Proceeding of the 6th International Symposium on Mechatronics and its Applications (ISMA09), Sharjah, UAE, March 24-26, 2009.

A DISTRIBUTED AND SCALABLE PERSON TRACKING SYSTEM FOR ROBOTIC VISUAL SERVOING WITH 8 DOF IN VIRTUAL REALITY TV STUDIO AUTOMATION

Suraj Nair, Giorgio Panin, Thorsten Röder, Thomas Friedlhuber, Alois Knoll, *Member, IEEE*

Technische Universität München, Fakultät für Informatik
Boltzmannstrasse 3, 87452 Garching bei München (Germany)
{nair,panin,roeder,friedelhu,knoll}@in.tum.de

ABSTRACT

In this paper, a distributed and scalable person tracking system for visual servoing using Industrial Robot Arms 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 to drive the main broadcasting camera mounted on a large robotic arm connected with a pan tilt unit. The system consists of a person tracking system operating on the TV camera, which localizes the moderator in its field of view, and a overhead tracking system which localizes the moderator over the complete studio environment. The system is completely scalable in scenarios where a single scene is to be shot from multiple angles using multiple TV cameras. A common overhead tracking system monitors the moderator over the complete studio environment allowing the individual camera systems to initialize themselves and re-initialize in case of target loss. Application of the proposed tracking system to real-time VR-TV results in a robot cameraman, able to keep the moderator inside the screen with jitter-free viewpoint adjustments, as required by the VR scene rendering engine.

1. INTRODUCTION

Virtual TV studios (Fig. 1) have gained immense importance in the broadcasting area, due to the developments in computer graphics hardware and software and their capability to 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 or odometry), 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 robot and the VR 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 manually 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 movements available (zoom, *fly-by*, etc.), which on one hand increases the workspace of camera operation and also provide the 3D pose of the camera directly through the robot kinematics, but on the other hand also limits 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, with almost no need for human intervention.

For this purpose, in this paper we present a distributed and scalable tracking system, based on a previous work [1] with several improvements as described in the following.

The new system directly uses video input from the robot mounted TV camera, without the need for an additional Firewire device on top of it. Robustness of tracking has been improved by a novel integration of visual modalities. Additional control strategies (Normal Mode and Hold Angle Mode) have been developed, in order to control the robot for better visual effects. The 6dof Staübli Industrial Robot Arm to which a pan-tilt unit (2dof) is connected, provides overall 8dof to control the TV camera. A centralized communication engine using TCP/IP sockets has been developed, in order to provide efficient and reliable communication between the tracking system and the robot controller, also allowing the system to be scaled to multiple robots when a scene has to be shot from multiple views.

Finally, in order to obtain a reliable localization over the whole area, we employ a distributed system consisting of a person tracker using the robot-mounted TV camera, and a overhead camera tracker monitor the target position over the complete studio environment. The overhead system allows initializing the person tracker operating on the robot camera, as well as to re-initialize it in case of target loss. The overhead tracking result is also used in order to drive the auto-zoom and auto-focus properties of the TV camera (according to the estimated relative depth of the person).

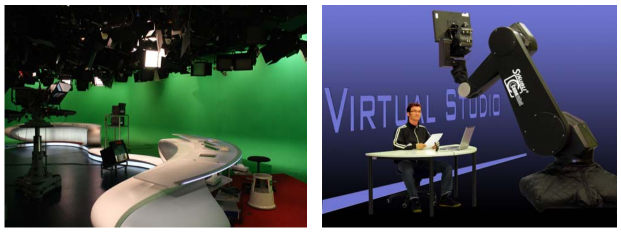

Figure 1: Left: A Virtual Reality TV Studio with manually operated TV Camera (*image courtesy: RTL Television Studio Köln, Germany*) Right: TV Camera operated by a Industrial Robot (6dof) together with a Pan Tilt Unit (2 dof)

The present paper is organized as follows: Sec. 2 briefly reviews the related state-of-the-art; Sec. 3 presents the system overview, and provides more details concerning the user interface for modeling (Sec. 3.2), the tracking methodology (Sec. 3.3) and the robot controller (Sec. 3.4). Experimental results are given in Sec. 4, and conclusions with proposed system developments are finally given in Sec. 5.

2. STATE-OF-THE-ART COMPARISON

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

Multiple people trackers [2, 3, 4], 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.

Therefore, many popular systems for single-target tracking are based on color histogram statistics [5, 6, 7, 8] and employ a base, pre-defined shape and appearance model during the whole task.

In particular, [7] uses a standard particle filter with color histogram likelihood with respect to a reference image of the target, while [6] 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 [8].

The person tracking system [5] 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.

By comparison, in our system the off-line modeling part is kept to a minimum extent, while at the same time retaining a more detailed information than a single color patch can provide.

3. THE INTEGRATED PERSON TRACKING SYSTEM

As stated in the Sec 1, the goal of our system is to robustly provide real-time information about the current position of the moderator in order to control the 3D position of the robot camera keeping the target in a desired ROI.

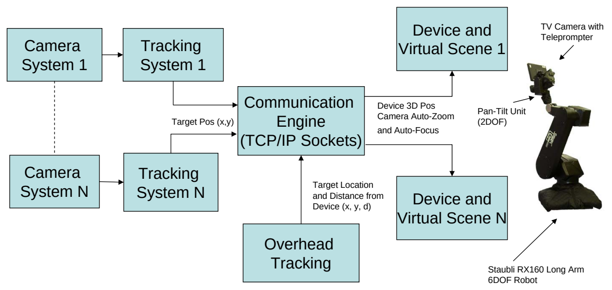
Camera System 1
Tracking System 1
Camera System N
Tracking System N
Target Pos (x,y)
Communication Engine (TCP/IP Sockets)
Device 3D Pos Camera Auto-Zoom and Auto-Focus
Device and Virtual Scene 1
Device and Virtual Scene N
Target Location and Distance from Device (x, y, d)
Overhead Tracking
TV Camera with Teleprompter
Pan-Tilt Unit (2DOF)
Staubli RX160 Long Arm 6DOF Robot

Figure 2: Block diagram of the system architecture

Fig. 1 (Right) shows the overall setup in the VR-TV scenario using a robot; the corresponding system architecture is depicted in Fig. 2, and hereafter described.

3.1. System overview

The hardware setup consists of single or multiple robots arms along with a pan-tilt unit, on which the main TV camera is mounted forming a camera system with 8 DOF. There exists a single overhead tracking system, managing all of the related cameras.

The frontal person tracker uses the TV camera in order to localize the target in its 2D field of view, and also sends control data to drive the robot arm on which the TV camera is mounted and used to control the virtual reality engine, that renders the 3D scene according to the current pose of the TV camera.

The overhead system tracks the moderator over the studio environment and computes the location and distance of the target from a *world* point, with respect to which all robots are calibrated; this system, as already mentioned, also computes the distance of the target from the robot in order to drive the auto-zoom and auto-focus of the frontal camera, and re-initializes in case of target loss.

In particular, the visual system uses color cues, assisted by background segmentation (only for the overhead tracker), through dynamic data fusion and Bayesian filtering, in order to improve robustness. The robot-mounted camera tracker estimates the horizontal and vertical (x, y) coordinates of the moderator, while the overhead cameras estimate his/her location (x, y) on the floor, as well as the distance d from the robot, and all together they provide output visualization of the estimated 3D position.

The estimation result is sent as a feedback to 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, offering a number of advantages over more conventional kalman filtering techniques. In fact, particle filters more robustly deal with multi-modal likelihoods due to clutter background, as well as other people that may occasionally interact with the moderator.

Concerning computational resources, the tracking software for each camera system runs on a seperate PC and uses the TV camera picture through a frame grabber. The overhead tracker instead uses a Firewire camera with a wider angle lens for covering the Tv studio scene. The robot controller runs on a separate machine with real-time OS capabilities.

All systems communicate with the devices and the virtual reality engine over an Ethernet network, through the communication engine using TCP/IP sockets. Each tracking system is identified by a unique IP and port. In the following sections the system is described in more detail.

3.2. Graphical user interface for modeling

In order to realize a simple and robust tracking system, we employ 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 information are obtained (Fig. 3):

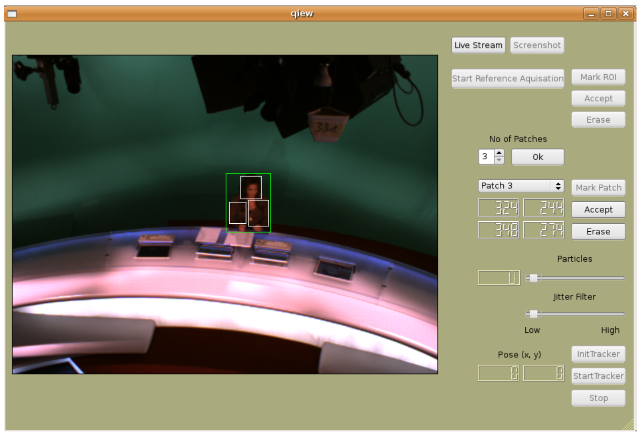

Figure 3: User interface and selected model patches

- *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, 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.

3.3. 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. 4 describes the complete pipeline for both camera types.

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. In the current system we have improvised on [1] by using only color information for the person tracker since it is installed on a moving device and using both color information and background information for the overhead tracker since the overhead camera is fixed. Considering a single person tracker and a overhead tracker, the 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). 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.

3.3.1. Image pre-processing

The sensor data for the person tracker is obtained from the TV camera in a raw PAL format through a grabber. The image is pre-processed by performming RGB to HSV conversion z_{col} for the color-based likelihood. The sensor data for the overhead tracker is obtained from the firewire camera in a raw RGB format and undergoes a static fusion at preprocessing stage by performing background segmentation and then by a RGB to HSV conversion z_{col} for the color-based likelihood.

3.3.2. 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

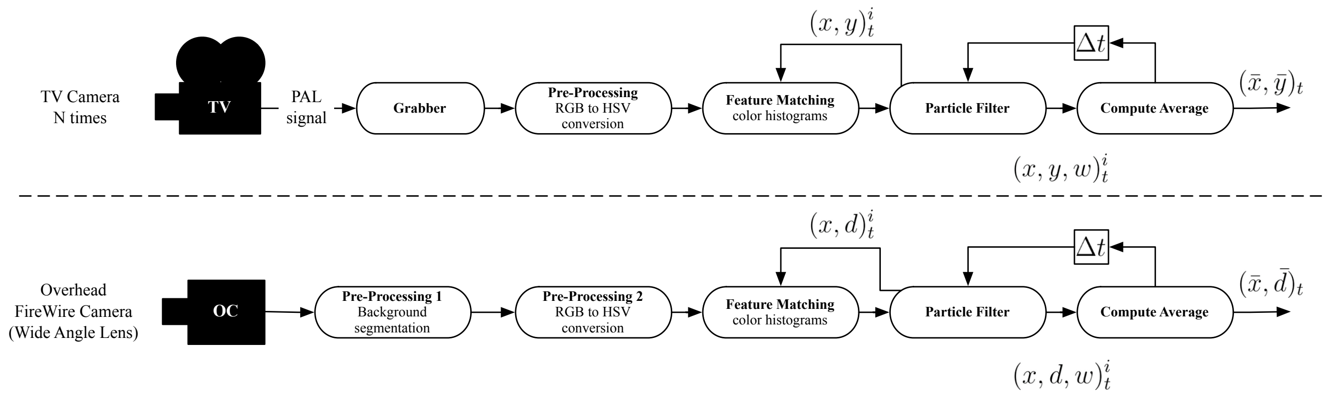

Figure 4: Tracking pipeline and estimated variables for the robot controller

brownian motion 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 $P(z_{col}|s^i)_n$ for the person tracker and $P(z_{col}|s_{oc}^i))$ for the overhead tracker.

3.3.3. 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 histogram $q_p\left(s_t^i\right)$, that is compared with the reference one q_p^* through the Bhattacharyya coefficient [7]

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

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

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

with given covariance λ_{col}

3.3.4. Computing the estimated state

The average state $\overline{s}_t$

$$\overline{s}_t = \sum_i w_t^i s_t^i \tag{4}$$

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).

3.3.5. Loss detection and handover between person and overhead tracker

One of the most important features 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 using the overhead tracking system. Whenever a target loss is detected by the robot camera the system immediately switches to the overhead tracker and uses this data to move the robot in order to bring the target in field of view and continue with its own tracker. The overhead tracker is also used to initialize the robot system when its has to go live from a random position.

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 to. 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.

Therefore, we employ the likelihood test on the estimated state $\overline{s}_t$ from both trackers, and declare a loss when-

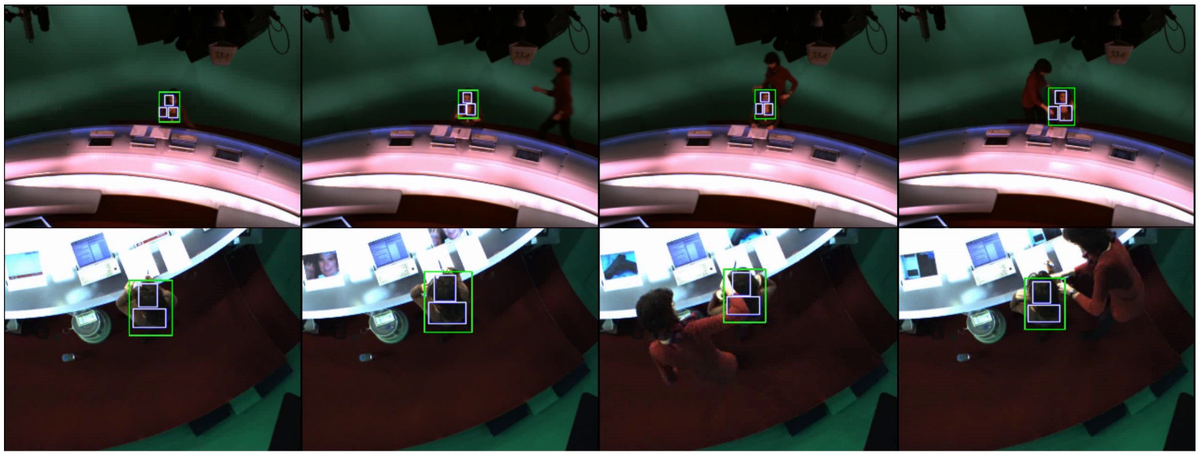

Figure 5: Experimental results. First and second rows: frontal camera, with color processing result. Green rectangles: average 2D pose estimate; Blue rectangles: corresponding color patches.

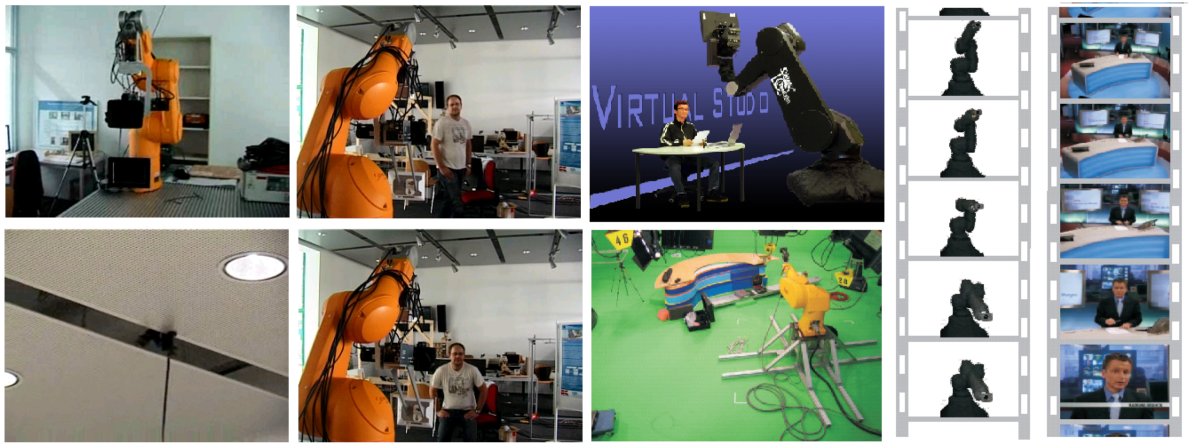

Figure 6: Sequence with the robot controller in action.

ever $P(z_{col}|\overline{s}_t)$ 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 diffuse prior, until the subject becomes again visible and the likelihood increases above the threshold.

3.4. Robot controller

To support many different types of camera systems, it is necessary to have a well defined methodology that generates device trajectory data out of tracking data. The person tracking systems return the 2D information of the target in image space coordinates and the global overhead tracking sytem sends 2D information along with the distance estimation. In order to keep the target in a predefined ROI it is neccessary to generate the relative motion parameters for the corresponding robot system. This is achieved by using these pixel-level information from the tracking system and converting them to 3D movement commands in world space.

For this conversion different additional parameters have to be considered namely:

Region of interest It is the deisred area in the image space of each camera system, where the target should be held, e.g. in weather broadcasts the target appears usually on the right hand side of the scene.

Balanced speed The speed of the robot is specified for X, Y, Pan and Tilt as an absolute percentage ranging from $[-100\%; 0\%; 100\%]$, meaning move with full speed away from the target, no motion and move with full

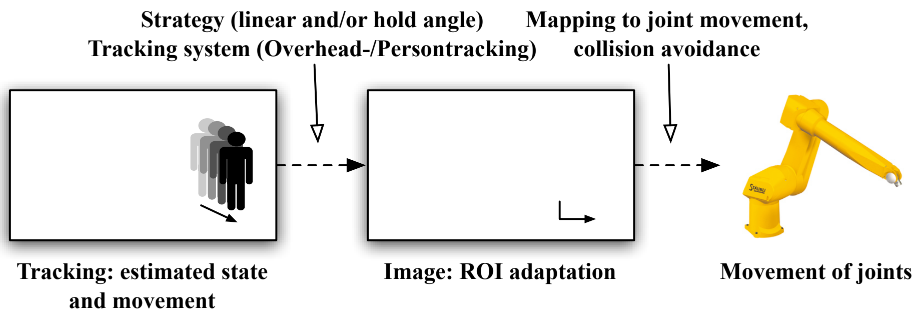
Strategy (linear and/or hold angle)
Tracking system (Overhead-/Persontracking)
Mapping to joint movement,
collision avoidance
Tracking: estimated state and movement
Image: ROI adaptation
Movement of joints

Figure 7: Robot control methodology

speed towards the estimated target position. If the target is outside the ROI, then this speed is used by the camera system in order to get the target back into the ROI. The actual speed used also depends on the distance between the target and the ROI. The effective speed is proportional to the calculated distance providing smoother motion properties.

As shown Fig. 7 the 3D cartesian movement is computed out of the X, Y, Pan, Tilt speeds. We propose two different operation modes for the joint control:

- Normal mode: The movement of the robot is limited by linear motion in the X and Y direction and by angular motion for Pan and Tilt in order to get the target back into the desired ROI.

- Hold-angle mode: The movement of the robot is done in 2 phases: In the 1st phase, the robot uses only Pan and Tilt to bring the target back into the ROI and then in the 2nd phase, it uses linear X and Y motion to compensate to hold a predefined viewing angle of the camera system.

  This algorithm moves the robot angular in Pan and Tilt to follow the target. But the linear axis X and Y try to move in a way, that a special angle of the camera system is maintained indirectly. Of course the X and Y axis cannot affect any angles, but if they move, then the target information changes and Pan and Tilt axis try again to correct this. So in the end X and Y axis are able to control the Pan and Tilt indirectly, until they achieve a camera system specific angle.

4. EXPERIMENTAL RESULTS

The system was evaluated by testing in real exisiting TV stduios ready for VR. For this purpose standard Desktop PCs with $2.4GHz$ Intel Pentium IV and standard graphics hardware have been used to realize the tracking for each camera system and the overhead tracking system, running on Linux OS and communicating with the central comminication engine using TCP/IP sockets.

Fig. 5 shows some experimental results of the visual tracker,obtained in a real TV studio environment. The results of both the person tracker and the overhead tracker with is shown by the main rectangular frame and the individual color patches. The tracker keeps good 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. The performance of the overhead tracker is improved with the inclusion of background segmentation. The robustness of loss detetcion has improved by fine tuning the parameters with respect to the lighting conditions. The usage of the TV camera signal has removed the dependencies of an additional camera requiring offset correction used in [1]. The automatic switching between the person tracker and the overhead tracker in case of target loss has improved the robustness of the system. With 100 particles for the particle filter and image resolution of 640×480 pixels, we achieved a speed around $20fps$, which we found more than sufficient for the robot controller which request a result every $200ms$.

Fig. 6 demonstrates the current improvements in the robot system. The first column illustrates a test robot with a pan-tilt unit and a TV camera and also the overehead tracker camera. The second column illustrates the robot tracking the target while the target is moving. The third column presents the robot used in the actual TV studio. Then last column illustrates the different moves of the robot along with the VR scene. The newly developed control techniques, Normal Mode and Hold Angle Mode, provide an elegant visual effect in the rendered scene.

5. CONCLUSIONS AND FUTURE DEVELOPMENTS

We presented a distributed and scalable model-based visual tracking system for robotic visual servoing with 8 dof (degrees of freedom) in Virtual-Reality TV applications. We have improved the robustness of the current system using new visual modalties. The robot control has improved by the inclusion of new control methodolgies. The centralized communication engine provides realivle communication in the distributed system also allowing the system to be scaled.

We consider to improve the overhead tracking by using an grid of cameras with robust switching in order to support large sized studio environments. The overhead firewire cameras will be replaced by TV cameras with wide angle lens for better picture quality during tracking.

6. 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.

7. REFERENCES

[1] S. Nair, G. Panin, M. Wojtczyk, C. Lenz, T. Friedelhuber, and A. Knoll, "A multi-camera person tracking system for robotic applications in virtual reality tv studio," in *Proceedings of the 17th IEEE/RSJ International Conference on Intelligent Robots and Systems 2008*. IEEE, Sep. 2008.

[2] 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.

[3] 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.

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

[5] 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.

[6] 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.

[7] 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.

[8] 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.

[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.