An Experimental Framework for Evaluating PTZ Tracking Algorithms

Pietro Salvagnini[1], Marco Cristani[1,2], Alessio Del Bue[1], and Vittorio Murino[1,2]

[1] Istituto Italiano di Tecnologia (IIT), Genova, Italy
[2] Computer Science Department, University of Verona, Verona, Italy
{pietro.salvagnini,marco.cristani,alessio.delbue,vittorio.murino}@iit.it

Abstract. PTZ (Pan-Tilt-Zoom) cameras are powerful devices in video surveillance applications, because they offer both wide area coverage and highly detailed images in a single device. Tracking with a PTZ camera is a closed loop procedure that involves computer vision algorithms and control strategies, both crucial in developing an effective working system. In this work, we propose a novel experimental framework that allows to evaluate image tracking algorithms in controlled and repeatable scenarios, combining the PTZ camera with a calibrated projector screen on which we can play different tracking situations. We applied such setup to compare two different tracking algorithms, a kernel-based (mean-shift) tracking and a particle filter, opportunely tuned to fit with a PTZ camera. As shown in the experiments, our system allows to finely investigate pros and cons of each algorithm.

1 Introduction

This paper proposes a platform to evaluate different single-target tracking algorithms for PTZ cameras. Our aim is toward *repeatability*, which is complex in such case because PTZ cameras "see" a different scenario given the choice of pan, tilt and zoom parameters, and such parameters are differently set by the diverse tracking algorithms taken into account. This means that there cannot be a unique video benchmark which allows a genuine global testing. The proposed system provides the same scenario to the PTZ camera as many times as desired, in order to test different tracking algorithms. The core idea consists in projecting a video containing the target on a screen in front of the camera. In this way, we are aware at each instant about the position of the target on the projector screen. This setup makes possible to compare the localization error and other metrics of the target during tracking. The setup, shown in figure 1, is composed by 3 different steps: (1) camera calibration, (2)implementation of the PTZ tracking algorithm, (3) projection of the video with the target on the wall and comparison of the different tracking results with the ground-truth (GT).

The paper is organized as follow. In section 2 the state-of-the-art is presented. In section 3 the whole system will be presented, describing the 3 parts introduced above. A quantitative evaluation of the effectiveness of our framework so as the comparison between two particular tracking algorithms are provided in section

J.L. Crowley, B. Draper, and M. Thonnat (Eds.): ICVS 2011, LNCS 6962, pp. 81–90, 2011.

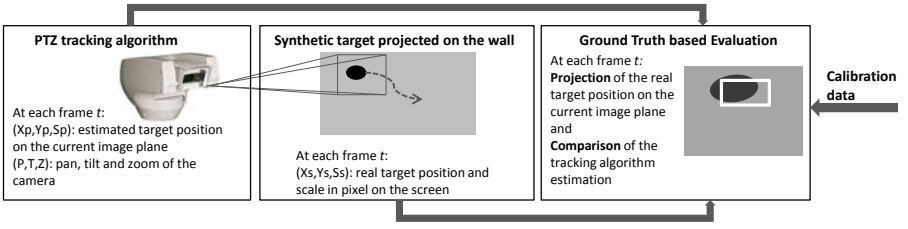

Fig. 1. The whole system

4, whereas in section 5 final considerations about the importance of our contribution and a possible future work are discussed.

2 State of the Art

PTZ cameras can be exploited and studied from different points of view. In our work, we are interested in the geometrical modeling for the PTZ camera control and target projection, as related to visual object tracking. Hence, this section presents the most relevant state of the art regarding four different aspects, PTZ camera modeling, visual object tracking in general and using a PTZ camera, and evaluation of tracking algorithms.

PTZ Geometry. The problem of calibrating a PTZ camera has been addressed in many papers with different methods and levels of approximation. One of the most important work on self-calibration of rotating and zooming camera is the paper of Agapito et al. [1]. It considers in particular how the changes of the zoom and the settings of the focus affect the intrinsic parameters. Other works are related to estimation of a geometrical model for a rotating camera, linking the rotation angles to the camera position in the 3D world. In [7], pan and tilt rotations are modeled as occurring around arbitrary camera axes in space, and the relative position between the axis is estimated.

Visual Object Tracking. An overview on different techniques can be found in [15]. The Bayesian recipe is one of the most widely used framework for tracking, that considers both an *a priori* information on the target (dynamical model), and the information from the current image acquired from the camera (observation model). The choice of the dynamical and the observation models characterizes each different algorithm together with the approximation of the evolution of the probability density function that describes the target state. Nowadays, particle filters are the most employed techniques; here we considered the classical filtering approach of Condensation [10]. A different philosophy sees the tracking as a mode seeking procedure, here represented by the Mean Shift tracker [6], and in particular by the *CamShift* approach [3]. The target model is an histogram and the area of the image that exhibits the most similar histogram is searched at each iteration. This algorithm proposes an extremely efficient technique to minimize the Bhattacharyya distance between histograms. In the last years both these

algorithms have been extended and improved, like in [4] for particle filter or [5] for mean-shift.

PTZ Cameras in Video Surveillance Systems. Typically, in video surveillance settings, a master-slave architecture is adopted using a wide zoom fixed camera (master) and a PTZ camera (slave) that is moved to highlight the relevant subjects of interest in the scene, as in [9]. When using multi-camera architectures, calibration between cameras is a key element and usually requires considerable effort. For this reason, methods that only require weak calibration or implements automatic calibration algorithms (e.g. [2]) are the most popular ones. Among them, two recent works are [13] and [14]. In the former, the scenario consists of a single PTZ camera that tracks a moving target which lies on the floor, and the focus of the work is on the control part of the process: the choice of the camera position at each step is formulated as an optimization problem. In the latter, the camera tracks the upper-body of a person that walks in a room with a fuzzy algorithm. It also compares the results obtained with other tracking algorithms, but such a comparison is performed off-line, using the frames obtained from their PTZ tracking algorithm. As a result, only the visual tracking algorithms are evaluated and not the performance of the system that also accounts for the camera motion.

Evaluation of Tracking Algorithms. The evaluation of tracking algorithms is often related to a specific application, for example surveillance in [8] or low frame rate areal imagery [12] or for a specific category of algorithms, e.g. template-based in [11]. In this work we will adopt similar metrics for the evaluation, but apply them to different, unexplored PTZ scenario.

3 Methodology

In this section, we present the different components of the whole system: the calibration of the PTZ camera, the implementation of the two tracking algorithms, and finally the performance evaluation testbed.

3.1 PTZ Camera Calibration

We adapt a standard pinhole camera model to a specific PTZ camera, shown in Fig. 1, used for our evaluation.

Intrinsic parameters. First, we calibrate the intrinsic or internal parameters, according to the pinhole model with one coefficient for the radial distortion.

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_d \\ y_d \\ 1 \end{bmatrix} \quad \begin{cases} x_d = \dfrac{x_r}{z_r}(1 + kr^2) \\ \\ y_d = \dfrac{y_r}{z_r}(1 + kr^2) \end{cases} \qquad r^2 = \frac{x_r^2 + y_r^2}{z_r^2} \tag{1}$$

where (x_r, y_r, z_r) are the coordinates in the optical center reference system, (x_d, y_d) are the coordinates after the distortion due to the camera lens and

(u, v) are the pixel coordinates on the image plane. The intrinsic parameters (f_x, f_y, c_x, c_y, k) are estimated for each zoom level between 1x and 20x, with a step of 1x.

Rotation Axis Model. The PTZ camera can rotate around two axes that are not aligned with the camera reference system. The rotation axes do not intersect in any points and do not pass through the optical center, so the correct misalignments should be computed to avoid approximations. Let (ϕ_C^i, θ_C^i) indicate the camera pan and tilt angles as measured by the motor encoder and (x_r^i, y_r^i, z_r^i) the coordinates of a point in the optical center reference system in that pose. When the camera is rotating from the initial pose $\phi_C^0 = 0$ and $\theta_C^0 = 0$ to a new pose (ϕ_C^1, θ_C^1), the transformation between the two reference systems can be described by the composition of two rotations, each of them around a translated axis:

$$\begin{bmatrix} x_r^0 \\ y_r^0 \\ z_r^0 \\ 1 \end{bmatrix} = R_0 T_0 \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} \quad \begin{bmatrix} x_r^1 \\ y_r^1 \\ z_r^1 \\ 1 \end{bmatrix} = T_\theta^{-1} R_x(\theta_C^1) T_\theta T_\phi^{-1} R_y(\phi_C^1) T_\phi \begin{bmatrix} x_r^0 \\ y_r^0 \\ z_r^0 \\ 1 \end{bmatrix} \tag{2}$$

$$R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c(\phi) & -s(\phi) & 0 \\ 0 & s(\phi) & c(\phi) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, R_y(\theta) = \begin{bmatrix} c(\phi) & 0 & -s(\phi) & 0 \\ 0 & 1 & 0 & 0 \\ s(\phi) & 0 & c(\phi) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, T_{\phi,\theta} = \begin{bmatrix} 1 & 0 & 0 & t_x^{\phi,\theta} \\ 0 & 1 & 0 & t_y^{\phi,\theta} \\ 0 & 0 & 1 & t_z^{\phi,\theta} \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

where $s(\cdot)$ and $c(\cdot)$ stand for sine and cosine trigonometric functions. R_0 and T_0 represent the roto-translation from the screen to the camera given the initial position (ϕ_C^0, θ_C^0). Finally we indicate with (x_r^0, y_r^0, z_r^0) in the optical center reference system when in the initial position. These parameters are estimated at the beginning of each experiment, because they depend on the relative position between the camera and the screen. This is achieved by projecting a checkerboard of known size on the screen. On the contrary, the translation vectors $[t_x^\theta, t_y^\theta, t_z^\theta]$ and $[t_x^\phi, t_y^\phi, t_z^\phi]$ are fixed and can be estimated by minimizing the projection over a set of checkerboard images. Please, note that they depend on the zoom values, actually, the focal length increases as the zoom increases and the position of the optical center with respect to the rotation axis also varies. For this reason the calibration of such parameters must be performed for different zoom values, as for the intrinsic parameters. Note also that (2) could be used to go from the coordinates (x_r^2, y_r^2, z_r^2) in a given position (ϕ_C^2, θ_C^2) to coordinates in the initial configuration (ϕ_C^0, θ_C^0) by simply inverting the matrix $T_\theta^{-1} R_x(\theta) T_\theta T_\phi^{-1} R_y(\phi) T_\phi$. Finally, using the transformation from (x_r^2, y_r^2, z_r^2) to (x_r^0, y_r^0, z_r^0) with the one from (x_r^0, y_r^0, z_r^0) to (x_r^1, y_r^1, z_r^1) it is possible to express a general transformation from two camera poses (ϕ_C^2, θ_C^2) and (ϕ_C^1, θ_C^1).

3.2 Tracking Algorithms for PTZ Camera

Two different tracking algorithms have been implemented for the PTZ camera. They are based on two well-known algorithms: the Camshift proposed in [3],

and the Particle Filter of [10]. They represent baselines of two different tracking philosophies here embedded in a PTZ scenario; this required to handle the position displacement and size variation of the target on the image plane due to the camera movements, as well as the control of the camera movements. In Algorithm 1, we present a general scheme for tracking using a PTZ camera, and later we will provide the details about the two different implementations. One of the crucial point of this scheme is that from frame to frame, the target is tracked using spheric coordinates, that is azimuth ϕ_T^t and elevation θ_T^t with respect to the reference coordinate system in the initial camera pose $\mathbf{C}_0 = (\phi_C^0, \theta_C^0, \mathcal{Z}_C^0)$. This is achieved in two steps: first, the estimated bounding box (BB) vertices are transformed in the initial camera pose reference system according to (2), then the spheric coordinates are computed knowing the zoom and focal length:

$$\begin{cases} x_w = f_x(\mathcal{Z}) \tan(\phi_T) \\ y_w = f_y(\mathcal{Z}) \dfrac{\tan(\theta_T)}{\cos(\phi_T)} \end{cases} \quad \begin{cases} \phi_T = \tan^{-1}\left(\dfrac{x_w}{f_x(\mathcal{Z})}\right) \\ \theta_T = \tan^{-1}\left(\dfrac{y_w \cos(\phi_T)}{f_y(\mathcal{Z})}\right) \end{cases} \tag{3}$$

In our algorithm, unlike similar previous works, at each frame, we assign to the camera a new speed and not a new position, in this way, we can produce a

Algorithm 1. A generic tracking algorithm for a PTZ camera

1: **Initialization** $t = 0$

- Move the camera to the starting point $\mathbf{C}_0 = (\phi_C^0, \theta_C^0, \mathcal{Z}_C^0)$
- Acquire the first frame from the camera
- Manually select the target Bounding Box:(c_x, c_y, l_x, l_y)
- Calculate the target model: *(depends on the adopted algorithm)*
- Transform the target position into the spheric coordinates $(c_x, c_y, l_x, l_y) \rightarrow (\phi_T^0, \theta_T^0, 1)$, the third coordinate representing the target scale, with respect to the initial zoom value.

2: **for** $t = 1$ to T **do**

3:
- Receive the current frame I_t from the camera and the camera configuration: $C^t = (\phi_C^t, \theta_C^t, \mathcal{Z}_C^t)$ from which it was captured
- Transform the previous target estimation from polar coordinate to the current view $(\phi_T^{t-1}, \theta_T^{t-1}, \mathcal{Z}_T^{t-1}) \rightarrow (\hat{c}_x^t, \hat{c}_y^t, \hat{l}_x^t, \hat{l}_y^t)$;
- Perform the tracking *(depends on the adopted algorithm)* starting from the initial guess $(\hat{c}_x^t, \hat{c}_y^t, \hat{l}_x^t, \hat{l}_y^t)$ and obtaining the new estimation $(c_x^t, c_y^t, l_x^t, l_y^t)$;
- Transform the new target estimation into polar coordinate considering the current camera position $(c_x^t, c_y^t, l_x^t, l_y^t) \rightarrow (\phi_T^t, \theta_T^t, \mathcal{Z}_T^t)$;
- Set the new camera speed and the zoom values, *(depends on the adopted control strategy)*, for both :

$$v_\phi^t = f_\phi(\phi_C^t, \hat{\phi_T^t}), \quad v_\theta^t = f_\theta(\theta_C^t, \hat{\theta_T^t}), \quad \mathcal{Z}_C^t = f_{\mathcal{Z}}(\cdot)$$

4: **end for**

smoother trajectory for the camera and the resulting video is more easily usable by a human observer.

CamShift. The target model is a 16-bin hue histogram and the target is represented as a rectangle whose sides are parallel to the image plane axis. CamShift (CS) algorithm estimates the position and scale of the rectangle at each frame. The camera control, basically proportional to the error, is set as follows:

$$\begin{aligned} v_\phi^t &= \lambda_\phi(k_\phi \frac{\hat{\phi}_T^t - \phi_C^t}{T_s}) + (1-\lambda_\phi)v_\phi^{t-1}, \quad v_\theta^t = \lambda_\theta(k_\theta \frac{\hat{\theta}_T^t - \theta_C^t}{T_s}) + (1-\lambda_\theta)v_\theta^{t-1} \\ \mathcal{Z}_C^t &= \lambda_\mathcal{Z}\mathcal{Z}_{opt}^t + (1-\lambda_\mathcal{Z})\mathcal{Z}_C^{t-1} \end{aligned} \tag{4}$$

where $\mathcal{Z}_{opt}^t$ is computed according to the estimated target size. Given the target spheric coordinates we can measure its horizontal and vertical angular extension, $\Delta\phi$ and $\Delta\theta$, and set the zoom adequately:

$$\mathcal{Z}_{opt}^t = \min\left(\frac{\tan(\Delta\phi_0/2)}{\tan(k_\mathcal{Z}\Delta\phi/2)}, \frac{\tan(\Delta\theta_0/2)}{\tan(k_\mathcal{Z}\Delta\theta/2)}\right) \tag{5}$$

where $\Delta\phi_0$ and $\Delta\theta_0$ are the fields of view at zoom 1x and $k_\mathcal{Z}$ expresses the desired ratio between the camera field of view and the object angular extension.

Particle Filter. We implemented a Particle Filter (PF) tracker that uses the same observation model as CS, the histogram on the hue values. The state x^j of each particle j has 4 dimensions, 2 for the position and 2 for the lengths of the rectangle. At each iteration t, the particle are sampled from a Gaussian distribution $\mathcal{N}(x_{t-1}^j, \Sigma)$.

To avoid the ambiguity on the target scale in case of a uniform target we also consider an external frame around the target BB and combine two histogram distances. Let h_T the histogram of the target, h_{int}^j the histogram of the region inside the candidate, and h_{ext}^j the histogram of the region external to the j candidate. The best candidate should have a small distance for the internal histogram $d(h_{int}^j, h_T)$ and a large distance for the external histogram $d(h_{ext}^j, h_T)$, where $d(h_1, h_2)$ is the Bhattacharyya distance. The weights w_i of the particles should be proportional to the likelihood $p(h_{int}^j, h_{ext}^j|x^j)$, so we could factorize and exploit the log-likelihood formulation:

$$w_i \propto p(h_{int}^j, h_{ext}^j|x^j) \propto e^{l(h_{int}^j|x^j)}e^{l(h_{ext}^j|x^j)} = e^{-(d(h_{int}^j,h_T))^2}e^{-(1-d(h_{ext}^j,h_T))^2}$$

At each iteration t, the set of N samples and their weights $\{x_t^j, w_t^j\}$ are used to set the camera control commands. The speeds are set again with (4), where $(\hat{\phi}_T^t, \hat{\theta}_T^t)$ are obtained from the particle with the highest weight (MAP criterion). Differently from (5), the $\mathcal{Z}_{opt}^t$ is chosen considering all the particles in order to keep all of them in the field of view of the camera. As shown in Sect. 4, this choice is important since it will enhance the tracker robustness.

3.3 Performance Evaluation

The camera is placed in front of a projector screen. Before starting a video, the extrinsic parameters R_0 and T_0 are estimated for the camera in the initial configuration $C^0 = (\phi_C^0, \theta_C^0, \mathcal{Z}_C^0)$ as explained above. At each iteration of the tracking algorithm, the following 3 values are saved for the next comparison with the GT: (1) the estimated target position on the current image plane $(c_x^t, c_y^t; l_x^t, l_y^t)$, (2) the current pose of the camera $C^t = (\phi_C^t \theta_C^t, \mathcal{Z}_C^t)$, (3) an absolute timestamp T_r^t. After that, in an off-line stage, the four vertices of the bounding box at time t are projected on the current image plane according to the camera pose, using (1) and (2), and compared with the tracker estimation at the same time. Obviously, this requires a quite precise synchronization between the ground-truth data and the actual tracking data, when they are stored during the tests. We indicate with T the number of frames in the sequence, collected by the tracker, and with T_c the number of frames before the target is lost (i.e., when it is no more recovered before of the end of the sequence). Given the target GT and the tracker estimation we use five criteria to evaluate the performances:

- the mean ratio between the estimated area $\|\mathcal{A}_{est}\|$ and the GT area $\|\mathcal{A}_{GT}\|$ over the valid frames: $r_{\mathcal{A}}^T$;
- the mean distance between the GT and the estimated centers (normalized on the target diagonal) over the valid frames: d_{ct};
- the rule $\dfrac{\|\mathcal{A}_{GT} \cap \mathcal{A}_{est}\|}{\|\mathcal{A}_{GT} \cup \mathcal{A}_{est}\|} \geq \dfrac{1}{2}$ to establish if the target is tracked properly, r_c is the percentage of correctly tracked frames over the valid frames;
- the mean ratio between the target area $\|\mathcal{A}_{GT}\|$ and the image area $\|\mathcal{A}_i\|$: $r_{\mathcal{A}}^i$;
- the mean distance between the GT target and the image center: d_{ci}.

The first three parameters evaluate the accuracy of the algorithms in tracking the target, while the last two evaluate the ability of the system to keep it in the center of the field of view and at the desired dimension on the screen.

4 Experiments

The system described above has been implemented in C++, using OpenCV functions for most of the vision algorithms. It works in real-time on a laptop, Intel Core 2 Duo CPU 2.8 GHz, 3.48 GB RAM. The projector is a commercial one, with resolution 1280×1024. The PTZ camera is an Ulisse Compact by Videotec, an analog camera, PAL format, whose pan, tilt and zoom are controlled through a serial port. The intrinsic parameters have been computed as in Sect. 3 and interpolated to get the intermediate values. The parameters used in the experiments are the following: $[t_x^\phi, t_y^\phi, t_z^\phi] = [50, 0, 180 - 21\mathcal{Z}]$, $[t_x^\theta, t_y^\theta, t_z^\theta] = [0, 60, -40 - 21\mathcal{Z}]$, for the camera model, $k_\phi = 0.3$, $k_\theta = 0.3$, $k_\mathcal{Z} = 5$, $\lambda_\phi = 0.7$, $\lambda_\theta = 0.7$, $\lambda_\mathcal{Z} = 0.4$ for the control part. The CS parameters are set to the default values $V_{min} = 10$, $S_{min} = 30$, $V_{max} = 256$, and the sampling time is set to $T_s = 0.1$. For the PF we used 400 particles, with variance $\Sigma = diag(25, 25, 2, 2)$, a 16 bin hue histogram

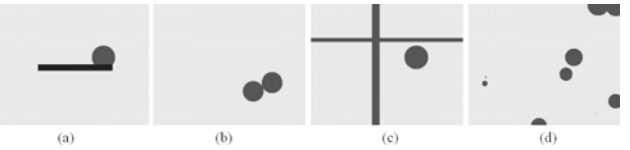

Fig. 2. The synthetic videos used in the comparison. (a): a black rectangle occludes the target; (b): the targets splits into two identical target, then one disappears;(c): a red-shape is in the background; (d) some dots similar to the targets appear and disappear in the background.

Table 1. Comparison between the CS and the PF trackers on a set of videos in which a synthetic target is projected in different scenarios

video	tracker	T	*end*	T_c	r_c	$r^T_{\mathcal{A}}$	d_{ct}	$r^i_{\mathcal{A}}$	d_{ci}	fps
basic	CS	165	yes	165	100.00	0.815	0.070	0.070	36.361	2.86
	PF	158	yes	158	82.91	1.306	0.106	0.017	20.239	3.77
occlusion	CS	998	yes	998	65.31	0.822	0.151	0.080	107.776	11.20
	PF	530	yes	530	66.26	0.826	0.183	0.020	40.918	5.90
splitting	CS	171	no	76	20.48	0.841	0.175	0.045	119.826	9.60
	PF	313	yes	313	28.74	1.191	0.560	0.012	81.189	3.90
red back	CS	153	no	86	56.29	0.844	0.043	0.051	74.479	10.07
	PF	316	no	204	14.57	3.369	1.841	0.004	40.654	3.81
lighting	CS	863	yes	863	81.88	0.897	0.122	0.057	25.546	12.35
	PF	348	yes	348	69.50	1.371	0.161	0.017	28.969	4.62

and $T_s = 0.2$. In Table 1, we report some examples on the effectiveness of the experimental evaluation setup and the comparison between the two algorithms. First, we applied the system to the simples case: a red ball moving in a cyan background. In this case, the CS algorithm works perfectly and this allows us to verify the precision of the evaluation testbed, the percentage of correctly tracked frames r_c is 100%, the area ratio $r^T_{\mathcal{A}}$ is almost 1, and the normalized distance between the centers d_{ct} is very small, as we expect in this successfully tracked sequence.

Then, we created some more challenging scenarios, shown in Fig. 2. In all these cases the red target follows a circular trajectory and its dimension varies periodically, with a global period of 40 secs. The more complex experiments allow a deep comparison between the two algorithms. The main difference is that PF can successfully track the splitting sequence, while CS breaks after some frames. As shown in Fig. 3, this is due to the control strategy for the zoom, that aims to keep all the particles in the field of view. On the contrary, CS is forced to choose only one hypothesis, and if it is the wrong one it fails. Nevertheless, this choice allows CS to provide a higher magnification of the target as listed in the column $r^T_{\mathcal{A}}$ of Table 1. Moreover, CS is typically more precise in term of percentage of correctly tracked frames, mainly because PF best candidate

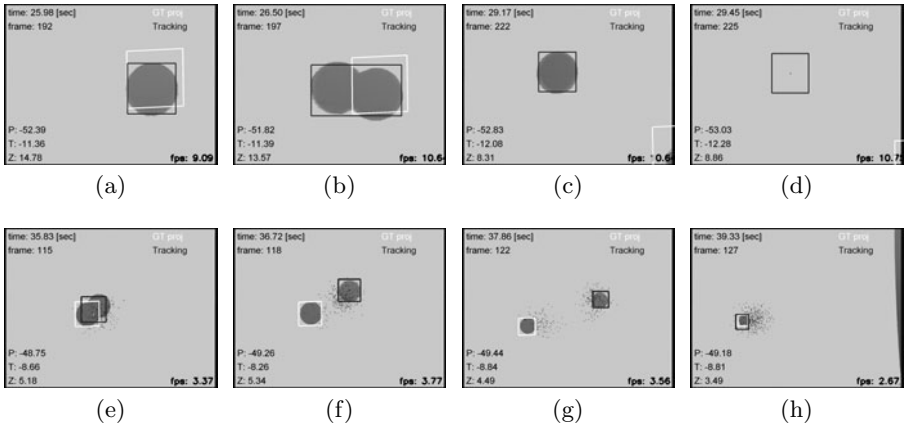

Fig. 3. Comparing CamShift (CS) and Particle Filter (PF) on the *splitting* sequence. The black BB is the target estimation from the tracking algorithm and the white one is the projection of the GT on the current image plane. Top row: some frames from the CS sequence; bottom row: some frames from the PF sequence. The CS algorithm (top row) works properly with a single target (a), but then when multiple targets are present (b) and the two candidates move in different direction, it chooses to follow a single one (c). If the wrong one is chosen, the tracking fails as soon as it disappears (d). On the other hand (bottom row), PF can recover from the error because both target candidates are automatically kept in the field of view of the camera by zooming out (f), (g); then, when the wrong one disappears the correct one is tracked again (h).

does not always fit perfectly the target, or the dynamic model do not succeed to follow quick changes of the target size. Finally, as the target in the PF tracking smaller because of a lower zoom, it is closer to the center of the screen as shown in column d_{ci} of the table.

5 Conclusions

In this work, we have proposed and implemented a novel method to test different PTZ algorithms. We adapted two classical tracking algorithms to the PTZ framework and evaluated them using the same experimental conditions. The obtained results show the effectiveness of the system and highlight the different behaviors of the two algorithms. Future work will focus on the tuning of the precision of the evaluation system and the comparison of the algorithms in more complex scenarios.

Acknowledgments. The authors would like to thank Michele Stoppa for helpful discussions and technical support.

References

1. Agapito, L., Hartley, R., Hayman, E.: Linear self-calibration of a rotating and zooming camera. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 1 (1999)
2. Bimbo, A.D., Dini, F., Grifoni, A., Pernici, F.: Uncalibrated framework for on-line camera cooperation to acquire human head imagery in wide areas. In: AVSS 2008: Proceedings of the 2008 IEEE Fifth International Conference on Advanced Video and Signal Based Surveillance, pp. 252–258. IEEE Computer Society, Washington, DC, USA (2008)
3. Bradski, G.R.: Computer vision face tracking for use in a perceptual user interface. Intel Technology Journal 2(2) (1998)
4. Cai, Y., de Freitas, N., Little, J.: Robust visual tracking for multiple targets. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006. LNCS, vol. 3954, pp. 107–118. Springer, Heidelberg (2006)
5. Collins, R., Liu, Y., Leordeanu, M.: Online selection of discriminative tracking features. IEEE Transactions on Pattern Analysis and Machine Intelligence 27, 1631–1643 (2005)
6. Comaniciu, D., Ramesh, V., Meer, P., Member, S., Member, S.: Kernel-based object tracking. IEEE Transactions on Pattern Analysis and Machine Intelligence 25, 564–577 (2003)
7. Davis, J., Chen, X.: Calibrating pan-tilt cameras in wide-area surveillance networks. In: Proceedings Ninth IEEE International Conference on Computer Vision, vol. 1, pp. 144–149 (October 2003)
8. Ellis, A., Shahrokni, A., Ferryman, J.M.: Pets2009 and winter-pets 2009 results: A combined evaluation. In: Twelfth IEEE International Workshop on Performance Evaluation of Tracking and Surveillance (PETS-Winter), pp. 1–8 (December 2009)
9. Greiffenhagen, M., Comaniciu, D., Niemann, H., Ramesh, V.: Design, analysis, and engineering of video monitoring systems: An approach and a case study. PIEEE 89(10), 1498–1517 (2001)
10. Isard, M., Blake, A.: Condensation: Conditional density propagation for visual tracking. International Journal of Computer Vision 29, 5–28 (1998), doi:10.1023/A:1008078328650
11. Lieberknecht, S., Benhimane, S., Meier, P., Navab, N.: A dataset and evaluation methodology for template-based tracking algorithms. In: 8th IEEE International Symposium on Mixed and Augmented Reality, ISMAR 2009, pp. 145–151 (October 2009)
12. Ling, H., Wu, Y., Blasch, E., Chen, G., Lang, H., Bai, L.: Evaluation of visual tracking in extremely low frame rate wide area motion imagery. In: 14th Conference on Information Fusion (FUSION, 2011). IEEE, Los Alamitos (2011)
13. Raimondo, D.M., Gasparella, S., Sturzenegger, D., Lygeros, J., Morari, M.: A tracking algorithm for ptz cameras. In: 2nd IFAC Workshop on Distributed Estimation and Control in Networked Systems, NecSys 2010 (September 2010)
14. Varcheie, P., Bilodeau, G.-A.: People tracking using a network-based ptz camera. Machine Vision and Applications 22, 1–20 (2010), doi:10.1007/s00138-010-0300-1
15. Yilmaz, A., Javed, O., Shah, M.: Object tracking: A survey. ACM Computing Surveys 38(4), 13:1–13 (2006)