EGVE Symposium (2008)
B. Mohler and R. van Liere (Editors)

# Real-time Color Ball Tracking for Augmented Reality

D. Sýkora[1,†], D. Sedláček[1], and K. Riege[2]

[1] Czech Technical University in Prague
[2] Fraunhofer Institute Intelligent Analysis and Information Systems IAIS, Germany

**Abstract**

*In this paper, we introduce a light-weight and robust tracking technique based on color balls. The algorithm builds on a variant of randomized Hough transform and is optimized for a use in real-time applications like on low-cost Augmented Reality (AR) systems. With just one conventional color camera our approach provides the ability to determine the 3D position of several color balls at interactive frame rates on a common PC workstation. It is fast enough to be easily combined with another real-time tracking engine. In contrast to popular tracking techniques based on recognition of planar fiducial markers it offers robustness to partial occlusion, which eases handling and manipulation. Furthermore, while using balls as markers a proper haptic feedback and visual metaphor is provided. The exemplary use of our technique in the context of two AR applications indicates the effectiveness of the proposed approach.*

Categories and Subject Descriptors (according to ACM CCS): I.4.6 [Image Processing and Computer Vision]: Segmentation (Edge and feature detection) I.4.7 [Image Processing and Computer Vision]: Feature Measurement (Size and shape) I.4.8 [Image Processing and Computer Vision]: Scene Analysis (Tracking) I.5.5 [Pattern Recognition]: Implementation (Interactive systems)

## 1. Introduction

A popular low cost technique for tracking objects in AR is optical recognition of planar fiducial markers (e.g. *ARToolKit* [KB99], [CF04], *ARTag* [Fia05], *reacTIVision* [BKJ05], *ARToolKitPlus* [WS07]). It allows estimating relative positions and orientations of a large amount of visible markers in real-time. However, the main limitation of this common technique is the necessity of a non-flexible planar area fixed close to or onto a real object. Such an unnatural modification typically decreases the ease of manipulation, inconveniently affects haptic feedback and causes objects to look strange. Another disadvantage of fiducial markers is their high sensitivity to occlusions. Even if a small part of the marker is obstructed, e.g. due to user's manipulation, the object tracking is lost. Redundancy is used to overcome this issue. Furthermore, a large amount of artificial markers within a working area are visually disturbing and therefore significantly decrease the important feeling of a real object being augmented. This is the main aspect of AR.

In this paper we propose a new interaction technique that uses color balls of known diameter as markers. The main advantage as compared to planar markers is that balls are naturally perceived as real objects; they provide pleasant haptic feedback and can be detected even under partial occlusion. Although a single ball allows estimating only a 3D position, dipoles [GF03] or balls with color dots [BR05] can be used to express the orientation.

In contrast to related work [GF03, BR05] we introduce a general, easy-to-implement color ball tracking that is robust to noise and clutter, overcomes occlusion and is able to track many color balls simultaneously in real-time (only 11 ms needed to process 8 balls in general position on a standard PC with a 0.3 Mpix camera). In practice our proposed approach is the first one published providing such high processing speed on commodity hardware. This important feature allows instant cooperation with another (e.g. marker-based) tracker and still keeps high performance.

† sykorad@fel.cvut.cz

Our paper is structured as follows: We first elaborate on related work done in the field of circle detection and the usage of tracked balls as 3D interaction devices. Then, our approach for a real-time color ball tracking technique suitable for AR is presented in Section 3. Our achievement regarding performance, accuracy and limitations of the proposed technique as well as the exemplary integration in two AR applications is demonstrated in Section 4. Final conclusions are given in Section 5.

## 2. Previous work

Ball detection and tracking can be understood as a special case of a more general problem studied in low-level vision: detection of parametric curves in images. Since the perspective projection of a sphere is always a circle, in our case we consider only circle detection.

Research in this field started with the seminal *Circle Hough transform* (CHT) [DH72] that extend the basic principles of Hough transform to circle detection. The key idea is similar as in line detection, i.e. to extract edges first and then for each edge pixel accumulate votes in an appropriate subset of parametric space (here represented by a 3D histogram of circle centers and radii) using all possible circles passing through it. Significant peaks in such a 3D histogram determine centers and radii of salient circles in the image. Although original CHT is recognized to be robust to noise and occlusions, its main drawback is computational complexity. To overcome this issue and still keep robustness of the original CHT a bunch of derived techniques were developed.

One of the first improvements is based on a hierarchical paradigm pioneered by *Li et al.* [LLL86] who start with coarse grid and perform subdivisions only when the number of votes exceeds some predefined threshold. Similarly dimensionality reduction of the parametric space is used to reduce the number of votes [XJ02, JC05]. In this method a selected subset of parameters is estimated first (e.g. a circle center) and then such an initial solution is used to reduce the number of required votes for remaining parameters (e.g. a radius). Although these approaches can reduce the number of votes considerably, their computational complexity is still far from being considered to be usable in real-time applications since the number of edge pixels in images can be very large.

Another idea is to take into account also the gradient orientation at edge pixels. *Kimme et al.* [KBS75] use this approach to estimate circle segments instead of whole circles. *Guil* and *Zapata* [GZ97] decompose circle detection into two phases. First circle centers are computed from intersections of gradient vectors and then corresponding radii are estimated using distances between circle center and image pixels by voting in an one dimensional histogram. Recently, *Rad et al.* [RFQ03] use pairs of opposite gradient vectors and clustering on Euclid distances to estimate most probable circle parameters. However, these methods have two important drawbacks. First is computational complexity hidden in the pre-processing phase where a robust edge detector (e.g. [Can86]) is used in order to estimate precise locations and orientations of edges in noisy images. The second issue is caused by the existence of negligible gradient magnitudes on ball boundaries due to shading and shadow casting. Their presence hinders a correct estimation of the orientation and forces lower gradient magnitude thresholds that increase the overall number of edge pixels that have to be processed.

Our approach is inspired by methods based on randomized access to input data. This technique was first introduced to circle detection by *Xu et al.* [XOK90] who iteratively take several randomly chosen triplets of edge pixels, construct a circle and use its three parameters to vote in 3D histogram represented by dynamic data structure. *Cheng* and *Lee* [CL95] and later *Chen* and *Chung* [CC01] use RANSAC based voting instead of blind CHT where the quality of each circle candidate is evaluated using distance of image pixels from counted circle boundary. This procedure is repeated several times and a circle with smaller error is taken. Although randomized approaches gain significant speed-ups, they still require many random votes to perform when the number of edge pixels is large, i.e. when the gradient magnitude thresholds are low (as in our case). However, as we show in this paper, color-based edge detection allows to reduce the number of edge pixels so that randomized voting becomes computationally tractable and more efficient as compared to other techniques.

Since approaches based on CHT allow only pixel precise estimation of circle center and radius least squares fitting techniques are used [GGS94] to gain sub-pixel accuracy. However, these methods are very sensitive to outliers and so they have to be used carefully. In practice it is necessary to first preselect a meaningful subset of edge pixels using some robust (e.g. CHT based) techniques. The same strategy is used also in our approach, but in addition to previous approaches we incorporate also weighting by gradient magnitude to improve the accuracy.

Balls were used as 3D interaction devices in few related works. *Bradley* and *Roth* [BR05] use a blue ball covered with red and green dots to estimate position and orientation. Their real-time ball detection algorithm is based on a simple color thresholding and bounding circle estimation that is unfortunately sensitive to background clutter and occlusions. *Greenspan* and *Fraser* [GF03] introduced a more robust ball tracking algorithm for sphere dipole pose estimation. In this case dipole means two balls connected with a stick yielding five degrees of freedom. Their algorithm has two phases. First is locking (1 to 2 seconds) that allows to find balls in general positions and the second phase is real-time tracking that takes into account the previous position of the balls. The main disadvantage of this approach is a locking phase which introduces noticeable lags in interaction.

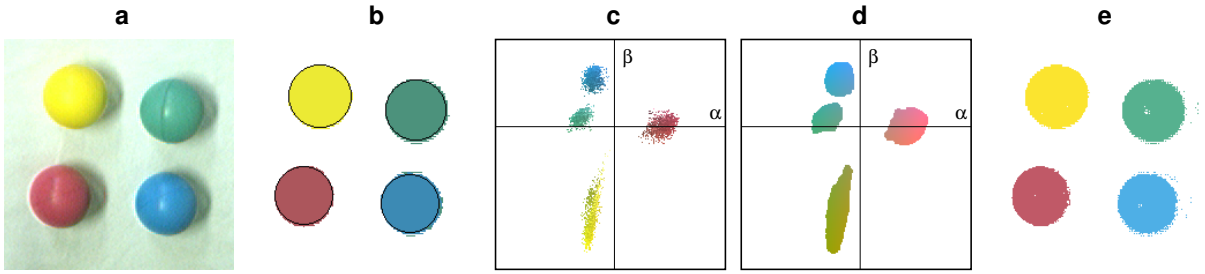


**Figure 1:** ***Color calibration****: (a) input color image, (b) result of mean-shift color image segmentation and circle fitting, (c) measured non-parametric color distribution, (d) four distinct color clusters, (e) resulting pixel-wise classification.*

## 3. Our approach

In this section we introduce our novel approach to real-time ball tracking. The key idea that actually enabled both robustness and high processing speed is the usage of color edge detection based on a fast color segmentation that produces a much lower number of edge pixels in contrast to standard approaches based on luminance. This reduction dramatically decreases the number of votes required for robust detection of circle parameters so that the pose of many color balls can be estimated in real-time.

The whole framework consists of two main phases. In the first off-line calibration phase the camera's intrinsic parameters and radial distortion are estimated and a simple color classifier is learned from an exemplar image of color balls. Then the on-line real-time tracking phase follows where the color classifier is applied to the input images, balls are detected and their 3D positions are returned. In the following sections we describe each phase in more detail.

### 3.1. Calibration

The off-line calibration phase consists of two steps. First intrinsic and radial distortion parameters of the camera are estimated from several photographs of a chessboard pattern taken under different orientations and positions. We exploit the *GML Camera Calibration Toolbox* [VV05] to perform image analysis and optimization. The extracted camera parameters are used immediately to precalculate a look-up table that allows unwrapping pixel positions at runtime.

In the second step we estimate the color distribution of each ball that should be recognizable during the real-time tracking phase. For that purpose we arrange color balls to be well visible from the camera and acquire a single color image (see Figure 1a). Then we perform color image segmentation using a modified version of the popular mean-shift based algorithm [CM97]. In our case only color constancy of regions is important, therefore we perform mean-shift segmentation regardless luminance using only color components (of *LUV* color space [WS82]). This modification provides robustness to brightness variation across the ball's surface (see Figure 1b).

When the input image is segmented we perform connected components analysis [RK82] to label regions. On each sufficiently large region a circle fitting algorithm is applied (see Section 3.2.2). When the ratio between the region area and the area of a fitted circle is nearly equal to one, we determine this region as a ball. Afterwards pixels inside the detected circle serve as color samples to non-parametric distribution (see Figure 1c) represented by a 2D image where rows and columns represent color components $\alpha$ and $\beta$ of a selected color space (in our implementation we use *a* and *b* from CIE *Lab*, however, any other color space with decoupled color and intensity components can be used, e.g. $l\alpha\beta$ [Rud98]).

When all color samples are gathered, an image closing operation [Ser93] is applied to fill-in small holes and filter noise (see Figure 1d). Finally, the resulting non-parametric distribution of colors is used to precalculate a simple *RGB* classifier that allows converting input color to a single index. The classifier is implemented as a 3D look-up table that converts *RGB* triplets to integer values. To avoid huge memory requirements a 6-bit representation is used for each color component). Figure 1e shows the result of image segmentation when the *RGB* classifier is applied on the input image.

### 3.2. Real-time tracking

The real-time tracking phase consists of four main steps: (1) color image segmentation where the input image is converted to several regions, (2) robust estimation of circle parameters, (3) refinement of circle parameters, and (4) ball to track assignment where each circle is assigned to a specific instance of a ball track to ensure temporal consistency. In the following sections each step is described in more detail.

#### 3.2.1. Segmentation

First we acquire an image from the camera (see Figure 2a) and apply the *RGB* classifier to obtain a unique color index

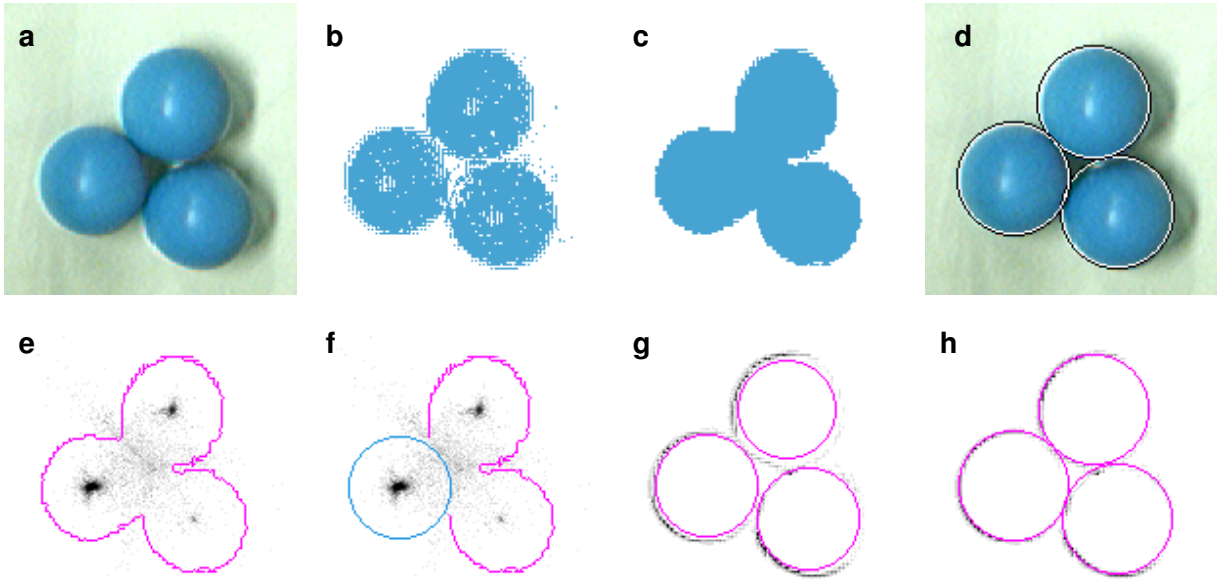


**Figure 2:** ***Ball detection****: (a) input color image, (b) pixel-wise classification, (c) noise reduction and hole filling, (d) superimposed detected balls, (e) region boundaries and circle center histogram, (f) first detected circle and removed part of the original boundary, (g) all detected circles and close image gradients, (h) circles refitted to real ball boundaries.*

for each pixel (see Figure 2b). Then as in the color calibration phase the connected component analysis [RK82] is used to obtain a list of regions and their neighboring relations. To reduce noise and remove holes small regions are connected to their sufficiently large neighbors. After this we obtain a set of regions where each can represent one or more balls (see Figure 2c). For the following processing each region is represented by a set of undistorted boundary pixels.

The problem is now much simpler as compared to general circle detection from luminance edges since now we have to search only in a very restricted subset of the 3D parametric space. However, still a robust estimation is necessary to handle multiple balls per region, occlusion and background clutter.

In our experiments we implemented and tested many different variants of Hough transform, RANSAC, and least-squares based approaches. Finally we decided to develop a new approach where in the first phase an initial circle is robustly estimated using blind randomized voting with dimensionality reduction of the parametric space and then the circle parameters are refined to better fit real ball boundaries using a least-squares technique. Such a combination allows to reach an optimal balance between robustness, accuracy and processing speed. Each step of the proposed algorithm will be described in more detail in the following sections.

#### 3.2.2. Robust estimation of circle parameters

As was stated before, the basic assumption is that the boundaries of detected regions trace approximately real ball boundaries. Therefore, if we perform many random picks of tree boundary pixels and construct a circle which passes them, then there is a high probability that we obtain similar circles repeatedly. An important fact is that computing the location of the circle center $(c_x, c_y)$ from three points $(x_i, y_y)$ (with integer coordinates) can be done very fast (2 integer divisions and 18 multiplications):

$$c_x = \frac{d_1 y_{32} + d_2 y_{13} + d_3 y_{21}}{2(x_1 y_{32} + x_2 y_{13} + x_3 y_{21})} \quad (1)$$

$$c_y = \frac{d_1 x_{32} + d_2 x_{13} + d_3 x_{21}}{2(y_1 x_{32} + y_2 x_{13} + y_3 x_{21})} \quad (2)$$

where $x_{ij} = x_i - x_j$, $y_{ij} = y_i - y_j$, and $d_i = x_i^2 + y_i^2$. Therefore we can perform many random votes and gather solutions in a 2D histogram with the same resolution as the original image (see Figure 2e). Voting is stopped when the accumulator in the currently updated bin crosses a predefined limit or when the maximum number of random votes exceeds. The latter case is rare and occurs when balls with the same color form blobby regions as shown in Figure 2c.

With the estimated circle center we proceed to a simple deterministic vote for the radius, i.e. we compute the distance (in pixels) of each region boundary pixel to the circle center and increment the accumulator of the correspond-

ing bin in a 1D histogram. Finally we treat the index of the most frequented bin as an estimation of the circle radius. We can do this estimation also during the previous random voting phase; however, in this case the computation takes two more multiplications and a floating point square root which is computationally demanding in contrast to the proposed deterministic vote.

When the most salient circle is estimated, we have to ensure that the underlaying region does not encompass any other ball with the same color index (as in Figure 2e,f). This can be done by searching for another maximum in the 2D histogram of circle centers. However, when the second circle occupies only a small portion of the whole region boundary then there is also a small probability that the center bin will have a sufficient number of votes. To overcome this issue, we simply remove boundary pixels (see Figure 2f) near the first estimated circle and perform repeatedly circle center and radius voting phases till the remaining number of boundary pixel become considerably low.

Another problem arises when non-circular objects with similar colors appear in the image. In this case the circle center voting phase typically exceeds the maximum number of random votes, the voting histogram is noisy without significant peaks and the global maximum is considerably small (see Figure 3). To recognize this situation we propose a simple but robust circle quality metric:

$$Q_c = \frac{c_{max} \cdot r_{max}}{N_{votes} \cdot N_{points}} \tag{3}$$

Here $c_{max}$ and $r_{max}$ are global maxima in the circle center and in the radius voting histograms, $N_{votes}$ is the number of circle center votes, and $N_{points}$ is the number of region boundary pixels. Typically $Q_c$ for nearly circular region is about four orders of magnitude higher as compared to non-circular regions. The most doubtful situation is the case where balls form blobby regions (like in Figure 2c), nevertheless also in this case the $Q_c$ is about one order of magnitude higher.

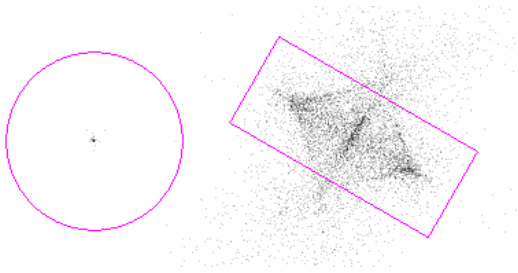

**Figure 3:** ***Circle center voting histogram****: When the region has a circular shape the histogram has one significant peak (left); in the case of a non-circular region the histogram is noisy and there are no significant peaks (right).*

### 3.2.3. Refinement of circle parameters

When the robust estimation of a circle is done we have to refine its center and radius to better fit the real ball boundary. To simplify this task we assume that pixels representing ball boundaries have a large magnitude of intensity gradient and are simultaneously close to the estimated circle. According to this basic assumption we apply simple central differences on pixels intensities in a small annulus around the estimated circle (see Figure 2g) and select pixels with large responses. In general this approach does not guarantee that we actually select real boundary pixels since occlusions and background clutter can easily violate our basic assumption. However, the shape of the annulus usually restricts the selection area in such a way that outliers do not affect the final solution considerably.

We formulate the task as a non-linear least-square optimization problem with the following energy function:

$$\mathrm{E}(\mathbf{c}) = \sum_p \|\nabla \mathrm{I} \cdot \mathrm{C}(\mathbf{c})\|^2 \tag{4}$$

where $\nabla \mathrm{I}$ is the image gradient at pixel $\mathbf{p}$ and $\mathrm{C}(\mathbf{c})$ is the distance of pixel $\mathbf{p}$ from circle $\mathbf{c}$:

$$\mathrm{C}(\mathbf{c}) = \sqrt{(p_x - c_x)^2 + (p_y - c_y)^2} - c_r \tag{5}$$

In order to minimize such a function we use a standard iterative Gauss-Newton method [PTVF92], where in each iteration first order derivatives of (4) are set to zero and the function is linearized using a first order Taylor expansion:

$$\nabla \mathrm{E}(\mathbf{c}) = 0 \quad \Rightarrow \quad 2 \sum_p \nabla \mathrm{C} \cdot \|\nabla \mathrm{I}\|^2 \cdot (\mathrm{C} + \nabla \mathrm{C} \cdot \Delta \mathbf{c}) = 0 \tag{6}$$

where

$$\nabla \mathrm{C} = \left( \frac{1}{d}(c_x - p_x), \frac{1}{d}(c_y - p_y), -1 \right) \tag{7}$$

and

$$d = \sqrt{(p_x - c_x)^2 + (p_y - c_y)^2} \tag{8}$$

From this equation we can easily obtain the incremental change of the circle parameters (in matrix notation):

$$\Delta \mathbf{c} = -(\nabla \mathbf{C}^t \cdot \mathbf{W} \cdot \nabla \mathbf{C})^{-1} (\nabla \mathbf{C}^t \cdot \mathbf{W} \cdot \mathbf{C}) \tag{9}$$

where $\nabla \mathbf{C}$ is a `3xN` matrix of first order derivatives (7), $\mathbf{W} = diag(\|\nabla \mathrm{I}\|^2)$ is a diagonal `NxN` weighting matrix, and $\mathbf{C}$ is an `1xN` column vector of point distances from circle $\mathbf{c}$ (5).

In practice the computation of (9) is very fast even for a large number of pixels since only a closed form `3x3` matrix inversion is computed and only one iteration is necessary to reduce the misalignment to sub-pixel accuracy (see Figure 2h). Moreover, in our experiments we observed that following iterations can be harmful when background clutter and outliers appear in the original image; therefore we suggest using only one iteration even if CPU power is not that expensive.

Once the circle parameters $c_x$, $c_y$, and $c_r$ are known the 3D position of the ball center can be estimated using a full perspective model (as in [GF03]):

$$x = c_x \frac{z}{f}, \quad y = c_y \frac{z}{f}, \quad z = r\sqrt{1 + f^2/c_r^2} \qquad (10)$$

where $r$ is the radius of a real ball, and $f$ is the camera's focal length.

#### 3.2.4. Ball to track assignment

To track ball instances consistently and to avoid short-term detection failures caused by occlusions or sudden light condition changes (e.g. due to fluorescent light flickering), we take a temporal history of ball positions into account.

To do that a nearest neighbor track assignment with temporal hysteresis is used. Each detected ball is assigned to the nearest active track. When all active tracks are occupied and still some detected balls remain, a new track instance is created and switched to active mode when it remains occupied for several frames. Conversely the track is deactivated when it has not been occupied for a longer period of time. In common practice this mechanism allows to avoid flickering and sudden interchange of the augmentation when different virtual objects are assigned to balls with the same color index.

## 4. Results

The proposed algorithm has been implemented and tested in several practical scenarios. In this section we discuss performance, accuracy as well as limitations arising from performed experiments and present several exemplary applications.

### 4.1. Performance & accuracy

The main advantage of the proposed ball tracking algorithm compared to previous approaches is the overall processing speed. In average it takes only 11 ms to analyze a 0.3 Mpix image with 8 balls (see Figure 4) using one core of a *Intel Pentium D 920* (2.8 GHz, 800 MHz FSB, 2 MB Cache). In simpler scenarios even better processing speed can be achieved; therefore another real-time tracking engine can be incorporated into the processing pipeline (in our implementation we combine ball tracking with *ARToolKitPlus* and reach 20 ms for 8 balls and 10 planar markers in a 0.3 Mpix image).

However, still the processing speed decreases linearly with the number of detected balls. The pixel classification and the connected component analysis consume in average 6 ms per frame and can be considered as a constant load. The estimation of the ball position has the most significant influence on processing speed. It takes in average 0.5 ms per ball for moderately large regions (about 300 boundary pixels) when the circle center accumulator threshold is set to 16. In order to reach this value it is in average necessary to perform about 300 iterations of randomized circle center voting.

The accuracy of the algorithm depends mainly on camera resolution, lighting conditions and extent of ball occlusion. In our experiments we used low-cost *Fire-i* color cameras running at resolution of 640x480 (YUV 4:1:1). Thanks to sub-pixel accurate circle refinement (Section 3.2.3) the deviation of real and estimated ball position is typically below 1 mm. This value holds for not occluded daylight illuminated ball with diameter of 7 cm positioned on the table at a distance 1 m from the camera (diameter 60 pixels in the image space). In the case of strong occlusion (more than 50%) and/or bad light conditions, the deviation from the real position can be much greater.

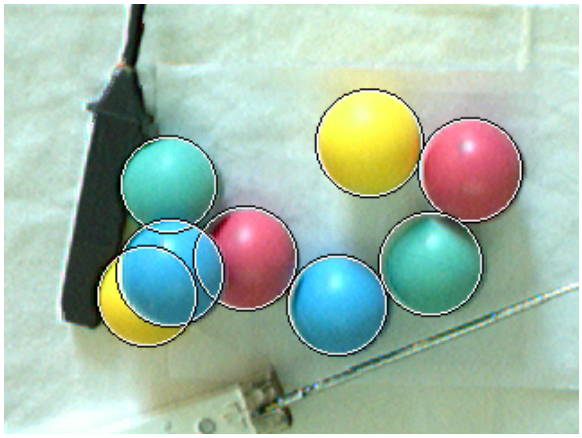

**Figure 4:** ***Performance analysis****: The image used for performance analysis of the proposed algorithm. It takes 11 ms to estimate all 3D positions of these 8 color balls.*

### 4.2. Limitations

The main limitation of the proposed algorithm is the assumption on a constant color spectrum of incoming light. This limitation is closely related to the number of distinct colors that should be recognized. There is a trade off between number of active colors and size of color clusters. Larger clusters bring more robustness to color shifts, but decrease number of distinct colors and increase probability of collision with background objects. According to our experiments 4 distinct color clusters bring optimal trade off between number of colors and robustness of the system. When daylight is used for illumination and more colors are needed it is necessary to run color calibration phase several times during the day.

Another important limitation is the accuracy of ball depth estimation. When a ball is far from the camera or strongly occluded, the estimation of the circle position and radius can be biased and so the resulting depth can be notably different from real depth. A similar situation occurs also when bad lighting conditions and/or shifts in light spectrum cause noisy color classification. This type of deviation is usually

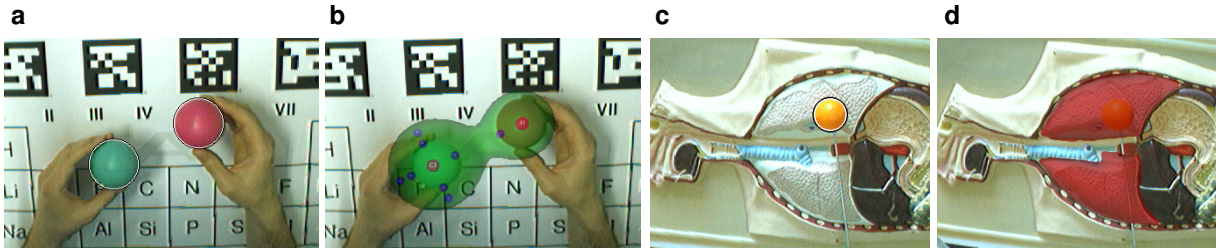


**Figure 5:** ***Applications*** *— Chemical scenario: (a) manipulating atoms of H and Cl represented by two color balls, (b) computer generated visualization of chemical reaction. Pointing device: (c) telescopic antenna equipped with color ball on the top, (b) computer generated visualization of lungs appears when the ball touched the torso.*

transient and so it can be suppressed by low-pass or *Kalman* filtering [Kal60] at the expense of motion latency.

### 4.3. Applications

To show that our color ball tracking fits the needs of near-field real-time AR applications as well as is suitable for doing interactions, we integrated our software into an interactive seated AR display called *Spinnstube* [WRB07]. While attaching the system to a table a user sitting in there looks through a half-silvered mirror in front of him/her. Interactions with objects located on the desktop are observed by a camera looking from atop.

Using this hardware platform we implemented an AR learning tool for teaching basic principles of chemical reactions. To give a user a proper haptic feedback color balls are used to represent atoms from a subset of a periodic table whereas the augmentation shows electrons in the valence layers of that specific atom. In combination with a printed version of the periodic table equipped with planar *ARToolKitPlus* markers a user is able to select atoms and build up simple molecules (see Figure 5, top row). As compared to a related AR-based chemistry teaching system [FFE*07] (based only on *ARToolKit*) our approach brings great benefit since balls themselves have a special semantic meaning and thus they are not just markers additionally attached to the real object where the system augments on.

Another application in the context of AR where a ball can have a special semantic meaning is to use it as a pointing device. In this case a ball provides a compelling visual metaphor as pointer and a direct haptic feedback important for the pointing operation. Using an appropriate hardware construction (e.g. a telescopic antenna) fixed to a suitable input device (e.g. a wireless presenter with some buttons used as triggers) an interaction device for point-and-click events can be created easily. The pictures on the right in Figure 5 show such a device used to highlight virtual organs augmented on a human torso model.

Furthermore, our color ball tracker can be also combined with previous ball-based tracking techniques providing upon the 3D position also an estimation of the orientation, i.e. dipole [GF03] and 6-DOF sphere [BR05]. As compared to ball detection algorithms used in these papers our approach is easier to implement, increases robustness and brings lower computational overhead.

## 5. Conclusion and Future work

In this paper we presented a novel color ball tracking technique suitable for cost effective AR systems. Using a standard color camera it allows to estimate 3D positions of several visible color balls in real-time. As compared to planar fiducial markers our approach is robust to partial occlusion featuring a more comfortable manipulation, a natural visual metaphor and a proper haptic feedback. Moreover, the proposed technique is fast enough to be easily combined with other real-time tracking engines such as *ARToolKitPlus*. So we hope it has potential to become a complementary tracking alternative for low-cost AR systems.

As future work we plan to develop an on-the-fly color recalibration phase to increase the robustness to slight changes in color constancy of incoming light. Another issue deserving attention is the accuracy of the ball radius estimation especially in case of considerable occlusion. In addition to that we are also looking for other interesting applications where a color ball will have a natural semantic meaning similar to the proposed pointing device and the equalization with atoms.

## 6. Acknowledgements

This work has been co-funded by the European Commission as part of the IST programme within the 6th framework (project ARiSE, contract number `IST-027039`) and partly supported by the Ministry of Education, Youth and Sports of the Czech Republic under the research program `LC-06008` (Center for Computer Graphics).

The authors would like to send thanks to *Jiří Žára*, *Jürgen Wind*, *Vlastimil Havran*, *Jan Ondřej*, and *Ondřej Jamřiška* for their devoted support during the work on this paper.

## References


[BKJ05] BENCINA R., KALTENBRUNNER M., JORDA S.: Improved topological fiducial tracking in the reactivision system. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition* (2005), vol. 3, pp. 99–106. 1

[BR05] BRADLEY D., ROTH G.: Natural interaction with virtual objects using vision-based six DOF sphere tracking. In *Proceedings of the International Conference on Advances in Computer Entertainment Technology* (2005), pp. 19–26. 1, 2, 7

[Can86] CANNY J.: A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence 8*, 6 (1986), 679–698. 2

[CC01] CHEN T.-C., CHUNG K.-L.: An efficient randomized algorithm for detecting circles. *Computer Vision and Image Understanding 83*, 2 (2001), 172–191. 2

[CF04] CLAUS D., FITZGIBBON A.: Reliable fiducial detection in natural scenes. In *In Proceedings of European Conference on Computer Vision* (2004), pp. 469–480. 1

[CL95] CHENG Y. C., LEE S. C.: A new method for quadratic curve detection using K-RANSAC with acceleration techniques. *Pattern Recognition 28*, 5 (1995), 663–682. 2

[CM97] COMANICIU D., MEER P.: Robust analysis of feature spaces: Color image segmentation. In *Proceedings of Conference on Computer Vision and Pattern Recognition* (1997), pp. 750–755. 3

[DH72] DUDA R. O., HART P. E.: Use of the hough transformation to detect lines and curves in pictures. *Communications of the ACM 15*, 1 (1972), 11–15. 2

[FFE*07] FJELD M., FREDRIKSSON J., EJDESTIG M., DUCA F., BÝTSCHI K., VOEGTLI B., JUCHLI P.: Tangible user interface for chemistry education: Comparative evaluation and re-design. In *Proceedings of SIGCHI Conference on Human Factors in Computing Systems* (2007), pp. 805–808. 7

[Fia05] FIALA M.: ARTag, a fiducial marker system using digital techniques. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition* (2005), vol. 2, pp. 590–596. 1

[GF03] GREENSPAN M., FRASER I.: Tracking a sphere dipole. In *Proceedings of International Conference on Vision Interface* (2003), pp. 154–161. 1, 2, 6, 7

[GGS94] GANDER W., GOLUB G. H., STREBEL R.: Least-squares fitting of circles and ellipses. *BIT 34* (1994), 558–578. 2

[GZ97] GUIL N., ZAPATA E. L.: Lower order circle and ellipse hough transform. *Pattern Recognition 30*, 10 (1997), 1729–1744. 2

[JC05] JIANG X., CHENG D.-C.: Fitting of 3D circles and ellipses using a parameter decomposition approach. In *Proceedings of International Conference on 3-D Digital Imaging and Modeling* (2005), pp. 103–109. 2

[Kal60] KALMAN R. E.: A new approach to linear filtering and prediction problems. *Journal of Basic Engineering 82* (1960), 35–45. 7

[KB99] KATO H., BILLINGHURST M.: Marker tracking and HMD calibration for a video-based augmented reality conferencing system. In *Proceedings of International Workshop on Augmented Reality* (1999), pp. 85–94. 1

[KBS75] KIMME C., BALARD D., SKLANSKY J.: Finding circles by an array of accumulators. *Communications of the ACM 18*, 2 (1975), 120–122. 2

[LLL86] LI H., LAVIN M. A., LEMASTER R. J.: Fast hough transform: A hierarchial approach. *Computer Vision, Graphics and Image Processing 36*, 2/3 (1986), 139–161. 2

[PTVF92] PRESS W., TEUKOLSKY S., VETTERLING W., FLANNERY B.: *Numerical Recipes in C: The art of scientific computing*. Cambridge University Press, 1992. 5

[RFQ03] RAD A. A., FAEZ K., QARAGOZLOU N.: Fast circle detection using gradient pair vectors. In *Proceedings of International Conference on Digital Image Computing* (2003), pp. 879–888. 2

[RK82] ROSENFELD A., KAK A. C.: *Digital Picture Processing*, vol. 1. Academic Press, Orlando, USA, 1982. 3, 4

[Rud98] RUDERMAN D. L.: Statistics of cone responses to natural images: Implications for visual coding. *Journal of Optical Society of America 15*, 8 (1998), 2036–2045. 3

[Ser93] SERRA J.: *Image Analysis and Mathematical Morphology*. Academic Press, 4th Edition, 1993. 3

[VV05] VEZHNEVETS V., VELIZHEV A.: GML C++ camera calibration toolbox, 2005. http://research.graphicon.ru/calibration. 3

[WRB07] WIND J., RIEGE K., BOGEN M.: Spinnstube: A seated augmented reality display system. In *Proceedings of Eurographics Symposium on Virtual Environments* (2007), pp. 17–23. 7

[WS82] WYSZECKI G., STILES W. S.: *Color Science: Concepts and Methods, Quantitative Data and Formulae*. Wiley, 2nd Edition, 1982. 3

[WS07] WAGNER D., SCHMALSTIEG D.: ARToolKitPlus for pose tracking on mobile devices. In *Proceedings of Computer Vision Winter Workshop* (2007). 1

[XJ02] XIE Y., JI Q.: A new efficient ellipse detection method. In *Proceedings of International Conference on Pattern Recognition* (2002), vol. 2, pp. 957–960. 2

[XOK90] XU L., OJA E., KULTANEN P.: A new curve detection method: Randomized hough transform (RHT). *Pattern Recognition Letters 11*, 5 (1990), 331–338. 2