A Multiple Hypothesis Approach for a Ball Tracking System

Oliver Birbach[1] and Udo Frese[2]

[1] Deutsches Forschungszentrum für Künstliche Intelligenz GmbH,
Safe and Secure Cognitive Systems, Enrique-Schmidt-Str. 5, 28359 Bremen, Germany
E-Mail: Oliver.Birbach@dfki.de
[2] Fachbereich 3 - Mathematik und Informatik, Universität Bremen,
Postfach 330 440, 28334 Bremen, Germany
E-Mail: ufrese@informatik.uni-bremen.de

Abstract. This paper presents a computer vision system for tracking and predicting flying balls in 3-D from a stereo-camera. It pursues a "textbook-style" approach with a robust circle detector and probabilistic models for ball motion and circle detection handled by state-of-the-art estimation algorithms. In particular we use a Multiple-Hypotheses Tracker (MHT) with an Unscented Kalman Filter (UKF) for each track, handling multiple flying balls, missing and false detections and track initiation and termination.
The system also performs auto-calibration estimating physical parameters (ball radius, gravity relative to camera, air drag) simply from observing some flying balls. This reduces the setup time in a new environment.

1 Introduction

Robotics is not only about relieving mankind from laborious and dangerous tasks. It is also about playfully exploring the world of dynamic motion and perception from the perspective of a machine. The latter view led to the RoboCup 2050 vision of a human-robot soccer match as proposed by Kitano and Assada [1]. Our interest lies in the vision part, namely visually perceiving a soccer match from the players perspective. In previous work [2] we have investigated how to predict a flying ball from images of a *moving* camera. This paper presents a system for *robust* tracking and prediction of *multiple* balls from a static camera.

We provide three contributions with the overall system: First, a novel circle detector avoids hard decisions and thresholds for the sake of robustness. Second, a multi-hypothesis tracking (MHT) system based on the implementation by Cox [3] robustly handles several flying balls and false or missing measurements. And third, an auto-calibration mechanism learns physical parameters (ball radius, gravity vector in camera coordinates, air drag) simply by observing flying balls.

Ball tracking is important in TV sport scene augmentation. The vision systems in [4–6] exploit the TV perspective, with the field as a static uniform background. In contrast, in the long run our system shall work from the player's

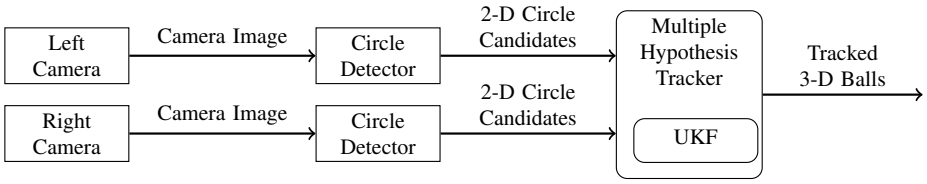

Fig. 1. System Overview: In the left and right image circle candidates are detected. These are fused by an MHT establishing the stereo correspondence, assignment to tracks and track initiation and termination. The two detectors run in parallel.

perspective with a moving camera. Hence we propose a novel circle detector to find the ball in front of moving and cluttered background. The system by Gedikli et al. [6] uses MHT as we do, but for player tracking, complementary to our task.

In RoboCup tracking is often 2-D with a coloured ball and manually tuned heuristics as for instance in the CMVision [7] software. Experience in the competitions shows, that usually hours are needed to tune the vision system to new environments. We experienced the same with a trade-fair demonstration of a ball catching robot [8]. This motivated our ball detection algorithm that does not need tuning parameters. Voigtländer et al. [9] propose a tracking system that fits a 3-D parabola to stereo triangulated ball positions. This ignores the uncertainty structure of stereo triangulation, where depth is most uncertain. We model the metrical uncertainty of circle detection in an UKF [10] to improve accuracy and the uncertainty, whether the circle is actually a ball, in an MHT to improve robustness.

Indeed, we see our contribution in presenting a "textbook-style" algorithm without heuristics that still works in real-time.

2 System Architecture

The architecture for our proposed ball tracking system is a two-staged bottom-up process (Fig. 1). In the first stage, circle candidates are extracted from a pair of cameras exhaustively searching through all centers and radii (Sec. 3). In the second stage, the circle candidates of both images are fused to ball tracks using the MHT algorithm (Sec. 5). We use the implementation by Cox [3]. This algorithm defines a systematic method for setting up association-hypotheses between multiple-targets and measurements considering false-alarm and missing measurements and evaluating them probabilistically. For our system, a target corresponds to a single ball trajectory whereas a measurement is a circle candidate from the detector. Hypothetical associations between tracks and measurements are evaluated by an UKF, which provides the probabilistic model for a single track (Sec. 4). The most likely association hypothesis and its tracked ball trajectories are returned by the MHT.

After the system description, the method for learning the parameters of the physical model is given in Sec. 6. Last, experimental results are presented.

3 Circle Detection

This section describes the circle detection algorithm that is used to find the flying ball(s) in the image. It assumes that the ball has a circular outline in the image with a radial intensity gradient. The algorithm ignores the circle's interior, which could in future be used for recognising false detections.

3.1 An Improved Definition for a Circle Response

Our approach is related to the well known circle Hough-transform [11, 12] as both define a circle response $CR(x_c, y_c, r)$ as a function of the circle center (x_c, y_c) and radius r and search for local maxima of $CR(x_c, y_c, r)$. The circle Hough-transform basically goes through all edge pixels of the image and moves perpendicular to the edge for a distance of $r \in [-r_{\max} \dots r_{\max}]$. For all resulting pixels $(x(r), y(r))$ the circle response $CR(x(r), y(r), |r|)$ is incremented.

So, $CR(x_c, y_c, r)$ is the number of edge pixels closer than $r_{\max}$, where this perpendicular line goes through (x_c, y_c) within half a pixel. This definition involves two hard decisions designed to facilitate efficiency. The first is the classification of edge pixels with a threshold that is sensitive to illumination and image blur. The second is the half-a-pixel criterion for the perpendicular line, which is sensitive to noise. Such early decisions impair robustness.

We now present a circle response definition that is illumination invariant and continuous with no thresholds. It represents *the average fraction of image contrast along the circle that can be explained as a radial intensity gradient.* This notion can be formalized mathematically, but we omit the derivation here for lack of space. The idea is as follows: A local image patch around a point can be decomposed into a sum of components, similar to Fourier decomposition. One component is the image structure we expect, namely a linear intensity gradient in radial direction. The sum of the squared norm of all components, is the local image variance. So, if we take the squared norm of the linear intensity component and divide it by the local image variance, we get an illumination invariant measure that indicates which fraction of the local image contrast is a radial gradient.

Practically, the result is a contrast-normalized Sobel filter C, where the vector length indicates *gradient purity* rather than *gradient intensity.* The circle response $R(\alpha)(x, y)$ in a point (x, y) is then the squared scalar product of the radial direction $\left(\begin{smallmatrix}\cos\alpha\\ \sin\alpha\end{smallmatrix}\right)$ with C.

$$R(\alpha) = \left(\begin{pmatrix} \cos\alpha \\ \sin\alpha \end{pmatrix} \cdot C \right)^2, \quad C = \frac{\sqrt{2}\left(\left(\begin{smallmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{smallmatrix} \right) * I,\ \left(\begin{smallmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{smallmatrix} \right) * I \right)^T}{\sqrt{16 \left(\begin{smallmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{smallmatrix} \right) * I^2 - \left(\left(\begin{smallmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{smallmatrix} \right) * I \right)^2}} \quad (1)$$

It can be seen, that in (1) the nominator is the ordinary Sobel filter. The denominator is the image variance computed as the weighted mean of I^2 minus the squared weighted mean of I. This follows from the well known variance formula $\mathrm{Var}(X) = \mathrm{E}(X^2) - (\mathrm{E}(X))^2$. The factors $\sqrt{2}$ and 16 make $R(\alpha) \in [0 \dots 1]$.

The circle response is now the average fraction of radial gradient on the circle.

$$CR(x_c, y_c, r) = \frac{1}{2\pi} \int_{\alpha=0}^{2\pi} \left(\begin{pmatrix} \cos\alpha \\ \sin\alpha \end{pmatrix} \cdot C \begin{pmatrix} x_c + r\cos\alpha \\ y_c + r\sin\alpha \end{pmatrix} \right)^2 d\alpha \qquad (2)$$

Normalizing the contrast on each pixel individually is important. Otherwise one large gradient dominates the response making it higher than for a small gradient on the whole circle. False circles tangential to intense edges would result.

Finally, the N_{cand} largest local maxima are circle candidates for the MHT.

3.2 Real-Time Implementation

We have taken considerable effort to implement (2) in real-time[3]. C in (1) is only computed once per pixel and well suited for Single Instruction Multiple Data (SIMD) optimization (Intel SSE here). We exploit that the convolutions in (1) are all combinations of blurring by $\frac{1}{4}(1,2,1)$ and differentiation by $(-1,0,1)$ in x and y respectively. This shows again the relation to the Sobel filter. The result is well represented with 7 bits plus sign reducing memory throughput.

The key challenge is computing the integrand of (2) for every pixel on every circle considered ($\pi r_{\max}^2 wh = 15{\cdot}10^9$ times in our case). Furthermore, the pattern of memory access on C is regular regarding x_c, y_c but not for r and α. Hence, we iterate on r, α, y_c, x_c from outer to inner loop, so we can SIMD-parallelize in x_c. However, this sequence of loops would require C to be loaded many times ($\pi r_{\max}^2$) from memory. To avoid the bottleneck, we process blocks of 16×16 pixel in x_c, y_c and nest the loops as y_c block, x_c block, r, α, y_c in block, x_c in block. Furthermore, the vector $\left(\begin{smallmatrix} \cos\alpha \\ \sin\alpha \end{smallmatrix}\right)$ and the relative memory address of $\left(\begin{smallmatrix} r\cos\alpha \\ r\sin\alpha \end{smallmatrix}\right)$ is precomputed in a table with one entry for every pixel along each circle.

Even though the integrand of (2) is computed in 0.42ns, the whole computation still needs 6.6s (Intel Xeon, 2.5GHz). Our solution is a multi-scale approach where we define a detection radius r_{det} (3 pixel) and successively halve the image. Starting from the lowest resolution we search for circles of radius $r_{\text{det}} \ldots 2r_{\text{det}} - 1$ and successively refine the circles found before. So every circle is detected at that resolution where its radius is in the above interval. This multi-scale approach just needs 13.1ms in which the image is scaled down up to one fourth of the original size.

For comparison, the openCV implementation of circle Hough-transform needs 85ms. However, we believe that with comparable implementation effort, Hough-transform would be faster than our detector. The main advantage is, that our detector uses no thresholds making detection more robust from our experience.

4 Single Track Probabilistic Model

This section describes the Kalman Filter used to track the ball as a single target from circle measurements of the camera images. It presents the probabilistic model both for the motion of the ball and for the cameras observing the ball.

[3] The implementation is available upon request.

When tracking a ball, we are interested in its position over time. We also need the velocity to predict the ball's upcoming position. Therefore, the estimated state vector $\boldsymbol{x}_b = [\boldsymbol{x}\ \boldsymbol{v}]$ consists of the ball's 3D-position $\boldsymbol{x}$ and 3D-velocity $\boldsymbol{v}$.

The change of the ball's state during flight, required by the filter's prediction step, is modeled by classical mechanics including gravitation $\boldsymbol{g}$ and air drag α.

$$\dot{\boldsymbol{x}} = \boldsymbol{v}, \quad \dot{\boldsymbol{v}} = \boldsymbol{g} - \alpha \cdot |\boldsymbol{v}| \cdot \boldsymbol{v} + \mathcal{N}(0, \sigma_v^2), \quad \alpha = \tfrac{1}{8}\pi c_d \rho d^2 m^{-1} \tag{3}$$

The factor α determines the air drag, with c_d drag coefficient, ρ density of air, d ball diameter, and m ball mass. The uncertainty in this process is modeled as Gaussian noise with zero mean and the covariances σ_v perturbing the velocity.

The measurement equations used in the update step model the position and radius of the circle detected in a single camera image. We use a pin-hole camera model plus radial distortion [2] that provides a function h which maps a point from a 3D-scene into the image plane. To project a ball into an image, we calculate two orthogonal vectors with length $d/2$ orthogonal to the ball's line-of-sight. We then add $\boldsymbol{x}$ and project these four points using h into the image plane. The results $(p_{x,i}, p_{y,i})$ are recombined to center and radius:

$$\left(\begin{smallmatrix} x_c \\ y_c \end{smallmatrix}\right) = \tfrac{1}{4}\sum_{i=1}^{4} \left(\begin{smallmatrix} p_{x,i} \\ p_{y,i} \end{smallmatrix}\right), \quad r = \sqrt{\tfrac{1}{4}\sum_{i=1}^{4}(p_{x,i} - x_c)^2 + (p_{y,i} - y_c)^2} \tag{4}$$

This method implicitly triangulates at the ball diameter to obtain depth.

Generally, tracking systems of this kind use the Kalman filter which represents the state distribution as a Gaussian random variable and requires linear dynamic and measurement models. For our models linearization is required. One could use the Jacobians which is known as the Extended Kalman Filter (EKF). Better results are achieved by the Unscented Kalman Filter (UKF) [10] which utilizes the so-called unscented transform to linearize the model at a given state. It represents the state's mean and covariance by a set of so-called sigma points, which are computed such that their sample mean and covariance corresponds to the state's mean and covariance. With these, the whole Gaussian is transformed by propagating each sigma-point through the nonlinear function. The propagated sigma-points are then combined to mean and covariance of the transformed state. This method reduces linearization errors that perturb tracking. Therefore, we use the UKF to estimate the state of a single track over time.

5 Multiple Hypotheses Tracking

In the previous section, we considered how to track a single ball (the target) from a series of measurements originating from that ball. In practice, associating a series of measurements with its corresponding track, when there are multiple measurement candidates and multiple targets, is difficult for a number of reasons. First, measurements might be missing, i.e. occluded or not detected. Second, all false measurement candidates (e.g. a person's head) need to be correctly classified as false-alarm measurements and therefore not associated with any track. Finally, tracks start and finish at an unknown points in time.

5.1 The approach of Cox and Reid

One solution to such a data association problem was proposed by Reid [13] and later enhanced to our case by Cox and Hingorani [3] and is known as the multiple hypothesis tracking algorithm (MHT). This algorithm systematically maintains a set of association hypotheses involving multiple targets and multiple, possibly false-alarm, measurements. Formally, a hypothesis is an assignment of each measurement candidate to a track or to "false-alarm". On arrival of new measurement candidates, each hypothesis from the previous time-step is expanded to a set of new hypotheses by considering all possible associations of measurement candidates to tracks in the current time-step. Each measurement may be at most associated with one track and vice versa. Furthermore, the initiation or termination of tracks are considered while generating hypotheses.

The algorithm computes the probability for each hypothesis. This probability is effectively the product of individual probabilities for "everything that happened". For most events (track initiated, track terminated, measurement missed, false-alarm) it is simply a preconfigured constant. For the event of a measurement being associated with a track, the single track model gives the probability. This probability expresses, how well the measurements assigned to the track fit metrically to the model. If they fit well, the most likely hypothesis is that they form a track, otherwise it is more likely that they are false-alarms. This behaviour is the essence of the functionality provided by the MHT.

Formally, for the hypothesis m at time k the probability is [3]

$$P(\omega_m^k|Z^k) = \frac{1}{c} P(\omega_{l(m)}^{k-1}|Z^{k-1}) \lambda_N^{\upsilon} \lambda_F^{\phi} \prod_{i=1}^{m_k} \mathcal{N}_{t_i}(z_i(k))^{\tau_i} \prod_t (P_D^t)^{\delta_t} (1 - P_d^t)^{1-\delta_t} (P_\chi^t)^{\chi_t} (1 - P_\chi^t)^{1-\chi_t} \quad (5)$$

where c is a normalizing factor, $P(\omega_{l(m)}^{k-1}|Z^{k-1})$ is the probability of the parent hypothesis and known from the previous time-step, λ_N is the density of new targets, λ_F is the density of false-alarm measurements, υ is the number of new targets and ϕ the number of false alarms in this hypothesis. $\mathcal{N}(z_i(k))^{\tau_i}$ is the likelihood of the measurement z_i at time k given the target t_i provided by the single track model, P_D^t is the probability of detecting and P_χ^t is the probability of ending track t. τ_t, δ_t, χ_t are 1 if $z_i(k)$ is associated with a known track, if track t detected at time $k-1$ is also detected at time k or if track t known at time $k-1$ is terminated at time k, respectively. Otherwise they are 0.

Although, we can evaluate the probability of every hypothesis, the exponential complexity of the ever-growing tree of hypotheses makes an efficient implementation infeasible. Therefore, several strategies for removing unlikely hypotheses proposed by Cox and Hingorani [3] are used. Ratio pruning removes all hypotheses whose ratio to the best hypothesis is below a certain threshold. This is efficiently done by generating only the k best hypotheses following Murty. Last, N-scan-back pruning removes hypotheses based on the assumption, that ambiguities can be resolved within N time-steps.

5.2 Track Initiation

Within the MHT algorithm, a track is automatically created for every observed measurement. Obviously, one measurement defines the ball's position via the inverse of (4) by triangulating over the ball diameter. However, it does not define velocity. So we set the initial velocity covariance to a large prior representing an upper bound on typical velocities. When later a nearby measurement is associated with the track, the velocity is implicitly computed by the UKF. Still, such a prior is important, because it bounds the area where measurements could be associated with this track thereby limiting the number of hypotheses.

Furthermore, we correctly initialize the covariance of the ball's position using the Jacobian of the inverse of (4). This prevents false associations compared to the widely used initialization with a constant covariance.

5.3 Circle candidate correspondence

When using stereo vision, correspondences between features found in both camera images must be established. With MHT there are two possibilities. First, one could match candidates from both images prior to the tracking and integrate the result as 3D-measurements into the MHT. Second, one could integrate the set of circle candidates of each camera image successively. Then, MHT implicitly handles the correspondences when associating candidates to the same track, leading to more hypotheses though. We implemented this, because it could process tracks observed from just one camera, where the other one would not find a match.

Future work is to compare with a particle filter replacing both UKF and MHT. The challenge is the following: Due to the low dynamic noise, a particle filter effectively selects from the particles initialized from the first few measurements the particle fitting best with all measurements. This requires many particles since, in particular the velocity uncertainty decreases by a large factor.

6 Physical Parameter Learning

The model (3) and (4) require the gravity vector $\boldsymbol{g}$ (relative to the camera), the air drag α, the ball diameter d and the stereo-camera calibration. The latter is calibrated only once, but $\boldsymbol{g}$, which depends on the camera's orientation, and α, d, which depend on the used balls, change with every new environment.

For easy setup, the parameters are estimated from flying balls during a calibration phase. There the UKF runs with $(\boldsymbol{g}, \alpha, d)$ in the state. The first challenge is that with the additional states, many more false tracks fit to the model, leading to wrong estimates. It would be most rigorous to run another MHT or robust estimator to fuse $(\boldsymbol{g}, \alpha, d)$ estimates of different tracks. We currently proceeds simpler. First, we provide some rough prior information ($|\boldsymbol{g}| = 9.81 \pm 0.05 \frac{\mathrm{m}}{\mathrm{s}^2}$, $d = 0.21 \pm 0.1\mathrm{m}$, $\alpha = 0.015 \pm 0.1\mathrm{m}^{-1}$) and second, only estimates from tracks which lasted at least 15 frames are accepted. Technically, the prior on $|\boldsymbol{g}|$ can only be integrated after the uncertainty in $\boldsymbol{g}$ is low enough to linearize $|\boldsymbol{g}|$.

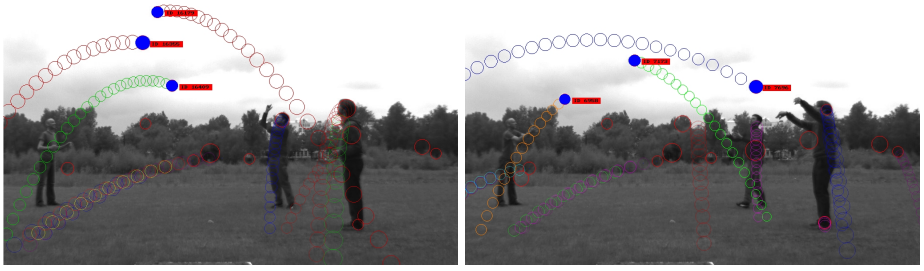

Fig. 2. Camera view of the tracking scenario and overlaid detection results (red circles) and predicted ball trajectories (coloured circles). A filled circle denotes an association to a track. For a video of the tracking results visit `http://www.informatik.uni-bremen.de/agebv/en/MHBallTracking`.

Tracks in roughly opposite direction are needed to distinguish $\boldsymbol{g}$ and α. This is because air drag and the component of $\boldsymbol{g}$ in the direction of flight have similar effect. Hence, when a track is accepted, the final $(\boldsymbol{g}, \alpha, d)$ estimate with covariance becomes the prior for new tracks. So information is collected over several tracks.

7 Experiments

The system described above has been implemented and evaluated. The experimental setup consists of a calibrated camera pair (1024×768 pixel, 30Hz) with a 20cm baseline. Both images are processed in parallel by Two Quad-Core Xeon 2.5 GHz processing units and then fused by the MHT (see Fig. 1). The preconfigured constants for the MHT algorithm were $P_D = 0.3$, $\lambda_\chi = 40$, $\lambda_N = 4 \cdot 10^{-9}$ and $\lambda_F = 3.65 \cdot 10^{-6}$. P_D was chosen lower than the actual detection probability, to allow tracks to continue for a few frames without the integration of measurements. This compensates for correlation between detection in successive frames which is not modeled by the MHT. The pruning strategies were initialized with 0.01 for ratio, 5 for k best hypothesis and 10 for N-scan-back pruning.

In the experiment, four people were throwing up to three volleyballs to each other outdoors. Fig. 2 shows two camera images, with overlaid detection and prediction. Not only the trajectory of the ball flight is visible, but also trajectories created by false-alarm measurements. This is inevitable, since every measurement might be a possible new track. To evaluate the performance of the system, the detection and tracking results were manually analyzed by annotating ball detections and tracks (Fig. 3). In the sequence of 417 stereo frames 2257 balls were visible forming 32 trajectories. The detector postulated 13637 candidates, from which 1567 were balls, leading to 69.4% detection rate. The large number of false alarms was intended, because the detector is used to return the $17 - 20$ circle candidates with highest response. This avoids making early decisions in the detector, where the MHT has much more information. When forming the best hypothesis, the MHT rejected to associate 100 correctly detected measurements. On the other hand, it wrongly associated 76 false-alarm measurements.

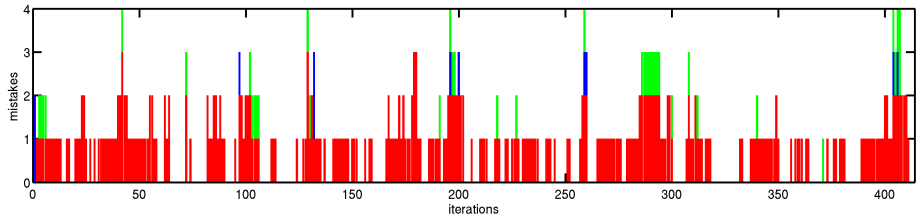

Fig. 3. Stacked bar chart denoting the number of errors the system made at each time-step. The red bar indicates missed balls, the blue bars show correctly detected balls not associated with any track and the green bars visualize the number of false-alarm measurements that were associated with a track.

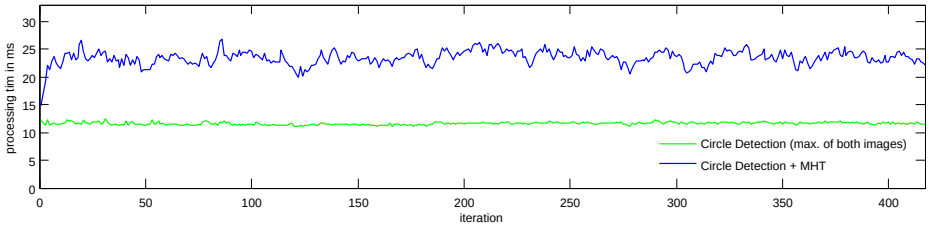

Fig. 4. Processing time of the circle detector (left/right parallel) and the overall system.

A large part of these occurs when the circle detector mistakes a shadow on the ball for the ball returning a circle that is slightly of center and too small. This suggests to avoid the two-stage feed-forward approach and instead look back in the original circle responses during tracking. Then the true circle could be accepted by the tracker, even though it has a slightly worse response than the false circle because it fits better to the motion model [14].

The computation time is shown in Fig.4. One could also be interested in evaluating the metrical prediction. This is left to future work since here the focus is on the MHT.

8 Conclusion and Outlook

We have presented a system for tracking and predicting multiple flying balls from a static stereo camera, consisting of a novel circle detector and the MHT algorithm applied to the physical model of a flying ball. The system is very "textbook-style" containing almost no heuristics and no tuning parameter in the circle detection. Still it works very robustly in real-time.

In the long term, we want to move the system onto a helmet worn by a human player and track the ball from the player's perspective [14]. This requires integrating an inertial sensor for egomotion perception into the single track model. It also poses a subtle challenge for the MHT tracker, because different tracks are not independent any more, but linked by the unknown pose of the camera.

Finally, a long term goal would be to avoid the two-stage architecture and instead optimize a likelihood that includes both the motion model and the circle response $CR(x_c, y_c, r)$. Then, a ball that is visually as vague as many false detections in the image, would still be accepted if it fits well with the motion model. We expect that this approach could further increase the systems robustness.

References

1. Kitano, H., Asada, M., Kuniyoshi, Y., Noda, I., Osawa, E.: Robocup: The robot world cup initiative. In: Proc. of IJCAI-95 Workshop on Entertainment and AI/Alife. (1995)
2. Birbach, O., Kurlbaum, J., Laue, T., Frese, U.: Tracking of Ball Trajectories with a Free Moving Camera-Inertial Sensor. In: RoboCup 2008: Robot Soccer World Cup XII. (2008)
3. Cox, I.J., Hingorani, S.L.: An Efficient Implementation of Reid's Multiple Hypothesis Tracking Algorithm and Its Evaluation for the Purpose of Visual Tracking. IEEE Trans. on Pattern Analysis and Machine Intelligence **18**(2) (1996)
4. Ren, J., Orwell, J., Jones, G., Xu, M.: Real-time 3D football ball tracking from multiple cameras. In: British Machine Vision Conference. (2004)
5. Yan, F., Christmas, W., Kittler, J.: Layered Data Association Using Graph-Theoretic Formulation with Application to Tennis Ball Tracking in Monocular Sequences. IEEE Trans. on Pattern Anal. and Machine Intel. **30**(10) (2008)
6. Gedikli, S., Bandouch, J., von Hoyningen-Huene, N., Kirchlechner, B., Beetz, M.: An adaptive vision system for tracking soccer players from variable camera settings. In: Proc. of the 5th Intern. Conference on Computer Vision Systems (ICVS). (2007)
7. Bruce, J., Balch, T., Veloso, M.: Fast and inexpensive color image segmentation for interactive robots. In: Proc. of the Intern. Conference on Intelligent Robots and Systems. (2000)
8. Frese, U., Bäuml, B., Haidacher, S., Schreiber, G., Schaefer, I., Hähnle, M., Hirzinger, G.: Off-the-Shelf Vision for a Robotic Ball Catcher. Proc. of the Intern. Conference on Intelligent Robots and Systems (2001) 1623–1629
9. Voigtländer, A., Lange, S., Lauer, M., Riedmiller, M.: Real-time 3D ball recognition using perspective and catadioptric cameras. In: Proc. of the 3rd European Conference on Mobile Robots. (2007)
10. Julier, S.J., Uhlmann, J.K.: A New Extension of the Kalman Filter to Nonlinear Systems. In: In The Proc. of AeroSense: The 11th Intern. Symposium on Aerospace/Defense Sensing, Simulation and Controls, Multi Sensor Fusion, Tracking and Resource Management II. (1997)
11. Hough, P.V.C.: Machine analysis of bubble chamber pictures. In: Proc. of the Intern. Conference on High Energy Accelerators and Instrumentation. (1959)
12. Yuen, H., Princen, J., Dlingworth, J., Kittler, J.W.: A comparative study of hough transform methods for circle finding. In: Proc. of the Alvey Vision Conf. (1989)
13. Reid, D.: An Algorithm for Tracking Multiple Targets. IEEE Trans. On Automatic Control **24**(6) (1979) 843–854
14. Frese, U., Laue, T.: (A) vision for 2050: The road towards image understanding for a human-robot soccer match. In: Proc. of the 5th Intern. Conference on Informatics in Control, Automation and Robotics. (2008)