Removing Object Reflections in Videos by Global Optimization

Donatello Conte, Pasquale Foggia, Gennaro Percannella, Mario Vento

Removing Object Reflections in Videos by Global Optimization

Donatello Conte, Pasquale Foggia, Gennaro Percannella, and Mario Vento

***Abstract*—This paper presents a novel algorithm for the removal of reflections generated by objects on reflecting floors. The algorithm uses both chromatic properties of the reflections and geometrical constraints on their positions; however, it does not make use of a model of the reflected objects, and so it can be applied to scenes containing several kinds of objects (e.g., people, baggage, animals, vehicles, etc.). The proposed method has been validated by an extensive set of experiments on a large video database. In these experiments, the method has been compared to two other recent reflection removal algorithms. The experimental results show that the proposed method is fast and effective, both in absolute terms and in comparison with the other algorithms.**

***Index Terms*—Scene analysis, shading, tracking.**

I. Introduction

IN THE CONTEXT of an object detection system the removal of shadows and reflections is an important task. In fact, if a shadow or a reflection is mistakenly included as part of a detected foreground object, several problems may considerably impact the accuracy of the subsequent phases of the application; among them, errors in the estimation of the actual size or shape of the objects, and of their position, and the unwanted merging of distinct objects into a single entity, even in cases when the objects are not touching or occluding each other.

While many papers have been devoted to shadow removal [1]–[8], the problem of reflections has received comparatively much less attention; however, in some environments, reflections can be more likely than shadows, and usually they are harder to deal with. Examples are indoor scenes when the floor is smooth and shiny, or outdoor scenes in rainy weather conditions. Shadows and reflections differ under several respects; the most important differences are in position and color. The position of a shadow depends on the light sources, while reflections (assuming that the reflecting surface is a horizontal floor) are always located below the corresponding object. As regards the color, a shadow depends only on the color of the background and on the light sources (it has a darker shade of the same color of the background); on the other hand, the color of a reflection also depends on the color of the object. As a consequence of these differences, methods for shadow removal cannot be effectively applied for removing reflections: in some cases they may not be able to detect some pixels as a reflection (e.g., when the reflected object has a brighter color than the floor), while in other cases they may consider as part of a reflection some pixels that could be easily excluded because of their position (e.g., pixels not located below the corresponding reflected object). So, in recent years, a small but growing number of authors started proposing removal algorithms specifically devised for reflections.

A first group of papers (e.g., Teschioni and Regazzoni [9] and Carmona *et al.* [10]) follows an approach similar to the techniques commonly used for shadow removal. In particular, in [9] a model of the color properties of a reflection is assumed; the pixels consistent with this model are grouped using a region growing technique, and then discarded from the foreground. The method makes the assumption that the pixels of the foreground objects are significantly different (in the RGB space) from both the ones in the background and the ones in the reflections; when this assumption is not satisfied, it is likely that parts of the objects will be mistaken as reflections, even if their position would make this unplausible. In [10] a different representation model is used, the so-called Angle-Module space: the color of a foreground pixel and of the corresponding one in the background are compared by considering separately the direction of the RGB triples (interpreted as 3-D vectors) and their magnitude. Then some rules, based on this Angle-Module representation, are proposed to decide whether a pixel belongs to a reflection or not. Even though the change of representation makes possible the formulation of more effective rules, also this method is limited by use of chromatic properties as the only source of information, leading to a significant risk of misinterpretation.

A completely different approach is proposed by Zhao and Nevatia in [11]. Their algorithm is based on the hypothesis that each foreground region is a person, and uses a geometrical model for a person: human shape is modeled by a vertical 3-D ellipsoid with the two short axes of the same length and with a fixed ratio to the length of the long axis. By using this model the algorithm is able to recognize those parts of the foreground that have to be labeled as reflections. Unfortunately this method is not usable if the scene includes other kinds of objects, or even people carrying large objects such as backpacks, suitcases or umbrellas.

The recent paper by Karaman *et al.* [12] presents a more sophisticated method that takes into account both geometric and chromatic information to remove the reflections. The method is based on the "generate and test" approach, where for each detected foreground region several hypotheses are made on the vertical position of the object baseline. For each position, the algorithm generates a synthetic reflection by combining the pixels of the background and of the part of the region that lies above the baseline, adding a blur effect to take into account the imperfect smoothness of the floor surface. Then, the baseline for which the synthetic reflection is most similar to the observed one, is selected, and all the pixels below this baseline are removed from the foreground object. This method is fairly general and robust, since it does not require an *a priori* knowledge of the shape of the objects. On the other hand, the "generate and test" process is computationally expensive, because for each hypothesis an image has to be generated and matched with the observed region. Furthermore, the pixel combination and blurring require parameters depending on the characteristics of the floor, implicitly assuming that the floor smoothness and reflectivity are uniform.

In this paper we propose a reflection removal technique that is similarly based on the evaluation of multiple hypotheses for the object baseline. The proposed method does not make assumptions on the characteristics of the floor surface, and so can easily work with heterogeneous floors. Moreover, the method does not need to know the actual shape of the object, so it can be used even when the scene contains several kinds of objects. Last but not least, the method is extremely efficient because it does not involve the actual generation of a synthetic reflection, and the test phase exploits an incremental scheme of computation to evaluate each baseline very quickly.

II. The Proposed Method

In this section we will first provide a formal definition of the problem that will be cast as the optimization of a goal function. We will then demonstrate some properties of this goal function. Finally, we will examine in more details the actual algorithm, which incorporates some heuristics to filter out noise and is devised so as to reduce the computational complexity of the evaluation of the goal function.

A. Problem Formulation

We assume that our algorithm is applied to the output of a foreground detection system based on background subtraction. It does not require any specific background subtraction technique and can be used as a postprocessing phase of any existing foreground detection module.

We briefly recall that a foreground detection system compares the current frame to a background reference image (suitably created and updated), and finds the frame pixels whose color is significantly different from the corresponding background pixels, using some sort of thresholding technique. Such pixels are grouped into connected components called *foreground regions*. Each detected foreground region is described by means of its *bounding box*. The latter is defined as the smallest rectangle (whose sides are parallel to the edges of the frame) in which the region is inscribed. Our method assumes that each foreground region contains either a single object or a group of objects at the same distance from the camera (e.g., a person with his/her luggage). Each bounding box, together with the foreground pixels it contains, is the input of the reflection removal algorithm. The goal of the algorithm is to detect the horizontal line, hereinafter called *cut line*, that separates the object (formed by the pixels above the line) from its reflection (formed by the pixels below the line), so that the latter can be removed from the foreground mask and ignored in the following processing phases. The hypothesis that a single horizontal line can be used to delimit the reflection in a foreground region does not hold when two objects are vertically stacked, so that the reflection of one object overlaps a different one. This is a limitation of our method and, in general, of most techniques based on geometric assumptions. On the other hand, our algorithm has no problem when two or more persons or objects are horizontally adjacent.

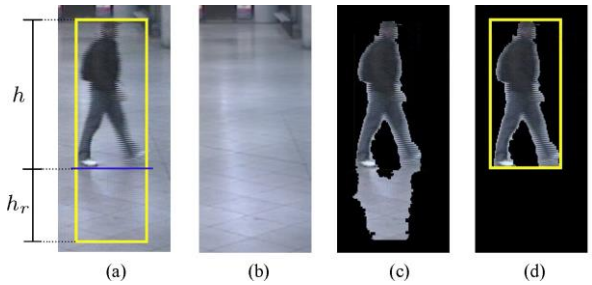

Fig. 1. (a) Portion of the input image containing a person whose height is h with its reflection on the floor whose height is h_r. The horizontal line represents the ideal cut separating the person from its reflection. (b) Background reference image. (c) Foreground mask. (d) Desired bounding box after the removal of the reflection.

The proposed method exploits the following property: reflection pixels are more similar than object pixels to the background; however, they are not so similar to be confused with the background by the foreground detection system. This happens because part of the color of the floor gets blended with the color of the reflected object to form the reflection color. Fig. 1 presents an example of a person with a reflection on the floor and the corresponding output of the foreground detection. The figure also shows the ideal cut line for this image and the background reference image.

On the basis of these assumptions, the ideal cut line is determined so that:

1) it minimizes the average difference in color between the detected object and the background, for all the rows below it;
2) at the same time, this line maximizes the average difference in color between the detected object and the background, for all the row above it.

Of course, in order to determine the optimal cut line, we need to reduce these two criteria to a single figure. A method commonly used in multiobjective optimization is the weighted sum of the objective functions. In our case, if we denote with D_y^a the average difference in color for all the rows above the row y, and with D_y^b the average difference in color for all the

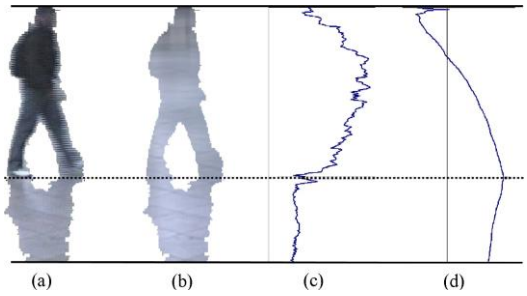

Fig. 2. (a) Foreground region and (b) corresponding background. (c) Function d_y for each row of the image (y is on the vertical axis, the value of d_y on the horizontal one). (d) Function O_y for each row of the image (y is on the vertical axis, the value of O_y on the horizontal one).

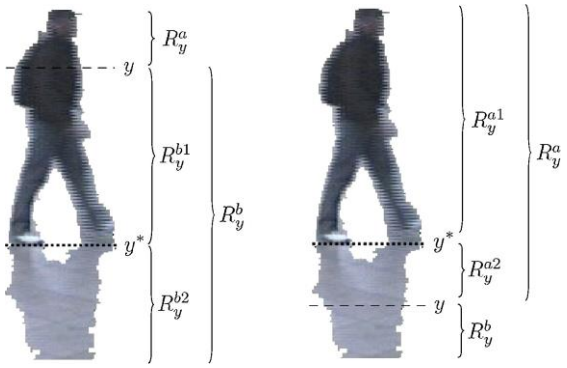

Fig. 3. Sets of pixel rows used in (3) and successive.

rows below y, then we can define a goal function O_y (to be maximized) as

$$O_y = c^a \cdot D^a_y + c^b \cdot D^b_y \tag{1}$$

where $c^a > 0$ and $c^b < 0$, because we want to maximize D^a and minimize D^b. These coefficients could be used to attribute a different weight to the criteria, if it were necessary for a specific application context. Since in our case we have no a priori reason to prefer one criterion over the other, we can set $c^a = +1$ and $c^b = -1$, thus reducing our problem to the maximization of $D^a_y - D^b_y$.

In order to quantitatively evaluate the difference in color, we introduce the following notations: $F(x, y)$ is the color of the pixel at position (x, y) in the foreground region, $B(x, y)$ the color of the corresponding pixel in the background image, and r_y the set of pixels belonging to the generic row y of the foreground region. We measure the average difference of color, along the row y of the detected foreground object, by the following quantity:

$$d_y = \frac{\sum_{(x,y)\in r_y} \|F(x,y) - B(x,y)\|}{|r_y|} \tag{2}$$

where $\|\cdot\|$ is the Euclidean norm in the RGB color space, and $|\cdot|$ is the cardinality of a set. Fig. 2(c) reports the graph representing d_y for any row y of the detected foreground image.

Given the definition of d_y, we can express D^a_y and D^b_y as follows:

$$D^a_y = \frac{1}{|R^a_y|} \sum_{i\in R^a_y} d_i \qquad D^b_y = \frac{1}{|R^b_y|} \sum_{i\in R^b_y} d_i \tag{3}$$

where R^a_y and R^b_y are the sets of rows above and below y, respectively.

Thus, we can express our goal function O_y as

$$O_y = D^a_y - D^b_y = \frac{1}{|R_y|} \cdot \sum_{i\in R^a} d_i - \frac{1}{|R_y|} \cdot \sum_{j\in R^b} d_j \tag{4}$$

Fig. 2(d) reports the graph representing O_y for any row y of the detected foreground image.

B. Properties of the Goal Function O_y

In this subsection we will demonstrate two important properties of O_y that confirm that it is a good choice for a goal function:

1) in the ideal case, that will be described later, O_y increases, starting from the top of the detected foreground object, reaching its maximum in correspondence with the ideal cut line, and then it decreases as y approaches the bottom of the reflection;
2) even in not ideal cases, as long as some assumptions on the the probability distribution of d_y are satisfied, it can be demonstrated that O_y is approximately equal to its expected value, which in turn has a maximum in correspondence with the ideal cut line.

In the most general case, when the mentioned assumptions on d_y are not satisfied, we cannot demonstrate analytically the optimality of our solution. However, the theoretical analysis is complemented by an experimental evaluation of our method, that will be presented in Section III, showing that the algorithm achieves a good performance in real cases.

The ideal case for the reflection removal is when all the pixels of the object have the same difference from the background, and also all the pixels of the reflection have the same difference, which is smaller. More formally, in the ideal case there are two constants μ_a and μ_b (with $\mu_a > \mu_b > 0$) such as $d_i = \mu_a$ for all the rows belonging to the object and $d_i = \mu_b$ for all the rows belonging to the reflection.

In order to demonstrate that in the ideal case the properties of O_y hold, let us suppose that both y and $y + 1$ are rows not at the border of the object, so that $|R^a_y| > 0$ and $|R^b_{y+1}| > 0$. Then, if we consider that $|R^a_{y+1}| = |R^a_y| + 1$ and $|R^b_{y+1}| = |R^b_y| - 1$, we can derive from (3)

$$D^a_{y+1} = \frac{1}{|R^a_y|+1} \left(d_{y+1} + \sum_{i\in R^a_y} d_i \right) = \frac{d_{y+1}}{|R^a_y|+1} + \frac{|R^a_y|}{|R^a_y|+1} D^a_y \tag{5}$$

Thus,

$$D^a_{y+1} - D^a_y = \frac{d_{y+1} - D^a_y}{|R^a_y| + 1}. \tag{6}$$

Analogously, we can derive from (3)

$$D^b_{y+1} - D^b_y = \frac{D^b_y - d_{y+1}}{|R^b_y| - 1}. \quad (7)$$

Now, if rows y and $y + 1$ are above the ideal cut line, since $d_{y+1} = D^a_y = \mu_a$, we have

$$D^a_{y+1} - D^a_y = 0. \quad (8)$$

On the other hand, $D^b_y < \mu_a$, because it is an average computed over a set containing the values μ_a and μ_b (where $\mu_b < \mu_a$); thus we have

$$D^b_{y+1} - D^b_y < 0. \quad (9)$$

As a consequence

$$O_{y+1} - O_y = (D^a_{y+1} - D^a_y) - (D^b_{y+1} - D^b_y) > 0 \quad (10)$$

and so we have demonstrated that O_y is increasing above the ideal cut line. Analogously, if rows y and $y + 1$ are below the ideal cut line, we have $d_{y+1} = D^b_y = \mu_b < D^a_y$, and so we obtain

$$D^a_{y+1} - D^a_y < 0 \quad D^b_{y+1} - D^b_y = 0 \quad (11)$$

$$O_{y+1} - O_y = (D^a_{y+1} - D^a_y) - (D^b_{y+1} - D^b_y) < 0 \quad (12)$$

demonstrating that O_y is decreasing below the ideal cut line.

While in real cases d_i is not a piecewise constant function, we can generalize this result to the less restrictive assumption that the values of d_i are independent random variables, with mean and standard deviation, respectively, equal to μ_a and s_a above the ideal cut line, and equal to μ_b and s_b (with $\mu_a > \mu_b$) below the ideal cut line. Notice that, since d_i is obtained as an arithmetic mean over the row i, it is reasonably well approximated by a Gaussian random variable (unless there are very few pixels on the row) because of the central limit theorem. This assumption is more general than the ideal case, but still simple enough to allow us to derive analytically a probabilistically good solution. Of course there are real cases where this assumption does not hold; for these cases we cannot prove the optimality or near-optimality of our solution, although the experimental validation (described in Section III) has shown that it is reasonably good on real world applicative scenarios.

Under the assumption described above, and remembering that the average and expectation operators can be exchanged we can compute the expected values of D^a_y and D^b_y (in the following, the notation $E[\cdot]$ will be used for the expectation, and Avg[·], for the average operator). If y is above the ideal cut line, R^a_y is made of a subset R^{b1}_y of rows above y^*, and a subset R^{b2}_y of rows below y^* (see Fig. 3). Thus

$$E[D^a_y] = E[\,\mathrm{Avg}_{i \in R^a_y}[d_i]\,] = \mathrm{Avg}_{i \in R^a_y}[E[d_i]] = \mathrm{Avg}_{i \in R^a_y}[\mu_a] = \mu_a. \quad (13)$$

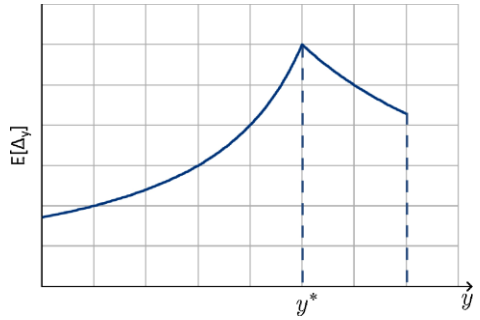

Fig. 4. Value of $E[O_y]$ as a function of y. $E[O_y]$ has a maximum in correspondence of the ideal cut line.

For D^b_y, still assuming $y < y^*$, we must consider that R^b_y is made of a subset R^{b1}_y of rows above y^*, and a subset R^{b2}_y of rows below y^* (see Fig. 3). Thus

$$E[D^b_y] = E[\,\mathrm{Avg}_{i \in R^b_y}[d_i]\,] = \mathrm{Avg}_{i \in R^b_y}[E[d_i]] = \frac{\mathrm{Avg}_{i \in R^{b1}_y}[E[d_i]] \cdot |R^{b1}_y| + \mathrm{Avg}_{i \in R^{b2}_y}[E[d_i]] \cdot |R^{b2}_y|}{|R^{b1}_y| + |R^{b2}_y|}. \quad (14)$$

Now, we can easily see that $|R^{b1}_y| = h - y$ and $|R^{b2}_y| = h_r$ (recalling that h is the height of the object, and h_r is the height of the reflection, see Fig. 1). Furthermore, $E[d_i]$ is equal to μ_a over the rows in R^{b1}_y, and to μ_b over the rows in R^{b2}_y. So we obtain

$$E[D^b_y] = \frac{(h - y) \cdot \mu_a + h_r \cdot \mu_b}{h + h_r - y} \quad (15)$$

$$y^* = \arg\max_y O_y. \quad (16)$$

For the case of y below the ideal cut line, we can easily derive

$$E[D^b_y] = \mu_b \quad (17)$$

while for D^a we have to consider that there are h rows in R^a_y that are above the ideal cut line, and $y - h$ rows below it. So, with a derivation similar to that of (15), we obtain:

$$E[D^a_y] = \frac{h \cdot \mu_a + (y - h) \cdot \mu_b}{y}. \quad (18)$$

Now, we can compute $E[O_y] = E[D^a_y] - E[D^b_y]$ by subtracting the previous formulas for the two cases of y above y^* and y below y^*

$$E[O_y] = \begin{cases} \dfrac{h_r \cdot (\mu_a - \mu_b)}{h + h_r - y} & \text{for } y \text{ above } y^* \\ \dfrac{h \cdot (\mu_a - \mu_b)}{y} & \text{for } y \text{ below } y^*. \end{cases} \quad (19)$$

Fig. 4 shows the function $E[O_y]$. This function is increasing to the left of the ideal cut line, and decreasing to the right of it, as it can be easily verified by taking the derivative of (19). Thus, the position of the ideal cut line corresponds to the

maximum of $E[O_y]$. Now, we cannot measure the value of $E[O_y]$, but only the value of $O_y = D^a_y - D^b_y$. However, since D^a_y and D^b_y are averages of the values of d_i, for the central limit theorem, they tend to be close to their expected values, except for values of y where the number of averaged values is small

$$D^a_y \approx E[D^a_y] \text{ except for } y \text{ near the top of the object} \quad (20)$$

$$D^b_y \approx E[D^b_y] \text{ except for } y \text{ near the bottom of the object.} \quad (21)$$

Thus, we have

$$O_y = D^a_y - D^b_y \approx E[D^a_y] - E[D^b_y] = E[O_y] \quad (22)$$

except for y near the top or the bottom of the object.

The ideal cut line y^* can be consequently determined by searching for the maximum of O_y.

C. Algorithm

In real cases, the O_y function is not as well-behaved as in the ideal case of Fig. 4, showing a few spurious maxima in addition to the one corresponding to the ideal cut line. These spurious maxima are due to the effect of noise and to dishomogeneity in the color of the foreground object, which may be locally very similar to the background color. In the following we will present the heuristics we have devised to take into account these effects, and then we will describe in detail the algorithm used for computing O_y and discuss its computational complexity.

To filter out the spurious maxima, we have introduced the following procedure, based on geometrical and physical considerations. The maximum is discarded under the following conditions.

1) It is too isolated: The rationale behind this criterion is that an isolated maximum is more likely due to noise than to the underlying trend of the function. A local maximum at row $\hat{y}$ is considered isolated if the average of O_y on a window centered on $\hat{y}$ is less than a fixed fraction of the value of O_y, that is

$$\frac{1}{2w} \sum_{\substack{i=\hat{y}-w \\ i\neq\hat{y}}}^{\hat{y}+w} O_i \leq \tau O_{\hat{y}} \quad (23)$$

where $2w$ is the width of the search window centered on $\hat{y}$ (in our experiments we have set w to 1/15 of the average height, in pixel, of a person) and τ is a parameter of the algorithm whose value is between 0 and 1 (the tuning of τ will be described in Section III).

2) The position of the maximum is above the middle of the detected foreground region: In fact, it is geometrically unlikely that a reflection is larger than the actual object, if the floor surface is horizontal and the object is not significantly inclined with respect to the vertical.
3) The value of the maximum is negative: A negative value of O_y would mean that the object is more similar to the background than its reflection, and this is incompatible with the nature of the reflection phenomenon.

Algorithm 1 Determination of the cut line.

```
{ Compute d and its sum }
Sum ← 0
for i = 0 to height − 1 do
    d_i ← Σ_{(x,y)∈r_i} ‖F(x,y) − B(x,y)‖/|r_i|
    Sum ← Sum + d_i
end for
{ Compute O }
SumAbove ← d_0
SumBelow ← Sum − d_0
for y = 1 to height − 1 do
    O_y ← SumAbove/y − SumBelow/(height − y)
    SumAbove ← SumAbove + d_y
    SumBelow ← SumBelow − d_y
end for
{ Compute the best local maximum among the ones satisfying the criteria of feasibility }
BestMax ← −1
BestCut ← −1
for y = height/2 to height − 1 do
    if O_y is a local maximum AND O_y > 0 AND O_y is not isolated then
        if O_y > BestMax then
            BestMax ← O_y
            BestCut ← y
        end if
    end if
end for
if BestMax > 0 then
    RETURN BestCut
else
    RETURN Nothing
end if
```

The algorithm that we have actually implemented, shown as Algorithm 1, is structured as follows:

1) it computes the values of d_i and keeps them in a data structure, so that each d_i is computed once; *height* corresponds to the total height of the region, indicated as $h + h_r$ (see Fig. 1);
2) while iterating over the rows for computing O_y, the algorithm keeps in two variables (*SumAbove* and *SumBelow*) the sum of the d_i above and below row y;
3) the algorithm finds the best local maximum of O_y, taking into account the criteria previously introduced.

It is important to notice that the proposed algorithm differs from a naive implementation in that it computes the function more efficiently. Indeed, a naive calculation of O_y would require to scan the whole detected region for each value of y, in order to compute the average difference with respect to the background above and below the yth row. Since this process would have to be repeated for each row, the resulting complexity would be $O(w \cdot h^2)$, where w and h are the width and the height of the region, respectively.

In our algorithm, the fact that the values of d_i are computed only once reduces the computational complexity to $O(w \cdot h + h^2)$, where the first term is due to the computation of d_i and the second term to the computation of O_y given the d_i values. Furthermore, the computation of O_y uses the variables *SumAbove* and *SumBelow*, that can be updated in

$O(1)$ at each step, avoiding the iteration over d_i for calculating the two sums of (4); hence, the overall complexity is reduced to $O(w\,h + h) = O(w\,h)$.

Thus, the proposed algorithm is very efficient even on large foreground regions, requiring a time that is negligible with respect to the overall processing of a frame.

III. Experimental Evaluation

In this section, we present and discuss the behavior of the proposed method. The tests were carried out on a set of sequences extracted from real-world videos. Furthermore, we also compare our method with respect to two recent and effective approaches for reflection removal on the same dataset.

For the sake of comparability, the experimental validation of the proposed method has been carried out using only publicly available videos. In particular, the test dataset contains four videos all referring to indoor scenarios with reflecting floorings. All videos were acquired at 4 CIF resolution and 25 fps. Unfortunately, there are no publicly available video sequences containing outdoor scenes with reflections. However, it should be also noted that the issue of outdoor analysis in case of reflection is quite marginal. In fact, reflections occur much more frequently in indoor scenarios than in outdoor ones. This can be simply explained by looking at the type of flooring used in typical indoor and outdoor areas under video surveillance. In public indoor areas (shopping malls, railway stations and airport halls, metro platforms, etc.) smooth floorings are used very often. Such floorings are prone to generate reflections. On the contrary, in outdoor scenarios rugged floorings are usually found; these floorings do not cause reflections with the exception of some specific situations (for instance in presence of puddles).

The first test video (hereinafter referred to as *AVSS* belongs to the dataset published during the *International Conference on Advanced Video and Signal-based Surveillance AVSS 2007* [13]; the scene shows a subway station. The second video sequence (hereinafter referred to as *CAVIAR*) is taken from the public dataset *CAVIAR* [14], widely used for video surveillance systems evaluation; the scene shows a shopping mall. Last, the third and the forth video sequences (hereinafter referred to as *PETS-1* and *PETS-2*), belong to the dataset published at the *International Workshop on Performance Evaluation of Tracking and Surveillance PETS2006* [15]; in particular we have used the S1-T1-C1 and S1-T1-C3 sequences. These videos show the hall of a railway station, from two different angles.

For each video, a ground truth has been produced by inspecting the objects detected in each frame and choosing by hand the expected cut line. The ground truth has been produced using the ViPER Ground Truth Authoring Tool [16], which allows frame-by-frame markup of video metadata. We had three independent persons examine the single frames to make the ground truth, taking the average of the cut line positions they provided. The obtained files, together with other material used for the experimentation, are available at [17].

Object detection was carried out running the algorithm described in [18]. We chose to use this algorithm as it is characterized by a good trade-off between the detection performance and the computational complexity [19]. Note that the objects missed by the detection algorithm, as well as the wrongly detected ones (those corresponding to partial detections of the persons) have been discarded, since reflection removal methods cannot recover from such errors. Table I reports the main features of the video sequences: the length expressed in terms of the number of frames and the number of objects.

TABLE I
The Value of the Mean Absolute Error Before (MAE_b) and After (MAE_a) the Application of the Method, and the Obtained Relative Improvement (I)

Sequence	No. of frames	No. of objects	MAE_b	MAE_a	I
AVSS	5474	2531	0.160	0.053	66.9%
CAVIAR	389	349	0.262	0.063	75.9%
PETS-1	3021	1610	0.601	0.286	52.4%
PETS-2	3021	2931	0.071	0.018	74.6%

A. Performance Indices

In the ideal case, after the application of the proposed approach the residual value of the height of the reflection $h_r(i)$ (Fig. 1) should be equal to zero. However, in the real case we have to consider the following types of errors:

1) the algorithm fails to completely remove the reflection of a detected object; we call the occurrence of this phenomenon *undercut*;
2) the algorithm completely removes the reflection, but also cuts away part of the object; we call the occurrence of this phenomenon *overcut*.

We denote with $e(i)$ the error in the estimation in the height of the ith object; it is calculated as the difference between the height of the object after the application of the method and its expected ideal height $h(i)$. Note that in case of *undercut* $e(i) > 0$, while in presence of *overcut* $e(i) < 0$.

The index used to report the performance is the mean absolute error (MAE), that is the absolute value of the relative error $RE(i) = e(i)/\,h(i)$ averaged over the video sequence and is defined as

$$MAE = \frac{1}{N} \cdot \sum_{i=1}^{N} |RE(i)| \qquad (24)$$

where N is the number of objects of the test sequence.

B. Performance Analysis

To assess the performance of the proposed method it is required to firstly set the value of the threshold τ used to filter out isolated maxima. For each video, we selected the value of τ that maximized MAE over a short excerpt (about 5%) of the sequence. Obviously, the subsequences used for tuning were excluded from the tests.

Table I reports the results obtained by the proposed method on the test videos. For each sequence we consider the value of the MAE before the application of our method (the performance is reported in the column denoted by the label MAE_b)

and after its application (column with label MAE_a). We also report the relative improvement I, calculated as

$$I = \frac{MAE_b - MAE_a}{MAE_b} . \quad (25)$$

The results in Table I reveal a consistent reduction of the reflections error on each test sequence. The error decreases by approximately 70% on all videos with the exception of the *PETS-1* video where the error decrease is slightly higher than 50%. It is interesting to note that the proposed method is able to guarantee a significant performance improvement not only on videos affected by very strong reflections, as in the case of the *PETS-1* sequence, but also on videos where the incidence of the reflection error is less evident, as on the *PETS-2* video. This behavior is more evident by considering the plots in Fig. 8, which show the histograms of the values of *RE* before and after the application of the proposed method over the four test sequences, together with the results of two other methods selected for comparison (more details in the next subsection). It is interesting to note how in all cases the peak of the histograms related to our method is located around zero, meaning that the residual detection error h_r is negligible. Plots in Fig. 8 allow also to analyze qualitatively how the residual detection error h_r is distributed between *undercut* and *overcut*. From all the plots, it is evident that even if the *overcut* phenomenon occurs quite frequently its incidence is very limited: in fact, it is easy to verify that in most cases $-10\% < RE < 0\%$.

In Fig. 5 we have reported an example that demonstrates the ability of the proposed method to remove reflections also in complex situations. In particular, Fig. 5(a) shows a frame from the *AVSS* sequence with two women that walk together so that they are detected as a single object. Fig. 5(b) shows the data used by the algorithm to find the cut line for the person: the cut line proposed by our approach coincides with the ideal cut line, and are both represented by the continuous horizontal line. In this type of situation, since the method does not make any assumption about the shape of the reflected object, the removal of the reflection related to a group of objects at the same distance from the camera is not different from the general case: thus, the same performance should be expected. The dataset proposes other difficult situations that our method is able to correctly solve. For instance, the video from the *CAVIAR* database contains a two-colored floor. The proposed algorithm works adequately in this condition. Indeed, a multicolor background does not pose a significant problem, as long as the background colors are reasonably different from the object color; if this is not true, then the main problem would be the camouflage, causing a degradation in the object detection performance that is more critical than the reflection removal.

C. Performance Comparison

In the experiments we compared our method with two other approaches. We have chosen the algorithm by Teschioni *et al.* [9] and the algorithm by Karaman *et al.* [12] because they are among the most representative algorithms within their category according to the taxonomy discussed in Section I.

TABLE II
PERFORMANCE OBTAINED BY THE COMPARED ALGORITHMS WHEN THE CORRESPONDING PARAMETERS VARY: THE BEST AND AVERAGE PERFORMANCE AND THE ONE OBTAINED WITH A LEAVE-ONE-OUT PROCEDURE

Video	*Index*	*MAE*		
		OUR	TES	KAR
AVSS	Best	**0.053**	**0.053**	0.076
	Mean	**0.067**	0.162	0.132
	L-Out	**0.055**	0.454	0.151
CAVIAR	Best	**0.063**	0.128	0.139
	Mean	**0.110**	0.269	0.166
	L-Out	**0.065**	0.230	0.205
PETS-1	Best	0.286	0.540	**0.276**
	Mean	0.373	0.626	**0.364**
	L-Out	*0.287*	0.582	0.371
PETS-2	Best	*0.018*	0.060	0.043
	Mean	*0.025*	0.089	0.053
	L-Out	**0.023**	0.072	0.047

Hereinafter, we will refer to the approaches in [9] and [12] as TES and KAR, respectively.

Each of the considered algorithms has a certain number of parameters to be tuned for optimal performance. For the sake of conciseness, details about such parameters are here omitted: the interested reader can refer to the original papers.

Obviously, the best performance is achieved when the parameters are tuned to the very scene the system is applied to. However, in real cases this is often unpractical, and the system has to work with parameters chosen in a similar but not identical context. From these considerations, in order to compare our method with respect to TES and KAR, we made the following two experiments.

1) The reflection removal algorithms are separately tuned on each video; in a real operational scenario this means that tuning is carried out during the deployment of each single camera.
2) Tuning is done once on a set of videos referring to different camera views; then the system is tested on a video not present in the training set. This corresponds to the case of a system that is provided *as-is* without requiring a tuning for each new installation.

In the first experiment, we used a grid search for determining the set of parameter values that maximized the index *MAE* for each algorithm on each video. The performance obtained in this configuration is reported in Table II on the rows labeled as *Best*. The proposed method achieves the best performance on three videos out of four, with the exception represented by the *PETS-1* video on which KAR performs the best. Nevertheless, it has to be noted that on this video the relative difference of the performance between our method and KAR is quite negligible (less than 4%).

In order to analyze and compare the robustness of the algorithms with respect to the choice of the parameters, in Table II we have reported also the average of the performance indices obtained using different values of the algorithms parameters. From the considered results it emerges evidently that the proposed method is more robust than KAR and

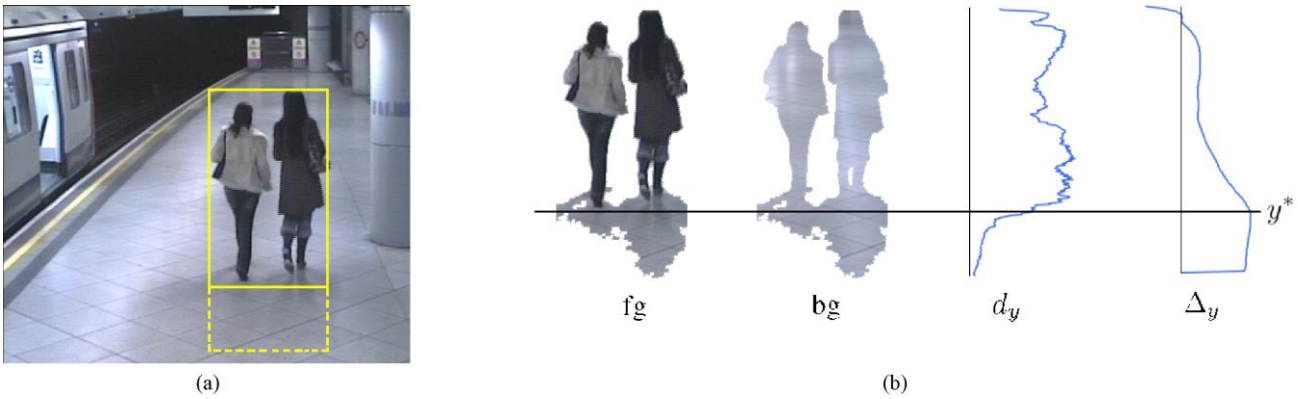

Fig. 5. (a) Frame from the *AVSS* sequence and the corresponding bounding box before (dashed) and after (solid) the application of our algorithm. (b) Data used for determining the cut line. In this specific case, the cut line proposed by our approach coincides with the ideal cut line: both lines are represented by the continuous horizontal line.

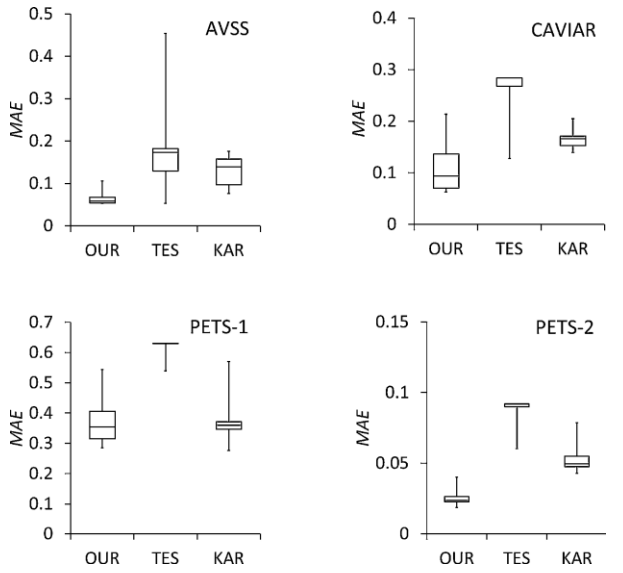

Fig. 6. Box-plots of the compared algorithms. Vertical lines represent the range of the MAE, while the boxes show the second and the third quartiles.

TES, with respect to the choice of parameter values. This is better highlighted in the box-plots in Fig. 6 where for each test sequence and for each algorithm the values of *MAE* obtained for all the different configurations of the parameters are grouped in quartiles.

In Fig. 7 we show some plots that are useful to visualize the correlation between the value of the detection error before and after the application of the considered algorithms. The gray level of the pixel (x,y) indicates the number of samples for which the initial detection error was x, while y was the residual error after the application of the method. Furthermore, the two diagonal lines delimitate the area in which the algorithm does not increment the absolute value of the error (the more are the samples within this area, the better is the behavior of the algorithm). Finally the horizontal line highlights the ideal performance when all the reflection errors are completely eliminated with neither undercut nor overcut. The first observation is that the proposed method shows a *bimodal* behavior: in fact, the samples in the plots in the first column of Fig. 7 are concentrated either along the horizontal line or along the upper diagonal line. The first cases are those in which the algorithm removes the reflection, and does so with a very small residual error. The second cases are those in which the algorithm is not able to find the best cut line, and leaves the image unchanged avoiding the risk of an overcut; this is mainly due to the filtering stage of our method that discards the isolated maxima of the function O_y. A similar bimodal behavior is shown also by TES, but in this case the number of reflection errors not corrected by the method is much higher than by our method. On the contrary, KAR is somehow *fuzzy*, with the points in the plots distributed on a wider area than the previous two approaches.

For the second experiment, we carried out tests using the *leave-one-out* procedure: for each test sequence and for each algorithm the optimal values of the parameters were determined as those which maximized *MAE* over the remaining three videos. The performance obtained in this configuration is reported in Table II on the rows corresponding to the configuration named *L-Out*. Again it is possible to notice that the proposed method outperforms the other algorithms in all cases, showing that it remains successfully usable even when an accurate calibration on the actual view is not feasible.

A further confirmation comes from the comparison between the rows labeled *Best* and *L-Out* in Table II: due to the robustness of our approach, we can notice that even going from a tuning procedure carried out *on the field* to one performed *in the lab*, the overall performance is not significantly affected. In fact, in all cases the relative improvement on *MAE* which can be obtained by tuning performance on each single camera with respect to the case of doing it once is negligible. Such behavior is better highlighted in Fig. 8 which shows the distribution of the *RE* values measured on the test sequences for each considered algorithm. It comes out evident that the proposed method outperforms the others in both the configurations used for tuning: in fact, in both cases the distributions are well centered around zero with small values on the tails. Furthermore, from the direct comparison of the plots related to the same sequence with different tuning procedures, it emerges that our approach results more robust than KAR and TES, as the respective plots do not change significantly.

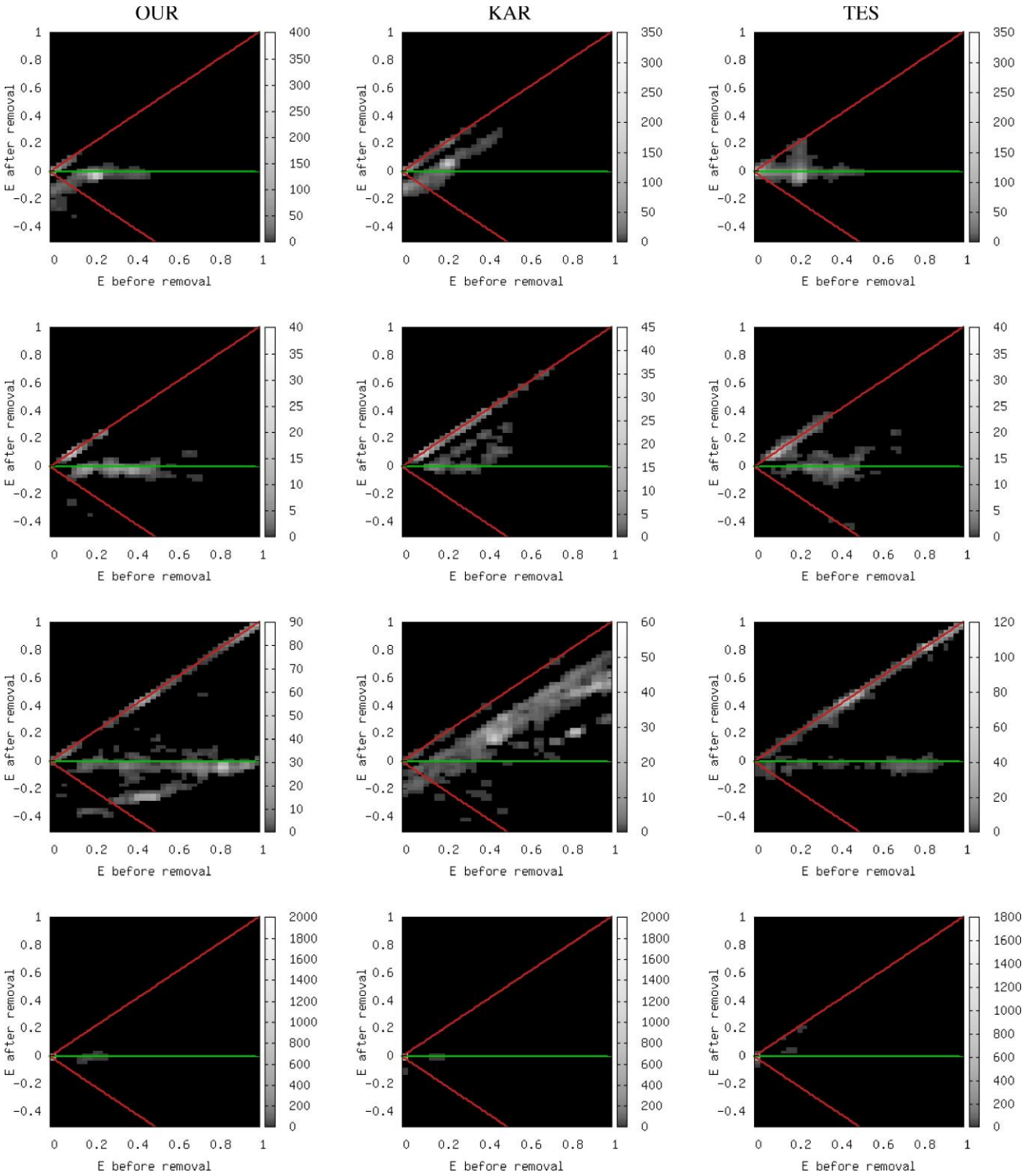

Fig. 7. Gray level of the pixel (x,y) indicates the number of samples for which the initial detection error was x, while y was the residual error after the application of the method. The diagonal lines delimitate the area in which the algorithm does not increment the absolute value of the error, while the horizontal line highlights the ideal performance. The plots refer from top to bottom to *AVSS*, *CAVIAR*, *PETS1*, *PETS2*.

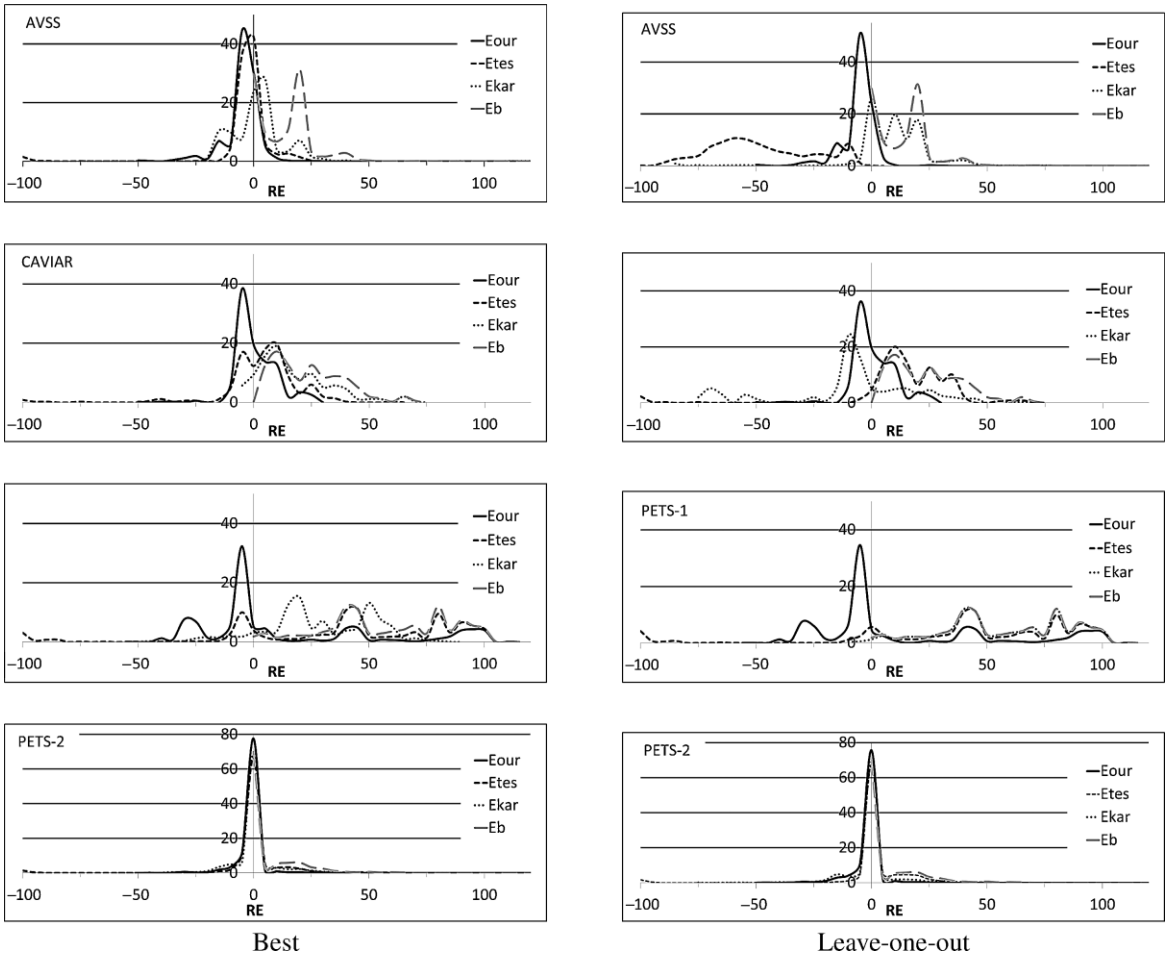

Fig. 8. Distribution of the relative error. *Eb* is the error before reflection. *Eour*, *Etes*, *Ekar* are the residual errors after the application of the considered methods. The plots on the left are obtained with the best parameter vales. The plots on the right are for the leave-one-out procedure.

Finally, we provide some notes about the processing times. As anticipated in Section II, the proposed reflection removal method has a negligible impact on the overall processing time. In fact, we have experimentally verified that the adoption of the reflection removal procedure produces an increase of 1.5% of the processing time with respect to the original foreground detection algorithm. This result is coherent with the expectations as the proposed implementation has a computational cost that is proportional to the overall number object pixels. In particular, there is approximately one object per frame in the considered test sequences (see Table I); furthermore the average size of the objects is around 1/100th of the frame size ranging between 1/200th to 1/50th.

Notice that TES has a similar processing time, while the processing time of KAR is about one order of magnitude higher than the other algorithms.

IV. Conclusion

In this paper we have presented a novel algorithm for reflection removal, based on fairly general assumptions on the chromatic and geometric properties of reflections, without requiring prior knowledge of the shape of the reflected objects. The algorithm has been designed to be at the same time fast and accurate. The proposed method has received an extensive experimental evaluation using a significant database of real videos, on which its performance has been compared with two state-of-the-art algorithms from the literature. The results of this experimentation have confirmed the effectiveness of the proposed method, which achieves a greater accuracy than the most sophisticated of the two other algorithms, with a computational cost comparable to the fast but less accurate other one. Future work will be devoted to the test of the proposed method on a larger dataset including also outdoor scenes videos.

References

[1] T. Horprasert, D. Harwood, and L. S. Davis, "A statistical approach for real-time robust background subtraction and shadow detection," in *Proc. 7th IEEE Int. Conf. Comput. Vision*, Sep. 1999, pp. 1–19.

[2] J. Shen, "Motion detection in color image sequence and shadow elimination," *Visual Commun. Image Process.*, vol. 5308, pp. 731–740, Jan. 2004.

[3] C. Jiang and M. O. Ward, "Shadow segmentation and classification in a constrained environment," *CVGIP: Image Understanding*, vol. 59–2, pp. 213–225, Mar. 1994.

[4] Y. Sonoda and T. Ogata, "Separation of moving objects and their shadows, and application to tracking of loci in the monitoring images," in *Proc. IEEE Int. Conf. Signal Process.*, Oct. 1998, pp. 1216–1264.

[5] A. Cavallaro, E. Salvador, and T. Ebrahimi, "Detecting shadows in image sequences," in *Proc. IEEE CVMP*, Mar. 2004, pp. 15–16.
[6] G. Finlayson, S. Hordley, C. Lu, and M. Drew, "On the removal of shadows from images," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 1, pp. 59–68, Jan. 2006.
[7] I. Huerta, M. Holte, T. Moeslund, and J. González, "Detection and removal of chromatic moving shadows in surveillance scenarios," in *Proc. IEEE 12th Int. Conf. Comput. Vision*, Sep.–Oct. 2009, pp. 1499–1506.
[8] J. Stauder, R. Mech, and J. Ostermann, "Detection of moving cast shadows for object segmentation," *IEEE Trans. Multimedia*, vol. 1, no. 1, pp. 65–76, Mar. 1999.
[9] A. Teschioni and C. S. Regazzoni, "A robust method for reflection analysis in color image sequences," in *IX European Signal Processing Conference (Eusipco98)*, G. C. S. Regazzoni, G. Fabri, Ed., Dordrecht, the Netherlands: Kluwer Academic Publishers, 1998, pp. 76–90.
[10] E. J. Carmona, J. Martinez-Cantos, and J. Mira, "A new video segmentation method of moving objects based on blob-level knowledge," *Pattern Recognit. Lett.*, vol. 29, pp. 272–285, Feb. 2008.
[11] T. Zhao and R. Nevatia, "Tracking multiple humans in complex situations," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 9, pp. 1208–1221, Sep. 2004.
[12] M. Karaman, L. Goldmann, and T. Sikora, "Improving object segmentation by reflection detection and removal," in *Proc. VCIP, IS&T/SPIE's Electronic Imaging*, R. L. S. Majid Rabbani, Ed., vol. 7257, Jan. 2009.
[13] *i-Lids dataset for AVSS 2007* [Online]. Available: http://www.eecs.qmul.ac.uk/~andrea/avss2007_d.html
[14] *CAVIAR dataset* [Online]. Available: http://homepages.inf.ed.ac.uk/rbf/CAVIARDATA1/
[15] *PETS2006 dataset* [Online]. Available: http://www.cvg.rdg.ac.uk/PETS2006/
[16] *ViPER: The Video Performance Evaluation Resource* [Online]. Available: http://viper-toolkit.sourceforge.net/
[17] *Ground Truth for Reflection Removal Algorithms Evaluation* [Online] Available: http://nerone.diiie.unisa.it/zope/mivia/databases/db_database/4/
[18] D. Conte, P. Foggia, M. Petretta, F. Tufano, and M. Vento, "Evaluation and improvements of a real-time background subtraction method," in *Lecture Notes in Computer Science*, vol. 3704, M. Kamel and A. Campilho, Eds. Berlin/Heidelberg, Germany: Springer-Verlag, 2005, pp. 1234–1241.
[19] D. Conte, P. Foggia, G. Percannella, F. Tufano, and M. Vento, "An experimental evaluation of foreground detection algorithms in real scenes," *EURASIP J. Adv. Signal Process.*, vol. 2010, p. 10, Feb. 2010.