Title

Robust Real-Time Detection Of Multi-Color Markers On A Cell Phone

Authors

Bagherinia, Homayoun

Manduchi, R

Publication Date

2023-12-12

Peer reviewed

J Real-Time Image Proc
DOI 10.1007/s11554-011-0206-9

ORIGINAL RESEARCH PAPER

Robust real-time detection of multi-color markers on a cell phone

Homayoun Bagherinia · Roberto Manduchi

Received: 9 November 2010 / Accepted: 17 May 2011

Abstract We describe a fast algorithm to detect special multi-color markers with a camera cell phone. These color markers can be used for environmental labeling, for example, as a wayfinding aid for persons with visual impairment. Using a cascade of elemental detectors, robust detection is achieved at an extremely low computational cost. We also introduce a strategy to select surfaces for the marker that ensure very low specular reflection, thus facilitating color-based recognition.

Keywords Color constancy · Mobile vision · Cascade classifiers · Assistive technology · Fiducial design

1 Introduction

Camera-equipped programmable cell phones have become the platform of choice for a wide variety of mobile computer vision applications, including augmented reality [26], gaming [31], mobile OCR (http://www.knfbreader.com), and barcode reading [10]. Our work is motivated by a specific goal: helping a blind person to find their way around in a suitably equipped environment. Specifically, our system is based on simple 'markers', easily detectable by a cell phone, that can be placed in key locations in the environment. A blind person can search for such markers by orienting the camera phone in different directions, effectively 'scanning' the environment. Once a marker is detected by the camera phone, the user is prompted by an acoustic signal. If desired, the user can move towards the marker (which could be placed near a point of interest) by keeping track of the marker location via the camera phone. The marker may also contain a certain amount of information, for example, in the form of an ID that can be used as a query to a locational database. In this way, the user could be provided with turn-by-turn instructions to reach a specific destination.

Our system uses multi-colored pie-shaped markers, specifically designed for fast recognition via mobile vision (see Fig. 1). Normally, color-based recognition requires some sort of color constancy operation to deal with varying and unknown illuminants [11]. In our case, this is not necessary because the colors of the different surfaces in the marker are approximately co-variant with respect to changes in illumination. Because no pre-processing is necessary, our color-based detection algorithm is inherently fast. For added speed, a cascaded scheme is implemented. Most pixels are ruled out by the first stages of the cascade, which reduces the overall average computational cost. Further processing stages filter out any remaining false detections and compute the approximate distance to the marker (by measuring the amount of foreshortening).

We introduced our marker design elsewhere [5], along with a very simple detector and a post-processing (segmentation) algorithm [6]. User studies with blind testers of this system have been reported in Manduchi et al. [17]. In this contribution, we present a new marker detection algorithm, which is more efficient and accurate than previous approaches, while achieving high computational efficiency. For example, our system only needs to perform 1.1 multiplication and additions and 1.65 comparisons per pixel (on average) when searching for a color marker with 98% correct detection rate and 0.001% false positive rate. Note that the false alarms rate is then reduced further via geometry-based processing [6]. On a Nokia N95 8GB cell phone processing images at VGA resolution, we

H. Bagherinia (✉) · R. Manduchi
University of California, Santa Cruz, USA
e-mail: hbagheri@soe.ucsc.edu

R. Manduchi
e-mail: manduchi@soe.ucsc.edu

Published online: 03 June 2011

Fig. 1 Our color marker, seen from different viewpoint and with different camera rotations. The *blue superimposed crosses* represent a fixed-size 'probe'

achieve an effective frame rate of 8 fps (frames per second) when no marker is visible in the scene, which reduces to 5 fps (due to shape analysis) when a marker is visible. We also describe a simple technique to choose surfaces for the marker that have good Lambertian characteristics, and thus small specular reflection. Reducing the effect of specular reflection is critical for our color-based detection algorithm, as specular reflection may change the perceived color of a surface in a way that is difficult to model.

This paper is organized as follows. We first review the related work in Sect. 2. Our approach to marker design is described in Sect. 3. We review the color rendering models used in our work (including linearization) in Sect. 4. Our detection algorithm is described in detail in Sect. 5, and experimental results are given in Sect. 6. An experimental comparison between our color marker and the popular ARToolKit marker is presented in Sect. 7. Section 8 has the conclusions.

2 Related work

Image-based labeling has been used extensively for product tagging (1-D and 2-D barcodes) and for robotic positioning and navigation. The ubiquitous 1-D barcode (e.g. UPC and EAN) and 2-D barcode (e.g. Semacode and Shotcode) are designed so as to contain a high density of information. However, barcodes are not the ideal choice for environmental labeling when the goal is quick detection from various distances in possibly complex and cluttered environments.

A number of markers (or *fiducials*) have been proposed enabling fast and robust detection, for applications such as augmented reality and robotic localization. The majority of the markers described in the literature use black-and-white patterns [e.g. 4, 8, 12, 13, 19, 21, 22, 30]. In this case, detection relies on the shape properties of the pattern in the marker. Relatively not much attention has been given to design concepts that rely on distinctive color, rather than (or in addition to) shape analysis [e.g. 3, 14, 25]. Unfortunately, these systems have been tested mostly in laboratory conditions, where one has the possibility to choose colors that are uniquely identifiable (possibly after color correction to account for different illuminants [14]). Without careful color choice and proper processing, color marker detection is likely to fail in realistic conditions with multiple distractors and large variation in illumination. The color markers presented in this paper use multiple colors to increase distinctiveness, and are detected by a model-based algorithm that takes explicitly into account the variability of illumination.

The use of color has also been proposed for increasing the capacity of 2-D bar codes. Examples include Microsofts High Capacity Color Barcode (HCCB) technology [20] and Colorzip Media's ColorCodes technology (http://www.colorzip.com).

3 Marker design

Our color markers are pie-shaped with four colored sectors. We briefly justify our design choice in the following; for an in-depth treatment, the reader is referred to [5]. Marker detection is obtained by sliding a 'probe' across the image, where a probe is the set of four pixels at the vertices of a square (with fixed size of a few pixels). Figure 1 shows a picture of our marker with a probe superimposed. When the probe is placed at or near the center of the image of a marker, each pixel in the probe records a color value from a different sector of the marker. A carefully designed classifier determines whether a given probe is on a marker or not. The advantage of the pie-shaped design is that the same probe size can be used regardless of the apparent (foreshortened) size of the marker (see Fig. 1). In addition, the probe does not need to be perfectly aligned with the marker for detection. Theoretically, a rotation of up to $\pm 45°$ around the camera's optical axes would still enable detection, as the

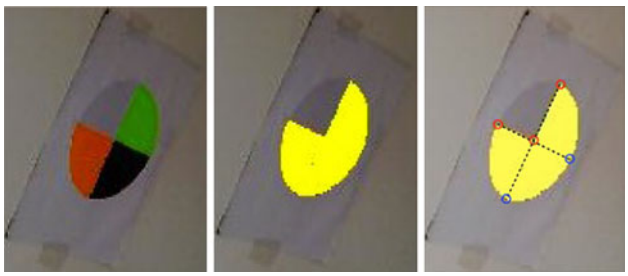


Fig. 2 A color marker (*left*) and its segmentation (*center*) using a fast greedy region growing algorithm [6]. Based on this segmentation, five points of known position can be identified (*right*). The points *circled* in *red* are detected as the three corners of the segmentation mask. By extending lines from the outer corners through the center corner, and computing their intersection with the mask's edge, two more points are identified (*circled in blue*). The location of the four outer points can be used to estimate the homography between the marker pattern and the camera image plane

points in the probe would still fit within the correct sectors. In practice, the system works well if the rotation is within $\pm 40°$ (see Fig. 1). Note that more color sectors would increase distinctiveness of the marker with respect to the background, and thus more reliable detection. However, this would affect the rotation invariance properties, as a smaller rotation angle would cause the probe to fall outside of the correct sectors. In addition, if the marker's real estate is divided into too many sectors, the number of pixels within each sector's image is reduced, which in turn reduces the maximum distance for detection due to foreshortening. Our four-color markers have a diameter of 15 cm and can be detected at a distance of about 4.5 m using a Nokia N95 8GB cell phone (with a field of view of 45° by 35°) using images with VGA resolution (640×480). This marker's size and detection distance are acceptable for our application.

The four color values in the probe are fed into a classifier, which decides whether or not they are consistent with the image of a marker. This operation needs to be very fast, as it is performed at each pixel in the image. Further processing is then performed at all locations that passed this initial detection step. The largest cluster of nearby locations is extracted, and if its size is larger than a threshold, the whole marker image area is segmented out using a fast greedy region growing algorithm seeded with the cluster center. A number of shape consistency tests are then performed on the segmented region to reduce the risk of false alarms. Analysis of the segmentation mask enables robust identification of four non-collinear points of known position (see Fig. 2) which can be used to estimate the homography between the marker pattern and the camera image plane, and thus the camera pose. The spatial ordering of the colors in the marker can be permuted [6] providing a very simple way to embed information within the pattern itself (at an added computational cost—see Sect. 5.5).

At the core of the marker recognition algorithm is the initial color-based detection. It is critical that the detection rate be high, since a missed marker cannot be recovered by subsequent stages. The rate of false alarms must also be kept to a minimum. Even though the post-processing operations can rule out a good number of false alarms, these operations (especially the segmentation) are more computationally demanding, and can reduce the effective frame rate if too many false positives need to be examined. Our classification algorithm is introduced in Sect. 5; in the following, we describe our strategy for choosing the surfaces in the marker in such a way so as to facilitate robust recognition.

3.1 Choice of surfaces

In previous work [5], we created markers using a color printer. The colors were chosen according to a criterion of maximum distinctiveness. Extensive experimentation with this marker revealed that the colored surfaces produced by a color printer, far from being Lambertian (matte), typically have a strong specular reflection component. This represents a serious problem for classification. In the presence of specular reflection, a viewpoint-dependent component with the same color as the illuminant adds to the perceived color of the surface from diffuse reflection [23]. Classification thus becomes more challenging, as the specular color component needs to be removed.

To reduce the effect of specular reflection, we decided to create new markers as collages of colored paper with good characteristics of Lambertianity. We purchased 25 colored paper samples from various sources, all with good nominal opacity characteristics. Fifteen images of each individual paper were taken under the same illuminant from multiple viewpoints. The scatterplot of the normalized values $R/(R+G+B)$ versus $G/(R+G+B)$ of the color of all surfaces is shown in Fig. 3. Note that, under an ideal Lambertian model, the normalized color for a given surface should be constant with respect to the viewpoint (as long as the illuminant does not change). In fact, the scatterplot shows that the normalized colors from most of the surfaces we tested are widely scattered. The point scatter within each color cluster is due in large part to the specular component, as a certain amount of illuminant color is recorded along with the surface color. In particular, most point clusters are oriented towards the cluster of the 'white' surface points, whose color is similar to the color of the illuminant.

This scatterplot allows us to quickly identify the surfaces with best Lambertian characteristics as the ones which produce the more compact point clusters. The compactness of a cluster can be measured, for example, by the sum or the product of the singular values of the collection of the normalized color values that form the cluster.

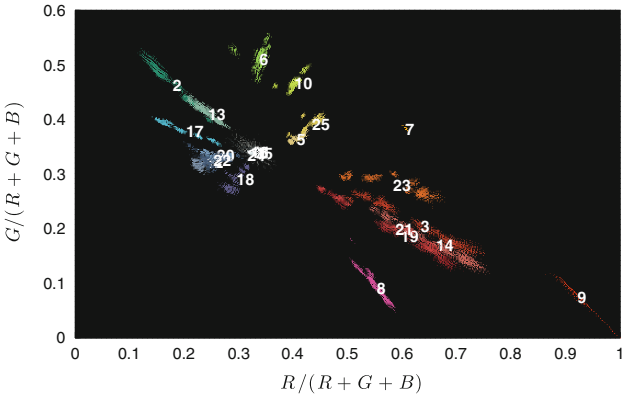


Fig. 3 A scatterplot of the normalized color values of the considered surfaces (the surface's ID is printed in the center of each cluster). Images of the surfaces were taken under the same illuminants from 15 different viewpoints

Based on this analysis, we selected one orange surface (ID = 7) which has an exceptionally compact point cluster in normalized color space (see Fig. 3). We also selected a white and a black surface, following [5]. The white surface is little sensitive to specular reflection (since the color of the diffuse and specular components are similar for a white surface). A white and a black surface nearby in the marker provide a strong albedo gradient that contributes to the marker's distinctiveness. For the remaining patch, we selected a surface on the basis of a combination of criteria: compactness of the point cluster in normalized color space; albedo, measured by the sum of the R, G, and B values for a given illuminant (large albedo values result in higher SNR); distance in RGB space to the other selected colors (large distance means higher distinctiveness). We selected a green patch (ID = 6) which provided a good trade-off based on these criteria. The resulting marker is shown in Fig. 1.

4 Color modeling

Our detection algorithm is based on a model of the perceived color of the four sectors in the marker under varying viewpoint and varying illuminant. We assume that the marker is on a flat surface, and that the illumination can be considered constant on the marker (i.e., that there are no shadow lines crossing it). In the following, we briefly review existing 'color rendering' models that can predict the color of a Lambertian surface when seen under different illuminants. We then describe our strategy for photometric camera calibration and linearization using a gamma model with bias. Finally, we show that the vectors formed by the colors in a probe live in a low-dimensional subspace. This notion is used for the design of the detector, presented in the next section.

4.1 Rendering model

Let us denote by $c_{(s)}^{(i)} = [c_{(s),1}^{(i)}, c_{(s),2}^{(i)}, c_{(s),3}^{(i)}]^T$ the recorded color of the surface of index s in the marker seen under an illuminant indexed by i, where $c_{(s),k}^{(i)}$ represents the kth color channel (R, G or B). As is well known, the recorded color $c_{(s)}^{(i)}$ changes with the illuminant i. These changes must be modeled and taken into consideration for the design of the color-based classifier. The model must consider the spectrum of the illuminant, the reflectance spectrum of the surface, the spectral sensitivity of the color filters in the camera, and the viewpoint and incidence angles. For Lambertian (opaque) surfaces, and assuming that the spectra of all possible illuminants and of all possible surface reflectances live in finite dimensional spaces (of dimension M_i and M_s respectively), the dependence of the surface color on the illuminant is described by the following quadratic form [15, 28]:

$$c_{(s),k}^{(i)} = \alpha^{(i)T} Q_k \beta_{(s)} \tag{1}$$

where Q_k is an $M_i \times M_s$ matrix with positive entries that only depend on the camera and on any linear transformation of the color channels (e.g. white-point calibration); $\alpha^{(i)}$ only depends on the illuminant and on its incidence angle on the surface, as well as on the camera's exposure time and gain; and $\beta_{(s)}$ only depends on the surface. In particular, $\alpha^{(i)}$ and $\beta_{(s)}$ are independent of the color channel k.

It is well known [9] that if $M_s = 3$, the relationship between the color of a surface s when seen under two different illuminants can be expressed as follows:

$$c_{(s)}^{(i_2)} = A^{(i_1,i_2)} c_{(s)}^{(i_1)} \tag{2}$$

where $A^{(i_1,i_2)}$ is a 3×3 matrix that is independent of the surface s. A further simplification is obtained by assuming that $A^{(i_1,i_2)}$ is diagonal (the so-called 'diagonal model'), in which case the color channels transform independently:

$$c_{(s),k}^{(i_2)} = a_k^{(i_1,i_2)} c_{(s),k}^{(i_1)} \tag{3}$$

where a_k is the kth diagonal entry of matrix A. Likewise, under the diagonal model, the color values of two different surfaces s_1 and s_2 seen under the same illuminant i are related by a diagonal matrix that is independent of the illuminant:

$$c_{(s_1),k}^{(i)} = a_{(s_1,s_2),k} c_{(s_2),k}^{(i)} \tag{4}$$

Although the diagonal model (3, 4) is extremely convenient in practice, it should only be considered as an approximation.

In fact, it holds true only if each matrix Q_k has only one non-null column. Finlayson et al. [9] showed that when $M_i = 3$ and $M_s = 2$ (or $M_i = 2$ and $M_s = 3$), there exists an invertible linear color transformation such that the diagonal model holds exactly on the transformed colors.

4.2 Linearization

Ideally, the color values produced by the camera would be linearly related to the recorded color vector $c^{(i)}_{(s)}$ defined in the previous section. In practice, this relation is more complex due to several factors: color values are obtained via a color mosaic; noise adds to the measurements (including quantization noise); non-linear (e.g., gamma) correction compresses the measured values; and white-point correction applies independent rescaling of the color channels. The last two factors (non-linear correction and white-point correction) are dominant and need to be modeled and possibly compensated for before processing. Fortunately, most camera and camera cell phones on the market let the user set white-point calibration to a fixed value. As for the non-linear correction, this can be by-passed in some high-end cameras, but not in cheaper cameras nor in most camera cell phones. Thus, to undo the undesired non-linear processing, it is necessary to "reverse engineer" the system via suitable calibration.

We modeled the non-linear correction performed by the camera as a biased gamma correction [7, 16, 24]:

$$\bar{c}_k = b_k + c_k^{\gamma_k} \tag{5}$$

where $\bar{c}_k$ is the kth channel color value at a given pixel produced by the camera via non-linear transformation of the recorded value c_k. (Note that any constant multiplicative coefficient can be considered to be already absorbed in c_k.) We assume that both the gamma coefficients $\{\gamma_k\}$ and the bias terms $\{b_k\}$ depend on the color channel.

To estimate $\{\gamma_k\}$ and $\{b_k\}$, we used a Macbeth color checkerboard with 24 color chips. The albedo value $v_{(s),k}$ of each color chip s in the checkerboard for each color channel k is known. Thus, under the diagonal model (4), one may expect the recorded color values $c_{(s),k}$ to be proportional to the albedos $v_{(s),k}$ for a given picture of the checkerboard. To increase the variety of sample data, we took pictures of the checkerboard under eight different illuminant conditions. Note that with our camera cell phone (Nokia N95 8GB), we are unable to control the camera's exposure time and the gain, as they are set by the camera's exposure control. Thus, each picture of the checkerboard may have a different (and unknown) value of these parameters, which can be expressed as a constant multiplicative term $h^{(i)}$ (where we highlighted the dependence on the illuminant i, which determines the overall scene brightness and thus the parameters chosen by the automatic exposure control). In addition, due to the varying illuminant, each color channel may undergo a different linear transformation (multiplication by a factor $a_k^{(i)}$) according to the diagonal model (3). In summary, we model the kth color channel value of the sth color patch under the i-th illuminant as:

$$\bar{c}^{(i)}_{(s),k} = b_k + (h^{(i)} a_k^{(i)} v_{(s),k})^{\gamma_k} = b_k + g_k^{(i)} v^{\gamma_k}_{(s),k} \tag{6}$$

where all linear coefficients have been absorbed in $g_k^{(i)}$.

Thus, our calibration task can be expressed as follows: Given a set of measurements $\{\bar{c}^{(i)}_{(s),k}\}$ and known albedos $\{v_{(s),k}\}$, estimate $\{\gamma_k\}$ and $\{b_k\}$ based on (6). The measurements $\{\bar{c}^{(i)}_{(s),k}\}$ were obtained by averaging the color values over hand-drawn rectangles in the images of the color checkerboard.

We obtained a least-squares solution using Matlab's `fminsearch()` minimization function. As part of this procedure, we also estimated the coefficients $g_k^{(i)}$, which allowed us to plot the results of our calibration in Fig. 4. Note that these coefficients are irrelevant for our detection algorithm. The parameters found by our calibration procedure are $\gamma_1 = 0.34$; $\gamma_2 = 0.24$; $\gamma_3 = 0.39$; $b_1 = -52$; $b_2 = -127$; $b_1 = -35$.

4.3 Dimensionality considerations

As mentioned in Sect. 3, the camera collects measurements of 'probes', where a probe is formed by the pixels at the vertices of a square of fixed side length. When a probe is centered at or near the center of the image of a marker, and the camera is correctly aligned, the probe contains color values of all surfaces in the marker. Let the 12-dimensional vector $p^{(i)}$ represent the concatenation of the three vectors (sub-probes) $p_k^{(i)} = [c^{(i)}_{(1),k}, \ldots, c^{(i)}_{(4),k}]^T$, where k from 1 to 3 represents the color channel, and let $\mathcal{S}$ be the linear space spanned by $p^{(i)}$ over varying illuminant i.

Fact 1 *Under the finite dimensional assumption* (1), *the following inequality hold*: $\dim(\mathcal{S}) \leq \min(M_i, 12)$.

Proof From (1) it follows that $p^{(i)T}_{(k)} = \alpha^{(i)T} B_k$, with $B_k = [Q_k \beta_{(1)} | \ldots | Q_k \beta_{(4)}]$. Thus, a generic vector $p^{(i)}$ can be written as $B\alpha^{(i)}$ where $B = [B_1 | B_2 | B_3]^T$. Because the matrices Q_k have size $M_i \times M_s$, we conclude that $\dim(\mathcal{S}) = \text{rank} B = \min(M_i, 3 \times 4)$.

Note that the dimension of the space of illuminants M_i can be safely considered to be <12. For example, Marimont and Wandell [18] showed that three basis functions were enough to model the color observation of 462 Munsell chips. It follows that the probes live in a subspace of dimension equal to M_i. In the case of the diagonal color model, one easily proves the following:

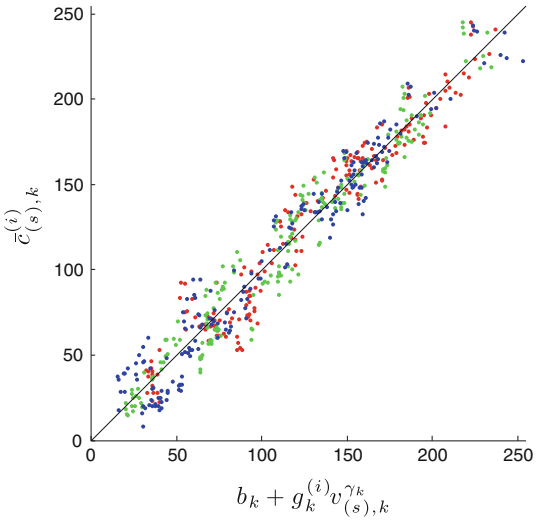


Fig. 4 An illustration of our calibration results based on (6). The predicted values $b_k + g_k^{(i)} v_{(s),k}^{\gamma_k}$ (using the values for γ_k, b_k, and $g_k^{(i)}$ estimated by our procedure and the known albedo values of the color patches $v_{(s),k}$) are plotted against the values $\bar{c}_{(s),k}^{(i)}$ produced by the camera. Each dot represents a color patch under a certain illuminant. The dot's color (*red*, *green*, or *blue*) indicates the color channel (R, G or B) for that value

Fact 2 *If the diagonal color model* (3) *holds, then the space* $\mathcal{S}$ *can be expressed as the direct sum of* $\mathcal{S}_1, \mathcal{S}_2$ *and* $\mathcal{S}_3$*, where* $\mathcal{S}_k$ *is the one-dimensional subspace of* $\mathcal{S}$ *spanned by vectors* $p_{(k)}^{(i)}$ *associated with one color component only*.

Figure 5, top, shows the sorted singular values of a matrix whose columns are the 12-D color probes extracted from 150 pictures of the color marker taken in a variety of indoor and outdoor illumination conditions and under different viewing angles (as described in detail in Sect. 6.1). Any probe that contained a color value larger than 245 was considered saturated and removed from the data set used to create this plot. All values were linearized by inverting the gamma transformation (5). It is seen that the first singular value is much larger than the others. This phenomenon is typical, as the first eigenvector represents the variation in intensity, which dominates other sources of variability. The remaining singular values decrease fairly smoothly, and thus do not provide a strong indication for the dimensionality of $\mathcal{S}$. Looking at the three subspaces $\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3$ (Fig. 5, bottom), spanned by the sub-probes with only one color channel, one again notices a dominant eigenvector in all three cases. In the following, we will assume for simplicity's sake that the diagonal model holds and therefore that the subspaces $\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3$ are one-dimensional.

5 Detection algorithm

The design of the marker detection algorithm requires careful consideration of several factors. Our application

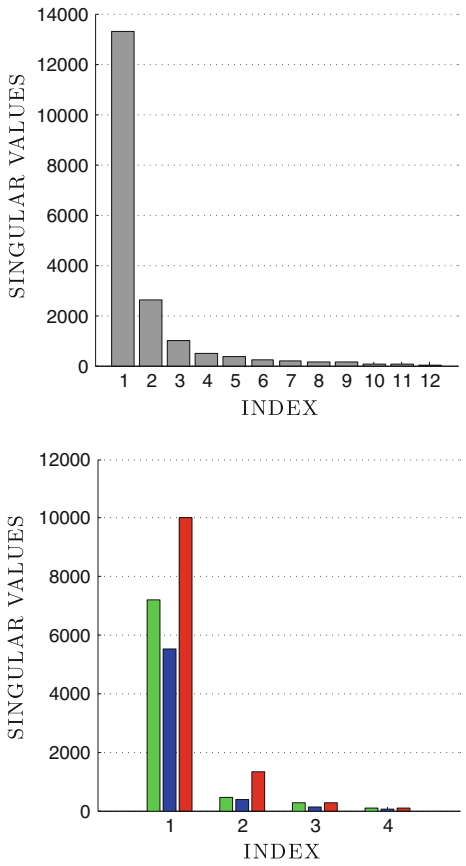


Fig. 5 Sorted singular values of the 12-D color probes (*top*) and of the 4-D sub-probes with only one color channel (*bottom*). The *color* of the *bars* indicates the color channel of the sub-probe

calls for robust detection with a very low rate of misses and false alarms under a wide variety of viewing conditions and of background. At the same time, the algorithm needs to be very light, so as to enable analysis of several frames per second at reasonably high resolution when implemented on a cell phone. In the following, we describe our design choices vis-a-vis these application requirements.

Our detection algorithm is based on a one-class classifier model: it analyzes a probe to determine whether or not it may belong to the image of a marker, without explicitly modeling the 'background' (non-marker) class. The reason for choosing a one-class classifier is that it would be very difficult to produce a general model of the background that can apply to any environment. The typical approach of collecting representative images of the background may not generalize well to previously unseen situations. Fortunately, as discussed earlier, the distribution of color values of the probes is well structured, as the 12-dimensional probe color vectors actually live in a much lower dimensional space. This allows us to use a relatively simple classifier, which is implemented in a cascaded structure [29] for improved efficiency.

5.1 Classifier design

Becuase the color probes are expected to live in a subspace $\mathcal{S}$ of R^{12}, a simple classifier could be obtained by thresholding the Euclidean distance of a probe p to such subspace. However, even this simple operation turns out to be too computationally expensive for real-time implementation on our cell phone with the desired resolution (640 × 480 pixels). We can reduce the computational complexity as follows. First, we assume that the diagonal color model (3) holds. Based on Fact 2, we note that the square of the distance $d(p,\mathcal{S})$ of the probe p to the subspace $\mathcal{S}$ is equal to the sum of the squared distances of p to $\mathcal{S}_1$, $\mathcal{S}_2$ and $\mathcal{S}_3$ respectively. Accordingly, the distance-based classifier declares a detection when:

$$\sum_{k=1}^{3} d(p_k,\mathcal{S}_k)^2 < \bar{\tau}^2 \tag{7}$$

where $\bar{\tau}$ is a suitable threshold, and p_k is the vector formed by the four values in the probe for the kth channel (for the sake of notational simplicity, we neglect to indicate the dependence on the illuminant i in the following). In addition, rather than setting a threshold on the sum of squared distances to the $\{\mathcal{S}_k\}$, we set a threshold $\tau = \bar{\tau}/3$ on the *maximum* of such distances. In other words, we declare a detection when:

$$\max_k d(p_k,\mathcal{S}_k) < \tau \tag{8}$$

The advantage of this detector is that it can be implemented as a cascade of three tests, each involving computation of $d(p,\mathcal{S}_k)$ and comparison with a constant.

If q_k is the unit vector originating the one-dimensional subspace $\mathcal{S}_k$, then

$$d(p_k,\mathcal{S}_k)^2 = p_k^T p_k - (p_k^T q_k)^2 \tag{9}$$

This operation requires nine multiplications and seven additions per pixel, which is still too demanding for our real-time implementation. To reduce the computational cost further, we introduce the following procedure. We begin by observing that, given any two surface types (s_1,s_2) in the color marker, the following inequalities hold:

$$d(p_k,\mathcal{S}_k) \geq d(p_{(s_1,s_2),k},\mathcal{S}_{(s_1,s_2),k}) \tag{10}$$

$$d(p_k,\mathcal{S}_k)^2 \leq \sum_{s_1=1}^{3}\sum_{s_2=s_1+1}^{4} d(p_{(s_1,s_2),k},\mathcal{S}_{(s_1,s_2),k})^2 \tag{11}$$

where $p_{(s_1,s_2),k}$ contains the kth color channel for surfaces s_1 and s_2, and $\mathcal{S}_{(s_1,s_2),k}$ is the (one-dimensional) projection of $\mathcal{S}_k$ onto the plane $P_{(s_1,s_2)}$ spanned by the vectors $p_{(s_1,s_2),k}$ for varying i. These inequalities show that a *necessary* condition for $d(p_k,\mathcal{S}_k) < \tau$ to hold is that, for all surface pairs (s_1, s_2),

$$d(p_{(s_1,s_2),k},\mathcal{S}_{(s_1,s_2),k}) < \tau \tag{12}$$

while a *sufficient* condition is that, for all surface pairs (s_1, s_2),

$$d(p_{(s_1,s_2),k},\mathcal{S}_{(s_1,s_2),k}) < \tau/\sqrt{6} \tag{13}$$

as there are six terms in the sum in (11). This suggests the use of a cascaded implementation with six *elemental* classifiers, each computing $d(p_{(s_1,s_2),k},\mathcal{S}_{(s_1,s_2),k})$ for a choice of (s_1,s_2). In practice, each elemental classifier examines whether $p_{(s_1,s_2),k}$ lies within a strip in the plane $P_{(s_1,s_2),k}$ (see Fig. 11), where the strip is parallel to $q_{(s_1,s_2),k}$, the projection of q_k onto $P_{(s_1,s_2),k}$. A fast cascaded implementation of each elemental classifiers can be derived by observing that (13) is satisfied when both of these inequalities are satisfied:

$$\begin{aligned} p_{(s_2),k} - a_{(s_1,s_2),k} p_{(s_1),k} &< \hat{\tau} \\ p_{(s_2),k} - a_{(s_1,s_2),k} p_{(s_1),k} &> -\hat{\tau} \end{aligned} \tag{14}$$

where $a_{(s_1,s_2),k} = q_{(s_2),k}/q_{(s_1),k}$ and $\hat{\tau} = (\tau/\sqrt{6})/\cos(\arctan a_{(s_1,s_2),k})$. Thus, each individual classifier in $P_{(s_1,s_2),k}$ requires one multiplication, one addition, and one or two comparisons. In fact, as we discuss below, we do not use a single threshold but two distinct thresholds, $\hat{\tau}_1$ and $\hat{\tau}_2$, that are learned from training data. The computation cost remains the same whether the same or different thresholds are used.

The vectors $\{q_k\}$ is computed via SVD from training data. As for the thresholds $\hat{\tau}_1$ and $\hat{\tau}_2$, we could use a simple criterion: choose the smallest values that ensure correct detection of all probes in the training data. In practice, this means expanding the 'strip' on either side of $q_{(s_1,s_2),k}$ until all training probes $p_{(s_1,s_2),k}$ are contained in the strip. This ensures that all training data is correctly classified [1, 5]. In our experiments, we noted that this choice is often too conservative, in which case we can multiply both thresholds by a constant *margin ratio (MR)* coefficient.

It is instructive to compare these elemental detectors with those originally proposed by Coughlan and Manduchi [5], which simply compared $p_{(s_2),k} - p_{(s_1),k}$ with a threshold τ. This is equivalent to declaring detection when a sub-probe lies in a 'detection sector', formed by all points $p_{(s_1,s_2),k}$ that are above (or below) a 45° line with intercept at $p_{(s_2),k} = \tau$. Clearly, this detector is less selective than the newly proposed one. In terms of computational cost, our new method requires one multiplication and up to one comparison more per pixel per elemental detector with respect to the original detector [5].

5.2 Computational cost

Because there are six permutations of the four surfaces in the marker taken two at a time, and three color channels,

the total number of elemental (cascaded) classifiers is eighteen. For a probe to be classified as a candidate marker, it must pass all 18 tests. Assuming statistical independence of the tests, it is well known that the overall probability of (correct) detection is

$$P_{\mathrm{D}} = \prod_{i=1}^{18} P_{\mathrm{D}_i} \tag{15}$$

where P_{D_i} is the probability of detection for the ith elemental classifier. Likewise, the overall probability of false alarm is equal to

$$P_{\mathrm{F}} = \prod_{i=1}^{18} P_{\mathrm{F}_i} \tag{16}$$

The expected number of operations (multiplications or additions) for a non-marker probe is

$$N_{\mathrm{ops}} = 1 + \sum_{i=2}^{18} \prod_{j=1}^{i-1} P_{\mathrm{F}_i} \tag{17}$$

As for the number of comparisons in the case of a non-marker probe, one can reason as follows. Suppose that color values of a non-marker probe are uniformly distributed in the space outside the 'detection strip' defined by (13), and that the classification strip is oriented approximately at 45°. The first test in (14) checks whether the probe is above the detection strip (with probability of 0.5 of finding it there). If this is not the case, the second test in (14) checks if it is below the strip. The average number of comparisons is thus $1.5 \times N_{\mathrm{ops}}$. This number should be taken as an upper bound: if the detection strip is at a different angle, the order of the tests in (14) can be chosen so that the first test detects a non-probe marker with probability larger than 0.5, thus reducing the average number of tests.

It is clear that the order of the elemental classifiers critically affects the average computational cost N_{ops} [2]. The maximum efficiency is obtained when the classifiers are ordered according to their false alarm rate P_{F_i}, with the first classifiers removing most of the false alarms.

5.3 Dealing with saturated point

When a color value is saturated, our linear model no longer applies. The effect of saturation to the color distribution can be seen clearly in the scatterplots of Fig. 11. In order to deal with saturated points, we considered three possible strategies. The first strategy is to identify those color values that are saturated, and change the classification rule for those points. This requires an additional number of comparisons for each pixel, thus increasing the computational cost. In our experiments, this resulted in an effective frame rate that was too low for our application. The second approach is to simply neglect the presence of saturation, and treat saturated pixels and unsaturated pixels alike. The third approach we considered is to remove saturated samples from the training data before computing the eigenvectors $\{q_k\}$. This helps ensuring that the slope of the stripe in the $P_{(s_1,s_2),k}$ plane is not biased by the saturated pixels. However, we consider all training samples when computing the thresholds $\hat{\tau}_1$ and $\hat{\tau}_2$. This is necessary as we require that all training samples are correctly detected. The resulting classifier is then applied on all new samples, regardless of whether they are saturated or not.

5.4 The advantage of gamma correction

The detector design introduced in Sect. 5.1 is based on the distance between the probe vector p and the subspace $\mathcal{S}$ in R^{12} where probe vectors are assumed to live. This is a simple and powerful approach, with one major pitfall. Since the subspace $\mathcal{S}$ contains the origin, any very dark probe (with color values close to zero) will be classified as a marker. Indeed, this was the single major cause of false positives in our experiments.

A simple fix could be to isolate very dark probes (via suitable thresholding) to avoid that they be mistakenly classified as markers. Note in passing that our marker contains surfaces with high albedo (except for the black one) so it is unlikely that all surfaces would produce very low color values. Unfortunately, our experiments have shown that choosing a correct threshold is very difficult, leading to the risk of missing a marker due to poor exposure.

We have found another (somewhat unexpected) solution by considering the gamma-corrected data produced by the camera, rather than the linearized data. Indeed, as we elaborate below, the linear color rendering model applies with only a small modification to the gamma-corrected data, and this modification is key to an improved algorithm with much reduced false alarms.

According to the diagonal model, the values $p_{(s_1),k}$ and $p_{(s_2),k}$ of a probe are linearly related (4):

$$p_{(s_2),k} = a_{(s1,s2),k} p_{(s_1),k} \tag{18}$$

The gamma-corrected versions of $p_{(s_1),k}$ and $p_{(s_2),k}$ (i.e., the values produced by the camera) are, according to (5):

$$\bar{p}_{(s_1),k} = b_k + p^{\gamma_k}_{(s_1),k}, \quad \bar{p}_{(s_2),k} = b_k + p^{\gamma_k}_{(s_2),k} \tag{19}$$

Combining (18) and (19) one obtains:

$$\bar{p}_{(s_2),k} = b_k(1 - a^{\gamma}_{(s1,s2),k}) + a^{\gamma}_{(s1,s2),k} \bar{p}_{(s_1),k} \tag{20}$$

Hence, the gamma-corrected values satisfy a linear equation with non-null intercept (except for the case $a_{(s1,s2),k} = 1$, in

which the intercept is null). An example with $a_{(s1,s2),k} = 4$ using the values for γ_1 and b_1 found in Sect. 4.2, is shown in Fig. 6. This suggests a simple modification to our algorithm, which allows it to work with gamma-corrected data. We first note that, since the (non-linearized) probes $\bar{p}_k$ span a line that does not necessarily intersect the origin, our previous approach to characterize this line by the dominant eigenvector of the probe data would fail. We can correct for this by simply removing the mean of the probes $\{\bar{p}_k\}$ before eigenvector analysis. Because the mean is supposed to lie on the line spanned by the $\bar{p}_k$, the dominant eigenvector q_k of the zero-mean data now reliably characterizes the line. Projection of q_k onto planes $P_{(s_1,s_2)}$ gives the slope $\bar{a}_{(s1,s2),k}$ of the line where points $\bar{p}_{(s_1,s_2),k}$ are supposed to live. Then, similarly to the previous case, a strip is expanded on either side of this line until all training points are contained in the strip. Note that this classification region is structurally identical to the case considered in Sect. 5.1, except that now it need not contain the origin. It is exactly this characteristic that allows this approach to substantially reduce the rate of false positives, as shown by the experimental results described in the next section.

5.5 Encoding information via color permutation

It is possible to encode a few bits of information within the marker by simply permuting the position of the color patches (with the permutation index representing the marker ID). Since there are $4! = 24$ permutations of the four colors, the information content is $\log_2 24 = 4.6$ bits. Note that this form of information embedding comes at no "area cost"—the overall marker area remains the same. Of course, more information could be embedded by adding other patterns (e.g. 2-D bar codes) near the color markers. In this case, the marker would be used simply as a distinctive fiducial, allowing for quick and reliable identification of the pattern location.

A disadvantage of this simple approach is that it increases the computational cost of marker detection. Although in the single-ID case (in which color patches have a fixed position) detection is obtained via a cascade of tests, each involving two patches (s_1,s_2) and one color channel (k), now the first step involves testing all possible permutation of surfaces taken two at a time for the same color channel k (in total, 12 tests). Each patch pair that passes the first test "fixes" the position of two patches in the permutation. Computing the expected number of operations for non-marker probes is difficult. Empirically, we have observed an increase in the number of operations by a factor of 16 when considering all possible color permutations (see Table 1). We should emphasize that this increased computational cost is incurred only when the marker's ID is unknown. If one is searching for a specific ID (i.e. if the color permutation is known), then the computational cost is the same as described in Sect. 5.2.

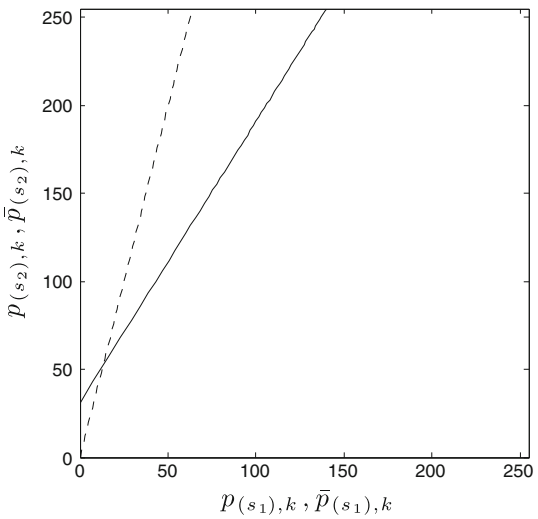


Fig. 6 The *dashed line* shows the relationship $p_{(s_2),k} = a_{(s1,s2),k} p_{(s_1),k}$ with $a_{(s1,s2),k} = 4$. The *solid line* shows the relationship between the gamma-corrected (non-linearized) values $\bar{p}_{(s_1),k}$ and $\bar{p}_{(s_2),k}$

Table 1 Comparative results in terms of detection rate and processing time for the tests considered in Sect. 7

	Color marker	ARToolKit marker
Detection rate		
Various placements	**18/18**	12/18
Illuminant 1		
Various placements	**17/18**	14/18
Illuminant 2		
Motion blur	**24/24**	20/24
Bright light		
Motion blur	**47/58**	0/58
Dim light		
Partially occluded	**6/7**	1/7
Processing time (ms)		
Individual ID	**2.6**	3.2
No visible markers		
Individual ID	3.9	**3.7**
Visible markers		
Multiple IDs	58.9	**3.2**
No visible markers		
Multiple IDs	58.9	**3.7**
Visible markers		

6 Experiments

6.1 Data sets

We collected 150 images, taken with the Nokia N95 8GB phone, of the marker under different conditions of illumination (both indoors and outdoors), from different viewing angles, and from different distances. Each image was hand-labeled. More precisely, a 15×15 square was drawn on each color sector, and a list was created with 25 pixels picked from each square. Then, probes were built by scanning the lists for the four squares in parallel, taking the color values for each point in each list. Thus, our marker training set contains $150 \times 25 = 3,750$ probes. Note that we do not low-pass filter the training data (nor the test data). Although low-pass filtering would help removing noise, its computational cost would reduce the effective frame rate. In addition, the blur generated by a low-pass filter could potentially corrupt the color values within the marker when the marker image is small (because taken from a distance).

We also took five different images (indoors and outdoors) of various 'background', that were used to estimate the false alarm rates. (Note that the classifier is designed only based on the marker images.) These background images, shown in Fig. 7, were scanned with a probe with width of 12 pixels. In total, we collected 163,020 samples of background data.

Figure 8 show the sorted false positive rates P_{F} for the different elemental detectors, trained on all marker data and tested on the background images. These detectors were designed on the gamma-corrected data $\bar{p}_{(s_1,s_2},k)$ after mean removal (Sect. 5.4) and without removing saturated pixels for the computation of the $\{q_k\}$. The margin rate *MR* was set to 1.

Figures 11 and 12 show the scatterplots of the original gamma-corrected $(\bar{p}_{(s_1,s_2),k})$ and linearized $(p_{(s_1,s_2),k})$ probes for different choices of the surfaces s_1 and s_2 and for different color channels k. The plots are ordered according to the false positive rates P_{F} of the elemental filters in Fig. 8. In each figure, we show the 'classification strip', that is, the region in the $P_{(s_1,s_2),k}$ plane where a probe is classified as a marker by an elemental classifier. The solid lines identify the classification strip for the case in which all samples are used for training, while the dashed lines represent the case in which saturated points are removed before computing the eigenvectors $\{q_k\}$ (see Sect. 5.3). Note that in several cases, the classification strips for the original gamma-corrected probes $\bar{p}_{(s_1,s_2),k}$ does not contain the origin.

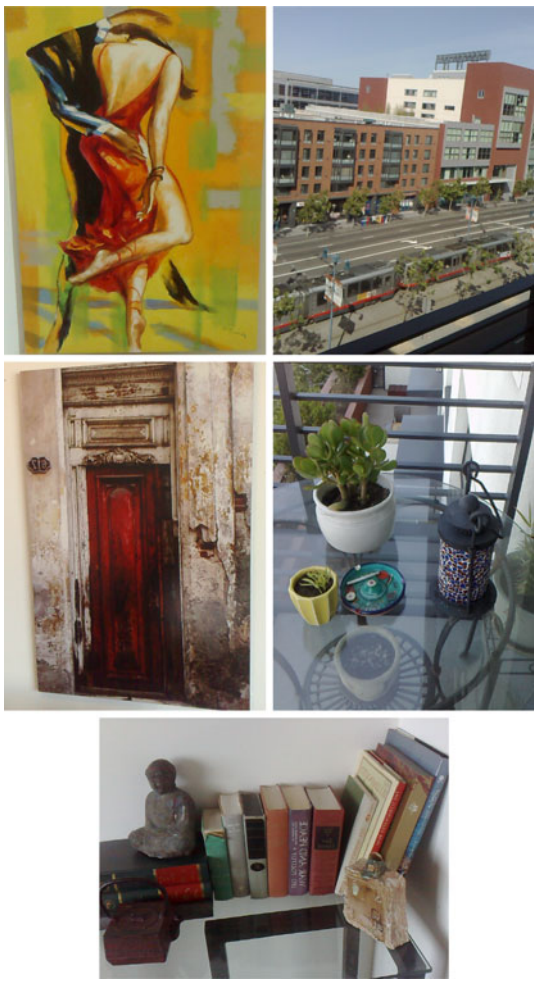


Fig. 7 The background image data set

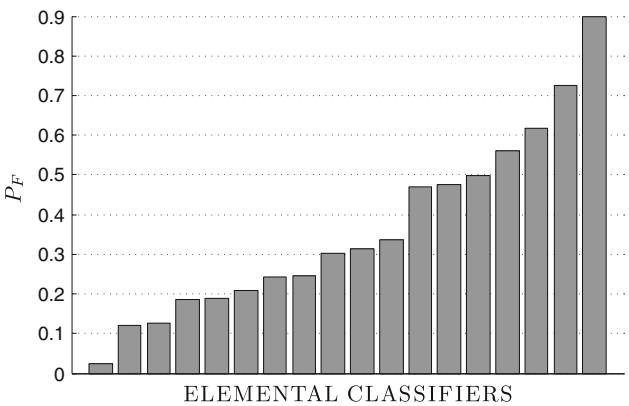


Fig. 8 The sorted false positive rate P_{F} for the different elemental detectors operating on the gamma-corrected training data. The detectors were designed after mean removal (Sect. 5.4) and without removing saturated pixels for the computation of the $\{q_k\}$. The margin rate *MR* was set to 1

6.2 Performance evaluation

To test our system, we performed a number of cross-validation experiments. At each experiment n, half of the marker images were chosen at random. The detector was trained on such images, and then tested on the data from the remaining marker images to establish the detection rate $P_D(n)$, as well on the 'background' data to compute the false alarm rate $P_F(n)$. The results of 30 such experiments were then averaged together. Note that even though the classifier is guaranteed to produce $P_D(n) = 1$ for the data it was trained on, it is still prone to false negatives for the remaining data.

Figure 9, top, illustrates the results in the form of 'pseudo-ROC' curves. Each pseudo-ROC curve is obtained by plotting P_F against P_D for varying 'cascade length', where the cascade length is the number of elemental detectors in the cascade. More precisely, we ordered the 18 elemental detectors (designed with margin rate *MR* set to 1) according to increasing value of their false alarm rate.

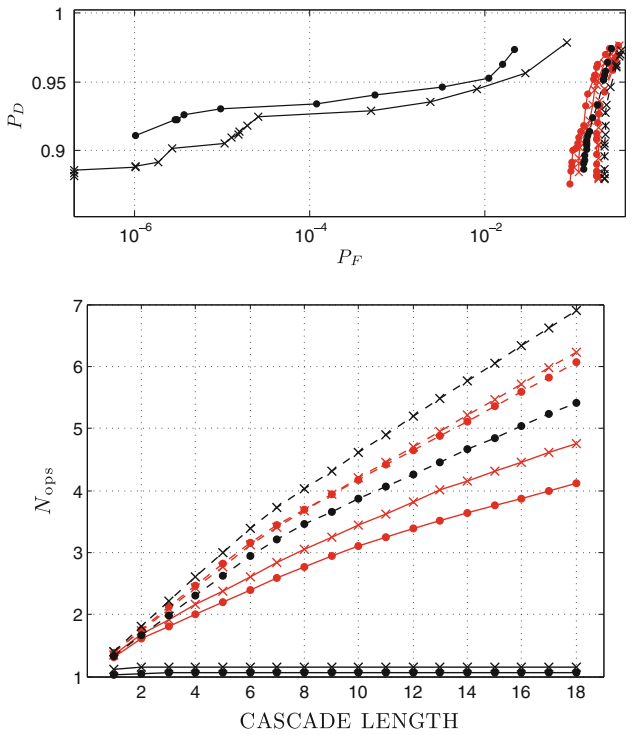


Fig. 9 *Top*. Pseudo-ROC curves for different design parameters. *Bottom*. Expected number N_{ops} of multiplication for input pixel when a marker is not visible as a function of the cascade length. The values for N_{ops} were computed using the actual values of P_D and P_F for different cascade lengths rather than their approximation (15–17). Each *marker* in the *curve* represents a different cascade length. *Solid line* original gamma-corrected data. *Dashed line* linearized data. *Red line* data mean not removed before computing the eigenvectors $\{q_k\}$. *Black line* data mean removed. *Circles* saturated points not removed before computing the eigenvectors $\{q_k\}$. *Crosses* saturated points removed. The margin rate *MR* was set to 1 for these experiments

Then, we removed detectors from the tail of the cascade to create cascaded detectors with different length. It should be clear from (15), (16) and (17) that reducing the number of elemental detectors increases P_D as well as P_F while reducing the computational cost N_{ops} (see also Fig. 9, bottom). These curves can be useful to choose the correct cascade length if the application at hand sets specific requirements in terms of P_D, P_F or N_{ops}.

The different pseudo-ROC curves in Fig. 9 correspond to different choices of design parameters:

- Whether training (and testing) was performed on the original, gamma-corrected data or on the linearized data (Sect. 5.4);
- Whether saturated data was removed before computing the slope of the strips in the elemental detectors or all trained data was used (see Sect. 5.3);
- Whether or not the eigenvectors $\{q_k\}$ originating $\mathcal{S}_k$ were computed by first removing the mean of the training probes (see Sect. 5.4).

From Fig. 9, it results clear that using the original gamma-corrected data and removing the mean of the training probes before computing the eigenvectors $\{q_k\}$ produces the best results. Other choices of parameters have large false alarm rates. A single elemental detector with the lowest false positive rate (the first one shown in Fig. 11, comparing data from the probe corresponding to the orange and the black surface in the red channel) has $P_D = 0.97$ and $P_F = 0.02$. Increasing the cascade length reduces both P_D and P_F, as expected.

Even with the best choice of parameters, it is seen that these results are not satisfactory. For example, a false positive rate P_F of 10^{-5} is achieved only at the cost of a relatively low detection rate ($P_D = 0.93$). As noted earlier, it is important to keep the false alarm rate low even if subsequent post-processing may remove remaining false alarms. For example, if $P_F = 10^{-5}$, a VGA-sized image without a marker will generate about three false alarms on an average, which then need to be processed further. The detection performance can be improved by increasing the margin ratio *MR*. Figure 10, top, shows the ROC curve for a detector operating on the original gamma-corrected data, with data mean removed, and saturated points not removed before computing the eigenvectors $\{q_k\}$ (corresponding to the best-performing pseudo-ROC curve in Fig. 9). This curve is obtained by increasing the margin ratio from 1 to 1.5. Note that now the detection rate P_D increased to 0.98 for $P_F = 10^{-5}$ with $MR = 1.2$. The corresponding computational cost per pixel is of $N_{ops} = 1.1$ multiplications and additions, and $1.5 \times N_{ops} = 1.65$ comparisons.

We implemented the cascaded detection algorithm on the Nokia N95 8 GB, programmed in C under the Symbian

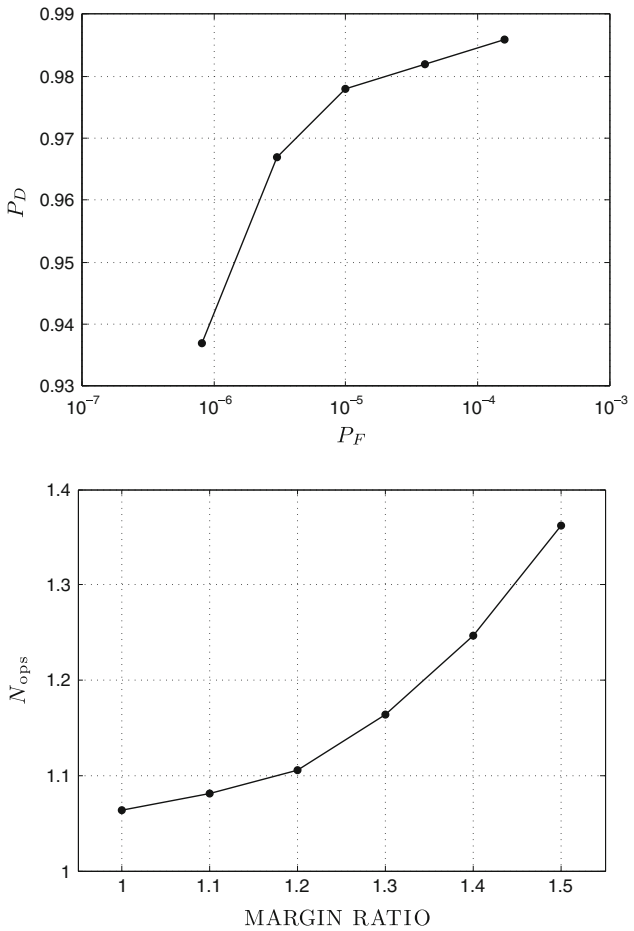


Fig. 10 *Top*. ROC curve for a detector operating on the original gamma-corrected data (data mean removed and saturated points not removed before computing the eigenvectors $\{q_k\}$). *Bottom*. Expected number N_{ops} of multiplication for input pixel when a marker is not visible. The different points in the curves correspond to different margin ratios *MR*

OS 9.6 S60. This cell phone is equipped with an ARM 11 332 MHz processor with 128 MB of RAM and 8 GB of Flash memory. The images are processed at full VGA resolution. The effective frame rate is about 8 frames per second (fps) when no marker is present. Owing to post-processing (including segmentation), the frame rate reduces to 5 fps when a marker is present. We tested the system extensively as a wayfinding tool for persons who are blind ([17]—see Fig. 13).

7 Comparison with ARToolKit

We benchmarked the performance of our color marker system against a popular marker, the ARToolKit [12, 13, 30] that does not use color information. The ARToolKit marker has been used extensively for augmented reality (AR) applications. It consists of a square black border encircling a grayscale pattern that contains the ID of the marker. An improved version, the ARTag Fiala [8], still uses the square black border but replaces its interior with a digital pattern of 36 bits.

For our comparative study, we used the open source implementation of the ARToolkit Library for Windows maintained by the University of Washington[1]. The detection algorithm works by first binarizing a greyscale image using a fixed threshold. The connected components of the binarized image are then computed, and their edges and corners are extracted. The vertices of the detected outer black square are used to compute the homography mapping the square onto its image in the camera. (Note that, as shown in Fig. 2, a similar operation can be performed with our color marker, by relying on the segmentation described in Sect. 3.) The interior of the black square border is then analyzed to extract the marker's ID (Figs. 11, 12).

In our experiments, we considered various realistic situations including different viewing distances and angles, illumination conditions, motion blur, and occlusions. Note that we are interested in the *detection* performance, and not in the ID computation. Hence, our results are only in terms of *detection rate*: we never checked whether the ID computed by the ARToolKit algorithm was correct or not. In each test, a color marker and an ARToolKit marker were placed side by side on a vertical surface, and images were taken by a Nokia N95 cell phone (at 640 × 480 resolution) for further processing on a laptop computer. This solution allowed us to compare the two algorithms on the same computing platform. The diameter of the color marker (15 cm) was set to be equal to the side length of the ARToolKit marker. There was never more than one color marker and one ARToolKit marker visible in the same image.

The detection rates for the various experiments described in the following are shown in Table 1, along with the average computational time (on the laptop) per frame for both types of markers.

7.1 Experiments

7.1.1 Multiple camera placements

In this experiment, images of the marker pair were taken from angles of 0°, 30° and 60° (with respect to the normal to the markers' surface), at 6 equispaced distances from 0.6 to 3.6 m. Two different illumination conditions were considered. The camera locations and sample images are shown in Fig. 14. Under the first illuminant, the color marker was detected from any location, while the ARToolKit was not detected at distances beyond 2.4 m.

[1] http://www.hitl.washington.edu/artoolkit.

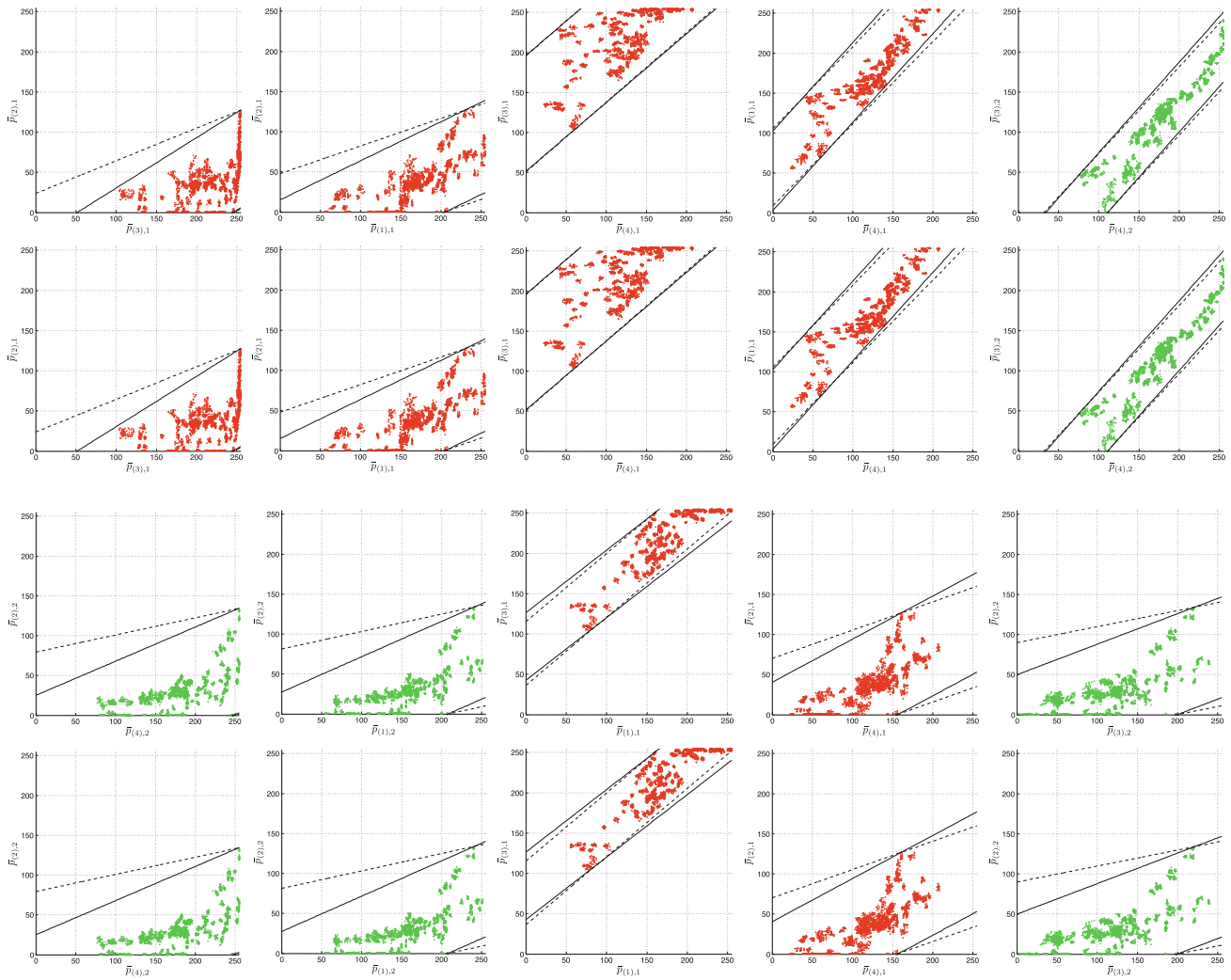


Fig. 11 The first ten scatterplots of color probe points from our training data, restricted to the planes $P_{(s_1,s_2),k}$. The gamma-corrected data $(\bar{p}_{(s_1),k}, \bar{p}_{(s_2),k})$ are shown on top of the linearized data $(p_{(s_1),k}, p_{(s_2),k})$. The scatterplots are ordered according to the increasing false positive rate P_{F} of the elemental detectors as shown in Fig. 8. The 'classification strips' are shown with different line types depending on whether saturated points were removed before computing the eigenvectors $\{q_k\}$ (*dashed line*) or not (*solid line*). $s = 1$: *white surface*. $s = 2$: *black surface*. $s = 3$: *orange surface*. $s = 4$: *green surface*. $k = 1$: *red channel*. $k = 2$: *green channel*. $k = 3$: *blue channel*

Under the second illuminant, the color marker was detected in all but one location. The ARToolKit marker was not detected in four locations.

7.1.2 Motion blur: bright light

24 images were taken of the marker pairs under a bright illuminant from various distances, while the camera was moving. Camera motion should always be expected with mobile vision applications; these experiments are meant to study the robustness of the detection algorithms under the ensuing motion blur. The color marker was detected all 24 times, while the ARToolKit marker was detected 20 times. Examples are shown in Fig. 15.

7.1.3 Motion blur: dim light

In this case, 58 images of the markers under dim light were taken while the camera was moving. In low light conditions, the camera is forced to increase exposure time and sensor gain. This gives rise to motion blur and noise, both clearly noticeable in the examples of Fig. 16. Under this challenging condition, the color marker was detected 47 times, while the ARToolKit marker was never detected,

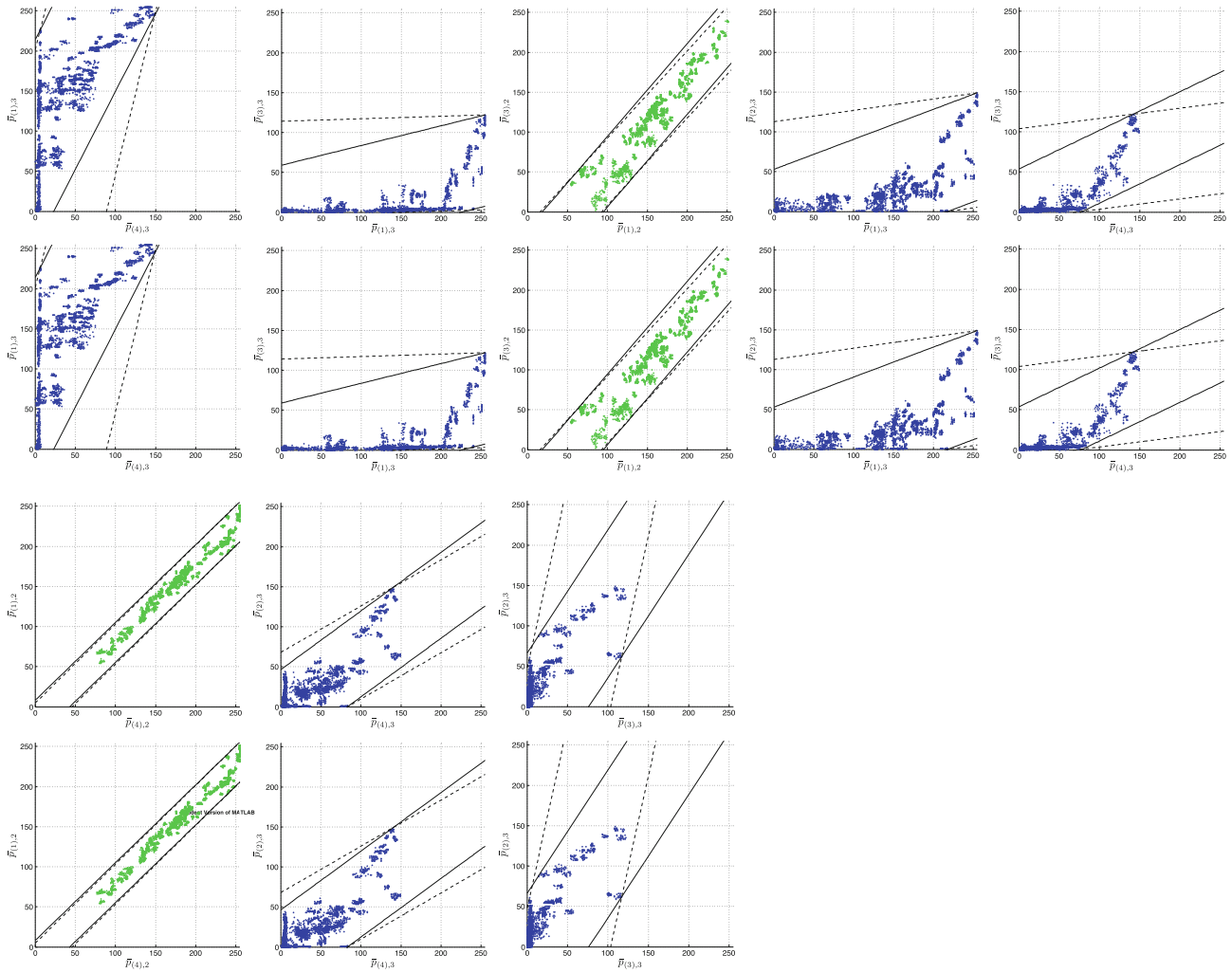


Fig. 12 The last eight scatterplots of color probe points from our training data, restricted to the planes $P_{(s_1,s_2),k}$ (see caption of Fig. 11)

Fig. 13 Example of use of our color marker as a wayfinding system for blind persons [17]

due to the fact that the fixed threshold was too high for correct binarization.

7.1.4 Occlusions

Seven images were taken with both markers being partly occluded by a surface. This situation may occur, for example, when one is searching for a marker in a crowded scene, with other persons impeding view of part of the marker. The color marker was detected in all, but one case (in which it was actually detected, but segmentation was not successful—see Fig. 17). The ARToolKit marker was detected only once in these experiments.

7.1.5 False positives

No instances of false positives were recorded using the color marker, even when the background contained a variety of colors. Sporadic episodes of false positives occurred with the ARToolKit marker (see e.g. Fig. 18).

7.1.6 Processing time

The average processing times for the two algorithms, computed on the laptop computer used for the experiments (Intel Pentium Dual CPU T3200 at 2 GHz with 3 GB

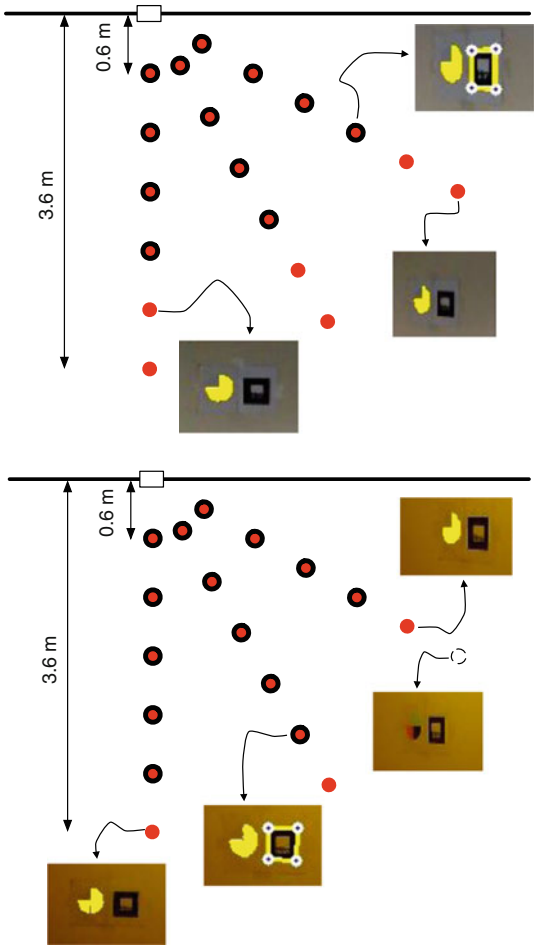


Fig. 14 Comparative detection experiments using the color marker and the ARToolKit marker for two different illuminants (Sec. 7.1.1). The diameter of the color marker (set to 15 cm) was equal to the side length of the ARToolKit marker. The markers were placed side by side on a wall, at a location shown by a *small rectangle* in the bird-eye view. Images were taken with a cell phone camera placed in the locations shown by the *circles*. Locations in which the color marker was successfully detected are marked in *red*. A *thick black border* indicates successful detection of the ARToolKit marker. The *image crop-outs* show samples of correct and missed detection. Successful detection is indicated by the *yellow area* on a color marker (which shows the result of segmentation, as described in Sect. 3) or by the *yellow edges* on a ARToolKit marker

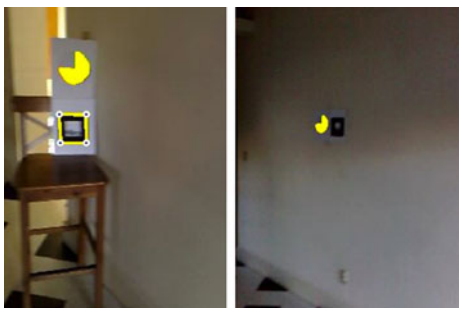


Fig. 15 Correct and missed detection examples for the motion blur: bright light experiments (Sect. 7.1.2)

Fig. 16 Correct and missed detection examples for the motion blur: dim light experiments (Sect. 7.1.3)

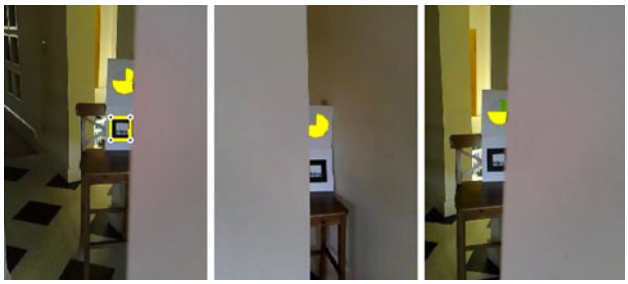
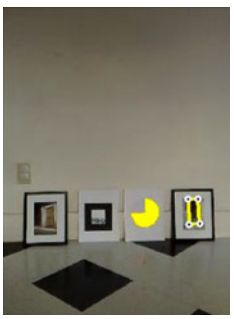


Fig. 17 Correct and missed detection examples for the *Occlusions* experiments (Sect. 7.1.4)

Fig. 18 An example of false positive and missed detection for the ARToolKit marker



RAM) for images with size of 640 × 480 pixels, are shown in Table 1. Two different versions of the color marker detection algorithm were implemented: one in which a specific ID marker was searched for, and one that considered all possible 24 color permutations as described in Sect. 5.5.

For the single-ID color marker, the processing time is lower than for the ARToolKit detection when there are no markers visible in the scene. When a marker is visible and detected, the color marker takes a slightly larger computational time. Note that these computational times include segmentation (described in Sect. 3). When a marker is detected, segmentation accounts for about 16% of the computational cost.

The situation is very different when the marker ID is not known in advance. In this case, as explained in Sect. 5.5, a much larger number of tests is required for each pixel, leading to an increase in processing time by a factor of 16.

7.2 Discussion

The basic conclusion from these experiments is that, for the same marker surface area, color markers can be detected far more robustly and at a wider range of distances than ARToolKit markers. The detection speed is comparable in the two cases when a specific ID marker is searched for. However, if all 24 marker IDs are considered during search using the color permutation technique of Sect. 5.5, color markers require a much larger (16 times) computational time than ARToolKit markers. A careful comparative analysis of the two marker types and detection algorithm can shed light on these performance differences.

Detection of an ARToolKit marker hinges on successful binarization of the marker's outer edge. Because the marker's outer edge is black on a white background, binarization is in most cases attainable using a fixed threshold. The ability to use a fixed threshold is vital for computational efficiency, a factor that is especially important with power-constrained mobile platforms and when high image resolution is used. Unfortunately, as seen in the experiments with dim ambient light (Sect. 7.1.3), a fixed threshold may lead to gross errors in some situations. Adaptive binarization [27] would most likely improve results, but at a heavier computational cost. This is, in fact, one of the principal advantage of using color markers: robust detection of carefully chosen color patterns is achievable with very few operations per pixel under any illuminant.

The outer black border of the ARToolKit marker needs to be resolved at a high enough resolution to enable geometric analysis. This places a heavy constraint on the maximum distance at which the marker can be detected. An advantage of the color marker is that it does not require geometric processing for detection: as long as the probe is contained in the marker's image (see Fig. 1), detection can be achieved.

The reason for the dramatic increase of computational time when the color marker's ID is not known in advance is that ID identification for permuted-color markers is embedded in the search process at the pixel level. In the case of ARToolKit markers, the ID information is inside the marker, while detection only uses the outside border. A similar solution would be impossible with our color marker: the whole surface of the color marker must be used for the color patches, since the probe's vertices may fall on different points of the marker's image depending on the viewing distance. It should be noted, however, that the maximum distance at which the pattern inside an ARToolKit marker can be decoded is likely to be smaller than the maximum distance at which the outer border of the marker can be detected, and that the decoding process is likely to be affected by motion blur (see e.g. Fig. 15).

Different solutions for embedding ID information in a color marker could be considered, such as using a color or grayscale pattern in an outer ring around the marker. In particular, it was shown recently that up to 7 bits of information can be embedded reliably within a single color patch [31]. Thus, a selected set of just a few color patches around the marker could convey enough ID information for most practical purposes.

8 Conclusions

We introduced a new detection algorithm that is suitable for multi-color, pie-shaped markers. This is a crucial component in a cell phone-based system that uses environmental labeling for blind wayfinding. The proposed algorithm is very light and has excellent performance in terms of detection rate and false alarm rate. The algorithm is implemented as a cascade of elemental detectors, each one of which operates on only two color values from a probe in the same color channel. The elemental detectors are derived based on a diagonal rendering model. One interesting result of our study (for which we provide formal justification) is that using the original, gamma-corrected data gives better results than using linearized data. In addition, we have introduced a very simple method for selecting surfaces for our color markers that have good Lambertian characteristics, and thus minimize the risk of mis-detection due to specular reflection.

When compared with a popular grayscale marker (ARToolKit), our color markers enable more robust detection in various realistic conditions for a similar processing time. However, the modality used by the ARToolKit (and other similar marker such as the ARTag) for embedding ID information allows for faster decoding than the simple approach of color permutation proposed for the color markers. We are currently exploring new strategies for encoding ID information using a grayscale or color pattern at the outer edge of the color marker.

Acknowledgments This material is based upon work supported by the National Science Foundation under Grant No. IIS - 0835645. The authors would like to thank the anonymous reviewers for their insightful comments, and in particular for suggesting comparison of our color marker against the ARToolKit. Dr. James Coughlan of SKERI provided useful comments and feedback during the development of this work.

References

1. Chen, X., Yuille, A.: Detecting and reading text in natural scenes. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, CVPR '04 (2004)
2. Chen, X., Yuille, A.L.: A time-efficient cascade for real-time object detection: With applications for the visually impaired.

In: Proceedings of the IEEE Workshop on Computer Vision for the Visually Impaired, p. 28. IEEE Computer Society, Washington, DC, USA (2005). doi:10.1109/CVPR.2005.399
3. Cho, Y., Neumann, U.: Multi-ring color fiducial systems for scalable fiducial tracking augmented reality. In: VRAIS '98: Proceedings of the Virtual Reality Annual International Symposium, p. 212. IEEE Computer Society, Washington, DC, USA (1998)
4. Claus, D., Fitzgibbon, A.W.: Reliable fiducial detection in natural scenes. In: Proceedings of the 8th European Conference on Computer Vision, pp. 469–480 (2004)
5. Coughlan, J., Manduchi, R.: Color targets: fiducials to help visually impaired people find their way by camera phone. EURASIP J. Image Video Process. **2** (2007)
6. Coughlan, J., Manduchi, R.: Functional assessment of a camera phone-based wayfinding system operated by blind and visually impaired users. Int. J. Artif. Intell. Tool **18**(3), 379–397 (2009)
7. Farid, H.: Blind inverse gamma correction. IEEE Trans. Image Process. **10**(10), 1428–1433 (2001)
8. Fiala, M.: ARTag, an improved marker system based on ARToolkit. Technical report, NRC: 47419, National Research Council Canada (2004)
9. Finlayson, G.D., Drew, M.S., Funt, B.V.: Color constancy: generalized diagonal transforms suffice. J. Opt. Soc. Am. A **11**, 3011–3020 (1994)
10. Gallo, O., Manduchi, R.: Reading 1-D barcodes with mobile phones using deformable templates barcodes with mobile phones using deformable templates. IEEE Trans. Pattern Anal. Mach. Intell. (2010)
11. Healey, G., Slater, D.: Global color constancy: recognition of objects by use of illumination-invariant properties of color distributions. J. Opt. Soc. Am. A **11**(11), 3003–3010 (1994). http://josaa.osa.org/abstract.cfm?URI=josaa-11-11-3003
12. Kato, H., Billinghurst, M.: Marker tracking and HMD calibration for a video-based augmented reality conferencing system. In: Proceedings of the 2nd IEEE and ACM International Workshop on Augmented Reality (IWAR '99), p. 85. IEEE Computer Society, Washington, DC, USA (1999)
13. Kato, H., Billinghurst, M., Blanding, B., May, R.: Ar-toolkit. Technical report, Hiroshima City University (1999)
14. Lee, W., Woo, W.: Real-time color correction for marker-based augmented reality applications. In: Proceedings of International Workshop on Ubiquitous Virtual Reality (2009)
15. Maloney, L., Wandell, B.: A computational model of color constancy. J. Opt. Soc. Am. **1**(1), 29–33 (1986)
16. Manders, C., Mann, S.: True images: a calibration technique to reproduce images as recorded. International Symposium on Multimedia (2006). doi:10.1109/ISM.2006.155
17. Manduchi, R., Kurniawan, S., Bagherinia, H.: Blind guidance using mobile computer vision: a usability study. In: ACM SIGACCESS Conference on Computers and Accessibility (ASSETS) (2010)
18. Marimont, D.H., Wandell, B.A.: Linear models of surface and illuminant spectra. J. Opt. Soc. Am. A **9**(11), 1905–1913 (1992)
19. Naimark, L., Foxlin, E.: Circular data matrix fiducial system and robust image processing for a wearable vision-inertial self-tracker. In: Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR'02), p. 27. IEEE Computer Society, Washington, DC, USA (2002)
20. Parikh, D., Jancke, G.: Localization and segmentation of a 2d high capacity color barcode. IEEE Workshop Appl. Comput. Vis. **0**, 1–6 (2008). doi:10.1109/WACV.2008.4544033
21. Rohs, M., Gfeller, B.: Using camera-equipped mobile phones for interacting with real-world objects. In: Advances in Pervasive Computing, pp. 265–271 (2004)
22. Scharstein, D., Briggs, A.J.: Real-time recognition of self-similar landmarks. Image Vis. Comput. **19**, 763–772 (1999)
23. Shafer, S.A.: Using color to separate reflection components. Color Res. Appl. **10**, 210–218 (1985)
24. Shafique, K., Shah, M.: Estimation of the radiometric response functions of a color camera from differently illuminated images. In: International Conference on Image Processing (ICIP) (2004). doi:10.1109/ICIP.2004.1421569
25. State, A., Hirota, G., Chen, D.T., Garrett, W.F., Livingston, M.A.: Superior augmented reality registration by integrating landmark tracking and magnetic tracking. In: Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '96), pp. 429–438. ACM, New York, NY, USA (1996). doi:10.1145/237170.237282
26. Takacs, G., Chandrasekhar, V., Gelfand, N., Xiong, Y., Chen, W.C., Bismpigiannis, T., Grzeszczuk, R., Pulli, K., Girod, B.: Outdoors augmented reality on mobile phone using loxel-based visual feature organization. In: Proceedings of MIR '08, pp. 427–434. ACM, New York, NY, USA (2008)
27. Trier, O., Taxt, T.: Evaluation of binarization methods for document images. IEEE Trans. Pattern Anal. Mach. Intell. **17**, 312–315 (1995). doi:10.1109/34.368197
28. Tsin, Y., Collins, R.T., Ramesh, V., Kanade, T.: Bayesian color constancy for outdoor object recognition. Computer Vision and Pattern Recognition **1**, 1132 (2001) doi:10.1109/CVPR.2001.990658
29. Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features. In: Proceedings of IEEE CVPR (2001)
30. Wagner, D., Schmalstieg, D.: ARToolKitPlus for pose tracking on mobile devices. In: Proceedings of Computer Vision Winter Workshop (2007)
31. Wang, F., Manduchi, R.: Color-constant information embedding. In: Color and Reflectance in Imaging and Computer Vision Workshop 2010 (2010)
32. Wang, J., Zhai, S., Canny, J.: Camera phone based motion sensing: interaction techniques, applications and performance study. In: Proceedings of the 19th Annual ACM Symposium on User interface Software and Technology, pp. 101–110. ACM, New York, NY, USA, UIST '06 (2006). doi:10.1145/1166253.1166270

Author Biographies

Homayoun Bagherinia is a PhD student of Electrical Engineering at the University of California, Santa Cruz. He also works at Carl Zeiss Meditec Inc. as an Algorithm engineer. His main research interests are in medical imaging, computer vision, and image processing.

Roberto Manduchi is an Associate Professor of Computer Engineering at the University of California, Santa Cruz. Prior to joining UCSC in 2001, he worked at Apple Computer, Inc. and at the NASA Jet Propulsion Laboratory. His main research interests are in assistive technology, computer vision, and sensor processing.