2008 IEEE/RSJ International Conference on Intelligent Robots and Systems
Acropolis Convention Center
Nice, France, Sept, 22-26, 2008

A Multi-Camera Person Tracking System for Robotic Applications in Virtual Reality TV Studio

Suraj Nair, Giorgio Panin, Martin Wojtczyk, Claus Lenz, Thomas Friedlhuber, Alois Knoll, *Member, IEEE*

Abstract— **In this paper, an integrated multi-camera person tracking system for virtual reality television studios (*VR-TV*) is presented. The system robustly tracks the moderator while freely moving, sitting or walking around the studio, and the estimation result can be used in order to drive the main broadcasting camera mounted on a large robotic arm. Application of the proposed tracking system to real-time VR-TV results in an autonomous robot cameraman, able to keep the moderator inside the screen with jitter-free viewpoint adjustments, as required by the scene rendering engine.**

I. Introduction

Virtual TV studios (Fig. 1) provide a very impressive virtual reality experience for educational, documentary movies, as well as weather or financial forecast transmissions, only to name a few. However, the quality of the result depends on the real-time robustness and accuracy of three major components of the system, namely: 1. The camera tracker, that recovers the absolute 3D pose of the camera (usually from external infrared sensors), 2. The rendering software, that uses the estimated camera pose in order to generate a synthetic background or additional scene items, 3. The video mixer, which combines the synthetic image with the real camera input, in order to produce the final VR scene.

Therefore, the whole system requires a smooth and precise motion input, in order to produce synthetic images with the correct overlap and without undesired jittering effects. However, currently in most situations the camera is still controlled by a cameraman, that may not achieve the smoothness and precision of motion required; therefore, in such cases the camera has been mounted on a robot arm, with a few pre-planned key movements available (zoom, *fly-by*, etc.), which on the other hand limit the moderator freedom of motion.

By using auxiliary video inputs looking at the scene, together with real-time computer vision tools, the system would instead be able to localize the moderator and keep her/him within the screen while sitting or freely walking inside the studio, at the same time performing smooth camera adjustments through robot control and filtering tools, with almost no need for human intervention.

For this purpose, in this paper we propose a multi-camera, model-based person tracking system integrating color and motion modalities, that achieves a robust real-time localisation of the moderator. In order to obtain a reliable localization over the whole area, we employ a distributed system consisting of two or more cameras with different viewpoints over the scene, with an overall modular and scalable system design.

Authors Affiliation: Technische Universität München, Fakultät für Informatik.
Address: Boltzmannstrasse 3, 87452 Garching bei München (Germany).
Email: { nair,panin,wojtczyk,lenz,friedelhu,knoll }@in.tum.de

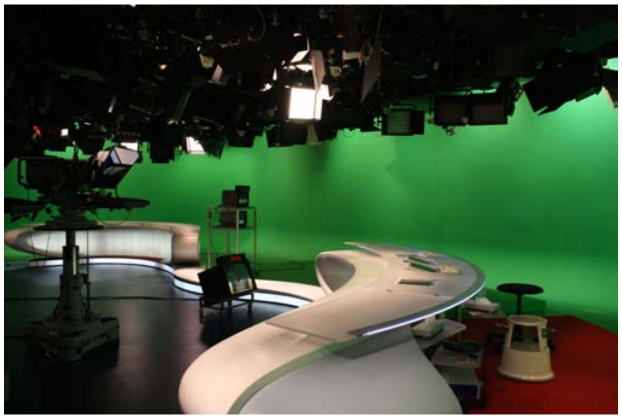
Fig. 1. A Virtual Reality TV Studio with manually operated TV Camera (*image courtesy: RTL Television Studio Köln, Germany*)

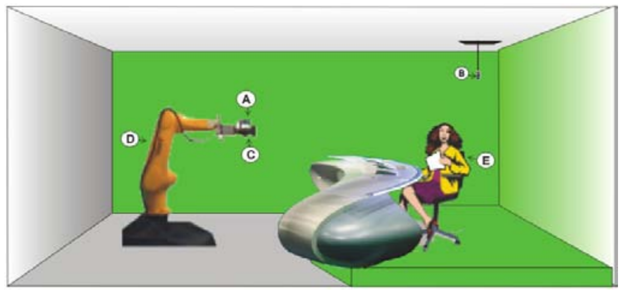

Fig. 2. Proposed model of the VR-TV studio with camera operated by a vision-assisted robot arm: A) Frontal Firewire camera for 2D person localization and tracking, B) Overhead camera for computing the moderator distance from the robot C) Broadcasting TV camera, D) Robot manipulator, E) Moderator

The present paper is organized as follows: Sec. II briefly reviews the current related state-of-the-art; Sec. III presents the system overview and provides afterwards more details, focusing on the user-interface for modeling (Sec. III-B), the tracking methodology (Sec. III-C) and the robot controller (Sec. III-D). Experimental results are given in Sec. IV, and conclusions with proposed system developments are finally given in Sec. V.

II. State-of-the-art comparison

To the knowledge of the authors, currently no vision-based robot cameraman software has yet been developed for VR-TV applications; however, the literature concerning single or multiple people tracking for application in video surveillance, mobile robotics and related fields, already counts several well-known examples, that we briefly review here.

Multiple people trackers, as also the literature shows [1][2][3], have the common requirement of using a very little and generic offline information concerning the person shape and appearance, while building and refining more precise models (color, edges, background) during the on-line tracking task; this unavoidable limitation is due to the more general context with respect to a known, single-target tracking task, for which instead specific models can be built off-line.

Many popular systems for single-target tracking are based on color histogram statistics [4][5][6][7] and employ a base, pre-defined shape and appearance model during the whole task, providing more robustness and precision as well as a relatively high speed of the tracker.

In particular, [6] uses a standard particle filter with color histogram likelihood with respect to a reference image of the target, while [5] improves this method by adapting the model on-line to light variations, which however may introduce drift problems in presence of partial occlusions; the same color likelihood is used in the well-known *mean-shift* kernel tracker [7], that follows the pose of an object in 2D by optimizing the kernel function from frame to frame; although reliable and fast, this method again suffers from drift problems in presence of occlusions, since the cost function may show false local minima and distract the tracker that afterwards needs to be manually re-initialized.

The person tracking system [4] employs a complex model of shape and appearance (color and shape blobs modeled by multiple Gaussian distributions, with articulated degrees of freedom), which require a difficult modeling phase as well as several parameters specification.

Some of the above mentioned systems [4][1][2][3] also employ *background subtraction* methods, which limit the application to the case of fixed cameras, and therefore cannot be applied to the main input from a VR-TV setup.

By comparison, in our system the off-line modeling part is kept to a minimum extent, while at the same time retaining a more detatiled information than a single color patch can provide, which makes the system more robust to unpredicted occlusions; and a robust fusion with the *motion history image* [8] is provided, which replaces the background subtraction modality that cannot be used for the mobile front camera.

III. THE INTEGRATED PERSON TRACKING SYSTEM

As stated in the Introduction, the goal of our system is to robustly provide real-time information about the current 3D position of the moderator in the scene, as well as controlling the robot camera and zooming in a fully automatic way.

Fig. 2 shows the overall setup in the VR-TV scenario; the corresponding system architecture is depicted in Fig. 3, and hereafter described.

A. System overview

The hardware setup consists of an industrial robot arm, on which the main TV camera is mounted, and two additional cameras for tracking. The first camera is mounted as a piggy back over the main TV broadcast camera, both solidal with the robot arm; this camera sees the same video scene but with a slight offset, that later on is corrected by registering the 3D output pose to the main camera frame. The tracker uses this input stream in order to obtain information about the frontal 2D image position of the moderator. The other camera is placed over the area where the moderator is supposed to move, and it is calibrated with respect to the robot base frame, in order to estimate the current distance with respect to the robot (see Fig. 2); in the current experimental setup, a single overhead camera has been sufficient, although the system can be scaled in order to cover larger floor areas, by providing the calibration parameters for each camera.

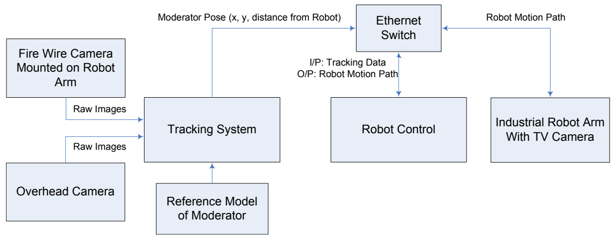
Moderator Pose (x, y, distance from Robot)
Ethernet Switch
Robot Motion Path
Fire Wire Camera Mounted on Robot Arm
I/P: Tracking Data
O/P: Robot Motion Path
Raw Images
Tracking System
Robot Control
Industrial Robot Arm With TV Camera
Raw Images
Overhead Camera
Reference Model of Moderator

Fig. 3. Block Diagram of the system

In particular, the visual system combines color and motion cues from multiple cameras, through dynamic data fusion and Bayesian filtering, in order to localize and track the moderator in real-time. This system estimates the (x, y) coordinates of the moderator through the robot-mounted camera, along with his/her distance d from the robot with the overhead cameras, and provides output visualization of the estimated 3D pose; the estimation result is sent as a feedback for the robot controller, which constantly keeps the moderator in focus (apart from occasional, pre-planned motion trajectories).

Concerning the choice of the filter, we choose in the present work a multi-patch Particle Filter, that offers a number of advantages over standard Kalman Filtering techniques.

First, particle filters more robustly deal with multi-modal likelihoods due to clutter background and other people which occasionally interact with the moderator, while a KF keeps only one Gaussian hypothesis. Therefore when additional, smaller peaks of the likelihood are present near the main one, some particles are shifted onto this region, while most of the sample remains on the correct peak.

When the temporary disturbance disappears, the sample set focuses again on the main one, and no track loss occur. A KF, instead, looks for one hypothesis only, so it may lock erroneously onto the false peaks, and never recover the right track.

Moreover, particle filters can initialize and re-initialize very easily, by means of a diffuse prior distribution of a higher number of particles, that also acts as a target detector; when detection converges to the target, the number of particles is reduced and normal tracking takes place. The reduced particle set reduces the computation cost. A KF based on local optimization would instead require a Gaussian prior, not too far from the real target location, for which an additional detection module should be implemented.

On the other hand, particle filters are usually considered slower because of the multiple hypotheses to evaluate; but in order to implement a KF measurement, a local likelihood optimization must be performed (e.g. using kernels, like mean-shift) possibly requiring state-space derivatives, so that in the end the single measurement step can be quite complex. Instead, a particle filter can be well-optimized and parallelized, so that each measurement can be quickly evaluated, without any requirement for differentiability.

Concerning computational resources, the tracking software runs on a single PC, to which both Firewire cameras are interfaced, while the robot controller runs on a seperate machine with real-time OS capabilities; the two machines communicate through a standard Ethernet link. In the following sections we describe the system in more detail.

B. Graphical user interface for modeling

In order to realize a simple and robust tracking system, we employ a minimal modeling information about the particular person to be tracked. For this purpose, a single picture is manually taken from each view, and the following informations are obtained (Fig. 4):

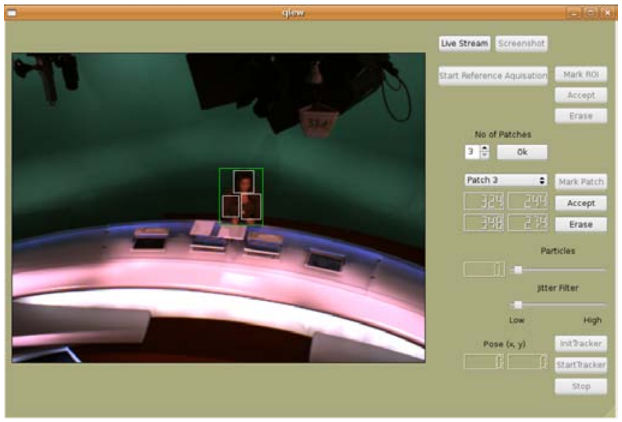

Fig. 4. User interface and selected model patches

- *Shape*: for each camera view a generic, planar shape model is given by a user-defined set of rectangular *patches* (preferably two or more) enclosing different, meaningful color regions. On each view, the main region enclosing the person is manually selected by dragging the mouse, and within this region the rectangular patches are selected as well.
- *Appearance*: The appearance model is represented by the underlying color pixels; from each reference patch, a joint histogram in 2D *hue-saturation* (HS) color space is collected and normalized.

Our multi-patch model can be defined in a flexible way for each view, and with an arbitrary number of patches; for example, from the frontal view of a standing, unoccluded person the three regions of head, torso and legs are typically defined, whereas for a sitting moderator other meaningful regions can be defined.

Both for the modeling and tracking phases, our system provides a simple and user-friendly GUI, that allows quickly obtaining the reference images and interactively defining the individual rectangular patches (Fig. 4). We design the GUI in a platform-independent way using *Trolltech-QT* and *OpenGL* rendering contexts, in order to provide the user with the main functionalities: accessing each camera and visualizing the video stream; taking a screenshot of the reference images; selecting the main region and the individual patches; setting system parameters like as motion covariance, particle filter size and jitter filter bandwidth. During tracking, over the same window the result is visualized in the form of a bounding rectangle or a cross hair, and parameters can be on-line tuned before feeding the result to the robot controller.

C. Tracking methodology

The tracking software is designed and implemented in an architecture inspired to a recently developed general-purpose framework [9], following a *tracking pipeline* concept (Fig. 5). Fig. 5 describes the complete pipeline for both cameras.

Each tracker holds a state-space representation of the 2D model pose, given by a planar translation (x, y) and scale h of the respective rectangular model in the image plane. Two particle filters provide the sequential prediction and update of the respective 2D states $s_1 = (x_1, y_1, h_1)$ and (x_2, y_2, h_2), by combining color and motion informations. Only three out of the six estimated variables are returned to the robot controller, namely the horizontal and vertical position in the first image x_1, y_1, and the distance from the robot in the overhead view d.

1) Image pre-processing: The sensor data is obtained from each firewire camera in a raw RGB format. The image is pre-processed in order to generate two pixel maps, related to the respective visual modality: an RGB to HSV conversion z_{col} is performed for the color-based likelihood, while for the motion-based likelihood the absolute motion history image z_{mot} [8] is computed.

2) Tracker prediction: The particle filter generates several prior state hypotheses s_t^i from the previous particle distribution $(s^i, w^i)_{t-1}$ through a simple dynamical model

$$s_t^i = s_{t-1}^i + v_t^i \tag{1}$$

with v a white Gaussian noise of pre-defined covariance in the (x, y, h) state variables. A deterministic resampling strategy [10] over the previous weights w_{t-1}^i is employed every time in order to keep a good distribution of the particle set.

For each generated hypothesis, the tracker asks for a computation of the likelihood values $P(z_{col}|s^i), P(z_{mot}|s^i)$, from both modalities.

3) Color likelihood: The rectangular boxes defining the person shape model are warped onto the HSV image at the predicted particle hypothesis s_t^i; for each patch p, underlying H and S color pixels are collected in the respective 2D

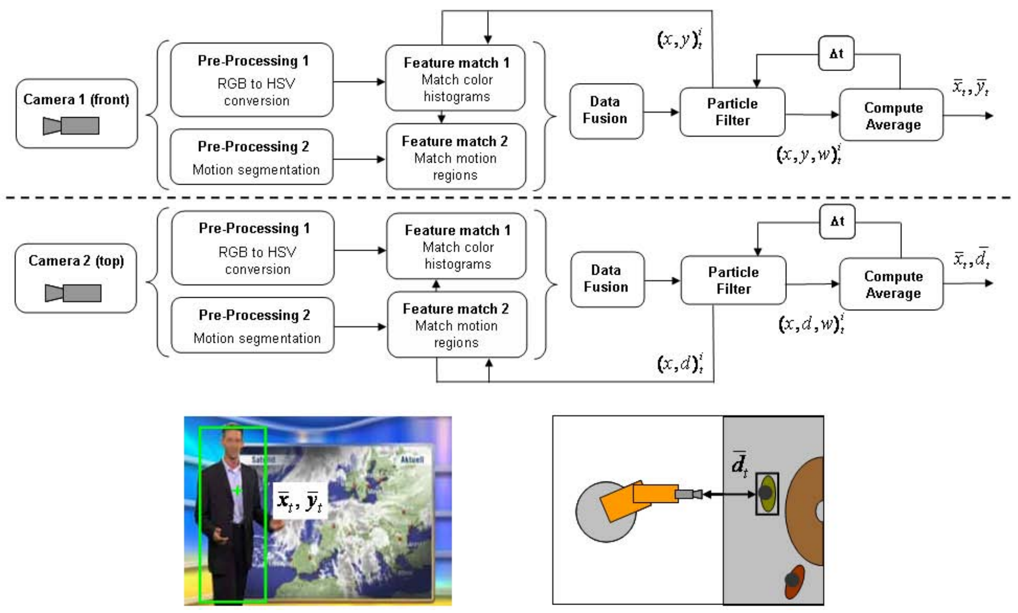

Fig. 5. Tracking pipeline and estimated variables for the robot controller

histogram $q_p\left(s_t^i\right)$, that is compared with the reference one q_p^* through the Bhattacharyya coefficient [6]

$$B_p\left(q_p\left(s\right), q_p^*\right) = \left[1 - \sum_n \sqrt{q_p^*\left(n\right) q_p\left(s, n\right)}\right]^{\frac{1}{2}} \qquad (2)$$

where the sum is performed over the $(N \times N)$ histogram bins (in the current implementation, $N = 10$).

The color likelihood is then evaluated under a Gaussian model in the overall residual

$$P(z_{col}|s_t^i) = exp(-\sum_p B_p^2/\lambda_{col}) \qquad (3)$$

with given covariance λ_{col}

4) Motion likelihood: The same multi-patch model is applied also to the motion image, and the matching residual E is computed as the overall number of non-motion pixels inside the covered area

$$P(z_{mot}|s_t^i) = exp(-E^2/\lambda_{mot}) \qquad (4)$$

with λ_{mot} the respective covariance.

5) Dynamic data fusion: The overall likelihood is finally obtained by multiplying the two independent likelihoods and updating the weights w^i

$$w_t^i = P(z_{col}|s_t^i)P(z_{mot}|s_t^i) \qquad (5)$$

that afterwards are normalized, in order to ensure $\sum_i w_t^i = 1$.

This corresponds to a *dynamic data fusion* scheme [11][12], where the two image measurements z_{col}, z_{mot} are combined at the tracker level, under temporal state predictions s_t^i.

By fusing complimentary visual modalities, both trackers obtain more robustness in presence of unexpected occlusions or light changes, although the latter seldom occur in a controlled studio environment.

6) Output: The average state $\overline{s}_t$

$$\overline{s}_t = \sum_i w_t^i s_t^i \qquad (6)$$

is computed for both trackers, and the depth $\overline{d}$ is computed from the two overhead variables $(\overline{x}_2, \overline{y}_2)$ using the calibration model. The three components $(\overline{x}_1, \overline{y}_1, \overline{d})$ are finally returned to the robot controller.

Another improvement, in order to ensure jitter-free operations, is obtained by filtering the tracking output by a further temporal low-pass filter, with higher cut-off frequency with respect to the expected covariance of motion parameters (v_x, v_y, v_d).

The multi-patch likelihood function is already quite target-specific, and its values around the correct peak are usually much higher than the other modes; therefore, the weighted average keeps a correct position. However, in order to achieve a very stable output for the robot controller, in the present work we further filter our small noise fluctuations; for future developments, a more robust evaluation of the particle average can be considered.

7) Loss detection and re-initialization: An important feature of our system is the possibility to automatically detect a track loss when the person leaves the scene or gets occluded, and to re-initialize the system in such situations. In principle there are two main techniques to determine loss of target, 1) Covariance check for the particle set, 2) Likelihood check. A covariance test would be independent on the actual likelihood value, but it may fail to detect a loss when the particle set concentrates on a small peak (a false positive) which has a low covariance as well. This is very undesirable in a TV studio application, where the only target that should be detected is the moderator, and never other people or objects to which the robot could drift. On the other hand, the likelihood test is dependent on the likelihood value, which may detect a loss for example in presence of light variations (false negative). However, in a TV studio the light conditions are strongly controlled, and an occasional false negative is acceptable, as long as the re-initialization is successful since it can be done in the last estimated position. During the search, the robot does not drift but simply keeps the last estimated position.

Therefore, we employ the likelihood test on the estimated state $\overline{s}_t$ from both trackers, and declare a loss whenever $P(z_{col}|\overline{s}_t)$ decreases below a minimum threshold value P_{min}. This threshold is set as a percentage (e.g. $\leq 10\%$) of a reference value P_{ref}, initially set to the maximum likelihood value.

In order to provide adaptivity to variable postures (e.g. when the moderator turns on a side), as well as light or shading variations, P_{ref} itself is slowly adapted if the last likelihood $P(z_{col}|\overline{s}_{t-1})$ is comparable to P_{ref} (e.g. $\geq 60\%$). When a track loss occur, the Particle Filters are re-initialized with the diffure prior, until the subject becomes again visible and the likelihood increases above the threshold. This is also demonstrated in the second experiment of the next Section.

D. Robot controller

The control system we propose for the robot cameraman keeps the moderator in sight by translating the camera along the three directions (x, y, z), by keeping the error vector as small as possible while, at the same time, avoiding jittering and abrupt accelerations.

The error vector is computed both in frontal image coordinates

$$e_f \equiv [\overline{x} - x^*, \overline{y} - y^*]^T \tag{7}$$

with respect to the target image location (x^*, y^*), and in the depth direction

$$e_d \equiv \overline{d} - d^* \tag{8}$$

with respect to a *target depth* d^*.

We also notice that the choice of (x^*, y^*) is not limited to the image center, since in many cases (e.g. weather forecasts, documentary movies) the moderator should sometimes stand in a fixed, lateral position while talking about the main background scene, virtually rendered by the VR-TV engine. The image error (7) can be used in order to drive the robot in the respective directions through a standard Cartesian controller, with suitable safety thresholds in order to limit the acceleration values, as well as to keep the robot far from singular configurations during all of the tracking operation; this can be achieved since the operational space for the end-effector is reduced only to a translational motion, occurring within a relatively limited space with respect to the large robot size. The depth error (8) instead can be used in order to control the zoom of the main TV camera, and keep a constant size on the screen. We also note here that the tracking camera has no zoom capabilities, but keeps always a large FOV in order to keep the target in sight and avoid tracking loss.

IV. Experimental results

We tested the person tracking system live and on sequences obtained in a real TV studio, using the above described hardware and software architecture. In particular, a standard Desktop PC with $2.4GHz$ Intel Pentium IV and standard graphics hardware has been used for both camera acquisition and processing pipelines, running on Linux OS and communicating with the main server through a TCP/IP link.

The model has been obtained as shown in Fig. 4, and 100 particles have been used for tracking with both Particle Filters; the input images have a resolution of 640×480 pixels and are obtained from a Guppy Firewire camera. With this configuration, we achieved a speed around $20fps$, which we found more than sufficient for the robot controller in most VR-TV applications.

Fig. 6 shows some experimental results of the visual tracker, obtained in a real TV studio environment. The result of both camera trackers with respect to both modalities is shown by the main rectangular frame and the individual color patches. In the first sequence (first two rows), when another person enters the scene and interacts with the moderator, the motion images tend to merge and provide clutter motion noise around, that however is not sufficient to distract the tracker because of the multi-patch likelihoods and the use of dynamic fusion.

The tracker keeps track of the person during this sequence, despite the far position with respect to the camera, the poor lighting, as well as the clutter background due to the nearby moving person.

Another example is shown in the last two rows of Fig. 6, this time involving a moving subject that occasionally stands up and leaves the field of view. The tracker here employs the automatic track loss detection criterion mentioned in the previous Section, and we can see how the system automatically re-initializes when the person comes back in the field of view. The loss detection technique requires fine tuning of the thresholds as it can be seen that in one of the images a target loss is detected even when the target is in the screen. This occurs mainly due to sudden change of pose and inconsistent illumination arising from shadows. This is a false negative and is acceptable as re-initalization is easy rather than the tracker converging to a false target. The loss detection threshold adaptibilty needs to be made more robust in the future to cope with such scenarios.

As we can see, another important feature of our multi-patch approach is to be able to successfully cope with inaccuracies in the defined model: for example, when only the head and torso regions are selected from a picture where the moderator is sitting behind the desk, the system still works correctly when the person stands up and walks inside the studio, and vice-versa. This property is enforced by the data fusion methodology described in the previous Section.

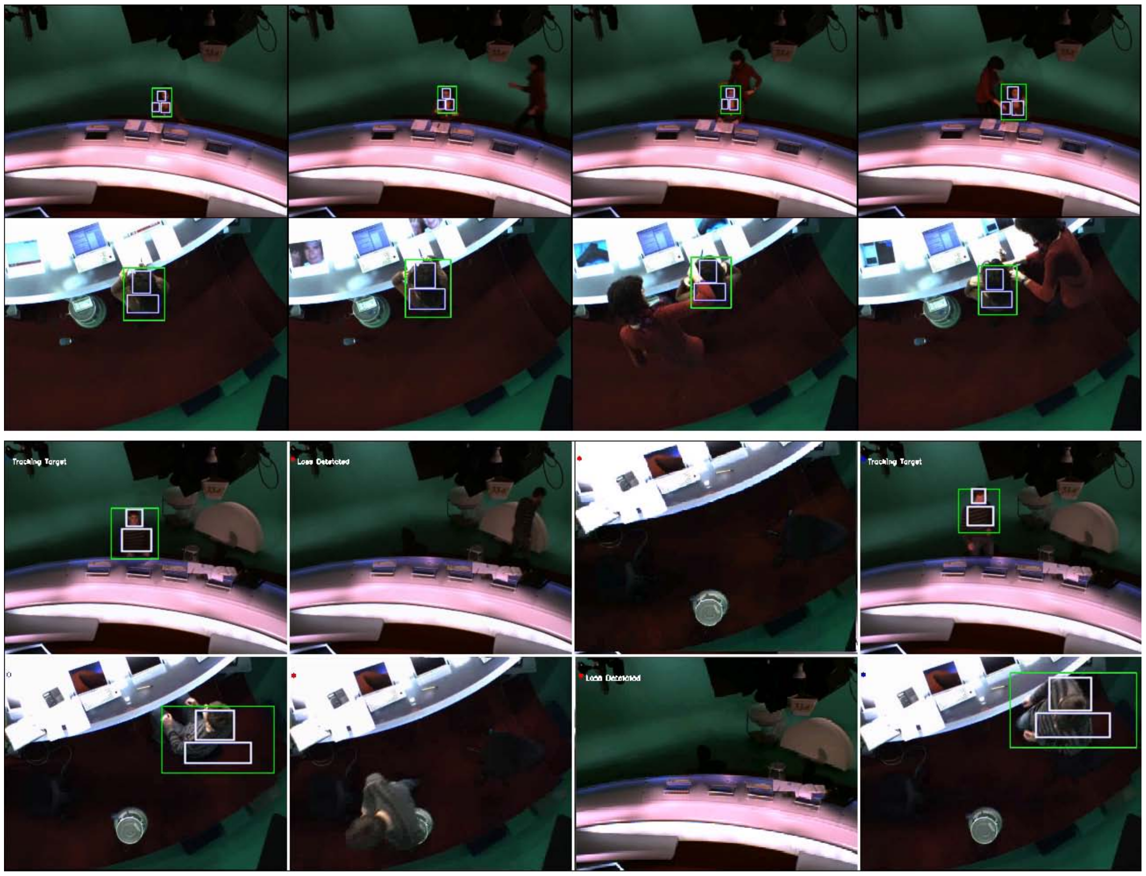

Fig. 6. Experimental results of the person tracking module. First and second rows: frames from the frontal camera, with color and motion processing result. Third and fourth row: another sequence, with a moving person and a different multi-patch color model; when the person is leaving the scene, a track loss is detected. Green rectangles: average 2D pose estimate; Blue rectangles: corresponding color patches.

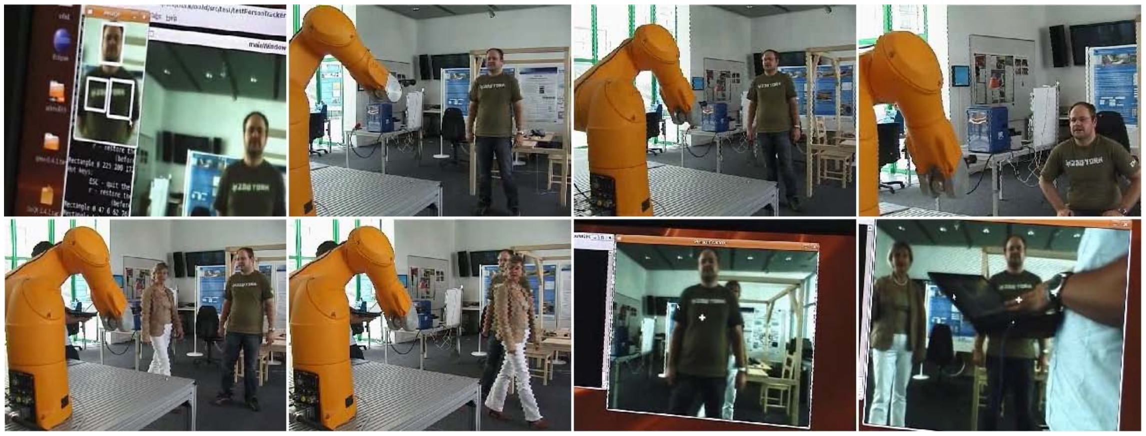

Fig. 7. Sequence with the robot controller in action. Top left: model acquisition. The sequence shows the system successfully handling mutual occlusions in a cluttered environment.

Overall, these results show how the integration of color and motion cues gives a stable and relatively precise result, that can be reliably used in order to steer the robot camera-man.

A prototype of the control system, involving an industrial 6dof robot arm, has been tested and successfully demonstrated in public; the snapshots in Fig. 7 show the robot following the person in presence of mutual occlusions, and a heavily cluttered background.

V. Conclusions and future developments

We presented a model-based visual tracking and robot control system for Virtual-Reality TV applications, that efficiently integrates multiple visual cues and multiple camera inputs in order to keep the moderator inside the main video field while freely moving, sitting or walking. The system provides a user-friendly GUI in order to specify a minimal model information for tracking a specific person, and requires a very small user intervention while at the same obtainining smooth position adjustments through the robot controller for the VR-TV rendering engine.

A current limitation of the system is that it allows freedom of movement for the moderator only within the single overhead camera field of view, which is sufficient when the field of view is wide enough to cover the whole floor area. If this is not the case, due to the modular nature of the tracker (Fig. 5) the overhead system can easily be scaled to an array of calibrated cameras, mounted in a regular grid that covers the whole area while keeping minimally overlapping fields of view. A simple input switching between cameras for tracking can then be implemented, whenever the moderator crosses two adjacent fields of view.

VI. Acknowledgements

The authors wish to express their acknowledgements to the RTL Television Studio Köln, Germany for providing the environment, and the pictures and live video sequences for our tracking system.

References

[1] I. Haritaoglu, D. Harwood, and L. S. Davis, "W4: A real time system for detecting and tracking people," in *CVPR '98: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. Washington, DC, USA: IEEE Computer Society, 1998, p. 962.

[2] N. T. Siebel and S. J. Maybank, "Fusion of multiple tracking algorithms for robust people tracking," in *ECCV '02: Proceedings of the 7th European Conference on Computer Vision-Part IV*. London, UK: Springer-Verlag, 2002, pp. 373–387.

[3] M. Isard and J. MacCormick, "Bramble: A bayesian multiple-blob tracker," in *ICCV*, 2001, pp. 34–41.

[4] C. R. Wren, A. Azarbayejani, T. Darrell, and A. Pentland, "Pfinder: Real-time tracking of the human body," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 7, pp. 780–785, 1997.

[5] K. Nummiaro, E. Koller-Meier, and L. J. V. Gool, "An adaptive color-based particle filter," *Image Vision Comput.*, vol. 21, no. 1, pp. 99–110, 2003.

[6] P. Pérez, C. Hue, J. Vermaak, and M. Gangnet, "Color-based probabilistic tracking," in *ECCV '02: Proceedings of the 7th European Conference on Computer Vision-Part I*. London, UK: Springer-Verlag, 2002, pp. 661–675.

[7] D. Comaniciu, V. Ramesh, and P. Meer, "Kernel-based object tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 5, pp. 564–575, 2003.

[8] G. R. Bradski and J. W. Davis, "Motion segmentation and pose recognition with motion history gradients," *Mach. Vision Appl.*, vol. 13, no. 3, pp. 174–184, 2002.

[9] G. Panin, C. Lenz, S. Nair, E. Roth, M. Wojtczyk, T. Friedlhuber, and A. Knoll, "A unifying software architecture for model-based visual tracking," in *IS&T/SPIE 20th Annual Symposium of Electronic Imaging*, San Jose, CA, January 2008.

[10] M. Isard and A. Blake, "Condensation – conditional density propagation for visual tracking," *International Journal of Computer Vision (IJCV)*, vol. 29, no. 1, pp. 5–28, 1998.

[11] Y. Bar-Shalom and X.-R. Li, *Multitarget-Multisensor Tracking: Principles and Techniques*. YBS Publishing, 1995.

[12] J. J. Clark and A. L. Yuille, *Data Fusion for Sensory Information Processing Systems*. Norwell, MA, USA: Kluwer Academic Publishers, 1990.