Learning to Remove Soft Shadows

MACIEJ GRYKA[1,2] , MICHAEL TERRY[3] and GABRIEL J. BROSTOW[1]

[1]University College London
[2]Anthropics Technology Ltd.
[3]University of Waterloo

Manipulated images lose believability if the user's edits fail to account for shadows. We propose a method that makes removal and editing of soft shadows easy. Soft shadows are ubiquitous, but remain notoriously difficult to extract and manipulate. We posit that soft shadows can be segmented, and therefore edited, by learning a mapping function for image patches that generates shadow mattes. We validate this premise by removing soft shadows from photographs with only a small amount of user input.

Given only broad user brush strokes that indicate the region to be processed, our new supervised regression algorithm automatically unshadows an image, removing the umbra and penumbra. The resulting lit image is frequently perceived as a believable shadow-free version of the scene. We tested the approach on a large set of soft shadow images, and performed a user study that compared our method to the state of the art and to real lit scenes. Our results are more difficult to identify as being altered, and are perceived as preferable compared to prior work.

Categories and Subject Descriptors: I.4.8 [**Image Processing and Computer Vision**]: Scene Analysis—*Shading*

Additional Key Words and Phrases: shadow matting, shadow editing

ACM Reference Format:

Gryka M., Terry, M. and Brostow, G.J. 2014. Learning to Remove Soft Shadows. ACM Trans. Graph. XX, X, Article XXX (Month 2014), 15 pages.

1. INTRODUCTION

Smart image editing algorithms are increasingly needed as non-experts demand more post-processing control over their photographs. Across the wide range of techniques available in commercial tools and research prototypes developed by the graphics community, soft shadow manipulation stands out as a severely under-explored, but important problem. Since shadows are a key cue for perceiving shape, curvature, and height [Kennedy 1974], countless Internet tutorials attempt to teach users how to separate out a shadow layer using tedious masking and thresholding operations. These manual techniques are employed in part because many real scenes have soft shadows, and most shadow detection and removal algorithms developed thus far address mainly hard shadows. Because soft shadows have been shown to be correlated with people's perception that an image is real [Rademacher et al. 2001], easier methods for extracting soft shadows are required.

We present a data-driven shadow removal method that is pre-trained offline and can deal with shadows of widely varying penumbra widths (*i.e.* where there is no clear boundary between the shadowed and unshadowed region). In contrast to previous work, our technique does not assume the existence of a specific model for the umbra, and processes the entire shadow with a unified framework while still giving users full control over which region to modify. We can deal with complex situations that were previously impossible, such as when the entire shadow is essentially penumbra, as is often the case (*e.g.* with shadows of leaves cast on the ground). Our technique requires user interaction only to roughly indicate which area of the image should be modified. The system then initializes and applies our model. Once the shadow matte is computed, the user can interactively manipulate it, or the rest of the image, using our simple interface.

Our regression model is trained through supervised learning to cope with our underconstrained problem: given a shadowed RGB image I_s, we aim to find a corresponding shadow matte I_m and the unshadowed image I_u that satisfy $I_s = I_u \circ I_m$ ($\circ$ is an element-wise product). Similar decompositions are explored in the intrinsic images domain [Land and McCann 1971], but we compute I_m to ignore both reflectance and shading, and only focus on cast shadows. Also, rather than aiming for physical accuracy, our practical objective is to produce a convincing-looking I_u as measured subjectively.

In a user study comprising hundreds of rankings and assessments, our technique was found to be significantly more likely to remove soft shadows successfully than the competing methods by Guo *et al*. [2012], Arbel and Hel-Or [2011] and Mohan *et al*. [2007] (Figure 1 shows selected results of our method). Additionally, when shown together with results produced by these other methods, our results were most often chosen as the most natural-looking (please refer to Section 7.5 for more details).

Our specific contributions are:

(1) A regression model that learns the relationship between shadowed image regions and their shadow mattes.

(2) A system that leverages existing inpainting and adapts large-scale regularization to our graph of suggested matte patches, producing results that compare favorably with the alternatives.

DOI 10.1145/XXXXXXX.YYYYYYY
http://doi.acm.org/10.1145/XXXXXXX.YYYYYYY

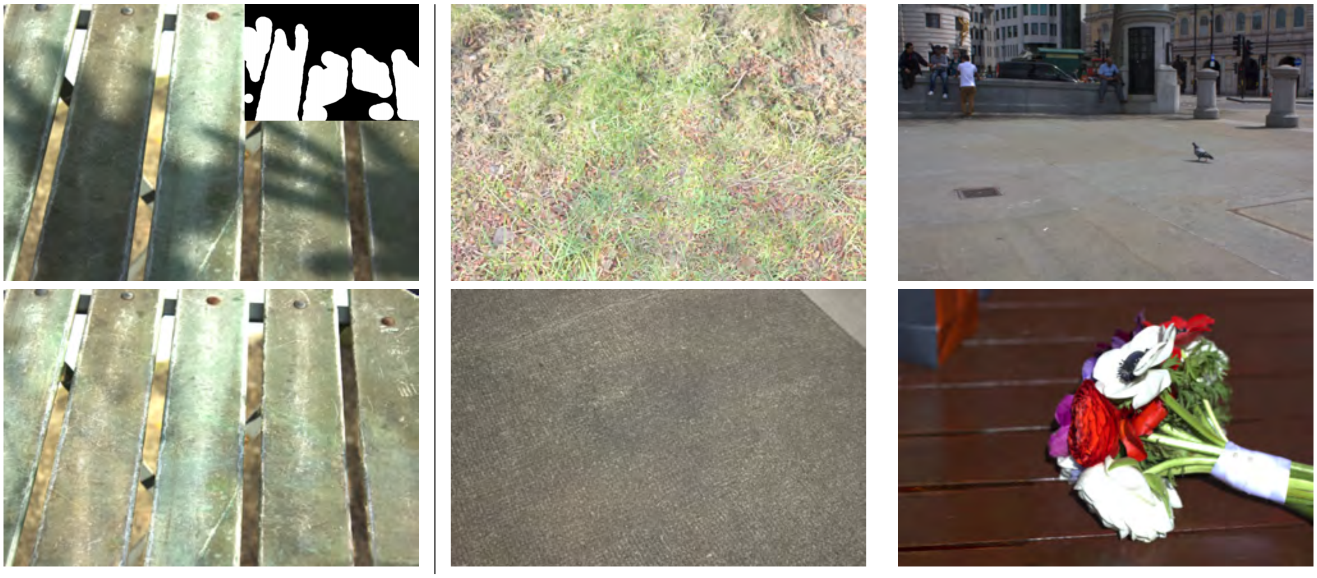


Fig. 1: The first column shows the input shadowed image (top) with a user-provided coarse shadow mask (inset) as well as the unshadowed image (below) produced by our method. The four images on the right present different unshadowed results for which the corresponding inputs can be seen in Figure 9. This technique could be used *e.g.* as a pre-processing step for texture extraction algorithms such as [Lockerman et al. 2013].

(3) Data: a large-scale dataset of real soft shadow test photographs as well as a system for generating countless training examples of scenes with both soft and hard shadows.

For easy replication, we will make the code available both for the method and for the user study experiments.

2. SYSTEM OVERVIEW

To use our system, the user first paints the region of the image containing the shadow they wish to modify. This masked region is then processed automatically, as follows. First, the input image is divided into non-overlapping 16×16 patches, and for each patch a_i a descriptive feature vector $f(a_i)$ is computed. Next, our pre-trained regressor maps each feature vector to m_i, a distribution of possible shadow mattes for that patch. A Markov Random Field (MRF) on the grid of shadow matte patches is regularized to generate the maximum a posteriori shadow matte image I_m for the red channel. A final optimization computes the corresponding shadow mattes for the green and blue channels, also yielding the unshadowed image I_u. With the shadow removed, our interface then allows the user to place a new shadow derived from the original shadow matte. This shadow matte can be translated, scaled, and distorted as desired to allow the user to create a new shadow in the image, or use the resulting layers for compositing and other creative tasks.

To create a regressor mapping intensity patches to shadow mattes, we have generated a large set of synthetic shadowed-unshadowed image pairs and fed them to a custom Multivariate Regression Random Forest. Our customizations allow us to take advantage of both parametric and non-parametric representations of shadow mattes without assuming specific penumbra models. Please refer to Figure 2 for the system diagram.

3. RELATED WORK

Most of the previous shadow removal work has focused on hard or nearly hard shadows. In contrast, in this work, we focus on soft shadows, mostly ignoring the specific case of hard shadows.

Intrinsic Images algorithms, as defined by Barrow and Tenenbaum [1978], separate images into the intrinsic components of reflectance and shading. This information can be used to aid other image manipulation techniques, scene understanding, *etc*. While much progress has been made in this space, many open problems remain. The work reviewed here describes approximate solutions that provide good results in specific cases. However, in general, this class of algorithms is not well equipped for dealing with cast shadows as we show in Section 7.2.

One approach to solving such under-constrained problems is to incorporate higher-level reasoning or additional data into the pipeline. For instance, Sinha and Adelson [1993] showed how to differentiate reflectance from illumination discontinuities in the world of painted polyhedra, which improved on previous approaches based on the Retinex theory [Land and McCann 1971]. Work by Laffont *et al.* [2013] used a multi-view stereo reconstruction to estimate intrinsic images by combining 3D information with image propagation methods, while [Weiss 2001] used multiple images of the same object under varying lighting conditions and a prior based on statistics of natural images to obtain convincing reflectance and shading separation.

Similarly in [Boyadzhiev et al. 2013], multiple images of the same scene with different illuminations were used to enable rich image relighting operations. Interestingly, this technique allows softening of lights by blending multiple images, while our approach performs image-space operations for control over sharpness. Bousseau *et al.* [2009] get their additional data from the user, who is asked to mark scribbles throughout the image on areas of constant reflectance and constant shading. Our user-input simply indicates where to operate, and does not require an understanding of which areas have constant reflectance.

Tappen *et al*. [2005] leveraged machine learning by first classifying each gradient in the image as either shading or reflectance, and then employing Generalized Belief Propagation to extend areas of high confidence to more ambiguous regions. Our approach is related in that we also use supervised learning followed by a regularization of a Markov Random Field. What makes our solution unique is a heavily customized learning algorithm and the ability to deal with hundreds of labels at each site.

Recently, Barron and Malik [2012] used a set of priors over reflectance and illumination combined with a novel multi-scale optimization to obtain results on the MIT Intrinsic Images dataset [Grosse et al. 2009], outperforming other methods by 60%, while also recovering shape and illumination. While this method works very well on images of single objects, we found in our experiments that its results are not as reliable when faced with complex scenes and cast shadows.

Image Matting provides another type of decomposition, and can be used to separate foreground and background objects. In principle, this formulation could be used to separate soft shadows as Guo *et al*. [2012] do when using a method by Levin *et al*. [2008] to matte out small penumbra regions. Wang *et al*. [2007] presented an intuitive brush interface combined with a fast algorithm to interactively matte out fuzzy objects. The challenge with using these techniques on noticeably soft shadows lies in specifying the correct affinity function to optimize. Our method effectively learns such a shadow-specific function from the data. Additionally, our users specify only a coarse binary mask, rather than a trimap.

While [Chuang et al. 2003] presented an effective method for shadow matting and compositing, they required much more input and did not tackle the challenge of wide penumbrae.

Inpainting is a technique that fills in missing image regions. This field has matured in recent years to the point of implementations being available in commercial tools. While useful in many cases, it is not a perfect solution to the problem of shadow removal as it completely discards potentially valuable information. Consequently, it often fails to reconstruct structure (see Figure 3). It does, however, often produce visually convincing results, and we exploit it to obtain a rough initial guess to guide our algorithm.

Inpainting methods need a source of data that can be used to fill in the missing parts. They can be seen as two categories based on where this data is taken from: a) bootstrapping algorithms that use the remainder of the image to be modified such as [Criminisi et al. 2003], [Barnes et al. 2009], [Pritch et al. 2009], and b) methods that rely on previously created datasets such as [Hays and Efros 2007]. Algorithms in the former category are appealing since one does not need to worry about creating an extensive training set. Yet, in practice, it is often difficult to make them scale to general scenarios. See Section 9 for an analogous extension of our method.

Both [Criminisi et al. 2003] and [Barnes et al. 2009] fill in the missing regions by finding patches in the rest of the image that “fit into” the hole. In both cases, care has to be taken to propagate the structure correctly into the missing parts. While Criminisi *et al.* achieve this in an automatic way by searching for matches along isophote lines, Barnes *et al*. and Sun *et al*. opt for user guidance to indicate structure lines crossing the hole, and thus manually constrain the search space. While not originally used for inpainting, a robust approach for finding non-rigid correspondences was shown in [HaCohen et al. 2011].

Shadow Removal [Finlayson et al. 2009] proposed a method of detecting shadows by recovering a 1-dimensional illumination invariant image by entropy minimization. Given this, they were able to discriminate between shadow and non-shadow edges in the original image and subsequently perform gradient domain operations for unshadowing. The process forces image derivatives at shadow edges to 0, which works well with hard shadows, but produces wide, noticeable bands of missing texture when applied to wide penumbrae.

[Shor and Lischinski 2008] tackled shadow detection and introduced a removal scheme using image pyramid-based processing. They deal with non-trivial umbras by compensating for the occluder obstructing ambient light. Their method is not, however, able to deal with complex shadow intensity surfaces such as leaves or shadows without any umbra. The estimation of non-uniform inner shadow surfaces is only done at the coarsest pyramid level, and so only takes into account large-scale variations. Further, it is not clear how to compute the “strips” used for parameter estimation in the case of complex shadows. Our approach is more generic, treating the entire shadow as potentially varying, and not limiting the variations to the coarsest scale. Further, their method is not well equipped to entirely deal with penumbrae, and inpainting around the shadow edges is still necessary to avoid artifacts.

[Mohan et al. 2007] proposed a method for removing as well as modifying soft shadows. They model the penumbra by fitting a piecewise quadratic model to the image intensity in user-marked areas, therefore separating texture variations from illumination changes. This enables them to work in the gradient domain and reintegrate the image after recognizing and removing gradients caused by shadows. The system first asks the user for input in the form of a shadow outline specified by control points along the shadow boundary. Additionally, the user is required to initialize the width of the penumbra as well as the shadow amplitude for each of the color channels separately. The algorithm then performs iterative optimization by fitting the aforementioned fall-off model to either vertical or horizontal intensity slices through the penumbra, updating the parameters and optimizing again. This procedure is repeated for each segment of the shadow boundary separately (the number of boundary segments is also user-specified) and values between the boundary points are obtained by linear interpolation. The method produces convincing results, but is labor- and time-intensive for the user and requires a significant amount of computation time. In our tests it took over 40 minutes per image, of which 10 were spent providing the input.

After the penumbra parameters are optimized, the user has control over which gradients to remove from the image. Due to the nature of gradient domain operations, this method often modifies the entire image noticeably, rather than just removing the shadow.

Finally, this technique operates under two assumptions that do not always hold: that penumbrae can be modeled accurately using a sigmoid-shaped curve and that an umbra region exists at all.

[Wu et al. 2007] presented a matting approach to natural shadow removal. In contrast to standard matting methods, however, they treat the shadow matte as a pixel-wise fractional multiplier of the unshadowed image. While their method works well on many shadows, it requires noticeably more user input than our technique: a quad map signifying the “definitely in shadow”, “penumbra”, “definitely out of shadow” as well as “excluded” regions. Additionally, their matting formulation requires a distance function to be optimized. While they presented one that performs well on many natural shadows, problems can occur in some scenarios (such as significant noise) since the matting cost function is not tuned for these. In contrast, our technique can theoretically adapt to new situations provided enough training data.

[Arbel and Hel-Or 2011] presented a critical survey of recent shadow removal literature and argued that matting approaches such

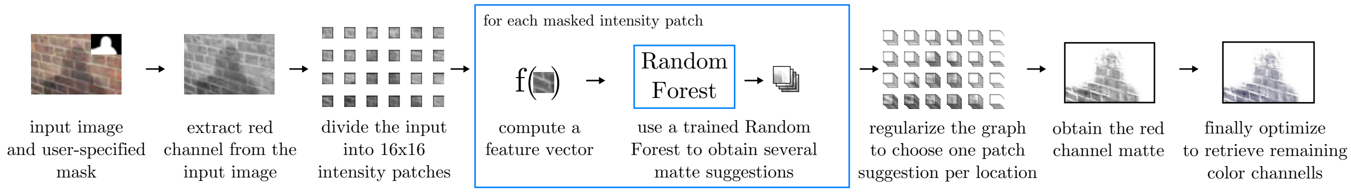

for each masked intensity patch
Random Forest
input image and user-specified mask
extract red channel from the input image
divide the input into 16x16 intensity patches
compute a feature vector
use a trained Random Forest to obtain several matte suggestions
regularize the graph to choose one patch suggestion per location
obtain the red channel matte
finally optimize to retrieve remaining color channells

Fig. 2: System overview. Obtaining the final matte (far right) allows us to remove, as well as modify, soft shadows.

as [Wu et al. 2007] are not an optimal way to pose shadow removal. Instead, they fit an intensity plane to the shadow-free surface and thus obtain an approximate unshadowed result and separate out the shadow. To recover the lost texture in the penumbra after fitting the intensity surface, they perform directional smoothing on the shadow matte in the direction perpendicular to the shadow edge. They demonstrated results on penumbrae up to 15 pixels wide.

Another method to detect as well as remove shadows was described by Guo *et al*. [2012]. The whole detection is region-based and is performed by running an SVM classifier followed by Graph-Cuts on the regions of the image to decide whether they are in shadow or not, based on their appearance and relationship to others. Once every pixel in the image is classified as either shadowed or shadow-free, constraints for the matting step are built by skeletonizing the obtained shadow mask. Next, the matting method by Levin *et al*. [2008] is used to obtain penumbra reconstruction.

As noted previously, matting-based approaches are problematic for shadow removal in that they use a heuristic affinity function at the core of their energy minimization. Since engineering a shadow-specific affinity function might be challenging, our method effectively learns it from the data. Another problem, as we found in our evaluation (please see the supplementary material for examples), is that the method by Guo *et al*. is not well suited for user input since it can be difficult to specify which shadows should be removed. In the cases of wider penumbrae, the matting often “misses” the subtle gradients and does not remove the shadow at all, even with a user-provided shadow mask. While this problem could potentially be addressed by changing how the shadow mask is used to build matting constraints, the authors reported experimenting with a few (*e.g*. treating the eroded mask as definitely-in-shadow region) and choosing the most successful one.

4. LEARNING AND INFERENCE

While shadow mattes are generally unique, our hypothesis is that they can be constructed from a finite set of patches tiled next to each other. We exploit this property and perform learning and inference on a patch-by-patch basis. A similar approach to this part of our pipeline was recently used by Tang *et al*. [2014] to tackle image dehazing. We considered alternatives to the machine learning approach we present here, such as fitting sigmoid functions to model the soft shadow fall-off. Even with complicated heuristics, these results were unconvincing. Those parametric models needed many degrees of freedom with hard-to-summarize constraints and relationships to adequately approximate different ground-truth mattes. Our learning-based approach focuses on the input/output dimensions that correlate most, and is, broadly speaking, a kind of supervised, non-Euclidean, nearest-neighbor model.

We have empirically determined that patches of 16×16 pixels work well for our purposes. Further, we assume that color channels of a shadow matte are related to each other by a scaling factor. Specifically, we assume it is possible to reconstruct the green- and blue-channel mattes given the red-channel matte and the inferred scaling factors σ_g and σ_b (see Section 5). We have chosen the red channel for performing learning and inference, and to reconstruct the color result in the post-processing step. While this splits the problem into two optimizations, it reduces the parameter space that must be learned from data.

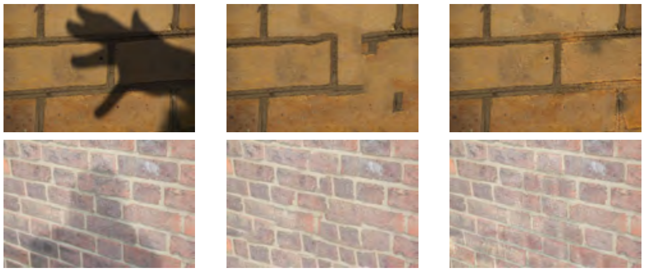


Fig. 3: Example initial guesses. From the input image (left) we use inpainting to obtain an initial guess for the unshadowed image (middle). That in turn yields an initial-guess matte that forms part of our feature vector and aids regularization. The right column shows the output of our algorithm: an unshadowed image respecting the texture present in the shadow. Note that inpainting alone is unable to recover structure in the above cases.

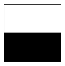
4.1 Preprocessing

Using an off-the-shelf inpainting method by Barnes *et al*. [2009], we first replace the user-specified shadow region completely with a plausible combination of pixels from the rest of the image as shown in the middle column in Figure 3. We then apply Gaussian blur in the inpainted region and divide the input image by it to obtain the first approximation to the matte (we constrain any resulting pixel intensities to the range $[0, 1]$). The blurring step is necessary to both minimize the impact of slight inpainting errors, and to avoid producing spurious frequencies in the image after division. We have examined alternatives to this initialization method, including guided inpainting, but were unable to find one producing more optimal results (please see Section 8 for more details).

For training, our regressor expects as input a small intensity patch a_i (or rather a feature vector $f(a_i)$ computed over this patch), as well as the corresponding ground truth matte patch m_i as the label, so we need to extract these pairs from our large set of training images. From each image we could potentially extract many thousands of such pairs. To avoid redundancy, we chose to sub-sample each training image to extract J training pairs overall (in all our experiments we have used a training set of size $J = 500\,000$). Additionally, we bias our sampling so that in each training image, half of the samples come from the penumbra region only and half are sampled evenly across the entire shadow (which also includes the penumbra). This avoids over-sampling of uninteresting regions of flat shadow profile and provides greater variety in the training set.

Alignment We observe that many patches, though seemingly different in their raw form, can ultimately appear very similar after aligning with an appropriate Euclidean transform. This allows us to perform inference on rotationally-invariant data, exploiting structure present in our labels. While a similar effect could be achieved by increasing the amount of training data used, we achieve equivalent results without noticeably increasing the training time.

For each intensity patch that we wish to include in our training set, we search through a limited set of rotations and translations to find one that results in the smallest distance to the template patch. As the template, we have chosen a simple black-and-white square, as shown on the right. We then apply this transformation to both the intensity and the matte patches. At test time, we perform the same procedure before computing features and, after obtaining the estimated label, apply the inverse transformation to it (see Figure 4).



4.2 Learning

For each patch, we form a column *feature* vector by concatenating the following features (we have chosen them based on intuitive interpretations of their relationships to shadow profiles as well as empirical evaluations).

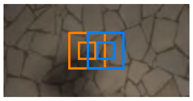
(1) **Distance from the edge** of the user-masked region normalized so that the values lies in range $[0.0, 1.0]$. The distance from the edge of the shadow often indicates the "flatness" of the profile, since shadow matte patches far from the edges are likely to contain the most uniform intensities.

(2) **Predicted matte** for this patch from the initial guess. While the initialization is often wrong when used directly, in many situations it provides a hint as to how the image should look without the shadow.

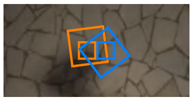
(3) **Vertical and horizontal gradients** (finite differences) of the patch, which convey information about the slope of the shadow.

(4) **Pixel intensity values of the patch** in the range $[0.0, 1.0]$ shifted in the intensity domain so that their mean falls at 0.5. The intensity values are normalized, since they are not directly correlated with the matte (given a dark shadowed intensity patch it is impossible to determine whether it is a dark shadow on a bright background, or a bright shadow on a dark background). Therefore we give the inference algorithm processed features that are likely to contribute to the decision (*i.e.* indicating the slope of the shadow), but without confusing differences in absolute intensity. While this information is theoretically redundant given the gradients, it provides the Random Forest with more choices to use for discrimination without hurting its performance.

Our *label* vector contains the pixel values from the shadow matte. Even though at test time we obtain suggestions for each patch in the 16×16 grid in the shadow region (just the inner squares in the inset figure), both our features and labels are computed over a larger 32×32 window (outer squares). This serves two purposes: to enable smoother results by providing more context to the features, and to aid the alignment and realignment described in Section 4.1.



Before alignment



After alignment

We have chosen to use Random Forest as the inference mechanism both because of its versatility and a widespread use in the literature (e.g. [Reynolds et al. 2011], [Shotton et al. 2012], [Criminisi et al. 2013]). A brief introduction to the traditional Random Forest algorithm below is followed by our modifications and the reasons for introducing them.

Given a standard supervised-training data set of input/output pairs (*i.e.* the feature vectors and the corresponding label vectors), we can use Random Forests to create a set of decision trees that will allow us to predict the label vectors for new, yet unseen feature vectors (provided they resemble the statistics of the training examples). Each of the separate decision trees is imperfect, usually only trained on a random subset of the training data (a process called bagging) and with no guarantees about a globally optimal inference due to the nature of the training process described below. However, averaging the responses of a collection of trees (*i.e.* a "forest"), often results in accurate predictions.

Given a "bagged" set of training data (that is, a subset of all available feature/label pairs), a decision tree is trained as follows. First, we define an impurity, or entropy, measure for a collection of labels, a value that is low when the labels at a node are homogeneous, and high when the labels are different to each other. Then, a binary tree is generated by splitting the available data along the dimensions of the *feature* vector in a way that minimizes the impurity of the split collections. Alternatively, this can be posed as maximizing the information gain—the difference between the original impurity and the sum of child impurities. The generation process starts at the root node, with all the data available to a given tree. It tests a number of random splits along the feature dimensions and chooses the one that results in the largest information gain (an example split could test if the 7th entry in a feature vector is greater than 0.3). It then creates two child nodes, left and right, and pushes the split data into them. The same process is repeated at each node until a stopping criterion is reached (usually a predefined tree depth or a number of samples).

After training, the forest can be used for predicting the labels for new feature vectors. The feature vector in question is "pushed" down each tree depending on the values of individual features and the node thresholds. After arriving at a leaf node, the mean label of all the training samples that landed at this node is taken as the answer of this tree. Finally, answers of all trees are averaged to get a more robust prediction.

We use a modified version of Multivariate Regression Random Forests in this work. While Random Forests in general have been well-explored already, their use for practical multivariate regression has been limited [Criminisi et al. 2013]. One of the challenges lies in computing node impurity—in classification, this can be done easily by counting samples belonging to each class, whereas in regression, one needs to evaluate the probability density function, which can be costly in high dimensions.

Our labels lie in $\mathbb{R}^{2N \times 2N = 1024}$ (where $N = 16$ and $2N$ comes from the fact that we store areas larger than the original patches), so it would not be feasible to directly regress entire patches. However, we observe that they can be effectively represented in lower-dimensional space, since penumbrae generally do not exhibit many high-frequency changes. Moreover, we only need the representation to be accurate enough to cluster similar labels together—we do not lose detail in the final answer because of the non-parametric nature of our inference method described below (in short we build the forest based on the low-dimensional representation, but retrieve final labels in the original, high-dimensional, space). Therefore, we use PCA to project our labels into $\mathbb{R}^{D=4}$, which provides a good balance between the degrees of freedom necessary to discriminate

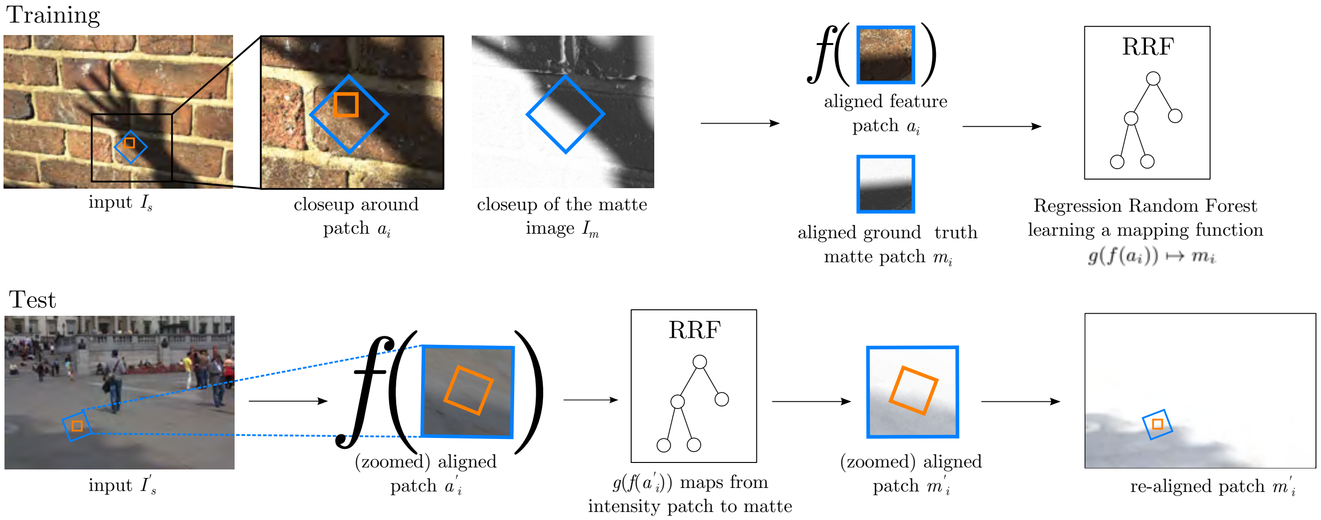

Training
input I_s
closeup around patch a_i
closeup of the matte image I_m
$f($ $)$
aligned feature patch a_i
aligned ground truth matte patch m_i
RRF
Regression Random Forest learning a mapping function $g(f(a_i)) \mapsto m_i$
Test
input I'_s
$f($ $)$
(zoomed) aligned patch a'_i
RRF
$g(f(a'_i))$ maps from intensity patch to matte
(zoomed) aligned patch m'_i
re-aligned patch m'_i

Fig. 4: Patch alignment. **At training time** (top) we look at each patch in the shadow area (orange square) and find a small-magnitude Euclidean transform to bring the patch as close as possible to the template. We then cut out the expanded patch a_i (blue square) and use it to compute features $f(a_i)$. Additionally, we cut out the same location from the ground truth matte to obtain the label m_i. Finally, we feed these feature/label pairs to the Regression Random Forest to learn a mapping function $g(f(a_i)) \mapsto m_i$. **At test time** (bottom), we also start with a grid patch (orange square) and, after finding an optimal offset, cut out the surrounding area a'_i (blue square) from which we compute the features $f(a'_i)$. After pushing these through the forest we obtain a label m'_i that we re-align and crop to paste into the original position in the output image (small, orange square).

between patches, and computational complexity while evaluating impurities. Specifically, at each tree node n, we assume a Gaussian distribution for the labels of all samples $\mathcal{S}_n$ falling into it, and evaluate impurity

$$H_n = \log(\det \Sigma_{\mathcal{S}_n}) \tag{1}$$

by taking the log of the determinant of its covariance matrix $\Sigma_{\mathcal{S}_n}$ following Criminisi *et al*. This allows us to define the information gain

$$G_n = H_n - \sum_{c \in \{l,r\}} (|\mathcal{S}_c|/|\mathcal{S}_n|) H_c \tag{2}$$

which we aim to maximize at each split node while building the trees. We weight the information gain by the proportion of samples falling into each child node (l and r) to encourage more balanced trees as in [Breiman et al. 1984].

We set the minimum sample count at a node to $K = 2D$ and grow our trees as deeply as necessary until we do not have enough samples to split. In principle, K could be as low as D (number of samples needed to compute a D-dimensional covariance matrix). However, in practice, we find that the samples are often not linearly independent, leading to degeneracies. After a leaf node is reached, instead of building a typical parametric distribution of all its labels, we save the indices of training samples falling into this node, allowing us to perform inference as described in the next section.

4.3 Inference

Our inference step acts as a constraint on the initial guess—we want to "explain" the initial-guess mattes as well as possible using samples from our training set, but only those suggested by the forest as relevant.

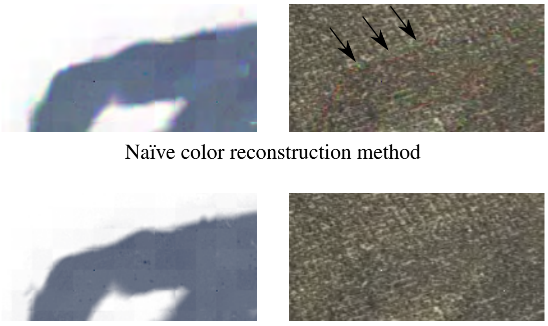
At test time, we compute the feature vector for each patch as before and, after pushing it through each tree, arrive at a leaf node. From here, instead of looking at the label distribution, we simply get the indices of training samples that fell into this leaf. Consequently, we obtain L label suggestions, where $L \geq TK$ and the number of trees in the forest $T = 25$. We do this for each patch in the 16×16 grid in the shadow region and arrive at an optimization problem: for each image patch in our matte we want to choose one of L labels that agrees both with our initial guess and any available neighbors.

In summary, the changes we have made to the original RRF algorithm are:

(1) Using two representations of labels: low-dimensional used to evaluate node impurity and build the forest, and high-dimensional used for retrieving the labels at test time. This is motivated by computational constraints and enabled by the non-parametric treatment of labels.

(2) Non-parametric treatment of the labels to avoid over-smoothing. Instead of one mean answer in label-space, we get a distribution of samples from the data, including extremes, which we want to preserve.

(3) Treating the inference algorithm as a step in the pipeline, rather than the entire pipeline. We only get an intermediate result from the forest (several matte patch suggestions for each patch) and use regularization later on to extract the final answer. As above, this allows us to benefit from relatively limited amounts of training data (compared to the number of theoretically possible labels in $256^{16 \times 16}$-dimensional space), without averaging out unusual shadow profiles.



Naïve color reconstruction method

Our method

Fig. 5: Treating each channel independently results in inconsistent shadow mattes (top left) that manifest themselves with colorful splotches in the unshadowed output image I_u (top right). Our method assumes that the three color channels are dependent—it only regresses one of them and reconstructs the other two as explained in Section 5. Please see the digital version for faithful a color representation.

4.4 Regularization

Finding a more specific combination of matte patches is not trivial due to the nature of our labels and the fact that there might be different numbers of label suggestions available at each node. Averaging all the candidate patches at each location would not be optimal, since any unusual shadow profiles would be lost. On the other hand, choosing best-fitting patches greedily and then trying to smooth out the edges between them would a) be extremely difficult to do for small patches that are likely to be incompatible and b) introduce an implicit, non-data-driven, shadow model in smoothed regions. Instead, at each location, we choose the best patch by regularizing the entire graph with the TRW-S message passing algorithm [Kolmogorov 2006]. We use the energy function

$$E = \sum_{i \in \mathcal{I}} \omega(m_i) + \lambda \sum_{i,j \in \mathcal{N}} \psi(m_i, m_j), \tag{3}$$

where $\mathcal{I}$ is the set of nodes in the regularization graph (*i.e.* all the masked image patches) and $\mathcal{N}$ denotes the set of neighboring nodes in a 4-connected neighborhood. The unary cost $\omega(m_i)$ is the SSD distance from the patch m_i to the corresponding area in the initial guess, the pairwise cost $\psi(m_i, m_j)$ is the compatibility of patch m_i to m_j, and λ is the pairwise weight ($\lambda = 1$ in all our experiments). We define the patch compatibility $\psi(m_i, m_j)$ as the sum of squared differences between adjoining rows (or columns) of these two patches:

$$\psi(m_i, m_j) = \begin{cases} SSD\Big(row_N(m_i), row_1(m_j)\Big), & \text{if } m_i \text{ is above } m_j \\ SSD\Big(col_N(m_i), col_1(m_j)\Big), & \text{if } m_i \text{ is to the right of } m_j \end{cases} \tag{4}$$

where $row_N(m_i)$ and $col_N(m_i)$ are the last row and column of patch m_i respectively. We also create boundary constraints to ensure that the shadow disappears outside of the user-selected region by forcing patches just outside of the user mask to constant 1.0 (meaning completely shadow-free).

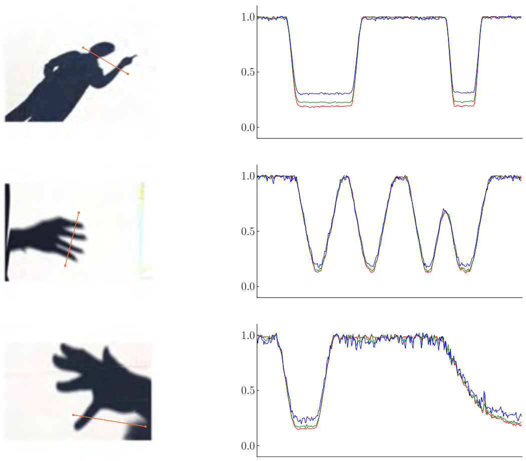


Fig. 6: Ground truth shadow matte profiles for RGB color channels in three real photographs. Note that, disregarding noise, all channels share the same basic "shape" and reach 1.0 (no-shadow zone). Our experiments indicate that acceptable green and blue mattes can usually be reconstructed by optimizing (5).

5. COLOR OPTIMIZATION

We could repeat the above procedure twice more to obtain the remaining green and blue channel mattes. Indeed, in theory, this would be the most general solution, assuming no relationship between different frequencies of light. In practice, however, we find that this relationship is quite strong, and providing an additional constraint to enforce it makes our method more robust. The top row in Figure 5 shows a shadow matte and the corresponding unshadowed image estimated by the naïve way, *i.e.* each channel separately. Note the splotches of different colors revealing where the mattes of different channels do not agree. The bottom row shows our default procedure, described here.

We assume that the surface of the shadow matte has the same shape in all three channels, but that it differs in magnitude as shown in Figure 6. For instance, while in outdoor scenes the shadow matte will be blueish, the red and green channels can be obtained by scaling the blue channel matte in the intensity domain so that areas with no shadow remain that way, but the overall depth of the shadow changes proportionately. This assumes that, while light sources can have different colors, they do not vary much spatially.

Relative to the estimated red channel shadow matte, we model each of the other channels with a single scale factor parameter, σ_g and σ_b respectively. To estimate them jointly, we discretize and search the 2D space to minimize the error function

$$E_{color}(\sigma_g, \sigma_b) = \log(\det(\Sigma_{\mathcal{R}})), \tag{5}$$

where $\Sigma_{\mathcal{R}}$ is the covariance of a three-column matrix $\mathcal{R}$ listing all the RGB triplets in the unshadowed image after applying that color matte. We constrain the search in each parameter to lie between 0.8 and 1.2 with discrete steps of 0.01. We find that the scaling factors σ_g and σ_b rarely exceed 1.1 and never fall below 1.0 in outdoor scenes.

The intuition behind this approach is that unshadowing an image should not significantly change the distribution of colors in it. Since introducing new colors would increase the entropy of $\Sigma_{\mathcal{R}}$, we use this measure to find scaling factors that minimize it.

For efficiency, we run this optimization on a version of the output image downsampled to 10% of its original size. This optimization serves to prevent our unshadowing method from introducing new colors into the images. The user can override these scale parameters with our shadow-editing interface (Section 7.1), but all our results are shown with the automatic adjustment unless specifically stated.

6. DATA GENERATION

To train our model, we need large amounts of data to capture a variety of scene configurations and shadow-receiving surfaces. Since it would be extremely difficult to capture a large enough set of real images, we follow previous works, such as [Mac Aodha et al. 2010] and [Tang et al. 2014], in training on a synthetically-generated training set and applying it to real data. We have carefully configured Maya with realistic lighting conditions to generate shadows cast onto various textures as illustrated in Figure 7. For each light-occluder-texture combination, we have rendered the image with and without shadow, implicitly obtaining the corresponding matte.

While shadows in the real world are cast by three-dimensional objects, for each shadow there also exists a 2D slice through the occluder that would produce identical results. Therefore, we have used automatically-segmented silhouettes of real objects from [Griffin et al. 2007] as occluders in our synthetic scenes (we have segmented them automatically by using [Boykov et al. 2001]). This has the advantage over using 3D models of providing realistic silhouettes, as long as the images used are easily segmentable. Additionally, a variety of real images[1] were applied as textures for the receiving surfaces.

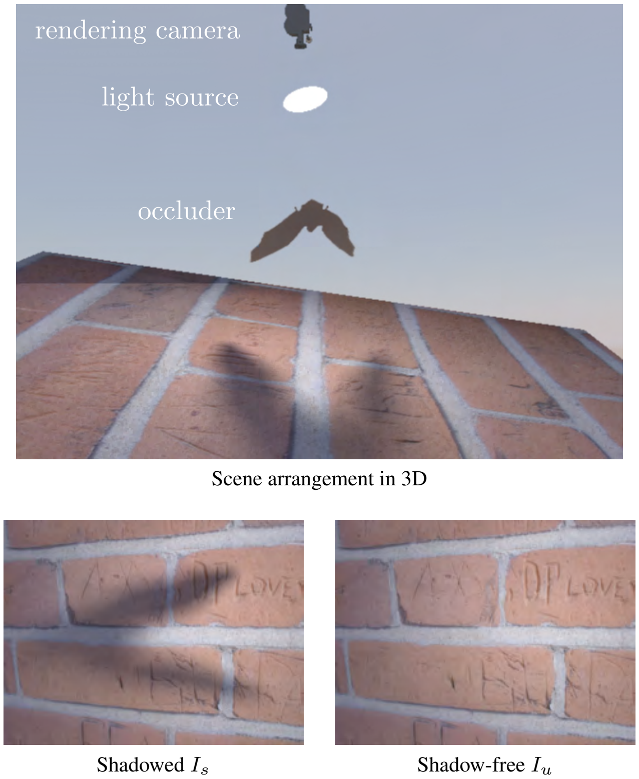
Finally, we varied light conditions in the scene by randomly choosing the shape and size of the light, its angle and distance from the ground, as well as the angles of the occluder and the ground plane.

In our experiments, we have trained the model on over 10,000 512×512 image pairs, choosing 50 patches from each image. Additionally, we have automatically generated binary shadow masks to only train on relevant regions. We have rendered all our images without gamma correction to make the relationship between image intensity and shadow attenuation easier to model. While this should mean that the shadow profiles we learn are slightly different than those observed in regular, non-linear images, we have not investigated this relationship, or the impact of real sensors and noise.

While our data generation method provides data diverse enough for our purposes, it is limited, containing only a single light source (in addition to a simulated sky and global illumination) and a single occluder in each image. Further, the shadows are always cast on textured planes, instead of more complex geometries. Finally, because we only learn single-channel shadow characteristics, no effort was made to include different light frequencies in the dataset.

7. EXPERIMENTS AND RESULTS

To capture real data for evaluation, we have used Canon 450D and 550D DSLR cameras to capture 14-bit-per-channel RAW, linear images. We provide this set of 137 photographs, 37 of which have the corresponding ground truth shadow-free images (and mattes), as a benchmark for future methods. The ground truth was obtained by placing the camera on a tripod and capturing two images: one with the shadow and one without by removing the shadow caster. For our experiments, we have converted the images to 8-bit-per-channel linear PNGs and, after processing, de-linearized them for display by applying gamma correction with $\gamma = 2.2$ (see Supplementary Material).



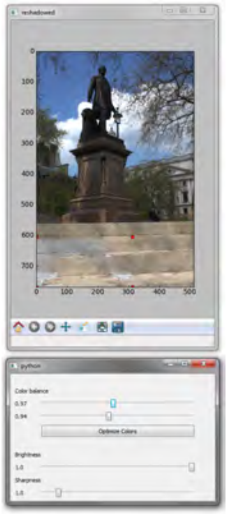
Scene arrangement in 3D

Shadowed I_s

Shadow-free I_u

Fig. 7: To generate the training data, we rendered 10,000 {shadowed, shadow-free} image pairs, each time varying the light source, the occluder, and the ground plane.

7.1 Shadow Editing



Color balance
0.97
0.94

Brightness
1.0

Sharpness
1.0

Knowing the shadow matte allows us to not only remove the selected shadow, but also enables a range of high-level image editing techniques. We have implemented a basic interface with four different alteration methods to give artists control over how the selected shadow looks: shape transformation, changing brightness and color, and sharpening the shadow. Color, brightness, and sharpness are adjusted using sliders, while direct manipulation controls enable a direct homography transform, allowing users to change the position and shape of the cast shadow. Please see the supplementary video for examples.

We can use the recovered shadow mattes to aid tasks such as compositing, texture extraction *etc*., which are normally challenging tasks requiring substantial manual work. Both the matte and the unshadowed image can be exported to any number of generic image editing tools for further processing.

[1] http://www.mayang.com/textures/

	Mean RMSE
Ours	**13.83**
Arbel and Hel-Or 2011	18.36
Guo at al. 2012	19.85
Guo at al. 2012 (automatic detection)	19.19

Table I. : RMSE between results of different shadow removal methods and the ground truth shadow-free images. While our scores best, our real aim is to convince subjects that the resulting images are unaltered.

Scene Name	Mean Pairwise RMSE
real22	8.35
real26	6.15
real138	14.55
real168	1.02
real249	1.58

Table II. : Differences between images unshadowed by different users. Each of the 5 scenes was unshadowed by 4 users. For each scene, RMS differences were computed between each image pair, and the mean of these errors is shown above. Please refer to the supplementary material to see all the user-provided masks and the resulting unshadowed images.

7.2 Visual comparison with other methods

We have evaluated our algorithm against other related techniques and display results in Figure 8. We have chosen the best results we were able to obtain for this image using the original implementations of [Mohan et al. 2007] and [Arbel and Hel-Or 2011], but since the user has some freedom in the use of their systems, we cannot exclude that a more skilled person could achieve better outcomes (note that for Mohan *et al.* we have downsampled the image by 50% to speed up processing). Please see the supplementary material for many more results.

While intrinsic image algorithms could be considered an alternative to shadow matting, it is important to note that they have different goals and it would not be fair to directly compare the two, so the examples are shown just for illustration. While shadow matting usually deals with cast shadows, intrinsic image techniques generally decompose images into reflectance and shading components where, in practice, shading mostly refers to attached shadows. Most of these techniques are poorly equipped to recognize cast shadows as illumination changes unless given access to additional data such as in [Weiss 2001].

Finally, the method presented by Finlayson *et al.* [2009], does not provide perfect illumination-invariant images, as shown in Figure 8. In the first image, while the ilumination differences are not visible, some reflectance-induced gradients were removed as well (flower patterns in the top part of the image). For a *different* input, in the image on the right, illumination differences are still visible.

Finally, binary masks and shadow mattes for sample images are presented in Figure 9.

7.3 Quantitative evaluation

Table I shows quantitative evaluation of our and related methods in terms of RMS error from ground truth (for this evaluation we have used all images, which were processed by all four methods), while Figure 10 shows pixel-wise differences to ground truth on a single example. Note that quantitative comparisons are not, in general, representative of perceptual differences and are included here only for completeness.

7.4 Impact of variations in the user input

Our method does not automatically detect shadows in the image, instead giving the user control over which regions should be modified. To establish how robust it is to variation in the user input, we have asked 4 users to create masks for 5 different scenes and ran the unshadowing algorithm for each input.

Each user was instructed to paint over the areas of the image containing the shadow and to prefer over- to under-selection for consistency with our assumptions (namely, that it is difficult to exactly determine boundaries of soft shadows and that our algorithm is only allowed to modify selected regions).

To properly investigate the impact of user input only, we have constrained the inpainting to be the same for each user. This is necessary, since the inpainting algorithm we use is non-deterministic and constitutes the main source of variation between runs. After having all the user-provided masks we have created a union of them (*i.e.* pixel-wise logical-`or`) and used it as the inpainting mask.

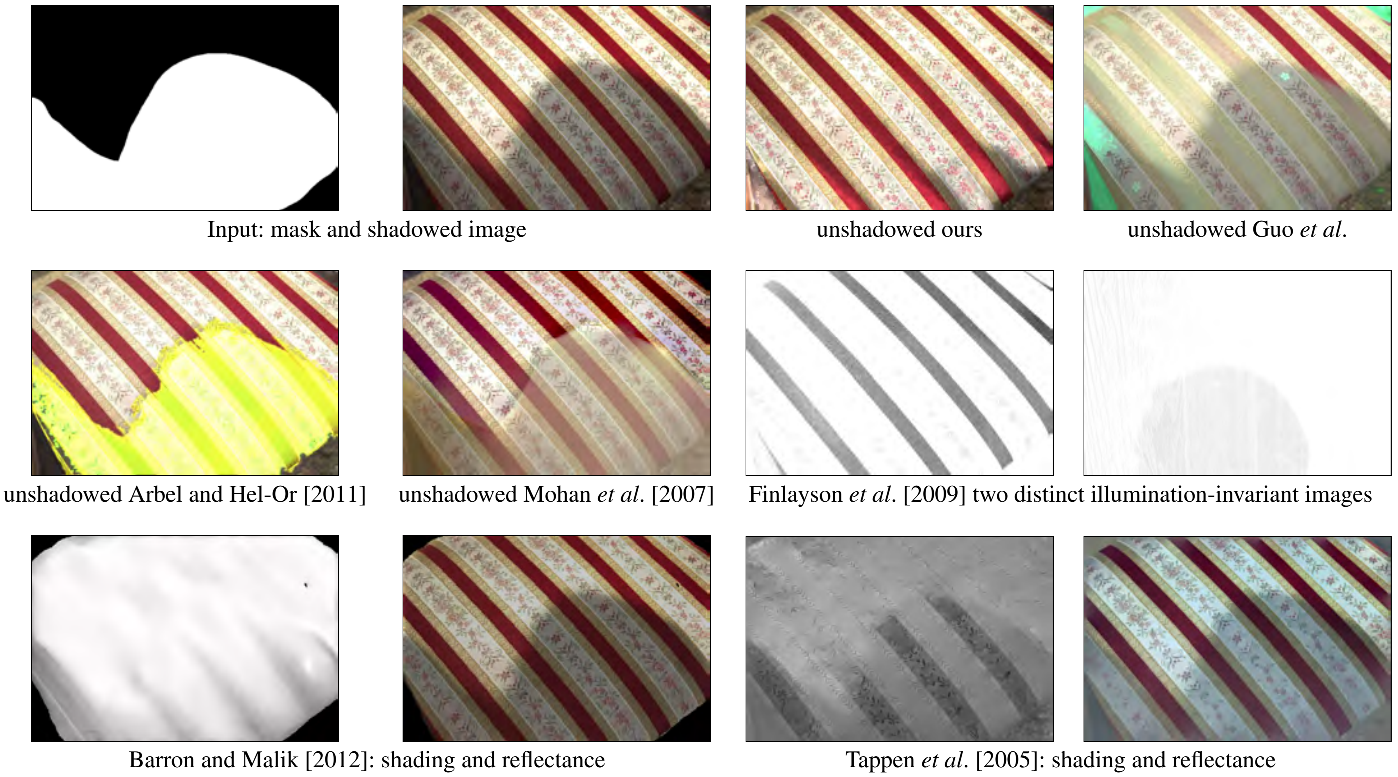
As Table II indicates, the final results are fairly robust to variance in user input. The largest differences are caused by users underestimating the extent of the shadow and thus not marking some regions for modification. We have included the different input mattes and corresponding unshadowed images in the supplementary material.

7.5 User Study

To understand how our results compare to those produced by prior work, specifically the methods of Guo *et al.* [2012] and Arbel and Hel-Or [2011], we conducted a user study similar in spirit to [Kopf et al. 2012]. We have modified the method of Guo *et al.* to sidestep the automatic shadow detection and instead use the same shadow mask that was given to our algorithm (though the results from Guo *et al.*'s unmodified version are included in the supplementary material). Please also note that the method of Arbel and Hel-Or required more input that was provided manually for each image.

We have assembled a set of 117 images for user evaluation by combining a subset (soft shadows only) of the dataset released by Guo *et al.* with a subset of our own images: 65 from Guo's and 52 images from our dataset. The complete set of images used can be found in the supplementary material. Our study consisted of two phases: a) a series of *ranking* tasks in which participants ordered a set of two or three images, and b) a series of *evaluation* tasks in which participants indicated the success of shadow removal on a particular image using a 4-point Likert scale. Both phases started with a tutorial example and displayed instructions on the right side of the screen throughout all trials. Additionally, in each phase, we have asked the participants to estimate their confidence in their choice on a 3-point Likert scale.

In the ranking phase, participants were shown 15 random image tuples (from a set of 117), where each tuple consisted of images of the same scene modified with one of three methods: ours, Guo *et al.*'s and Arbel and Hel-Or's. Using a drag-and-drop interface, participants were asked to order the images according to how natural they looked (*i.e.* from most to least natural). The aim of this part of the study was to establish how believable the produced results were, without the participants being aware that any shadows were removed. Roughly half of the tuples showed results from each of the three methods, and half paired our result with one of either Guo *et al.* or Arbel and Hel-Or. For all participants, the order of the tuples was randomly chosen, along with the order of the images within the tuple.



Input: mask and shadowed image

unshadowed ours

unshadowed Guo *et al.*

unshadowed Arbel and Hel-Or [2011]

unshadowed Mohan *et al.* [2007]

Finlayson *et al.* [2009] two distinct illumination-invariant images

Barron and Malik [2012]: shading and reflectance

Tappen *et al.* [2005]: shading and reflectance

Fig. 8: Comparison with other methods on the “real151” input image. Note that both Barron and Malik and Tappen *et al.* perform slightly different image decomposition: rather than matting cast shadows, they extract shading and reflectance components. (The second illumination-invariant result for Finlayson *et al.* comes from the shadowed image in Figure 9.)

In the second phase, 15 images were randomly drawn for each user from a pool of 282 images. These were the same set of images as in the first phase, however, now each image was shown separately rather than in a tuple. Of these images, 118 were processed using our technique, 114 were processed using Guo *et al.*’s, and 50 using Arbel and Hel-Or’s. The set of 15 images was randomly chosen subject to the constraint that the images seen during the first phase could not be used in the second. Each of these images was augmented with a bright arrow pointing to the centroid point of the shadow that was to be removed, and participants were asked to assign a score $\{1, 4\}$ based on how successfully they thought the shadow was removed. Low values corresponded to cases where the shadow was not removed or where the removal introduced artifacts, while high scores indicated successful shadow removal and no visible defects. The centroid was computed automatically by finding a point on the skeleton of the binary shadow mask closest to the mean position of the masked pixels. The order of the two phases (ranking then evaluation) was fixed for all participants.

Results The study was deployed on a website. We recruited 51 participants through email lists, word-of-mouth, and Facebook. Individuals could participate by visiting a link using their own computing device. Of the 51 participants, 39 completed the whole experiment, 7 quit before finishing the first phase, and 5 quit during phase two. Because of the randomized design of the study, we include all participant data, including data from participants who did not complete the entire study. Additionally, 28 people visited the experiment, but did not answer any questions.

We analyze our study data using Bayesian data analysis methods [Kruschke 2011]. Unless otherwise stated, reported results represent the posterior mean, and the confidence interval (CI) represents the range encompassing 95% of the posterior probability.

Participants rated a total of 694 image tuples in the ranking phase and analyzed 605 images in the evaluation phase. In the ranking phase, we calculate the posterior probability of each method being ranked first (*i.e.* appearing the most natural). We model this as a Bernoulli random variable with a uniform Beta prior. As shown in Figure 11, results produced by our method were significantly more likely to be ranked first than the competing methods.

In the second phase, participants ranked the success of shadow removal with a score $\{1, 4\}$ for each image. Figure 12 shows the normalized histograms of scores assigned to results produced with each of the three methods. As can be seen, our method obtained high scores in the evaluation task more often than the other methods. Additionally, we have evaluated how likely each method was to unshadow images perfectly (we define “perfect” shadow removal as one with mean evaluation score across all participants $\mu_{eval} > 3.5$). Figure 13 (left) shows the posterior probabilities for each method to produce a perfect result (as before we have modeled this using a Bernoulli random variable with a uniform Beta prior). The results show that our algorithm is significantly more likely than others to succeed in this scenario.

We have also characterized the results by considering the data from both user study phases together. The right part of Figure 13

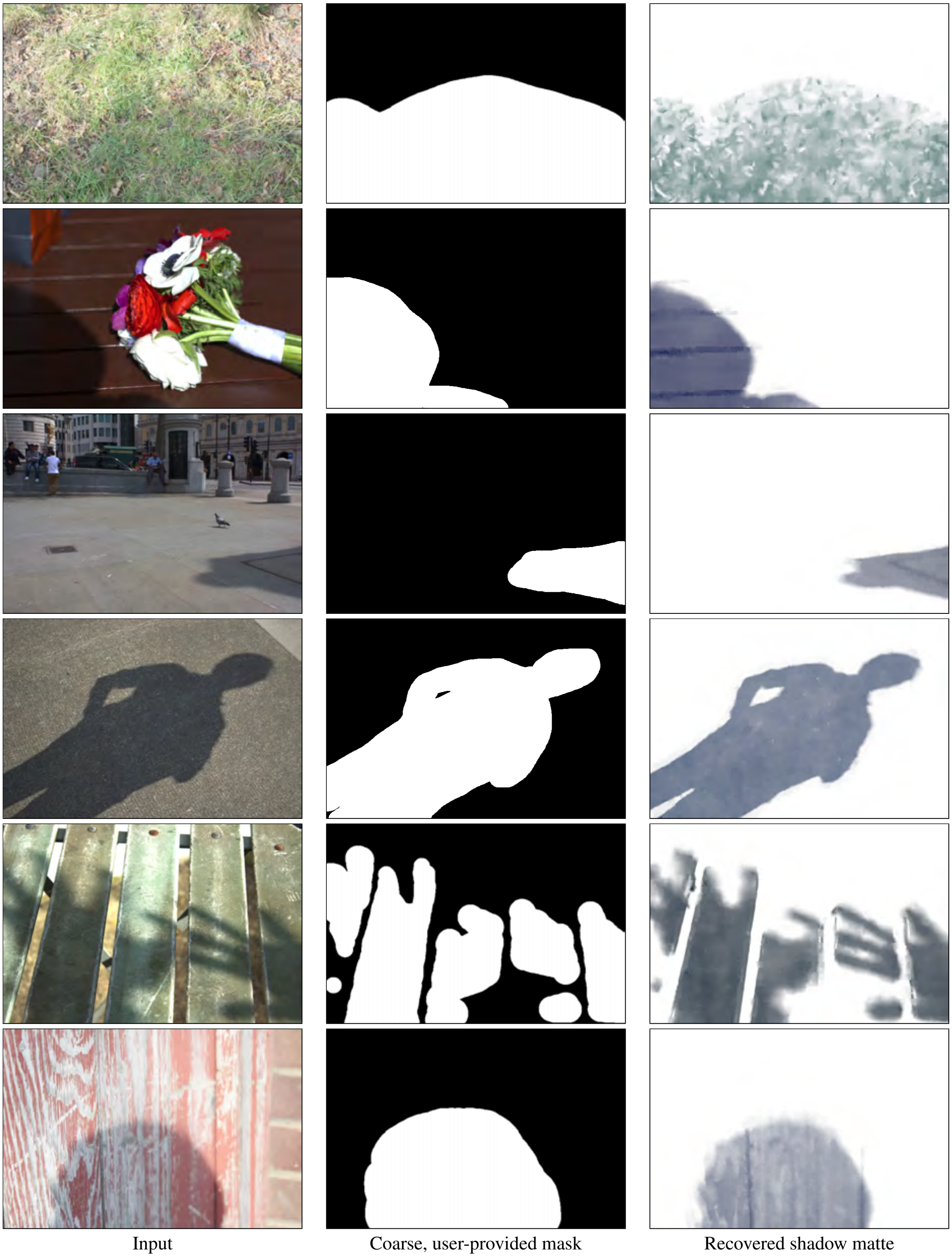


Fig. 9: Results of shadow removal using our method on a variety of real photographs. The left column shows original images, the middle column shows user input, and the right column shows the obtained shadow mattes (the resulting unshadowed images are presented in Figure 1). The mattes could also be used to modify the properties of the shadows as we show in Section 7.1.

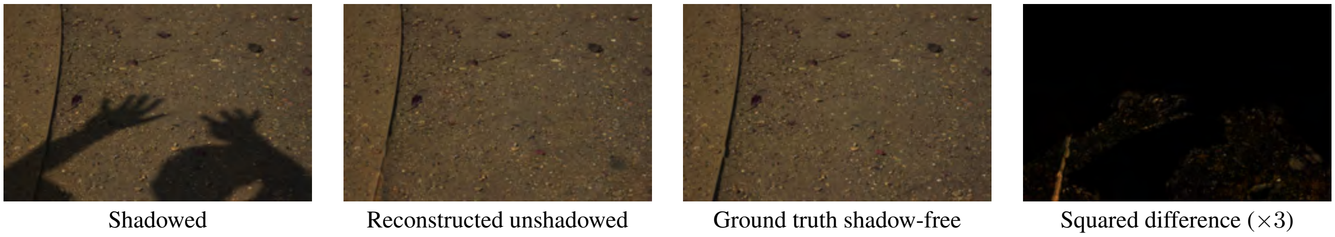
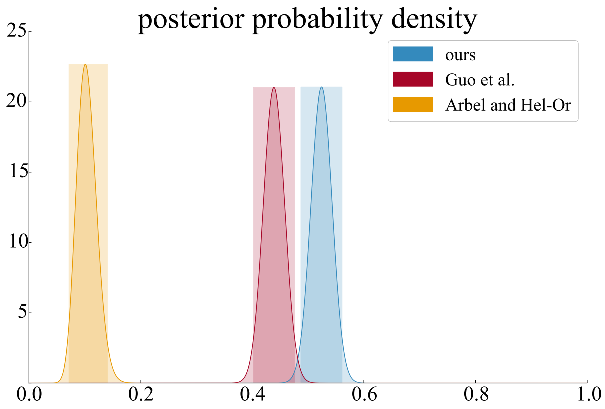


Fig. 10: While the aim of our work is to enable perceptually-plausible results, here we show the differences between the output and the ground truth.



Arbel and Hel-Or	Guo *et al.*	Our technique
10% (7-14% CI)	43% (40-48% CI)	52% (49-56% CI)

Fig. 11: Posterior probabilities of each method winning a ranking round. The shaded, rectangular regions signify the 95% Confidence Interval (CI).

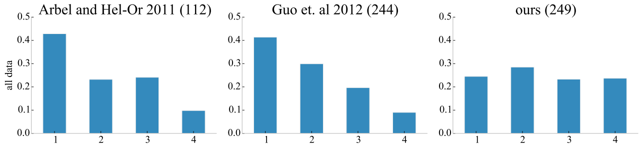


Fig. 12: Normalized histograms of image evaluation scores for different methods. Higher scores correspond to images where the shadow was removed successfully and no artifacts were introduced, while low scores mean that the shadow removal failed and/or there were visible artifacts introduced. Numbers in brackets above each plot show how many evaluations contributed to it. Overall, our method has the highest chance of obtaining high scores and therefore removing shadows successfully.

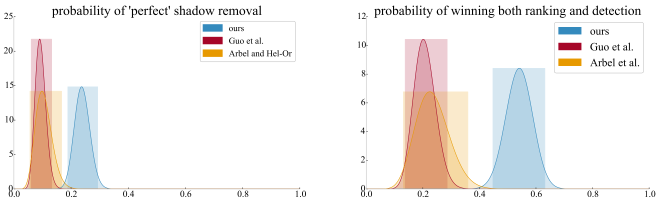


Fig. 13: Left: posterior probability of image having a shadow removed perfectly (mean score $\mu_{eval} > 3.5$). Right: posterior probability of image modified with a given method winning both in the ranking and evaluation phases.

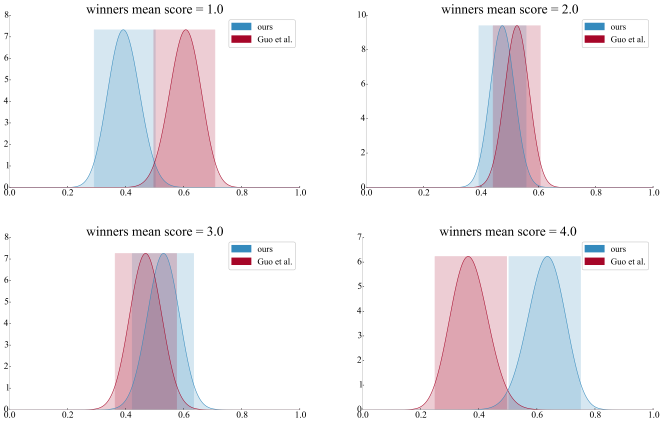


Fig. 14: Probability of winning the ranking task conditioned on the winner's mean score in the evaluation task. Note that when Guo *et al.* win, their scores are likely to be low, while the opposite is true for our method.

shows the probability of a given image winning both the ranking phase and the evaluation phase.

Additionally, in Figure 14 we show the probability of our method or Guo *et al.*'s method winning the ranking task while simultaneously having different evaluation scores. We show that when images unshadowed by Guo *et al.*'s method win, they are likely to have low scores, while our method winning likely means high scores. This can be explained by the fact that in their case, the method sometimes "misses" softer shadows and does not modify them at all. In these cases, the image is likely to rank high on the naturalness scale, but still fail the shadow removal evaluation.

Closer inspection of this combined test set revealed that the mean maximum penumbra width of images from Guo *et al.* is 32 pixels, while for the the test images we have introduced it is 55. We have therefore analyzed how the performance of different methods varies on different subsets of the data. As shown in Figure 15 in the case of testing on Guo's data only, no significant difference between our and Guo *et al.*'s method was observed (while the MAP estimate of our method is lower, the confidence intervals overlap significantly). On the other hand, our method performs much better on the dataset with higher mean penumbra width (*i.e.* softer shadows).

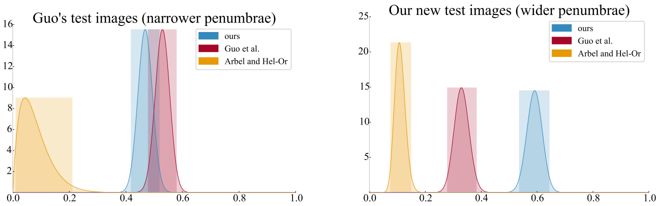


Fig. 15: The different performance characteristics on different slices through the dataset seem to be correlated with the softness of the shadows: our technique has the biggest advantage on images with softer shadows.

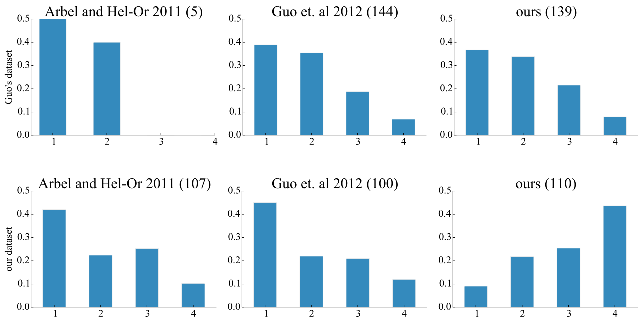


Fig. 16: Histograms of evaluation scores conditioned on the dataset used. Our method is either significantly better or comparable to the competition.

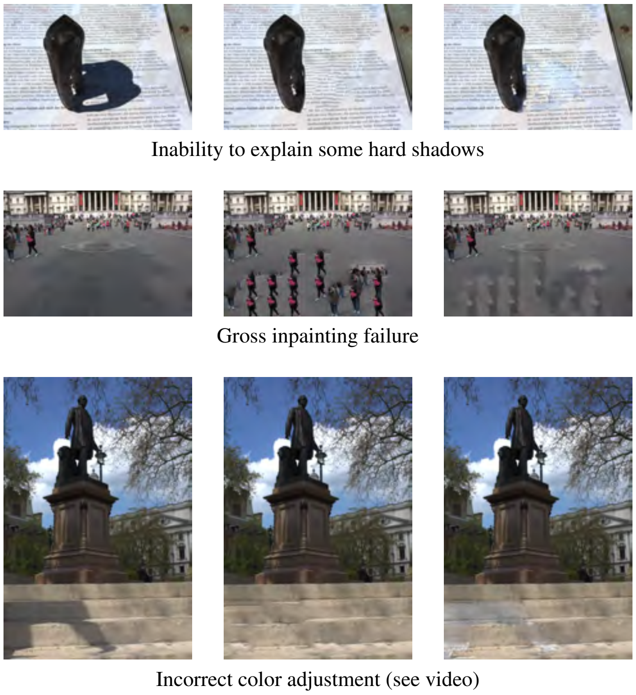
Figure 16 shows similar trends as in the ranking case: when using the dataset with moderately soft shadows our method is indistinguishable from Guo *et al.*'s, but as the penumbra size increases our performance becomes relatively higher.

Finally, we have conducted a separate user study comparing our method to that of Mohan *et al.* [2007]. We found that participants preferred our results 65% of the time when shown against Mohan *et al.*'s, and were significantly more likely to highlight artifacts in their results than in ours.

8. LIMITATIONS

Though our method is somewhat robust to inpainting errors, it is often unable to recover when the initialization fails significantly. Trying to remedy this, we have evaluated three different strategies for producing an initial guess for our algorithm: a) plane-fit to the unmasked regions of the image, similar to Arbel and Hel-Or 2011, b) guided inpainting and c) regular inpainting.

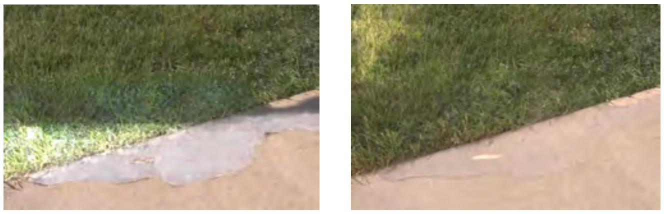
In guided inpainting, our aim was to replace the shadowed regions of the image with unshadowed pixels from the same image and, ideally, the same material. We have modified the PatchMatch algorithm to replace masked image areas with patches from unmasked regions in the same image. Further, we have modified the distance function used by PatchMatch aiming to make it illumination invariant. To achieve that, we have transformed the image from RGB space to a different, multi-channel space, where the new channels included the illumination-invariant image from Finlayson et al. 2009, pixel chromaticity, as well as Gabor filter responses.


Inability to explain some hard shadows

Gross inpainting failure

Incorrect color adjustment (see video)

Fig. 17: The left column shows input images, the inpainted initializations are in the center, and the outputs can be seen on the right. Please note that in the case of incorrect color optimization, the user can easily rectify any mistakes by using our simple shadow editing interface as shown in the supplementary video.


a) Our result

b) Result from Guo *et al.* [2012]

Fig. 18: An example image (p4_1), where our method produces visibly worse results than that of Guo *et al.*

Unfortunately, the final results obtained when using this method proved to be comparable, but still noticeably worse than off-the-shelf inpainting. One of the main issues was that images often contained other shadows that were not to be removed as per user input. As a consequence, despite our efforts, the most closely matching patches used to replace the in-shadow regions came from other shadows, therefore providing a poor guidance for unshadowing. A few examples comparing the three approaches are presented in Appendix 4 in the supplementary material.

Another limitation is that the technique is not able to deal with narrow penumbrae *i.e.* hard shadows, since we have biased our training set and feature representation for the specific challenge of soft shadows (see Figure 18). A valuable future direction would be to either extend the existing approach with an expanded feature representation, possibly based on Convolutional Neural Networks [Farabet et al. 2013] and significantly more hard-shadow training examples, or to pair it with one of the hard-shadow specific methods, such as Guo *et al.* The second approach could follow MacAodha *et al.* [2010], who tackled switching between different optical flow algorithms for different scenarios. In our case,

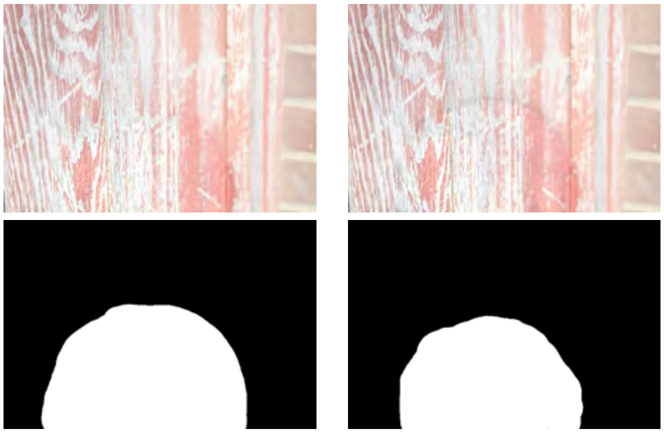


Fig. 19: The success of the method also depends on the user input. The images show the same image ("real26") unshadowed by different people using the masks shown below. The mask on the right does not cover the entire shadow and results in some of penumbra being left in the image. Please see supplementary material for more examples.

the task-classification would be simpler, (hard vs. soft) though our preliminary experiments showed that hard-shadow detection is itself hard, as sudden reflectance changes are often falsely detected as hard-shadows. As classifiers or user-interfaces emerge to discern hard and soft shadows, our method provides effective treatment of the soft-shadowed regions.

Additionally, we explicitly trust our users to provide correct input data. When this is not the case, our method will produce suboptimal outputs, as demonstrated in Figure 19.

Finally, in some cases the inference stage is unable to produce compatible matte suggestions at neighboring regions in the image, which results in visible grid-like artifacts (see *e.g.* the top-right image for Figure 17). While the effects of this limitation are often hidden by the texture underneath the shadow, a possible solution could be to increase the feature patch overlap or to create a stronger pairwise distance constraint. Both of these solutions are challenging, however, as they require more training data and therefore computational effort.

9. CONCLUSIONS AND FUTURE WORK

We have presented a model for removing soft shadows that is not based on heuristics, but instead draws from the experience of the graphics community to learn the relationship between shadowed images and their mattes, from synthetically generated, but realistic, data. Our approach can deal with soft and complex shadows, and produces results faster than the most related techniques in the literature. It requires little time and input from the end-user, and our study showed that our method is significantly better than existing methods in successfully removing soft shadows.

More generally, we have presented a unique use of Regression Random Forests for supervised clustering of high-dimensional data, coupled with a regularization step that is adaptable to general scenarios. Similar ideas could be applied to video *e.g.* by performing regularization across frames.

There are several ways this technique could be extended in future work. One of the most obvious additions could be some understanding of the scene and its contents. With more information about *e.g.* normals and depth discontinuities, our technique might be able to better identify and composite shadows. This information could also feed into the following bootstrapping extension to sample the unshadowed texture more efficiently.

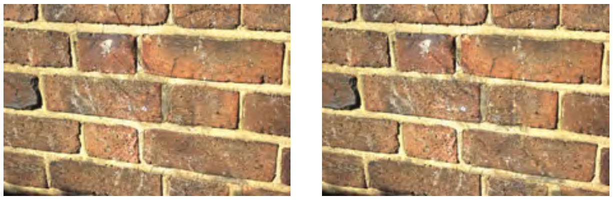


Fig. 20: Comparison with bootstrapping extension. While for some images bootstrapping allows us to obtain comparable results with a much smaller training set (the image on the right used a training set of $J = 50$ images) it makes much stronger assumptions and is therefore not as generally applicable.

Another interesting problem would be the creation of guided inpainting systems that could be used for initializing our method. For example, a method similar to [HaCohen et al. 2011] could help find correct out-of-shadow correspondences, while more user input could provide constraints for the initialization (*e.g.* structure cues as in [Sun et al. 2005]). As better guided inpainting algorithms emerge, our framework will be increasingly effective.

Possible extension As mentioned previously, inpainting algorithms can be divided into those that use a pre-built dataset and those that use the remainder of the image being modified. Similarly, some super-resolution techniques (*e.g.* [Glasner et al. 2009]) use parts of the image to be modified as exemplars for synthesis. Using the same reasoning, we can adapt our method so that it bootstraps the training set from the input image. For this variant, we prepared a set of prerendered shadow mattes and applied a random subset of them to different positions in the shadow-free areas of the input image. This results in pairs of [shadowed, shadow-free] images that we use to train the forest, which is then used for inference in the same process as previously.

The advantage of this extension is that it builds a finely-tuned regressor for this particular image which yields high performance given a smaller training set. On the other hand, it is critically reliant on the assumption that the image has enough out-of-shadow areas with similar texture as the shadowed parts, which limits the number of images suitable for this method. Nevertheless, in the right circumstances this method can produce good results—see Figure 20. More work in this area could lead to more robust solutions.

Further, it might be possible to automatically detect hard and soft shadows in the given image and to selectively apply our method for soft shadows only, and a hard shadow-specific method otherwise.

Additionally, techniques for detecting forgeries, such as [Kee et al. 2013], may gain more power given the ability to explain soft shadows. Finally, works such as [Shih et al. 2013], addressing the problem of image relighting from a single photograph belong to an exciting area that could benefit from the ability to seamlessly remove and modify shadows.

Acknowledgements

Maciej Gryka was supported by Anthropics Technology Ltd. and the UK EPSRC-funded Eng. Doctorate Centre in Virtual Environments, Imaging, and Visualisation (EP/G037159/1). Mike Terry is a Leverhulme Visiting Professor (VP2-2012-034), courtesy of The Leverhulme Trust. Gabriel Brostow is grateful for the UK EP-

SRC's support (EP/J021458/1), and funding from the EU project CR-PLAY (no 611089) http://www.cr-play.eu/.

REFERENCES

ARBEL, E. AND HEL-OR, H. 2011. Shadow removal using intensity surfaces and texture anchor points. *IEEE Transactions on Pattern Analysis and Machine Intelligence 33,* 6.

BARNES, C., SHECHTMAN, E., FINKELSTEIN, A., AND GOLDMAN, D. B. 2009. PatchMatch: A randomized correspondence algorithm for structural image editing. *ACM Transactions on Graphics (Proc. SIGGRAPH) 28,* 3.

BARRON, J. T. AND MALIK, J. 2012. Color constancy, intrinsic images, and shape estimation. In *ECCV*.

BARROW, H. AND TENENBAUM, J. 1978. Recovering intrinsic scene characteristics from images. *Computer Vision Systems*.

BOUSSEAU, A., PARIS, S., AND DURAND, F. 2009. User assisted intrinsic images. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia) 28,* 5.

BOYADZHIEV, I., PARIS, S., AND KAVITA, B. 2013. User-assisted image compositing for photographic lighting. *ACM Transactions on Graphics (Proc. SIGGRAPH)*.

BOYKOV, Y., VEKSLER, O., AND ZABIH, R. 2001. Efficient approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence 20,* 12.

BREIMAN, L., FRIEDMAN, J., STONE, C. J., AND OLSHEN, R. A. 1984. *Classification and regression trees*. Chapman & Hall/CRC.

CHUANG, Y. Y., GOLDMAN, D. B., CURLESS, B., SALESIN, D. H., AND SZELISKI, R. 2003. Shadow matting and compositing. *ACM Transactions on Graphics (Proc. SIGGRAPH) 22,* 3.

CRIMINISI, A., PÉREZ, P., AND TOYAMA, K. 2003. Object removal by exemplar-based inpainting. In *CVPR*.

CRIMINISI, A., ROBERTSON, D., KONUKOGLU, E., SHOTTON, J., PATHAK, S., WHITE, S., AND SIDDIQUI, K. 2013. Regression forests for efficient anatomy detection and localization in computed tomography scans. *Medical Image Analysis*.

FARABET, C., COUPRIE, C., NAJMAN, L., AND LECUN, Y. 2013. Learning hierarchical features for scene labeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence 35,* 8, 1915–1929.

FINLAYSON, G., DREW, M., AND LU, C. 2009. Entropy minimization for shadow removal. *International Journal of Computer Vision 85,* 1.

GLASNER, D., BAGON, S., AND IRANI, M. 2009. Super-resolution from a single image. In *ICCV*.

GRIFFIN, G., HOLUB, A., AND PERONA, P. 2007. Caltech-256 object category dataset. Tech. Rep. 7694, California Institute of Technology.

GROSSE, R., JOHNSON, M. K., ADELSON, E. H., AND FREEMAN, W. T. 2009. Ground-truth dataset and baseline evaluations for intrinsic image algorithms. In *ICCV*.

GUO, R., DAI, Q., AND HOIEM, D. 2012. Paired regions for shadow detection and removal. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

HACOHEN, Y., SHECHTMAN, E., GOLDMAN, D., AND LISCHINSKI, D. 2011. Non-rigid dense correspondence with applications for image enhancement. *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH 2011) 30,* 4, 70:1–70:9.

HAYS, J. AND EFROS, A. A. 2007. Scene completion using millions of photographs. *ACM Transactions on Graphics (Proc. SIGGRAPH) 26,* 3.

KEE, E., O'BRIEN, J. F., AND FARID, H. 2013. Exposing photo manipulation with inconsistent shadows. *ACM Transactions on Graphics 32,* 4. Presented at SIGGRAPH 2013.

KENNEDY, J. M. 1974. *A Psychology of Picture Perception*. Jossey-Bass Publ.

KOLMOGOROV, V. 2006. Convergent tree-reweighted message passing for energy minimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence 28,* 10.

KOPF, J., KIENZLE, W., DRUCKER, S., AND KANG, S. B. 2012. Quality prediction for image completion. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia) 31,* 6.

KRUSCHKE, J. K. 2011. *Doing Bayesian Data Analysis*. Academic Press.

LAFFONT, P., BOUSSEAU, A., AND DRETTAKIS, G. 2013. Rich intrinsic image decomposition of outdoor scenes from multiple views. *IEEE Transactions on Visualization and Computer Graphics 19,* 2.

LAND, E. H. AND MCCANN, J. J. 1971. Lightness and retinex theory. *Journal of the Optical Society of America 61,* 1.

LEVIN, A., LISCHINSKI, D., AND WEISS, Y. 2008. A closed-form solution to natural image matting. *IEEE Transactions on Pattern Analysis and Machine Intelligence 30,* 2.

LOCKERMAN, Y. D., XUE, S., DORSEY, J., AND RUSHMEIER, H. 2013. Creating texture exemplars from unconstrained images. *International Conference on Computer-Aided Design and Computer Graphics*.

MAC AODHA, O., BROSTOW, G. J., AND POLLEFEYS, M. 2010. Segmenting video into classes of algorithm-suitability. In *CVPR*.

MOHAN, A., TUMBLIN, J., AND CHOUDHURY, P. 2007. Editing soft shadows in a digital photograph. *IEEE Computer Graphics and Applications 27,* 2.

PRITCH, Y., KAV-VENAKI, E., AND PELEG, S. 2009. Shift-map image editing. In *ICCV*. Kyoto.

RADEMACHER, P., LENGYEL, J., CUTRELL, E., AND WHITTED, T. 2001. Measuring the perception of visual realism in images. EGWR. Eurographics Association.

REYNOLDS, M., DOBOŠ, J., PEEL, L., WEYRICH, T., AND BROSTOW, G. J. 2011. Capturing time-of-flight data with confidence. In *CVPR*.

SHIH, Y., PARIS, S., DURAND, F., AND FREEMAN, W. 2013. Data-driven hallucination for different times of day from a single outdoor photo. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia)*.

SHOR, Y. AND LISCHINSKI, D. 2008. The shadow meets the mask: Pyramid-based shadow removal. *Computer Graphics Forum 27,* 2.

SHOTTON, J., GIRSHICK, R., FITZGIBBON, A., SHARP, T., COOK, M., FINOCCHIO, M., MOORE, R., KOHLI, P., CRIMINISI, A., KIPMAN, A., AND BLAKE, A. 2012. Efficient human pose estimation from single depth images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

SINHA, P. AND ADELSON, E. 1993. Recovering reflectance and illumination in a world of painted polyhedra. In *ICCV*.

SUN, J., YUAN, L., JIA, J., AND SHUM, H. 2005. Image completion with structure propagation. *ACM Transactions on Graphics (Proc. SIGGRAPH) 24,* 3.

TANG, K., YANG, J., AND WANG, J. 2014. Investigating haze-relevant features in a learning framework for image dehazing. In *CVPR*.

TAPPEN, M. F., FREEMAN, W. T., AND ADELSON, E. H. 2005. Recovering intrinsic images from a single image. *IEEE Transactions on Pattern Analysis and Machine Intelligence 27,* 9.

WANG, J., AGRAWALA, M., AND COHEN, M. F. 2007. Soft scissors: an interactive tool for realtime high quality matting. *ACM Transactions on Graphics (Proc. SIGGRAPH) 26,* 3.

WEISS, Y. 2001. Deriving intrinsic images from image sequences. In *ICCV*.

WU, T., TANG, C., BROWN, M. S., AND SHUM, H. 2007. Natural shadow matting. *ACM Transactions on Graphics 26,* 8.

Received XXXX; accepted XXXX