Automatic Palette Extraction for Image Editing

Mairéad Grogan, Matis Hudon, Daniel McCormack, Aljosa Smolic

School of Computer Science and Statistics, Trinity College Dublin

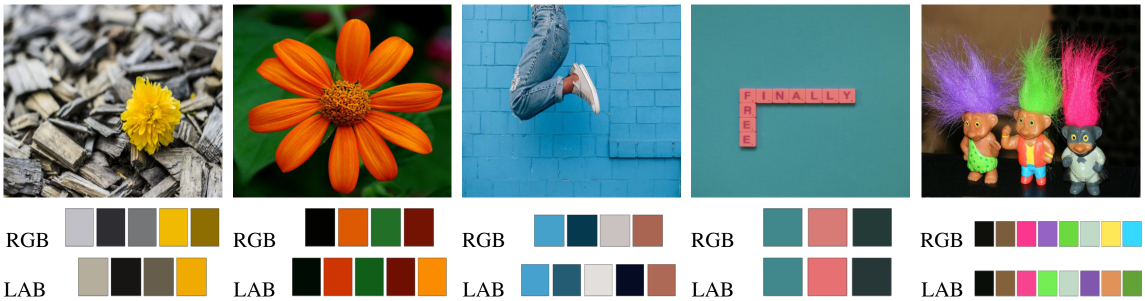

Figure 1: Palettes generated by our approach in the RGB and LAB spaces.

Abstract

Interactive palette based colour editing applications have grown in popularity in recent years, but while many methods propose fast palette extraction techniques, they typically rely on the user to define the number of colours needed. In this paper, we present an approach that extracts a small set of representative colours from an image automatically, determining the optimal palette size without user interaction. Our iterative technique assigns a vote to each pixel in the image based on how close they are in colour space to the colours already in the palette. We use a histogram to divide the colours into bins and determine which colour occurs most frequently in the image but is far away from all of the palette colours, and we add this colour to the palette. This process continues until all pixels in the image are well represented by the palette. Comparisons with existing methods show that our colour palettes compare well to other state of the art techniques, while also computing the optimal number of colours automatically at interactive speeds. In addition, we showcase how our colour palette performs when used in image editing applications such as colour transfer and layer decomposition.

Keywords: Palette Extraction; Image Editing; Colour Editing; Colour Theme;

1 Introduction

The manipulation of colours in images is one of the most important problems in many computer vision applications, however using commercial colour manipulation toolboxes is often arduous for non-experts. As with many image processing problems, there is also a natural trade-off between a tool's ease of use and the actual result quality and expressiveness one can obtain. A pioneering colour editing algorithm proposed by Reinhard et al [Reinhard et al., 2001], created a way to transfer the colour feel of a palette image to a target image, removing the need to apply a series of edits to get the desired result. Since then, many other colour transfer approaches have been proposed based on histogram matching, Gaussian mixture modelling and optimal transport [Pitié et al., 2005, Tai et al., 2005, Grogan and Dahyot, 2017, Papadakis et al., 2011, Ferradans et al., 2013, Bonneel et al., 2015].

When an example image with the desired colour distribution is not readily available, interactive colour editing approaches give the user the ability to determine what the final recoloured image will look like. Edit propagation techniques have been popular for many years, and allow the user to create scribbles on different regions of the image, indicating how they should be recoloured [An and Pellacini, 2008, Chen et al., 2014]. Recently, palette based recolouring methods have also become very popular, allowing the user to recolour an image by changing a small palette of colours that represent its colour distribution [Chang et al., 2015, Zhang et al., 2017, Grogan et al., 2017]. Some examples of such palettes can be seen in Figure 1. Chang et al. [Chang et al., 2015] proposed one of the first palette based recolouring techniques, and provide an easy to use interface that allows the user to recolour images in real time. Zhang et al. [Zhang et al., 2017] also propose a palette based recolouring method that decomposes the colours of each pixel in the image into a linear combination of the palette colours. They show that their technique is free from the artifacts seen in [Chang et al., 2015]. Askoy et al. [Aksoy et al., 2017] propose a similar method for image editing, but extend it further by splitting the image into a number of layers, with each layer containing colours similar to one of the palette entries. They show that these layers can be used for image compositing, image recolouring and green screen keying.

As many recent recolouring and image editing approaches are based on colour palette representation, there is no denying that in these cases the colour palette generation is key to a successful result. While many methods propose fast palette extraction techniques, they often rely on the user to define the actual number of colours for populating the palette. There is a lack of fully automatic methods for high quality, efficient palette generation from images. In this paper we propose a fast and fully automatic approach for extracting representative colour palettes from images. Our method automatically finds the right number of representative colours of an image and generates palettes suitable for colour transfer and alteration methods. The remainder of this paper is organised as follows: In Section 2 we discuss previous palette extraction methods and present our proposed technique in Section 3. We show that our technique compares well to other state of the art methods in Section 4 and finally present our conclusions and suggestions for future work in Section 5.

2 State of the Art

A palette should typically consist of the most representative colours in the image, and the correlation between palette colours should be low. The palette should also contain a moderate number of colours - too few will limit the users ability to properly edit the image, and too many can make editing cumbersome. Early works in palette extraction include O'Donovan et al. [O'Donovan et al., 2011], who use a dataset of user rated palettes to develop a technique for palette generation. However, their method tends to perform poorly on natural photographs, as shown in [Chang et al., 2015], and can take over a minute to compute. Later, Lin et al. [Lin and Hanrahan, 2013] proposed a technique that worked well on natural images, but again it is slow to compute and like O'Donovan et al., only generates palettes with 5 colours. Shapira et al. [Shapira et al., 2009] propose to generate a model based on a Gaussian mixture model, but again computation is slow, taking around 10 seconds to compute.

Both Chang et al. [Chang et al., 2015] and Zhang et al. [Zhang et al., 2017] introduce a k-means variant for palette extraction. Both techniques first reduce the colours to be processed by dividing the colours in the image into histogram bins and compute the k-means algorithm on the mean colours of each bin only. They also propose to initialize the k cluster centres of the k-means algorithm so that the palette results are deterministic - they select the k mean colours that represent the most pixels in the image and add a constraint to ensure that the k cluster centres are far apart. The k-means algorithm is then computed to find k palette colours. Zhang et al. also add a step which ensures that two significant, but similar, colours in the image do not get merged into one palette entry, while Chang et al. propose to discard very dark palette colours as they can be redundant when recolouring. However, both of these techniques require a predefined palette size k.

Aksoy et al. [Aksoy et al., 2017], on the other hand, automatically determine the number of palette entries that the input image should have. Their method iteratively selects pixel colours and adds them to the palette until all of the colours in the image are well represented. Again, they use a 3D histogram to divide up the pixel colours of the image. Each pixel is assigned a vote, with a higher vote given to pixels that are not well

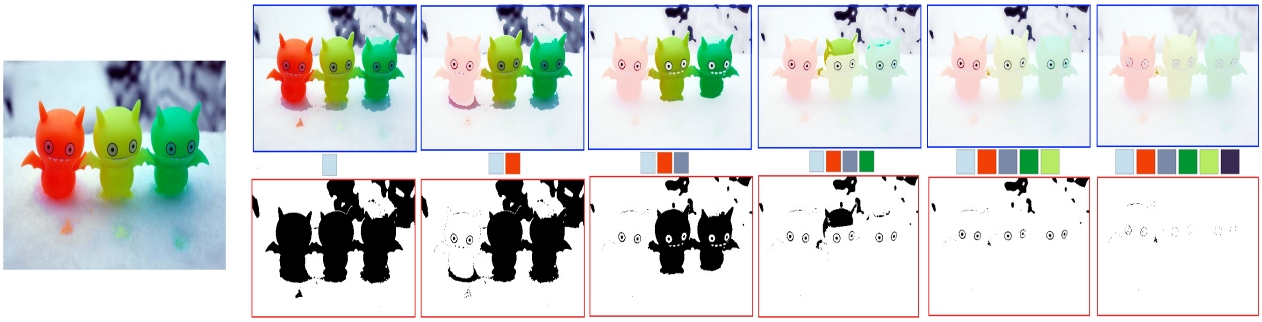

Figure 2: Our iterative method continues to add colours to the palette until all pixels are well represented. Bottom row: Binary masks indicate the votes of each pixel based on the current palette. Top row: The masks are overlaid on the original image to highlight what colours are still unrepresented by the palette.

represented by the current palette, and the histogram bin with the highest vote is used to determine the next colour to be added to the palette. However, the palettes generated can have quite a large number of colours and the computation can take several seconds. In this paper, we combine the advantages of several palette generation methods and propose a technique which automatically determines a palette of colours from an image at interactive speeds, allowing users to use it immediately to edit the image.

3 Palette Extraction

Similar to Aksoy et al. [Aksoy et al., 2017], we use a greedy iterative scheme which continues to add colours to the palette until all pixels in the image are well represented (see Fig 2). Our first step is to divide the colours in the image into $10 \times 10 \times 10$ bins. Each pixel p in the image is then assigned a vote v^p, based on how far the colour p is from the colours $\{c_i\}_{i=1,..n}$ in the palette. The vote v^p is computed as follows:

$$v^p = \begin{cases} 0, & \text{if } d(p, c_i) < \tau_1 \text{ for any } c_i. \\ 1, & \text{otherwise.} \end{cases} \tag{1}$$

where $d(x, y)$ is the Euclidean distance between x and y and τ_1 is a fixed threshold. In this way, only pixels that are far enough away from the colours in the palette will be given a vote of 1. Figure 2 presents masks indicating the votes of each pixel for a given palette. Pixels with a vote of 1 are coloured black in the mask, while those with a vote of 0 are coloured white. The votes of the pixels in each bin are summed, and the bin b with the highest vote is computed. The next palette colour c_i is selected from bin b as follows:

$$c_i = \arg\max_{p \in b} S^p \tag{2}$$

where S^p is the number of pixels in the 20×20 neighbourhood of p that also lie in the bin b. We continue to add colours c_i to the palette in this way until the number of pixels in the bin with the highest vote is less than a threshold τ_2.

This differs from the technique proposed by Aksoy et al. [Aksoy et al., 2017] in that they compute a vote v_p for each pixel p based on the pixel's representation score r^p. This representation score is more complex to compute and is used to determine whether a given pixel's colour can be created by combining the colours already in the palette. Colours that can be created in this way will not be added to the palette. While this is important when splitting an image into additive layers, it is not necessarily useful when generating a palette for image recolouring. We therefore simplify their voting system by taking into account only how far a given pixel is from the other palette colours. This reduces the computation time significantly while still ensuring that a high quality palette is generated. The parameter τ_1 is used to ensure that palette colours stay reasonably far apart in the colour space and τ_2 ensures that colours are only added to the palette when they represent a large enough portion of the image.

4 Experimental Results

4.1 LAB vs RGB

We tested our palette extraction technique in both the LAB and RGB spaces, setting $\tau_1 = 0.25$ in LAB and $\tau_1 = 0.45$ in RGB. In both cases $\tau_2 = m \times n/1000$, where m and n are the width and height of the image respectively. In Figure 1 we compare the palettes generated in both colour spaces. In general, we found that both colour spaces created similar palettes, or palettes that differed in size by one colour (Fig 1, col 1-4) . In some cases we found that one of the colour spaces generated a more descriptive palette than the other, as can be seen in Figure 1, column 3. Here, the RGB palette includes blue and yellow colours, which are absent from the LAB palette. This is because blue and yellow lie too close to the grey and green palette colours in LAB space, and so are discarded when these palette colours are added. This is one of the drawbacks of the binary voting system proposed in Equation 1 and future work could concentrate on changing the value of τ_1 dynamically to ensure that significant colours are not discarded. As there is very little difference between the colour spaces, in the remainder of this paper we chose to present the results of our method computed in the LAB space.

4.2 Comparison with other methods

In Figures 3 and 4 we compare our palette generation to the k-means algorithm (computed in LAB space) and the palette extraction techniques proposed by Chang et al., Aksoy et al. and Zhang et al. [Chang et al., 2015, Aksoy et al., 2017, Zhang et al., 2017]. As our method estimates the number of colours automatically, for comparison when using k-means and the algorithm proposed by Chang et al., we estimate palettes with the same number of colours as ours. Note that Chang et al. also set the maximum number of palette colours allowed to 7 (see row 2, column 4 of Figs. 3 and 4). Both Zhang et al. and Aksoy's et al's palette results are sourced from their papers [Zhang et al., 2017, Aksoy et al., 2017].

From both figures we can see that our colour palettes are very similar to Chang et al's technique, although their algorithm cannot compute the number of palette colours automatically. In many cases the k-means algorithm performs similarly to the other palette extraction techniques, however it also requires a predefined palette size k and the results are non-deterministic. We can also see that in some cases, important colours are missing from the k-means palettes. For example, in row 1, column 4 of Figure 4, a brown palette colour is missing. Similarly, in row 1, column 4 of Figure 3, a bright red palette colour is missing. In Figure 3 we again see that our palette extraction is very similar to Zhang et al's, and both their technique and ours capture brighter colours that are sometimes missing from Chang et al's palette. Again, however, the user must indicate how many colours should appear in the palette. In comparison, Aksoy et al's method correctly identifies the main colours of the image (see Fig 4). However, in some cases too many palette colours are detected when fewer would be sufficient (see Fig 4, row 2, col. 1). While a large number of palette colours may be suitable for image decomposition, many palette entries can become cumbersome when used for colour editing.

In terms of computation time, Aksoy et al's algorithm is the slowest, taking an average of 9 seconds. In comparison, k-means, our algorithm, and those of Chang et al. and Zhang et al., are much faster, taking an average time of 0.2, 0.4, 0.06 and 0.06 seconds respectively, ensuring that the palette can be presented to the user in real time when used for editing.

4.3 Size of the palette

The size of the estimated palette is essential to most use-cases. As stated by Chang et al., a moderate size of palette is paramount to a good recolouring - too small a size might cause an inaccurate representation of some colours, while a palette that is too large could promote correlations between different palette colours and make it difficult to achieve the desired colour editing result. In Zhang et al's method, users are allowed to choose the palette size according to their needs, therefore the results presented in their paper show different palette sizes (chosen manually by users) for multiple input images. We tried our fully automatic palette generation method on their data and found the number of palette colours that our algorithm automatically extracts from

Figure 3: Comparison between our technique, Chang et al's technique [Chang et al., 2015], the k-means algorithm applied in the LAB space (KM), and Zhang et al's palette extraction technique [Zhang et al., 2017].

their images is often very close to what a human user would naturally select. Also, as can be seen in Figures 3 and 4, the colours selected by both our algorithm and theirs are very similar. On average over the 15 images presented in [Zhang et al., 2017], the difference between the number of manually selected palette colours versus our method is less than one.

4.4 Applications

In this section we also explore the use of our extracted palettes for both recolouring and layer decomposition. In Fig 5, row 1, we present the results of a palette based image recolouring technique [Grogan et al., 2017] which takes as input an image, a palette of colours generated from the image, and an edited palette indicating which colours in the image should change after recolouring. For example, in the case of the yellow flower (Fig 5, right), the yellow palette colour was changed to red, indicating that the yellow flower should become red. In both cases the recolouring results are successful using our palette as input.

As previously mentioned by Chang et al. [Chang et al., 2015], including purely black or white palette entries is often not desirable when they are being used as part of an image recolouring application, as making changes to very dark or bright colours in an image can create unwanted artifacts in the recoloured result. In order to compare with Aksoy et al. we have included all palette entries in this result section, but black and white palette colours could easily be removed before presenting them in a recolouring application.

In Figure 5, rows 2 and 3, we present the layers that are generated when our palette of colours is used as

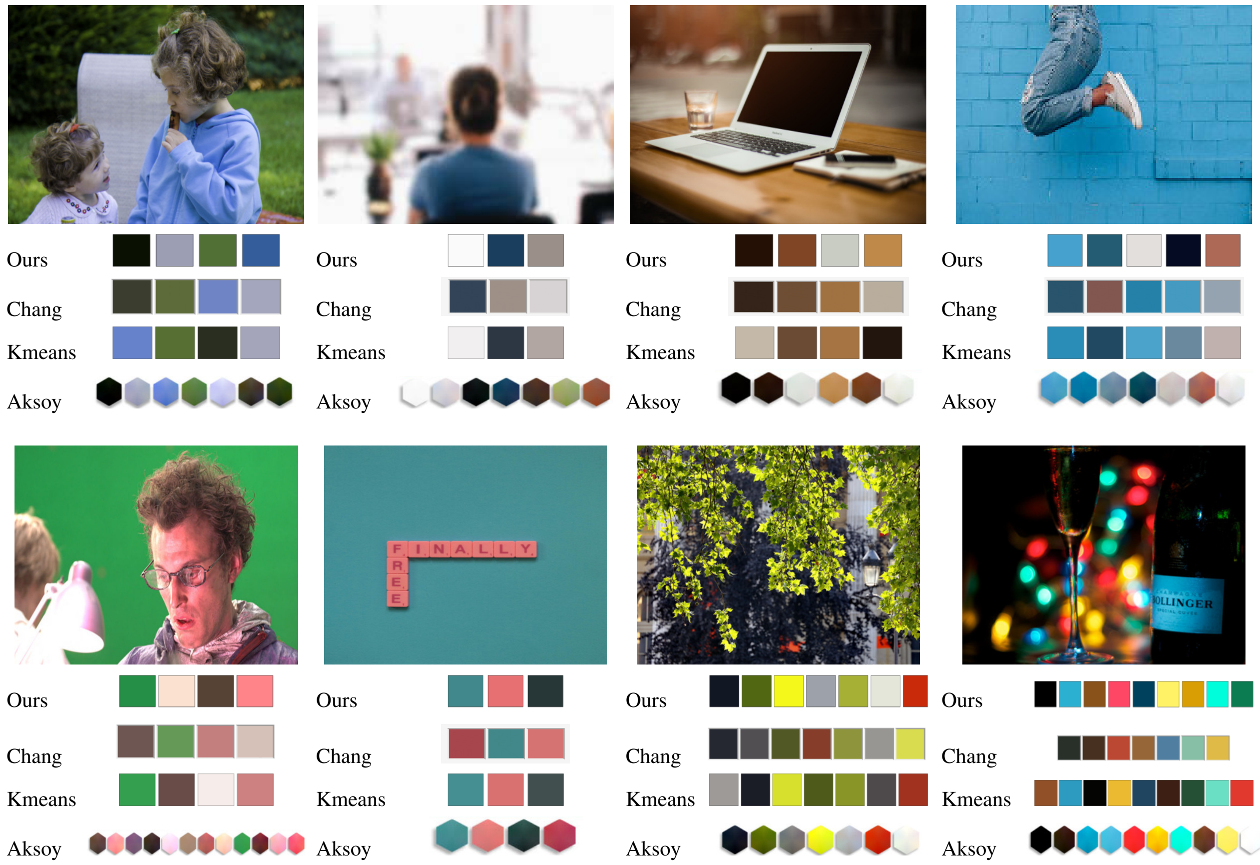

Figure 4: Comparison between our technique, Chang et al's technique [Chang et al., 2015], the k-means algorithm applied in the LAB space, and Aksoy et al's palette extraction technique [Aksoy et al., 2017].

input in Aksoy et al's colour unmixing method. This technique takes an image and palette as input and for each palette colour generates a layer containing the colours in the image that are similar to it. Combining all of the layers recreates the input image. Although the original method proposed by Aksoy creates palettes with a large number of colours to ensure that the colours in each image layer are very similar, in this figure we can see that even using our palette of colours, which is smaller than that of Aksoy et al., the layers created are reasonably coherent in terms of colour, and can be easily used in other image editing tasks such as image compositing and colour editing.

5 Conclusion and Future Work

We have presented an approach for fast, automatic palette extraction from images using an iterative approach which continues to add colours to the palette until the colours in the image are well represented. We have shown that palettes generated using our approach are similar to other palette extraction techniques, and the optimal number of colours computed is highly correlated with the number of palette colours selected by users in [Zhang et al., 2017]. While our technique is similar to that of Aksoy et al., we have simplified the computational cost to ensure that the palette can be extracted quickly, and reduced the number of palette colours that they estimate to ensure that the palette is not too cumbersome when used in applications such as image recolouring. Future work will investigate ways to extend the voting system to ensure that important colours that are within a distance τ_2 of each other, but are still reasonably different, do not get merged into one palette colour.

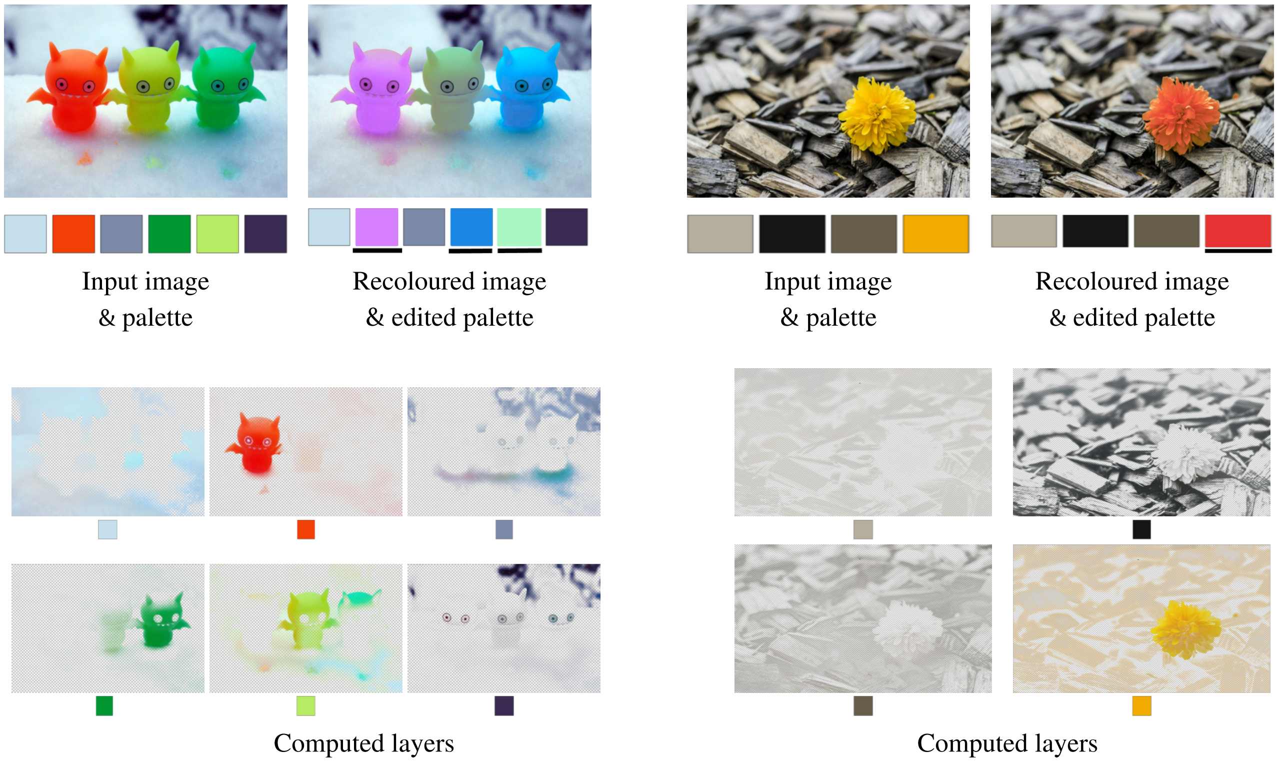

Figure 5: Top: Recolouring results using a palette based image recolouring technique [Grogan et al., 2017]. Bottom: The layers generated when our palette of colours is used as input in Aksoy et al's colour unmixing algorithm [Aksoy et al., 2017].

Acknowledgments

This publication has emanated from research conducted with the financial support of Science Foundation Ireland (SFI) under the Grant Number 15/RP/2776.

References

[Aksoy et al., 2017] Aksoy, Y., Aydın, T. O., Smolić, A., and Pollefeys, M. (2017). Unmixing-based soft color segmentation for image manipulation. *ACM Trans. Graph.*, 36(2):19:1–19:19.

[An and Pellacini, 2008] An, X. and Pellacini, F. (2008). Appprop: All-pairs appearance-space edit propagation. In *ACM SIGGRAPH 2008 Papers*, SIGGRAPH '08, pages 40:1–40:9, New York, NY, USA. ACM.

[Bonneel et al., 2015] Bonneel, N., Rabin, J., Peyre, G., and Pfister, H. (2015). Sliced and radon wasserstein barycenters of measures. *Journal of Mathematical Imaging and Vision*, 51(1):22–45.

[Chang et al., 2015] Chang, H., Fried, O., Liu, Y., DiVerdi, S., and Finkelstein, A. (2015). Palette-based photo recoloring. *ACM Transactions on Graphics (SIGGRAPH)*, 34(4).

[Chen et al., 2014] Chen, X., Zou, D., Li, J., Cao, X., Zhao, Q., and Zhang, H. (2014). Sparse dictionary learning for edit propagation of high-resolution images. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2854–2861.

[Ferradans et al., 2013] Ferradans, S., Papadakis, N., Rabin, J., Peyre, G., and Aujol, J.-F. (2013). Regularized discrete optimal transport. In Kuijper, A., Bredies, K., Pock, T., and Bischof, H., editors, *Scale Space* *and Variational Methods in Computer Vision*, volume 7893 of *Lecture Notes in Computer Science*, pages 428–439. Springer Berlin Heidelberg.

[Grogan and Dahyot, 2017] Grogan, M. and Dahyot, R. (2017). Robust Registration of Gaussian Mixtures for Colour Transfer. *ArXiv e-prints*.

[Grogan et al., 2017] Grogan, M., Dahyot, R., and Smolic, A. (2017). User interaction for image recolouring using L2. In *Proceedings of the 14th European Conference on Visual Media Production (CVMP 2017)*, CVMP 2017, pages 6:1–6:10, New York, NY, USA. ACM.

[Lin and Hanrahan, 2013] Lin, S. and Hanrahan, P. (2013). Modeling how people extract color themes from images. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '13, pages 3101–3110, New York, NY, USA. ACM.

[O'Donovan et al., 2011] O'Donovan, P., Agarwala, A., and Hertzmann, A. (2011). Color compatibility from large datasets. *ACM Trans. Graph.*, 30(4):63:1–63:12.

[Papadakis et al., 2011] Papadakis, N., Provenzi, E., and Caselles, V. (2011). A variational model for histogram transfer of color images. *Image Processing, IEEE Transactions on*, 20(6):1682–1695.

[Pitié et al., 2005] Pitié, F., Kokaram, A. C., and Dahyot, R. (2005). N-dimensional probability density function transfer and its application to color transfer. In *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, volume 2, pages 1434–1439 Vol. 2.

[Reinhard et al., 2001] Reinhard, E., Adhikhmin, M., Gooch, B., and Shirley, P. (2001). Color transfer between images. *Computer Graphics and Applications, IEEE*, 21(5):34–41.

[Shapira et al., 2009] Shapira, L., Shamir, A., and Cohen-Or, D. (2009). Image appearance exploration by model-based navigation. *Computer Graphics Forum*, 28(2):629–638.

[Tai et al., 2005] Tai, Y.-W., Jia, J., and Tang, C.-K. (2005). Local color transfer via probabilistic segmentation by expectation-maximization. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 747–754 vol. 1.

[Zhang et al., 2017] Zhang, Q., Xiao, C., Sun, H., and Tang, F. (2017). Palette-based image recoloring using color decomposition optimization. *IEEE Transactions on Image Processing*, 26(4):1952–1964.