in two different scenarios: Video-conferences [5] and meetings in a smart room [6].

In a video-conference the participants are in different locations. Each participant is recorded with a camera and a microphone. This audio-visual data is then transmitted to all other participants. Usually the audio stream is preprocessed such that only the active speaker is indeed played. This process is similar to phone conferences (see e. g. *Skype* as a non- and *Spiderphone* as a commercial version). The video channel is different: Current versions either show the active speaker and therefore simply reuse the audio information; or they show a selection or all participants of the meeting at the same time by scaling down the individual video streams until all persons fit on the display (see e. g. *InterCall*'s *InView* solution, or *Visual Nexus*). Neither approach is a good solution: Showing all participants is limited to a few participants. With an increasing number the individual videos get to small. The second approach of simply showing the video of the active speaker is straight-forward and reduces the video size problem. But by doing that the video has only limited extra information: Imagine someone gives a presentation. As he is the only person speaking, he will always be shown. This way you loose the very important information, that the project manager is shaking his head constantly, indicating he is not satisfied with the idea.

This work is supported by the European IST Programme Project FP6-0033812. This paper only reflects the authors' views and funding agencies are not liable for any use that may be made of the information contained herein.

Meetings are truly multi-modal in nature [7], thus it can be very important to show persons who currently do not speak. Professional directors of talk-shows follow this rule and from time to time show facial reactions or gestures of the participants. Thus a good video-conference system should neither show all participants at the same time, nor simply show the speaker, but choose one of the participants based on both the audio information, as well as visual information.

In the second scenario all participants are located in the same room and the meeting is recorded with multiple cameras and microphones. Such smart meeting rooms become increasingly important, as the recordings allow to analyse the meeting content, as well as a later comprehension of the decisions [3, 4]. Then the recordings together with some high level information can be watched in a meeting browser [8]. However it is usually not possible to simply view all recorded video streams at the same time; thus it is necessary to select one camera and show this stream to the user. Of course this view will in general change within the course of the meeting.

Thus, while video-conferences and local meetings are sociologically quite different; the problem of selection a camera is the same for both scenarios: for each time instance (generally frames) of the meeting we need to select one camera or – as we refer them to – video mode that shows best what happens in the meeting. Generally a mode is a camera view, but could also be a slide or two merged videos (see Sec. 3). This mode is then transmitted to the other participants or stored for browsing. The problem can therefore be described as an automatic, virtual meeting director. While the task is commercially very interesting, it has not yet been deeply researched. Previous works suggest video editing rules for the camera decision [9, 10]. In [11] a controllable camera is used and the view is automatically learned. [12] proposes a system to extract relevant meeting regions from wide screen cameras. A user study with expert camera operators [13] offers suggestions how to design an interface. For video surveillance, [14] suggests how to select cameras, but the decision concentrates only on video quality. Thus, the results from these works can not be directly applied to conference scenarios.

We suggest to formulate the camera selection as a pattern recognition problem, where each possible video mode is modelled as a pattern class. The problem can then be reduced to classify each frame of the meeting to one of the classes (i.e. video modes). This way we can train machine learning algorithms and use them for the camera selection. We propose a system based on different Hidden Markov Model (HMM) techniques. We extract audio-visual features (Sec. 4) from a data set (Sec. 2) and use them in an early fusion HMM (Sec. 5.2), as well as in a problem adapted two layer HMM (Sec. 5.3). Finally the proposed methods are evaluated (Sec. 6) and compared to the state-of-the-art rule-based approach (Sec. 5.1).

Fig. 1. Sample shots from the data set: centre (C) view of the room, shot from the left (L), and a closeup view of a participant (C_4).

2. MEETING ROOM AND DATA SET

The data for this work has been collected in the AMI project and is publicly available [15]. Each meeting has four participants. We use a subset of 24 five minute videos, each with different participants.

All meetings have been recorded in the IDIAP smart meeting room [6]. This room is equipped with a table, a whiteboard, and a projector with a screen. Close-talking audio is recorded with an omni-directional lapel and a headset with condenser microphone for each participant. Far-field recordings are performed with two microphone arrays. Video is recorded with seven static cameras: four cameras record participants closeup views ($C_1 - C_4$). Two cameras record a left (L), resp. right (R) view of the room; each showing two participants and the table in front of them. The last camera (C) captures a total of the room with all four participants, the table, as well as the whiteboard, and the projector screen. Three sample shots from these cameras are shown in Fig. 2. The closeup recording corresponds to the camera recordings in a video-conference scenario.

3. VIDEO MODES AND ANNOTATION

For each frame of the meeting we have to select one camera or one view. We will refer to these possible views as video modes V_k. In the case of a video-conference, each participants camera represents one mode, furthermore slides could be another mode. Thus in a video-conference with four persons we would have five modes. For browsing a recorded meeting, we use each camera in the meeting room as one possible video mode. Thus we have seven modes. However, the method is not limited to these modes. New ones can easily be added: If – for example for discussions – one needs a view where one person is blended into the corner of another person (i. e. correspondent view of news shows), we could define this as a new mode and simply train a new class without influencing the existing modes. This way the system can easily be adapted to various needs and applications without changing the underlying system. For an extensive discussion on possible video modes in meetings see [10].

To apply our pattern recognition approach we needed training data for the video mode classes. We therefore set up a limited set of annotation rules, ensuring some basic guidelines: Mainly preventing annotators from very fast switches between the cameras (we encouraged them to stay for at least 10 seconds on one view). However we gave the annotators the freedom to select cameras they thought would best represent the meeting at a given time. Thus the degree of freedom was rather high. Consequently, first studies showed that inter-annotator agreement on the data set was rather low ($\kappa < 0.5$) and therefore not consistent enough. Further studies showed, that persons where very consistent if they annotated the same meeting more than once. This shows that the annotation and the desired camera view indeed depends on the taste of the annotator, but then represents a consistent selection. We therefore decided to use only two annotators, to ensure a consistent training data set.

4. FEATURES

Global Motion Features: As first feature we use global motions (GM). They are simple, but have been successfully applied to various meeting tasks [16] and can be calculated in real-time with a latency of only one frame. We split the room into six locations L. Each of the four closeup cameras represents one location. From the centre view camera we extract the projection board and the whiteboard location. Then a difference image sequence $I_d^L(x,y,t)$ is calculated for each of these six locations and each frame t by subtracting the pixel values of two subsequent frames from the video stream. Then the centre of motion is calculated for the x- and y-direction:

$$m_x^L(t) = \frac{\sum_{(x,y)} x \cdot |I_d^L(x,y,t)|}{\sum_{(x,y)} |I_d^L(x,y,t)|}, \quad m_y^L(t) = \frac{\sum_{(x,y)} y \cdot |I_d^L(x,y,t)|}{\sum_{(x,y)} |I_d^L(x,y,t)|} \tag{1}$$

The changes in motion are used to express the dynamics of the movements:

$$\Delta m_x^L(t) = m_x^L(t) - m_x^L(t-1), \quad \Delta m_y^L(t) = m_y^L(t) - m_y^L(t-1) \tag{2}$$

Furthermore the mean absolute deviation of the pixels relative to the centre of motion is computed:

$$\sigma_x^L(t) = \frac{\sum_{(x,y)} |I_d^L(x,y,t)| \cdot \left(x - m_x^L(t)\right)}{\sum_{(x,y)} |I_d^L(x,y,t)|}$$

and

$$\sigma_y^L(t) = \frac{\sum_{(x,y)} |I_d^L(x,y,t)| \cdot \left(y - m_y^L(t)\right)}{\sum_{(x,y)} |I_d^L(x,y,t)|} \tag{3}$$

Finally the intensity of motion is calculated from the average absolute value of the motion distribution:

$$i^L(t) = \frac{\sum_{(x,y)} |I_d^L(x,y,t)|}{\sum_{x,y} 1} \tag{4}$$

These seven features are concatenated for frame t in the location dependent motion vector $\vec{x}^L(t)$. With this vector the high dimensional video is reduced to a seven dimensional vector, but it preserves the major characteristics of the motion. Concatenating the vectors of the six positions L leads to the final GM feature vector $\vec{x}_{\text{GM}}(t)$ that describes the motion in the meeting with only 42 features.

Skin Blob Features: A further way to access the participants activities are hand and head movements. In [17] it was shown how skin blobs can be used to detect the activity of individual meeting participants. We therefore add skin blobs (SB) as a visual feature.

We extract the head and hand SBs with a skin colour look up table. The RGB-images are transformed into the rg-space. Each pixel is then compared to a 16 bit rg-look up table, which results in a binary image, where each possible skin pixel is marked. To fill gaps in skin areas, a 5x5 dilation filter is applied. The found skin areas are then analysed for their shape, the relation of their eigenvalues, and context knowledge about possible positions. Finally subsequent images are averaged with a recursive approach, that is applied individually to blobs in the meeting videos $\vec{m}(t) = 1 - \frac{1}{T}\vec{m}(t-1) + \frac{1}{T}\vec{x}(t)$, where $\vec{x}(t)$ is the current measured value, $\vec{m}(t)$ is the resulting averages vector for the blob position, $\vec{m}(t-1)$ the position in the last image, and T a constant that determines the relation between previous frames and the current measurement. The position and movement of each participant's blobs are concatenated in the final SB motion vector $\vec{x}_{\text{SB}}(t)$. This approach is simple but can be performed in real-time; more details can be found in [10].

Acoustic Features: From each participant's lapel microphone we extract 12 Mel frequency cepstral coefficients (MFCC) and the energy, as well as the first and second derivations. This results in a 39 dimensional acoustic feature vector $\vec{x}_{\text{MFCC}}(t)$ for each participant.

5. VIDEO MODE SELECTION MODELS

5.1. State-of-the-Art Rule-Based Model

For comparison we summarise the state-of-the-art rule-based approach (for details see e.g. [10]). In the following let t denote the current time step, W the window size, $P \in \{P_1, P_2, P_3, P_4\}$ one of the meeting participants, and $E^P(t)$ the audio energy for person P at time t. The windowed output of the feature is denoted as $D^P(t)$ and derived by summing up the energy in the window:

$$D^P(t) = \sum_{\tau=t-W}^{t} E^P(\tau) \tag{5}$$

The output $D^P(t)$ therefore represents what has recently happened in the audio channel of person P. For each time step t, the rule-based systems then chooses the "most active" person with

$$k(t) = \operatorname*{argmax}_P D^P(t) \tag{6}$$

Depending on the desired output, this decision $k(t)$ is now directly mapped to one of the video modes $V_k(t)$ (e.g. an activity of person two will of course show the mode corresponding to camera two). This process does not optimise the features, nor does it model interactions between the features, it simply uses the energy. Yet, it is reliable and the behaviour well controlled, thus it is widely applied.

5.2. Hidden Markov Model

We search for a sequence of camera views from the meeting. As we formulated this video selection as a pattern recognition problem and provided data with annotated video modes, we can apply the Hidden Markov Model (HMM) [18]. It can be used for classification of feature streams. In combination with the Viterbi algorithm [19] it also segments the stream into a sequence of video modes.

For the recognition with HMMs, each video mode is modelled by one HMM. Each HMM k (and thus each video mode) is represented by a set of parameters $\lambda_k = (\mathbf{A}, \mathbf{B}, \vec{\pi})$, where $\mathbf{A}$ denotes the transition matrix, $\vec{\pi}$ the initial state distribution, and $\mathbf{B}$ is the output distribution, here modelled with mixtures of Gaussians.

For the HMMS, we can use only audio ($\vec{x}_{\text{MFCC}}$), visual ($\vec{x}_{\text{GM}}$ and/or $\vec{x}_{\text{SB}}$), or all features. The selection of the video-mode should be based on both the acoustic and the visual information. Thus we use an early fusion HMM: The frame rates of the streams are adjusted and then concatenated into one multi-modal feature stream $\vec{x}$.

Given this multi-modal training data $\mathbf{X}_k$ from our data set for mode k, the parameters λ_k of the HMM k can be trained with the well known EM-algorithm [20]. The aim of this training is to maximise $p(\mathbf{X}_k|\lambda_k)$. For the training of this HMM k only representatives of the video mode k are used. The resulting models are therefore independent from each other. The HMM corresponding to the centre view is only trained with representatives of this mode. This HMM neither takes the number of classes into account, nor does it know other modes. Thus the system can easily be expanded with new modes: The other – already trained – HMMs are not influenced. One simply needs to train a new HMM for each new video mode, this makes the approach very flexible and easily adaptable.

Once an HMM for each class (i.e. video mode) is trained, the unknown video feature stream $\vec{x}$ is presented to all HMMs λ_k and we select the model k with $k = \operatorname{argmax}_i p(\vec{x}|\lambda_i)$ the highest likelihood. This is done with an online version of the Viterbi algorithm [19], which can also perform a segmentation of the streamed input vector $\vec{x}$. This way, the feature stream of the meeting is automatically segmented into a sequence of video modes: the desired sequence of camera views from the meeting.

5.3. Two-layer Hidden Markov Model

Compared to the rule-based approach, the early fusion HMM reacts on both visual and acoustic information and implicitly models the relation between the streams. However, the virtual director should react on the individual actions. Mainly it should stay on the speaker, but if somebody reacts, the system should switch to this person. If the training data represents this behaviour, we can assume that the early fusion HMM learns and therefore models this behaviour.

On the other hand we can explicitely model this with a two-layer HMM: the first layer recognises the individual actions of each participant. These recognised actions together with group related features (e.g. the motion in front of the whiteboard) are then used as input for the second layer that decodes the actual video mode.

For the person HMM layer we defined 14 important individual actions: e.g. standing up or sitting down, but also more subtle actions like nodding or shaking of the head. We use the actions of all four participants in the meeting to train the models, i.e we have a person independent training. Thus we effectively have four times the training data available. The second layer is then trained analogous to the early fusion HMM. However we extend the early fusion feature vector $\vec{x}$ with the person actions: we add the action of each participant in a coded way for each frame of the meeting resulting in the extended feature vector $\vec{x}^e$. This way the video mode HMM explicitely learns the relation between person actions and desired video mode output, but preserves the implicit learning of feature relations. The complete training procedure can then be summarised in

Algorithm 1 Two-Layer HMM Training

Require: Training feature vectors $\mathbf{X}$
  for all person actions A_j **do**
    $\lambda_{A_j} \leftarrow$ train person action HMM, s.t. $\max P(\mathbf{X}_{A_j}|\lambda_{A_j})$
  end for
  $\mathbf{X}^e \leftarrow$ extend the features $\mathbf{X}$ with the true person actions a_i
  for all videomodes V_k **do**
    $\lambda_{V_k} \leftarrow$ train video mode HMM, s.t. $\max P(\mathbf{X}^e_{V_k}|\lambda_{V_k})$
  end for

In the recognition phase we apply a two-fold decoding: First the unknown feature stream $\vec{x}$ is used to classify the actions of each person in the meeting. Then the feature vector $\vec{x}$ is extended with the found person actions, resulting in the extended stream $\vec{x}^e$. This feature stream now explicitely comprehends the found individual actions. Finally $\vec{x}^e$ is used to segment and classify the video mode in the second layer. This way the video mode HMM has explicit information about the person actions, however they are of course afflicted with some uncertainty (note the difference to the training, where the true actions are available). While the process separates the individual actions from the video mode, it introduces some latency: The first layer first has to decode the feature streams, and then this output is fed into the second layer, thus the second layer is always a couple of frames behind. The overall decoding can be summarised in

Algorithm 2 Two-Layer HMM Decoding

Require: Unknown feature vector stream $\vec{x}$
  for all persons P_i in the meeting **do**
    $a_i \leftarrow$ classify individual person action $\operatorname{argmax}_j P(\vec{x}|\lambda_{A_j})$
  end for
  $\vec{x}^e \leftarrow$ extend the stream $\vec{x}$ with the found person actions a_i
  $V \leftarrow$ classify the video mode $\operatorname{argmax}_k P(\vec{x}^e|\lambda_{V_k})$

6. EXPERIMENTS

To evaluate our proposed system we performed two experiments: For the first experiment we assumed the true shot boundaries were known, and the only task was to assign a video mode to each segment. In the second experiment the shot boundaries were unknown and the system had to segment and classify the videos, i.e. the second scenario is the true application. We used the 24 five minute videos from our data set and performed a six-fold cross-validation. We further split the experiments into two scenarios: The first contained seven video modes (all four persons, left, right, and centre camera); the second experiment corresponds to a video-conference with five modes (four persons and the centre camera for presentations).

For the classification we measured the recognition results (RR, i.e. correct found modes; high numbers are better). For the joint segmentation and classification, we measured the frame error rate (FER, i. e. proportion of frames, where a wrong video mode was selected; low numbers are better). All results are shown in Tab. 1.

In the classification task, the rule based system achieves a RR of 45% for seven, resp. 57% for five modes. The proposed multi-modal systems are significantly better: the early fusion HMM achieves 51%, resp. 72% RR. The layered HMM does not outperform the early fusion HMM. A further analysis showed that this is mainly caused by the first action layer (RR only 43%). Thus we also analysed the maximum possible performance of the two-layer HMM by providing the ground truth (GT) individual actions to the second layer. Then the two-layer HMM is slightly better than the early fusion HMM. Of course, this GT is not available in a real system.

The tendency of the classification task is even increased in the real application of joint segmentation and classification: Here the rule based approach is highly outperformed by the proposed systems. For the video-conference scenario (five modes), the rule based system selects the wrong video mode for over half the frames (53% FER). Here the early fusion HMM selects the wrong frame in only a quarter of the meeting (27% FER). Thus by applying standard machine learning techniques, we get a much better video.

Interestingly, while the absolute FERs seem quite high, the video output of the system represents a very good view upon the meeting, and only some actions of the participants are missing.

7. CONCLUSIONS AND FUTURE WORK

In this work we proposed a system for selecting a camera view in video-conferences and for browsing recorded meetings. We formulated the task as a pattern recognition problem and could therefore apply Hidden Markov Models for the segmentation of a meeting into a series of camera views. The proposed system is very flexible and can easily be adapted to different applications. Whenever a new view or camera is desired, only one new model has to be trained, without influencing the existing models, or the underlying system.

In an experimental section we showed that the proposed HMM highly outperforms the state-of-the-art rule-based method. While this system always stays on the active speaker, the proposed system changes to other channels, if somebody reacts. This leads to a video that represents the meeting much better. Currently most commercial video-conferencing systems use DSPs, thus the computational time required for the HMM decoding could easily be performed. Given the good performance of the system, this seems worth the effort.

In the future we will integrate a higher "grammar-level", to prevent fast switches between video modes and retrain the models based on used studies. Furthermore we will evaluate different machine learning techniques to further improve the system performance.

Modell	**Classification**		**Segmentation**	
	RR-7	*RR-5*	*FER-7*	*FER-5*
Rule Based	45.4%	56.6%	61.4%	53.3%
Early Fusion HMM	51.4%	71.6%	47.9%	27.0%
Two-layer HMM	51.0%	69.6%	45.9%	27.1%
Two-layer HMM (GT)	51.5%	74.2%	42.5%	22.8%

Table 1. Recognition rates (RR, high better) for classification; frame error rates (FER, low better) for segmentation and classification.

8. REFERENCES

[1] D. Heylen, A. Nijholt, and D. Reidsma, "Determining what people feel and think when interacting with humans and machines: Notes on corpus collection and annotation," in *Proc. Conf. on Recent Advances in Eng. Mechanics*, 2006.

[2] A. Janin et al., "The ICSI meeting corpus," in *ICASSP*, 2003.

[3] A. Waibel et al., "CHIL: Computers in the human interaction loop," in *Proc. NIST ICASSP Meeting Recogn. Worksh.*, 2004.

[4] M. Al-Hames et al., "Audio-visual processing in meetings: Seven questions and current AMI answers," in *P. MLMI*, 2006.

[5] S. Sabri and B. Prasada, "Video conferencing systems," *Proceedings of the IEEE*, vol. 73, no. 4, 1985.

[6] D. Moore, "The IDIAP smart meeting room," Research Report 07, IDIAP, 2002.

[7] M. Al-Hames et al., "Multimodal integration for meeting group action segmentation and recognition," in *P. MLMI*, 2006.

[8] P. Wellner, M. Flynn, and M. Guillemot, "Browsing recorded meetings with Ferret," in *Proc. MLMI*, 2004.

[9] S. Sumec, "Multi camera automatic video editing," in *Proc. ICCVG*, 2004.

[10] M. Al-Hames et al., "Using audio, visual, and lexical features in a multi-modal virtual meeting director," in *P. MLMI*, 2006.

[11] Q. Liu and D. Kimber, "Learning automatic video capture from human's camera operations," in *Proc. ICIP*, 2003.

[12] Q. Liu et al., "An online video composition system," in *Proc. ICME*, 2005.

[13] S. Uchihashi, "Direct camera control for capturing meetings into multimedia documents," in *Proc. ICME*, 2001.

[14] L. Snidaro, R. Niu, P.K. Varshney, and G.L. Foresti, "Automatic camera selection and fusion for outdoor surveillance under changing weather conditions," in *Proc. AVSS*, 2003.

[15] J. Carletta et al., "The AMI meetings corpus," in *Proc. Symposium on Annotating and Measuring Meeting Behavior*, 2005.

[16] F. Wallhoff, M. Zobl, and G. Rigoll, "Action segmentation and recognition in meeting room scenarios," in *Proc. ICIP*, 2004.

[17] I. Potucek, S. Sumec, and M. Spanel, "Participant activity detection by hands and face movement tracking in the meeting room," in *Proceedings CGI*, 2004.

[18] L.R. Rabiner, "A tutorial on HMMs and selected applications in speech recognition," *Proc. of the IEEE*, vol. 77, no. 2, 1989.

[19] A. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," in *IEEE Trans. on Information Theory*, 1977.

[20] A.P. Dempster, N.M. Laird, and D.B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society B*, vol. 39, no. 1, 1977.