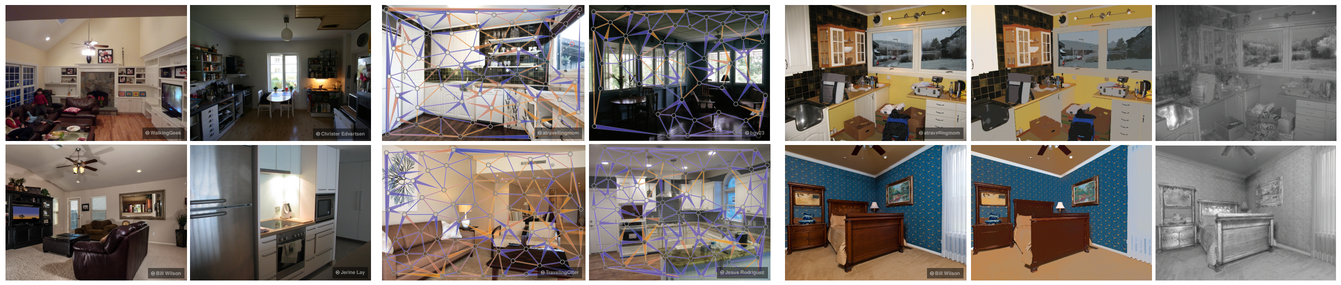
Intrinsic Images in the Wild

Sean Bell   Kavita Bala   Noah Snavely
Cornell University*



(a) 5000+ Internet photographs   (b) Human judgements of surface reflectance   (c) Intrinsic image decompositions: input image, reflectance, shading

Figure 1: *We introduce a public dataset of indoor scenes for intrinsic images in the wild (a). (b) Our crowdsourcing pipeline lets users annotate pairs of points in each image with relative reflectance judgements. (c) Our intrinsic image decomposition algorithm performs well in respecting the human judgements and is based on a fully-connected conditional random field (CRF) that incorporates long-range interactions in the reflectance layer while simultaneously maintaining local detail. All source images are licensed under Creative Commons (🅭).*

Abstract

Intrinsic image decomposition separates an image into a reflectance layer and a shading layer. Automatic intrinsic image decomposition remains a significant challenge, particularly for real-world scenes. Advances on this longstanding problem have been spurred by public datasets of ground truth data, such as the MIT Intrinsic Images dataset. However, the difficulty of acquiring ground truth data has meant that such datasets cover a small range of materials and objects. In contrast, real-world scenes contain a rich range of shapes and materials, lit by complex illumination.

In this paper we introduce *Intrinsic Images in the Wild*, a large-scale, public dataset for evaluating intrinsic image decompositions of indoor scenes. We create this benchmark through millions of crowdsourced annotations of relative comparisons of material properties at pairs of points in each scene. Crowdsourcing enables a scalable approach to acquiring a large database, and uses the ability of humans to judge material comparisons, despite variations in illumination. Given our database, we develop a dense CRF-based intrinsic image algorithm for images in the wild that outperforms a range of state-of-the-art intrinsic image algorithms. Intrinsic image decomposition remains a challenging problem; we release our code and database publicly to support future research on this problem, available online at `http://intrinsic.cs.cornell.edu/`.

CR Categories: I.4.6 [Image Processing and Computer Vision]: Scene Analysis—Photometry, Shading

Keywords: intrinsic images, crowdsourcing, reflectance, shading

Links: DL PDF WEB

*Authors' email addresses: {sbell, kb, snavely}@cs.cornell.edu

1 Introduction

Intrinsic image decomposition is a long-standing inverse problem with many applications in graphics and vision [Land and McCann 1971]. The goal of intrinsic images is to separate an image into two layers, a reflectance (albedo) image and a shading (irradiance) image, which multiply to form the original image. Reliable algorithms for separating illumination from reflectance in scenes would enable a range of applications, such as image-based resurfacing, texture transfer between images, relighting, material recognition, and other interior design tasks. There has been significant recent progress on the problem of intrinsic image decomposition, aided by the release of the MIT Intrinsic Images dataset [Grosse et al. 2009], which contains carefully constructed ground truth for images of objects. However, intrinsic image decomposition is still very challenging, especially on images of real-world *scenes*. There is currently no standard dataset for evaluating intrinsic images on images of such scenes, due in part to the challenge of capturing real-world photos with known ground truth reflectance and illumination. To span the rich range of real-world scenes we need a large set of images. For this scenario, both careful measurement (as in [Grosse et al. 2009]) and using rendered images of synthetic scenes are not practical or satisfactory. Instead, to cover a rich variety of real-world conditions, we select thousands of images from public photo collections [Bell et al. 2013] and leverage human perception by turning to crowdsourcing to collect material annotations for each image.

We present a new, large-scale database of *Intrinsic Images in the Wild*—real-world photos of indoor scenes, with crowdsourced annotations of reflectance comparisons between points in a scene. Rather than creating per-pixel (absolute) annotations, we designed a scalable approach to human annotation involving humans reasoning about the relative reflectance of *pairs* of pixels in each image. This dataset is the first of its kind for intrinsic images, both in its scale and in its use of human annotation: the dataset contains over 5,000 images featuring a wide variety of scenes, and has been annotated with millions of individual reflectance comparisons (on average 100 judgements per image). This makes our dataset several orders of magnitude larger than existing intrinsic image datasets. In addition, we include a dataset of about 400 images with a dense set of annotations (on average 900 per image). Figure 1 illustrates our dataset, including the pairwise comparisons we collect (Figure 1(b)).

Motivated by our new dataset, we have also created a new intrinsic image decomposition algorithm designed for images of real-world scenes. Our algorithm makes use of the fact that many surfaces in indoor scenes share the same material and reflectance, resulting in *long-range* sharing of reflectances across a scene (for example, a painted wall spanning an entire image). We build on recent work in fully connected conditional random field (CRF) inference [Krähenbühl and Koltun 2013] to enable such long-range connections in our algorithm. We evaluate our method on our new benchmark, and show that it outperforms several state-of-the-art algorithms.

In summary, our contributions are:

- A new, large-scale dataset for intrinsic images annotated via crowdsourcing that includes more than 5,000 images. This dataset is completely open and public, including both images and annotations, with the aim of enabling others to design and evaluate new algorithms for scene-level intrinsic images.
- A new intrinsic images algorithm based on a dense CRF formulation that considers long-range material relations to achieve better decomposition on our new database.

We evaluate our new algorithm, as well as a suite of other public intrinsic image algorithms, on our dataset. For each algorithm, we perform extensive cross-validation to find the optimal parameters on our dataset, and find that our new algorithm outperforms these recent algorithms. Nonetheless, intrinsic image decomposition for real-world scenes of this complexity remains a challenging problem, with much room for improvement. We release our database of images and annotations to help drive future research in this problem.

2 Related work

Intrinsic images. Intrinsic image decomposition has a rich history in computer graphics and vision, and has been studied since the 1970s. One of the earliest observations is that large discontinuities in image intensity correspond to changes in reflectance, and all other variations are due mostly to changes in shading [Land and McCann 1971]. This led to the Retinex algorithm in which each image gradient is classified as belonging to either the reflectance layer or the shading layer according to its magnitude. The resulting decomposition is obtained by solving for the pair of layers whose gradients best match these classified gradients. Nearly forty years later, a version of this algorithm was the best performing on the MIT Intrinsic Images dataset [Grosse et al. 2009]. Subsequent algorithms perform much better on this dataset, as described below.

Many others ideas have been proposed to solve the problem. Some recent techniques use classifiers trained on local grayscale patterns [Tappen et al. 2005; Tappen et al. 2006], priors on texture statistics [Oh et al. 2001; Liu et al. 2012], complex priors on shape, albedo, and illumination [Barron and Malik 2012b; Barron and Malik 2012a; Barron and Malik 2013b], meso- and macro-scales of shading [Liao et al. 2013], chromaticity segmentation [Garces et al. 2012], sparse sets of basis reflectances [Omer and Werman 2004; Gehler et al. 2011], non-local texture constraints [Shen et al. 2008; Zhao et al. 2012], and red-black wavelets [Shen and Yeo 2011].

Further, many methods have been proposed that require additional input to solve the problem. With user interaction, Bousseau [2009] showed that a very high-quality decomposition can be obtained, and Carroll [2011] showed that diffuse interreflections can be separated. Others have addressed the problem using additional photos of the same scene either registered as an image stack [Weiss 2001; Hauagge et al. 2013], or taken from different angles (enabling additional 3D reasoning) [Haber et al. 2009; Laffont et al. 2012; Laffont et al. 2013]. Finally, recent methods have been proposed for RGB-D (Kinect-style) imagery [Barron and Malik 2013a; Chen and Koltun 2013]. In contrast, our paper is focused on completely automated methods that work on a single image.

Crowdsourcing. Recently many authors have shown how to effectively gather high-quality data from workers online in an economical and scalable way. Many types of problems have been addressed this way, including: acquiring material and object labels [Bell et al. 2013], solving micro-problems with humans in the loop [Gingold et al. 2012], labeling parts and attributes of birds [Branson et al. 2010], evaluating image retargeting [Rubinstein et al. 2010], and using gauge figures for understanding shape perception [Cole et al. 2009]. Additionally, many studies of worker dynamics have guided the design of tasks in this space, such as modeling workers with a confusion matrix [Dawid and Skene 1979] or with multidimensional classifiers that incorporate notions of user competence, bias, and expertise (CUBAM model) [Welinder et al. 2010].

Intrinsic images databases. Our goal is to scale to the wide diversity of scenes in the real world in various lighting situations, and to acquire "ground truth" data for such scenes. One could create such a dataset in a few different ways. The MIT Intrinsic Images dataset [Grosse et al. 2009] captured images of single objects and rigorously acquired irradiance by spray painting the objects for diffuse measurements. However, this approach does not scale to large numbers of scenes in the wild. Another approach is to render synthetic scenes along with their reflectance and illumination (as has been done in prior work on a few test scenes, or on CG movie sequences [Chen and Koltun 2013]); but these approaches are constrained by the availability of models that span the range of real world materials, lightings and scenes. Instead, we chose to use human judgements as our ground truth data.

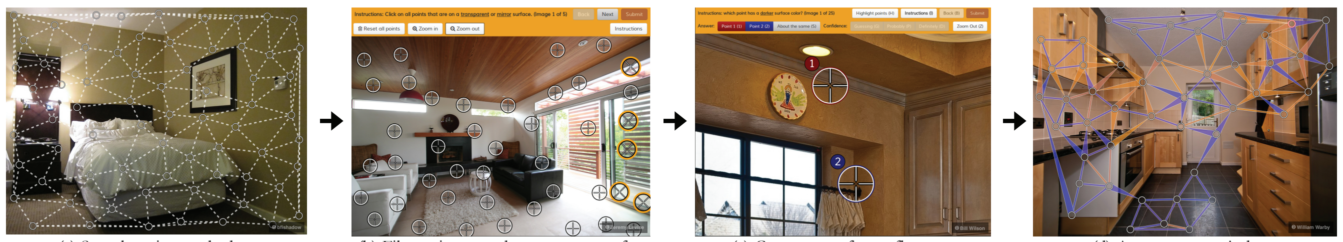
3 Database

We designed a crowdsourcing pipeline where human subjects (on Amazon Mechanical Turk (AMT)) report which of two points in an image has a darker surface reflectance. Our pipeline (shown in Figure 2) scales to millions of human judgements across thousands of real-world images. We aimed to produce a dataset that was broad enough to draw conclusions about intrinsic image algorithms. To design this pipeline we needed to address several questions:

- What judgements should the human annotators make?
- Which images and points should we show each user?
- How to present tasks to the user in an intuitive interface?
- How to identify users who are correctly performing the task and how to aggregate accurate judgements across users?

3.1 What judgements should we collect?

We would ideally collect ground truth reflectance information for each point in an image. However, there are several challenges in collecting such information in a crowdsourced setting that guided how we collected judgements. A key issue is that humans are not good at making absolute judgements about illumination intensity or albedo, though they are reasonably accurate at *relative* judgements. Thus, we decided to ask humans to compare pairs of points in an image, rather than judge individual points. Asking for humans to judge *every* pair in an image is prohibitively expensive, so instead we gather judgements for pairs of sparsely sampled points (though we experiment with different densities, as described below). Another key issue is what question to ask of users—for instance, do we ask them to compare relative illumination at a pair of points, or relative surface albedo/material? We considered asking for illumination judgements (e.g., "which point is more in shadow?") but decided against this, because explicit questions about illumination are chal-



(a) Sample points and edges (b) Filter mirrors and transparent surfaces (c) Compare surface reflectance (d) Aggregate user judgements

Figure 2: Our data collection pipeline. *(a) We sample points and edges from over 5,000 photographs, (b) workers flag points on mirrors and transparent surfaces, (c) users judge pairs of points and indicate which point has a darker surface reflectance (or if they are equal), and (d) we aggregate judgements from 5 users for each pair of points (each edge is styled according to the aggregate results, as described in Figure 3).*

lenging for humans [Ostrovsky et al. 2005]. Hence, we decided to ask for material comparisons for pairs of points. We found that it was important to ask this question in the right way, as phrasing is often key for crowdsourced tasks, and material questions are particularly challenging for novices. We considered various possibilities and phrasings and finally settled on showing users pairs of points and asking them "which point has a darker surface color?" To make sure that workers understand our task, we instruct new workers with an interactive tutorial (described later).

3.2 Which images and which pairs of points?

Given the task described above, we next had to select our images and the points in each image for which we collect annotations.

Images. Our goal was to assemble a broad collection of images from the real world that are representative and free from image editing and other filters. We chose images from the OpenSurfaces dataset [Bell et al. 2013], which were gathered from Flickr and contain a variety of indoor scenes (kitchens, living rooms, bedrooms, and so on). We manually curated photos to remove images that do not represent how the world looks to the naked eye, with effects such as blur, excessive defocus, high noise, depth-of-field, fisheye, poor exposure, black-and-white, visible vignetting, washed out colors, infrared filters, distorted color tones, long-exposure effects, visible HDR artifacts, stitched panoramas, text overlays, and extra borders.

Points. Next, we must decide which points in each image to compare. To achieve good image coverage, and since it can be difficult to reason about the relative diffuse reflectance of points that are far apart in an image, we select points that are approximately equally spaced. In particular, we sample image points using Poisson disk sampling with a minimum radius (we use 7% of the image diameter).

Since we are sampling points that could lie on reflectance discontinuities, we remove potentially difficult judgements from the initial set of Poisson-disk-sampled points. Specifically, we remove (a) points within 4 pixels of a strong edge (as determined with a Canny edge detector), (b) over- or under-saturated points (removing points where $(r + g + b)/3 < 0.02$ or > 0.98) and (c) points whose local neighborhoods have significant variance in color (removing points where the coefficient of variation of chromaticity within a 9×9 pixel window is above 0.5).[1]

Finally, we chose to exclude points where the concept of diffuse reflectance is not well-defined, in particular, points with transparent or mirror-like appearance. We filtered out such points using a separate user task run as a preprocess on the set of points (see Figure 2(b)).

Pairs of points. Now that we have a list of candidate points in an image, we want to choose a set of pairs of points to compare. Since we are asking users the question: "which point has a darker surface color?", we want the points to have similar chromaticity, so that the surface reflectance intensity can be easily compared. We also do not want to sample all $O(n^2)$ pairs of points in each image, since this would lead to excessive redundancy and over 1,000 pairs per photo (which is expensive to annotate for all images). Instead, we compute a Delaunay triangulation of the points, then reject edges of this triangulation where the difference in chromaticity between the point pair is too high (pairs where the Euclidean distance between points in $[r, g, b]/(r + g + b)$ chromaticity space is > 0.125).

After removing these edges, we go back and try to add new edges, by considering all possible edges that do not intersect existing edges and are within our chromaticity threshold. We greedily add such edges, starting from the shortest edge in the set, skipping edges that intersect an existing edge. Finally, we delete any edge that is not part of a cycle; edges in cycles can be cross-checked against each other, allowing us to verify consistency of edges during our subsequent data analysis. On average, the above process results in 44 ± 16 points and 106 ± 45 pairs per image.

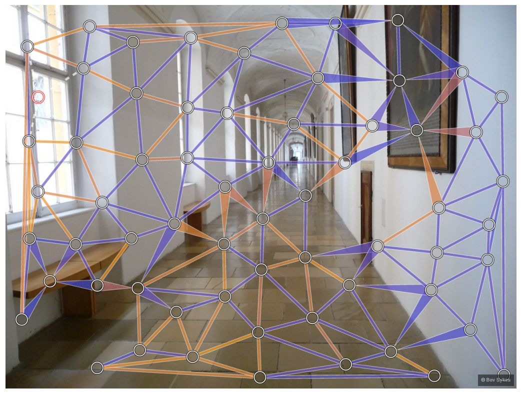
Dense sampling. To understand the effect of image scale on human comparisons, we created another set of more densely sampled edges on a subset of images. In particular, we selected about 400 whitebalanced photos and sampled points with a Poisson-disk sampling radius of 3% of the image diameter. Of the set of whitebalanced photos, we selected about 200 from images that had the most edges discarded by our chromaticity threshold, 100 photos at random, and a final 100 from photos where intrinsic image algorithms perform poorly (according to our metric described in Section 3.5). For these denser points, we do not threshold edges based on image chromaticity, as nearby points tend to be on the same object, and are thus easier to judge. Figure 3 shows example sparse and dense points.

3.3 Annotation interface

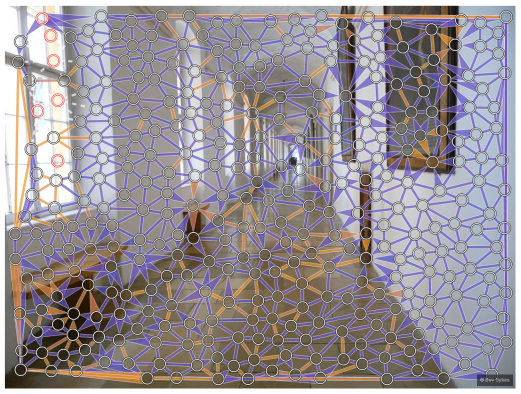
Our user interface for collecting annotations, shown in Figure 2(c), shows the user an image and asks them, for a particular pair of pixels (indicated with crosshairs and labeled Points 1 and 2), which of the two points has a darker surface color. The user can then select one of three options: **Point 1**, **Point 2**, and **About the same**. We ask users to specify their confidence in their assessment as **Guessing**, **Probably**, or **Definitely**, as was done by [Branson et al. 2010].

Tutorial. While humans are skilled at brightness and color constancy—i.e., discounting lighting when comparing surface reflectance (across a range of typical illumination conditions)—they are not used to explicitly thinking about this effect. We found that if one asks users to compare surface color and ignore lighting, a sizeable portion (almost half) of workers misunderstand the task as "which image pixel is darker?". We were able to reduce this proportion to about 25% by adding a tutorial which explains exactly the distinction between pixel intensity difference and surface reflectance difference. The tutorial then presents several test scenes and provides feedback and the correct answer if the worker makes a

[1]The coefficient of variation is a "normalized" measure of variance computed as the sample standard deviation over the mean.



(a) *Sparse sampling.*



(b) *Dense sampling.*

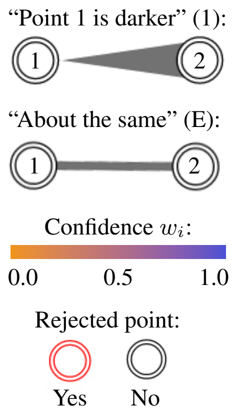


Figure 3: Aggregated human judgements for an example scene. *The edges connecting points indicate the aggregated reflectance judgement comparing the two points. We sample points at two different densities: (a) sparsely, at 7% of the image width and (b) densely, at 3%.*

mistake. We chose difficult test scenes that force users to understand concepts such as: local highlights do not affect surface reflectance, and viewing the scene as a whole is sometimes necessary to visually comprehend and compare reflectance. The tutorial we developed is shown in the video and included on our website.

Efficient input. Since users are to provide thousands of answers, we designed an interface that allows for rapid input. For each photo, our interface displays the photo in its entirety, pauses for 1 second, and then zooms to show a pair of points. Once the user indicates their answer for that pair, the interface smoothly zooms to the next pair of points. We use van Wijk smooth zooming [van Wijk and Nuij 2003] (implemented by D3 [Bostock 2013]) to quickly show the next pair of points. At any time, users can zoom in/out, pan around the image, or repeat the zoom animation and have the points flash to make them easier to see. Users can also return to the previous pair and enter a new answer.

Users had unanimously positive feedback regarding our task UI:

- "Fun. It's exactly what I wish there was more of on MTurk as far as image categorization/similar HITs go."
- "This task is very well-designed and easy to understand and complete. The zoom function is quite helpful."
- "These are addicting as all hell."

Mirror and transparent surfaces. Prior to asking users to judge relative surface reflectance, we filter points through an initial stage in which users flag points as being either on a mirror or on a transparent surface (Figure 2(b)). As with our comparison task, a tutorial explains the types of surfaces we want filtered and lets users practice. See the video for more illustrations of our two user interfaces.

3.4 Data verification

After showing a pair of surface points to multiple workers, we want to classify that pair into one of three categories: (1) Point 1 has a darker reflectance than Point 2, (2) Point 2 has a darker reflectance than Point 1, and (3) they have approximately the same reflectance. We could do simple majority voting on the user-provided input to determine the category; however, consistent with other work on crowdsourcing, we found that the raw input is too noisy for this simple approach to work well.

On Mechanical Turk, there are thousands of workers of varying skill. While we believe that almost all the workers are capable of performing the task, we found that a significant fraction of workers (about 31%, based on our analysis below) either did not try to do the task correctly, or did not understand the instructions, even after the

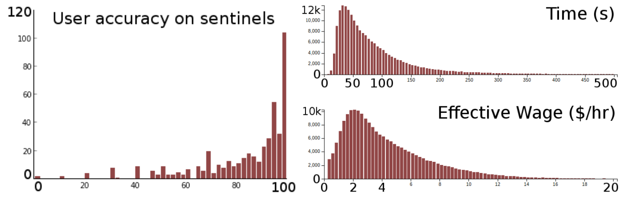


Figure 4: Histograms of worker performance (left) and time spent (right). *Vertical axis on both plots: number of users. Left: percentage of sentinel data answered correctly. Right: time spent and effective wage (pay per task / time spent).*

tutorial where they can try out the task and receive feedback explaining the correct answer. To address this problem, we take three steps to ensure high-quality data: (1) we replicate every task (i.e., each pair of points) to at least 5 workers, (2) we insert "sentinel" objects with known answers into each task, and (3) we use the CUBAM machine learning algorithm [Welinder et al. 2010] to automatically model user competence and bias when computing consensus labels.

Sentinels. Inspired by the microtasks work of Gingold, et al. [2012], each of our tasks (also known as a "HIT" on AMT) contains 25 comparisons plus 5 "sentinel" comparisons drawn from a set of known answers. To hide which items are sentinels, our server dynamically selects 5 test items that the worker has not seen before. As a user submits tasks, we measure their accuracy on the sentinels. The moment a user makes at least 5 mistakes *and* has an average accuracy below 80%, that user is blocked from performing the task. We add the test for average accuracy to avoid prematurely blocking users. When a given user has seen all test items, we stop serving the 5 extra comparisons, as the user has proven to be accurate enough on all sentinels. We chose to test a user across HITs, rather than including a large amount of sentinel items inside each HIT since we found that workers maintain approximately the same accuracy between submissions. This wastes less resources ensuring that workers are behaving correctly. Figure 4 shows the distribution of worker scores on sentinel items. With this method, about 31% of users were blocked. Users seemed appreciative of the fact that the effective pay increased after the sentinels were finished, and we only identified a single user who passed our tests and later began submitting random answers (some time after the sentinels finished).

Modeling workers. Once we have filtered out users who are not correctly performing the task, we must aggregate the reliable answers to obtain a single judgement J_i for each pair of points. Since some comparisons can be genuinely ambiguous, we would like to assign a confidence (or weight) w_i to each judgement J_i. Further, different users may have different competence and internal thresholds when comparing surface reflectances, and we would like to compensate for these effects. The CUBAM user model [Welinder et al. 2010] lends itself to these issues, but requires that the questions have binary answers, while our judgements have 3 possible answers ("Point 1", "Point 2", "About the same"). However, we can convert our 3-class answers into two binary answers by imagining that users are making the following decisions in a small decision tree:

1. Do the two points have the same reflectance?
2. If not, does the darker point have a darker surface reflectance?

For example, a response of "Point 1" is converted to ("No", "No"), if Point 2 is darker than Point 1. Given this conversion, we solve for the most likely competence and threshold for each user and the most likely binary response for each decision using CUBAM. Thus, for each pair of points, we are able to model two forms of user bias: decision (1) models how similar two surfaces have to be before the user considers them to be the same, and decision (2) models whether the user confuses reflectance and image intensity by indicating that points in shade always have a darker reflectance.

To simplify notation, we denote the two points as "D" (darker pixel) and "L" (lighter pixel). We use luminance for this step (CIELAB) since we want to capture human perception. For example, in Figure 5, point L has a darker surface reflectance and thus the correct judgement for this pair is $J_i = \text{L}$.

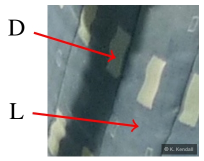


Figure 5: *Points on a fabric surface with a cast shadow.*

Given the two-decision model of each comparison, we use CUBAM to compute a score for each decision, where a score of 0 indicates ambiguous, $+1$ is strong "yes", and -1 is strong "no". If we let $c_{i,1}$ and $c_{i,2}$ be the CUBAM scores for the i^{th} user judgement for our two decisions above, we use the distance from 0 as our confidence/weight w_i for judgement J_i:

$$(J_i, w_i) = \begin{cases} (\text{E}, c_{i,1}) & \text{if } c_{i,1} > 0 \\ (\text{D}, c_{i,2}) & \text{if } c_{i,1} \leq 0 \text{ and } c_{i,2} > 0 \\ (\text{L}, -c_{i,2}) & \text{else} \end{cases} \quad (1)$$

where E indicates that users judge the points to have equal or "about the same" reflectance intensity. Finally, we map the judgements from $\{\text{E}, \text{L}, \text{D}\}$ to the original set of possible answers $\{\text{E}, 1, 2\}$.

Running the experiment. As workers were performing tasks, we periodically reviewed subsets of the aggregated answers. Whenever we decided that the aggregated answer for an unambiguous comparison was incorrect, we corrected the comparison, excluded it from future aggregation with CUBAM, and added it to our set of sentinel test items. This ensures that our sentinel items are difficult—i.e., discriminative in selecting workers who are not trying very hard. This strategy is similar to "hard negative mining" used in machine learning [Felzenszwalb et al. 2008] (the idea of using hard examples to train good classifiers).

To fairly compensate workers, we approved all submissions instantly (even those performed incorrectly). We were surprised to discover that this strategy roughly doubled the number of simultaneous workers for the same rate of pay. Users were very appreciative of the fast and certain approval.

3.5 Error metric: WHDR

In order to use our judgements to evaluate intrinsic image decompositions, we need a way to numerically evaluate a decomposition (R, S) given judgements J_i and weights w_i. We propose a new metric, the "weighted human disagreement rate" (WHDR), which measures the percent of human judgements that an algorithm disagrees with, weighted by the confidence of each judgement:

$$\text{WHDR}_\delta(J, R) = \frac{\sum_i w_i \cdot \mathbf{1}\left(J_i \neq \hat{J}_{i,\delta}(R)\right)}{\sum_i w_i} \quad (2)$$

where R is the algorithm output reflectance layer, $\mathbf{1}(\cdot)$ is the unit indicator, $\hat{J}_{i,\delta}$ is the judgement predicted by the algorithm being evaluated, and δ is the relative difference between two surface reflectances where people *just* begin to switch between saying "they are about the same" (E) to "one point is darker" (1 or 2). To transform the algorithm reflectance layer R into the same units as the judgements (answers in the set $\{1, 2, \text{E}\}$), we must threshold differences between the points used in the human judgement in the reflectance layer R. Because R is only defined up to a scale factor, we compare two reflectances using their ratio, as:

$$\hat{J}_{i,\delta}(R) = \begin{cases} 1 & \text{if } R_{2,i}/R_{1,i} > 1 + \delta \\ 2 & \text{if } R_{1,i}/R_{2,i} > 1 + \delta \\ \text{E} & \text{else} \end{cases} \quad (3)$$

where $R_{1,i}$ is the reflectance sampled at point 1 for the i^{th} judgement (and similarly for $R_{2,i}$). For all of our results, we set $\delta = 10\%$.

3.6 Discussion and results

Using our AMT pipeline, we obtained 4,880,372 responses from 1,381 workers which we aggregated to obtain 875,833 comparisons across 5,230 photos. Of these, 397 photos were also sampled at a higher density to obtain 358,293 comparisons. In Figure 3, we show example aggregated judgements, where directed edges indicate which point is darker, blue/orange edges have high/low confidence, and red points are on a mirror or transparent surface.

Judgement self-consistency. Since we discarded edges that are not part of cycles, we can estimate the self-consistency of our dataset by measuring the fraction of triangles that are consistent (e.g., inequalities $\{R_1 > R_2, R_2 > R_3, R_3 > R_1\}$ are not consistent). For each photo, we divide the total weight of consistent triangles by the total weight of all triangles (the "weight" of a triangle is the sum of the weights of its edges). Averaging across all photos, we find a mean weighted triangle consistency of 92.8%.

User-reported confidence. To assess the accuracy of user-reported confidence, we correlated our judgement weights w_i with user-reported confidence by computing the mean weight w_i for all answers at each level of confidence. We found that inter-user agreement and user-reported confidence were indeed correlated: "Guessing", "Probably", and "Definitely" responses had a mean weight ($\pm$ standard deviation) of 0.54 ± 0.41, 0.63 ± 0.39, and 0.74 ± 0.34, respectively. However, we found that in general, most users did not reliably report their confidence, and thus, despite the positive correlation, we chose not to use the user-reported confidence for any further analysis. In future runs of this task we could eliminate the confidence question to decrease both annotation time and cost.

Validation: varying lighting. To further validate our user judgements, we collected 11 photographs across 4 scenes with identical camera viewpoint but varying lighting conditions from [Boyadzhiev et al. 2013], and then measured the extent to which user judgements changed as a result of the lighting change. The scenes are included on our website and in the supplemental material. On average, we found that for sparsely sampled points, 10.3% of the judgements changed state, and for densely sampled points, 8.7% changed. Judgements that change state are incorrect in at least one of the photos, so we would like our confidence score to be lower for these judgements. Indeed, we find that the mean confidence for consistent judgements (dense: 0.80, sparse: 0.75) is about twice that of judgements that change state (dense: 0.38, sparse: 0.48). This indicates that CUBAM is correctly assigning a lower weight to unreliable judgements.

4 Intrinsic Images Algorithm

Our overarching goal is to develop algorithms that can perform accurate intrinsic image decompositions for real-world photographs of whole scenes "in the wild." For images of scenes, our hypothesis is that it is important to model long-range interactions in which objects tend to share a small number of reflectance values. With this in mind, we designed a new algorithm that has a discrete working set of hypothesis reflectances, and considers all $O(n^2)$ pairs of pixels simultaneously when computing reflectances. Further, because we have collected a large dataset of image judgements, we can explore new algorithms whose parameters can be automatically learned using training data, rather than by hand.

Problem formulation. To recap, an intrinsic image algorithm takes as input a photograph (in our case, an Internet photo of an indoor scene), and seeks to decompose the image into a product of reflectance (albedo) and shading (irradiance) at each point. This is a very under-constrained problem, as there are infinitely many possible reflectance and shading layers that multiply to explain an input image; hence we want to find the decomposition that *most likely* explains the image, under some priors on what makes a likely decomposition. In other words, given RGB image $\mathbf{I}$, we want to find the RGB reflectance layer $\mathbf{R}^*$ and shading layer $\mathbf{S}^*$ that is most likely under probability distribution p:

$$\mathbf{R}^*, \mathbf{S}^* = \arg\max_{\mathbf{R},\mathbf{S}} p(\mathbf{R}, \mathbf{S} \,|\, \mathbf{I}) \qquad (4)$$
$$\text{such that } I_i^c = R_i^c \cdot S_i^c$$

where I_i^c is color channel $c \in \{r, g, b\}$ for image pixel i (and similarly for $\mathbf{R}$ and $\mathbf{S}$).

Formulated this way, the key questions are: (1) how to define the probability distribution $p(\mathbf{R}, \mathbf{S}|\mathbf{I})$ and (2) how to efficiently find the best $\mathbf{R}$ and $\mathbf{S}$ under this distribution. While a full answer to (1) would involve understanding the statistics of natural scenes and illuminations, often these statistics are approximated with a set of priors on $\mathbf{R}$ and $\mathbf{S}$. Our algorithm builds on important priors that have appeared in the literature; our key insight is to apply them in a more global sense, reasoning about all $O(n^2)$ pairs of pixels in an image, whereas most previous methods only consider pairs of neighboring pixels. This global reasoning is a powerful way to encode the observation, true of many real world indoor scenes, that pixels far apart in the image can still often have the same reflectance—for instance, all of the walls in a scene often have the same paint color. In particular, we assign a high probability to decompositions that are consistent with the following priors, each of which have been suggested in prior work:

- Pixels that are nearby, and that have similar chromaticity or intensity, also have similar reflectance.
- Reflectances are piecewise-constant [Land and McCann 1971; Liao et al. 2013; Barron and Malik 2013b].
- Reflectances are sampled from a sparse set [Omer and Werman 2004; Gehler et al. 2011; Shen and Yeo 2011].
- Certain shading values are *a priori* more likely than others [Barron and Malik 2013b].
- Neighboring pixels have similar shading [Garces et al. 2012].
- Shading is grayscale, or the same color as the light source.

These priors are not new, and different methods of global reasoning have been proposed, such as connecting distant image regions with similar texture [Zhao et al. 2012] or optimizing for sparsity in the reflectance layer [Shen and Yeo 2011; Barron and Malik 2013b]. Our method constructs the reflectance layer by drawing from a sparse set of reflectances, a technique inspired by [Gehler et al. 2011].

Our key contribution is that we show how to incorporate all these priors in a global sense, simultaneously reasoning about all $O(n^2)$ pairs of pixels. To do so, we make use of recent work on efficient inference for dense conditional random fields (CRFs) by Krähenbühl and Koltun [2011; 2013], but show how to apply their framework to the problem of intrinsic image decomposition.

Algorithm Overview. Our algorithm works as follows. First, we hypothesize a set of reflectances $\mathcal{R}$ that are likely to exist in the image—one can think of this set as a "palette" of reflectances that we can draw on for that image. The set $\mathcal{R}$ is unique for each image and, for a typical run of our algorithm, will contain 20 entries once our algorithm converges. After first choosing an initial set of reflectance colors $\mathcal{R}$ using clustering, we iterate between two stages:

1. We label each pixel with a reflectance chosen from $\mathcal{R}$ such that $p(\mathbf{R}, \mathbf{S} \,|\, \mathbf{I})$ is maximized.
2. We adjust the reflectances in $\mathcal{R}$ by minimizing discontinuities in the shading layer $\mathbf{S}$.

The first stage improves the reflectance layer, and is optimized using discrete labeling; the second stage improves the shading layer, and is optimized using continuous L^1 minimization. We now describe each stage of the algorithm in detail.

Note on grayscale shading. Since we assume that shading is grayscale, our problem only requires solving for scalar reflectance intensity R and shading intensity S. Given these scalar values, we can expand to a full RGB decomposition by:

$$\mathbf{R}_i = \frac{R_i}{\frac{1}{3}\sum_c I_i^c}\mathbf{I}_i \qquad \mathbf{S}_i = \frac{\frac{1}{3}\sum_c I_i^c}{R_i}[1, 1, 1] \qquad (5)$$

where the chromaticity of $\mathbf{S}$ is assumed to be grayscale and the chromaticity for $\mathbf{R}$ is taken from the input $\mathbf{I}$.

4.1 Initialization

We assume in what follows that our input image $\mathbf{I}$ has a linear response (in our work, we map our downloaded images from an assumed sRGB space to linear). To compute an initial set $\mathcal{R}$, our algorithm starts by performing a k-means clustering of pixel colors in the input image (similar to [Garces et al. 2012]). Rather than work in RGB space, which has correlated channels, we transform the colors to better cluster reflectances. For diffuse surfaces lit with pure white light, image pixels on that surface will have the same chromaticity. To take advantage of this invariant, we transform RGB space to three new axes: (1) pixel intensity, (2) red chromaticity, and (3) green chromaticity:

$$[r, g, b] \mapsto \left[\beta \cdot \frac{r+g+b}{3}, \frac{r}{r+g+b}, \frac{g}{r+g+b}\right] \qquad (6)$$

Under ideal conditions (white lighting, colored diffuse surfaces, no inter-reflections), this transform perfectly separates different reflectance colors along the second two axes. While more sophisticated color spaces have been proposed [Omer and Werman 2004], we find that Equation 6 is sufficient for initialization. When computing color distances, we scale the intensity axis by β since chromaticity is a better predictor of reflectance than intensity. Training against our dataset, we found that a weight of $\beta = 0.5$ works well.

After performing k-means clustering in this color space, we transform back to RGB and collect the cluster centers as our initial set $\mathcal{R}$. Note that while each reflectance in $\mathcal{R}$ is a RGB color, we only use its intensity for the final output. We find that for our algorithm, keeping $\mathcal{R}$ as RGB colors and then projecting down to scalar intensities produces better results than simply using scalar intensity values. In practice, we found that our algorithm is reasonably insensitive to the number of clusters k, though $k \approx 20$ performs the best. Variations on our method are explored later in Figure 7(b) and Table 1.

As described above, our algorithm then alternates between (1) assigning each pixel with a reflectance selected from $\mathcal{R}$ and (2) optimizing the palette of reflectances $\mathcal{R}$ itself to improve shading intensity S.

4.2 Stage 1: Optimize reflectance

In the first stage, we would like to label every pixel in the input image with a reflectance chosen from $\mathcal{R}$. We use x to denote the labeling, where x maps each pixel i to a reflectance chosen from our hypothesis reflectances $\mathcal{R}$, denoted as $\mathcal{R}(x_i)$. Each x_i can be thought of as an integer from 0 to $k-1$ that acts as an index into our palette $\mathcal{R}$, so reflectance intensity $R_i = \frac{1}{3}\sum_c \mathcal{R}^c(x_i)$.

The optimal reflectance labeling is obtained by maximizing $p(x \mid \mathbf{I})$, which is a discrete labeling problem. We now describe how we model this probability.

4.2.1 Using fully connected conditional random fields

Recently, Krähenbühl and Koltun proposed an algorithm for approximately minimizing a global objective function with a quadratic number of terms in linear time [2011; 2013]. While their model was originally proposed to improve object recognition and segmentation, we show that it can be adapted for intrinsic image decomposition. Their model considers objective functions of the form:

$$p(x \mid \mathbf{I}) = \frac{1}{Z(\mathbf{I})} \exp\left\{ -\sum_i \psi_i(x_i) - \sum_{i<j} \psi_{ij}(x_i, x_j) \right\} \tag{7}$$

$$\psi_{ij}(x_i, x_j) = \sum_m \mu^{(m)}(x_i, x_j)\; k^{(m)}(\mathbf{f}_i - \mathbf{f}_j) \tag{8}$$

where,

$p(x \mid \mathbf{I})$	probability of labels x given image $\mathbf{I}$
$Z(\mathbf{I})$	partition function (normalizes the distribution)
i, j	pixel indices
x_i	discrete label for pixel i
$\mathbf{f}_i$	feature vector for pixel i
$\psi_i(\cdot)$	unary cost function for pixel i
$\psi_{ij}(\cdot,\cdot)$	pairwise cost function for pixels i, j
$\mu^{(m)}(\cdot,\cdot)$	m^{th} (negative-semidefinite) label compatibility function
$k^{(m)}(\cdot)$	m^{th} (positive-definite) kernel function

The objective is maximized when x is chosen to minimize the costs incurred by $\psi_i(x_i)$ and $\psi_{ij}(x_i, x_j)$. An important feature of this objective function is that there are $O(n^2)$ pairwise terms ψ_{ij}. When maximizing $p(x \mid \mathbf{I})$, we can omit the partition function $Z(\mathbf{I})$ and minimize the negative logarithm to arrive at an energy function:

$$\begin{aligned} E(x) &= -\log p(x \mid \mathbf{I}) + \text{const} \\ &= \sum_i \psi_i(x_i) + \sum_{i<j} \psi_{ij}(x_i, x_j) \end{aligned} \tag{9}$$

The key insight of their algorithm is that when using a mean field approximation to the CRF distribution, message passing (which is $O(n^2)$) can be approximated as high-dimensional filtering if certain pairwise functions are used. They use the permutohedral lattice of Adams et al. [2010], to perform filtering in linear time, thus obtaining an $O(nd)$ algorithm that approximately maximizes $p(x \mid \mathbf{I})$, despite having $O(n^2)$ terms, where d is the dimensionality of features $\mathbf{f}$. While their prior work does not quantify the error introduced by their approximation, the model performs well in practice and converges quickly. For example, with 640×480 images and 5-dimensional features, optimization takes about 1 second with one CPU thread.

4.2.2 Probability model

Recall that in our model, the layer of labels x maps each pixel to a reflectance chosen from our hypothesis reflectances $\mathcal{R}$. We now explain how we design functions $\psi_i, \psi_{ij}, \mu, k$ and features $\mathbf{f}$ such that the optimal labeling x is a good estimate of reflectance according to our priors.

Our energy function consists of two types of costs: (1) unary costs ψ_i for each pixel, where we want shading to be smooth and avoid extreme values, and (2) pairwise costs ψ_{ij} where we want reflectance to be piecewise constant for all pairs of similar pixels. These costs are represented in an energy function $E(x)$ defined as a weighted sum of three costs:

$$E(x) = \underbrace{w_p E_p(x)}_{\text{pairwise } \psi_{ij}} + \underbrace{w_s E_s(x) + w_l E_l(x)}_{\text{unary } \psi_i} \tag{10}$$

where $E_p(x)$ is a pairwise reflectance term, $E_s(x)$ is a pairwise shading term encoded as a unary term, and $E_l(x)$ penalizes extreme values of shading. We now describe each term in more detail.

Pairwise reflectance E_p. The most important term in our model is our pairwise reflectance term E_p, which encourages pixels that are nearby in position, chromaticity, and intensity, to be assigned the same surface reflectance. We map each pixel to a 5-dimensional feature space $\mathbf{f}$ that includes position, intensity, and chromaticity, such that we expect Euclidean distance in this space to predict reflectance distance. For every pair of pixels in the image, if the two pixels are assigned a different reflectance, we pay a cost proportional to the L^1 difference in this reflectance, Gaussian-weighted according to the distance between the pixels in our feature space.

Mathematically, we define E_p as:

$$\begin{aligned} E_p(x) &= \sum_{i<j} \mu(x_i, x_j) \exp\left(-\frac{1}{2}||\mathbf{f}_i - \mathbf{f}_j||_2^2\right) \\ \mu(x_i, x_j) &= ||\log \mathcal{R}(x_i) - \log \mathcal{R}(x_j)||_1 \end{aligned} \tag{11}$$

where $\mu(\cdot,\cdot)$ is the label compatibility function and $\mathbf{f}_i$ is the feature vector for pixel i:

$$\mathbf{f}_i = \left[\frac{p_i^x}{\theta_p\, d}, \frac{p_i^y}{\theta_p\, d}, \frac{\frac{1}{3}\sum_c I_i^c}{\theta_l}, \frac{I_i^r}{\theta_c \sum_c I_i^c}, \frac{I_i^g}{\theta_c \sum_c I_i^c} \right] \tag{12}$$

Our feature vector describes each pixel i in terms of its position (p_i^x, p_i^y) normalized by the image diameter d, its intensity, and its red/green chromaticity. Each feature is weighted by model parameters θ_p, θ_l, and θ_c. Under white illumination, two pixels with the same reflectance will have the same chromaticity, so instead of using RGB values for color features, we separate intensity from chromaticity and weight them differently. The particular design of this term (e.g., L^1 distances and Gaussian weights) is chosen in part so that our dense CRF (Equation 7) can be efficiently optimized.

As examples of how this term works, consider pixels that are nearby, have similar chromaticity, and have similar intensity. For these

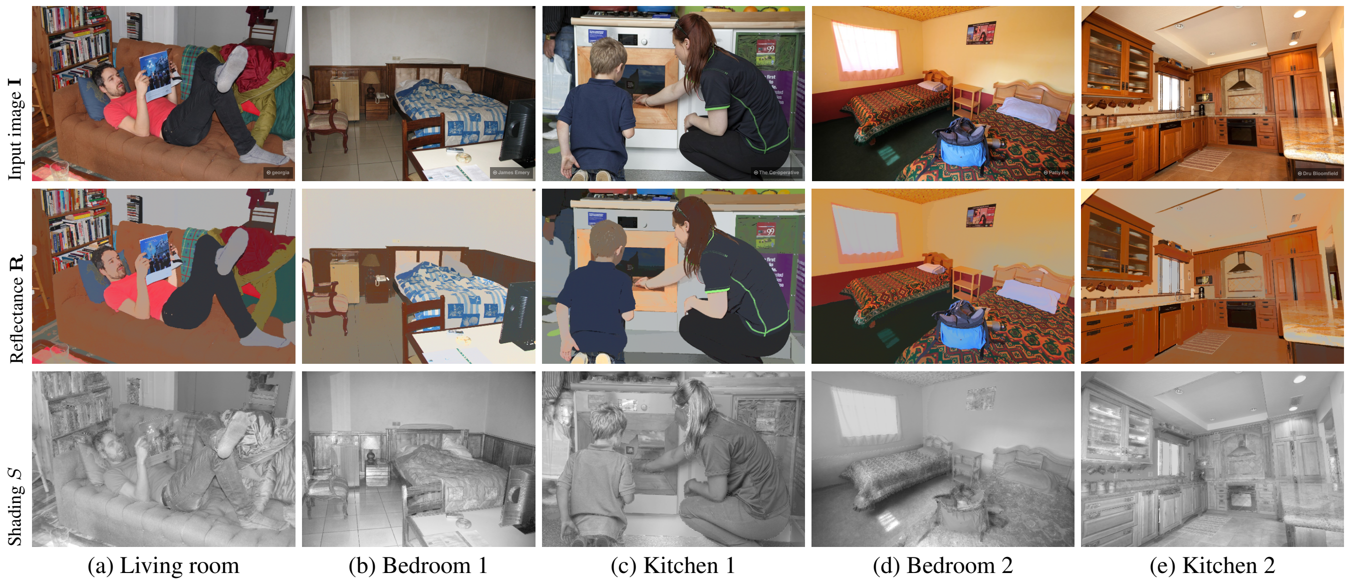


Figure 6: *Example decompositions using our algorithm for a variety of scenes. All outputs are rescaled and mapped to sRGB for display.*

pixels, the Gaussian kernel $\exp\left(-\frac{1}{2}||\mathbf{f}_i - \mathbf{f}_j||_2^2\right)$ will be large, thus constraining these pixels to receive similar reflectances. In contrast, for pixels that are nearby but have a very different chromaticity and/or reflectance, our pairwise term E_p will be small, allowing the shading smoothness term E_s (described below) to constrain the relative reflectance between nearby regions.

Shading smoothness E_s. Another important feature of good decompositions is that the shading channel tends to vary smoothly across smooth surfaces. Such a smoothness prior on shading helps to determine the relative reflectance across a smooth, textured object—we want the texture of such an object to be correctly attributed to the reflectance layer, with a smooth shading layer.

Hence, we would ideally include in our optimization a pairwise term that depends on shading, such as the following:

$$E_s^{\text{desired}}(x) = \sum_{i<j} \mu_s(x_i, x_j) \exp\left(-\frac{1}{2}\left|\left|\frac{\mathbf{p}_i - \mathbf{p}_j}{\sigma_s}\right|\right|_2^2\right)$$

$$\mu_s(x_i, x_j) = (\log S_i - \log S_j)^2 \tag{13}$$

where σ_s is a Gaussian kernel parameter, $\mathbf{p}_i = (p_i^x, p_i^y)$ is the position of pixel i, and S is shading intensity, $S_i = \sum_c I_i^c / \sum_c \mathcal{R}^c(x_i)$. However, including this term would require a label compatibility function μ_s that depends on image features (since S_i depends on $\mathbf{I}_i$), which does not fit within the constraints of Equation 8. Instead, we can approximate S_j by solving the model iteratively, using the shading channel from the previous iteration as our value for S_j. This term now has the functional form of a unary term:

$$E_s(x^{(t)}) = \sum_i \left(\log S_i^{(t)} - \log \tilde{S}_i^{(t-1)}\right)^2 \tag{14}$$

$$\tilde{S}_i^{(t-1)} = \frac{1}{A_i} \sum_j S_j^{(t-1)} \exp\left(-\frac{1}{2}\left|\left|\frac{\mathbf{p}_i - \mathbf{p}_j}{\sigma_s^{(t)}}\right|\right|_2^2\right) \tag{15}$$

where A_i is a normalizing term chosen so that the Gaussian weights sum to 1. Note that $\tilde{S}^{(t-1)}$ is a Gaussian blur of $S^{(t-1)}$ and is fast to compute. Each iteration, we gradually decrease the blur size $\sigma_s^{(t)}$:

$$\sigma_s^{(t)} = \frac{\hat{\sigma}_s\, d}{t} \tag{16}$$

where $\hat{\sigma}_s$ is the normalized initial blur radius, d is the image diameter, and t is the iteration number. In the first iteration $(t = 1)$, we find that omitting this term (E_s) works better than initializing shading intensity $S^{(0)}$ with the image intensity or with a constant. Since the blur radius decreases each iteration, this term effectively results in a coarse-to-fine method of smoothing the shading channel.

Absolute shading intensity E_l. In addition to encouraging smoothness of shading, we want to ensure that the optimizer does not choose extreme values of shading for too many pixels, so we add a penalty that pulls shading intensity S towards a constant:

$$E_l(x) = \sum_i |S_i - \bar{S}| \tag{17}$$

Training against our dataset, we find that $\bar{S} = 0.5$ works well.

Optimizing the model. Using the method of Krähenbühl and Koltun [2013], we can find the discrete labeling x that approximately minimizes Equation 10 in just a few seconds. If any labels are not used in the solution, they are dropped from $\mathcal{R}$.

4.3 Stage 2: Optimize for shading

In Stage 1, we obtained a hypothesis decomposition (R, S), as well as an indication of which pairs of points have the same reflectance (i and j have the same reflectance iff $x_i = x_j$). In Stage 2, we hold the label assignments x fixed and continuously optimize the palette of reflectances $\mathcal{R}$ in order to improve shading intensity S.

Following Garces et al. [2012], we improve our guess of reflectances by minimizing shading discontinuities between adjacent regions across the image. Since adjacent pixels are likely to be nearby in 3D, they are likely to receive the same illumination. However, inside an image region that is assigned a single reflectance label, shading is already constrained (since setting a value for either R or S fixes the other layer). Thus, if we hold the labels x fixed, we can only minimize shading discontinuities across reflectance boundaries, where $x_i \neq x_j$.

We can update the set of reflectances by multiplying $\mathcal{R}$ by a vector of scalars r, which allows us to regularize the relative change r:

$$\mathcal{R}^{(t)}(i) = r^{(t)}(i) \cdot \mathcal{R}^{(t-1)}(i) \tag{18}$$

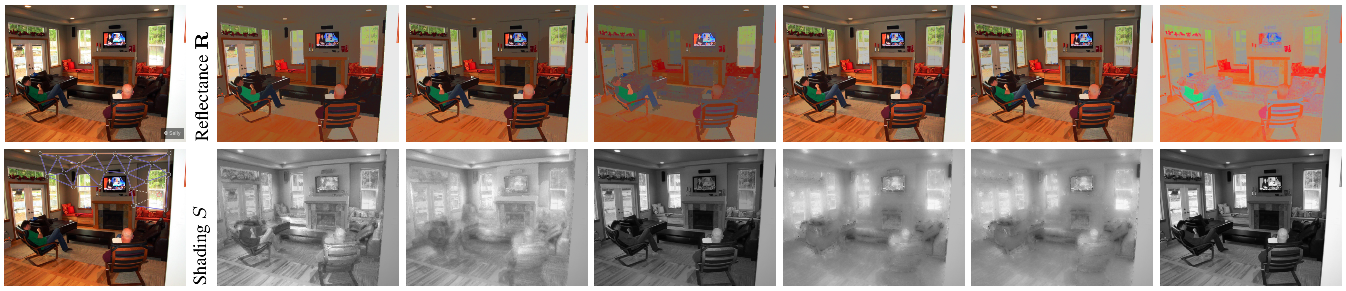

Reflectance $\mathbf{R}$
Shading S
Image $\mathbf{I}$ and user judgements
(a) As described
(b) scalar $\mathcal{R}$
(c) $w_s = 0$ (No S smoothness)
(d) $\theta_p = 0.001$ (Local-only pairwise)
(e) $w_p = 0$ (No pairwise)
(f) $w_l = 0$ (No S regularization)

Figure 7: Algorithm variants. *To illustrate the effect of each term, we take the parameter settings found from training and remove different terms. The WHDR errors for each variant increase from left to right, and are listed in Table 1.*

We solve for r by minimizing shading discontinuities:

$$r^{(t)} = \arg\min_r \sum_{(i,j)\in B} |\log S_i - \log S_j| \tag{19}$$

$$\log S_i = \log\left(\sum_c I_i^c\right) - \log\left(\sum_c \mathcal{R}^{(t-1),c}(x_i)\right) - \log r(x_i)$$

where S_i is the shading intensity, and B is the set of adjacent pixels (i, j) with different reflectance labels $x_i \neq x_j$. By working in log space, we avoid potential problems with numerical instability and divide-by-zero. We efficiently optimize Equation 19 with iteratively reweighted least squares (IRLS), adding a small damping term $10^{-8}||\log r||^2$ to each iteration of IRLS since Equation 19 is only well-defined up to a constant. Note that we are solving for $|r| = k \approx 20$ variables, not an entire shading layer, and thus this step converges in a few seconds.

Final iteration. As a final optional step, we can split apart connected components in the discrete reflectance map, prior to the last L^1 minimization (Equation 19). While this step is not crucial, we find that it provides a small improvement for most scenes. Specifically, we take each connected region in the reflectance image and assign it to a new unique label. As a result, our number of labels in $\mathcal{R}$ jumps from around 20 to around 3000. We find that small connected components in the reflectance channel are typically part of detailed textures. In these cases, smoothing the shading channel performs better than attempting to reason in reflectance space.

Number of iterations. We always run the algorithm for a fixed number of iterations since our shading smoothness term E_s changes every iteration (the blur radius decreases). We find that while the algorithm produces a reasonable result in just a few iterations, the decomposition continues to improve, and that about 25 iterations is sufficient. As listed in Table 1, we obtain mean error (WHDR$_{10\%}$) of 23.5%, 21.5%, and 21.0% after 1, 10, and 25 iterations respectively.

5 Results

In this section we use our dataset to evaluate our algorithm and a range of public algorithms for intrinsic image decomposition. We evaluated our algorithm on all images in our dataset. Some example decompositions are shown in Figure 6. Generally we find that our algorithm performs very well for real-world scenes. While it is particularly good at finding a single reflectance to explain large continuous regions, it can also handle intricate textures such as wallpapers and bricks. Even when there are many surfaces that do not fit the diffuse reflectance model (such as glossy metals or tinted windows), the model often returns a reasonable result.

Algorithm variant	**WHDR**
(a) As described	21.0%
Add reflectance prior $E_r(x) = -\sum_i \log p(\mathcal{R}(x_i))$	21.0%
Add chromaticity prior $E_c(x) = \sum_i \left\Vert \frac{\mathcal{R}(x_i)}{\sum_c \mathcal{R}^c(x_i)} - \frac{\mathbf{I}_i}{\sum_c I_i^c} \right\Vert_1$	21.1%
Use more clusters ($k = 30$)	21.2%
Initialize shading $S_i^{(0)} = \bar{S}$	21.2%
Initialize shading $S_i^{(0)} = \frac{1}{3}\sum_c I_i^c$	21.4%
Eq. 19: expand B to include 3-pixel windows	21.4%
Stop after $n_{\text{iter}} = 10$ iterations	21.5%
Don't split connected components in final iteration	21.6%
Eq. 19: use L^2 instead of L^1	22.2%
Stop after $n_{\text{iter}} = 1$ iteration	23.5%
Use L^1 in shading smoothness (E_s in Eq. 14)	24.6%
(b) Use scalar $\mathcal{R}$ instead of RGB	25.4%
(c) Remove shading smoothness ($w_s = 0$)	26.0%
(d) Limit pairwise to local ($\theta_p = 0.001$)	36.1%
(e) Remove pairwise reflectance ($w_p = 0$)	36.3%
(f) Remove shading regularization ($w_l = 0$)	36.4%

Table 1: *To justify our design choices, we show that many variants produce higher error. Variants (a) through (f) are shown in Figure 7. The variants are described in more detail in the supplemental.*

5.1 Training

Using our large dataset, we explored the effects of changing both the parameters and the structure of our algorithm. We can delete different terms from our energy function $E(x)$ (Equation 10) or we can try adding new terms such as priors on absolute reflectances. Figure 7 shows the effect of deleting different terms and Table 1 lists the training error for an even larger set of variants.

In total, we evaluated 295 variants on the full data set and found the following configuration to have the lowest mean error (WHDR$_{10\%}$): $w_l = 500$, $w_p = 10000$, $w_s = 20000$, $\theta_c = 0.025$, $\theta_l = 0.1$, $\theta_p = 0.1$, $\hat{\sigma}_s = 0.1$, $\beta = 0.5$, $\bar{S} = 0.5$, $k = 20$, $n_{\text{iter}} = 25$.

5.2 Ranking

We evaluated a set of existing open-source algorithms over our entire dataset. To compare these algorithms, we evaluate all of them using two scores. First, we compute their weighted human disagreement rate (WHDR) for each photo and average the scores across all photographs. Second, for each photograph, we rank each algorithm by its WHDR score. If two algorithms tie (which is possible since WHDR is discrete), we average together the ranks of those algorithms. For example, if 4 algorithms get scores $[20\%, 11\%, 20\%, 2\%]$, the corresponding ranks would be

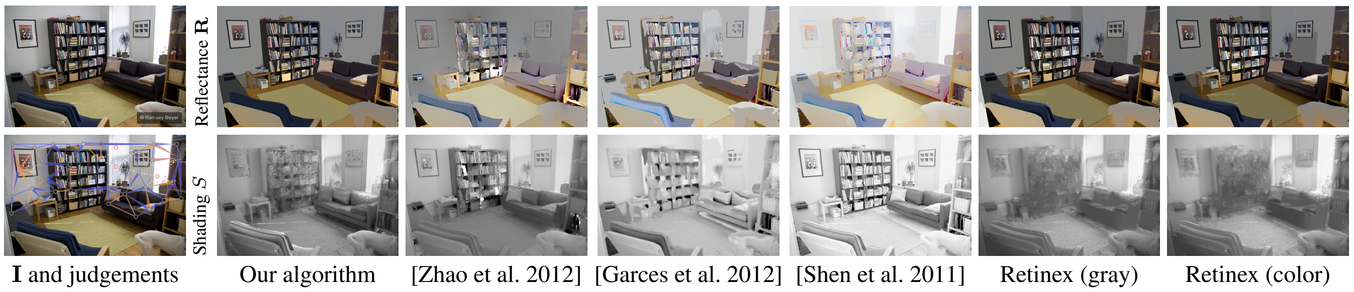
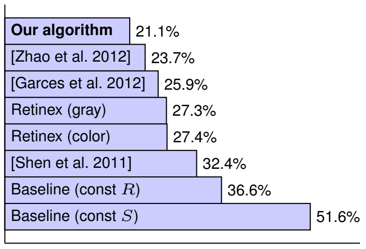
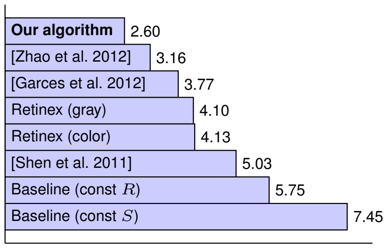


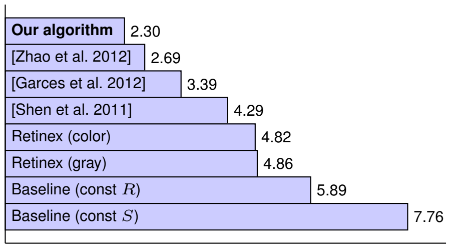
Figure 8: *Visual comparison of our algorithm against several recent open-source algorithms. Each algorithm uses the best parameters found from training (i.e., minimizes mean WHDR$_{10\%}$ across all photos).*



(a) *Mean WHDR$_{10\%}$ over all edges.*



(b) *Mean rank of WHDR$_{10\%}$ over all edges.*



(c) *Mean rank of WHDR$_{10\%}$ over dense edges only.*

Figure 9: *Quantitative comparison of our algorithm against several recent open-source algorithms, using mean weighted human disagreement rate (WHDR$_{10\%}$). For (c), only the 397 densely sampled photos are considered (about 900 judgements per photo with a minimum separation of 3% of the image diameter). All values are computed using leave-one-out cross validation. See Section 5.2 for details.*

$[3.5, 2, 3.5, 1]$. We then compute the *mean rank* of each algorithm as its average rank across all photos. When computing WHDR$_\delta$ we use $\delta = 10\%$, though we find similar results with 5% and 15%.

Figure 9 illustrates our evaluation of several algorithms according to both WHDR and mean rank, including: [Zhao et al. 2012], [Garces et al. 2012], Retinex [Grosse et al. 2009], [Shen et al. 2011], and two baselines. The baseline decompositions are "const R" ($R_i = 1$) and "const S" ($S_i = 1$), and are converted from intensity to RGB using Equation 5. We attempted to include [Gehler et al. 2011] but found that it was too slow to include in our 5,000+ photo evaluation.

Fair evaluation. Since none of these algorithms were given the opportunity to train for our dataset, we tried to make the evaluation fair by trying a range of parameters for each algorithm. We varied both thresholds for Retinex; k for [Garces et al. 2012]; w_d for [Shen et al. 2011]; and chromaticity threshold t, texture patch distance, and texture patch variance for [Zhao et al. 2012]. In addition, we tried two variants of [Shen et al. 2011] and [Zhao et al. 2012]: one as published, and a second where we take sRGB into account when loading the image. In total, we computed over 2,500,000 decompositions for our 5,000+ photo data set. To measure a final test error for each photo, we use leave-one-out cross-validation (i.e., test each photo using the best parameters for all other photos).

Our algorithm. Out of all the methods we evaluated, our algorithm both has the lowest mean error and the best average rank, averaged across all photos. This is consistent with our visual judgement of the results.

[Zhao et al. 2012]. We find that for real-world images, [Zhao et al. 2012] performs quite well. The method has the strength of Retinex in that it is accurate at decomposing texture on smooth surfaces, while also having sparse global constraints to connect distant parts of the image together.

[Garces et al. 2012]. The next best performing algorithm is [Garces et al. 2012], which is designed for real-world scenes. Most errors in output from [Garces et al. 2012] appear to come from incorrect clustering of surface reflectances. Highly textured regions are often merged together into a single component, causing the texture to incorrectly appear in the shading layer.

Retinex [Grosse et al. 2009]. The Retinex algorithm [2009] is the simplest algorithm, and performs surprisingly well for its algorithmic complexity. We found that the optimal threshold for whole scenes (color threshold 0.7, gray threshold 0.5) is very different than the optimal threshold for the MIT Intrinsic Images dataset (color threshold 0.075, gray threshold 1.0). We were also surprised to find that grayscale intensity was a better predictor of reflectance than color, for scenes "in the wild". Consistent with [Grosse et al. 2009], we note that the L^2 variant of Retinex performs significantly worse than L^1 (41.8% versus 27.4%).

[Shen et al. 2011]. Finally, we found [Shen et al. 2011] to perform only slightly better than the baseline in terms of mean error. This seems to correspond to our visual judgement of the output, in which high-frequency texture details are often put in the shading channel, and shading details are not removed from the reflectance layer.

Visual comparison. To understand the typical kinds of visual artifacts made by each algorithm, we include an example decomposition in Figure 8. Since there is significant variety across photos, and the relative ranking between algorithms changes for each photo, this single example is not sufficient to rank algorithms. More visual comparisons are included in the supplemental material.

Dense vs sparse edges. We have two edge lengths: "sparse", with a point spacing of 7% of the image diameter, and "dense", with a point spacing of 3% of the image diameter. We can repeat the evaluation with only dense edges, as shown in Figure 9(c). We find that [Shen et al. 2011] performs better in a close-range evaluation, which is to be expected since the method optimizes over local windows. To make sure that it is not the distribution over edge types that is affecting the results, we compared the distribution of edges and found that it was about the same for both groups: for sparse edges, 62.8% are equal ("about the same") and 63.9% of the dense edges are equal.

Median individual human. Finally, we can place humans on the scale in the same units, by excluding some user responses from the aggregation and then testing them against the user majority, using our WHDR metric. We exclude 100 random users from the aggregation, and test the excluded users against the responses aggregated from the remaining users. Blocked users are excluded from both sets. Since we are evaluating algorithms against humans, who are inconsistent, this is effectively the lowest error we can expect an algorithm to obtain. Averaged across all photos in our dataset, we find that humans have a median WHDR $= 7.5\%$.

5.3 MIT Intrinsic Images dataset

We have validated our algorithm against the MIT Intrinsic Images dataset [Grosse et al. 2009], and found that it performs reasonably. We obtained a mean training and test error of $\text{LMSE}_{20} = 0.027$ and $\text{LMSE}_{20} = 0.031$ respectively (which matches color Retinex $\text{LMSE}_{20} = 0.031$). This is certainly not the lowest error reported for this dataset. For example, [Barron and Malik 2013b] explicitly models shape and illumination and achieves a lower error, but is limited to individual objects. Our method focuses on real-world scenes, which have very different statistics from small, cleanly segmented purely diffuse objects.

6 Limitations and future work

Weaknesses of WHDR. When visually evaluating decompositions, there are many different mistakes that algorithms make: texture variations in $\mathbf{S}$, shadows/highlights in $\mathbf{R}$, etc. No algorithm avoids all mistakes. An important motivation was therefore to have a large dataset to evaluate algorithms on a wide variety of scenes, each with their own challenges. The question of ranking algorithms becomes: how much should we penalize each mistake? Our "weighted human disagreement rate" (WHDR) metric is one such answer. We recognize that certain mistakes are not well captured by WHDR, such as a decomposition that does not model colored shading (since our annotations only measure intensity) or that leaves texture in the shading channel (since the sampling is sparse). Conversely, WHDR is sensitive to errors on single-color surfaces (e.g., walls), where humans tend to be confident and consistent. While there is room for proposing other metrics, we found that algorithms that achieved lower WHDR also achieved qualitatively better results.

Reflection models. Our dataset currently filters out mirror and transparent objects, but there is still a wide range of surfaces that do not fit diffuse reflectance models. We chose not to ask users to estimate gloss since many glossy items appear diffuse if they are not oriented to receive a highlight. Nonetheless, we would like to explore this issue in the future and expand our image formation model to include glossy reflection.

Colored lighting and hard shadows. Scenes in the real world have colored illumination (Figure 10), especially when considering interreflections between colored surfaces. It is common for scenes to have multiple dominant illuminating colors, such as a blue sky and an incandescent light bulb. Further, hard shadows with colored illumination cause the shadowed region to have a shift in chromaticity. For these scenes, our algorithm may put the shadow discontinuity in the texture channel. While the common assumption that shading is grayscale works reasonably well for many scenes, we would like to consider methods and collect datasets that capture color in the shading channel and do not assume that $\mathbf{S}$ is grayscale.

Synthetic images. We would like to further evaluate the quality of human judgements by using ground truth synthetic rendered images. However, the main challenge with synthetic images is that it is extremely difficult to replicate the true diversity and variety found

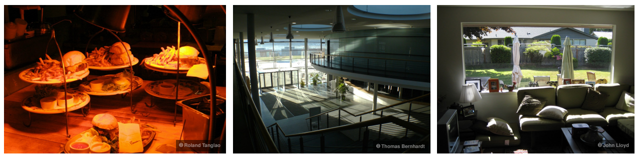


Figure 10: *Challenging scenes that violate our assumptions. Using the parameters from Sec. 5.1, we obtain WHDR* $= 67\%, 45\%, 43\%$.

in Internet photo collections. Further, many 3D scenes available online do not accurately model reality and include features such as meso-scale geometry in albedo maps.

Photograph variety. While our dataset does a reasonable job of covering typical indoor photographs, there are only a few outdoor and nature scenes (e.g., in the "staircase" category), and thus it does not fully represent the statistics of all images available online. Future work could include determining whether the same priors (piece-wise constant reflectances, sparse set of intensities, etc.) are useful in decomposing outdoor photographs.

7 Conclusion

In this paper we introduced a large scale dataset (5000+ images) for *Intrinsic Images in the Wild*, with crowdsourced pairwise annotations of reflectance intensity comparisons. We also introduced a dense-CRF-based intrinsic image decomposition algorithm that considers long-range interactions to achieve better decomposition on our new database, while maintaining local detail. While we have taken steps towards intrinsic images for real-world scenes and uncalibrated photos, many challenges and avenues of future work remain on this problem. Just as the availability of the MIT Intrinsic Images dataset was important to galvanizing research on intrinsic image algorithms, we hope our public database and code will help drive improvements in intrinsic image decomposition in the wild.

Acknowledgements

We would like to thank Kevin Matzen for his invaluable help in putting together our submission. This work was supported in part by a NSERC PGS-D scholarship, the National Science Foundation (grants IIS-1149393, IIS-1011919, IIS-1161645), and by the Intel Science and Technology Center for Visual Computing. In the supplemental material, we acknowledge the Flickr users who released their images under Creative Commons licenses.

References

Adams, A., Baek, J., and Davis, A. 2010. Fast high-dimensional filtering using the permutohedral lattice. *Computer Graphics Forum (Eurographics) 29*, 2.

Barron, J. T., and Malik, J. 2012. Color constancy, intrinsic images, and shape estimation. In *Proc. European Conference on Computer Vision*.

Barron, J. T., and Malik, J. 2012. Shape, albedo, and illumination from a single image of an unknown object. In *Proc. Computer Vision and Pattern Recognition*.

Barron, J. T., and Malik, J. 2013. Intrinsic scene properties from a single RGB-D image. In *Proc. Computer Vision and Pattern Recognition*.

Barron, J. T., and Malik, J. 2013. Shape, illumination, and reflectance from shading. Tech. rep., UC Berkeley.

BELL, S., UPCHURCH, P., SNAVELY, N., AND BALA, K. 2013. OpenSurfaces: A richly annotated catalog of surface appearance. *ACM Trans. on Graphics (SIGGRAPH) 32*, 4.

BOSTOCK, M., 2013. D3. `http://d3js.org/`. [Online; accessed 24-Mar-2014].

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

BOYADZHIEV, I., PARIS, S., AND BALA, K. 2013. User-assisted image compositing for photographic lighting. *ACM Trans. on Graphics (SIGGRAPH) 32*, 4.

BRANSON, S., WAH, C., BABENKO, B., SCHROFF, F., WELINDER, P., PERONA, P., AND BELONGIE, S. 2010. Visual recognition with humans in the loop. In *Proc. European Conference on Computer Vision*.

CARROLL, R., RAMAMOORTHI, R., AND AGRAWALA, M. 2011. Illumination decomposition for material recoloring with consistent interreflections. *ACM Trans. on Graphics (SIGGRAPH)*.

CHEN, Q., AND KOLTUN, V. 2013. A simple model for intrinsic image decomposition with depth cues. In *Proc. International Conference on Computer Vision*.

COLE, F., SANIK, K., DECARLO, D., FINKELSTEIN, A., FUNKHOUSER, T., RUSINKIEWICZ, S., AND SINGH, M. 2009. How well do line drawings depict shape? *ACM Trans. on Graphics (SIGGRAPH) 28*, 3.

DAWID, A. P., AND SKENE, A. M. 1979. Maximum likelihood estimation of observer error-rates using the EM algorithm. In *J. Roy. Statistical Society*.

FELZENSZWALB, P., MCALLESTER, D., AND RAMANAN, D. 2008. A discriminatively trained, multiscale, deformable part model. In *Proc. Computer Vision and Pattern Recognition*.

GARCES, E., MUNOZ, A., LOPEZ-MORENO, J., AND GUTIERREZ, D. 2012. Intrinsic images by clustering. *Computer Graphics Forum (Eurographics Symposium on Rendering) 31*, 4.

GEHLER, P., ROTHER, C., KIEFEL, M., ZHANG, L., AND SCHOLKOPF, B. 2011. Recovering intrinsic images with a global sparsity prior on reflectance. In *Neural Information Processing Systems*.

GINGOLD, Y., SHAMIR, A., AND COHEN-OR, D. 2012. Micro perceptual human computation. *ACM Trans. on Graphics 31*, 5.

GROSSE, R., JOHNSON, M. K., ADELSON, E. H., AND FREEMAN, W. T. 2009. Ground truth dataset and baseline evaluations for intrinsic image algorithms. In *Proc. International Conference on Computer Vision*.

HABER, T., FUCHS, C., BEKAER, P., SEIDEL, H.-P., GOESELE, M., AND LENSCH, H. P. 2009. Relighting objects from image collections. In *Proc. Computer Vision and Pattern Recognition*.

HAUAGGE, D., WEHRWEIN, S., BALA, K., AND SNAVELY, N. 2013. Photometric ambient occlusion. *Proc. Computer Vision and Pattern Recognition*.

KRÄHENBÜHL, P., AND KOLTUN, V. 2011. Efficient inference in fully connected CRFs with gaussian edge potentials. In *Neural Information Processing Systems*.

KRÄHENBÜHL, P., AND KOLTUN, V. 2013. Parameter learning and convergent inference for dense random fields. In *Proc. International Conference on Machine Learning*.

LAFFONT, P.-Y., BOUSSEAU, A., PARIS, S., DURAND, F., AND DRETTAKIS, G. 2012. Coherent intrinsic images from photo collections. *ACM Trans. on Graphics (SIGGRAPH Asia) 31*, 6.

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

LAND, E. H., AND MCCANN, J. J. 1971. Lightness and retinex theory. *J. Opt. Soc. Am. 61*, 1.

LIAO, Z., ROCK, J., WANG, Y., AND FORSYTH, D. 2013. Non-parametric filtering for geometric detail extraction and material representation. In *Proc. Computer Vision and Pattern Recognition*.

LIU, X., JIANG, L., WONG, T.-T., AND FU, C.-W. 2012. Statistical invariance for texture synthesis. *IEEE Trans. on Visualization and Computer Graphics 18*, 11.

OH, B. M., CHEN, M., DORSEY, J., AND DURAND, F. 2001. Image-based modeling and photo editing. In *Proc. Conference on Comp. Graphics and Interactive Techniques (SIGGRAPH)*.

OMER, I., AND WERMAN, M. 2004. Color lines: image specific color representation. In *Proc. Computer Vision and Pattern Recognition*.

OSTROVSKY, Y., CAVANAGH, P., AND SINHA, P. 2005. Perceiving illumination inconsistencies in scenes. *Perception 34*, 11.

RUBINSTEIN, M., GUTIERREZ, D., SORKINE, O., AND SHAMIR, A. 2010. A comparative study of image retargeting. *ACM Trans. on Graphics (SIGGRAPH Asia) 29*, 6.

SHEN, L., AND YEO, C. 2011. Intrinsic images decomposition using a local and global sparse representation of reflectance. In *Proc. Computer Vision and Pattern Recognition*.

SHEN, L., TAN, P., AND LIN, S. 2008. Intrinsic image decomposition with non-local texture cues. In *Proc. Computer Vision and Pattern Recognition*.

SHEN, J., YANG, X., JIA, Y., AND LI, X. 2011. Intrinsic images using optimization. In *Proc. Computer Vision and Pattern Recognition*.

TAPPEN, M., FREEMAN, W., AND ADELSON, E. 2005. Recovering intrinsic images from a single image. *IEEE Trans. on Pattern Analysis and Machine Intelligence*.

TAPPEN, M. F., ADELSON, E. H., AND FREEMAN, W. T. 2006. Estimating intrinsic component images using non-linear regression. In *Proc. Computer Vision and Pattern Recognition*.

VAN WIJK, J. J., AND NUIJ, W. A. 2003. Smooth and efficient zooming and panning. In *Proc. IEEE Conference on Information Visualization (INFOVIS)*.

WEISS, Y. 2001. Deriving intrinsic images from image sequences. In *Proc. International Conference on Computer Vision*.

WELINDER, P., BRANSON, S., BELONGIE, S., AND PERONA, P. 2010. The multidimensional wisdom of crowds. In *Neural Information Processing Systems*.

ZHAO, Q., TAN, P., DAI, Q., SHEN, L., WU, E., AND LIN, S. 2012. A closed-form solution to retinex with nonlocal texture constraints. *IEEE Trans. on Pattern Analysis and Machine Intelligence 34*, 7.