Pigmento: Pigment-Based Image Analysis and Editing

Jianchao Tan, Stephen DiVerdi, Jingwan Lu, Yotam Gingold

Abstract—The colorful appearance of a physical painting is determined by the distribution of paint pigments across the canvas, which we model as a per-pixel mixture of a small number of pigments with multispectral absorption and scattering coefficients. We present an algorithm to efficiently recover this structure from an RGB image, yielding a plausible set of pigments and a low RGB reconstruction error. We show that under certain circumstances we are able to recover pigments that are close to ground truth, while in all cases our results are always plausible. Using our decomposition, we repose standard digital image editing operations as operations in pigment space rather than RGB, with interestingly novel results. We demonstrate tonal adjustments, selection masking, cut-copy-paste, recoloring, palette summarization, and edge enhancement.

Index Terms—Painting, color, RGB, non-photorealistic editing, NPR, kubelka-munk, pigment, paint, mixing, layering, image, editing.

✦

1 INTRODUCTION

Stated generally, a "painting" in the physical world is a two-dimensional arrangement of material. This material may be oil or watercolor paint, or it may be ink from a pen or marker, or charcoal or pastel. These pigments achieve a colorful appearance by virtue of how they absorb and reflect light and their thickness. Kubelka and Munk [1], [2] described a model for the layering of physical materials, and Duncan [3] extended it to include homogeneous mixing. In this model, the appearance of a material (reflectance and transmission of light) is defined by how much it scatters and absorbs each wavelength of light and its overall thickness. These models are widely used to model the appearance of paint, plastic, paper, and textiles; they have been used previously in the computer graphics literature [4], [5], [6], [7].

When painting, artists choose or create a relatively small set of pigments to be used throughout the painting. We call this set the *primary pigment palette*. We assume that all observed colors in the painting are created by mixing or layering pigments from the palette.

When we view a painting, either directly with our eyes or indirectly after digitizing it into a three-channel RGB image, we observe only the overall reflectance and not the underlying material parameters. In RGB-space, the underlying pigments which combine to form the appearance of a pixel are not accessible for editing. One color in the palette cannot be easily changed or replaced. Translucent objects, common in paintings due to the mixing of wet paint, cannot be easily extracted or inserted.

We propose an approach to decompose a painting into its constituent pigments in two stages. First, we compute a small set of pigments in terms of their Kubelka-Munk (KM) scattering and absorption parameters. Second, we compute per-pixel mixing proportions for the pigments

that reconstruct the original painting. We show that this decomposition has many desirable properties. Particularly for images of paintings, it is able to achieve lower error reconstructions with smaller palettes than previous work. Furthermore, the decomposition enables image editing applications to be posed in pigment space rather than RGB space, which can make them more effective or more expressive. We demonstrate tonal adjustments by editing pigment properties; recoloring; selection masking; copy-paste; palette summarization; and edge enhancement.

Thematically, this work is similar to Lillicon [8] and Project Naptha [9], which both present ways to interpret structure in unstructured documents to enable high level edits based on the interpreted structure. In Lillicon's case, the structure is an alternate vector representation of the artwork, while in Project Naptha, the structure is styled text within the image. Our contribution is to apply this strategy to flat, unstructured RGB images of paintings, which are created via a complex structure (physical pigments and brush strokes). Our analysis allows us to interpret the complex structure of the painting from the RGB image, which enables editing operations based on that structure.

2 RELATED WORK

Our work is inspired by the recent efforts of Tan et al. [10], Lin et al. [11], Aksoy et al. [12] and Zhang et al. [13] to decompose an arbitrary image into a small set of partially transparent layers suitable for RGB compositing. Tan et al. [10] use RGB-space convex hull geometry to extract a palette, and then solve an optimization problem to extract translucent layers for the Porter-Duff "over" compositing operator (alpha compositing), which is the standard color compositing model. Lin et al. [11] extract translucent layers from images and videos based on an additive color mixing model. They use locally linear embedding, which assumes that each pixel is a linear combination of its neighbors. Aksoy et al. [12] extract translucent layers from images, also based on an additive color mixing model. However, unlike Tan et

---

- *J. Tan and Y. Gingold are with George Mason University.*
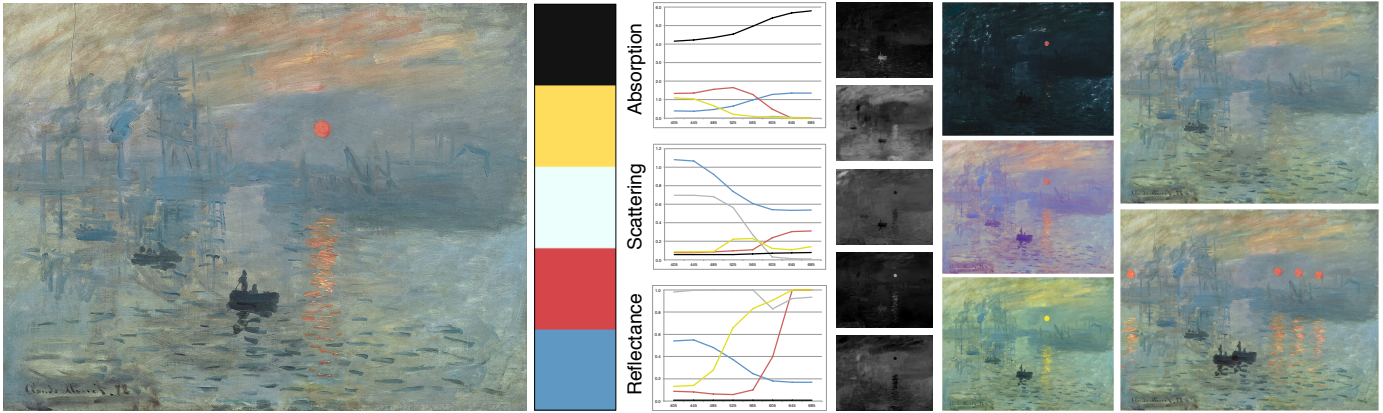
- *S. DiVerdi and J. Lu are with Adobe Research.*

Fig. 1: Analysis and editing of Monet's "Impression, soleil levant." From left to right, input image, extracted palette in RGB, multispectral coefficient curves for palette pigments, mixing weights, recoloring, and cut-copy-paste.

al. [10] and Lin et al. [11], each layer's color varies spatially. Zhang et al. [13] use a clustering-based method to extract palette colors and then decompose the entire image into a linear combination of them. This is a similar representation as the additive mixing layers from Lin et al. [11] and Aksoy et al. [12]. All of these decompositions allow users to edit the image in a more intuitive manner, effectively segmenting the image by color and spatial coherence. Similarly, Chang et al. [14] extract a small palette of colors from an image and implicitly model each pixel as a mixture of those palette colors to enable image recoloring using radial basis functions. We extend these results specifically for physical paintings by using a physically-inspired model of pigment mixing (Kubelka-Munk) and estimating multispectral (greater than RGB) pigment properties.

Our work is contemporaneous with Aharoni-Mack et al. [15], who decompose watercolor paintings into linear mixtures of a small set of primary pigments also using the Kubelka-Munk mixture model. The primary differences between our approaches is that they target (translucent) watercolor paintings and use 3-wavelength (RGB) parameters with varying thickness, while we evaluate our approach with (opaque) acrylic and oil paintings and compute an 8-wavelength constant-thickness decomposition. They similarly use a convex-hull in color-space to identify palette colors. Both methods regularize the problem at least in part with spatial smoothness. Both methods leverage existing datasets of measured Kubelka-Munk scattering and absorption parameters (3-wavelength watercolor pigment parameters from Curtis et al. [4] versus 33-wavelength acrylic parameters from Okumura [16]).

Algorithmically, our work is most similar to that of Kauvar et al. [17], which optimizes a set of multispectral illuminants and linear mixing weights to reproduce an image. This is suitable for their scenario (choosing projector illuminants) but not for mimicking physical paintings. The nonlinear nature of the Kubelka-Munk equations makes our problem much harder.

While the Kubelka-Munk (KM) equations [1] can be used to reproduce the color of a layer of pigment, the pigment coefficients are difficult to acquire [16], so researchers have pursued a simplified model. Curtis et al. [4] use a three wavelength model they compute from samples of paint
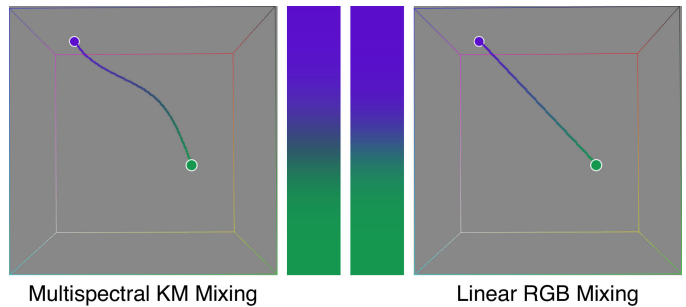

Fig. 2: Comparing mixing models. Left: A gradient interpolating between purple and green pigments using the KM equation, and the resulting colors in RGB-space. Right: A gradient interpolating the same pigment colors in RGB-space.

over white and black backgrounds. In our multispectral scenario given RGB data and a fixed background, direct extraction is ill-posed. The IMPaSTo system [5] uses a low dimensional approximation of measured pigments to enable realtime rendering. In contrast, we focus on the problem of decomposing existing artwork. Xu et al. [18] use a neural network to learn to predict RGB colors from a large number of synthetic examples. RealPigment [6] estimates composited RGB colors from exemplar photos of artist color mixing charts. In our scenario, we are given the RGB colors and estimate the multispectral scattering and absorption parameters.

There is extensive work on multispectral acquisition systems using custom hardware [19]. Berns et al. [20] use a standard digital camera with a filter array. Parmar et al. [21] use a bank of LED's to capture the scene under different illumination spectra. Park et al. [22] optimize a set of exposures and LED illuminants to achieve video rates. Multispectral images have many useful applications. Ibrahim et al. [23] demonstrate intrinsic image reconstruction and material identification. Berns et al. [24] compare a multispectral imager to a point spectrophotometer for measurements of paintings. Multispectral imaging provides a non-invasive way to preserve paintings and analyze their construction. Liang et al. [25] used a combination of optical coherence tomography (OCT) imaging with multispectral
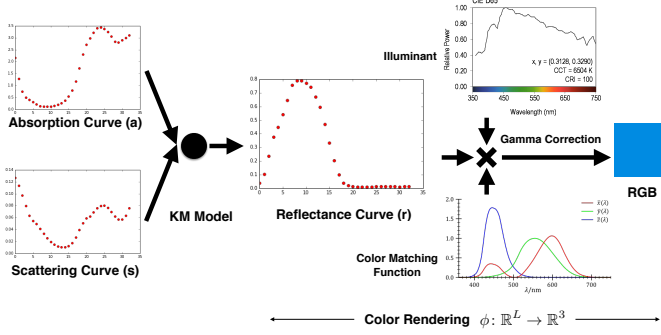
Fig. 3: Rendering from multispectral KM coefficients (absorption and scattering) to sRGB color, for cyan pigment, totalling 33 wavelengths ranging from 380nm to 700nm (every 10nm). It is rendered on pure white substrate with pigment thickness equal to 1, under D65 illuminant.

imaging to identify pigments' reflectance, absorption, and scattering parameters. Berns et al. [26] estimate the full reflectance spectrum of a painting using a reduced dimension parameterization made from spectra of known KM pigments. Zhao et al. [27] achieve better reconstructions by fitting mixtures of known pigments to estimated multispectral reflectances. Pelagotti et al. [28] and Cosentino [29] both use multispectral images as feature maps to identify single layers of known pigments. Most similar to our work, Zhao et al. [30] use multispectral measurements of Van Gogh's "Starry Night" to estimate one parameter masstone KM mixing weights for known pigments to reconstruct the painting. Delaney et al. [31] use fiber optic reflectance spectroscopy and X-ray fluorescence to help identify and map pigments in illuminated manuscripts. Abed et al. [32] described an approach to identify pigment absorption and scattering parameters and extract pigment concentration maps from a multispectral image via a simplified, one-parameter Kubelka-Munk model. *All* of these works require exotic acquisition hardware, whereas we focus on generating plausible results using standard, easy-to-obtain RGB images. There are plenty of high-quality RGB images of paintings freely available via the Google Art Project.

3 THEORY

The intuition behind this work comes from how pigments mix in real versus digital media. Digital RGB color mixing is a linear operation: all mixtures of two RGB colors lie on the straight line between them in RGB-space. Mixing two physical pigments with the same two apparent RGB colors, however, produces a curve in RGB-space (Fig. 2). The shape of this curve is a function of the multispectral Kubelka-Munk coefficients of the pigments being interpolated. Our intuition is that those multispectral coefficients can be deduced by the observed shape of a mixing or thinning curve in RGB.

3.1 Kubelka-Munk Equations

The Kubelka-Munk equations (KM) are a physically-inspired model for computing the per-wavelength reflectance value

of a layer of homogeneous pigment atop a substrate:

$$r = \frac{1 - \xi(x - y \cdot coth(yst))}{x - \xi + y \cdot coth(yst)}$$
$$x = 1 + \frac{a}{s}, \quad y = \sqrt{x^2 - 1}$$
(1)

where t is the thickness of the pigment layer, a and s are the pigment's absorption and scattering per unit thickness, ξ is the substrate reflectance, and r is the final reflectance of the pigment layer. a, s, ξ, and r are all per-wavelength, while the thickness t is constant for all wavelengths. For convenience, we use $\mathbf{k} = [\{a_\lambda\}, \{s_\lambda\}]$ to represent both KM coefficients with a single vector variable across all wavelengths λ. We denote the vectorized Equation 1 as $\mathbf{r} = \text{km}(\mathbf{k}, \xi, t)$.

Mixtures of pigments are modeled as the weighted average of KM coefficients:

$$\mathbf{k}_{mix} = \frac{\sum w_i \mathbf{k}_i}{\sum w_i}$$
(2)

where $\mathbf{k}_i$ is the i_{th} pigment parameter vector.

To render a KM pigment to RGB requires knowing the pigment's KM coefficients, the substrate reflectance, the layer thickness, the illuminant spectrum, and the color matching functions which map from a reflectance spectrum to a tristimulus value, which can then be converted to linear RGB and gamma corrected to sRGB. Figure 3 shows the pipeline for a single pigment. We use the D65 standard illuminant and CIE color matching functions [33].

Every pixel has a parameter vector $\mathbf{k}_p$. For a pixel p in the image with mixed KM coefficients $\mathbf{k}_p$, $\mathbf{r}_p = \text{km}(\mathbf{k}_p, \xi, t)$ yields a reflectance spectrum defined at each of the L wavelengths. We denote the spectrum rendering pipeline in Figure 3 as a function $\phi \colon \mathbb{R}^L \to \mathbb{R}^3$, so $\phi(\mathbf{r}_p)$ is the sRGB color for pixel p. Thus to render an image $\mathbf{I}$ we have:

$$\mathbf{I} = \phi(\text{km}(\mathbf{K}, \xi, t))$$
(3)

where $\mathbf{K}$ is the matrix of all pixels' pigment parameters.

In contrast to RGB color compositing, this model is highly non-linear and results in much more of an "organic" feel of traditional media paints as compared to digital paintings (Fig. 2).

It is important to consider the required number of wavelengths to simulate. Too many wavelengths will be difficult to optimize, whereas too few may not be able to accurately reconstruct the image appearance. We experimented with mixtures of cyan, magenta, and yellow pigments from 33 wavelengths to 3. We found that below 8 wavelengths, the color reproduction loses fidelity (Fig. 4). We can also see that the size of the RGB gamut that can be reconstructed is artificially restricted at 3 wavelengths versus 8. This is in agreement with prior work such as RealPigment [6] and IMPaSTo [5]. (Aharoni-Mack et al. [15] is based on a gamut of 3-wavelength KM pigment parameters, which is potentially more restrictive than our gamut of 8-wavelength KM pigment parameters.)

3.2 Problem Formation

A painter creates a palette from a set of e.g. tubes of paint, which we call the *primary pigments*. Every color in the painting is a mixture of these primary pigments. Therefore,
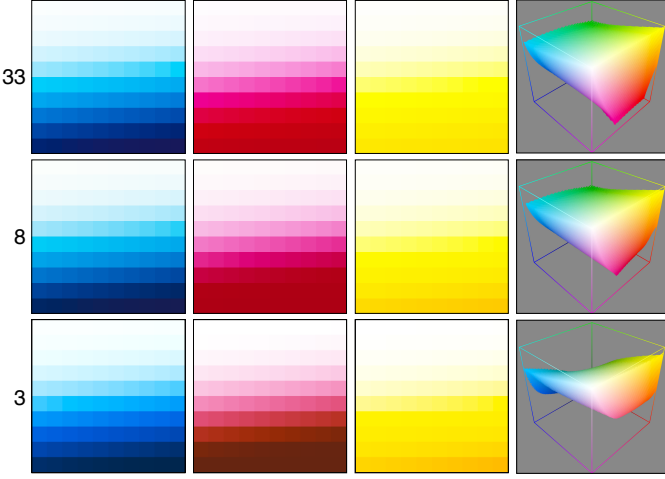
Fig. 4: Visualizing rendering with different numbers of wavelengths. The original cyan, magenta, and yellow pigment coefficients, sampled at 33 wavelengths between 380nm and 700nm, are downsampled to 8 and 3 wavelengths and rendered with varying thickness. The RGB gamuts achieved by mixing them are plotted. The 8 wavelength gamut appears similar to the 33 wavelength gamut, but the 3-wavelength gamut is significantly distorted.

mixtures of the primary pigments' KM coefficients are sufficient to reproduce the RGB color of every pixel in the painting. Our method estimates the coefficients of a small set of primary pigments to minimize the RGB reconstruction error.

For L wavelengths, each primary pigment $\mathbf{k}_m$ is a vector of $2L$ coefficients. We represent the set of M primary pigments as an $M{\times}2L$ matrix $\mathbf{H} = [\mathbf{k}_1, \mathbf{k}_2, \ldots \mathbf{k}_M]^T$. Every pixel in the painting can be represented as a convex combination of these primary pigments, $\mathbf{w} \cdot \mathbf{H}$, where $\mathbf{w}$ is the $1{\times}M$ vector of mixing weights ($\mathbf{0}_{1{\times}M} \leq \mathbf{w} \leq \mathbf{1}_{1{\times}M}$). We can express all N pixels in the image as the matrix product $\mathbf{K} = \mathbf{WH}$, where the $\mathbf{w}$ form the rows of the $N{\times}M$ matrix $\mathbf{W}$. Eq. 3 becomes:

$$\mathbf{I} = \phi(\mathrm{km}(\mathbf{WH}, \xi, t)) \qquad (4)$$

where $\mathbf{I}$ is the $N{\times}3$ matrix of our painting's per-pixel RGB colors.

To simplify the problem, we assume the canvas is pure white ($\xi = 1$). We further assume that the entire canvas is covered with a single layer of constant thickness $t = 1$ paint, where each pixel's paint is a weighted mixture of pigments. Thus, our equation becomes:

$$\mathbf{I} = \phi(\mathrm{km}(\mathbf{WH})) \qquad (5)$$

We use Eq. 5 to pose an optimization problem:

$$E_{data} = \|\mathbf{I} - \phi(\mathrm{km}(\mathbf{WH}))\|^2$$
$$E_{sum} = \|\mathbf{W}\mathbf{1}_{M{\times}1} - \mathbf{1}_{N{\times}1}\|^2 \qquad (6)$$
$$\mathbf{W}^*, \mathbf{H}^* = \mathrm{argmin}\{E_{data} + w_{sum}E_{sum}\}$$

where $\mathbf{1}_{M{\times}1}$ and $\mathbf{1}_{N{\times}1}$ are column vectors of ones, and subject to the constraints $\mathbf{0}_{N{\times}M} \leq \mathbf{W} \leq \mathbf{1}_{N{\times}M}$ and $\mathbf{H} > \mathbf{0}_{M{\times}2L}$. E_{sum} forces our per-pixel weights to sum to one, since each pixel's coefficients are a convex combination of the primary pigments. As an alternative to E_{sum}, one could use

$W = \mathrm{softmax}(\mathbf{W}')$ in E_{data}. This would allow unconstrained variation of $\mathbf{W}'$ while maintaining that the weights (rows) of $\mathbf{W}$ sum to one. However, in our experiments we found that E_{sum} has better convergence properties.

We make the assumption that thickness $t = 1$, because we are primarily focused on acrylic and oil paints, which are quite thick, especially as compared to watercolor. Our assumption means that we cannot capture impasto effects or thin watercolor effects accurately. Note, however, that the choice of *which* constant thickness value to use is arbitrary. Thickness t appears in the KM equations as a scale factor for s, but neither a nor s appear elsewhere except as a ratio. Therefore, changing the constant thickness t to another value is equivalent to uniformly scaling all a and s.

Allowing the thickness to vary introduces an additional degree-of-freedom per pixel. Figure 5 shows an experiment in which we solve for two pigments' multispectral a and s parameters and per-pixel mixing weights; we optionally allow thickness to vary per-pixel. When thickness varies, the problem is under-constrained. To make the problem tractable, we add a smoothness regularization term. However, this leads to incorrect thickness estimation and less accurate multispectral reflectance (and slower optimization performance). While varying thickness may be particularly useful for watercolor or translucent paint, we did not pursue it in our thick-paint scenario beyond these initial experiments.

3.3 Solution Space

In Eq. 6, $\mathbf{W}$ and $\mathbf{H}$ are both unknown, so we have $NM + 2LM$ unknown variables and $3N + N$ known equations, which makes our problem under-constrained for $M > 3$. We can use regularization to make the problem over-constrained. While this results in *a* solution, there are infinitely many solutions to the problem as originally stated for any particular image. This is for two reasons.

First, $\phi(\cdot)$ projects from L-dimensional reflectance spectra to 3-dimensional *tristimulus* values. For any given tristimulus value there are infinitely many possible spectra (metamers) that could produce it. This is analogous to seeing only the 2D projection or "shadow" of a 3D curve. No matter how many high dimensional samples we obtain, ϕ projects them all in parallel.

Second, if there exists $\mathbf{G}$ s.t. $\mathbf{WH} = \mathbf{WGG}^{-1}\mathbf{H}$, for $\mathbf{0}_{N{\times}M} \leq \mathbf{WG} \leq \mathbf{1}_{N{\times}M}$, and $\mathbf{G}^{-1}\mathbf{H} > \mathbf{0}_{M{\times}2L}$, then $\mathbf{W}' = \mathbf{WG}$ and $\mathbf{H}' = \mathbf{G}^{-1}\mathbf{H}$ is another solution that generates the same RGB result. In a simple geometric sense, $\mathbf{G}$ could be a rotation or scale of the KM coefficients associated with the primary pigments. So long as the set of observed pigment parameters all lie within the polytope whose vertices are the rows of $\mathbf{H}$, then e.g. rotations and scales that maintain that property will also produce solutions. If the colors are near the edges of the gamut, or the pigment parameters are near the edges of the KM space (i.e. have small values in the KM coefficients), then there will be very little "wiggle room" for the pigments to move. Conversely, if the set of observed pigment parameters are compact (i.e. no KM coefficients near zero), then many different gamuts may be possible.

4 METHOD

Our naively posed optimization problem (Eq. 6) is too slow to run on an entire, reasonably-sized image at once. To
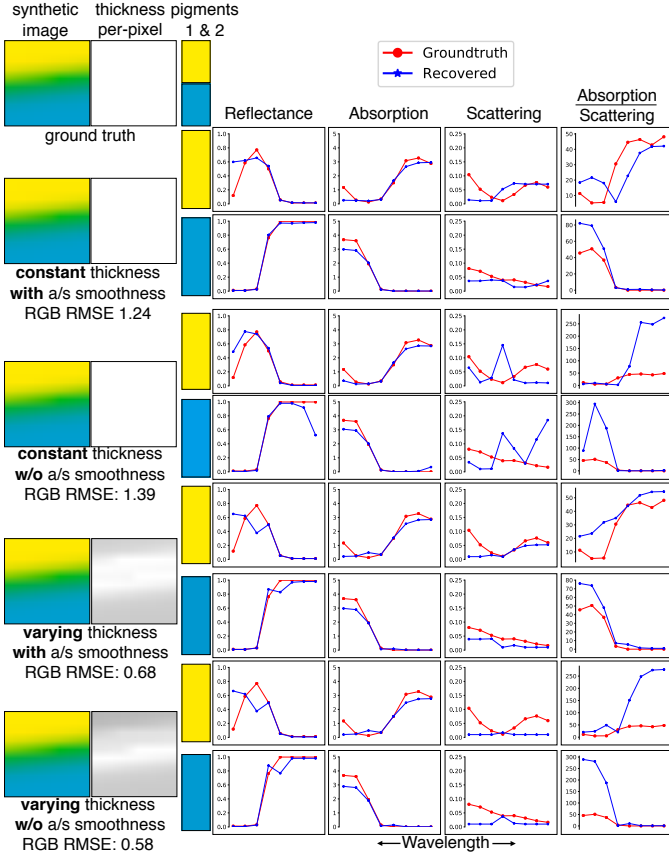
Fig. 5: The effects of constant versus varying thickness paint and our $\frac{a}{s}$ masstone smoothness term (Equation 8). The "synthetic image" column shows the reconstruction of the ground truth image using two pigments' recovered absorption and scattering parameters shown at right. Allowing paint with varying thickness results in an underconstrained problem. With a smoothness regularization term, the solution deviates from ground truth in our experiments. The masstone smoothness term results in scattering parameters that more closely match ground truth.

improve performance, we decompose our task into two subproblems: estimating primary pigments and estimating per-pixel mixing weights.

4.1 Estimating Primary Pigments

The first step in our pipeline is to estimate a set of primary pigment coefficients $\mathbf{H}$ that can reconstruct the painting. Even for small input images of 0.25 megapixels, doing this estimation over every pixel in the image would be computationally very expensive. We observe that it is not necessary to consider every pixel, since many pixels contain redundant information. Therefore, we optimize over a small subset of representative pixels, carefully chosen to well-represent the image's color properties.

To find a small subset of representative pixels $\mathbf{I}_{subset}$, we find the 3D convex hull of the set of RGB colors in the image using the QHull algorithm [34]. For the images we tested, this usually results in a few hundred unique colors. These pixels are particularly well suited to the task of estimating the primary pigments because they span the full gamut of

the painting's colors. They are guaranteed to include the most extreme combinations of pigments. Conversely, pixels in the interior of the convex hull are less distinct, resulting in less vibrant recovered primary pigments.

The optimization problem as posed in Eq. 6 is very similar to non-negative matrix factorization—which is non-convex—with the added non-linearities of the KM equation and gamma correction. Therefore, we use the Alternating Nonlinear Least Squares (ANLS) method for our optimization.

In the first step, we fix the set of primary pigment coefficients $\mathbf{H}$ and solve for the mixing weights $\mathbf{W}_{subset}$:

$$\mathbf{W}^*_{subset} = \operatorname{argmin}\{E_{data} + w_{sum}E_{sum}\} \qquad (7)$$

with the constraint $\mathbf{0}_{N_1 \times M} \leq \mathbf{W}_{subset} \leq \mathbf{1}_{N_1 \times M}$, where N_1 is the number of representative pixels, and M is the number of pigments, and $w_{sum} = 10.0$.

In the second step, we fix $\mathbf{W}_{subset}$ and solve for $\mathbf{H}$. When estimating the primary pigments, we add an additional regularization term to avoid creating physically implausible pigment coefficients. Specifically, KM pigment coefficients a and s should vary smoothly across wavelengths [6], and the ratio $\frac{a}{s}$ (which determines the pigment's masstone) should also vary smoothly across wavelengths (Figure 5). We encode these smoothness observations as:

$$
\begin{aligned}
E_{smooth} = \frac{N}{M(L-1)} \sum_{i=1}^{M} \sum_{\lambda=1}^{L-1} &\bigg(w_a(a_{i,\lambda} - a_{i,\lambda+1})^2 \\
&+ w_s(s_{i,\lambda} - s_{i,\lambda+1})^2 \\
&+ w_{ratio}\left(\frac{a_{i,\lambda}}{s_{i,\lambda}} - \frac{a_{i,\lambda+1}}{s_{i,\lambda+1}}\right)^2 \bigg)
\end{aligned} \qquad (8)
$$

over all M primary pigments and L wavelengths, where $w_a = w_s = 1$ and $w_{ratio} = 0.001$ control the relative influence of the terms. Putting it all together, our optimization for the second step is:

$$\mathbf{H}^* = \operatorname{argmin}\{E_{data} + w_{smooth}E_{smooth}\} \qquad (9)$$

with the constraint $\mathbf{H} > \mathbf{0}_{M \times 2L}$, where M is the number of pigments, and L is the number of wavelengths, and $w_{smooth} = 0.001$.

Initialization As with any non-convex optimization problem, initialization is a key factor in how quickly the solution converges and whether a local minimum is found. In our case, the solution is not unique, so there are many potential minima to converge upon. Therefore, initialization is very important for finding a good solution.

One option is to initialize randomly, which produces somewhat unpredictable results, though we do find that plausible solutions (where the colors of the primary pigments roughly match a painter's expectations) are often well-represented. In the absence of other information, we can present the user with the results of e.g. ten random initializations to choose their preferred solution.

An alternative is to take advantage of a prior, such as a natural distribution of real pigments in KM space. We use a set of 26 measured acrylic paints from Okumura [16]. When the prior is similar to the pigments used in the painting, reconstruction often finds the approximately correct KM coefficients. When the prior is of a different media than
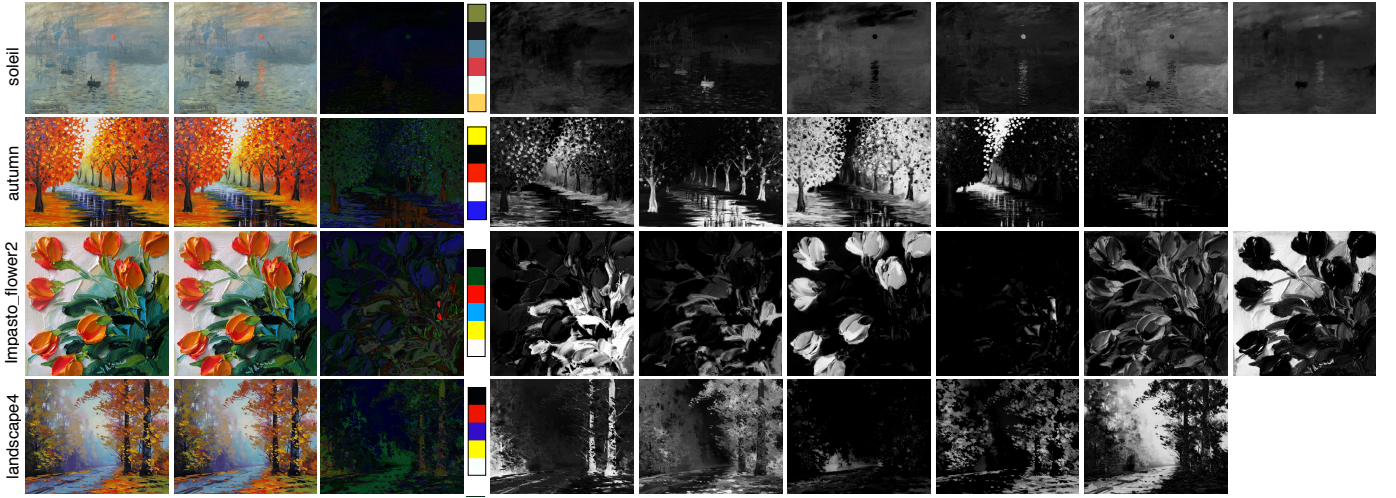
Fig. 6: Our results for multiple images, one per row. From left to right the columns are the original image, the reconstruction, the error (10x), the extracted palette, and the mixing weight maps. Because our pigments are multispectral, we show them as RGB colors rendered on a white canvas with unit thickness. From second to end: © MontMarteArt, Jan Ironside, Graham Gercken.

the painting (e.g. using an acrylic prior with a watercolor painting), then while the result will have low reconstruction error and look plausible, the mixing properties of the pigments may not be correct (e.g. a watercolor painting may have more opaque pigments in the reconstruction than in reality). When multiple such priors are available, a user could select the correct prior to use for a given painting. In our experiments, we rely on the dictionary of Okumura's 26 acrylic pigments. To boost the size of the dictionary, we also include every pair of pigments mixed 50%, for a total of 351 entries.

To initialize $\mathbf{H}$ with M pigments using our prior, we start from the convex hull of the RGB colors in the image. We simplify the convex hull to M vertices as in Tan et al. [10]. We then match these M RGB colors to the closest matching (Euclidean distance) RGB colors in the dictionary and use the corresponding KM coefficients. The RGB color of a dictionary pigment is obtained by rendering with the same pipeline as Eq. 3 (with thickness $t = 1$ and substrate $\xi = 1$). If two convex hull colors match to the same dictionary color, the closer match is used and the other convex hull vertex matches to its second closest dictionary color.

4.2 Estimating Mixing Weights

The second step of our pipeline uses the estimated set of primary pigments to compute per-pixel mixing weights for the entire image. We use observations about the nature of painting construction to add additional regularization terms, improving convergence and making the results more useful for editing applications.

First, we add a term for per-pixel weights sparsity [10], which encourages each pixel's M pigment weights to be close to 0 or 1:

$$E_{sparse} = -\frac{1}{M} \left\| \mathbf{1}_{N \times M} - \mathbf{W} \right\|^2 \qquad (10)$$

where $\mathbf{1}_{N \times M}$ is matrix of ones. This term has the effect of maximizing color separation throughout the painting, so that

each pigment influences as small a portion of the image as possible. This is desirable because it results in more localized pigment editing operations.

Second, we add a term for spatial smoothness of the weights:

$$E_{spatial} = \frac{1}{M} \left\| \mathbf{SW} \right\|^2 \qquad (11)$$

where $\mathbf{S}$ is a Laplacian or a bilateral smoothing matrix. We use a bilateral operator [35] in order to preserve edges that appear between brush strokes of different colors of paint. In our experiments, the Laplacian operator blurred edges and caused pigments to incorrectly bleed into image regions.

With these additional terms, our optimization to reconstruct mixing weights becomes:

$$\mathbf{W}_{all}^* = \operatorname{argmin}\{E_{data} + w_{sum}E_{sum} + w_{sparse}E_{sparse} \\ + w_{spatial}E_{spatial}\} \qquad (12)$$

where $w_{sum} = 10$, $w_{sparse} = 0.1$, and $w_{spatial} = 1$ subject to $\mathbf{0}_{N \times M} \leq \mathbf{W}_{all} \leq \mathbf{1}_{N \times M}$, where N is the number of pixels in the entire image and M is the number of pigments.

This optimization is still very large and difficult to solve directly. Instead, we solve it in a coarse-to-fine manner. We downsample the image by factors of two until the short edge is less than 80 pixels. We solve the optimization on the smallest image, initializing each pixel's mixing weights to $1/M$. We upsample each solution (mixing weights) as the initialization for the next larger optimization. We repeat this procedure to obtain a solution for the original image.

Pseudocode for our method can be found in Algorithm 1. Computational complexity is difficult to analyze because our approach is based on iterative nonlinear optimization Runtime performance is dominated by the optimization for all pixels' weights (line 15), as discussed in the following section. To evaluate the convergence of our pigment parameter estimation's two alternating optimization steps, we measure the total energy ($E_{data} + w_{smooth}E_{smooth} + w_{sum}E_{sum}$) per iteration. Fig. 7 plots this for six of the examples used in Table 1. In all

Algorithm 1: Extract mixing weights and multi-spectral pigment parameters from single RGB image

Input: RGB Image $\mathbf{I}_{N \times 3}$ and user-provided number of primary pigments M.
Output: Primary pigment KM parameters $\mathbf{H}_{M \times 2L}$ and mixing weights $\mathbf{W}_{N \times M}$.

1 $\mathbf{I}_{subset} \leftarrow$ vertices of ConvexHull($\mathbf{I}$)
2 $\text{palette}_{RGB} \leftarrow$ Simplify(ConvexHull($\mathbf{I}$), M)
3 $\mathbf{H}^{(0)} \leftarrow$ ClosestColors(palette_{RGB}, Okumura mixtures)
4 $i \leftarrow 0$
5 **while** *true* **do**
6    $\mathbf{W}^{(i)}_{subset} \leftarrow$ Solve Equation 7($\mathbf{I}_{subset}$, $\mathbf{H}^{(i)}$)
7    $\mathbf{H}^{(i+1)} \leftarrow$ Solve Equation 9($\mathbf{I}_{subset}$, $\mathbf{W}^{(i)}_{subset}$)
8   // Terminate upon small relative change in $\mathbf{H}$
9   // Absolute value and min() are element-wise.
10   **numerator** $\leftarrow |\mathbf{H}^{(i+1)} - \mathbf{H}^{(i)}|$
11   **denominator** $\leftarrow \min(\mathbf{H}^{(i+1)}, \mathbf{H}^{(i)})$
12   // Maximum is over all elements.
13   **if** $maximum(\textbf{numerator}/\textbf{denominator}) < 0.001$ *or* $i = 1000$ **then**
14      $\mathbf{H}^* \leftarrow \mathbf{H}^{(i+1)}$
15   break
16   **end**
17    $i \leftarrow i + 1$
18 **end**
19 $\mathbf{W}^*_{all} \leftarrow$ Solve Equation 12($\mathbf{I}$, $\mathbf{H}^*$)
20 **return** $\mathbf{H}^*$, $\mathbf{W}^*_{all}$

Fig. 8: Recovering ground truth. Our reconstruction has low RGB error and the palette and mixing weight maps are similar upon inspection. The graphs of spectral curves show reflectances are recovered well, but absorption and scattering less so. Numeric results are in Table 2. Ground truth curves are dashed, recovered are solid, and colors correspond to palette colors. Note: ground truth black coefficients are plotted with a scale factor of 0.2 to achieve a similar range as the other pigments. © Nel Jansen.

5 RESULTS

To demonstrate our results, we conducted a series of experiments on synthetic and real images, comparing amongst different conditions and with previous work. All tests were run on a single core of either a 2.53 GHz Intel Xeon E5630 or a 2.50 GHz Intel Core i7-4870HQ, implemented in Python using the L-BFGS-B [36] solver. Runtime information is presented in Table 1, which shows that we are generally faster than Tan et al. [10]. Once the primary pigments and mixing weights are estimated, all of our editing applications occur in realtime.

TABLE 1: Performance data for Fig. 6 and 10. Our pipeline extracts M primary pigments in a few seconds and mixing weights maps in less than 10 minutes for a normal size image, with low RGB image reconstruction error.

Image	Size	M	CPU	Primary (sec)	Weights (sec)	RGB RMSE
soleil	600×467	6	i7	35	155	0.007
autumn	600×458	5	xeon	16	255	0.024
four_colors_2	600×598	4	i7	9	211	0.020
Impasto_flower2	595×600	6	xeon	44	615	0.02
landscape4	600×479	5	xeon	26	256	0.018
portrait2	600×441	6	xeon	29	243	0.017
tree	600×492	4	i7	14	151	0.016

Synthetic Data We used synthetic images to evaluate our pipeline's recovery performance against ground truth. We used our pipeline to obtain weight maps from a painting. We then created five synthetic paintings by randomly choosing sets of pigments from a dataset of measured multi-spectral KM coefficients of real acrylic paint [16]; mixing them according to our weight maps; and rendering them to sRGB using Eq. 3. These five synthetic paintings appear to depict the same flower painted with different colors. To make our initialization fair, we used a hold-one-out methodology for the pigment dictionary: we removed the five pigments used to construct the synthetic image from the set of candidate pigments for initialization, leaving a dictionary of 21 (plus
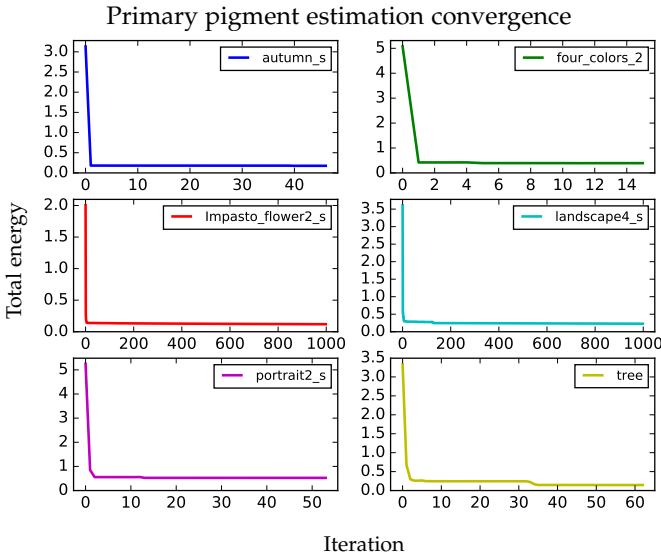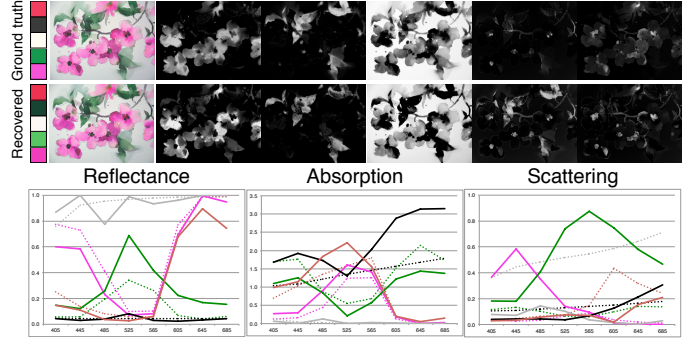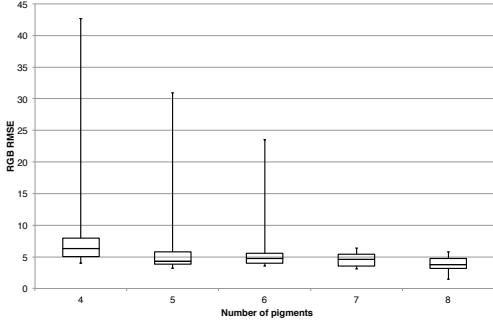

Fig. 7: The total energy for our primary pigment estimation decreases monotonically per iteration ($E_{data} + w_{smooth}E_{smooth} + w_{sum}E_{sum}$). Each iteration performs both alternating steps, minimizing Equations 7 and Equation 9.

examples, the energy decreases rapidly after a few iterations. Some examples reach the maximum number of iterations rather than our strict convergence criteria (Algorithm 1).

Fig. 9: We plot the distribution of RGB RMSE of 12 example images' reconstructions on different palette size. Generally, RMSE will decrease when palette size increase, and RMSE distribution deviation will decrease when palette size increase.

mixed pigments, so 231 in total). Fig. 8 shows one example of our five synthetic images and its recovery. All reconstruction errors are presented in Table 2.

The results of this experiment are that the pigment coefficients a and s have relatively high error, where for our measured pigments, $a \in [0, 10]$ and $s \in [0, 1]$. Our reflectance spectra r have lower recover error, $r \in [0, 1]$, because there are many values of a and s that can create the same appearance r (metamers). Since the pigments are different from ground truth, the recovered weight maps $\mathbf{W} \in [0, 1]$, are different as well. However, the RGB image's reconstruction error is small, generally below the noticeable threshold. We also tested the weight map recovery step in isolation by using ground truth pigments to estimate $\mathbf{W}^*$, which has a smaller but still significant recovery error. The final RGB* image reconstruction error stays low. This experiment confirms that there are many solutions to our reconstruction problem, but that we are able to reproduce plausible values.

TABLE 2: Reconstruction errors for synthetic data experiments (Fig. 8) with constant weight maps and different pigments. Each reported number is RMSE, for pigment absorption a and scattering s coefficients and reflectance r. From those pigments, weight map $\mathbf{W}$ and RGB image are recovered. To test weight map recovery in isolation, $\mathbf{W}^*$ and RGB* use the ground truth pigments. Because there are many solutions (Section 3.3), we cannot recover ground truth parameters (a,s,$\mathbf{W}$). However, the RGB image's reconstruction error is always small and unnoticeable.

Exp	a	s	r	$\mathbf{W}$	$\mathbf{W}^*$	RGB	RGB*
1	6.1	1.2	0.3	0.114	0.060	0.019	0.023
2	1.4	0.9	0.3	0.078	0.046	0.027	0.017
3	4.5	0.5	0.7	0.247	0.084	0.026	0.023
4	7.1	1.2	0.6	0.166	0.055	0.033	0.024
5	1.0	0.7	0.3	0.065	0.041	0.023	0.020
Mean	4.0	0.9	0.4	0.134	0.057	0.026	0.021
Std	2.7	0.3	0.2	0.074	0.017	0.005	0.003

Influence of Palette Size Since the number of primary pigments is not automatically determined by our algorithm, we evaluated a set of images over a wide range of palette sizes (Fig. 9). Unsurprisingly, as the number of pigments
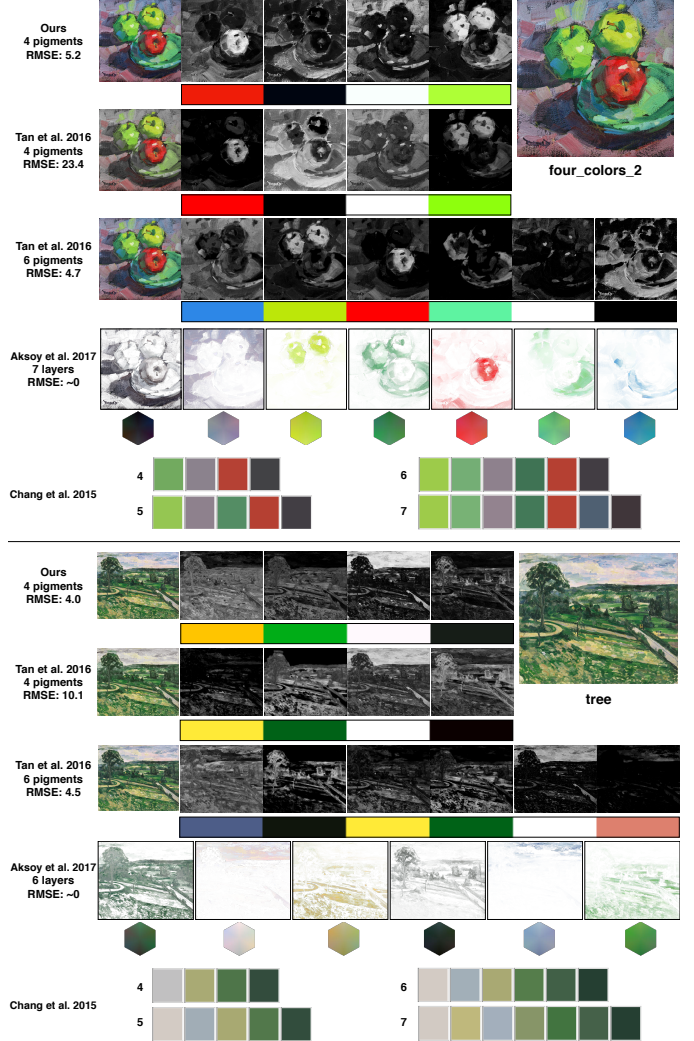

Fig. 10: Comparison with the layer decompositions of Tan et al. [10] and Aksoy et al. [12], and with the palettes extracted by Chang et al. [14]. The upper four_colors_2 example was painted with exactly four physical pigments. When constrained to four colors, Tan et al.'s approach has higher reconstruction error. To match our reconstruction error, Tan et al.'s approach needs to use more colors. Aksoy et al.'s approach extracts layers guaranteed to have zero reconstruction error, but the extracted layers are not composed of a single color. Chang et al.'s approach extracts a palette whose size is manually chosen by the user. For the four_colors_2 image (top), Chang et al.'s palettes never contain the known ultramarine blue pigment, even for very large palettes. Top example: © Cathleen Rehfeld.

increases, aggregate reconstruction error decreases. Interestingly, each image seems to have a number of pigments beyond which the RMSE stops decreasing. Intuitively, this would be the natural number of pigments in the painting. For paintings with very large numbers of pigments, it is unlikely that this property would hold, as eventually a large set of primary pigments would be over-complete and no additional information could be gained. However, painters often use relatively small palettes in practice.

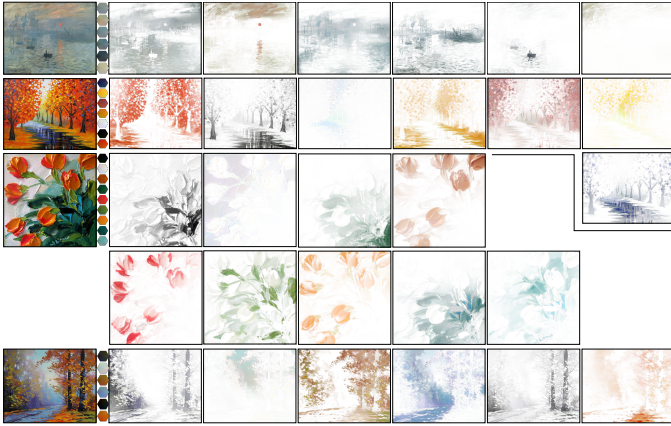**Physical Paintings** We show our pipeline running on

Fig. 11: The approach of Aksoy et al. [12] applied to the same examples as in Figure 6. The columns show the input image, their extracted palettes, and their layers. Reconstructions are not shown, because Aksoy et al.'s approach has no reconstruction error. This is because their palettes contain color distributions, not single colors. As a result, their layers are sometimes quite colorful and difficult to edit. The approach automatically chooses a palette size balancing choosing larger (sometimes redundant) palettes with less colorful layers. From second to end: © MontMarteArt, Jan Ironside, Graham Gercken.
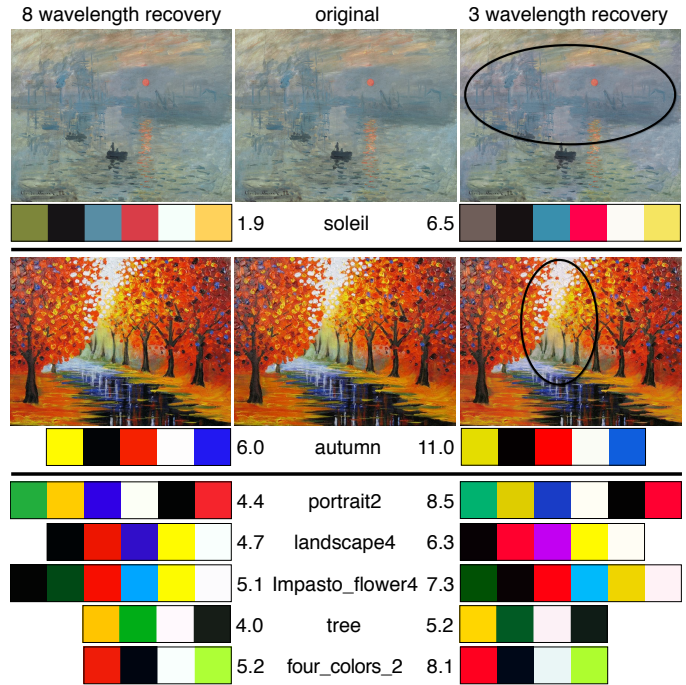

Fig. 12: Comparison of 3 and 8 wavelength recovery, with RGB RMSE. We find 3 wavelength reconstruction error is higher for all examples. Soleil and autumn example show color distortion. Second example: © MontMarteArt.

scans of physical paintings in Figures 6 and 10: extracted primary pigments, weight maps and reconstructed RGB images. Reconstruction errors are reported in Table 1. We reconstructed with 4 to 6 pigments for every example, but only show the result with the smallest palette that produced low reconstruction error. Painting four_colors_2 is known to have been created with only four paints: titanium white, cadmium yellow lemon, cadmium red, and ultramarine blue. Our extracted palette's RGB colors are very similar, though the yellow is a bit greenish and the blue is dark.

Comparison to Tan et al. [10]. Our algorithm uses multispectral pigments with the nonlinear KM model, in contrast to previous work like Tan et al. [10], which solves a similar problem using a linear compositing RGB model. Intuitively, we would expect that our model would be able to reconstruct paintings at lower error with fewer parameters. The experiment we show in Fig. 10 confirms this. For two paintings, our technique is able to reconstruct the images with low error using only four pigments. Tan et al. [10]'s algorithm results in much higher error for the same number of colors. In order to achieve a similar RGB reconstruction error, Tan et al. [10] must increase the palette to six colors. In general, it is easier to edit a painting with a smaller palette.

Comparison to Aksoy et al. [12]. Figures 10 and 11 show the same examples decomposed using the approach of Aksoy et al. [12]. Their approach extracts additive linear RGB mixing layers. The layers contain color distributions, not single colors, though their approach guarantees zero reconstruction error. The number of layers is automatically selected, balancing the colorfulness of layers against smaller palettes. The colorful layers are difficult to edit, since color distributions must be modified.

Comparison to Chang et al. [14]. Figure 10 shows palettes for the same input images extracted by the approach of Chang et al. [14]. In their approach, the palette size is manually chosen by the user. For the four_colors_2 image with known four ground truth pigments, even very large palettes never contains a color similar to ultramarine blue. Instead, "redundant" colors are added.

Influence of Wavelengths Our pipeline recovers 8 wavelength pigment absorption and scattering coefficients, because of the experiment in Figure 4 that shows a limited RGB gamut for 3 wavelength rendering. For completeness, we compare with 3 wavelength recovery in Figure 12. As 3 wavelength recovery is not multispectral anymore, we slightly amend our model equation, Eq. 3: we change the illuminant from D65 to pure white ($\mathbf{1}_{3\times1}$), and we set the color matching function to be the identity matrix ($\mathbf{I}_{3\times3}$). This has the effect of directly mapping the a and s coefficients to the RGB color channels, as done by Curtis et al. [4].

We find that 3 wavelength recovery has larger RGB reconstruction RMSE for the same size palettes in all of our experiments, though many of the achieved errors are still low enough to be generally unnoticeable. For some images, such as the two pictured in Fig. 12, there is obvious color distortion. We believe this is due to the restricted gamut of the 3 wavelength pigment model, which has a significant (visible) impact only on paintings that include colors in those extreme portions of the gamut, notably certain greens and reds. For paintings with colors entirely within the 3 wavelength gamut, the differences will be negligible.

We also compare 3 wavelength recovery with linear RGB of Tan et al. [10] on example images tree and four_colors_2 in Fig. 10 and 12. The 3 wavelength KM recovery still produces better RGB reconstruction error for the same number of
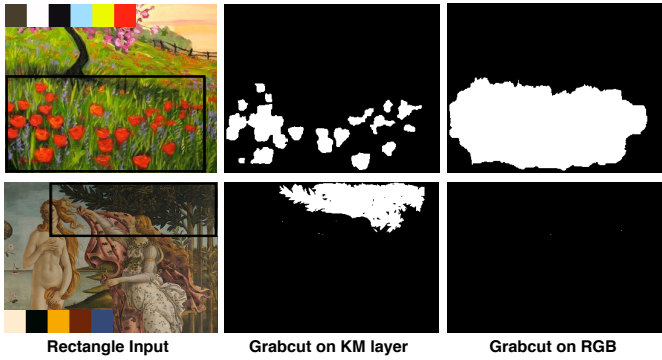
Fig. 13: GrabCut on selected KM pigment mixing weights (top: red, bottom: black) outperforms GrabCut on the RGB image. First example: © Patty Baker.

colors than the linear model.

6 APPLICATIONS

Once we have analyzed a painting to extract its primary pigments (inset for most figures) and mixing weight maps, we can re-pose a number of image editing operations in pigment space to enable interesting paint-aware applications.

6.1 Masking

Selection masking in images of paintings can be improved by optimizing on pigment weights instead of RGB colors. Semantic image boundaries are likely to correspond with changes in paint, whereas RGB edges may be less obvious, when different paint mixtures are used to paint distinct objects. Also, paint thickness can create lighting variations across the surface of the painting that can confuse RGB boundary analysis. We demonstrate a standard GrabCut [37] implementation on two paintings on pigment maps vs. RGB values in Figure 13, which clearly shows improved localization of painted objects in the black rectangular regions. GrabCut was performed on the red pigment for the top painting, and on the black pigment for the bottom painting. No background and foreground scribbles are provided to the GrabCut algorithm.

6.2 Adjustments

The pigment mixing maps provide a novel parameterization for image edits that may be difficult in RGB, by adjusting the relative values of the mixing weights, or adjusting the coefficients of the extracted pigments. First, we can vary the weight of any of the extracted pigments by scaling its map up or down and optionally re-normalizing the per-pixel weight sum to one. For a painting that has e.g. yellow pigment, this change corresponds to varying the amount of yellow in the image, in a way that would be difficult to reproduce using the features of a digital image manipulation program (Fig. 14).

Similarly, most extracted palettes include some white and black pigments for creating tints and shades. Adjusting the relative weights of these pigments is akin to adjusting the brightness and contrast of an image, but again with different results. For example, in Fig. 15a, the result of increasing the
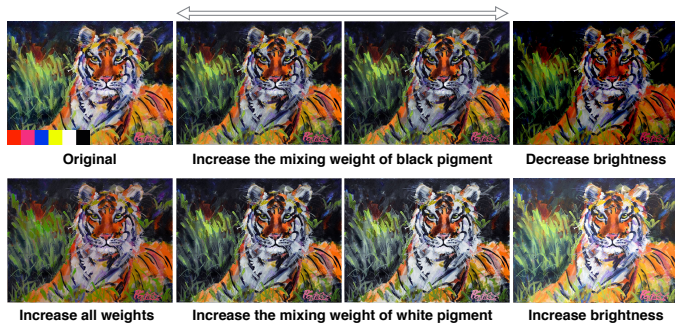

Fig. 14: Adjusting the absolute mixing weight of a pigment without re-normalizing the weight sum creates variations that would be difficult to reproduce in RGB. First and second: © Graham Gercken, John Larriva.

black weight is more like emphasizing shadows and detail, instead of just darkening, while the result of increasing the white weight is desaturation of the colors.

The KM coefficients of the pigments can also be relatively adjusted for interesting effects. Fig. 15b shows scaling the per-wavelength scattering coefficients of the green pigment, while keeping absorptions constant. Increasing scattering means that more light will be reflected back, so in some sense this is similar to brightening the green and making it more opaque, while decreasing scattering creates a darker green that absorbs more than it scatters, so perhaps more like a stained glass. Changing the scattering coefficients produces different hues of green compared to manipulating the pigment mixing weight (rightmost image in Fig. 15b).

6.3 Recoloring

Previous work focused on recoloring images by changing the extracted palette colors. Tan et al. [10] reconstructs each image pixel as a set of RGB layers, so changing a palette color has a straightforward impact on the resulting image. Our recoloring result is similar to Tan et al. [10], with the difference being that we replace KM pigments in the extracted palette with other KM pigments (from Okumura [16]) and re-render the image, creating different mixed colors in the style of real traditional media paints. Fig. 16a shows three examples. To enable a more direct comparison, we use our extracted palette RGB colors as the layer colors in Tan et al. [10]. In the cat painting, the

(a) Adjusting the absolute mixing weights of black and white pigments. The effects (middle two images) are different from adjusting the brightness level using photo manipulation software (rightmost image). Increasing the mixing weights of all layers (bottom left image) results in pigments reaching their respective masstones.© Pamela Gatens
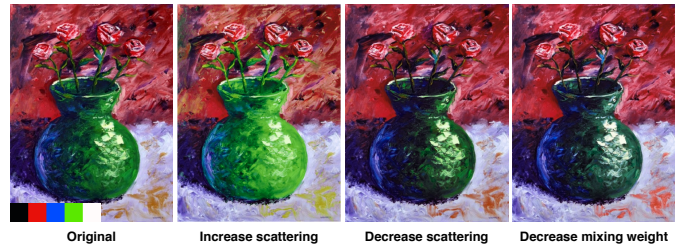

(b) Adjusting the scattering cofficients of the green pigment. Changing the scattering coefficients produces different effect from manipulating the mixing weights (rightmost image). © Mark Adam Webster

Fig. 15: Tonal adjustments in pigment space.

KM mixing weight map for the blue pigment is sparse and therefore the recoloring effect is localized on the body of the cat. The weight map from Tan et al. [10] has non-zero values in the background resulting in recoloring artifacts. For the rooster painting, using our KM model, more vibrant green is obtained from mixing yellow and blue in the circled region. For Starry Night, when swapping the extracted yellow pigment with a different yellow, the KM recoloring result reveals the green hue in the new yellow pigment, whereas the RGB recoloring result is similar to the original painting since the two yellow pigments have similar masstones in RGB space. Fig. 16b shows a different recoloring comparison between Tan et al. [10], Chang et al. [14] and ours. Both Tan et al. [10] and Chang et al. [14] have color artifacts when using their own pipelines to recolor the painting to be similar to our result.

6.4 Cut, Copy, Paste

From a selection mask, we can use the pigment weight maps to do painterly cut, copy, and paste operations on images as well. For copy-paste, the user can specify a mask (using any mechanism) and the subset of pigments to copy. The selected region can then be pasted elsewhere as a new layer of paint on top of the image and re-composited. The paste operation can adjust paint properties simultaneously such as the thickness of the pasted layer, to achieve different compositing results, or can be added into the mixture model as additional paint mixed into the painting layer, with relative scaling and renormalization. These options result in different painterly variations on standard image copy-pasting (Fig. 17 and Fig. 1).
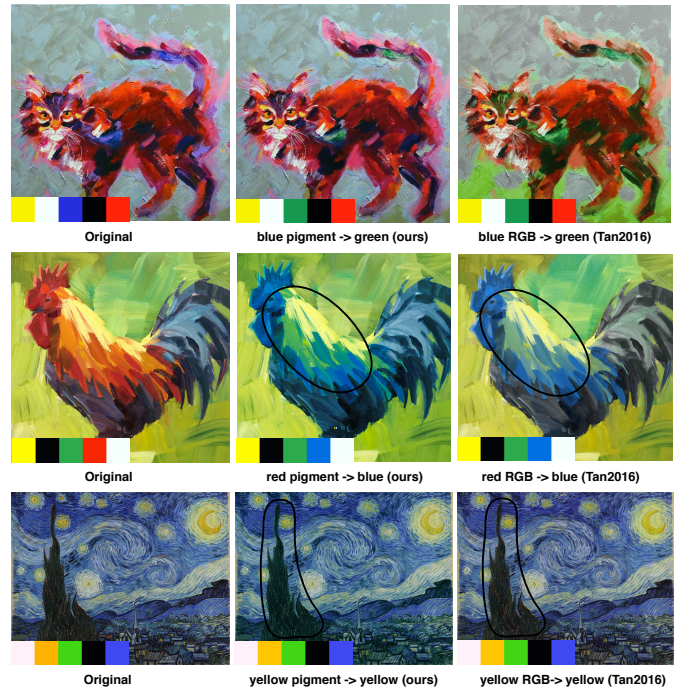
(a) We use our palette's RGB colors for layers in Tan et al. [10] for direct comparison. Top: blue is replaced by green. Middle: red is replaced by blue. Bottom: yellow is replaced with a different yellow. First and second: © Pamela Gatens, Patti Mollica.
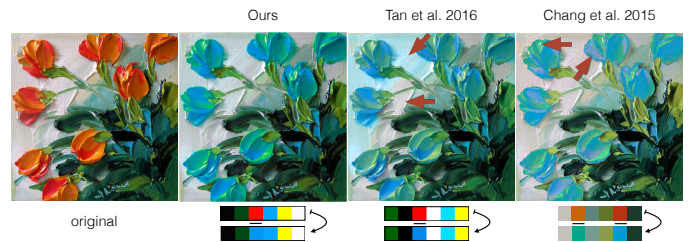
(b) Each method extracts its own palette from the input image, so we attempt to mimic our result as closely as possible. Tan et al. [10] suffers from lack of sparsity, while Chang et al. [14] has surprising local colors (red arrows). © Jan Ironside.

Fig. 16: Recoloring comparisons.

The cut operation deletes the selected pixels' pigments from the painting, for which inpainting fills the resulting hole (Fig. 1). We use a fast marching method [38], though alternatives such as PatchMatch [39] would also work, so long as they can operate on arbitrary numbers of image channels.

6.5 Palette Summarization

The first stage of our algorithm can also be seen as yet another method for extracting a small palette from an arbitrary image, not necessarily of paintings. Tan et al. [10] and Chang et al. [14] both present palette-extraction methods, as does Adobe's Color CC app [40]. We compare these results in Fig. 18a, where it is clear to see that Chang et al. [14] and Kuler attempt to find "salient" or meaningful colors in some sense, whereas Tan et al. [10] and our work focus on colors that reconstruct the images. We achieve similar results to Tan et al. [10], but as we showed earlier our reconstructions are much lower error for the same number of colors, as we have

Fig. 17: Results of copy paste in pigment space. Each classical painting has been modified by selecting some set of pigments from a region of pixels, and adding them as a new layer on top elsewhere in the image. While the pasted regions are not identical to the copied regions (as they would be with standard RGB copy paste), they appear as if they were painted as part of the image.

(a) Palette summarization applied to photos, as compared to Tan et al. [10], Color, and Chang et al. [14].

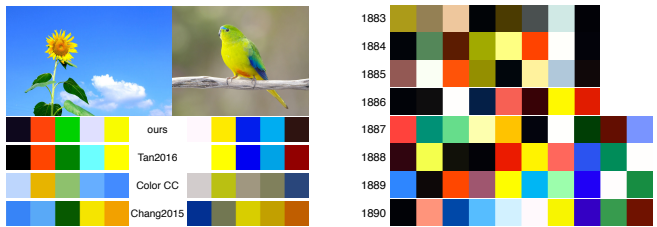(b) Summarizations of Van Gogh's paintings arranged by year to show evolution of style.

Fig. 18: Examples of palette summarization.

more success with paint-like color mixtures such as green and cyan.

We can also use our palette extraction method to analyze collections of images, by amending our method to jointly reconstruct the pixels of multiple images. We use this approach to extract aggregate palettes from paintings of Van Gogh organized by year (Fig. 18b). Two results are clear from this analysis. First, the range of colors Van Gogh painted with expanded over the 1880's, as we expanded from eight pigments to ten pigments to achieve good reconstruction errors. Second, the vibrancy increased dramatically as well.

6.6 Edge Detection and Enhancement

Our weight maps can improve edge-based image analysis (Fig. 19). We apply an existing edge detection method [41]

Fig. 19: Paint-aware edge detection and enhancement.

to each weight map separately and merge the per-pigment response as the per-pixel max. Paint edge images can be used to adapt standard image processing routines to be paint-aware. For example, we do edge enhancement by thickening pigments near boundaries according to the edge response, which can visually emphasize painted objects in a different way than RGB edge enhancement.

7 CONCLUSION

We demonstrate a method that can recover plausible physical pigments from only an RGB image of a painting, and then recover the mixing proportion of those pigments at each pixel. We are able to accurately reconstruct the RGB values of the image, and even closely match multispectral reflectance per-pixel as well, though the underlying pigment coefficients may differ. We use this decomposition to enable a number of image editing operations that occur in "pigment space," which creates results in a style more consistent with natural media imagery rather than digital RGB edits.

Limitations First, our approach requires users to choose a target number of primary pigments. While this is the only user interaction in our entire pipeline, it is still a decision that the user must make. As shown in Fig. 9, a large number of primary pigments will have lower RGB image reconstruction error at the cost of more tedious edits and additional processing time. Second, if a ground truth primary pigment can be mixed from other primary pigments, our technique will not find it. However, this is an ambiguous situation, since it is not needed for perfect reconstruction. Still, it may be important for applications like pigment identification. Third, we use the Okumura [16] dataset as a prior to help find initial a, s values for our optimization, though our solutions are not limited to them. We do not have other pigment datasets to verify whether this prior causes us to overfit our recovered pigment parameters. For example, this prior information may be more helpful for finding acrylic or oil pigments than watercolor. Fourth, we assume constant pigment thickness. This is a simplifying assumption that speeds our optimization since the solution space is already under-constrained. However, if we allowed for varying thickness, darker and lighter tones of a pigment could be obtained without a black or white pigment. We could reduce our palette size by computing the 2D convex hull of the chromaticity of the pixels, ignoring brightness (Section 4.1). This is similar to the approach used by Aharoni-Mack et al. [15]. We would also be better able to handle media like watercolor with varying translucency. Finally, our approach does not recover ground truth, just plausible results enabling paint-like editing.

Future work In the future, we would like to extend our result to estimate pigment layers instead of just mixtures.

We plan to use our decomposition to help extract brush stroke-level structure from images of paintings, to enable manipulation of the brush strokes in painting images. We predict that interpreting complex image structures with more appropriate models will have applications in many applications of computer graphics.

8 ACKNOWLEDGMENT

REFERENCES

[1] P. Kubelka and F. Munk, "An article on optics of paint layers," *Zeitschrift für Technische Physik*, vol. 12, no. 593–601, 1931.

[2] P. Kubelka, "New contributions to the optics of intensely light-scattering materials. Part I," *Journal of the Optical Society of America*, vol. 38, no. 5, pp. 448–448, 1948.

[3] D. R. Duncan, "The colour of pigment mixtures," *Proceedings of the Physical Society*, vol. 52, no. 3, p. 390, 1940.

[4] C. J. Curtis, S. E. Anderson, J. E. Seims, K. W. Fleischer, and D. H. Salesin, "Computer-generated watercolor," in *SIGGRAPH*, 1997, pp. 421–430.

[5] W. V. Baxter, J. Wendt, and M. C. Lin, "IMPaSTo: A realistic, interactive model for paint," in *NPAR*, 2004, pp. 45–56.

[6] J. Lu, S. DiVerdi, W. A. Chen, C. Barnes, and A. Finkelstein, "RealPigment: Paint compositing by example," in *NPAR*, 2014, pp. 21–30.

[7] J. Tan, M. Dvorožňák, D. Sýkora, and Y. Gingold, "Decomposing time-lapse paintings into layers," *ACM Trans. Graph.*, vol. 34, no. 4, pp. 61:1–61:10, Jul. 2015.

[8] G. L. Bernstein and W. Li, "Lillicon: Using transient widgets to create scale variations of icons," *ACM Trans. Graph.*, vol. 34, no. 4, pp. 144:1–144:11, Jul. 2015. [Online]. Available: http://doi.acm.org/10.1145/2766980

[9] K. Kwok and G. Webster, "Project Naptha: highlight, copy, and translate text from any image," https://projectnaptha.com/, 2016, accessed: 2017-01-15.

[10] J. Tan, J.-M. Lien, and Y. Gingold, "Decomposing images into layers via RGB-space geometry," *ACM Trans. Graph.*, vol. 36, no. 1, pp. 7:1–7:14, Nov. 2016. [Online]. Available: http://doi.acm.org/10.1145/2988229

[11] S. Lin, M. Fisher, A. Dai, and P. Hanrahan, "Layerbuilder: Layer decomposition for interactive image and video color editing," *arXiv preprint arXiv:1701.03754*, 2017.

[12] Y. Aksoy, T. O. Aydin, A. Smolić, and M. Pollefeys, "Unmixing-based soft color segmentation for image manipulation," *ACM Transactions on Graphics (TOG)*, vol. 36, no. 2, p. 19, 2017.

[13] Q. Zhang, C. Xiao, H. Sun, and F. Tang, "Palette-based image recoloring using color decomposition optimization," *IEEE Transactions on Image Processing*, vol. 26, no. 4, pp. 1952–1964, 2017.

[14] H. Chang, O. Fried, Y. Liu, S. DiVerdi, and A. Finkelstein, "Palette-based photo recoloring," *ACM Trans. Graph.*, vol. 34, no. 4, Jul. 2015.

[15] E. Aharoni-Mack, Y. Shambik, and D. Lischinski, "Pigment-based recoloring of watercolor paintings," in *NPAR*, July 2017.

[16] Y. Okumura, "Developing a spectral and colorimetric database of artist paint materials," Master's thesis, Rochester Institute of Technology, 2005.

[17] I. Kauvar, S. J. Yang, L. Shi, I. McDowall, and G. Wetzstein, "Adaptive color display via perceptually-driven factored spectral projection," *ACM Trans. Graph.*, vol. 34, no. 6, pp. 165:1–165:10, Oct. 2015. [Online]. Available: http://doi.acm.org/10.1145/2816795.2818070

[18] S. Xu, H. Tan, X. Jiao, F. C. Lau, and Y. Pan, "A generic pigment model for digital painting," *Computer Graphics Forum*, 2007.

[19] B. Boldrini, W. Kessler, K. Rebner, and R. Kessler, "Hyperspectral imaging: a review of best practice, performance and pitfalls for inline and online applications," *Journal of Near Infrared Spectroscopy*, vol. 20, no. 5, pp. 438–508, 2012.

[20] R. Berns, L. Taplin, and M. Nezamabadi, "Spectral imaging using a commercial colour-filter array digital camera," in *ICOM-CC*, 2005, pp. 743–750.

[21] M. Parmar, S. Lansel, and J. Farrell, "An LED-based lighting system for acquiring multispectral scenes," in *Digital Photography VIII*, 2012. [Online]. Available: http://dx.doi.org/10.1117/12.912513

[22] J.-I. Park, M.-H. Lee, M. D. Grossberg, and S. K. Nayar, "Multispectral imaging using multiplexed illumination," in *ICCV*, 2007, pp. 1–8.

[23] A. Ibrahim, T. Horiuchi, S. Tominaga, and A. Ella Hassanien, "Spectral reflectance images and applications," in *Image Feature Detectors and Descriptors: Foundations and Applications*, A. I. Awad and M. Hassaballah, Eds., 2016, pp. 227–254.

[24] R. S. Berns, L. A. Taplin, F. H. Imai, E. A. Day, and D. C. Day, "A comparison of small-aperture and image-based spectrophotometry of paintings," *Studies in Conservation*, vol. 50, no. 4, pp. 253–266, 2005. [Online]. Available: http://dx.doi.org/10.1179/sic.2005.50.4.253

[25] H. Liang, K. Keita, B. Peric, and T. Vajzovic, "Pigment identification with optical coherence tomography and multispectral imaging," 2008.

[26] R. S. Berns and F. H. Imai, "The use of multi-channel visible spectrum imaging for pigment identification," in *ICOM-CC*, September 2002, pp. 217–222.

[27] Y. Zhao, R. S. Berns, Y. Okumura, and L. A. Taplin, "Improvement of spectral imaging by pigment mapping," in *Color and Imaging Conference*, 2005, pp. 40–45.

[28] A. Pelagotti, A. D. Mastio, A. D. Rosa, and A. Piva, "Multispectral imaging of paintings," *IEEE Signal Processing Magazine*, vol. 25, no. 4, pp. 27–36, July 2008.

[29] A. Cosentino, "Identification of pigments by multispectral imaging; a flowchart method," *Heritage Science*, vol. 2, no. 1, p. 8, 2014. [Online]. Available: http://dx.doi.org/10.1186/2050-7445-2-8

[30] Y. Zhao, R. S. Berns, L. A. Taplin, and J. Coddington, "An investigation of multispectral imaging for the mapping of pigments in paintings," in *Electronic Imaging*, 2008.

[31] J. K. Delaney, P. Ricciardi, L. D. Glinsman, M. Facini, M. Thoury, M. Palmer, and E. R. d. l. Rie, "Use of imaging spectroscopy, fiber optic reflectance spectroscopy, and x-ray fluorescence to map and identify pigments in illuminated manuscripts," *Studies in Conservation*, vol. 59, no. 2, pp. 91–101, 2014.

[32] F. M. Abed, *Pigment identification of paintings based on Kubelka-Munk theory and spectral images*. Rochester Institute of Technology, 2014.

[33] N. Ohta and A. R. Robertson, *CIE Standard Colorimetric System*. John Wiley & Sons, Ltd, 2006, pp. 63–114.

[34] C. B. Barber, D. P. Dobkin, and H. Huhdanpaa, "The quickhull algorithm for convex hulls," *ACM Trans. Math. Softw.*, vol. 22, no. 4, pp. 469–483, Dec. 1996.

[35] J. T. Barron and B. Poole, "The fast bilateral solver," *ECCV*, pp. 617–632, 2016.

[36] C. Zhu, R. H. Byrd, P. Lu, and J. Nocedal, "Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization," *ACM Trans. Math. Softw.*, vol. 23, no. 4, pp. 550–560, Dec. 1997.

[37] C. Rother, V. Kolmogorov, and A. Blake, ""GrabCut" — interactive foreground extraction using iterated graph cuts," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 309–314, 2004.

[38] A. Telea, "An image inpainting technique based on the fast marching method," *JGT*, vol. 9, no. 1, pp. 23–34, 2004.

[39] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman, "Patch-Match: A randomized correspondence algorithm for structural image editing," *ACM Trans. Graph.*, vol. 28, no. 3, pp. 24:1–24:11, Aug. 2009.

[40] Adobe, "Adobe Color CC," https://color.adobe.com, 2016.

[41] P. Isola, D. Zoran, D. Krishnan, and E. H. Adelson, "Crisp boundary detection using pointwise mutual information," in *ECCV*, 2014, pp. 799–814.
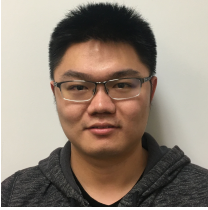
Jianchao Tan is currently a fifth year Ph.D. student in the Department of Computer Science at George Mason University, advised by Dr. Yotam Gingold. He obtained his B.S. in 2013 from the Electronic Engineering and Information Science Department at the University of Science and Technology of China.

Stephen DiVerdi received his B.S. from Harvey Mudd College in 2002 and his Ph.D. from the University of California, Santa Barbara in 2007. Since then, Stephen has worked at Adobe and Google on enabling novel interfaces for digital artwork and digital experiences, including natural media painting, automatic image enhancement, audio synthesis, and virtual and augmented reality.

Jingwan Lu joined Adobe Research in 2014. Her current research interests include deep-learning-based image editing and generation, sketch-based search, creative applications for AR and VR, computational photography, data-driven artistic content creation, and other vision, graphics and machine learning topics.

Yotam Gingold is an assistant professor in the Department of Computer Science at George Mason University, where he directs the Creativity and Graphics Lab (CraGL). He received his Ph.D. from New York University in 2009. His research interests include interactive modeling, geometry processing, color, and creativity.