Learning-based Shadow Recognition and Removal from Monochromatic Natural Images

Mingliang Xu, Member, IEEE, Jiejie Zhu, Member, IEEE, Pei Lv, Member, IEEE, Bing Zhou, Member, IEEE, Marshall F. Tappen, Member, IEEE and Rongrong Ji, Senior Member, IEEE

Abstract—This paper addresses the problem of recognizing and removing shadows from monochromatic natural images from a learning based perspective. Without chromatic information, shadow recognition and removal are extremely challenging in the literature, mainly due to the missing of invariant color cues. Natural scenes make this problem even harder due to the complex illumination condition and ambiguity from many near-black objects. In this paper, a learning based shadow recognition and removal scheme is proposed to tackle the challenges above. Firstly, we propose to use both shadow-variant and invariant cues from illumination, texture and odd order derivative characteristics to recognize shadows. Such features are used to train a classifier via boosting a decision tree and integrated into a Conditional Random Field, which can enforce local consistency over pixel labels. Secondly, a Gaussian model is introduced to remove the recognized shadows from monochromatic natural scenes. The proposed scheme is evaluated using both qualitative and quantitative results based on a novel database of hand-labeled shadows, with comparisons to the existing state-of-the-art schemes. We show that the shadowed areas of a monochromatic image can be accurately identified using the proposed scheme, and high-quality shadow-free images can be precisely recovered after shadow removal.

Index Terms—Shadow Recognition, Shadow Removal, Monochromatic Image, Nature Scene, Decision Tree, Conditional Random Field, Gaussian Model.

I. INTRODUCTION

Shadows are among the most noticeable effects on understanding the structure and semantic of a natural scene. While shadows can provide useful cues regarding scene properties such as object size, shape, and movement [16], their existence has also complicated various image recognition tasks for natural images, ranging from object recognition to scene parsing [4], [35], [32], [29] and removal [9], [2], [30], [3].

While promising progress has been made in the literature, most existing works still retains on chromatic information for shadow recognition and removal. For instance, Finlayson et al. [9] located and removed the shadows using an invariant color model. Shor and Lischinski [30] propagated the shadows from pixels to a region using a color-based region growing method. Salvador et al. [28] adopted invariant color features to segment cast shadows. Levine et al. [17] and Arévalo et al. [3] both studied the color ratios across boundaries to assist shadow recognition. Note that a serial of schemes have also been proposed to detect shadows in videos by using color-related motion cues [24], [23], [34], [19], [14].

Fig. 1. Ambiguity of shadow recognition in monochromatic domain. The diffuse object (swing) in (a) and the specular object (car) in (b) both have a dark albedo. The trees in the aerial image (c) appear black because of self-shading. Such objects are very difficult to be separated from shadows which are also relatively dark.

One fundamental assumption for chromatic-based shadow recognition and removal is that the chromatic appearance of image regions does not change across shadow boundaries, while the intensity component of a pixel does. Such approaches work well given color images where object surfaces are still discernible inside the shadow. However, such schemes cannot be deployed to monochromatic images where shadows also prevalently exist. To the best of our knowledge, shadow detection and removal for monochromatic images is left unexploited. In particular, there exist two particular challenges:

• Objects in monochromatic images are more likely to appear black or near black, as shown in Figure 1. Such black objects exhibit similar intensity statistics to the shadow regions to be removed, which increases the difficulty in recognizing and removing shadows.
• Shadow recognition highly relies on separating changes in albedo from illumination change. Therefore, chromatic information is the most important cue, which is however missing in monochromatic images.

Despite the above difficulties, detecting and removing shadow in monochromatic images are still highly demanded in real-world applications. On one hand, color is not always available in all types of sensors, such as sensors tuned to specific spectra, or designed for specific usages like aerial, satellite, and celestial images, as well as sensors to capture images from very high dynamic scenes. On the other hand, it is widely known that humans are still able to identify shadows in monochromatic images [11], which inspires us to exploit the underlying principles in shadow modeling from monochromatic images.

In this paper, we present a learning-based approach for detecting and removing shadows from a single monochromatic image. The shadow recognition approach is based on boosted decision tree classifiers and integrated into
a Conditional Random Field (CRF). Finally, we further present a Gaussian model based method for removing shadows, which uses less parameters than previous work. Different from the existing works, we target at capturing the unique statistics of shadows within monochromatic images by exploiting both shadow-variant and invariant cues in a data-driven, learning-based manner. Given sufficient training data, our approach is more general than previous works that exploited grayscale cues to remove general illumination effects \cite{4, 35, 32}, as quantitatively evaluated on vast natural images ranging from highways, parking lots to indoor office scenes. Comparing to the most related work proposed in \cite{25}, the training set improved or separating shading and reflectance changes is still limited to small objects, while our work focuses on shadows in large-scale outdoor scenes.

The remainder of this paper is organized as follows: Section II introduces the features we extracted for shadow recognition. Section III formulates the proposed learning model. Section IV discusses two practical issues of feature boosting for accuracy improvement and parallel training for acceleration. Section V presents the experimental results with comparisons to alternative schemes. Finally, Section VII concludes this paper. Along with this paper, we have also released the dataset and all the codes, which are available from http://sites.google.com/site/jiejiezhu1/Home/project_shadow.

II. MONOCHROMATIC BASED FEATURE EXTRACTION

A. Dataset

We have built a database consisting of 355 images\footnote{All the dataset used in the paper can be downloaded from http://sites.google.com/site/jiejiezhu1/Home/project_shadow}. For each image, the shadows have been handcraft-labeled at the pixel level, which are further cross-validated by a group of volunteers to ensure the label precision. In this database, approximately 50\% images were collected from various outdoor environments such as campus and downtown areas. To ensure a wide variety of lighting conditions, we also collected images at different times throughout the day. The dataset includes additional 74 aerial images with a size of 257 × 257 from Overhead Imagery Research Dataset (OIRDS) \cite{31} and another 54 image with a size of 640 × 480 from LabelMe dataset \cite{27}\footnote{The camera we used to capture part of the images in the database is a Canon Digital Rebel XTi. The camera model used for the shadow images from the LabelMe database and from the OIRDS database are not reported.}. Figure 2 shows several examples.

Compared to the database used in our previous work \cite{13}, the shadows in the newly added 110 images are extremely challenging for recognition (see the last row in Figure 2). Our motivation is to check how good the learning methods introduced in this paper perform on this challenging scenes. Nevertheless, results from the previous work can be also used as a baseline for comparison purpose.

\begin{figure}[h]
\centering
\includegraphics[width=0.5\textwidth]{example.png}
\caption{Example of shadows in the dataset. The images captured by the authors are linearly transformed from raw format to TIF format. The images from LabelMe are in JPG format. The images from OIRDS are in TIF format. All the color images are converted to gray scale images. The last row shows some challenging examples.}
\end{figure}

B. Feature Extraction

Given a single monochromatic natural image, our goal is to identify those pixels that are associated with shadows. Without chromatic information, the features should be able to identify illumination, texture and odd order derivative characteristics. Rather than using pixel values alone, we also include the features from homogeneous regions generated by over-segmentation using intensity values \cite{20}. To reliably capture the cues across shadow boundaries, following \cite{6}, we collect statistics of neighboring pairs of shadow/non-shadow segments from all individual images in the dataset. These statistics are represented as histograms from shadow and non-shadow segments given equal bin centers.

We propose three types of features: shadow-variant features that describe different characteristics in shadows and in non-shadows; shadow-invariant features that exhibit similar behaviors across shadow boundaries and near-black features that distinguish shadows from near-black pixels. The above complimentary cues are further integrated to provide strong predictions of shadows jointly. For example, if the segment has invariant texture (compared with its neighbor) but has variant intensities, it is more likely that this segment belongs to shadows. Furthermore, each type of feature is characterized by scalar values to provide information relevant to shadow properties. Below we introduce in details the above features:

C. Shadow-Variant Features

\textbf{Intensity:} Since shadows are expected to be relatively dark, we gather statistics (Figure 3 (a)) about the intensity of pixel values, based on which a histogram is calculated and further augmented with the averaged intensity value and the standard deviation inside a segment.

\textbf{Local Max:} In a local patch, shadows have very low intensity values. Therefore, its local max value is expected to be small. On the contrary, non-shadows often have high intensity values and its local max value is expected to be large (Figure 3 (b)). We capture this cue by a local max completed at each segment.
Smoothness: Shadows are often smoothed among neighborhood pixels. This is because shadows tend to suppress local variations on the underlying surfaces. We use the method proposed by Forth and Fleck [21] to capture this cue. The method subtracts a smoothed version from the original image. To measure the smoothness, we also use the standard deviation of the smoothed version from each segment.

Skewness: We gathered several statistical variables (standard deviation, skewness and kurtosis) to find a mean value of 1.77 for shadows and -0.77 for non-shadows in skewness empirically. It indicates that the asymmetries in shadows and in non-shadows are different, which helps to locate shadows. This odd order statistic is also found to be useful in extracting reflectance and gloss from natural scenes [12] [1].

D. Shadow-Invariant Features

Gradient Similarity: We have observed that transforming the image with a pixel-wise log transformation makes the shadow being an additive offset to the pixel values in the scene. Therefore, the distribution of image gradient values is expected to be invariant across shadow boundaries. To capture this cue, we measure the similarity between the distributions of several first order derivatives of Gaussian filters.

Texture Similarity: We have also observed that the textural properties of surfaces are changed little across shadow boundaries. We measure the textural properties of an image region using the method introduced in [20]. The method filters a database of images with a bank of Gaussian derivative filters with 8 orientations and 3 scales, and then applies clustering to form 128 discrete centers. Given a new image, the texon is assigned as the histograms binned at these discrete centers.

As shown in Figure 5 (f) where the color indicates the texon index selected at each point, the distributions of textons inside and outside shadows are similar. The primary difference is that the distortion artifacts in the darker portions of the image lead to a slight increase in the number of lower-index textons, as indicated by more blue pixels.

E. Near-Black Features

As shown in Figure 3 (a), the pixel intensity is a good indicator of the shadow, which is usually dark. Unfortunately, this heuristic alone cannot be used to reliably identify shadows as it fails when facing dark objects. We have found that objects with a dark albedo are some of the most difficult image regions to separate from shadows. Similarly, tree regions are also difficult because of the complex self-shading caused by the leaves. Due to the complexity of hand-labeling, the self-shading within a tree is not considered to be a shadow. We refer to these objects as near-black objects (see Figure 1). To correctly distinguish shadows from near-black objects, we introduce two additional features:

Discrete Entropy: We have observed that the entropy of patches inside shadows differs from patches inside objects. The entropy is computed for each segment using the formula below:

\[ E_i = \sum_{x \in \omega} -p_i \times \log_2(p_i), \]

where \( \omega \) denotes all the pixels inside the segment, \( p_i \) is the probability of the histogram counts at pixel \( i \).

Figure 4(a) shows the distribution of the entropy values in patches from shadows and near-black objects respectively. These histograms show that the entropy of diffuse objects with dark albedos are relatively small, as most black objects are textureless in natural scenes. Figure 4(a) also shows histograms of the entropy of patches from specular objects (shown in blue) and image patches from trees (shown in green). There is a small mode at lower entropy values in the histogram from tree patches. Explanation lies in that some parts of trees are over-exposed to the sun, leading to saturated areas with low entropy.

Edge Response: Another feature we found useful is the edge response, which is often small in shadows. Figure 4 (b) shows an example where segments in shadows have near zero edge response, while that of specular object (the body of the car) has a strong edge response. We quantize...
this cue by summing up edge responses (computed using the Canny edge with a threshold setting to 0.01) inside a segment. Figure 5 shows a typical scene to visualize all 8 proposed features.

III. LEARNING TO RECOGNIZE SHADOWS

From the dataset introduced in Section II-A, we randomly selected 125 images as training data, and used the remaining 120 images as test data. We formulate the shadow detection problem as a per-pixel binary labeling problem, where every pixel is classified as either being inside shadow or not. Formally speaking, our goal is to assign \( y_i \in \{-1, +1\} \) for all pixels \( i \). Here, a Logistic Random Field (LRF) model is adopted, whose parameters are trained in the learning process to specify the decision boundary.

A. Logistic Random Field

The Logistic Random Field is essentially a logistic regression model, which is generalized to the conditional random field [33]. If we regard each pixel as a sample, the probability that a pixel should be labeled as shadows +1 given its observation \( x \) is:

\[
p(+1|x) = \sigma(w^Tf).
\]

(2)

The vector \( w \) is a weighting vector that defines a decision boundary in the feature space and \( f \) is the feature of this pixel. The function \( \sigma(\cdot) \) is the logistic function:

\[
\sigma(a) = \frac{1}{1 + \exp(-a)}.
\]

(3)

Logistic regression can be viewed as converting the linear function \( w^Tf \), which ranges from \(-\infty\) to \(+\infty\), into a probability, which ranges from 0 to 1.

To generalize Logistic Regression to a random field model, we introduce response image, \( r^* \). \( r^* \) is a linear response map computed by \( w^Tf \). We express the probability of a pixel belong to shadow as \( l_i = \sigma(r^*_i) \).

We use a pairwise CRF to model the labeling problem, which provides an elegant means to enforce local consistency and smoothness. The conditional distribution over labels \( l \), given an input image \( o \), has a form of

\[
P(l|o) = \frac{1}{Z} \prod_{i} \phi_i(l_i|a_i) \prod_{<i,j>} \psi_{i,j}(l_i,l_j|a_i),
\]

(4)

where \( \prod_{<i,j>} \) denotes the product of all pairs of neighboring pixels, both horizontal and vertical. The constant \( Z \) is the normalizing factor of the distribution.

The LRF model discriminatively estimates the marginal distribution over each pixel’s label. Essentially, a Gaussian CRF is used to estimate the response image \( r^* \) which is passed in a pixel-wise fashion through the logistic function \( \sigma(\cdot) \). Formally, the likelihood of pixel \( i \) taking the label +1 (denoted as \( l_i \)) is expressed as

\[
l_i = \sigma(r^*_i)
\]

(5)

where \( C(r|o) \) is a quadratic cost function that captures the same types of propagation behavior desired from the CRF model. A similar model was also proposed in [7]. We then introduce how to formulate \( C(r|o) \) below.

B. Learning LRF to Recognize Shadows

The cost function \( C(r|o) \) is based upon interactions between the expected response \( r \) and the current observations \( o \). It expresses the relationship between responses in a neighborhood jointly to recognize shadows. We define \( C(r|o) \) as

\[
C(r|o) = \sum_i w(o;\theta_1)(r_i - 10)^2 + w(o;\theta_2)(r_i + 10)^2 + \sum_{<i,j>} (r_i - r_j)^2.
\]

(6)

Each term \( r_i \) refers to the entry pixel \( i \) in the response image \( r \). These two terms pull each pixel to either -10 or +10 in the response image \( r^* \). While the response image should technically vary between \(-\infty\) to \(+\infty\), ranging the values between -10 and +10 is sufficient in practice. In particular, setting a specific pixel to +10 makes the probability of using the logistic function \( \sigma(\cdot) \) being equal to \( 1 - (4 \times 10^{-5}) \).

The weight \( w(o;\theta_k) \), \( k \in \{1, 2\} \) at pixel \( i \) is given by

\[
w(o;\theta_k) = \sum_{j \in N_f} \exp \left( \theta^T_k f_i \right),
\]

(7)

where \( \theta^T_k \) is the parameter vector associated with the \( j^{th} \) feature \( f_i \) for term \( k \), \( N_f \) is the total number of features.

By concatenating \( \theta_1; \theta_2; \theta_3 \) into a vector \( \theta \), we show that \( \theta \) can be found by minimizing the negative log-likelihood of the training dataset. We define the negative log-likelihood of model response over a training image by \( L(\theta) \) as

\[
L(\theta) = \sum_i \log \left( 1 + \exp(-t_i r_i^*) \right) + \lambda \sum_{j \in N_0} \theta^2_j,
\]

(8)

where \( t_i \) is the ground-truth probability of each pixel labeled to shadows, and the second term is a quadratic regularization term to avoid overfitting. \( \lambda \) is manually set to \( 10^{-4} \). We use a standard gradient descent method to iteratively update the parameters \( \theta \) which are all initialized
at zero. Note that $L(\theta)$ depends on $\theta$ via $r^*$. The regularizer penalizes the model parameters uniformly, corresponding to imposing a uniform variance onto $\theta$. This motivates a normalization of each type of the feature into $[-1, 1]$.

We show the details of the gradient computation of Equation (8) and the computation of $r^*$ using a matrix representation in the following subsection.

### C. Gradient Computation in Matrix Formulation

We rewrite the cost function $C(r; o)$ in a matrix representation by

$$C(r; o) = (Ar' - b)^T W(\theta, \theta)(Ar' - b), \quad (9)$$

where $A$ is a set of matrices $A_1...A_{N_\theta}$ that perform the same linear operations as convolving an image $r$ with a filter $f$. For the first two terms in $C(r; o)$, we derive two filters $f_1, f_2$ by setting only one of the pixel to 1 in a $3 \times 3$ window while others are remaining zero. For the last term, we loop through all possible combinations of $[-1, 1]$ pair in the $3 \times 3$ window, and choose each of them as a convolutional filter. Finally, $A$ is pre-computed by convolving the filters to a virtual image with identical image size to the images from the dataset. If the total number of pixel in the image $k$ is $n$, then $A_k$ is a $n \times n$ matrix.

$r'$ is a vector created by unwrapping the response image $r$, and performing $A_1 r'$ is identical to performing $f_1 \otimes r$. We concatenate $b$ using $-10$ and $+10$ for the first two terms in $C(r; o)$, and simply set $b = 0$ for the third term.

If the number of unknowns in Equation (9) is $n$, the number of constraints is $m = N_\theta \times n$, where $N_\theta$ is the total number of filters. We can see it is an over-determined problem. In addition, Equation (9) is a quadratic function, therefore its minimal can be computed using pseudo-inverse

$$r^* = (A^T A)^{-1} A^T W(\theta, \theta)b. \quad (10)$$

Now, we can derive the computation of the gradient for the loss function $L(\theta)$ with respect to $\theta$ as:

$$\frac{\partial L(\theta)}{\partial \theta} = \frac{\partial L(\theta)}{\partial r^*} = \frac{\partial L(\theta)}{\partial \theta} - \frac{\exp(\frac{-t \cdot r^*}{1+\exp(\frac{-t \cdot r^*}{\lambda})})}{1+\exp(\frac{-t \cdot r^*}{\lambda})} + 2\lambda \sum_{j \in N_\theta} \theta$$

$$\frac{\partial r^*}{\partial \theta} = (A^T A)^{-1} A^T \frac{\partial W(\theta, \theta)}{\partial \theta} b$$

### IV. PRACTICAL ISSUES

In this section, we introduce two practical issues in our model. In Feature Boosting, we show that by integrating Boosted Decision Tree instead of the raw features, the recognition accuracy of LRF can be further improved. In Parallel Training, we introduce a low cost acceleration method (without using of cluster servers) to update the gradient.

---

3It is encouraged to refer to the `filt2mat.m` file in the online code for details on how to construct $A_k$.
sample. In this paper, we handle the efficiency challenge from an architecture level, i.e., parallel training without special requirements of a cluster.

For each pixel, our LRF model requires 1,290 or even more features to be used to in classification. In the boosting step, there are 40 outputs for each pixel. Together with them, a horizontal and vertical derivative, and an one bias feature are added as new features in the LRF model, which gives us totally 43 features. As in Section III-C, in pixel-level, we include a $3 \times 3$ neighboring window to construct the filters. In this $3 \times 3$ window, for the data term, we also concatenate its 8 neighbors’ feature for both negative (-10) and positive (+10) cases. For the smoothness term, the number of filters ([1-1,1] pair) is 12. The total number of filters used is $9 + 9 + 12 = 30$. By timing 43 features, we have $43 \times 30 = 1290$ features in total. To train 125 images, it requires around 9G of memory without considering additional matrix operations.

Based upon the above analysis, it is obvious that gradient computation of $\frac{\partial L(\theta)}{\partial \theta}$ requires lots of matrix computation, which is the most time consuming task to update the gradient of parameters $\theta$. Given 125 training images, we update the gradient after summing the gradients over all the training images. This is different from online gradient descent, such as stochastic approaches, where the gradient is updated for each training image. The summation provides an elegant way to distribute the gradient computation task into multiple processors by assigning a subset of training images for each processor.

We used MatlabMPI [15] for model updating, which is a set of Matlab scripts that implement a subset of Message Passing Interface (MPI) and allow Matlab programs to be run on multiple processors. One advantage of MatlabMPI is that it does not require a cluster to execute the parallel commands. We configured three individual PCs including 10 processors and allow them to share 20G of memory. It takes around 10 hours to learn the LRF parameters.

V. EVALUATION ON RECOGNITION RESULTS

To evaluate how well the classifiers can locate the shadows, we select a subset of the images (126) including high quality (can be easily labeled by people) shadows as the training dataset. We predict the shadow label at every pixel for another 124 images with ground truth shadow labels.

The pixels that identified as shadows are then compared with the shadow masks associated with each image. True Positives $TP$ are measured as the number of pixels inside the mask. False Positives $FP$ are measured as the number of pixels outside the mask. False Negatives $FN$ are measured as the number of pixels falsely located outside the mask.

Our results are divided into two main groups. The first group includes three comparisons using monochromatic based features:

- different types of features
- different classification models
- different levels of over-segmentations

In the second group, we compared the performance between monochromatic and chromatic based features.

For all the comparisons, we used the accuracy computed by $\frac{TP}{TP+FP+FN}$, where $TP$ is the true positive, $FP$ is the false positive and $FN$ is false negative. Dropping the term of the true negative can help us to understand the essential performance difference between classifiers focused on the shadows.

Overall, our results show that features expressing illumination, textural and odd derivative characteristics can successfully identify shadows. Our results show BDT integrated with LRF using two combined levels of segments achieves highest recognition rate at 38.1, which is 6.9% higher than LRF alone; 3.7% higher than Support Vector Machine (SVM) and 2.2% higher than BDT. Our results also show the chromatic features are superior than the monochromatic feature, and their combination contributes a 1.8% accuracy improvement.

It is also interesting to see that several combination of the features, such as all variant features with edge and all variant features wit Entropy and Edge, achieve best accuracy in our dataset. We would like to point out that although our dataset includes variance illumination and places where shadow is casted, but they are not complete and perfect. This leads to training results biased towards some features such as the edge and the entropy. With more training data available, we are expected to see results from combining all variant features and invariants outperform all the other feature combinations.

A. Quantitative Results from Monochromatic Features

Comparisons between different types of features In this experiment, we test all the possible combination of variant and invariant features. For variant features, we add each feature at a time. Given all the variant features, we add each invariant feature at a time. The combination gives us totally 30 types of features.

For brevity, we train the classifiers at segment level with a number of 500, and using both the per-pixel features and the segment features. Per-pixel features is the feature value computed using the method introduced in II-B for each pixel. To generate the segment features, we histogram the per-pixel features inside a segment into 10 bins, and augmenting them by its mean and standard deviation. We compare the results in Figure V-A and Table I.

Comparisons between different classification models In this comparison, we compared results with Support Vector Machine (SVM) using radial basis kernel function, Boosted Decision Tree (BDT), Logistic Random Field (LRF) and a combined approach (BDT+l-LRF). We obtained around 30,000 training samples (segments) for SVM and BDT, and we trained the classifiers using all the features.

4Note that in our previous work [13], we report the accuracy using $\frac{TP}{TP+TP+FN+FP}$. We found it is an unfair comparison because of the number of nonshadow pixels is several times the number of shadow pixels
with their histogram values. We set the number of nodes in the decision tree to 40 and the initial gain step in the LRF model is set to 0.01. It takes around 15 minutes to train the BDT and around 10 hours to train the LRF in 500 iterations. We set the iteration number to 500 because we found the loss in Equation 8 decreases very slow afterwards.

We display their PR curves in Figure 8 (g). Our results show that LRF alone performs worst with only 26.9% accuracy. We believe this is because linear classifier trained from LRF performs poorly on complex conditional probabilities, such as distinguishing shadows from near black objects. We can also see SVM using single classification method can only achieve 30.1% accuracy. BDT with over-segmentation performs well with 31.6% accuracy. By enforcing the local consistency, BDT+LRF achieves highest accuracy at 33.8%.

**Comparisons between different levels**

In this comparison, we did experiments to train the BDT using different number of over-segments in an image. We tested the number of 200, 500 and their combined experiments. In the combined test, totally 83 features are used in the LRF, and the number of parameters reaches to 2490. It takes around 24 hours to train the classifier using our parallel version.

From the numerical results, we can see by combining two levels’ over-segmentations, the accuracy reaches 38.1% which is the highest accuracy among all the experiments. We believe the single level approaches perform worse than the combined approach because over-segmentation is not perfect in each level. By incorporating multiple levels of segment information, our approach achieves best results but sacrifices in computing recourses.

**B. Experiments with color-based features**

In this section, we compare results from classifiers using monochromatic and chromatic based features, respectively. While preparing the monochromatic dataset, we keep a local copy of a color version which is also available from the project website. Our motivation of experimenting with color features is aimed to answer two questions:

1) How well the chromatic classifier is compared with the monochromatic classifier?

2) Does their combination perform better than either chromatic or monochromatic classifier alone?

The chromatic-based features we selected are from an illumination invariant model proposed by Finlayson *et al.* [9] for linear images. Essentially, [9] observed that illumination changes in a specified 2D space tend to fall on parallel *straight lines* for a given camera. Their results show that this 2D space is a 2D band-ratio chromaticity color space, $\log\left(\frac{G}{R} \right), \log\left(\frac{B}{R} \right)$. The *straight lines* are observed as the invariant directions while lighting is changed. The illumination invariant space is therefore defined as an orthogonal direction to all the parallel lines. A gray scale shadow free image can be obtained by projecting the 2D band-ratio points into the orthogonal direction.

In our experiment, we choose three types of log chromatic features:

<table>
<thead>
<tr>
<th>TABLE II</th>
<th>Numerical Comparison of different methods</th>
</tr>
</thead>
<tbody>
<tr>
<td>Models</td>
<td>Method</td>
</tr>
<tr>
<td>Liner SVM</td>
<td>Boosted Decision Tree</td>
</tr>
<tr>
<td>Levels</td>
<td>200 Segments</td>
</tr>
<tr>
<td>Color</td>
<td>Monochromatic</td>
</tr>
</tbody>
</table>

* log values for R,G,B channels.
* log values for band-ratio, $\log\left(\frac{G}{R} \right), \log\left(\frac{B}{R} \right)$.
* A number of sampled directions that generates the gray scale shadow free image.

The first two types of the features are shadow variant features. The third type of the features includes both shadow invariant and shadow variant features. The shadow invariant features are the *true* directions (or almost *true*) orthogonal to the parallel straight lines denoting illumination invariant space. We are expected to see consistent properties across the shadow boundary from these features. The shadow variant features are obtained from the *false* directions that are not actually orthogonal to the parallel straight lines. We are expected to see inconsistent properties across the shadow boundary from these features.

Because the camera response from different digital cameras are not same, the orthogonal direction for each camera
We plot their ROC curves in Figure 7, the features which gives us 2%. In addition, we also trained a system by concatenating all features. In the chromatic test, we sampled 36 directions (the angles are from 0 – 180 degree, each interval is 5 degree), and trained totally 2 * 39 + 12 * 39 = 546 features. In the monochromatic test, we trained 2*8+12*8 = 112 features. In addition, we also trained a system by concatenating all the features which gives us 2 * 47 + 12 * 47 = 658 features. We plot their ROC curves in Figure 8 (h).

We can see from the result that chromatic classifier performs better than monochromatic classifier, with an improvement of recognition accuracy around 4%. By combining them, we obtain another 3.8% accuracy improvement.

Note that the accuracy from this test is higher than that from previous two tests. This is because the shadows from our captured dataset (using a DLSR camera) is more easily observed. And this leads to a better recognition accuracy both in training and testing.

C. Qualitative Results of Monochromatic Classifier

We present part of the qualitative results from our monochromatic tests in Figure 8. Note that to avoid presenting the duplicated visualized results, we show complimentary results to our previous work [13] in this Figure. Similar results presented in our previous work are also obtained in the new dataset. We set the threshold to 0.5 when obtaining the binary estimates. For brevity, we only present the results using combined levels of segments for BDT+LRF.

The first case shows a simple scene with shadows casted on the grass. The probability map from BDT is unsurprisingly inconsistent because BDT treats each segment individually. The result from LRF model alone is neither perfect as we can see some misclassified pixels on the grass and near object boundaries.

The second and third cases show two examples of objects with dark albedo. We can see the probabilities of such objects being shadows from LRF alone are falsely as high as real shadows. Results from BDT are also not acceptable as thresholding the correct shadows from all the probabilities is very difficult.

The fourth and fifth cases show two complex examples. We can see LRF model can not distinguish shadows from trees. BDT does good job on separating shadows from trees, but the labeling is inconsistent.

From all these test examples, we can see our combined approach performs better, in terms of both accuracy and consistency, than either of BDT or LRF model alone.

Nevertheless, we found that water is the most difficult object to be misclassified as shadows. We show two examples from very challenging case in the last row of Figure 8.
Fig. 8. Qualitative and quantitative results. The visualized results ((a) to (f)) are compared from the LRF model, Linear Support Vector Machine (SVM), Boosted Decision Tree (BDT) and combined approach (BDT+LRF). The learned classifier are all using monochromatic based features. The last row shows the result from a very challenging case. The water is flowing from cement. The shadows are the darkest part cast by the cement in the water. Quantitative comparisons are from different types of features and different types of learning models. In (g), we compare results from different learning models using the variant+invariant+bin monochromatic features. In (h), we compare results from chromatic-based and monochromatic-based features. To be more broadly present our work, the results presented in the paper is complimentary to our previous work [13].
In the test, shadows in this scene are casted on water, which is a transparent media and also appears black. In the second overhead scene, all the methods introduced in the paper tend to falsely label the water as shadows. We conclude that the optical properties of classify shadows from the water are not captured well using our proposed features. We believe this is a field for further research.

VI. SHADOW REMOVAL IN MONOCHROMATIC NATURAL IMAGES

In this section, we introduce our approach to remove the recognized shadow in monochromatic natural images. Our method is based on the observation that illumination, caused from shadow, across shadow boundary is smoothly changed. We find that in the gradient domain, this leads to a Gaussian approximation to cancel the shadows. We also find that a Gaussian distributed assumption of the underlining texture across shadow boundary is helpful to obtain high quality shadow-free images.

A. Linear Shadow Model

We assume that all the images for shadow removal are taken in the raw format using DSLR cameras. We introduce two assumptions that make shadows separable from underlining surfaces in a linear model.

We make our first assumption that the input shadow image, \( I(x, y) \), can be expressed as the product of a reflectance image, \( R(x, y) \), and the shadow image, \( S(x, y) \)

\[
I(x, y) = R(x, y) \times S(x, y)
\]

The reflectance of a scene is the material of the surfaces in the scene, and the shadow of a scene refer to the illumination that is occluded from objects in the scene.

Considering the images in the log domain, the derivatives of the input image are the sum of the derivatives of the reflectance and the illumination. We find that significant shadow boundaries and reflectance edges is unlikely occur at the same point. Although this assumption is not consistent to occlusion shadows, where edges indicate both a shadow and a material edge [8], we find such cases are actually very few in natural scenes.

Consequently, we make the second assumption that every image derivative is either caused by shadow or reflectance. A similar assumption of shading and reflectance is reported in [32]. This reduces the problem of assuming that the image’s \( x \) and \( y \) derivative changes on the shadow boundary are mainly from illumination changes, in our case, the shadows.

B. Parametric Illumination Change Model

We parameterize the linear shadow model for each pixel across the shadow boundary by sampling horizontal and vertical line segments to model the illumination change \( \Delta I(x, y) \). Line segments provides easily parameterizable illumination profile, which is also used in [18]. Figure 9 shows an example of the sampled line segments.

The difference between our approach and [18] is that we model \( \Delta I(x, y) \) in the log derivative domain instead of the RGB domain as we observed that a simple Gaussian model, with fewer parameters, can well describe the shadow canceling process by favoring the smoothness inside a neighborhood. Therefore, our approach actually models \( \nabla(\Delta I(x, y)) \), the derivative of the illumination change model.

To sample the line segments, we first use Canny edge detection method to locate the shadow edge from a binary version of the shadow probability map obtained by the recognition process. And then, we dilate \( p \) pixels inward and outward given the Canny edge map and obtain boundary that is sufficiently across the shadow boundary. The dilation parameter \( p \) is chosen based on three factors: the position of the occlusion object, the direction of the illumination and the image size. In our experiments, we set \( p = 10 \) for all the tests.

C. Monochromatic Shadow Removal

To cancel the monochromatic shadow effect in the gradient domain, we first fit a local Gaussian function, for each line segment, to simulate illumination changes. To further preserve the underlining texture across the shadow boundary, we formulate a Markov Random Field (MRF) model to globally optimize the parameters of the Gaussian function. Although other advanced techniques, such as Belief Propagation, Graph Cut and Variational Model can also be used to optimize the parameters, we find standard gradient descent works also well.

**Shadow Canceling**

For each line segment, the Gaussian function, \( f \), is fitted by the gradients’ difference to the mean

\[
f = A \ast \exp\left(\frac{- (x - u)^2}{2 \ast \sigma^2}\right)
\]

where \( u \) is the center of the derivative values computed as \( u = \frac{1}{n} \sum_i (y_i \ast x_i) \). \( y_i \) and \( x_i \) forms a local coordinate system with \( x_i \) denotes the line segment’s horizontal coordinates and \( y_i \) denotes its vertical coordinates. \( n \) is the total number of the pixels in the line segment. \( \sigma \) is the standard deviation of the distribution and \( A \) is a constant scaling the whole distribution. We initialize \( A = 0.1 \) and \( \sigma = 0.1 \). We show two examples of the fitted Gaussian in Figure

---

5We also experiment other techniques, such as setting \( p \) automatically by doing a matting approach on the shadow boundary. But we found the automatically results are comparable to a tuned parameter.
VI-C. Note that the signs of these two Gaussian peaks are different because the sign of the gradients across shadow boundary from shadows to non-shadows are different. They are either positive or negative.

Texture Preserving
To preserve the texture across the shadow boundary after canceling the shadows, we assume the texture distribution is a Gaussian distribution in the gradient domain. We formulate a Markov Random Field (MRF) to optimize the parameters $A, u, \sigma$ of the Gaussian function $f$ by forcing the texture distribution as a Gaussian. In the MRF, each line segment is viewed as a node, and its two nearest line segments are edges. The energy function is formulated as

$$E = \sum_i \phi(y, f) + \lambda \sum_{<i,j>} \varphi(A, u, \sigma)$$

where $y$ is the observed derivative values. The data term $\phi$ is designed as a function to measure the texture distribution of the canceled derivatives $y - f$.

For natural images, this texture distribution can be well expressed as a Gaussian distribution with mean $u_0$ and standard deviation $\sigma_0$.

$$\phi(y, f) = \exp\left(\frac{-(y - f - u_0)^2}{2 + \sigma_0^2}\right)$$

In our approach, $u_0, \sigma_0$ are calculated using additional 6 pixels by extending the line segment on both sides.

The smoothness term $\varphi$ is simply a summation of $L_2$ function that penalties consistency within neighboring area for each parameter $A, u, \sigma$. The weight from the smoothness term $\lambda$ is set to 0.1 in all the tests.

We minimize the Equation (13) by standard gradient descent method. Although this method does not guarantee global optimum, we find high quality results can be obtained within 500 iterations of gradient updates.

In all our tests, to recover the shadow-free image, we re-integrate the canceled derivatives $y - f$ (note that we do horizontal and vertical derivative canceling separately) by iteratively solving a Laplace equation given the horizontal and vertical derivatives for 5000 times.

Figure 11 shows a comparison of the recovered shadow-free image using local Gaussian fitting method and global texture preserving refinement. We can see the global approach can generate more natural and visual pleasing result by preserving the textures correctly.

We summarize all the steps in Algorithm VI-C.

**Algorithm 1 Gaussian-based Shadow Removal**

1. Thresholding the shadow probability map returned from the recombination process. The threshold is set to 0.1.
2. Detecting the edges from the binary map.
3. Dilate and erode the edge map to sufficiently include shadow boundary. The number of dilation and erosion pixels is set to 10.
4. For each pixel on the edge map, we sample one horizontal line segment and one vertical line segment from the boundary map.
5. In the log domain, for each line segment, we fit a Gaussian function given their gradients.
6. We formulate a MRF model to globally refine the underlying texture by forcing its gradient as a Gaussian distribution and penalty the consistency in a neighboring area.
7. Finally, we recover the shadow-free image by solving a Laplace equation given the shadow canceled derivatives.

D. Results
We show part of the shadow removal results in Figure 14. The first four rows show results from the images taken by a Canon Digital XT1 camera, with a lens size of 50mm, and the last two rows show results from the overhead image library. We can see, overall, the local method works well while the global method performs better than local method. We also see saturation artifacts exist in local results (in 1st and 2nd rows). We believe the artifacts near the right corner of 1st row is because of the imperfection line sampling due to the acute angle.

It is known that typical digital photos have more noise in shadow (dark) areas. This artifact can be clearly seen from two overhead images showed in Figure 14. By successfully removing the shadows, we show that the recovered shadow-free images are more semantically useful – missing context such as the cars is enhanced.
A number of recent shadow removal algorithms have been proposed to preserve the textures using color [37], [30], gradient [22], [18], second order smoothness [2] or intrinsic image [9]. These approaches work well and descent results are demonstrated.

Losing chromatic information makes shadow removal more challenging because approaches rely on colors to find affinity between shadows and non-shadows, or seeking an illumination invariant space where intrinsic image can be recovered can not be used. However, monochromatic shadows do have some special properties that are useful for shadow removal. One observation we find is that gradient changes, in some channels, caused by shadows from color images is more sever compared with that from monochromatic images, a linear combination of all the channels. We show a comparison of the horizontal $x$ derivatives of a textured scene from its color and monochromatic version with ground truth data.

To obtain the ground truth data, we fix the camera, and quickly capture two scenes with/without shadows using a remote controller. The shadows casted on the grass are from a plane board occluding the sun light. We choose the derivatives inside shadows and histogram them into 50 bins. The distance between histograms from shadows and ground truth are measured using Earth Mover’s Distance [26]. We show $x$’s derivative results in Figure 13. We find results from $y$’s derivatives are very similar.

We can see derivative distortion inside shadows from monochromatic images are actually smaller compared to that from a color version. We believe this makes our shadow removal algorithm focus on shadow boundary while leaving the interior unconcerned a success.

One interesting result is shown in the first row in Figure 12 where shadow and reflectance edge are very complex. Although this may violate our assumption that significant shadow boundary and reflectance edges are unlikely happen at the same point, the recovered result is still acceptable.

Although our shadow removal approach works well for most of the scene, it still relies on the shadow recognition results, especially the correctness of the shadow boundary. We show a failed example in the second row in Figure 12.

E. Discussion

Most of earlier shadow removal algorithms zero the derivatives of pixels and add a constant scale to the intensities enclosed within the shadow boundary [10]. These approaches produce good results when the shadow boundary is sharp, but often it falsely nullifies textures on curved and textured surfaces for complex natural scenes.

A number of recent shadow removal algorithms have been proposed to preserve the textures using color [37], [30], gradient [22], [18], second order smoothness [2] or intrinsic image [9]. These approaches work well and descent results are demonstrated.

Losing chromatic information makes shadow removal more challenging because approaches rely on colors to find affinity between shadows and non-shadows, or seeking an illumination invariant space where intrinsic image can be recovered can not be used. However, monochromatic shadows do have some special properties that are useful for shadow removal. One observation we find is that gradient

changes, in some channels, caused by shadows from color images is more sever compared with that from monochromatic images, a linear combination of all the channels. We show a comparison of the horizontal $x$ derivatives of a textured scene from its color and monochromatic version with ground truth data.

To obtain the ground truth data, we fix the camera, and quickly capture two scenes with/without shadows using a remote controller. The shadows casted on the grass are from a plane board occluding the sun light. We choose the derivatives inside shadows and histogram them into 50 bins. The distance between histograms from shadows and ground truth are measured using Earth Mover’s Distance [26]. We show $x$’s derivative results in Figure 13. We find results from $y$’s derivatives are very similar.

We can see derivative distortion inside shadows from monochromatic images are actually smaller compared to that from a color version. We believe this makes our shadow removal algorithm focus on shadow boundary while leaving the interior unconcerned a success.

VII. Conclusions

We have proposed a system for recognizing and removal shadows from monochromatic natural images. Using the cues presented in this paper, our method can successfully identify shadows in real-world data where color is unavailable. We found that while single-pixel classification strategies work well, a Boosted Decision Tree integrated into a CRF-based model achieves the best results. We also found that a simple assumption of Gaussian illumination changes in the gradient domain across the shadow boundary can help to remove the shadows.

Acknowledgement

The authors thank Edward H. Edelson, Graham Finlayson, Dani Lischinski and Srinivasa Narasimhan for helpful discussions. M.F.T. was supported by a grant from the NGA (HM1582-08-1-0021).

REFERENCES

are taken in outdoor environment under the sun.

Fig. 13. Comparison of texture distortion from color shadows and monochromatic shadows. The horizontal values are the horizontal gradients of the log channels. The vertical values are the normalized count in each bins. The texture distortion from the blue channel is more severe because the images are taken in outdoor environment under the sun.

Fig. 14. Results of shadow removal from monochromatic images. We can see while the local method works well, a global method achieves better results. This can be observed by comparing results from first two rows. It is also interesting to see that more information can be observed by removing the shadows in the last two images from overhead database. See that after shadow removal the