Leave-one-out Kernel Optimization for Shadow Detection and Removal

Tomás F. Yago Vicente, Minh Hoai, and Dimitris Samaras

Abstract—The objective of this work is to detect shadows in images. We pose this as the problem of labeling image regions, where each region corresponds to a group of superpixels. To predict the label of each region, we train a kernel Least-Squares Support Vector Machine(LSSVM) for separating shadow and non-shadow regions. The parameters of the kernel and the classifier are jointly learned to minimize the leave-one-out cross validation error. Optimizing the leave-one-out cross validation error is typically difficult, but it can be done efficiently in our framework. Experiments on two challenging shadow datasets, UCF and UIUC, show that our region classifier outperforms more complex methods. We further enhance the performance of the region classifier by embedding it in a Markov Random Field(MRF) framework and adding pairwise contextual cues. This leads to a method that outperforms the state-of-the-art for shadow detection. In addition we propose a new method for shadow removal based on region relighting. For each shadow region we use a trained classifier to identify a neighboring lit region of the same material. Given a pair of lit-shadow regions we perform a region relighting transformation based on histogram matching of luminance values between the shadow region and the lit region. Once a shadow is detected, we demonstrate that our shadow removal approach produces results that outperform the state of the art by evaluating our method using a publicly available benchmark dataset.

Index Terms—Shadow detection, shadow removal, kernel optimization

1 Introduction

Shadow removal is desirable in many situations. Shadows are common in natural scenes, and they are known to complicate many computer vision tasks such as image segmentation and object detection. Therefore the ability to generate shadow-free images would benefit many computer vision algorithms. Furthermore, for aesthetic reasons, shadow removal can benefit image editing and computational photography algorithms.

Automatic shadow detection and removal from single images, however, are very challenging. A shadow is cast whenever an object occludes an illuminant of the scene; it is the outcome of complex interactions between the geometry, illumination, and reflectance present in the scene. Identifying shadows is therefore difficult because of the limited information about the scene's properties.

There has been a number of approaches for shadow detection. Purely physics-based methods such as the illumination invariant approaches of [6, 7] only work on high quality images. For consumer photographs and web quality images, statistical learning-based approaches [9, 10, 21, 41, 43] have proven more successful, but the final results are still far from perfect.

In this paper, we propose a novel algorithm for shadow detection. We pose shadow detection as an image labeling problem, similar to some statistical learning-based methods [9, 11, 41]. Given an image, we first divide it into multiple regions, where each region is a group of superpixels, as illustrated in Fig. 1. We use a region classifier to estimate the shadow probability of each region based on its appearance features. Subsequently, we improve shadow detection by considering contextual cues between neighboring regions. The contextual cues are incorporated in our framework as pairwise potentials in a Markov Random Field (MRF).We solve the optimization with QPBO [20, 27] to produce the final shadow labels.

One particular novelty of our approach is the framework for training a strong shadow region classifier that can effectively integrate multiple types of local cues. In particular, we jointly learn a classifier and a discriminative kernel that combines chromatic, intensity, and texture properties for shadow detection. Unlike existing approaches for shadow detection, we propose to use Least Square Support Vector Machine (LSSVM). LSSVM has been shown to perform equally well as SVM in many classification benchmarks [33]. LSSVM has a closed-form solution, which is a computational advantage over SVM. Furthermore, once the solution of LSSVM has been computed, the solution for a reduced training set obtained by removing any of the training data points can be found efficiently. This enables using the same training data for learning both the classifier and the kernel parameters. As will be shown, optimizing the kernel parameters is crucial for improving the discriminative power of the shadow classifier, and this can be done efficiently using our framework. Moreover, our method can be implemented in GPU, further reducing the computational cost.

As will be shown, our shadow region classifier outperforms more complex methods even without using contextual cues. Nonetheless, context is important for shadow detection as it is often difficulty to discern shadows based on the local appearance of individual regions, even for human observers. We therefore enhance our method by incorporating contextual cues as pairwise potentials in an MRF framework. We introduce two types of potentials: affinity and

• *Tomás F. Yago Vicente, Minh Hoai and Dimitris Samaras are with the Department of Computer Science, Stony Brook University, Stony Brook, NY 11794. E-mail: tyagovicente@cs.stonybrook.edu*

Manuscript received XXX, YYYY; revised

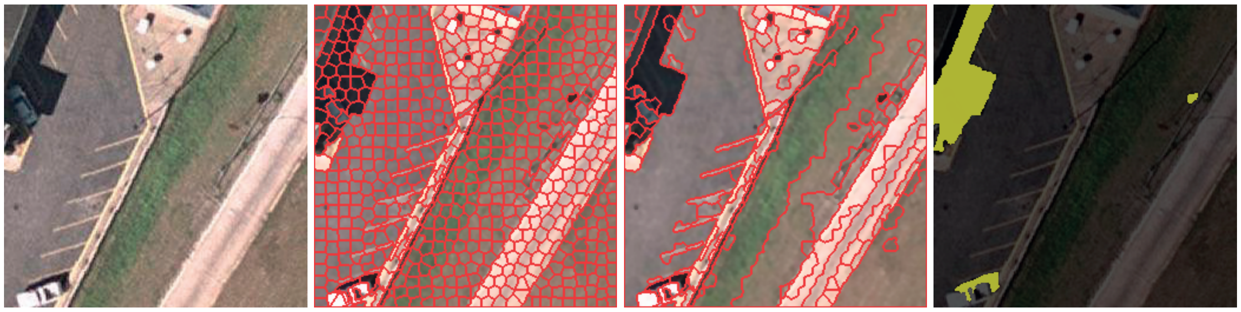

Fig. 1: Shadow detection as a region labeling problem. From left to right: input image, initial superpixels, segmented regions, and shadow predictions

disparity. The affinity potentials encourage similar adjacent regions to have the same label, while the disparity potentials prefer different labels for shadow-nonshadow region pairs (using the output of a classifier for region pairs).

We perform experiments on the challenging UCF [9] and UIUC [43] shadow datasets and observe that the proposed method outperforms the current state-of-the-art method [11]. On the UIUC dataset, our method reduces the false negative rate of [11] by 35.3% while maintaining a similar false positive rate. On the UCF dataset, our method reduces the false negative rate and false positive rate by 9% and 13.5%, respectively.

We complete our pipeline with a final shadow removal step. We propose a method to generate shadow free images using the detected shadow masks. We perform experiments on the publicly available shadow removal benchmark dataset [9]. Our method reduces the root mean square error (RMSE) of Guo *et al*. [10] by 18%. In shadow areas the error reduction is 30%.

2 Previous Work

2.1 Shadow detection in images

Shadow detection in images is a well studied problem. Earlier methods such as [6, 7] detect shadows by comparing the gradients of an image and its illumination invariant representations. These methods show impressive results in high quality images, but their performance degrades significantly with consumer photographs or web quality pictures [21]. More recent methods use image datasets with annotated shadow masks to learn the appearance of shadows in images. These methods follow two main approaches: detecting shadow boundaries or detecting shadow regions. Lalonde *et al*. [21] focus on shadow boundaries on the ground. They train a shadow boundary classifier based on color and texture features and combine it with scene layout cues from [14] using a Conditional Random Field(CRF) to encourage boundary continuity. Huang *et al*. [15] use a set of physically inspired features to train a shadow boundary pixel classifier using an SVM. They join pixels confidently predicted as shadow boundaries with weakly predicted adjacent pixels in a Canny-like manner. Shadow boundary detection methods [15, 21] achieve good results, but struggle to segment closed shadow contours.

For detecting shadow regions, Zhu *et al*. [43] propose a set of shadow variant and shadow invariant features in monochromatic images to learn a shadow region classifier, and refine the results with a CRF. Guo *et al*. [9, 10] train two pairwise classifiers to find pairs of regions in an image that share the same material and are viewed under the same illumination conditions (both in shadow or both in not shadow), and same material but illuminated differently (only one region in shadow). They minimize an energy functional that combines the predictions of a single region classifier and the positive predictions of their pairwise classifiers. However, their single region classifier is not particularly accurate, especially for shadow regions. They use an SVM with a $\mathcal{X}^2$ kernel that has limited discriminative power.

In our previous work [41], we propose a multi-kernel model to learn a shadow region SVM classifier. This multi-kernel model is a summation of base kernels, one for each type of local feature. The main limitation of this model is the assumption of equal importance for all features. The weights and the scaling factors of base kernels are not learned. Furthermore, this approach is computationally expensive.

Most recently, Khan *et al*. [11] propose a deep learning approach to learn features for shadow detection. They train two Convolutional Neural Networks(CNN), one for detecting shadow regions and the other for shadow boundaries. They use a CRF to label pixels as shadow/non-shadow where the predictions of the two neural nets are combined into a unary potential, and the pairwise potential is an Ising prior where the pairwise penalty is determined by the similarity in intensities between adjacent pixels.

2.2 Shadow removal in images

Early works in shadow removal focus on eliminating the effects of shadows in the image gradients and then integrate the modified gradient field to obtain a shadow free image. For instance, Finlayson *et al*. [6, 7] remove shadows by zeroing shadow edges in the gradient domain and then integrating it to obtain a shadow free image. They achieve good results with high quality images, however the integration often introduces changes in color balance, global smoothness and loss of textural properties, specially in the penumbra or boundary areas. Liu *et al*. [25] propose an integration based algorithm that aims to improve the loss of texture that commonly accompanies integration methods. They construct a gradient field for the penumbra area to cancel out the effects of the illumination change. Their results improve in terms of texture consistency but they cannot handle non uniform shadows or complex textures. Integration based methods are highly sensitive to accurate segmentation of the shadow edges.

Shor *et al*. [32] present an affine shadow formation model with a multi scale scheme to remove shadows. They require minimal user assistance to identify shadow and lit areas of the same surface material. Based on those pairings, they obtain the constant parameters of the shadow model. Due to the assumed constant coefficient their method has problems with non uniform shadows and sometimes rich textures. Xiao *et al*. [40] extend the affine model of Shor *et al*. [32] by allowing the attenuation factor to be adaptive. This accounts for reflectance variations on the shadow areas.

Wu *et al*. [39] perform shadow matting to remove shadows. They estimate shadow intensities based on intensity ratios in the umbra region and use a Bayesian framework to regularize the shadow scale factor in the shadow regions. The umbra regions of the shadows are assumed to be roughly uniform. Guo *et al*. [9] also remove shadows based on shadow matting, they generate a soft shadow mask from the ground truth and randomly sample patches from both sides of the shadow boundary to compute the illumination ratios. Guo *et al*. [9] extensively evaluate their results on a shadow dataset that is publicly available. It is the first work to present qualitative and especially quantitative evaluation results on a somewhat large dataset as opposed to a few selected images. Most recently, Khan *et al*. [19] use a Bayesian formulation to extract a shadow matte and remove the shadows. First, the umbra and penumbra regions, and the object shadow boundaries are estimated. Then, a rough shadow-less image is obtained using multiscale color transfer using Gaussian mixture models. Lastly, assuming shadows are cast by a single light source and uniform ambient light, a Bayesian formulation is optimized to solve for the final matte from the rough shadow-less image.

2.3 Least-Squares SVM Classifiers

We use Least-Squares Support Vector Machines (LSSVM) [34], also known as Kernel Ridge Regression [29]. LSSVM has a closed-form solution, which is computationally advantageous compared to SVMs. Furthermore, once the solution of LSSVM has been computed, the solution for a reduced training set obtained by removing any training data point can be found efficiently. This enables reusing training data for further calibration, e.g., [12, 13]. This section reviews LSSVM and the leave-one-out formula.

Given a training set of n data points $\{\mathbf{x}_i\}_{i=1}^{n}$* and associated labels $\{y_i | y_i \in \{1, -1\}\}_{i=1}^{n}$, LSSVM optimizes the following:

$$\underset{\mathbf{w},b}{\text{minimize}} \;\; \lambda||\mathbf{w}||^2 + \sum_{i=1}^{n}(\mathbf{w}^T\phi(\mathbf{x}_i) + b - y_i)^2. \tag{1}$$

Here $\phi(\mathbf{x}_i)$ is the feature mapping from the input space to a feature space. If the dimension of the feature space is high ($\gg n$) or even infinite, it is more efficient to obtain the solution for $(\mathbf{w}, b)$ via the representer theorem, which states that $\mathbf{w}$ can be expressed as a linear combination of training data in the feature space, i.e., $\mathbf{w} = \sum_{i=1}^{n}\alpha_i\phi(\mathbf{x}_i)$. Let $\mathbf{K}$ be the kernel matrix, $k_{ij} = \phi(\mathbf{x}_i)^T\phi(\mathbf{x}_j)$. Let $\mathbf{k}_i$ denote the i^{th} column of $\mathbf{K}$, Eq. (1) is equivalent to:

$$\underset{\boldsymbol{\alpha},b}{\text{minimize}} \;\; \lambda\boldsymbol{\alpha}^T\mathbf{K}\boldsymbol{\alpha} + \sum_{i=1}^{n}(\mathbf{k}_i^T\boldsymbol{\alpha} + b - y_i)^2. \tag{2}$$

Let $\overline{\boldsymbol{\alpha}} = [\boldsymbol{\alpha} \; b]^T, \mathbf{Z} = [\mathbf{K} \; \mathbf{1}_n]^T, \mathbf{R} = \begin{bmatrix} \lambda\mathbf{K} & \mathbf{0}_n \\ \mathbf{0}_n^T & 0 \end{bmatrix}$, Equation (2) is equivalent to:

$$\underset{\overline{\boldsymbol{\alpha}}}{\text{minimize}} \;\; \overline{\boldsymbol{\alpha}}^T(\mathbf{R} + \mathbf{Z}\mathbf{Z}^T)\overline{\boldsymbol{\alpha}} - 2\mathbf{y}^T\mathbf{Z}^T\overline{\boldsymbol{\alpha}} + \sum_{i=1}^{n}y_i^2. \tag{3}$$

This is an unconstrained convex optimization problem, so the optimum value is attained at zero-gradient. Taking the gradient with respect to $\overline{\boldsymbol{\alpha}}$, the optimal solution can be obtained by solving the linear equation: $(\mathbf{R}+\mathbf{Z}\mathbf{Z}^T)\overline{\boldsymbol{\alpha}} = \mathbf{Z}\mathbf{y}$. With $\mathbf{C} = \mathbf{R} + \mathbf{Z}\mathbf{Z}^T, \mathbf{d} = \mathbf{Z}\mathbf{y}$, the optimal solution is:

$$\overline{\boldsymbol{\alpha}} = \mathbf{C}^{-1}\mathbf{d}. \tag{4}$$

Suppose $\mathbf{x}_i$ is removed from the training set, and let $\mathbf{C}_{(i)}, \mathbf{d}_{(i)}, \overline{\boldsymbol{\alpha}}_{(i)}$ be the corresponding values for removing $\mathbf{x}_i$. We have $\overline{\boldsymbol{\alpha}}_{(i)} = \mathbf{C}_{(i)}^{-1}\mathbf{d}_{(i)}$. Note that, even though we remove $\mathbf{x}_i$ from the training data, we can still write $\mathbf{w}$ as linear combination of $\{\phi(\mathbf{x}_i)\}$ without excluding the term $\phi(\mathbf{x}_i)$ out. The only change is the removal of the term $(\mathbf{k}_i^T\boldsymbol{\alpha} + b - y_i)^2$ from the objective function. Thus we have $\mathbf{C}_{(i)} = \mathbf{C} - \mathbf{z}_i\mathbf{z}_i^T$ and $\mathbf{d}_{(i)} = \mathbf{d} - y_i\mathbf{z}_i$. Using the Sherman-Morrison formula [30, 31], we can quickly compute the inverse of $\mathbf{C}_{(i)}$:

$$\mathbf{C}_{(i)}^{-1} = (\mathbf{C} - \mathbf{z}_i\mathbf{z}_i^T)^{-1} = \mathbf{C}^{-1} + \frac{\mathbf{C}^{-1}\mathbf{z}_i\mathbf{z}_i^T\mathbf{C}^{-1}}{1 - \mathbf{z}_i^T\mathbf{C}^{-1}\mathbf{z}_i}. \tag{5}$$

With $\mathbf{M} = \mathbf{C}^{-1}\mathbf{Z}$ and $h_{ii} = \mathbf{z}_i^T\mathbf{m}_i$, following the derivation as above, we obtain the following formulas:

$$\text{The optimal solution : } \overline{\boldsymbol{\alpha}} = \mathbf{M}\mathbf{y}, \tag{6}$$

$$\text{Leave-one-out solution: } \overline{\boldsymbol{\alpha}}_{(i)} = \overline{\boldsymbol{\alpha}} + \frac{(\overline{\boldsymbol{\alpha}}^T\mathbf{z}_i - y_i)}{1 - h_{ii}}\mathbf{m}_i, \tag{7}$$

$$\text{Leave-one-out error} = \overline{\boldsymbol{\alpha}}_{(i)}^T\mathbf{z}_i - y_i = \frac{\overline{\boldsymbol{\alpha}}^T\mathbf{z}_i - y_i}{1 - h_{ii}}. \tag{8}$$

3 Shadow Region Classification

We pose shadow detection as a region classification problem. Given an image, we first segment it into regions using a two step process [41]: 1) apply SLIC [1] superpixel segmentation to oversegment the image and obtain an initial set of superpixels; 2) apply Mean-shift clustering [5] and merge superpixels in the same cluster into a large region. This segmentation process is illustrated in Fig. 1. Once the image has been segmented into regions, we use an LSSVM to predict the shadow probability of each region.

Because shadows are the outcome of the complex interaction between scene geometry and illumination sources, it is necessary to consider multiple feature types for shadow classification. In particular, we propose to base the classification decision on the chromatic, intensity, and texture properties of the region. However, it is difficult to combine

*. Bold uppercase letters denote matrices (e.g. $\mathbf{K}$), bold lowercase letters denote column vectors (e.g. $\mathbf{k}$). $\mathbf{k}_i$ represents the i^{th} column of the matrix $\mathbf{K}$. k_{ij} denotes the scalar in the row j^{th} and column i^{th} of the matrix $\mathbf{K}$ and the j^{th} element of the column vector $\mathbf{k}_i$. Non-bold letters represent scalar variables. $\mathbf{1}_n \in \Re^{n\times 1}$ is a column vector of ones, and $\mathbf{0}_n \in \Re^{n\times 1}$ is a column vector of zeros.

heterogeneous feature types and manually tune the importance of each type. We therefore consider this as a kernel learning problem where the kernel function has the form:

$$K(x, y) = \sum_{i=1}^{k} w_i \exp\left(-\frac{1}{\sigma_i} D_i(x, y)\right). \quad (9)$$

Here $K(x, y)$ denotes the kernel value between two regions x and y. The function $D_i(x, y)$ is the distance between x and y in some feature space (e.g., $\mathcal{X}^2$ distance between texton histograms) and it is predetermined. The function $\exp(-\frac{1}{\sigma_i} D_i(x, y))$ is called the extended Gaussian kernel [17, 36, 42], and the kernel K is the linear combination of extended Gaussian kernels. The parameters $\{w_i, \sigma_i\}$ are what needs to be learned. Additionally, we constrain the kernel weights to be non-negative and have unit sum, i.e., $\sum_{i=1}^{k} w_i = 1$.

We propose to jointly learn the kernel and the LSSVM classifier. Given a set of training regions and corresponding shadow indicator labels, our goal is to find a set of parameters $\{w_i, \sigma_i\}$ that yields the lowest leave-one-out balanced error rate. The balanced error rate is the average of false positive rate and false negative rate. For brevity, we refer to the balanced error rate simply as error rate or error. The leave-one-out error for a given kernel is defined and conceptually computed as follows. First, the leave-one-out confidence values are computed for all training examples. The leave-one-out confidence value for a particular training example is obtained by training a classifier on the remaining examples and evaluating on the left-out sample. The leave-one-out confidence values are then compared against the ground truth shadow annotation to compute the leave-one-out error rate. In general, estimating the leave-one-out error is computationally prohibitive because classifier training must be done many times, once per training example. However, as explained in Sec. 2.3, using LSSVM, the leave-one-out confidence values can be obtained efficiently without training the leave-one-out classifiers.

The leave-one-out error is a function of the kernel parameters. Even though calculating the value of this function for a particular kernel can be done efficiently, it is still unclear how to find a set of kernel parameters that yields the lowest leave-one-out error. Unfortunately, the leave-one-out error function is not convex. Even worse, this function is noncontinuous and piece-wise constant, and therefore a gradient-based optimization approach is unlikely to work well. To see this, recall that the set of possible error rates are discrete; the interval between two adjacent discrete values is inversely proportional to the number of training examples. This function is non-differentiable at many locations, and has zero gradients at the other locations.

To optimize the set of kernel parameters, we propose to use beam search with random steps. We first discretize the space of kernel parameters using a grid (details will be provided in Sec. 3.2). Starting from a random parameter vector, we perform a number of iterative updates. Each update involves the following steps:

1) Randomly choose one kernel parameter and assign a new random value. If necessary, re-normalize $\{w_i\}$ to have unit sum.
2) Train an LSSVM classifier and compute the leave-one-out error for the new set of parameters.
3) Update the parameter set if it yields lower leave-one-out error than the current best value.

In our experiments, we perform 500 iterations. If the leave-one-out error does not decrease after 25 consecutive iterations, we randomly assign new values to all parameters.

The method proposed here has advantages over some existing kernel learning approaches. One popular approach is multiple kernel learning, e.g., [2, 16, 22, 37]. Many multiple kernel learning methods, however, can only learn a linear combination of base kernels; they cannot be used to learn other parameters such as the scaling factor of a generalized Gaussian kernel. This problem can be circumvented by creating multiple kernel instances with different parameter settings. However, this explodes the number of base kernels, so the optimization typically requires a differentiable objective function [16, 37]. Furthermore, most existing approaches learn the kernel parameters to optimize an objective function defined on the surrogate loss of training data, not the held-out data. Thus the same training data is used for both classifier training and kernel learning. The double-use of training data reduces the generalization ability of the algorithms. To avoid this problem, one can maintain a separate set of validation data and use the wrapper approach [4] for optimizing kernel parameters. This assumes we have enough labeled data for training and validation. Furthermore, optimizing the kernel's parameters on a single set of validation data has the risk of overfitting to the validation data.

3.1 Details about features and kernels

In order to determine if a region is in shadow we will look at its chromatic, intensity and textural properties. For each region, we compute a 21-bin histogram for each of the components (L*,a*,b*) of the perceptually uniform color space CIELAB. To represent texture, we compute a 128-bin texton histogram. We run the full MR8 filter set [38] in the whole data set and cluster the filter responses into 128 textons using k-means. Shadow regions tend to be less textured and darker[†]. The CIELAB color space has been shown to perform well for shadow edge identification in outdoor scenes [18] as well as to improve reflectance segmentation [8]. The two color opponent channels behave differently under illumination changes. Especially in outdoor environments, the b* channel (yellow-blue) is more sensitive to shadows than the a* channel (red-green), which is shadow invariant to a certain degree [35]. To compare textures between regions we use the $\mathcal{X}^2$ distance between their texton histograms. For color histograms, it is more appropriate to use the Earth Mover's Distance (EMD) [28] because neighboring bins in the L*,a*,b* histograms represent proximate values and their ground distance is uniform (property of the CIELAB space), in contrast to texton histograms. Furthermore, EMD is more accurate in measuring distances between histograms of continuous entities (such as L,*a,*b), it is less sensitive to quantization error and it

†. We work with web quality type of images, captured by sensors that are not linearly calibrated. Hence, darker regions tend to have more compressed ranges and due to quantization, some contrast is lost.

can be efficiently computed for 1D histograms. Since our features are normalized histograms (unit mass), both the $\mathcal{X}^2$ and EMD distances are metrics. Hence, we can use them in the form of extended Gaussian distances [17, 42]. Our kernel therefore has the form:

$$K(x,y) = \sum_{l \in \{L,a,b,t\}} w_l \exp\left(-\frac{1}{\sigma_l} D_l(x,y)\right), \quad (10)$$

where D_L, D_a, D_b are EMD distances for L*, a*, b* histograms, and D_t is $\mathcal{X}^2$ distance for texton histograms.

3.2 Details about the optimization grid

Our task is to optimize the leave-one-out error over 8 kernel parameters, which are the kernel weights $\{w_L, w_a, w_b, w_t\}$ and the scaling factors $\{\sigma_L, \sigma_a, \sigma_b, \sigma_t\}$. We define an 8-dimensional grid; each dimension corresponds to a kernel parameter. The discrete values for each scaling factor σ_l form a set of multiples of the mean distance. This is inspired by a common heuristic in practice: using the mean of the pairwise distances as the scaling factor [42]. If $\{x_1, \ldots, x_n\}$ is the set of training examples, the mean distance is computed as $\mu_l = \frac{1}{(n-1)n}\sum_{i \neq j} D_l(x_i, x_j)$. The possible discrete values of σ_l are $\{s\mu_l | s \in \{\frac{1}{8}, \frac{1}{6}, \frac{1}{4}, \frac{1}{2}, 1, 2, 4, 6, 8\}\}$. For the weight w_l of a base kernel, we use $\{s/40 | s \in \{1, \cdots, 10\}\}$ as the set of possible values.

3.3 Details about error rate computation

Our optimization criterion is the leave-one-out balanced error rate. This requires having a threshold for separating between positive and negative predictions. While the LSSVM classifier also has the default threshold of 0, this setting is optimized for the total error rate instead of the balanced error rate. To resolve this issue, we first use Platt scaling [26] to map the decision values of LSSVM to probabilities, and then use the probability threshold of 0.5.

More specifically, suppose f_i is the leave-one-out score for the i^{th} training example. We map f_i to a probability by fitting a sigmoid function:

$$P_{a,b}(f_i) = \frac{1}{1 + \exp(af_i + b)}, \quad (11)$$

where a, b are the parameters of the function. Let N_+ and N_- be the number of positive and negative training examples, and y_i the label of the i^{th} training example. Assuming a uniform uninformative prior over probabilities of correct labels, the MAP estimates for the target probabilities[‡] are: $t_i = \frac{N_+ + 1}{N_+ + 2}$ if $y_i = 1$ and $t_i = \frac{1}{N_- + 2}$ otherwise. The parameters a, b are set by solving the regularized maximum likelihood problem:

$$\max_{a,b}. \sum_i \left(t_i \log(P_{a,b}(f_i)) + (1-t_i)\log(1 - P_{a,b}(f_i))\right).$$

We solve the above optimization problem using Newton's method with backtracking line search [24].

‡. The target probabilities model the out-of-sample data with the empirical density of the training set, but with a finite probability of opposite label. Using these non-binary targets reduces overfitting of the sigmoid [26].

3.4 GPU acceleration and running time

An advantage of the proposed kernel learning algorithm is its efficient GPU implementation. The kernel learning algorithm consists of many iterations, each iteration involves computing a kernel, solving an LSSVM, and calculating the leave-one-out balanced error rate. Computing a kernel for a set of kernel weights requires several element-wise matrix exponentiation and additions. These can be done efficiently using a GPU. Furthermore, the distance matrices between training regions $D_l(\cdot, \cdot)$ need to be computed just once. Solving an LSSVM involves a series of matrix operations (as described in Sec. 2.3), and these operations have GPU implementation. Computing the leave-one-out error can be done efficiently, even on a CPU.

In our experiments, the number of training examples is around 12K, corresponding to a kernel matrix of size $12K \times 12K$. For each iteration, our Matlab GPU implementation takes 1.25s to load the distance matrices to the GPU, 0.25s to compute the kernel, 7.5s to train an LSSVM, and 0.2s to compute the leave-one-out error (including Platt's scaling). In total, it takes less than 10s per iteration. Without GPU, each iteration takes about 30s. We run our algorithm for 500 iterations, and the training procedure typically terminates within 1.5 hours. Notably, the random grid optimization procedure can also be parallelized to further reduce the training time.

4 ADDING CONTEXT TO SHADOW DETECTION

We enhance shadow detection by embedding the shadow region classifier in an MRF framework. As before, we first segment an image into regions $\{R_i\}$. We construct a graph where each node corresponds to a region and each edge corresponds to a pair of neighboring regions. We associate a binary label x_i for each graph node to indicate whether the corresponding region is in shadow or not ($x_i = 1$ if R_i is shadow, and $x_i = -1$ otherwise). Shadow detection is then posed as the minimization of the following energy function:

$$\sum_i \Phi(x_i) + \overbrace{\sum_{i,j \in \Omega^a} \psi_a(x_i, x_j)}^{affinity} + \overbrace{\sum_{i,j \in \Omega^d} \psi_d(x_i, x_j)}^{disparity}. \quad (12)$$

In the above, $\{x_i\}$ is the set of variables that need to be optimized. The first term of the above energy is the sum of unary potentials. The unary potential $\Phi(x_i)$ is based on the probability that region R_i is in shadow, and this depends on the decision value of the shadow region classifier (Sec. 3). The last two terms are the sums of pairwise potentials. They correspond to contextual cues between neighboring regions.

4.1 Unary potentials

We define the unary potential $\Phi(x_i)$ in terms of the predictions of the single region classifier (LSSVM with probabilistic output): $\Phi(x_i) = -\omega_i P(x_i|R_i)$, where ω_i is the area in pixels of the region R_i, and $P(x_i|R_i)$ is Platt's scaling probability. The unary potential encourages agreement between the label of a region and the prediction based on the appearance of the region.

4.2 Affinity pairwise potentials

We introduce two types of pairwise potentials to model relationships between neighboring regions. The affinity potential ψ_a penalizes assigning different labels for similar regions. We define a similarity metric between two regions R_i, R_j based on the kernel used for the single region classifier. For R_i and R_j where $K(R_i, R_j) > 0.5$, we establish the affinity potential term as:

$$\psi_s(x_i, x_j) = \begin{cases} \omega_{ij}\, K(R_i, R_j) & \text{if } x_i \neq x_j, \\ 0 & \text{otherwise.} \end{cases} \tag{13}$$

The penalty for having different shadow labels is the similarity between the two regions weighted by the geometric mean of the areas of the regions, i.e., $\omega_{ij} = \sqrt{\omega_i \omega_j}$, where ω_i and ω_j are the areas of R_i and R_j, respectively.

4.3 Disparity pairwise potentials

For disparity potentials, we use a classifier to identify shadow-nonshadow transitions between two regions of the same material. We train an LSSVM that takes a pair of neighboring regions and predicts if the input is a shadow-nonshadow pair. We use an RBF kernel with the following features:

- The $\mathcal{X}^2$ distance between the texton histograms.
- The EMD between corresponding L*, a* and b* histograms of the two regions.
- The average RGB ratios. Given two regions i, j, compute the ratios of region average intensity for each R, G and B channels: $\rho_R = R_i/R_j, \rho_G = G_i/G_j, \rho_B = B_i/B_j$, and the feature vector is: $((\rho_R + \rho_G + \rho_B)/3, \rho_R/\rho_B, \rho_G/\rho_B)$.

We penalize the same shadow labeling for the pairs of regions that are classified as positive by the learned classifier. The penalty is the prediction confidence weighted by the geometric mean of the regions' areas:

$$\psi_d(x_i, x_j) = \begin{cases} 0 & \text{if } x_i \neq x_j, \\ \omega_{ij}\, P^d(1|R_i, R_j) & \text{otherwise.} \end{cases} \tag{14}$$

The energy function (12) requires optimizing the node labels for a sparse graph. This energy function has submodular pairwise interactions $\psi_a(x_i, x_j)$ and supermodular interactions $\psi_d(x_i, x_j)$. We optimize it using QPBO [20, 27].

5 Shadow Removal

We propose a method to generate a shadow-free image once the shadow regions in the image have been detected. For each shadow region, we first identify a neighboring non-shadow region that shares the same material properties. We refer to the later as the lit neighbor. Then, we use the lit neighbor appearance to relight the shadow region, effectively removing the shadow from the image.

5.1 Region Relighting

Given a shadow region R_s and a neighboring non-shadow region of the same material R_l, we look for a transformation T that relights R_s. Since the two regions are close to each other and have the same material, a transformed version of R_s should closely resemble the appearance of the lit region R_l. The relighting transformation T depends on the appearance of the lit region. We have:

$$T(R_s, R_l) = \widehat{R_s}, \text{ such that } \widehat{R_s} \approx R_l \tag{15}$$

We perform the relighting transformation in CIELab color space. First, we compute the 50 bin histogram of the luminance values, L channel, of R_l ($H_{R_l(L)}$). Then, we apply histogram matching so that the shadow region L values match the lit region histogram (we use Matlab's histeq function).

The resulting luminance histogram, $H_{\widehat{R_s}(L)}$, resembles that of the lit region $H_{R_l(L)}$ while still preserving a similar shape to the original shadow values $H_{R_s(L)}$. Figure 2(b) depicts the results of this step if we convert back to RGB with the adjusted luminance values for the shadow region. Figure 2(a) shows the original input image with R_l boundaries drawn in yellow and R_s boundaries drawn in black. The image segmentation often produces small inaccuracies around the regions' boundaries. That is, few shadow pixels leaking into a lit region (or vice versa) or small chunks of different material(s) are getting added to an otherwise homogeneous region. These spurious pixels modify the range of luminance values of a given region, which can severely affect the histogram matching results. Hence, we apply the relighting transformation using only the core pixels of each region. That is, we exclude the outer perimeter pixels (resulting of eroding each region with a 3x3 identity matrix as neighborhood structure).

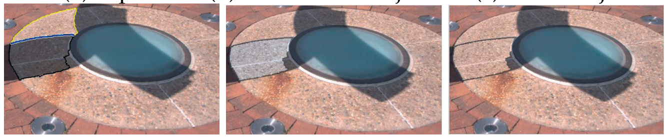

Fig. 2: Shadow region relighting using lit neighbor. (a) Shadow region and lit neighbor: shadow region depicted with black boundaries, lit neighboring region depicted with yellow boundaries, common boundary drawn in blue. (b) RGB reconstruction showing the result of histogram matching on L channel for the shadow region. (c) Shadow region relit, results after the adjustments in a and b channels.

As a second step, we adjust the a channel of the shadow region by adding the difference between the median a values of R_l and the median a values of R_s. Finally, the same operation is carried out for the b channel to complete the relighting process $T(R_s, R_n)$ yielding $\widehat{R_s}$. In figure 2(c), we can see the reconstructed RGB image showing the final relighting results.

5.2 Classifier for Lit Neighbors

We propose a classifier that takes as input a shadow region and a neighboring lit region. For each shadow region R_s we need to identify which of its lit neighbors R_i shares the same material with R_s. If a lit neighbor shares the same material then it can be used to relight R_s by applying the transformation T previously described in Section 5.1. Hence, we select features that describe: i) the similarity between R_s and R_i, ii) the transformation defined by the pair of regions $T(R_s, R_i)$, and iii) the results of applying that transformation. If R_s and R_i, have the same material, the relit shadow region $\widehat{R_s}$ and the lit region should be similar in color and texture. We compute the following features:

- RGB color ratios between R_i and R_s: t_r, t_g, t_b encoded as $\frac{t_r+t_g+t_b}{3}$, $\frac{t_r}{t_b}$, $\frac{t_g}{t_b}$ [15].
- Earth Mover's Distance(EMD) between each region's luminance histograms.
- Median a and b offsets defined by $T(R_s, R_i)$.
- EMD between the a and b histograms of R_i and the relit region $\widehat{R_s}$
- $\mathcal{X}^2$ distance between the texton histogram of the relit region $\widehat{R_s}$ and the texton histogram of the combined regions $\widehat{R_s}$ and R_i.

For all the feature computations we only consider the central pixels of each region. The L, a and b histograms contain 50 bins. Positive training examples are pairs of neighboring regions sharing the same material with one being shadow and the other lit. For negative training examples the lit region is of a different material than the shadow region. We train a probabilistic SVM classifier[3] with a Gaussian RBF kernel. For model selection, we perform grid search with 5-fold cross validation. We use the fast version of LibSVM implemented by Li *et al.* [23].

To generate a texton codebook we ran the full MR8 filter set [38] on the whole data set and cluster the filter responses into 128 textons using K-means.

5.3 Iterative Shadow Removal

Our shadow removal method takes as input an RGB image and a binary shadow mask. First, we segment the image into regions and automatically label each region as shadow or lit (as described in Section 3). For each shadow region we extract its lit neighboring regions building a set of lit-shadow pairs.

Second, we compute the features for each pair of regions, as described in the previous section, and run the classifier. The positive classifications are selected as candidate relighting pairs. If for a shadow region more than one lit neighbor is classified as positive we only consider the one with the highest classification confidence.

On the next stage, region relighting is performed on the candidate relighting pairs according to the process described in section 5.1. After that, we label the set of relit regions as lit. Hence, new pairs of lit-shadow regions are created so we can start a new cycle of identifying candidate relighting pairs using the classifier and then relighting regions based on the positive classifications. Figures 3(b) and 3(c) depict the shadow removal results after the first and second iterations of our method, respectively. As can be observed, there are three isolated shadow regions (no lit neighbors) that are successfully relit in the second round.

At the final step, we address the so far ignored boundary pixels. To remove the shadow in the outer perimeter p_s of a relit shadow region $\widehat{R}_s$, we propose a two step operation:

1) Adjust the L, a and b values of the pixels in p_s based on the core pixels of $\widehat{R}_s$. First, we compute the mean L, the median a and the median b for the core pixels and for the boundary pixels. Then, we add the differences to the pixels in p_s.
2) Smooth the new boundary pixels' values. We convert the results from the previous step to RGB. Then, we run a Gaussian filter at the locations of p_s to obtain the final values for the boundary pixels.

TABLE 1: Shadow detection performance of several region classifiers on the UIUC dataset.

Method	Shadow	Non Shadow	BER
UnarySVM [10]	45.7	8.9	27.3
MK-SVM [41]	20.5	4.3	12.4
ConvNet [11]	16.4	5.3	10.6
LooKOP (this paper)	**14.9**	**4.2**	**9.5**

This table shows the error rates for shadow area (2^{nd} column), non-shadow area (3^{rd} column), and the balanced error rate (last column). For error rates, shown as percentages, a lower number indicates better performance. Best results are printed in bold.

TABLE 2: Performance of shadow detection pipelines on the UIUC dataset. All the methods use pairwise potentials between neighboring regions.

Methods	Shadow	Non Shadow	BER
UnarySVM+Pairwise [10]	28.4	4.8	16.6
ConvNet+CRF [11]	15.3	4.5	9.9
LooKOP+MRF (this paper)	**9.9**	**4.4**	**7.2**

6 Shadow Detection Experiments

We perform experiments on the UCF [43] and the UIUC shadow datasets [9]. Both of these datasets come with shadow masks, which are used for evaluating performance. For the UCF dataset, the shadow masks are provided by human annotators. In contrast, the shadow masks of the UIUC dataset are obtained automatically. Each shadow image of the UIUC dataset has a corresponding non-shadow version, taken without the objects that cast the shadows. The shadow masks for the UIUC dataset are created based on the difference between the shadow and nonshadow images. This process is illustrated in Fig. 4. It typically leads to a good shadow mask, but not always.

For quantitative evaluation, we compare the shadow masks produced by our method to the provided shadow masks. We compute the classification error rates at the pixel level on shadow and non-shadow areas separately. We also report the Balanced Error Rate (BER), defined as:

$$\text{BER} = 1 - \frac{1}{2}\left(\frac{TP}{TP+FN} + \frac{TN}{TN+FP}\right) \qquad (16)$$

We experiment with the different settings of our method. First, we evaluate the single region classifier without the contextual cues from pairwise potentials. We refer to this method as Leave-one-out Kernel Optimization (LooKOP). Second, we evaluate the fully-developed shadow detection framework, embedding the LooKOP in the MRF framework; this will be called LooKOP+MRF. We also experiment with a variant of LooKOP+MRF where the Disparity Pairwise potentials are removed.

6.1 Comparison between Single Region Classifiers

Table 1 compares the performance of several region classification methods, which predict shadow/non-shadow labels for each region separately (i.e., no pairwise potentials are

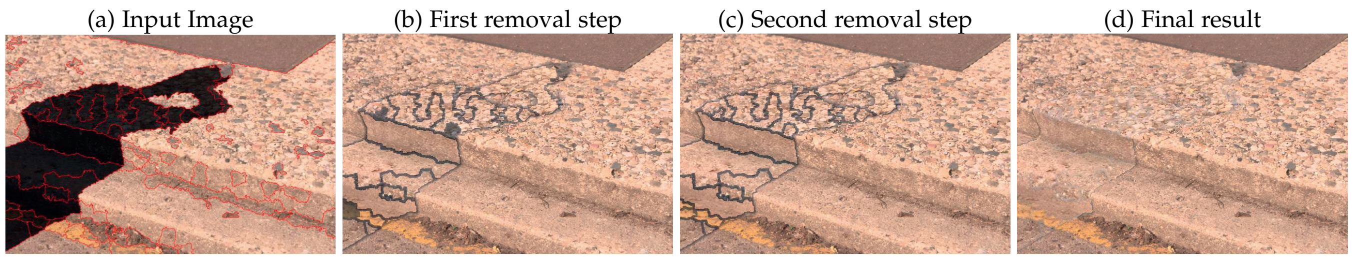

Fig. 3: Shadow removal pipeline. (a) Input image with overlaid shadow mask, boundary of segmented regions depicted in red. (b) Removal results after first iteration of our method. (c) Removal results after the second iteration. (d) Final removal results after boundary areas are relit.

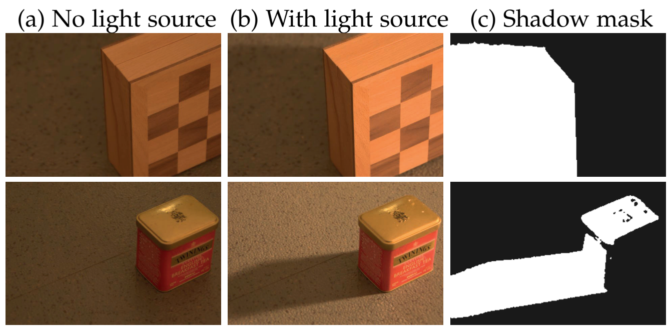

Fig. 4: Generation of 'ground truth' shadow masks on UIUC dataset. Two images of the same scene are taken, with and without blocking a light source. The shadow mask is obtained by considering the difference between two images. This process typically yields a good shadow mask, but not always. Top: good shadow mask. Bottom: bad shadow mask; the top of the tea box is not in shadow, and should have not been a part of the shadow mask.

TABLE 3: Performance on the UCF dataset.

Methods	Shadow	Non Shadow	BER
UnarySVM [10]	63.3	**2.7**	33.0
ConvNet [11]	27.5	7.9	17.7
LooKOP (this paper)	22.9	6.2	14.5
UnarySVM+Pairwise [10]	26.7	6.3	16.5
ConvNet+CRF [11]	22.0	7.4	14.7
LooKOP+MRF (this paper)	**20.0**	6.4	**13.2**

UnarySVM, ConvNet, and LooKOP are the methods that predict the shadow label of each region individually. The others are fully-developed methods, incorporating contextual cues in terms of pairwise potentials. For each evaluation category, the best performance is printed in bold.

used). UnarySVM [10] uses a predefined kernel SVM. MK-SVM [41] combines multiple kernels, but kernel weights are not learned. ConvNet [11] combines the predictions of tw convolutional neural networks: one for shadow regions an the other for shadow boundaries. This method is referred t as ConvNets (Region+Boundary) in [11]. LooKOP, propose in this paper, optimizes the kernel parameters to minimiz the leave-one-out balanced error rate. As can be seen LooKOP outperforms the other methods in all evaluatio categories. Comparing the balanced error rate of LooKOI and MK-SVM, we note more than 20% error reduction. MK-SVM has two main differences from LooKOP: (i) MK-SVM uses SVM instead of LSSVM; (ii) MK-SVM uses predefined kernel weights and scaling factors, instead of learning them. This demonstrates the importance of learning the kernel parameters. The advantage of LooKOP over other methods is more significant for shadow regions. All methods have higher error rates for shadow regions, commensurate with the difficulty of detecting shadows.

6.2 Incorporating pairwise potentials

Table 2 compares the performance of several fully-developed shadow detection methods that enhance the single region classifiers by incorporating pairwise potentials between neighboring regions. LooKOP+MRF, proposed in this paper, combines the benefits of a learned kernel and the contextual cues from affinity and disparity pairwise potentials. ConvNet+CRF [11] achieved the prior state-of-the-art result on this dataset. Considering the balanced error rate, we see that LooKOP+MRF outperforms ConvNet+CRF by 27.3%. The performance gap between these two methods is even wider in shadow regions.

Table 3 reports the performance of these methods on the UCF dataset. The first three methods only use the unary potentials while the last three combine both unary and pairwise potentials. The incorporation of pairwise potentials improves the performance of all methods, judging by the balanced error rate. Our proposed method, LooKOP+MRF

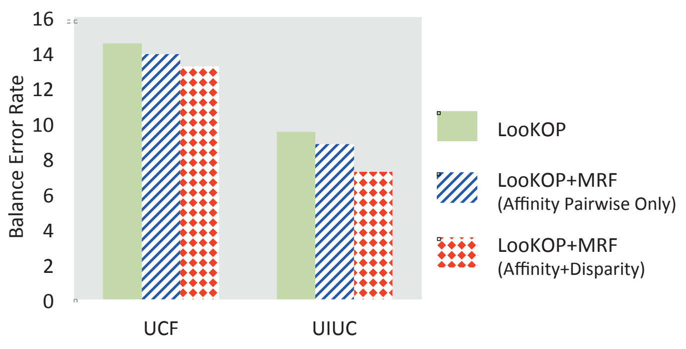

Fig. 5: Benefit of pairwise potentials. This figure shows the balanced error rates for three methods: LooKOP, LooKOP with Affinity pairwise potential, and LooKOP with both Affinity and Disparity pairwise potentials.

Fig. 5 shows the benefits for embedding LooKOP in an MRF framework, adding pairwise potentials to incorporate contextual cues. These pairwise potentials reduce the balanced error rates on both datasets. This figure also illustrates the importance of the disparity pairwise potentials.

6.3 Error distribution analysis

To gain insight into our method's performance, we analyze the distribution of per image error within each test set. In terms of balanced error rate (BER), 74% of the images in UIUC have less than 10% BER (see Figure 6.b solid black line) and no image has more than 30% BER. For UCF, 66% of the images have less than 10% BER, whereas around 15% of the images have BER of 30% or higher (see Figure 6.a).

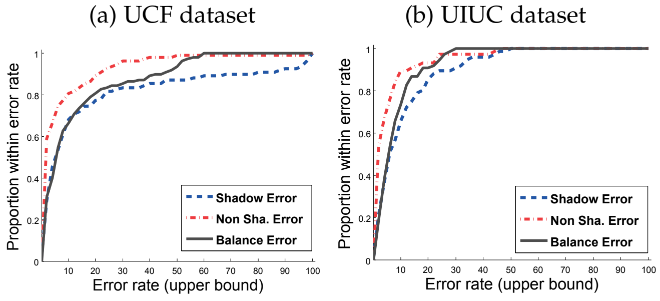

Fig. 6: Per image error distribution within test sets. Proportion of images (Y axis) within a given error rate upper bound threshold (X axis).

In Figure 6 we plot the cumulative proportion of images within the classification error rate in non-shadow areas as red dashed lines. As can be seen from this figure, the majority of images have low error rate in non-shadow areas. The classification error rate in non-shadow areas is commonly referred to as False Positive Rate. We present results in terms of the classification error in shadow areas with blue dashed lines . As can be seen, for around 78% of the images in UIUC less than 15% of the shadow pixels are miss-classified. Whereas for UCF, 74% of the images have less than 15% shadow pixels miss-classified. The classification error rate in shadow areas is commonly referred to as False Negative Rate.

6.4 LooKOP iterative optimization procedure

We train a shadow region classifier using an iterative optimization procedure with 500 iterations. In Figure 7, we plot the leave-one-out balanced error rate (of training data) as a function of the number of iterations involved in optimizing the kernel parameters. For a reliable result, we perform this experiment multiple times (10 times for UIUC and 5 times for UCF). We plot the mean balanced error rate and the standard deviation over multiple experiments in Figure 7. As can be seen, we use 500 iterations, but the optimization procedure converges after 300 iterations.

To further verify the effectiveness of the LooKOP optimization we sampled the actual error rate on the test set every 20 iterations. Results are plotted in Figure 8. As can be seen, the error decreases significantly over iterations (28% and 20% decrease in UIUC and UCF respectively). This verifies the benefits of the proposed learning approach. The testing error reduction is consistent with the leave-one-out cross validation error shown in Figure 7.

6.5 Kernel type choice

To analyze how the choice of kernel type affects the results of our method we run LooKOP using a model with all

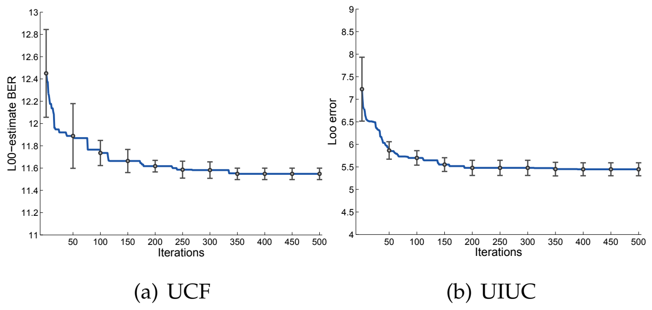

Fig. 7: Leave-one-out balanced error rate as a function of iterations of the kernel optimization. The optimization algorithm converges after around 300 iterations.

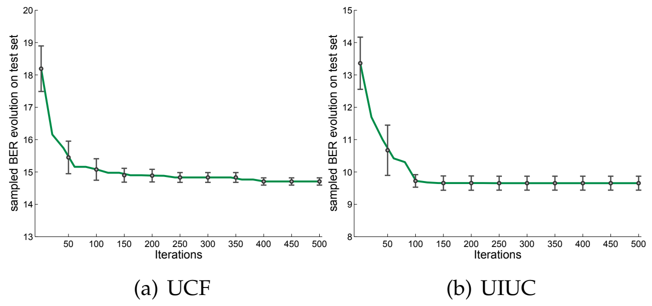

Fig. 8: Balanced error rate on test set. This shows mean BER on the test set over 5 trials. Error bars illustrate variance among trial runs.

$\mathcal{X}^2$ kernels. There is some difference between using $\mathcal{X}^2$ versus using EMD, but the gap is small in favor of EMD. We run our method with all $\mathcal{X}^2$ kernels and track the testing error. With all $\mathcal{X}^2$ kernels the mean error after 500 iterations is 10.2 versus 9.5 in UIUC, and 15.2 versus 14.5 in UCF. We show detailed results in Table 4. We show that our method improves the performance of a model with all $\mathcal{X}^2$ kernels: 26% error reduction in UIUC (24% in UCF) after 500 iterations. These findings suggest that our kernel optimization is not exclusive to any specific type of kernel.

TABLE 4: Role of kernel choices in our framework.

Dataset		1	100	300	500
UIUC	LooKOP	13.5	10.8	9.8	9.6
UIUC	LooKOP-$\mathcal{X}^2$	13.9	11.3	11.0	10.2
UCF	LooKOP	18.2	15.3	15.0	14.6
UCF	LooKOP-$\mathcal{X}^2$	20.1	16.5	15.9	15.2

Balanced error rate on the test set as a function of iterations. LooKOP uses a mixture of $\mathcal{X}^2$ and EMD kernels, while LooKOP-$\mathcal{X}^2$ uses all $\mathcal{X}^2$ kernels.

6.6 SVM vs LSSVM

Our proposed single region classifier (without the MRF) achieves good shadow detection performance. To understand whether it is due to the use of LSSVM as a classifier, we train a regular SVM with the same learned kernel parameters (learned using LooKOP optimization). We observe that regular SVM achieves similar error rate as LSSVM. The error rates of using SVM on UIUC and UCF datasets are 9.4% and 15.3% respectively. These figures are similar to the error rates of LSSVM, which are 9.5% and 14.5%. Thus, the improved performance is due to the kernel learning approach rather than the SVM formulation.

6.7 Cross-dataset evaluation of shadow detection

To evaluate cross-dataset performance of our proposed method, we train LooKOP on UCF (or UIUC) and test on UIUC (or UCF). In these experiments, we use per pixel accuracy as evaluation metric to be able to compare to the published results of Guo *et al*. [10] and Khan *et al*. [19].

TABLE 5: Cross-dataset evaluation

	Train on UIUC		Train on UCF	
	UIUC-Test	UCF-Test	UCF-Test	UIUC-Test
UnarySVM [10]	81.7	68.9	87.5	75.5
ConvNets [19]	92.3	82.8	89.3	80.5
LooKOP	**93.1**	**85.2**	**90.2**	**92.5**

This table shows the per pixel accuracies for shadow detection as percentages. The higher number indicates better performance. Best results are printed in bold.

Table 5 shows the cross-dataset performance. The results for training on UIUC dataset are shown in Columns 2 and 3. As expected, models trained on the smaller UIUC training set do not generalize well on the more challenging and larger UCF testing set. All methods experience a notable decrease in performance when testing cross-dataset. However, when testing on UCF testing set, our method, LooKOP trained on UIUC training set, has the lowest decrease in performance (8.5% versus 10.3% and 15.6% respectively), and also achieves the best shadow detection results at 85.2% per pixel accuracy. The last two columns of Table 5 shadow the performance for methods trained on the UCF training set. Our method, LooKOP generalizes remarkably well to UIUC testing set, in contrast to [10, 19]. It is worth noting, that LooKOP trained on the larger UCF training set, achieves 92.5% per pixel accuracy on UIUC testing set, which is just slightly worse than when LooKOP is trained on UIUC training set, at 93.1%. This indicates that our proposed method is able to generalize well if it is trained on decently large dataset.

6.8 Qualitative evaluation

Fig. 9 shows some examples of shadow detection using LooKOP+MRF. Overall, this method works well, showing detection results with high precision and recall. Most of the errors occur at the boundaries between shadow and non-shadow areas. These errors are possibly propagated from the process of superpixel segmentation and grouping.

Fig. 10 shows several cases where there are significant differences between the predicted shadow mask and the annotated shadow mask. Interestingly, not all mismatches correspond to a bad result, due to the imperfection of the annotation. The shadow mask in the first row of this figure should not have contained the top of the box, as explained in Fig. 4. For the second row, the self-shadow regions should have been part of the shadow mask. The third row shows a challenging case. Our method correctly classifies almost all regions, except for a small brick. This example illustrates a limitation of appearance-based approaches that ignore scene geometry; it cannot distinguish between a dark brick from a brick in shadow. Unfortunately, the Markovian assumptions and pairwise potentials between neighboring regions do not help in this case. Fig. 11 illustrates another failure mode.

TABLE 6: Evaluation of Shadow Detection-Removal pipelines on the UIUC dataset.

Method	All Regs.	Lit Regs.	Shadow Regs.
Original Error	13.7	4.6	42.0
Auto Matting [10]	7.4	5.4	13.9
Bayesian Refinement [19]	6.8	5.1	12.1
Region Relighting (ours)	**6.1**	**4.9**	**9.6**

The evaluation metric is Root Mean Squared Error(RMSE), the lower the RMSE the better. Best results are printed in bold. Original Error denotes RMSE when no shadow removal is performed.

Our algorithm fails to detect elongated soft shadows in the image (i.e., the shadow corresponding to the fingers). This is partly due to the propagated error from the process of superpixel segmentation and grouping.

6.9 Testing times

The average running time of our method for an image of size 500x335 pixels, is 13 seconds. The individual steps are: initial segmentation, merge superpixels, compute features, precompute distance matrices, compute test kernel, predict region confidences, MRF optimization, which take: 1, 2, 6, 2, 0.25, 0.25, and 1 seconds, respectively.

7 Shadow Removal Results

In this section we present quantitative and qualitative results of our shadow removal method using the shadow removal subset of the UIUC dataset[9]. This dataset contains 32 training images for which we manually annotated ground truth for our lit neighbor classifier. The test set contains 48 shadow images for which there is a corresponding shadow-free image, which is used as ground truth for shadow removal evaluation.

7.1 Quantitative Results

In table 6, we evaluate our shadow removal method using as input our shadow detection results. We compare to the state of the art shadow detection-removal pipelines of Guo *et al*. [10] (Auto Mating) and Khan *et al*. [19] (Bayesian Refinement). As evaluation metric we use the Root Mean Squared Error (RMSE) in CIELab space between the provided shadow-free images and the results of applying shadow removal on automatically detected shadow masks. We also include the RMSE between the shadow-free image and the input shadow image when no shadow removal is performed, which we denote as Original Error. As we can see in Table 6, our overall shadow removal error is significantly lower than the state of the art: 6.1 versus 7.4 and 6.8. For shadow region pixels our performance reduces the error by a 30% and a 20% respectively, yielding a RMSE of 9.6 units. The performance we get on lit regions is also the lowest error at 4.9. However, this is slightly worse than the Original Error on these non shadow regions. This is due to small faults in the segmentation such that lit pixels leaked into shadow regions. With no shadow removal applied, the RMSE is 4.6.

We also evaluate our performance in shadow regions for the inner pixels and for the border pixels separately, The shadow removal error for inner regions is 8.81, whereas the

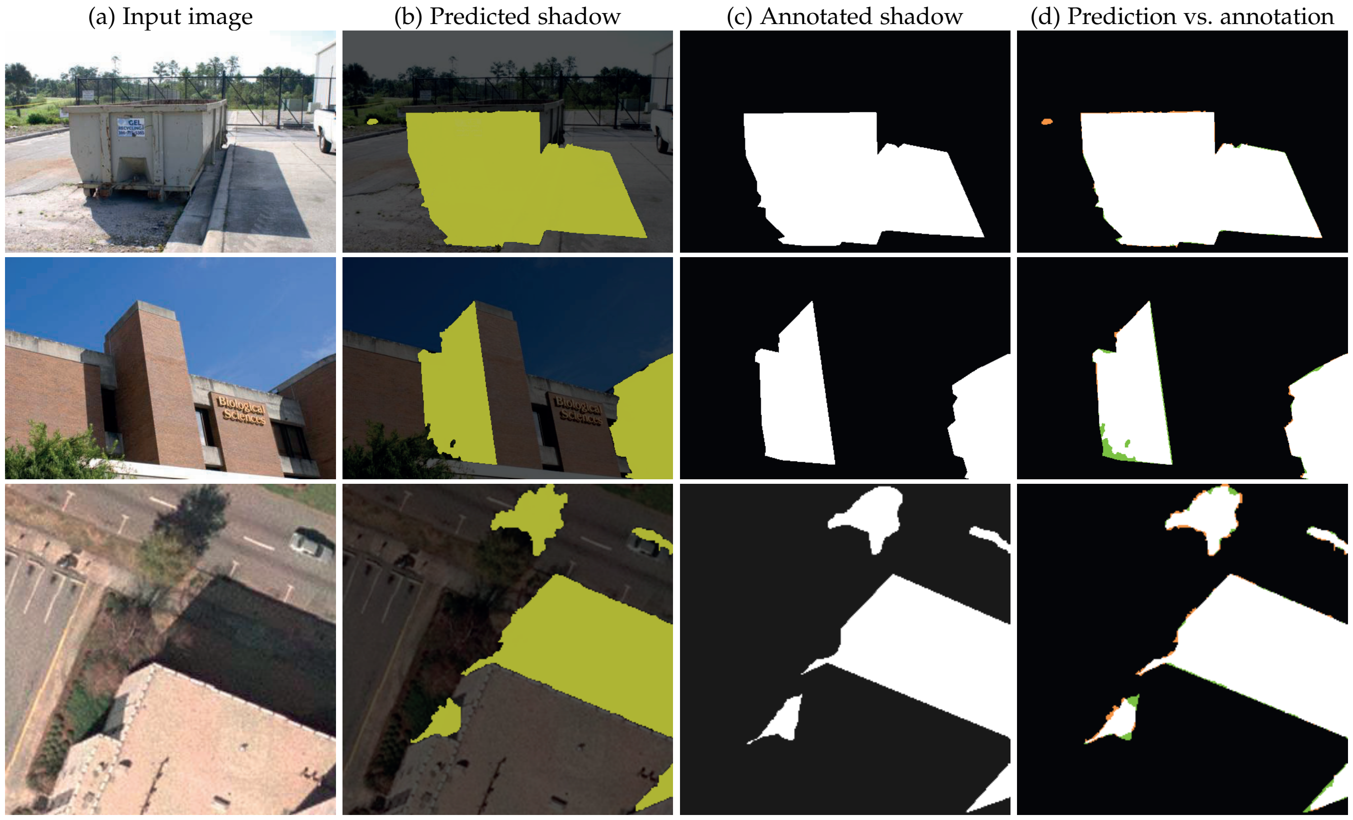

Fig. 9: Shadow detection examples. The last column compares the predicted shadow and the provided annotation; false positive is shown in orange, false negative in green. This figure is best seen in color. Most of the errors occur at the shadow boundaries.

error in the border regions is noticeably worse at 14.09. Here the inner shadow regions are the shadow regions excluding their outer perimeter pixels (resulting of eroding each region with a 3×3 identity matrix as neighborhood). Moreover, some shadow regions cannot be relit as no suitable lit neighbor is detected by our classifier, or does not exist in the image. The considering only the inner pixels of the shadow regions that were actually relit by our method the RMSE drops to 8.12.

To better evaluate our proposed shadow removal method, we ran it using as shadow mask the provided ground truth instead of detected masks. In Table 7, we compare the results of our shadow removal with the state of the art methods [10, 19] all using ground truth shadows masks.

TABLE 7: Evaluation of Shadow Removal using ground truth shadow masks on the UIUC dataset.

Method	All Regs.	Lit Regs.	Shadow Regs.
Auto Matting [10]	6.4	4.7	11.8
Bayesian Refinement [19]	6.1	4.7	10.5
Region Relighting (ours)	**5.6**	**4.6**	**8.6**

The evaluation metric is Root Mean Squared Error (RMSE), the lower RMSE the better. Best results are printed in bold.

As we can see, with ground truth shadow masks our proposed method achieves the lowest overall error. We also perform best on shadow region pixels at 8.6 RMSE. Which is a 10% improvement with respect to using our detected shadow masks.

7.2 Qualitative Results

Figure 12 presents some qualitative results. Our method produces high quality shadow-free images for a variety of materials and textures. In the first and forth rows our shadow free image presents a noticeable boundary effect around the shadow regions. This is mostly due to inaccuracies in the region segmentation with respect to the actual shadows. However, the quality of the shadow removal in the inner areas is quite high. Table 8 contains the actual error numbers for the images depicted in figure 12. In the fifth row image we can notice some boundaries between relit regions due to poor performance by our boundary processing. The image in the sixth row depicts a case where some regions within the person's shadow were not able to be recovered as no suitable lit region was found by the classifier.

8 Summary

We have proposed a framework for shadow detection and shadow removal in single images. To detect shadows in an image, we first divide it into multiple disjoint regions and use a Least-Squares SVM to compute the shadow probability of each region. In an MRF framework, we jointly optimize the labels of the regions, taking into account contextual influences of neighboring regions. We have performed experiments on two challenging datasets, and observed that

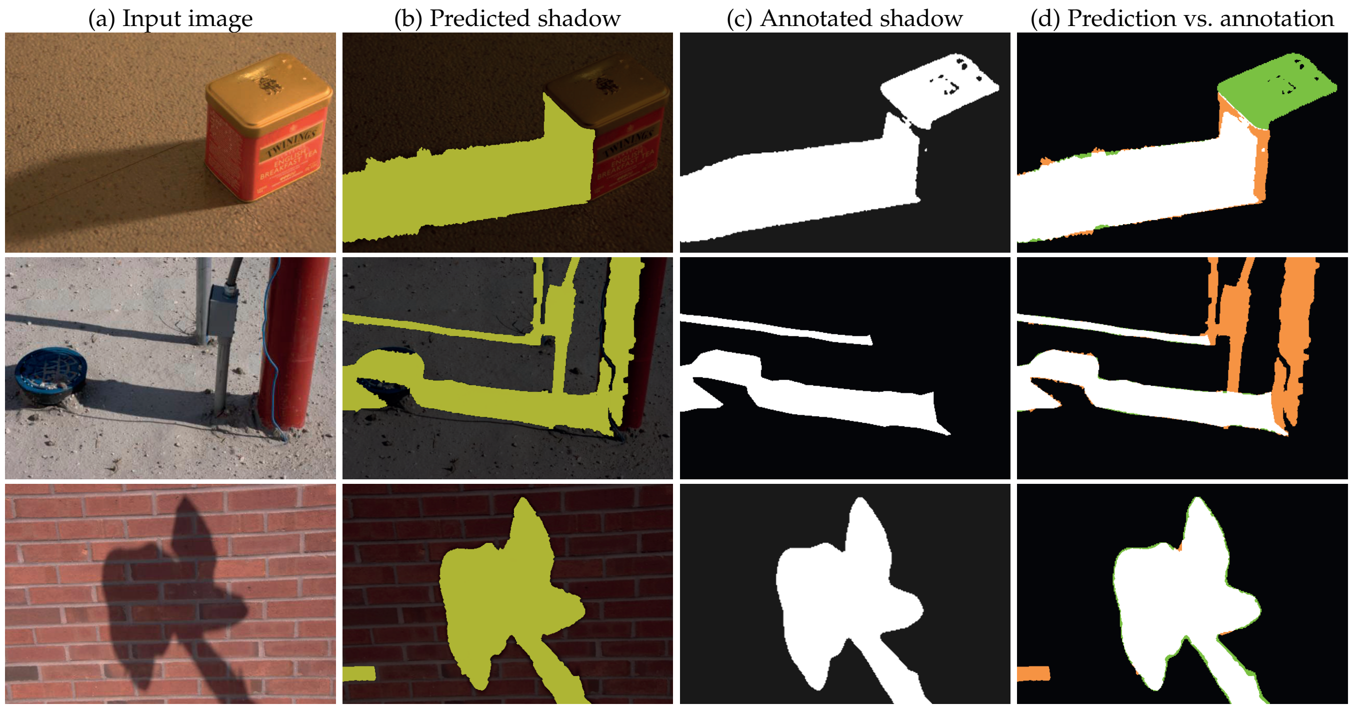

Fig. 10: Examples of significant mismatch between predicted and annotated shadow regions. The last column compares predicted shadow and provided annotation; false positives in orange, false negatives in green. Rows (1,2): imperfect shadow masks cause mismatches. Row (3): limitation of appearance-based approaches that ignore scene geometry.

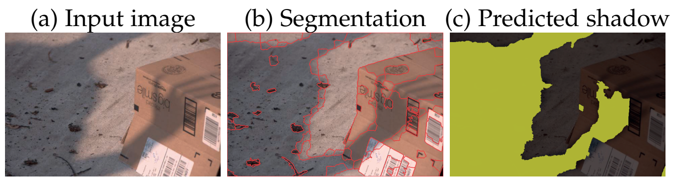

Fig. 11: Failure due to segmentation. Soft and elongated shadow regions, such as the shadows created by the fingers, are hard to detect, partly due to the error during superpixel segmentation and grouping. This process may produce regions that contain both shadow and non-shadow pixels.

TABLE 8: Shadow removal performance on qualitative examples

Image	Total	Shadow	Inner Shadow	Inner Original
Fig.12 1st row	8.88	9.91	9.81	29.64
Fig.12 2nd row	13.76	12.30	12.21	37.49
Fig.12 3rd row	6.47	11.18	11.09	24.16
Fig.12 4th row	3.09	6.07	5.81	41.69
Fig.12 5st row	4.55	11.54	11.20	43.97
Fig.12 6nd row	6.89	12.96	12.66	28.25

RMSE from shadow removal on the images shown in Figure 12. First column shows the total error. Second column depicts the error in shadow regions. The inner shadow error is the error on the inner shadow region, that shadow regions excluding their outer perimeters. Inner original is the error in the shadow core pixels for the original image, with no shadow removal performed.

our method achieves lower error rate than the prior state-of-the-art; the reduction in balanced error rate is as high as 27.3% on the UIUC dataset. Qualitatively, we observe minor errors at the boundaries between shadow and non-shadow areas. Moderate errors can be attributed to the inability to reason about scene geometry and the propagation of error from the segmentation process. We also find multiple cases where there is significant difference between the predicted shadow mask and the annotated mask, but those correspond to imperfect annotation.

We conducted extensive experiments to evaluate the proposed shadow detection method: Leave-one-out kernel optimization (LooKOP). The main strength of LooKOP resides in its ability to efficiently find the optimal kernel parameters using beam search and leave-one-out estimates(loo) of the error rate. Our optimization procedure converges fast (after around 200 iterations). Using Least Squares SVM (LSSVM) with its closed form solution and computationally cheap LOO estimates is what makes our approach feasible. We have shown that using a regular SVM trained with the optimal kernel parameters found by LooKOP achieves similar perfomance. Moreover, LooKOP is flexible enough to work with different kernel metrics. We used LooKOP with all $\mathcal{X}^2$ kernels and obtained comparable perfomance.

We extended our shadow detection pipeline adding a final shadow removal step. When we apply the proposed removal method to the detected shadows, we achieve results that outperform the state of the art in single image shadow removal[10] by 5% in total error, with a 19% reduction in error on shadow pixels. The main contribution of our shadow removal approach is a new region relighting transformation based on histogram matching of luminance values between the shadow region and the neighboring lit region, plus addition of median based offsets in the a and b channels. Furthermore, we propose a new classifier to automatically identify suitable pairs of lit-shadow regions. We demonstrated that the iterative application of the proposed transformation in positively classified pairs of regions outperforms the state of the art on the shadow removal benchmark dataset. Our

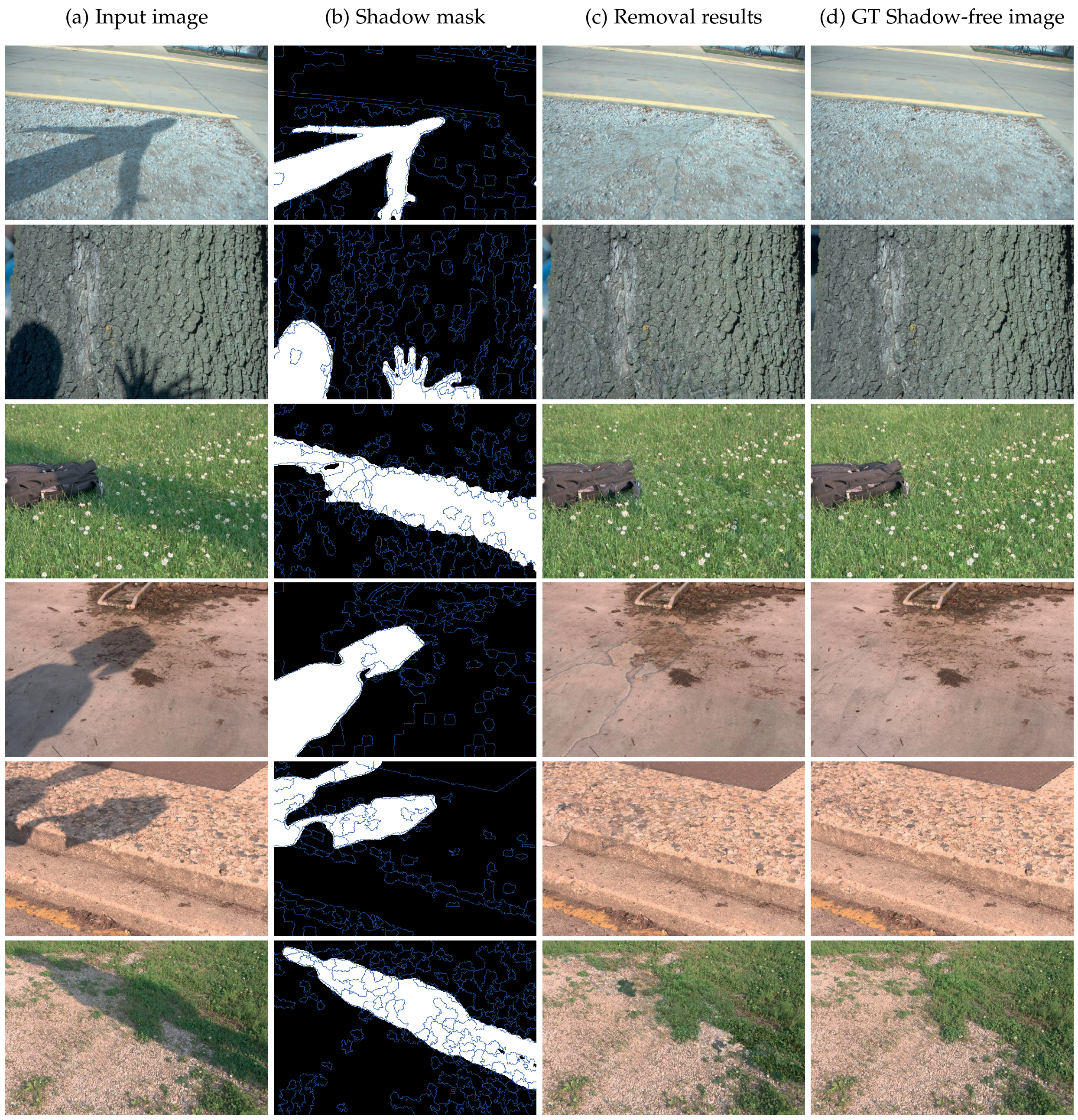

Fig. 12: Shadow removal results. a) Input image (b) Ground truth shadow pixel mask with the region segmentation overlaid in blue. (c) Our shadow removal results. (d) Ground truth shadow free image.

results are specially accurate in the core pixels of the shadow regions.

In future work we will explore alternative ways to deal with the boundary pixels such as in-painting techniques. We will also investigate the benefits of region segmentation tailored to the task of shadow removal.

REFERENCES

[1] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Susstrunk. Slic superpixels compared to state-of-the-art superpixel methods. *IEEE PAMI*, 34(11):2274–2281, 2012.
[2] F. R. Bach, G. R. G. Lanckriet, and M. I. Jordan. Multiple kernel learning, conic duality, and the smo algorithm. In *Proc. ICML*, 2004.
[3] C.-C. Chang and C.-J. Lin. Libsvm: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011.
[4] O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee. Choosing multiple parameters for support vector machines. *Machine Learning*, 46(1–3):131–159, 2002.
[5] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE PAMI*, 24(5):603–619, 2002.
[6] G. Finlayson, M. Drew, and C. Lu. Entropy minimization for shadow removal. *IJCV*, 85:35–57, 2009.
[7] G. Finlayson, S. Hordley, C. Lu, and M. Drew. On the removal of shadows from images. *IEEE PAMI*, 28(1):59–68, 2006.
[8] E. Garces, D. Gutierrez, and J. Lopez-Moreno. Graph-based reflectance segmentation. In *SIACG*, 2011.
[9] R. Guo, Q. Dai, and D. Hoiem. Single-image shadow

detection and removal using paired regions. In *Proc. CVPR*, 2011.
[10] R. Guo, Q. Dai, and D. Hoiem. Paired regions for shadow detection and removal. *IEEE PAMI*, 35(12):2956–2967, 2012.
[11] S. Hameed Khan, M. Bennamoun, F. Sohel, and R. Togneri. Automatic feature learning for robust shadow detection. In *Proc. CVPR*, 2014.
[12] M. Hoai. Regularized max pooling for image categorization. In *Proc. BMVC*, 2014.
[13] M. Hoai and A. Zisserman. Improving human action recognition using score distribution and ranking. In *Proc. ACCV*, 2014.
[14] D. Hoiem, A. A. Efros, and M. Hebert. Recovering surface layout from an image. *IJCV*, 75(1):151–172, 2007.
[15] X. Huang, G. Hua, J. Tumblin, and L. Williams. What characterizes a shadow boundary under the sun and sky? In *Proc. ICCV*, 2011.
[16] A. Jain, S. V. N. Vishwanathan, and M. Varma. Spg-gmkl: Generalized multiple kernel learning with a million kernels. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2012.
[17] F. Jing, M. Li, H. jiang Zhang, and B. Zhang. Support vector machines for region-based image retrieval. In *Proc.ICME*, 2003.
[18] E. Khan and E. Reinhard. Evaluation of color spaces for edge classification in outdoor scenes. In *Proc. ICIP.*, 2005.
[19] S. H. Khan, M. Bennamoun, F. Sohel, and R. Togneri. Automatic shadow detection and removal from a single image. *IEEE PAMI*, 38(3):431–446, March 2016.
[20] V. Kolmogorov and C. Rother. Minimizing non-submodular functions with graph cuts - a review. *IEEE PAMI*, 2007.
[21] J.-F. Lalonde, A. A. Efros, and S. G. Narasimhan. Detecting ground shadows in outdoor consumer photographs. In *Proc. ECCV*, 2010.
[22] G. R. Lanckriet, N. Cristianini, P. Bartlett, L. E. Ghaoui, and M. I. Jordan. Learning the kernel matrix with semidefinite programming. *Journal of Machine Learning Research*, 5:27–72, 2004.
[23] F. Li, J. Carreira, and C. Sminchisescu. Object recognition as ranking holistic figure-ground hypotheses. In *Proc. CVPR*, 2010.
[24] H.-T. Lin, C.-J. Lin, and R. C. Weng. A note on platt's probabilistic outputs for support vector machines. *ML*, 68(3):267–276, 2007.
[25] F. Liu and M. Gleicher. Texture-consistent shadow removal. In *Computer Vision–ECCV 2008*, pages 437–450. Springer Berlin Heidelberg, 2008.
[26] J. C. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *Advances in Large Margin Classifiers*. MIT Press, 1999.
[27] C. Rother, V. Kolmogorov, V. Lempitsky, and M. Szummer. Optimizing binary mrfs via extended roof duality. In *Proc. CVPR*, 2007.
[28] Y. Rubner, C. Tomasi, and L. J. Guibas. A metric for distributions with applications to image databases. In *Proc. ICCV*, 1998.
[29] C. Saunders, A. Gammerman, and V. Vovk. Ridge regression learning algorithm in dual variables. In *Proc. ICML*, 1998.
[30] J. Sherman and W. J. Morrison. Adjustment of an inverse matrix corresponding to changes in the elements of a given column or a given row of the original matrix (abstract). *Annals of Mathematical Statistics*, 20(621), 1949.
[31] J. Sherman and W. J. Morrison. Adjustment of an inverse matrix corresponding to a change in one element of a given matrix. *Annals of Mathematical Statistics*, 21(1):124–127, 1950.
[32] Y. Shor and D. Lischinski. The shadow meets the mask: Pyramid-based shadow removal. *Computer Graphics Forum*, 27(2):577–586, Apr 2008.
[33] J. A. K. Suykens, T. V. Gestel, J. D. Brabanter, B. DeMoor, and J. Vandewalle. *Least Squares Support Vector Machines*. World Scientific, 2002.
[34] J. A. K. Suykens and J. Vandewalle. Least squares support vector machine classifiers. *Neural Processing Letters*, 9(3):293–300, 1999.
[35] T. Troscianko, R. Baddeley, C. A. Parraga, U. Leonards, and J. Troscianko. Visual encoding of green leaves in primate vision. *JoV*, 3:137–, 2003.
[36] V. N. Vapnik. *The nature of statistical learning theory*. Springer-Verlag New York, Inc., New York, NY, USA, 1995.
[37] M. Varma and B. R. Babu. More generality in efficient multiple kernel learning. In *Proc. ICML*, 2009.
[38] M. Varma and A. Zisserman. Classifying images of materials: Achieving viewpoint and illumination independence. In *Proc. ECCV*, 2002.
[39] T.-P. Wu, C.-K. Tang, M. S. Brown, and H.-Y. Shum. Natural shadow matting. *ACM Trans. Graph.*, 26(2), Jun 2007.
[40] C. Xiao, R. She, D. Xiao, and K.-L. Ma. Fast Shadow Removal Using Adaptive MultiScale Illumination Transfer. *Computer Graphics Forum*, 2013.
[41] T. F. Yago Vicente, C.-P. Yu, and D. Samaras. Single image shadow detection using multiple cues in a supermodular MRF. In *Proc. BMVC*, 2013.
[42] J. Zhang, S. Lazebnik, and C. Schmid. Local features and kernels for classification of texture and object categories: a comprehensive study. *IJCV*, 73:2007, 2007.
[43] J. Zhu, K. Samuel, S. Masood, and M. Tappen. Learning to recognize shadows in monochromatic natural images. In *Proc. CVPR*, 2010.

Tomás F. Yago Vicente is a PhD. candidate and a member of the Computer Vision Lab at at Stony Brook University. Before joining Stony Brook he was affiliated with the 3D Group for Interactive Visualization at the University of Rhode Island. He received the Computer Engineer diploma from University of Zaragoza, Spain in 2008.

[

Minh Hoai Nguyen is an Assistant Professor of Computer Science at Stony Brook University. He received a Bachelor of Software Engineering from the University of New South Wales in 2005 and a Ph.D. in Robotics from Carnegie Mellon University in 2012. His research interests are in computer vision and machine learning, especially human action and activity recognition and prediction.

Dimitris Samaras received a diploma degree in computer science and engineering from the University of Patras in 1992, the MSc degree from Northeastern University in 1994, and the PhD degree from the University of Pennsylvania in 2001. He is an associate professor of Computer Science at Stony Brook University. He specializes in deformable model techniques for 3D shape estimation and motion analysis, illumination modeling and estimation for recognition and graphics, and biomedical image analysis.