Real-time 3D Ball Recognition using Perspective and Catadioptric Cameras

Arne Voigtländer Sascha Lange Martin Lauer Martin Riedmiller
Institute of Cognitive Science and Institute of Computer Science, University of Osnabrück, Germany

***Abstract*— In this paper we present methods capable of performing ball recognition and tracking on a RoboCup Midsize robot in three-dimensions in real-time. The system uses information from a perspective as well as a catadioptric camera to yield stereoscopic depth. Robustness, accuracy and efficiency are evaluated.**

***Index Terms*— Vision, Multi-Robot Systems, Sensor Fusion, Catadioptric Cameras**

I. Introduction

In this paper we give insight into central parts of the computer vision system developed for our championship winning robot soccer team, the Brainstormers Tribots. Since 2003 the Brainstormers Tribots actively participate in the Middle-Size League of the RoboCup (see http://www.robocup.org), where two teams of six fully autonomous robots play soccer matches on a field of 18x12m. The Brainstormers Tribots won the annual inofficial European Championships four times in a row and became world champion during the RoboCup 2006 held in Bremen, Germany.

Besides very reliable hardware and precise self-localization, the very robust ball detection and accurate prediction of its movement have always been among the key strengths of the Tribots. During games, this allowed the Tribots to intercept freely rolling balls earlier than most of the opponents, to dribble the ball better than any other team and to intercept opposing attackers advancing the ball very aggressively. All these game deciding abbilities heavily depend on a powerful computer vision subsystem. Besides coping with the uncontrolled natural lighting conditions present in the Middle Size League since late 2005[1] and the need for very high reliability and robustness, the challenge in the development of the computer vision system was the need for very capable, yet ressource-saving algorithms, that allow *real* real-time processing (30Hz, upper bound of 33ms per cycle) of images, world modelling, actuator control and decision making— all this despite the very limited computation power of the on-board computers of the soccer robots.

[1]Actually the RoboCup has been often criticized for simplifying the problem too much by placing unnatural demands on the environment and especially the lighting conditions in the early years. Therefore, in 2005 the TC of the Middle Size League decided to remove all constraints on uniformity and minimal and maximal brightness of the lighting from its rules; only a minimal brightness of at least 300 lux is necessary. During the Robocup 2006, games were held inside a hall right in front of a 100m^2 window allowing direct sunlight to enter. The Brainstormers Tribots recently even have demonstrated their vision and self localization to be working in free air with direct exposure to sunlight and in the presence of clouds (a video could be found here: http://www.ni.uos.de/index.php?id=525).

But the really unique feature of the *hybrid* vision system described in this paper is the utilization of two different camera types in order to realize— for the first time on a soccer robot participating in official competitions— real stereo vision for tracking and predicting the position of the ball in three dimensions. In recent years, catadioptric camera systems (often named omnidirectional cameras) have become the gold standard for soccer robots in the Middle Size League since they provide a 360° field of view without needing a pan-tilt assembly or several cameras. Ball recognition, opponent tracking and especially self-localization algorithms heavily rely on the possibility to observe the complete surroundings of the robot in each frame. While fully maintaining these benefits our system adds the additional capability of three-dimensional ball tracking in a region of interest by integrating an additional perspective camera.

The future importance of this new 3D ball tracking feature could not be underestimated. As long as no team was able to lift the ball, the limitation of the standard system using only a single sensor and therefore modelling objects and movements only in two dimensions was not a real problem. But nowadays all successfull teams are able to do high kicks: In the 2006 RoboCup we received a total of 4 goals (while scoring about 80). All were either lob shots or bouncing balls that the goalie robot misinterpreted assuming the ball rolling flat on the ground.

In the rest of this paper we discuss all issues that had to be addressed in order to realize real-time capable and robust three-dimensional ball tracking on our robots using a hybrid camera system. We were the first to introduce real-time stereo vision into the Middle-Size league during the Technical Challenge of RoboCup 2006 (first place of 28 submissions) and during real games at the German Open 2007. Furthermore, to our knowledge this is the first paper discussing a stereo vision system fusing information of an omnicamera and a perspective camera. Although this system has been developed and optimized for the utilization in the RoboCup environment, we believe it might be of interest for other application areas like e.g. home or office robotics as well. The omnidirectional camera facilitates self-localization, mapping and path planning in any environment, whereas the perspective camera provides three-dimensional object positions for e.g. grasping objects with a robot arm.

Related Work

Stereo vision systems based on two perspective cameras relying on epipolar geometry are well established, e.g. [13].

While they provide superior precision compared to the presented system and can be acquired commercially, these systems suffer from the limited field of view which originally has led to the use of omnidirectional cameras. The usage of more than two cameras, e.g. a stereo camera system and an omnicamera, on a mobile platform can be problematic due to the bandwidth limits of current bus interface systems.

In our hybrid approach, we employ an omnicamera and a perspective camera that both need to be calibrated beforehand to recover metric information. Another possibility would be to use structure from motion (SfM) [4] in order to relax this prerequisite or to extract 3D information with a single camera. The major drawback of SfM is its current inability to cope with dynamic scenes as RoboCup where all robots might be in motion and it therefore becomes non-trivial to distinguish ego-motion from a changing scene.

Another possibility to recover the 3D position of the ball is by its observed size, which is known beforehand, using one camera only. However, the ball covers only a few pixels in the omnicamera when viewed at rather short distances of more than 3m. Therefore, the main advantage, using only one omnicamera, cannot be realized. Even utilizing a perspective camera, at least the stark highlights and shadows that are observed have to be modeled in order to gain adequate area estimates. Occlusions and motion blur are highly problematic, too. The center coordinate of colored regions yields a more stable position under these conditions.

Finally, it has also been proposed to use two omnicameras to realize stereo vision[12]. While being very appealing, in our opinion a robust assembly is hard to realize mechanically on a Middle-Size robot.

Considering the pros and cons, we feel that a hybrid system presently might be the best solution for RoboCup providing a fair compromise between computational demands, practicability and precision.

Overview

Whereas it was partly possible to rely on existing work, several algorithms had to be newly developed or adapted in order to fit the strict demands. Special attention has been put on the robustness, on low computational demands as well as on the smooth integration of each submodule into the hybrid vision system. The system consists of the following submodules:

1) Integrating and placing the cameras is briefly discussed in section II.
2) Calibrating (establishing a mapping between world coordinates and image coordinates) the perspective camera is done using a well established model based method proposed by [11] (sec. III-A)
3) Several model-based methods for calibrating catadioptric cameras have been published as well [1]. But due to some impracticable restrictions these models pose on the camera assembly, we developed a new model-free automatic calibration method in order to increase precision (sec. III-B).
4) For the submodule detecting the ball in the images of each camera we had to find a rather clever solution, since it is not even feasible to read out every pixel of the images only once given the available computing power. Nevertheless, since ball detection is that important in RoboCup, we had to come up with a very reliable and robust solution we discuss in section IV-A.
5) Triangulation of the 3D-position of the ball given the detected ball regions and the camera models is done using basic geometry (sec. IV-B).
6) Filtering and predicting the ball movement is solved by extending an approved method for 2D-ball tracking to the 3D case (sec. IV-C).

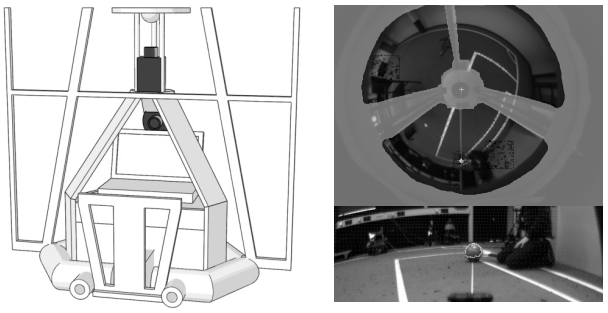

Fig. 1. Left: The position of the two cameras on the soccer robot. Right: The processed images of the two cameras showing a ball at a distance of about 300cm.

II. Camera Integration

The robot is conforming the rules of the RoboCup Middle-Size league and has a height of about 80cm. To give the best viewing angle the catadioptric camera has been placed at the topmost position allowed by the rules. The catadioptric camera consists of an upward facing progressive scan Firewire camera pointing at a hyperbolic mirror. Preferably, the perspecitve camera should be placed in such a way that the optics could not be hit by the ball, too. Therefore we have chosen to place the camera *inside* the robots chasis. For the distance between perspective camera and omnicamera there is a tradeof: obviously, more distance between the cameras would further reduce the error in the 3D position estimates due to noisy ball region detection. But a lower position would also further decrease the available space for other equipment and allow other robots to occlude the ball more easily. Considering this, we have chosen an intermediate position that still allows easy access to other important components of the robot (see fig. 1).

The image integration of both cameras should be synchronized to avoid bad position estimates due to time differences in the image acquisition. This can be achieved easily by using Firewire digital cameras of the same manufacturer relying on the build-in auto-synchronizing feature or by manually triggering the image integration process.

Both cameras capture frames at 30 Hz using the 4:2:2 color subsampling scheme. Using the Format7 specified by the DCAM standard, we set up a region of interest for the perspective camera as big as possible while not exceeding the bandwidth of the IEEE1394a bus.

III. Calibration

To obtain metric information from the camera images and to be able to combine the information from both cameras, it is necessary to calibrate the cameras onto the same world coordinate system. Due to the different nature of the two camera setups, we employ different techniques for each setup. The world coordinate system is defined in a robocentric way with the positive y-axis pointing to the front of the robot, the x-axis pointing to the right and the z-axis pointing upwards.

A. Perspective Camera

Calibrating perspective cameras is well established. We use the comfortable method proposed by Zhang [11]. The calibration process calculates a mapping between image coordinates and world coordinates given a set of points with manually determined image and world coordinates. The calibration assumes a pinhole camera model and relates a model point $M = [X, Y, Z, 1]^T$ and its corresponding image projection $\mathbf{m} = [u, v, 1]^T$. M is specified in homogeneous world coordinates and $\mathbf{m}$ is specified in homogeneous image coordinates. Both coordinates are related up to a scale factor by the transformation:

$$\mathbf{m} \propto \mathbf{A}[\ \mathbf{R} \quad \mathbf{t}\ ]M \quad \text{with} \quad \mathbf{A} = \begin{bmatrix} \alpha & c & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{1}$$

The matrix $\mathbf{A}$ is called the intrinsic matrix. The principal point of the camera is specified by (u_0, v_0), α and β are scale factors in the u and v axes and c represents the skew between these axes. $[\ \mathbf{R} \quad \mathbf{t}\ ]$ are called extrinsic parameters. $\mathbf{R}$ represents the rotation and $\mathbf{t}$ the translation that relate the world coordinate system to the camera coordinate system. Since the used perspective camera exhibits lens distortion, especially due to the use of a wide-angle lens, it is necessary to model radial distortion, as well. This is done using only the first two terms.

Using the inverse of this model, image coordinates can be mapped to lines in the real world. The line is formed by the position of the pixel on the image plane after undistorsion and the camera's focal point, with the exact distance on the line being undefined.

The actual calibration procedure is done by recording a set of images from which correspondence points are determined. Each of these points relates a world coordinate M to its image projection $\mathbf{m}$. Using the calibration procedure described in [11], the points in world coordinates have to be coplanar, which makes it easier to find correspondences compared to older calibration techniques such as [10]. The algorithm further requires the images to be taken from different orientations with regard to the correspondence points. At least three images from different orientations need to be acquired in order to derive a unique solution to the problem.

B. Omnidirectional Camera

Although model-based self-calibration methods for catadioptric camera assemblies have recently been described in literature, e.g. [6, 8], they turn out to be not applicable in this framework since they share at least the following assumptions: The principal axis of the mirror has to be perfectly aligned with the camera axis and the shape of the mirror has to be symmetric in every direction in order to reduce the calibration problem to a circular distortion. Already a small deviation of the mirror's principal axis of less than a millimeter (or small skew) notably impairs the precision of the calibration. However, the device holding the mirror above the camera is assembled from several rather filigree (minimize occlusion) custom made parts. Considering the shocks and stress this device is exposed to during transports of the robots and during frequent crashes, obviously these assumptions are violated irremediably and at least hard to meet when (re-)assembling outside a lab. Therefore, we use a model-free approach for camera calibration that does not depend on such constraints in exchange for needing a lot more correspondence points during the calibration.

The idea for our model-free calibration is based on the property of omnidirectional cameras to preserve the angle at which an object is seen in the image. Hence, if we work with polar coordinates, both in the world coordinates and image coordinates, we immediately get the angle at which an object is mapped into the camera image. Only the mapping between distances in world coordinates and distances in image coordinates (or, for short, distance mapping) must be calibrated.

Due to the aforementioned inaccuracies in the camera mounting, the distance mapping cannot be reduced to a one dimensional function but must be chosen individually for each direction. We use piecewise linear functions in polar coordinates to model these mappings. For each of 45 directions (covering 8 degrees each), 12 support vectors are determined in an automatic calibration process, which then define the distance mapping.

The calibration process is based on a 8m long calibration carpet which is rolled out on the floor and that is built of white, blue and red patches. The robot is placed at one end of the carpet so that the edges of the colored patches are at predefined distances, see Fig. 2 (top left). To calibrate the distance mapping, the robot automatically extracts the colored patches in the image and records the distance of the edges to the image center. Furthermore, the robot slowly performs mulitple revolutions on the spot while calibrating to obtain observations from different directions. In this manner, multiple samples are acquired for each direction and edge yielding a total of over 1000 samples equably spread over the mirror.

After recording the edges, the 12 support vectors of the distance mapping for each direction are determined from the acquired correspondence points along that direction fig. 2 (top right). Since the changes from white to blue and vice versa are not unique and there may occur a few errors in the sampling of the correspondence points, we apply the k-means clustering algorithm (12 cluster centers) to distinguish the samples of different edges. Afterwards cluster centers form the support vectors. Finally, a smoothing approach based on median and moving average filters is used to guarantee that support vectors of neighboring directions referring to the same edge are at similar positions.

A record of acquired edges and their interpolation is given

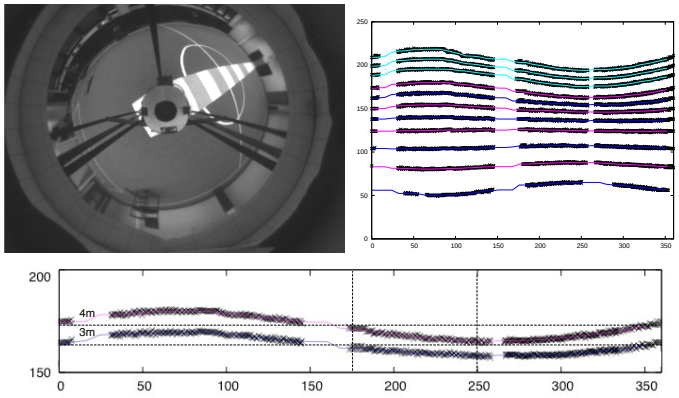

Fig. 2. Top left: camera image captured during calibration. The calibration carpet is built from alternating white and blue patches. Top right: The diagram shows the result of calibration. The y-axis refers to the distance from the image center (in pixel) at which an edge between neighboring patches has been seen while the x-axis refers to the direction in which it has been seen. Each cross refers to one observed edge. The lines represent the support vectors defining the distance mapping found by our model free approach. Bottom: area of the top right diagram showing the support vectors for the distances of 3m and 4m. The dashed horizontal lines represent hypothetical calibration results employing the one direction assumption (mirror symmetry) with the resulting errors

in Fig. 2 (bottom). The directions in which no samples were acquired are occluded by the camera mounting. The wavy structure of image distances for a particular calibration patch is primarily due to imprecisions in the mounting between the camera and the mirror. Imprecisions caused by the perpendicularly mounted camera setup and the ground plane also add to the asymmetry. This illustrates the problem model-based calibration methods would create. By assuming symmetry, they would assume this distance to be fixed for all directions, i.e. a straight horizontal line in this graph. These lines are shown dashed in the graph and represent a possible result of a fitted model-based calibration method. As can be seen, while at a direction of $175°$, both calibrations overlap, the error of the model-based calibration at $250°$ is as large as to mistake the fifth patch with the fourth patch, which constitutes an error of 1m.

IV. Ball Detection

First, the position of the ball is detected in both images separately. The mapping established during the calibration procedure allows the construction of a 3D-line from the camera through the center of the ball as detected in this camera's image. The particular position of the center on the line is then given by intersecting the two lines constructed for each camera.

A. Locating the Ball in the Images

We start with a description of how the ball is detected and tracked in the images of the omnidirectional camera. Due to efficiency reasons, we still use a color-based segmentation algorithm to detect the ball in the image. But instead of segmenting the whole image (computational demands too high), we search for reddish pixels that might be interesting only along a set of radial scan lines extending from the image center to the outer region. We use the same scanlines to detect all other objects of interest (robots, lines, goals) as well. To further speed up this time consuming processing we use lookup tables for the classification of color values.

The detected red pixels are used to seed a contour tracing algorithm that generates chaincodes [5] for the connected region each pixel is a part of. Chaincodes allow to very efficiently calculate region descriptors like e.g. center of gravity and roundness. If more than one red region (ball) has been detected, some heuristics are used to filter out the most likely region: a) we check for a minimal roundness, b) regions definitely inside the playing field are favored c) if necessary, we use the expected position of the ball (see sec. IV-B) to select the most likely region. The expected position (3D) could be mapped back to image coordinates using the inverse mapping (world to image coordinates) established during calibration.

The idea behind this heuristic for preventing the robots to gather around other reddish objects close to the game field is the following: If the ball is on the ground, it is possible to decide whether or not it is inside the playing field using only a single camera. Since the rules guarantee that there is always only one single red region (ball) *inside* the playing field, we can reinitialize safely our tracking with this region. If the ball leaves the ground, the decision whether or not it is inside the field might not be possible considering only one camera's image. In this case, more than one single region may have to be considered. From these regions we select the region closest to our prediction.

Once the ball has been detected in a frame, the estimate of the ball movement described in section IV-B could be used to find the ball in subsequent frames. This is realized using a Monte Carlo method. Some normal distributed pixels near the predicted position (predicted world coordinates mapped back to image coordinates using the models established during calibration) are tested for being red and— if positive— are used as additional seeds for the chaincoding. This attention shift and allocation of further computing power to interesting regions effectively prevents loosing a once detected ball in subsequent frames without needing to process every single pixel of the image.

B. 3D Position Estimation

Once the two ball regions are found, we can use the calibrated model of the cameras to determine two lines on which the ball is assumed to be. The line l_P determined for the perspective camera is defined— as stated above— by the focal point of the camera $\mathbf{f}_P$ and the vector $\mathbf{r}_P$ that connects the focal point and the point of the undistorted pixel on the image plane. In a similar way, we can define a line l_O for the omnidirectional camera assuming a single focal point within the mirror $\mathbf{f}_O$ and the vector $\mathbf{r}_O$ that connects $\mathbf{f}_O$ with the point on the ground plane which we get from the calibrated image-to-world mapping, see section III-B. Although the assumption of a single focal point only holds for special camera assemblies, the error introduced with this simplification turns out to be small.

Once having determined the two lines l_P and l_O the center of the individual ball regions must be on, we can determine the 3D position of the ball by calculating the intersection of the lines. However, in practice both lines are skewed due to noise and imprecision in image processing and calibration. Therefore, we use the point of minimal distance $\mathbf{c}_S$ to both lines as approximation to the actual ball position. Using standard methods from geometry we receive the following formula that has been implemented on the robots:

$$\mathbf{c}_S = \frac{1}{2}\Big(\mathbf{f}_P + \mathbf{f}_O + \frac{\langle \mathbf{r}_P, \mathbf{r}_O\rangle\langle \mathbf{r}_O, \mathbf{d}\rangle - \langle \mathbf{r}_O, \mathbf{r}_O\rangle\langle \mathbf{r}_P, \mathbf{d}\rangle}{\langle \mathbf{r}_P, \mathbf{r}_P\rangle\langle \mathbf{r}_O, \mathbf{r}_O\rangle - \langle \mathbf{r}_O, \mathbf{r}_P\rangle^2}\mathbf{r}_P + \frac{\langle \mathbf{r}_P, \mathbf{r}_P\rangle\langle \mathbf{r}_O, \mathbf{d}\rangle - \langle \mathbf{r}_P, \mathbf{r}_O\rangle\langle \mathbf{r}_P, \mathbf{d}\rangle}{\langle \mathbf{r}_P, \mathbf{r}_P\rangle\langle \mathbf{r}_O, \mathbf{r}_O\rangle - \langle \mathbf{r}_O, \mathbf{r}_P\rangle^2}\mathbf{r}_O\Big) \tag{2}$$

where $\mathbf{d} = \mathbf{f}_P - \mathbf{f}_O$ and $\langle \cdot, \cdot \rangle$ denotes the dot product.

C. Ball Tracking

The camera system provides a sequence of ball positions at exposure time. To be able to interact with the ball and to bridge the time gap between perception and actuation by prediction [2] we need information on the dynamics of the ball, i.e. estimates of the ball velocity and its direction of movement.

Here, we basically rely on an approved method for tracking and predicting the movement of the ball [7]. Since the original method did only utilize a two dimensional motion model, we needed to extend it to the third dimension. As long as the ball does not bounce on a surface, the gravity does only affect the new z-component of the ball movement and does not affect the movement component perpendicular to the z-axis (x component and y component). Thus, the extension of the existing ball filter could be done independently of the existing movement model. Neglecting air drag, we added new rules for the three possible movements characterized by its height $z(t)$ and its vertical velocity $v(t)$:

- rolling on the ground: $z(t) = 0$
- flying on a parabola trajectory: $z(t) = z_0 + v_0(t - t_0) - \frac{1}{2}a(t - t_0)^2$
- bouncing at point in time t: $\lim_{\tau \to t+0} v(\tau) = -b \lim_{\tau \to t-0} v(\tau)$

where t_0 is a reference point in time, v_0 the linear vertical velocity at t_0 and b is a factor that controls the loss of energy during bouncing, chosen from experiments. The constant factor a is the acceleration due to gravity. The parameters z_0 and v_0 and which one of the three movement models should currently be used for calculating predictions is estimated from the n last measurements of the ball position. For more details we would like to refer the reader to [7].

V. Experiments

The presented setup was tested and compared to ground truth information in several experiments to evaluate its accuracy. The ground truth information was acquired by a three-dimensional laser scanner[9]. The scanner system consists of a 2D laser scanner that is mounted on a servo in order to sequentially acquire laser scans by tilting the scanner followed by composing the individual scans into one three dimensional scan using ICP[3]. The scanner was calibrated to the reference coordinate system of the soccer field and scanned one 3D scan every time a camera measurement was performed. In each scan, a spherical region of interest was manually selected to represent the ball and serves as ground truth for the camera information. Figure 3 shows a laser scan superimposed by camera information.

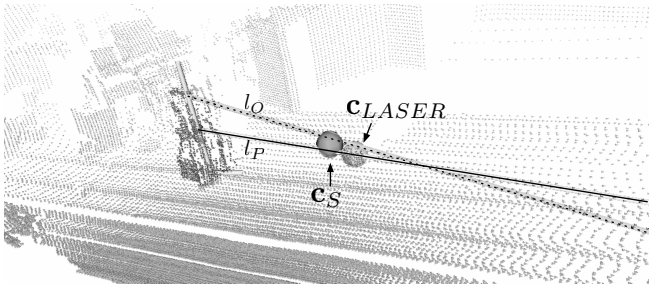

Fig. 3. Ground truth 3D laser data with superimposed camera-processed data. The lines shown represent l_O and l_P while the spheres represent the camera-detected ball as well as the ball within the laser scan.

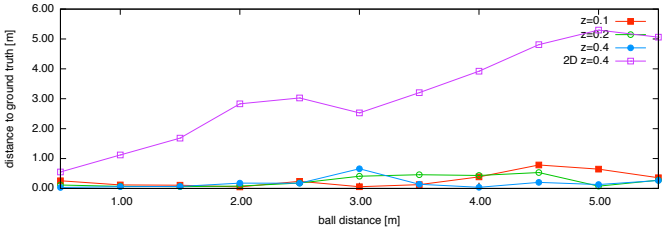

Fig. 4. Error for different ball heights in relation to the ball distance. The line marked with "2D" gives the resulting error of using only the omnicamera and assuming the ball on the ground as done up to now.

A series of experiments has been conducted with a stationary ball of 11cm radius. The distance between robot and ball as well as the ball height was varied. The ball was presented on the ground ($z = 11$cm) and in two lifted positions ($z = 21$cm and $z = 41$cm). The distances varied between 0.5m and 5.5m in intervals of 0.5m. No constraints about the ground plane were modeled that would artificially improve the error rate for the $z = 11$cm case. Figure 4 shows the error between the ground truth and the estimated ball position for the different heights and distances. The 2D $z = 41$cm case represents the error of position estimates if we would only use the omnidirectional camera and assume the ball to be on the ground plane. As can be seen from the figure, this assumption yields very large errors, even for relatively small distances. This shortcoming, which has prevented any reasonable predictions for above-ground balls up to now, has been greatly reduced with the stereo vision approach.

To measure the performance of the stereo vision approach in a dynamic situation, we examined the case of a ball that is being vertically bounced at a distance of 2m from the robot. The limited temporal resolution of the scanner did not allow to use it in this experiment so that we are left with the recognized positions and estimated velocities of the vision system. Figure 5 (left) shows the observed height of the ball over time and the

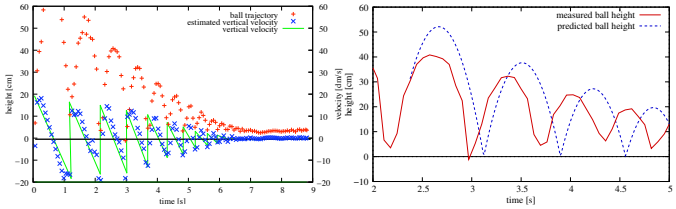

Fig. 5. Left: trajectory of a bouncing ball (red), estimated vertical velocity over time (blue), and true vertical velocity exhibiting typical saw-tooth pattern (green). Right: usage of the position and velocity estimates to initialize the new predictive model for the z-movement of the ball. The dotted blue line shows the movement of the ball as predicted into the future from the point in time 2.4s using the then available measurements.

estimated vertical ball velocities that are the important values for initializing the predictive models. The limited scope of the camera did not allow to recognize the ball above a height of 58cm, hence these values are absent near the top of the first parabola. The graph of the ball height shows the characteristic form of consecutive parabolas while the graph of velocity estimates takes the form of a saw-tooth diagram (as expected) with decreasing amplitudes.

In the vision system the position and velocity estimates are used to initialize the predictive motion models. Figure 5 (right) exemplary shows the movement as predicted at $2.4s$ using the then current position and velocity estimates. Over the relatively long period of about 2.5 seconds into the future (typically only time spans of about 200ms are extrapolated in order to bridge the gap between image acquisition and first physical effects of executed commands) the model gives reasonable predictions for the amplitudes of the parabolas as well as for the points in time where the ball bounces on the ground, slightly overestimating the initial velocity (therefore the higher amplitudes) and missing the real bouncing events by less than 0.15s.

The experiments were performed using the onboard computer of the robot, which is an off-the-shelf subnotebook with a 1 GHz Intel Centrino CPU. The average processing time to detect the ball in the omnicamera was 4.9ms including the time for checking the scan lines (4.6ms), calculating the contour descriptions of the red regions (0.3ms) and selecting the most probable region. This time already includes the time needed to detect seeding pixels of the other objects of interest (goal, obstacle, etc.). The total processing of the image of the omnicamera including line, goal and obstacle detection took 5.8ms. The processing of the image from the perspective camera is slightly faster since presently we only look for potential balls and scan lines can be placed more efficiently. Combined with the image acquisition time, this yields a total vision time of less than 12ms.

VI. Conclusion

The hybrid vision system presented in this paper is driven by the idea to combine the advantages of omnidirectional cameras with their large visual field with the advantages of stereoscopic 3D vision. The ability to recognize the ball position in three dimensions became necessary as many teams in the RoboCup Middle-Size League lift the ball.

To achieve this goal several algorithms were developed or adapted to fit the different camera types and the RoboCup framework: precise camera calibration for the perspective camera was adapted from a standard approach whereas a new algorithm had to be developed for the omnidirectional camera. Furthermore, a computationally feasible method to find the correct position of the ball in the two images of both cameras, an approach to determine the 3D ball position and a dynamic model of ball movement that incorporates the vertical ball velocity were newly developed.

Due to the limited available computational power of the onboard computers we proposed very efficient algorithms to analyze the camera images. The time needed to analyze the two camera images and estimate the ball position and velocity was measured to be less than 10ms which allows to work with the maximal frame rate of 30Hz in real-time.

Experiments provide that the accuracy of position estimates is within acceptable bounds, with an error of less than 50cm most of the time, even for balls at a distance of 5m. Compared to single camera systems, the error is reduced significantly, thus enabling the robots to react on lifted balls and to show senseful behavior, e.g. in defense strategies and in intercepting the ball. Shortcomings of the currently used system are the limited height and limited distance up to which the ball can be recognized. This problem can be easily addressed by changing the placement of the camera.

References

[1] Joao P. Barreto. A unifying geometric representation for central projection systems. *Comput. Vis. Image Underst.*, 103(3):208–217, 2006.

[2] Sven Behnke, Anna Egorova, Alexander Gloye, Raúl Rojas, and Mark Simon. Predicting away robot control latency. In *RoboCup 2003: Robot Soccer World Cup VII*, pages 712–719, 2003.

[3] Paul J. Besl and Neil D. McKay. A method for registration of 3-d shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, February 1992 1992.

[4] B. Boufama and A. Habed. Three-dimensional structure calculation: Achieving accuracy without calibration. *Image and Vision Computing*, 22(12):1039–1049, October 2004.

[5] H. Freeman. On the encoding of arbitrary geometric configuration. *IRE Transaction on Electronic Computers*, 10(2):260–268, 1961.

[6] Christopher Geyer and Konstantinos Daniilidis. Catadioptric camera calibration. In *ICCV*, pages 398–404, 1999.

[7] Martin Lauer, Sascha Lange, and Martin Riedmiller. Modeling moving objects in a dynamically changing robot application. In Ulrich Furbach, editor, *KI 2005: Advances in Artificial Intelligence*, pages 291–303. Springer, 2005.

[8] B. Micusik and T. Pajdla. Para-catadioptric camera auto-calibration from epipolar geometry. In *Asian Conference on Computer Vision, Jeju Island, Korea*, 2004.

[9] Hartmut Surmann, Kai Lingemann, Andreas Nüchter, and Joachim Hertzberg. A 3d laser range finder for autonomous mobile robots. In *Proceedings of the 32nd International Symposium On Robotics*, 2001.

[10] R.Y. Tsai. An efficient and accurate camera calibration technique for 3d machine vision. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 364–374, Miami Beach, FL, 1986.

[11] Zhengyou Zhang. Flexible camera calibration by viewing a plane from unknown orientations. In *Proceedings of the International Conference on Computer Vision*, pages 666–673, 1999.

[12] Zhigang Zhu. Omnidirectional stereo vision. In *The 10th IEEE International Conference on Advanced Robotics*, 2001.

[13] A. Zisserman, P. Beardsley, and I. Reid. Metric calibration of a stereo rig. In *IEEE Workshop on Representation of Visual Scenes, Boston*, pages 93–100, 1995.