Electronic Theses and Dissertations, 2004-2019

2006

Image-based Material Editing

Erum Khan

University of Central Florida

Part of the [Computer Sciences Commons](#), and the [Engineering Commons](#)

Find similar works at: <https://stars.library.ucf.edu/etd>

University of Central Florida Libraries <http://library.ucf.edu>

This Doctoral Dissertation (Open Access) is brought to you for free and open access by STARS. It has been accepted for inclusion in Electronic Theses and Dissertations, 2004-2019 by an authorized administrator of STARS. For more information, please contact STARS@ucf.edu.

STARS Citation

Khan, Erum, "Image-based Material Editing" (2006). *Electronic Theses and Dissertations, 2004-2019*. 1091.

<https://stars.library.ucf.edu/etd/1091>

IMAGE-BASED MATERIAL EDITING

by

ERUM ARIF KHAN
M.S. University of Central Florida, USA, 2005

A dissertation submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy
in the School of Electrical Engineering and Computer Science
in the College of Engineering and Computer Science
at the University of Central Florida
Orlando, Florida

Fall Term
2006

Major Professor:
Erik Reinhard

© 2006 Erum Arif Khan

ABSTRACT

Photo editing software allows digital images to be blurred, warped or re-colored at the touch of a button. However, it is not currently possible to change the material appearance of an object except by painstakingly painting over the appropriate pixels. Here we present a set of methods for automatically replacing one material with another, completely different material, starting with only a single high dynamic range image, and an alpha matte specifying the object. Our approach exploits the fact that human vision is surprisingly tolerant of certain (sometimes enormous) physical inaccuracies. Thus, it may be possible to produce a visually compelling illusion of material transformations, without fully reconstructing the lighting or geometry. We employ a range of algorithms depending on the target material. First, an approximate depth map is derived from the image intensities using bilateral filters. The resulting surface normals are then used to map data onto the surface of the object to specify its material appearance. To create transparent or translucent materials, the mapped data are derived from the object's background. To create textured materials, the mapped data are a texture map. The surface normals can also be used to apply arbitrary bidirectional reflectance distribution functions to the surface, allowing us to simulate a wide range of materials. To facilitate the process of material editing, we generate the HDR image with a novel algorithm, that is robust against noise in individual exposures. This ensures that any noise, which would possibly have affected the shape recovery of the objects adversely, will be removed. We also present an algorithm to automatically generate alpha mattes. This algorithm requires as input two images—one where the object is in focus, and one where the background is in focus—and then automatically produces an approximate matte, indicating which pixels belong to the object. The result is then improved by a second algorithm to generate an accurate alpha matte, which can be given as input to our material editing techniques.

TABLE OF CONTENTS

LIST OF TABLES	vii
LIST OF FIGURES	viii
CHAPTER 1 INTRODUCTION	1
1.1 Problem Definition	3
1.2 System Overview	5
1.3 Uses	7
1.4 Previous Work	9
1.5 Human Visual Perception	11
CHAPTER 2 HIGH DYNAMIC RANGE IMAGE GENERATION	15
2.1 Previous Work	17
2.2 Iterative Noise and Ghost Removal	19
2.3 Results	24
2.4 Conclusions	25
CHAPTER 3 SELECTING A COLOR SPACE	32
3.1 Color Spaces	34
3.2 Behavior of Color across Edges	37
3.3 Quantitative Evaluation	39
3.3.1 Data collection	39
3.3.2 Analysis	41
3.4 Discussion	47

CHAPTER 4	ALPHA MATTE GENERATION	50
4.1	Previous Work	50
4.2	Algorithm	53
4.2.1	Non-linear response compression	53
4.2.2	Center-surround analysis	55
4.2.3	Filling-in	58
4.2.4	Clean-up	59
4.3	Results	60
4.4	Conclusions	61
CHAPTER 5	ALPHA MATTE GENERATION REVISITED	68
5.1	Previous Work	70
5.2	Algorithm	72
5.2.1	Composite equation	73
5.2.2	Regularization	75
5.2.3	Constrained estimation	75
5.2.4	Minimizing the objective function	76
5.3	Results	79
CHAPTER 6	MATERIAL EDITING	84
6.1	Problem Definition	85
6.2	Overview	87
6.3	Object Shape (Ω')	88
6.3.1	Depth recovery	88

6.3.2	Gradients and surface normals	92
6.4	Incident Light (L_i)	93
6.4.1	Light from behind the object	94
6.4.2	Reconstructing a 3D environment	96
6.5	Material Transforms	97
6.5.1	Glossy surfaces	98
6.5.2	BRDF replacement	100
6.5.3	Retexturing	101
6.5.4	Transparency and translucency	103
6.6	Conclusions	107
CHAPTER 7 CONCLUSIONS AND FUTURE WORK		111
LIST OF REFERENCES		116

LIST OF TABLES

3.1	<i>Channels, their ROC curve areas and Standard Error for Sunlit Scenes (Area1 and $\sigma 1$), and indoor scenes (Area2 and $\sigma 2$). This table is continued on the next page.</i>	43
3.2	<i>Channels, their ROC curve areas and Standard Error for Sunlit Scenes (Area1 and $\sigma 1$), and indoor scenes (Area2 and $\sigma 2$)</i>	44
3.3	<i>z statistic when CIELAB - a, $L\alpha\beta$ - β, HSV - H and Y_{xy} - y are compared with other discriminating channels.</i>	45
3.4	<i>z statistic when $L\alpha\beta$ - β, Y_{xy} - x, HSV - H, and CIELAB - a are compared with other discriminating channels.</i>	46

LIST OF FIGURES

1.1	<i>The liquid metal version of the robot in the movie Terminator II.</i>	2
1.2	<i>Given a high dynamic range image such as shown at the top, we present techniques to make objects transparent and translucent (left vases in middle and bottom images), as well as apply arbitrary surface materials such as aluminum-bronze (middle) and nickel (bottom).</i>	4
1.3	<i>Flow chart for creating transparent and translucent materials, and materials that can be represented by a BRDF.</i>	7
1.4	<i>Manipulating luminance values allows the specularity of an object to be adjusted. The original image is shown at the left. The middle image shows a luminance remapping resulting in a more specular object, whereas the rightmost image shows an object with reduced specularity.</i>	8
1.5	<i>A method to study change blindness. Observers typically fail to observe the difference in the wall in the two images. (Image provided by Ronald Rensink).</i>	12
1.6	<i>An example of the stimulus given to observers, who were asked to recognize the material of objects in images. Observers performed better at recognizing materials of objects in natural illumination. (Image provided by Roland Fleming).</i>	13
1.7	<i>The transmitted and reflected components of this object were computed for inconsistent lighting. (Image provided by Roland Fleming).</i>	14

2.1	<i>The arrays represent images of the same scene taken at R different exposures. For each vector $\mathbf{x}_{ijr}$ (the white elements shown in the centers of the arrays), $\mathbf{y}_{pqs}$ consists of a $p \times q$ neighborhood around (i, j) in all the R exposures (shown as the dark gray region around the center elements). p and q equal three in the above diagram. Note that the representation of the background is identical for all pixels at location (i, j).</i>	22
2.2	<i>Initial weights assigned to each pixel as a function of pixel value.</i>	23
2.3	<i>The left column shows the second, fifth, and seventh exposure of a sequence in which people are walking across the scene. The right column shows tonemapped HDR images that are generated using (top) weights determined only by absolute pixel values, (middle) weights that were estimated with a single iteration of our algorithm, (bottom) weights that were estimated after nine iterations of our algorithm.</i>	27
2.4	<i>The top two rows shows the second, fourth, sixth and eighth exposures of a sequence in which a book is being placed upon a stack of books. The bottom rows (left to right) shows tonemapped HDR images that were generated using weights determined only by absolute pixel values, and weights that were estimated with one, two, and fourteen iterations of our algorithm.</i>	28
2.5	<i>The first two images in the top row show two of the nine exposures used to generate the HDR images. The third image shows the tonemapped HDR image created without using any ghost removal algorithm. The bottom row shows tonemapped HDR images that were generated using (left) the ghost removal technique as proposed by [RWP05a], (middle) two iterations of our algorithm, and (right) ten iterations of our algorithm.</i>	29

2.6	<i>The top 3 rows show the individual exposures of the flower sequence, that were used to generate the HDR images. Noise has been added artificially to these images. In the bottom row, the image on the left was generated without using our algorithm. A single iteration of our algorithm reduces the noise in this image considerably, as shown on the right.</i>	30
2.7	<i>MSE values are plotted against the standard deviation of the added noise. The blue line indicates MSE values if our algorithm is not applied, while the red line plots MSE values obtained after four iterations of our algorithm.</i>	31
3.1	<i>Colors appear to have a bluer shade in shadows. The patches on the left have been obtained by cropping the image on the right.</i>	38
3.2	<i>Subset of colors used in the experiment. 3.2(a) is the plot of x,y coordinates of the colors and 3.2(b) is the histogram of their Y values.</i>	40
3.3	<i>ROC curves for the 11 color spaces provide qualitative assessment of the color spaces. (The remaining 5 are shown on the next page).</i>	48
3.4	<i>ROC curves for the 11 color spaces provide qualitative assessment of the color spaces.</i>	49
4.1	<i>Input to our algorithm. The top image is focused on the object, whereas the bottom image is focused on the background.</i>	62
4.2	<i>Comparison of the logarithm and sigmoid operators.</i>	63
4.3	<i>Plots of the variability computed for different scales. The numerator of (4.1) is used in the left plot, whereas the full division of (4.1) is plotted on the right. To construct V', one would have sum over all scales in either plot.</i>	63

4.4	<i>Plots of the variability computed for different scales. The numerator of (4.1) is used in the left plot, whereas the full division of (4.1) is plotted on the right. To construct V', one would have sum over all scales in either plot. These plots are for the middle scan line of the two images shown in Figure 4.1.</i>	64
4.5	<i>Signal V' for a single scale. The top image shows the difference between foreground and background image without the division, and the bottom images shows the same with the division in (4.1). Black encodes negative values, white encodes positive values and gray indicates zero, i.e. no difference between foreground and background.</i>	64
4.6	<i>Result after running the diffusion algorithm for 200 iterations, i.e. B^{200}. Foreground and background regions are white and black, and undecided regions are encoded as gray. The top image shows the result for V' without the division, and the bottom image is for V' with the division. In either case, a single scale was used ($n = 1$).</i>	65
4.7	<i>From left to right: foreground image, background image, results after applying sigmoidal compression, center-surround processing and filling-in.</i>	66
4.8	<i>Thresholded results, with minimal clean-up applied. Luminance and color cues could be applied to further improve the results. The full division in (4.1) was used. The input images are depicted in Figures 4.1 and 4.7.</i>	67
5.1	<i>System of Equations for Alpha Matte Extraction. This system does not yet incorporate the weights mentioned earlier. Note that $\mathbf{A}$ is a sparse matrix.</i>	77
5.2	<i>An example of a filled, sparse matrix $\mathbf{A}$. Elements with values 0, -1, and 1, are shown in gray, black, and white respectively.</i>	78

5.3	<i>Smoothness constraints are shown in green in this figure, for a 3×3 image. The number of constraints shown here are multiplied by 9 as they are present between each of the variables αF, αB, and B, in the 3 channels.</i>	78
5.4	<i>The top row shows two differently focused images of the same object. Focus information is used to obtain images in the second row. The left image is the output of the algorithm before diffusion, and right image is the output after only 60 iterations of diffusion. A threshold may easily be applied to this image to generate a binary matte, as shown in the left image in the third row. The right image shows how the object may now be placed into a new environment by using the matte. . . .</i>	80
5.5	<i>The top row shows two differently focused images of the same object. Focus information is used to obtain images in the second row. The left image is the output of the algorithm before diffusion, and right image is the output after 260 iterations of diffusion. A threshold is difficult to apply to this data to generate an accurate binary matte. An attempt is shown in the left image in the third row. The right image shows discrepancies when the object is placed into a new environment by using the matte.</i>	81
5.6	<i>Using the two differently focused images shown in the previous figure, a trimap may be generated using the method discussed in the last chapter. This trimap is the left image in the top row. The image on the right is the alpha matte that is generated using this trimap. The middle row shows that the object may now be placed more realistically into a new scene, or the matte may be used to create an image that is focused correctly both in the foreground and the background. The last row shows a results obtained with the method proposed by Levin et al.</i>	83

6.1	<i>Bilateral filtering applied to the object. The kernel size in the intensity domain was varied in this image sequence, while the spatial filter was kept constant.</i>	90
6.2	<i>The value of σ_{spatial} controls the smoothness of the depth-map. Here, its value is doubled from 1 to 2, 4 and 8.</i>	91
6.3	<i>A family of gradient reshaping functions.</i>	93
6.4	<i>Removing the elephant from the original image (left) is shown in the middle. The masking effect of further processing is demonstrated with the simulation of transparency (right).</i>	95
6.5	<i>The background image with the object removed is used to create an HDR environment map. First a circle is cut from the middle of the image. This is then placed in the image plane, and then extruded to become half the environment. The image is also pulled out in the opposite direction to form the other half of the environment. .</i>	97
6.6	<i>Test images used in this chapter.</i>	98
6.7	<i>The luminance values of the statue (left) were remapped to create a more specular (middle) and more diffuse (right) image.</i>	99
6.8	<i>Example of mapping a BRDF representing nickel onto a SIGGRAPH mug.</i>	101
6.9	<i>Examples of mapping arbitrary BRDFs onto an object.</i>	102
6.10	<i>A marble texture is applied to the left image, whereas a stone texture is applied to the right image.</i>	103
6.11	<i>A texture suitable for simulating transparency (left, see also middle panel of Figure 1.2, and translucency (right, as well as the right panel of Figure 1.2.)</i>	104
6.12	<i>Histogram of a high dynamic range image. The derivative is shown in black. . . .</i>	106

6.13	<i>Using the minimum in the derivative of the histogram, highlight pixels of the left image are detected, and for demonstration purposes painted red in the right image.</i>	107
6.14	<i>Transparency results demonstrated on different objects.</i>	108
6.15	<i>Colored glass (left) and darkened glass (right) simulated with a linear attenuation to achieve color (G in Equation 6.25, and non-linear scaling to simulate dark glass (Equation 6.26). Note that the refractions in these images are different from the ones shown in Figure 6.14 due to a different choice of filter kernel sizes of the bilateral filter, as well as a different choice of scale factor s.</i>	109
6.16	<i>Translucency applied to various different objects.</i>	109
6.17	<i>Comparison between HDR and LDR input for a transformation to a glassy material. The top row shows input and output in HDR (tonemapped for display), and the bottom row shows the low dynamic range equivalent. The HDR images were tonemapped with the photographic operator [RSS02a], whereas the LDR images were linearly scaled to fit the display range.</i>	110

CHAPTER 1 INTRODUCTION

When humans inspect their environment they see a world full of objects that interact with each other and with themselves. Human vision is a wonderfully complex system that analyzes the world, and under normal circumstances, is able to make sense of it. The analysis of an observer's environment involves figuring out the shape of each object, its distance to the observer, its material, lighting, color, motion, and texture. Several high level evaluations need to be made as well — for instance, is the object a threat to the observer?

Somewhere along the way, human vision is able to disentangle the lighting of the object from the intrinsic object reflectance [Hor74]. Detecting rough classes of materials, for instance being able to differentiate between plastic and steel, is effortless [ASL05]. If two otherwise identical objects are made of different materials, we are able to tell them apart. We use this ability to make various decisions on a daily basis. For instance, if an object does not 'look' edible, we will be inclined to not eat it, or if a diamond ring does not look shiny, we will probably not pay a high price for it. The manufacturing industry is aware of this phenomenon, and therefore takes great care to make its products with appropriately chosen materials [Ade01].

The impact of material perception is not limited to the manufacturing industry. Movies often need to show drastic material transformations that might not even occur in nature. Terminator II, in which a robot's appearance changes from human skin to liquid metal, is a classic example (See Figure 1.1). The most obvious and cumbersome way to bring about these transformations would be to draw them by hand, one frame at a time. This process would not only require great artistry, but would also be extremely time consuming and prone to imperfections. An image-based approach would be more convenient, and possibly lead to more plausible results.

Figure 1.1: *The liquid metal version of the robot in the movie Terminator II.*

Rendering requires at least a representation of the material, along with the illumination of the scene, and the 3D model of the object. The material may be specified by supplying a function that represents the material's response to incident illumination [War92, LFT97, APS00, AS00], or by supplying measured data of the same [Dan01, MPB03].

The illumination of the scene may be specified by capturing a high dynamic range (HDR) image of a shiny metal sphere in the scene for which the illumination is required [Deb02], or may be modeled with a 3D modeler, such as Alias Wavefront's Maya. The first is a somewhat cumbersome process that works as long as the scene actually exists. If it does not, and is impossible to recreate, then it would be impossible to capture the illumination of the scene. Modeling the environment realistically with a 3D modeler is a time-consuming, and difficult process.

Obtaining the object model is also fraught with similar problems. Typically, expensive scanning equipment would be required to obtain the geometric model of an object [MML98]. This equipment may only be used for objects with specific material properties. For instance, scanning an object with highlights will almost always produce a contorted 3D model, requiring post processing [PK04]. Scanning is limited to finding 3D models of objects that exist. Those that do not exist any more, or are not freely accessible to the public (such as objects in museums), cannot be rendered with a new material.

We aim to bypass these problems by employing an image-based approach to material editing, and introduce our approach in the remainder of this chapter. We define the problem in Section 1.1, and introduce various steps of our algorithm in Section 1.2. In Section 1.3, we specify some of the foreseeable uses of this work, and then discuss previous research in image-based material editing in Section 1.4. Previous work related to each of the individual steps in our algorithms is discussed in later chapters where we detail these steps. We conclude this chapter with a section discussing some of the idiosyncrasies of human visual perception, including possible ways to exploit these within image-based material editing.

1.1 Problem Definition

Our work takes as input a photograph of an object as well as an alpha matte outlining this object, and produces as output a new photograph where the object has been given the appearance of an entirely new material. The transformations presented range from applying a texture to the surface of an object, to the application of any arbitrary bi-directional reflectance distribution function (BRDF). In addition, we present techniques to handle light-matter interactions that are not well modeled with BRDFs, such as transparency and translucency. Examples of some of our results are shown in Figure 1.2.

Human vision tolerates some deviations from reality, (but not others). Indeed, sometimes, even enormous inaccuracies go un-noticed by observers. Because of the inherent ambiguity of visual cues, the human visual system makes certain assumptions when reconstructing a scene from an image. When these assumptions are violated, perception breaks down. For example, we find it harder to recognize objects that are illuminated or viewed from unusual angles [TKB98, LBD03]. Due to the nature of human visual perception, there may be a possibility of changing material properties in an image using simple image processing operations [Ade01].

Figure 1.2: *Given a high dynamic range image such as shown at the top, we present techniques to make objects transparent and translucent (left vases in middle and bottom images), as well as apply arbitrary surface materials such as aluminum-bronze (middle) and nickel (bottom).*

One of the key contributions of our work is showing instances in which the idiosyncrasies of the human visual system are exploited to allow a purely image-based editing of materials, although we leave validation of our results for future work. Our input — a single digital photograph — is not sufficient to accurately reconstruct the 3D shape of an object. We have therefore developed a set of approximate tools and techniques that allow material changes to be applied to an object without introducing perceptually objectionable artifacts. As a result, we would be able to show that all the pixels in our output are wrong. Nonetheless, we obtain visually plausible imagery, which we deem a more realistic goal than perfect physical accuracy.

In computer graphics, and for our work specifically, it is thus important to know which aspects of modeling, rendering and display violate the assumptions that the human visual system makes about its environment. Corners should not be cut for those aspects. However, significant approximations to a real environment do not necessarily lead to significant changes in perception, as long as certain basic assumptions are not violated. We discuss this issue further in Section 1.5.

1.2 System Overview

The general approach we adopt is first to acquire a high dynamic range (HDR) photograph of the object of interest by concatenating multiple exposures [DM97b]. Many of the techniques we present also work to some extent with standard (low dynamic range) images. However, we have found that HDR input yields more robust results, for several reasons: First, HDR luminance values yield smoother depthmaps, that are not clamped to 1 for object luminance values greater than 1. Second, highlights are easier to identify in HDR images, and third, HDR images are better suited to represent the environment in techniques such as image-based lighting. Finally, if the input image is noisy, the noise may be removed more easily and accurately if the image has several exposures (making noise removal an over-constrained problem), than if it is a standard image. We

have developed an algorithm to create HDR images that are free from noise and ghosting artifacts despite presence of noise and/ or movement in the individual exposures. We present this algorithm in Chapter 2.

Second, we create an alpha matte to separate the object from the background. Alpha mattes may be generated semi-automatically using sophisticated techniques [LST04, RKB04]. We have developed an algorithm to automatically extract approximate alpha mattes [RK05], and we present this in Chapter 4, along with results. We use the approximate mattes from this technique to generate trimaps, which we give as input to a second algorithm. This algorithm, which is presented in Chapter 5, produces an accurate alpha matte that can be given as input to the material editing framework. The alpha matte extraction algorithms use the $L\alpha\beta$ color space, the justification for which is provided earlier in Chapter 3.

Once the matte has been created, the object is then manipulated with several new and existing techniques. These techniques are described in Chapter 6. By aligning our image processing with the object boundaries, the illusion of a different material may be created. For the most straightforward material manipulations such as highlight removal or increasing the specularities of an object, the HDR pixel values of the object may then be remapped using simple non-linear functions. More complex transforms such as applying a BRDF to an object, or making the object transparent or translucent, require several more steps. These include for instance, extraction of depth and environment maps for the object, derivation of surface normals, sampling of the environment, and warping and/or color correction of the environment to simulate transparency or translucency. The main steps we take to bring about complex material transformations are shown in Figure 1.3.

Figure 1.3: Flow chart for creating transparent and translucent materials, and materials that can be represented by a BRDF.

1.3 Uses

Using the above processes, we are able to generate images that are otherwise time consuming, difficult, or impossible to create. Though this application is an important one, the impact of our work is not limited to it. First, by generating realistic looking images through processes that are not physically based, our work indicates that the human visual system does not require the image to be generated by a physically based process in order to be perceived as realistic (see Figure 1.4). Our work will therefore form the foundation of future investigative psychological research into the perception of materials.

Insight into the limitations of the human visual system could possibly have a profound impact on several fields of research, such as computer graphics, where realistic depictions of different

Figure 1.4: *Manipulating luminance values allows the specularity of an object to be adjusted. The original image is shown at the left. The middle image shows a luminance remapping resulting in a more specular object, whereas the rightmost image shows an object with reduced specularity.*

materials are desired [ASL05]. If more is known about which parts of the algorithm can be approximated, then computation time may be reduced drastically without visible loss of quality in the generated images - a feature of importance to the gaming industry. In computer vision, current research to recognize objects in images and videos is usually limited to obtaining features of the object that represent either the shape, color, or texture of the object, and classifying the object based on these features [Ade01]. Material properties other than color and texture are typically not used as a characteristic that could help identify the object. With knowledge of how the human visual system perceives materials, algorithms may be produced that use this information to partially or completely categorize objects in the scene. In robotics, insight into how humans perceive materials is important if robots are to be endowed with an understanding of their environment similar to that of humans [Ade01, ASL05]. Our work therefore provides exciting new venues of research in several areas.

1.4 Previous Work

Several image-based techniques have been developed successfully that use images to model the scene, as well as to render new images of the same scene [Oli02]. For instance, Oh et al take a single image and provide a variety of tools to the user to specify detailed 3D information of the scene. Among other options, the user may paint absolute depth values onto a scene, alter old ones, or use geometric primitives to facilitate the task of depth specification. The user is also given the ability to view the scene from different angles. Once the 3D structure has been specified, the user may edit the scene in various ways. For instance, the scene may be rendered from a new viewpoint, objects in the scene may be moved or re-colored, and new light sources may be added to the scene. To separate texture from illumination, Oh et al assume that illumination is represented by the large scale features, while textures are represented by the small scale features in the image. They can therefore separate these features by means of a bilateral filter [OCD01].

Using a purely image-based approach, the lighting of the scene in an image may also be altered [ED04]. The method proposed by Eisemann et al takes in two images as input, one taken with the available light in the scene (which is often insufficient), and the other taken with a flash. The algorithm then generates a new image with desirable features from both images, ambiance from the first, and detail and sharpness from the second image. The separation of features in the input images is again done with a bilateral filter. The final image therefore has large scale features from the first image, and small scale features from the second one.

Images have also been used to represent the lighting of a scene in a technique known as image-based lighting [Deb98, Deb02]. When HDR images are used, the objects in the scene are essentially being lit with light from the real world, and therefore the rendered images appear very realistic.

Based on a single image, Fang and Hart [FH04] have shown that it is possible to apply textures to objects.¹ They require the user to first specify the object in the image by hand, and then use this information to compute the surface normals of the object by shape from shading techniques. Shape from shading is known to be an ill-posed problem and its solution is not unique [Oli91, RT92, Pen94, BKY99], so the user is given the opportunity to modify them by hand. Next, the object is segmented into regions with similar surface normals.

Texture is then applied to each patch separately. It is foreshortened according to the surface normals of the pixels in each patch, thus giving the illusion that the texture follows the underlying surface curvature of the object. To further increase realism, a displacement map is generated by applying shape from shading techniques to the texture to obtain its depth information, and then modifying the shape of the object with this map. Finally, the optimal seams are found in overlapping regions of texture to disguise inconsistencies between the texture applied to adjacent patches.

This technique demonstrates an interesting image-based way of applying texture to an object. However, as this method re-textures objects, it is not able to account for lighting in the scene, i.e. shading information including highlights are lost. The interaction of the object with the environment is lost, unless some form of environment mapping is applied. Secondly, this method applies texture only to primarily Lambertian objects of uniform reflectance properties, since shape from shading techniques may only be used to recover the depth of objects that have these properties [ZFG05].

Recently, this work was improved by Zelinka et al [ZFG05]. One of the improvements is that they allow the user to specify the object boundary in a semi-automatic manner. Secondly, they improve the quality of the initial surface normals of the object by making changes to the shape from shading process. Specifically, their approach allows normal recovery of textured objects. The

¹Note that Fang and Hart's work was developed concurrently with ours.

object is first divided into clusters of color, and it is assumed that the change across these clusters is due entirely to change in albedo. Thus, any change in normals at these boundaries is ignored, and the resulting surface normals do not change across color boundaries.

Finally, they use jump-map based texture synthesis [ZG04]. A jump-map is a parallel image to a texture, which stores for each pixel, a small set of similar pixels in the image. Texture is generated by copying pixels from the input to the output. Whenever a sharp change in the image is encountered, a 'jump' is made to a new location on the texture, and copying proceeds from this new location. While this technique produces lower quality results than other texture mapping approaches, it is the fastest known method. Zelinka et al are able to apply texture to the object in interactive time and thus experiment with their parameters more easily.

The work by Zelinka makes an important contribution to a more robust object shape retrieval. However, objects that are to be textured, may have different colors, but are still required to be Lambertian. Furthermore, the clustering required during normal recovery may fail due to varying illumination in the scene, resulting in incorrect normal recovery. The method also shares the limitations of Fang and Hart's work by being limited to re-texturing of objects.

1.5 Human Visual Perception

In this section, we introduce some of the research in human visual perception that is relevant to our work. By giving several examples, we hope to argue that the human visual system is more sensitive to some distortions in images than others. If this is true, then it may be possible for us to change the materials of objects using an image-based approach. The transformed material should appear plausible to an observer as long as we only approximate those aspects of the images, which human visual perception is insensitive to.

Figure 1.5: *A method to study change blindness. Observers typically fail to observe the difference in the wall in the two images. (Image provided by Ronald Rensink).*

One of the well studied limitations of the human visual system is 'Change Blindness' [SL97]. This phenomenon is the failure of the human visual system to notice large changes in surrounding scenes. This typically happens when the change is unexpected, or when there is more than one stimulus (sound could accompany the visual stimulus, for instance), or when the change is introduced during a break in the visual stimulus, such as during a blank screen, a blink, or an eye movement [SA05]. One very interesting example is the result of an experiment conducted by Simons and Levin [SL98]. They exchanged heads of people in photographs that were given to the observers as input. This change occurred when the observers were shifting their eyes from one part of the image to another. About 50% of the observers failed to notice this change! Change blindness may be studied by showing two different images to an observer, one after the other, with a time in between where no image is being shown. Such a setup is explained in Figure 1.5.

Often, perceptual breakdown seems to occur because of violation of sub-conscious assumptions that humans make about their surroundings. Given that separating the effects of illumination and materials from scene geometry is an under constrained problem, it seems natural that humans make

Figure 1.6: *An example of the stimulus given to observers, who were asked to recognize the material of objects in images. Observers performed better at recognizing materials of objects in natural illumination. (Image provided by Roland Fleming).*

subconscious assumptions about their surroundings when interpreting visual stimuli. An example of such an assumption is the direction of incident light on an object. For example, people recognize objects around them effortlessly — until the illumination direction changes and objects are lit from unusual angles. Such illumination changes greatly hamper the ability of people to recognize the same objects [TKB98, LBD03].

Humans recognize the materials of objects by making some assumptions about the illumination in the scene [FDA01]. In an experiment conducted by Fleming et al [FDA01], observers were asked to recognize the material of spheres, lit by real-world illuminations from indoor and outdoor scenes, as well as by a variety of artificial illuminations. Figure 1.6 shows an example of such a stimulus. Subjects performed more accurately for stimuli viewed under real-world patterns of illumination than under artificial illuminations, suggesting that subjects use stored sub-conscious assumptions about the regularities of real-world illuminations to recognize materials [FDA01].

There is strong evidence that humans use specular reflections on an object to deduce its 3D shape [FTA04]. Fleming et al [FTA04] gave their subjects renderings of perfectly mirrored objects, without any context to specify the scene surrounding the objects. Subjects were able to deduce the correct shape of the object from these renderings. Nonetheless, people can tolerate physical inconsistencies between the transmitted and reflected components of the image [FJB04].

Figure 1.7: *The transmitted and reflected components of this object were computed for inconsistent lighting. (Image provided by Roland Fleming).*

For example, Fleming et al [FJB04] generated realistic looking translucent objects, for which the specular component was rendered under full-scene illumination, while the translucent component was rendered under a point light source. Figure 1.7 shows an object rendered in this manner. They also demonstrated that the presence of highlights greatly aided the illusion of translucency for transparent materials.

The above examples give us an exciting glimpse of the inner workings of the human visual system. They also provide preliminary guidance into the design of image-based methods for the manipulation of object materials. However, they are not sufficient to guide the design of such algorithms from beginning to end. Where unavoidable, we will therefore rely on common sense and visual comparison to argue the plausibility of our solutions. We see those instances as excellent candidates for future validation.

CHAPTER 2

HIGH DYNAMIC RANGE IMAGE GENERATION

The illumination of typical world scenes around us varies over several orders of magnitude, ranging from 10^{-3}cd/m^2 in starlight, to 10^5cd/m^2 in sunlight [Wan95]. Conventional sensors in image capture devices are only able to capture a limited part of this range. In order to map the desired scene range to properly exposed sensor data, the aperture, exposure time, and ISO value of the capturing device can be varied. In addition, photographers often need to ensure that strong light sources are not directly visible in the image.

Instead of taking these measures, spectrally weighted radiance of a scene may be captured more accurately by spatially varying pixel exposures [MN00], using multiple imaging devices, or devices that use special sensors [RWP05a]. These devices are expensive and will not be affordable for the average consumer for some years to come [RWP05a].

Meanwhile, there exist methods of obtaining high dynamic range (HDR) images using conventional devices [MP95, DM97a, MN99, RBS99]. Such techniques require the user to take several images of the same scene at different exposures, and apply a weighted average over these to compute radiance values of the scene. Mann et al were the first to propose such a method, which involved estimation of the camera response curve using the variation of pixel values across exposures, and then using it to linearize the exposures. The HDR image was formed from weighted averages of the pixels in these exposures [MP95]. Weights were computed to decrease the role of those exposures for each pixel, that were over- or under-exposed. Since then, modifications have been proposed to this basic process, specifically to the way in which the camera response curve was computed, and what function was used to assign weights to each exposure for a pixel [DM97a, MN99, RBS99].

A disadvantage of using multiple exposures to generate HDR images is that if there is any movement in the scene while exposures are being captured, the moving objects will appear in different locations in these exposures. Therefore, merging corresponding pixel values from different exposures to produce an HDR image will cause a ghosting effect. As such, existing techniques are only useful for creating HDR images of scenes that are completely still. This is rather restricting as most scenes contain motion. Without a solution to this problem, we are unable to use multiple capture techniques to produce HDR images of scenes that have any moving objects, such as people, animals, and vehicles. This is especially problematic in natural scenes where wind causes dynamic behavior in leaves, trees, flowers, clouds, etc.

Another problem arises if there is noise in individual exposures. Due to the manner in which HDR images are generated, noise is often compounded in the HDR version, thus significantly decreasing image quality.

In this work, we present a novel approach to robustly generating HDR images. The generated images have substantially reduced noise and ghosting artifacts. Unlike previous techniques, the proposed approach does not require any intermediate representation, such as optical flow or explicit detection. Instead, we generate an HDR image directly from image information. Thus, our approach is not conditional on the success of some intermediate process. We make an important deviation from the standard HDR image generation process by iteratively weighting the contribution of each pixel according to its chance of belonging to the noise-free (or static) part of the scene, (henceforth referred to as background), as well as its chance of being correctly exposed. We use a non-parametric model of the background, which enables us to compute a pixel's membership in the model, and therefore its weight. Since the model is non-parametric, we do not impose any restrictions on the background. The only assumption we make is that the exposure sequence predominantly captures the background, so that in any local region in image space, the number of pixels that capture the background is significantly greater than the pixels that capture the object.

Given this assumption, the neighborhood of a pixel in image space may serve as a reasonable representation of the background, and the probability of the pixel’s membership in this neighborhood may serve as the weight for that pixel.

The remainder of this chapter is organized as follows: We present previous work to address this problem in Section 2.1. We then describe the standard HDR image generation process in Section 2.2, and explain how we improve this process to remove noise and ghosting artifacts. In Section 2.3, we present images in which noise and ghosts have been removed using our technique, and compare these with results from previous work. Conclusions are presented in Section 2.4.

2.1 Previous Work

Removing noise from an HDR image is an overconstrained problem, if the removal is done at the time of image generation. As the HDR image is created using multiple exposures, there will be correctly exposed, noise-free exposures for each pixel, even if some exposures contain noise. Instead of generating the HDR image and later attempting to remove noise by computing new values for noisy pixels, noise may be removed at the time of image generation by carefully selecting the appropriate exposures to use for each pixel. While multiple works have addressed the problem of noise removal from a single image [Win01, VNV03, XP], few have exploited the over-constrained nature of the problem. Mitsunaga et al [MN99] remove noise from HDR images by giving larger weights to pixels with high signal to noise ratio, but make the assumption that noise is independent of the pixel value, which is often not the case. Robertson et al [RBS99] reduce noise in the HDR image by giving greater importance to longer exposures. If longer exposures have noise, then that noise would be amplified in the final image.

Considerable work has been done to remove ghosts from images. One solution to the problem is to track the movement of objects across exposures, and average pixel values according to this movement. For instance, Bogoni [Bog00] estimates an optical flow field between the different exposures, and warps these exposures such that all scene features are in accurate alignment. Similarly, Kang et al. [KUW03] use gradient-based optical flow between successive frames to compute a dense motion field, which is then used to warp pixels in exposures so that the appropriate values may be averaged together to generate a ghost-free HDR image. Techniques that use motion estimation work as long as the motion estimation is accurate. Currently, there are no approaches to motion estimation that work reliably for all kinds of movement. For instance, such techniques will fail for scenes with effects such as inter-reflections, specularities, and translucency [KUW03]. Even for simple Lambertian objects, motion estimation fails when objects appear in some exposures and not others due to occlusion [WA94].

One solution that avoids motion estimation altogether assumes that ghosting occurs in regions where the dynamic range is low enough to be represented accurately by a single exposure [RWP05a, UER01]. First, regions in the image where ghosting is likely to occur are detected. This may be done by computing the weighted variance of pixel values for each location in the image, and selecting regions where this variance is above a threshold [RWP05a]. Then, for each of these regions, a single exposure is selected from the exposure sequence, and its values are used directly in the HDR image. While this technique removes easily segmentable ghosts well, it will fail to capture ghosting in regions where the object color is similar to the background. Jacobs et al. [JWL05] address this issue by applying the threshold on a measure they derive from entropy, which is invariant to the amount of contrast in the data. Regions that are detected as possible ghost regions in this manner are again replaced with values from single exposures. This solution works well for many scenes, but fails when ghosting occurs in regions where the dynamic range is

high. This locally high dynamic range may be due to the presence of features in the scene such as windows, strong highlights, or light sources.

2.2 Iterative Noise and Ghost Removal

A set of exposures of the same scene may be used to obtain the camera response function $g(\cdot)$, of the capturing device, which is then applied to the exposures to convert them to radiance maps. The individual radiance maps, normalized by their exposure time, are averaged together to generate an HDR image [RWP05a]:

$$E(i, j) = \frac{\sum_{r=1}^R w(Z_r(i, j)) \left(\frac{g^{-1}(Z_r(i, j))}{\Delta t_r} \right)}{\sum_{r=1}^R w(Z_r(i, j))} \quad (2.1)$$

where $E(i, j)$ is the radiance at location (i, j) in the image, r denotes the exposure, R is the total number of exposures, $Z_r(i, j)$ is the value of the pixel at location (i, j) in the r^{th} exposure, (ranging from 0 to 255) and Δt_r is the exposure time of the r^{th} exposure. Note that the above is a weighted average of the radiance in each exposure. $w(\cdot)$ is typically chosen to diminish the contribution of pixel values that are under- or over-exposed. This weight is small for pixel values that are close to 0 and 255. The above equation is typically evaluated thrice for each pixel, for the red, green, and blue channels [RWP05a, JWL05]. However, since the three channels are correlated for typical natural images, a single weight for a pixel is necessary to preserve the original information [Bog00].

We make an important improvement to this process by computing weights that are determined not only by their probability of being correctly exposed, but also by the probability that they capture the background. Unlike the first attribute, which is determined by the pixel's value, there is no existing method to find the probability that a pixel captures part of a moving object. The object may be of any color, shape, or size, and its movement may be slow or fast, rigid or non-rigid. These features make it difficult to model the background, and to assign weights to pixels accordingly.

Given a set of R exposures, each of size $I \times J$, our objective is to compute a set of $I \times J \times R$ weights, that will be used to determine the contribution of each pixel in the exposure sequence.

If the pixel's color is very different from the colors of the pixels in its neighborhood, then it may be assumed that the pixel represents noise or a moving object in the image. In such cases, its probability of 'belonging' to the neighborhood is small, and it should be given a small weight in the HDR creation process. On the other hand, if a pixel's color is similar to the color of its neighboring pixels, it has a high probability of belonging to the background, and needs to be assigned a large weight during HDR generation. Since the probability of a pixel belonging to its neighborhood is correlated with the weight that the pixel should be assigned, we will treat the probability as the weight for each pixel.

We use a non-parametric estimation scheme to determine this probability so as to impose as few restrictions on our data as possible. Such estimation schemes give a high probability of membership to elements that lie in densely populated regions of the distribution's feature space. In particular, we use the kernel density estimator to compute this probability [Par62, Ros56]. To find the probability that a vector $\mathbf{x}$ belongs to a class F , an estimate can be computed:

$$P(\mathbf{x}|F) = M^{-1} \sum_{m=1}^M K_{\mathbf{H}}(\mathbf{x} - \mathbf{y}_m), \quad (2.2)$$

where M is the number of vectors in the class, $\mathbf{y}_m$ is the m^{th} vector in the class, $\mathbf{H}$ is a $d \times d$ bandwidth matrix (d being the space dimension), and

$$K_{\mathbf{H}}(\mathbf{x}) = |\mathbf{H}|^{-\frac{1}{2}} K(\mathbf{H}^{-\frac{1}{2}} \mathbf{x}), \quad (2.3)$$

where K is a d -variate kernel function. The d -variate Gaussian density is a common choice for the kernel K :

$$K_{\mathbf{H}}(\mathbf{x}) = |\mathbf{H}|^{-\frac{1}{2}} (2\pi)^{-\frac{d}{2}} \exp\left(-\frac{1}{2} \mathbf{x}^T \mathbf{H}^{-1} \mathbf{x}\right). \quad (2.4)$$

and we use this kernel function in our implementation.

Intuitively, if we represent a pixel with a vector, then kernel density estimation provides the means to compute the average distance of the pixel with the pixels in its neighborhood. This distance is based on the gaussian function, as seen in the last equation. The variance information for each dimension, which is required by the gaussian function, is provided in matrix $\mathbf{H}$.

We represent each pixel by a vector in a feature space such that $\mathbf{x}_{ijr} \in \mathbf{R}^5$, $i = 1, 2, \dots, I$, $j = 1, 2, \dots, J$, $r = 1, 2, \dots, R$. Three dimensions represent color, and two represent the location of the pixel in image space. By including each pixel's location in its representation, we ensure that neighboring pixels which are positioned near the pixel will play a larger role in determining its weight, then pixels farther away. We use the decorrelated $L\alpha\beta$ color space to represent each pixel's color values in the feature space, as this color space is perceptually uniform to a first approximation [RCC98a].

For a vector $\mathbf{x}_{ijr}$, the background is represented by a $p \times q \times s$ neighborhood, N , around the vector. Thus, for each $\mathbf{x}_{ijr}$, $F = \{\mathbf{y}_{pqs} | (p, q, s) \in N(\mathbf{x}_{ijr}), (p, q) \neq (i, j), \text{ and } s = 1, 2, \dots, R\}$, (see Figure 2.1). Thus, the neighborhood for a pixel in a particular exposure (which will be used in kernel density estimation to compute the pixel's weight), comprises of pixels in a small window around that pixel *in all exposures*. The representation of the background naturally remains the same for all the r pixels at location (i, j) .

Thus far, we have assumed that all vectors $\mathbf{y}_{pqs}$ are equally a part of the background. In practice, we know that many of these vectors represent pixels that are under- or over-exposed, and therefore do not represent the background well. These should not be considered a part of the background. Also, some vectors represent pixels that capture the moving object, and are also not a valid part of the background. Initially, while we do not know which vectors represent the moving object, we may reduce the effect of under- and over-exposed vectors by weighting their contribution in kernel density estimation [Goe03]. The weights assigned to each vector $\mathbf{y}_{pqs}$ are based on a simple hat

Figure 2.1: The arrays represent images of the same scene taken at R different exposures. For each vector $\mathbf{x}_{ijr}$ (the white elements shown in the centers of the arrays), $\mathbf{y}_{pqs}$ consists of a $p \times q$ neighborhood around (i, j) in all the R exposures (shown as the dark gray region around the center elements). p and q equal three in the above diagram. Note that the representation of the background is identical for all pixels at location (i, j) .

function [RWP05a] shown in Figure 2.2. This function is computed as follows:

$$w(Z) = 1 - \left(2 \cdot \frac{Z}{255} - 1 \right)^{12}, \quad (2.5)$$

where Z represents pixel values.

This produces three weights for each pixel. The final weight, w_{pqs} , is an average of these three. Using these weights, the probability that a vector $\mathbf{x}_{ijr}$ belongs to the background, now becomes:

$$P(\mathbf{x}_{ijr} | F) = \frac{\sum_{p,q,s \in N(\mathbf{x}_{ijr})} w_{pqs} K_{\mathbf{H}}(\mathbf{x}_{ijr} - \mathbf{y}_{pqs})}{\sum_{p,q,s \in N(\mathbf{x}_{ijr})} w_{pqs}} \quad (2.6)$$

Figure 2.2: *Initial weights assigned to each pixel as a function of pixel value.*

If our assumption holds and the neighborhood around each pixel predominantly represents the background, vectors which capture the moving object will get lower probabilities than vectors that capture the background. Therefore, once the probabilities have been computed for each vector $\mathbf{x}_{ijr}$, these can be used as weights of the corresponding pixels in HDR image generation. An HDR image created using these weights will show diminished ghosting as compared to an image generated using the initial weights, which are determined only by absolute pixel values.

Now that we have a better set of weights for each pixel, we can repeat the above process of kernel density estimation. This time the initial weights of the vectors $\mathbf{y}_{pqs}$ will be the ones computed in the previous iteration of kernel density estimation as shown below:

$$w_{pqs,t+1} = w(Z_s(p, q)) \cdot P(\mathbf{x}_{pqs}|F) \quad (2.7)$$

where $w(Z_s(p, q))$ is the initial weight for the pixel at location (p, q) with exposure s , $P(\mathbf{x}_{pqs}|F)$ is the weight of the pixel as determined by kernel density estimation, and $w_{pqs,t+1}$ is the weight that will be used in kernel density estimation in iteration $t + 1$. As before, we want to diminish the probability of pixels that are under- or over-exposed, and we multiply the newly computed weights with the initial weights obtained from the hat function before using them in density estimation. Now, even in cases where the distribution around a pixel is predominantly composed of under- or over-exposed pixels, they do not get high weights, as their initial weights are low.

2.3 Results

We used a Nikon D2X camera to capture all the exposures required to create HDR images. We used manual settings, doubling only the exposure time between consecutive captures of the same scene. A tripod was used to keep the camera stable during image capture. All HDR images shown in this chapter have been tonemapped with the Photographic tonemapping operator [RSS02a]. Our algorithm requires the user to specify the size of the neighborhood around a pixel, which we have kept at a constant 3×3 for all the results that we show in this section. The user is also required to specify the matrix $\mathbf{H} = \text{diag}(h_x, h_y, h_L, h_\alpha, h_\beta)$, which we kept as the identity matrix for our runs. Figure 2.3 shows our algorithm applied to a scene in which people are walking by. In this sequence, the amount of movement is large, and ghosting occurs in most of the image as a result. In Figure 2.4, ghosting is more localized, but involves the movement of a highlight (on the edge of the green book), which implies that the ghosting region has high dynamic range. Successive iterations of our algorithm remove both instances of ghosting from the HDR images.

We have compared our results with the approach described in [RWP05a], using *Photosphere*. Figure 2.5 shows results generated from a set of nine exposures of a flame. Due to air movement, and the high contrast of the flame with its background, this is a difficult scene to capture. The

method in [RWP05a] does not correctly identify ghosting artifacts in this scene. In addition, replacing parts of a flame scene with a single exposure, as proposed in [RWP05a], would yield a low dynamic range result in an area of the image that is inherently high dynamic range. As shown, our method overcomes these limitations, and produces a plausible result.

Our algorithm may also be used to remove noise in HDR images. The flower in Figure 2.6 appears clearer after a single iteration of our algorithm. We artificially added noise to the image sequence and applied our method to remove this noise. Noise was introduced by adding normally distributed random numbers to the color values of all the pixels in the exposures. After applying 4 iterations of our algorithm, we compute the mean squared error (MSE) between our result, and the HDR image made from the same sequence without adding noise. We repeated this process for different noise strengths, which we vary by changing the standard deviation of the noise distribution. In Figure 2.7, we compare these MSE values with those that we get if we do not apply our algorithm. Clearly, our method may be used to reduce noise in HDR images. Wherever there is not enough information for the algorithm, it fails gracefully.

2.4 Conclusions

In this chapter, we have presented an approach to create HDR images from multiple exposures, where noise and ghosting artifacts are suppressed. We achieve this by making a small, but significant deviation from the standard HDR image making process: We compute each pixel's weight, based on the probability of it belonging to its neighborhood. The only assumption we make is that this neighborhood is an adequate representation of the static part of the scene. Results presented in this chapter demonstrate the effectiveness of our approach.

In general, we have found that random noise typically takes fewer iterations to remove than ghosting artifacts. This is possibly due to the fact that moving objects are better represented in the background than random noise. Consequently, weights assigned to pixels that capture moving objects will decrease more slowly.

Figure 2.3: *The left column shows the second, fifth, and seventh exposure of a sequence in which people are walking across the scene. The right column shows tonemapped HDR images that are generated using (top) weights determined only by absolute pixel values, (middle) weights that were estimated with a single iteration of our algorithm, (bottom) weights that were estimated after nine iterations of our algorithm.*

Figure 2.4: The top two rows shows the second, fourth, sixth and eighth exposures of a sequence in which a book is being placed upon a stack of books. The bottom rows (left to right) shows tonemapped HDR images that were generated using weights determined only by absolute pixel values, and weights that were estimated with one, two, and fourteen iterations of our algorithm.

Figure 2.5: *The first two images in the top row show two of the nine exposures used to generate the HDR images. The third image shows the tonemapped HDR image created without using any ghost removal algorithm. The bottom row shows tonemapped HDR images that were generated using (left) the ghost removal technique as proposed by [RWP05a], (middle) two iterations of our algorithm, and (right) ten iterations of our algorithm.*

Figure 2.6: *The top 3 rows show the individual exposures of the flower sequence, that were used to generate the HDR images. Noise has been added artificially to these images. In the bottom row, the image on the left was generated without using our algorithm. A single iteration of our algorithm reduces the noise in this image considerably, as shown on the right.*

Figure 2.7: *MSE values are plotted against the standard deviation of the added noise. The blue line indicates MSE values if our algorithm is not applied, while the red line plots MSE values obtained after four iterations of our algorithm.*

CHAPTER 3

SELECTING A COLOR SPACE

In Chapters 4 and 5, we will present our algorithm to automatically extract an alpha matte given two differently focused images. Alpha matte extraction, like all segmentation algorithms, does not always produce accurate results due to complicating factors such as the presence of shadows [CGP01]. Therefore, it is important to use a color space that is not susceptible to changes across shadows. The color space also needs to be perceptually uniform, as we would like it to differentiate between colors only if humans do the same. In this chapter, we present a comparative study of color spaces in the context of shadow identification.

To successfully account for shadows in any algorithm, they must first be identified. Shadow identification may proceed by considering edges of shadows and distinguishing them from all other types of edges. Edges may be classified as either reflectance edges or luminance edges [Pal99]. Reflectance edges are changes in reflected light caused by changes in the reflectance of two adjacent surfaces. Luminance edges are changes in reflected light caused by different amounts of light falling on a single surface of homogeneous reflectance. These edges may be caused by cast shadows, reflected highlights on glossy surfaces, or changes in surface orientation. A subset of luminance edges is formed by cast shadows. We call these edges “shadow edges”. If the objects in the scene are predominantly Lambertian, shadow edges will account for most of the luminance edges in the scene. We are therefore interested in distinguishing shadow edges from all other types of edges.

Efforts to classify edges so far rely on simplified scenes to make the problem tractable. Camera calibration may aid edge classification [DF03]. Other simplifying assumptions may be that objects do not occlude each other [CL88] or that shadows are cast on flat surfaces [SCE01]. Moving

shadows may be detected along with moving objects using the difference between the current frame and a reference frame [PMC01].

Color has also been used to detect shadow edges. The most obvious characteristic of a shadow is that it appears darker than the same surface not in shadow, although shadowed regions still receive light from their surroundings. This non-direction specific light is called “ambient light”. The similarity of the spectral composition of ambient light to direct light depends in part on the albedos of objects in the scene, as well as the presence of participating media. A simplifying assumption may be that the average albedo of all surfaces in the scene is a spectrally flat gray [Hur86]. This is called the gray world assumption. With only one type of light source lighting the environment, and no scattering events taking place in participating media such as water and sky, a gray world allows ambient light to have the same spectral composition as the dominant light source in the scene. Thus, the difference between a shadow and non-shadow region will only be apparent in luminance, but not in chromatic content and this feature may be exploited for the purpose of shadow detection [CGP01, SCE01, GS03].

For sunlit scenes however, this assumption might not be true. While direct sunlight has a distinctly yellow spectrum, Rayleigh scattering causes the rest of the sky to appear blue [Min54, LL95]. This means that ambient light in outdoor scenes has a blue tinge. We may therefore expect shadows to also exhibit a blue color cast. Thus for typical sunlit scenes the gray-world assumption does not hold

The choice of color space in which images are analyzed may make the task of edge classification easier. We hypothesize that color opponent spaces qualify as the most suitable representations of color for the purpose of edge classification in sunlit scenes, since shadows, across which there is a change only in blue and yellow color, will not show in the red-green channel of opponent color spaces. We provide support for our conjecture by means of experimental results which rank a set of 11 color spaces with respect to suitability for shadow edge classification. The remainder

of this chapter is organized as follows: In Section 3.1, we specify how we compute the 11 color spaces we analyze in this work. Section 3.2 includes our theoretical analysis of the color spaces for the purpose of shadow detection, and this analysis is corroborated with experimental evidence in Section 3.3. We conclude with a discussion of our results in Section 3.4.

3.1 Color Spaces

The color spaces that we evaluate in this chapter are: *RGB*, *LMS*, *XYZ*, *Yxy*, *Luv*, *HSV*, *normalized RGB*, *AC₁C₂*, *CIELAB*, *Lαβ* and *Linear Lαβ*, of which *CIELAB*, *Lαβ*, *Linear Lαβ*, and *AC₁C₂* are opponent color spaces [WS82]. Most of these are standard color spaces, with the exception of *Lαβ*, which is the result of running a Principle Components Analysis on an ensemble of spectral images depicting natural scenes [RCC98b]. The result is a color space akin to opponent color spaces. This color space is defined in logarithmic space. For our comparisons we include a version of *Lαβ* in linear space to allow assessment of the importance of this non-linearity.

We now show conversion to all color spaces mentioned above, starting with device-dependent RGB color space. Normalized RGB is computed from RGB as follows:

$$\begin{aligned} \text{normalized} - R &= \tan^{-1} \left(\frac{R}{\max\{G, B\}} \right) \\ \text{normalized} - G &= \tan^{-1} \left(\frac{G}{\max\{R, B\}} \right) \\ \text{normalized} - B &= \tan^{-1} \left(\frac{B}{\max\{R, G\}} \right) \end{aligned} \tag{3.1}$$

If the white point of the capturing device is unknown, then a reasonable approximation is to assume a D65 white point. Conversion between RGB and device-independent XYZ color space is then

given by the following matrix transform [ITU90]:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 0.4306 & 0.3415 & 0.1784 \\ 0.2220 & 0.7067 & 0.0713 \\ 0.0202 & 0.1295 & 0.9394 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 3.0627 & -1.3928 & -0.4759 \\ -0.9689 & 1.8756 & 0.0417 \\ 0.0585 & -0.2286 & 1.0690 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

Further we may normalize to obtain a luminance channel Y and two chromatic channels x and y :

$$\begin{aligned} Y &= Y \\ x &= \begin{cases} \frac{X}{X+Y+Z} & \text{if } X+Y+Z > 0 \\ 0 & \text{otherwise} \end{cases} \\ y &= \begin{cases} \frac{Y}{X+Y+Z} & \text{if } X+Y+Z > 0 \\ 0 & \text{otherwise} \end{cases} \end{aligned}$$

The inverse of this normalization may be computed as:

$$\begin{aligned} Y &= Y \\ X &= \begin{cases} \frac{xY}{y} & \text{if } X, Y, Z > 0 \\ 0 & \text{otherwise} \end{cases} \\ Z &= \begin{cases} \frac{X}{x} - X - Y & \text{if } X, Y, Z > 0 \\ 0 & \text{otherwise} \end{cases} \end{aligned}$$

Alternatively, we may convert from XYZ color space to LMS cone space with the following matrix transform:

$$\begin{bmatrix} L \\ M \\ S \end{bmatrix} = \begin{bmatrix} 0.3897 & 0.6890 & -0.0787 \\ -0.2298 & 1.1834 & 0.0464 \\ 0.0000 & 0.0000 & 1.0000 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 1.9102 & -1.1121 & 0.2019 \\ 0.3710 & 0.6291 & 0.0000 \\ 0.0000 & 0.0000 & 1.0000 \end{bmatrix} \begin{bmatrix} L \\ M \\ S \end{bmatrix}$$

Conversion from LMS to $L\alpha\beta$ color opponent space proceeds by taking the logarithm of each pixel and then rotating the axes by the following transformation:

$$\begin{bmatrix} L \\ \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{3}} & 0 & 0 \\ 0 & \frac{1}{\sqrt{6}} & 0 \\ 0 & 0 & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & -2 \\ 1 & -1 & 0 \end{bmatrix} \begin{bmatrix} L \\ M \\ S \end{bmatrix}$$

$$\begin{bmatrix} L \\ M \\ S \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & -1 \\ 1 & -2 & 0 \end{bmatrix} \begin{bmatrix} \frac{\sqrt{3}}{3} & 0 & 0 \\ 0 & \frac{\sqrt{6}}{6} & 0 \\ 0 & 0 & \frac{\sqrt{2}}{2} \end{bmatrix} \begin{bmatrix} L \\ \alpha \\ \beta \end{bmatrix}$$

For linear $L\alpha\beta$, the conversion from LMS color space is the same, except that the logarithm is not applied.

Conversion from XYZ to CIELAB color space requires the specification of a white point, for which we take D65. This white point is specified in XYZ coordinates as $X_n Y_n Z_n = \{95.047, 100.0, 108.883\}$, so that for tristimulus values greater than 0.008856 the conversion to CIELAB becomes:

$$\begin{aligned} L &= 116 \sqrt[3]{Y/Y_n} - 16 \\ a &= 500 \left(\sqrt[3]{X/X_n} - \sqrt[3]{Y/Y_n} \right) \\ b &= 200 \left(\sqrt[3]{Y/Y_n} - \sqrt[3]{Z/Z_n} \right) \end{aligned}$$

The conversion between XYZ to CIELUV follows a similar set of equations:

$$\begin{aligned} L &= 116 \sqrt[3]{Y/Y_n} - 16 \\ u &= 16 L \left(\frac{4X}{X + 15Y + 3Z} - \frac{4X_n}{X_n + 15Y_n + 3Z_n} \right) \\ v &= 16 L \left(\frac{9Y}{X + 15Y + 3Z} - \frac{9Y_n}{X_n + 15Y_n + 3Z_n} \right) \end{aligned}$$

The conversion to AC_1C_2 from LMS is as follows:

$$\begin{bmatrix} A \\ C_1 \\ C_2 \end{bmatrix} = \begin{bmatrix} 2.00 & 1.00 & 0.05 \\ 1.00 & 1.09 & 0.09 \\ 0.11 & 0.11 & 0.22 \end{bmatrix} \begin{bmatrix} L \\ M \\ S \end{bmatrix} \quad (3.2)$$

and the inverse is as follows:

$$\begin{bmatrix} L \\ M \\ S \end{bmatrix} = \begin{bmatrix} 0.328 & 0.322 & 0.206 \\ 0.328 & 0.06353 & -0.185 \\ 0.328 & 0.1569 & 0.4535 \end{bmatrix} \begin{bmatrix} A \\ C_1 \\ C_2 \end{bmatrix} \quad (3.3)$$

Finally, the conversion between RGB and HSL/HSV is achieved with short algorithms which are outlined by Foley et al [FvF90].

3.2 Behavior of Color across Edges

As mentioned earlier, Raleigh scattering causes the sky to appear blue. This means that ambient light in outdoor scenes has a blue tinge. We may therefore expect that shadows will have a blue color cast, while lit regions will have a yellow cast. Thus a difference in luminance across a shadow edge in sunlit scenes will be accompanied by a difference in chromatic content, as shown

in Figure 3.1. In indoor scenes however, where it may be assumed that the gray world assumption holds and that no scattering events are taking place, ambient light will have the same chromatic content as direct light and there will be no change in color across shadow edges. The information

Figure 3.1: *Colors appear to have a bluer shade in shadows. The patches on the left have been obtained by cropping the image on the right.*

encoding in the human visual system gives us some insight into how humans treat color: the output of photoreceptors in the eye is recombined to form a new decorrelated color space where the axes consist of a luminance channel and two chromatic channels [Pal99]. One chromatic channel encodes red versus green, and the other encodes yellow versus blue. Research on monkey foraging behavior in Kibale Forest, Uganda confirms that the red-green channel is invariant to shadows, and may therefore aid primate vision in distinguishing edible fruit from leaves [TBP03]. While shadows do not show in the red-green channel, this is generally not the case for the yellow-blue channel. In sunlit scenes some difference will occur in the yellow-blue channel across shadow edges since the color of the scene will change from a yellow cast in lit regions to a blue cast in shadows across shadow edges. While color opponent spaces have been used in image processing tasks before [GS03], the two chromatic channels are invariably used indiscriminately, even though the yellow-blue channel is not invariant to shadows.

We hypothesize that opponent color spaces encode information in a way that is most suitable for edge classification for sunlit scenes due to the invariance of the red-green channel to shadows. In the following sections we test this hypothesis by measuring the suitability of 11 different color spaces in edge classification tasks for sunlit scenes.

3.3 Quantitative Evaluation

When an image is represented in a particular color space, one channel in the color space might encode the difference across reflectance edges only while another channel might encode the difference across both shadow and reflectance edges. In that case, classification of edges in the image may be done by first converting the image to such a color space, and then using these two channels to distinguish between the two types of edges. The accuracy of the classification depends on how well the first channel discriminates between the two kinds of edges. Hence, in our evaluation of color spaces, we base our rating on two characteristics. The color space should have a discriminating channel as well as a non-discriminating channel. In addition, the discriminating channel should be the best discriminator of edges among the color spaces compared, such that a suitable threshold over the change in values in the channel yields the most accurate classification of edges.

3.3.1 Data collection

We have photographed a set of partly shadowed, colored paint samples and cropped them to obtain patches, each patch consisting of only a single color, as it appears in shadow or lit regions. Figure 3.1 shows two patches obtained from a single color as it appears in or out of shadow. We have

photographed 143 different colors in a variety of sunlit scenes to ensure that the change across shadow edges is not always the same.

Ideally, the set of colors we use should fill the range of representable colors in current display devices. We have converted the set of colors to Yxy color space and plotted the *CIE xy* chromaticity coordinates and luminance values in Figure 3.2. From this figure, we conclude that our set of patches indeed fills both the range of representable chromaticities (indicated by the triangle) and the range of representable luminance values.

Figure 3.2: *Subset of colors used in the experiment. 3.2(a) is the plot of x,y coordinates of the colors and 3.2(b) is the histogram of their Y values.*

To compute the response of a color space to a shadow or reflectance edge, we convert the two patches representing the edge into that color space, average the values of each patch to obtain a single color value, and assess how different these values are for the two patches. For each of the shadow and reflectance edges, we compute the difference across the edges for all channels in all the 11 color spaces mentioned in the previous section.

We have generated another set of patches from the same color samples to represent shadow and reflectance edges in indoor scenes. We have obtained 349 shadow edges and 349 reflectance edges in both lit and shadow regions separately, all under tungsten light. We have measured the discriminating ability of channels separately for indoor and sunlit scenes and compare the results in the following sections.

3.3.2 Analysis

To measure the discriminating ability of all channels in the 11 color spaces, we use Receiver Operating Characteristic (ROC) curves. The area under an ROC curve has been empirically shown to compute the probability of correct classification, θ [HM82]. The standard error σ for this index may be computed as follows [HM82]:

$$\sigma(\theta) = \sqrt{\frac{(\theta(1-\theta) + (n_F - 1)(Q_1 - \theta^2) + (n_T - 1)(Q_2 - \theta^2))}{n_F n_T}}$$

where θ is the probability of correct classification, n_T is the number of true data items, n_F is the number of false data items, and Q_1 and Q_2 may be computed as follows:

$$Q_1 = \theta / (2 - \theta)$$

$$Q_2 = 2\theta^2 / (1 + \theta)$$

Quantitative comparison of 2 ROC curves is achieved by computing the z statistic. The value of this statistic indicates whether or not the difference in the discriminating abilities of the two classifiers as shown by the Area under the ROC curves (AUCs) is statistically significant, and it may be computed as follows:

$$z = (\theta_1 - \theta_2) / \sqrt{\sigma^2(\theta_1) + \sigma^2(\theta_2)}$$

If we let the probability of falsely rejecting the null hypothesis be 0.05, and that of falsely accepting it be 0.1, then a value of z above 1.96 indicates that discriminating abilities are indeed significant.

For our experiment, ROC curves are created by thresholding each channel's response to sample reflectance and shadow edges, and then plotting the true positive fraction against the false positive fraction. The terms positive or negative indicate the classification and true or false indicate correctness of the classification. The AUC for each channel measures the probability of correct edge classification by that channel. A suitable color space should have one channel that has an AUC close to 0.5, and another channel with an AUC close to 1. The most suitable color space is one for which the discriminating channel's AUC is closest to 1.

The ROC curves computed for each channel for sunlit scenes are shown in Figures 3.3 and 3.4 and the AUCs for each channel for both sunlit and indoor scenes are shown in Tables 3.1 and 3.2 along with their standard error. We note that all color spaces except *normalized RGB* color space have at least one discriminating and one non discriminating channel, and thus provide some discrimination between the two types of edges. For all these color spaces, we select the channel with the highest AUC. We select the four color spaces with the highest AUCs and compare them to all other color spaces using z values for sunlit scenes in Table 3.3 and for indoor scenes in Table 3.4.

For outdoor scenes, *CIELAB*'s a channel as well as $L\alpha\beta$'s β channel provided the highest AUC. These two channels were not statistically different, since their z -score remains below 1.96. The y channel of the Yxy color space and H channel of HSV have the second highest AUC value after *CIELAB* and $L\alpha\beta$. They were both statistically different from *CIELAB*, but there was no statistical difference between the two. For indoor scenes the two highest ranked color channels are $L\alpha\beta$'s β channel and the x channel of the Yxy color space, whose difference in performance is not statistically significant. The difference in the performance of the third and fourth color spaces is statistically insignificant, but is significant when their performance is compared with that of the first two.

Channel	Area1	$\sigma 1$	Area2	$\sigma 2$
<i>RGB - R</i>	0.435	0.013	0.598	0.018
<i>RGB - G</i>	0.477	0.013	0.573	0.018
<i>RGB - B</i>	0.649	0.012	0.567	0.018
<i>XYZ - X</i>	0.362	0.012	0.553	0.019
<i>XYZ - Y</i>	0.397	0.013	0.547	0.019
<i>XYZ - Z</i>	0.590	0.013	0.521	0.019
<i>LMS - L</i>	0.384	0.013	0.554	0.019
<i>LMS - M</i>	0.412	0.013	0.545	0.019
<i>LMS - S</i>	0.590	0.013	0.521	0.019
<i>Yxy - Y</i>	0.397	0.013	0.547	0.019
<i>Yxy - x</i>	0.795	0.009	0.902	0.009
<i>Yxy - y</i>	0.798	0.009	0.881	0.010
<i>Luv - L</i>	0.397	0.013	0.547	0.019
<i>Luv - u</i>	0.588	0.012	0.643	0.017
<i>Luv - v</i>	0.647	0.012	0.612	0.018
<i>HSV - H</i>	0.798	0.009	0.847	0.011
<i>HSV - S</i>	0.707	0.011	0.760	0.015
<i>HSV - V</i>	0.372	0.013	0.549	0.019

Table 3.1: Channels, their ROC curve areas and Standard Error for Sunlit Scenes (*Area1* and $\sigma 1$), and indoor scenes (*Area2* and $\sigma 2$). This table is continued on the next page.

Channel	Area1	$\sigma 1$	Area2	$\sigma 2$
<i>normalized - R</i>	0.796	0.009	0.903	0.009
<i>normalized - G</i>	0.806	0.009	0.891	0.010
<i>normalized - B</i>	0.782	0.010	0.790	0.014
<i>CIELAB - L</i>	0.394	0.013	0.569	0.018
<i>CIELAB - a</i>	0.826	0.008	0.848	0.012
<i>CIELAB - b</i>	0.762	0.010	0.778	0.015
<i>L$\alpha\beta$ - L</i>	0.422	0.013	0.532	0.019
<i>L$\alpha\beta$ - α</i>	0.798	0.009	0.910	0.009
<i>L$\alpha\beta$ - β</i>	0.818	0.009	0.895	0.009
<i>linear L$\alpha\beta$ - L</i>	0.408	0.013	0.530	0.019
<i>linear L$\alpha\beta$ - α</i>	0.682	0.012	0.626	0.018
<i>linear L$\alpha\beta$ - β</i>	0.635	0.012	0.613	0.018
<i>AC₁C₂ - A</i>	0.388	0.013	0.549	0.019
<i>AC₁C₂ - C₁</i>	0.627	0.012	0.653	0.017
<i>AC₁C₂ - C₂</i>	0.679	0.012	0.601	0.018

Table 3.2: Channels, their ROC curve areas and Standard Error for Sunlit Scenes (Area1 and $\sigma 1$), and indoor scenes (Area2 and $\sigma 2$)

Channel	<i>CIELAB</i> - a	$L\alpha\beta$ - β	<i>HSV</i> - H	Y_{xy} - y
<i>RGB</i> - B	11.868	11.274	9.725	9.757
<i>XYZ</i> - Z	15.328	14.729	13.177	13.209
<i>LMS</i> - S	15.328	14.729	13.177	13.209
Y_{xy} - y	2.140	1.532	0.032	-
L_{uv} - v	12.011	11.414	9.869	9.901
<i>HSV</i> - H	2.173	1.565	-	0.032
$L\alpha\beta$ - β	0.609	-	1.565	1.532
<i>CIELAB</i> - a	-	0.609	2.173	2.141
<i>Linear</i> $L\alpha\beta$ - α	9.889	9.290	7.744	7.776
AC_1C_2 - C_2	10.069	9.471	7.925	7.957

Table 3.3: z statistic when *CIELAB* - a , $L\alpha\beta$ - β , *HSV* - H and Y_{xy} - y are compared with other discriminating channels.

Four of the color spaces we consider show color opponency: *CIELAB*, $L\alpha\beta$, linear $L\alpha\beta$ and AC_1C_2 . While *CIELAB* is a perceptually uniform space, and $L\alpha\beta$ is perceptually uniform to a first approximation, the latter two are linear color spaces. The performance of the perceptually uniform color spaces is significantly better than their linear counterparts. To verify the performance of the red-green channels versus the yellow blue channels in the perceptually uniform color opponent spaces, we have computed z statistics between red-green channels and the blue-yellow channels of both these spaces. We expect a difference between these pairs of channels and find this to be the case. The a channel in *CIELAB* performs significantly better than the b channel, as the z statistic for this comparison is 4.733. In $L\alpha\beta$, the α channel performs better than the β channel, but the z statistic is 1.515 and so this difference is not significant.

Channel	$L\alpha\beta - \beta$	$Y_{xy} - x$	$HSV - H$	$CIELAB - a$
$RGB - R$	15.565	15.026	11.573	11.674
$XYZ - X$	17.417	16.876	13.406	13.510
$LMS - L$	17.376	16.836	13.366	13.468
$Y_{xy} - x$	0.639	-	3.754	3.651
$Luv - u$	13.738	13.195	9.732	9.833
$HSV - H$	4.374	3.754	-	0.105
$L\alpha\beta - \beta$	-	0.639	4.374	4.271
$CIELAB - a$	4.3	651.0	105.271	-
$Linear L\alpha\beta - \beta$	14.454	14.421	10.968	10.558
$AC_1C_2 - C_1$	13.344	12.799	9.330	9.431

Table 3.4: z statistic when $L\alpha\beta - \beta$, $Y_{xy} - x$, $HSV - H$, and $CIELAB - a$ are compared with other discriminating channels.

3.4 Discussion

For the purpose of edge classification we have evaluated several color spaces using ROC curves. For outdoor scenes, our results support our hypothesis that color opponent spaces perform well. The two perceptually uniform color opponent spaces prove to be the most suitable choices. For indoor scenes, other color spaces tend to perform as well as color opponent spaces. For applications that would process both indoor and outdoor scenes, the $L\alpha\beta$ color space would be a suitable choice. In addition, by transforming the data to log space the pixel data along each axis becomes symmetrical. This symmetry, along with the color opponency appears to aid in the separation of shadow and reflectance edges.

In conclusion, we have shown that color is an important cue in the classification of edges for sunlit scenes even when we do not make assumptions about the gray world and the effects of participating media. Through experimental results, we have validated the conjecture that opponent color spaces are a good choice for the purpose of edge classification.

Figure 3.3: ROC curves for the 11 color spaces provide qualitative assessment of the color spaces. (The remaining 5 are shown on the next page).

Figure 3.4: ROC curves for the 11 color spaces provide qualitative assessment of the color spaces.

CHAPTER 4

ALPHA MATTE GENERATION

As outlined in Chapter 1, the input into our material editing process consists of a high dynamic range image along with an alpha matte specifying the object in the image. The purpose of the alpha matte is to separate the object from the background. Our algorithms use this input to change the material of the object in a variety of ways. Creating a matte by hand for each image that the user wants to alter with techniques may be cumbersome, and the user may generate a matte more conveniently with the help of semi-automatic or automatic matte generation techniques. In this chapter, we discuss our method to obtain alpha mattes automatically. Two images may be composited by linearly blending them using an alpha matte. The alpha matte indicates for each pixel how much of the foreground and how much of the background should be blended. To create an alpha matte from an image, which is called *matting* or *pulling a matte*, an object would normally have to be segmented from the background. This is an under-constrained problem, and most solutions are therefore user-assisted. The quality of the results that our algorithm generates is typically not good enough for direct use as input into our material editing techniques. We therefore use this approach to obtain a reasonable estimate of the matte, and then use another algorithm (discussed in the next chapter), to obtain an accurate alpha matte.

4.1 Previous Work

The brute force approach to matting is to draw the outline of an object by hand using a drawing program such as Photoshop. This is both tedious and error-prone, and has recently led to the development of several algorithms which aim to replace this type of user input with simpler user input. The first is called *intelligent scissors*, and constitutes a lasso tool which snaps to nearby

edges [MB95, MB99]. *Lazy snapping* is a more recent boundary editing tool, proposed to further take the tedium out of user-assisted alpha matte extraction [LST04]. For video sequences outlined objects may be tracked over time using *roto-scoping* [MYT95], a technique which tracks editable boundaries over time.

Other algorithms, such as Poisson matting [SJT04], Bayesian approaches [CCS01, CAC02], graph cut algorithms [GPS89, BJ01] and others [LLY06], take as input a user-supplied trimap, which classifies each pixel as definitely foreground, definitely background or undecided. The algorithm then classifies the undecided pixels. The *GrabCut* algorithm requires the user to draw a rectangle around the object of interest and then uses iterated graph cuts to perform the object segmentation [RKB04]. *Level set* methods [CKS97] may also be used for object segmentation, but may require some (user-assisted) initialization. Several of these algorithms allow further user input to refine initial solutions.

To automatically separate an object from its background, an algorithm could segment an image based on one or more cues such as texture or luminance gradients. In such cases it may be necessary to make additional assumptions on the content of the image, for instance that the image contains depth of field, or that the object is in sharp focus. The local variance in the image is then a reasonable indicator of whether a pixel belongs to the foreground or the background [WLG01, WPG02]. This problem is related to the recovery of depth from focus; see Ens and Lawrence [EL93] for an overview.

Our interest is in automatically separating an object from its background. To aid this process, we require two registered images—one with the object in sharp focus, and one with the background in sharp focus (Figure 4.1). Rather than relying on a single absolute measure of local variance, this gives us the ability to pair-wise compare pixels between these two images. The advantage is that the input is more reliable such that accurate object segmentation becomes easier. The disadvantage is that the user needs to supply two images. However, we believe that using two photographs as

input is much less of a burden to the user than manually creating trimaps, or babysitting semi-automatic alpha extraction algorithms.

Concurrent to our work, McGuire et al [MMP05] have developed a similar idea to compute alpha mattes for videos. They use differently focused captures of the same scene to compute an initial estimate of the alpha matte, and refine this by solving an optimizing function. Their initial estimate is computed in a way similar to ours, but as they do not include a diffusion step (that we discuss later in the chapter), they can only obtain reasonable estimates if the whole image has sufficient texture. As they work with standard images, detail is often missing, (some of which may have been available in an HDR capture of the same scene). Furthermore, their work requires the use of a special device, which simultaneously captures multiple differently focused frames of the scene, instead of just one. As the optimization part of their work is related more closely to our algorithm which is introduced in the next chapter, we leave discussion of this work for later.

We have found that the tools required to compare the two differently focused images can be closely modeled on current knowledge of early human vision, including non-linear response compression, center-surround mechanisms and a filling-in stage. While we generally follow the same sequence of processing steps as the human visual system (HVS) does, the problem we are solving is different from that solved by the HVS, and we therefore introduce specific deviations into the design of our algorithm.

In the following section, we outline a short sequence of steps modeled after the HVS and show how minor deviations will allow us to detect a foreground object from its background using only two differently focused images as input. We show results in Section 3.3.2, followed by conclusions in Section 4.4.

4.2 Algorithm

The basis of our approach is that two images of the same scene, but focused at different distances, provide complementary information which may be analyzed for the purpose of pulling an alpha matte. Such image processing is generally less accurate on quantized data. We therefore resort to the use of high dynamic range (HDR) imaging [RWP05b], and therefore both foreground and background images are reconstructed from multiple exposures.

To allow pixel-wise comparison between a pair of images, we assume the images are in registration. A sufficiently high accuracy is obtained by photographing the images using a tripod. This is not a severe limitation over acquiring single HDR images, since any multi-exposure technique requires a reasonable level of alignment. In case further alignment is necessary, an image alignment technique may be applied [War03].

The algorithm consists of four consecutive steps. The first three are modeled on human vision and consist of non-linear response compression, center-surround analysis as well as a filling-in stage. Finally, a reasonable trimap may be generated by thresholding the output. Each component part of our algorithm is discussed in the following sub-sections.

4.2.1 Non-linear response compression

If the camera is operated in (semi-) automatic mode, such as aperture priority mode, the overall luminance levels within the two images may be somewhat different. To make the images comparable some normalization should occur. A first idea may be to apply all subsequent processing in the logarithmic domain. This is a reasonable approach, but very dark areas in either image may now be represented by negative numbers which may throw off later processing. In particular, a difference

of Gaussians scheme is sensitive to this issue. In addition, the cross-over between positive and negative numbers after taking the logarithm occurs for a luminance of 1 cd/m^2 , which is arbitrary.

To achieve a similar compression, but without the disadvantages of introducing arbitrary negative numbers, we scale each image by its log average luminance $\bar{L}$, and then apply a sigmoid¹:

$$\bar{L} = \exp \left(\frac{1}{N} \sum_{(x,y)} \log (\delta + L(x, y)) \right)$$

$$L'(x, y) = \frac{L(x, y)}{L(x, y) + \bar{L}}$$

In this equation, N is the number of pixels in each image, and δ is a small constant (we used 10^{-8}) to avoid taking the logarithm of 0. This approach compresses the range of luminance values around $\bar{L}$ in a manner similar to logarithmic compression, but also introduces asymptotically less compression for very light and very dark regions. The similarity of these two functions is illustrated in Figure 4.2. Applying this function conditions the input images such that pixel-wise comparison becomes feasible. In addition, we note that such non-linear response compression is a feature of the human visual system, and is measured both psychophysically [HF79, Ade82] and using single-cell recordings (in other species) [NR66, KD75]. In a different context, variations of the above non-linear scaling are used as tone reproduction operators [Sch94, THG99, PTY00, RSS02b, RD05]. The above non-linear scaling is applied independently to both foreground and background images. Note that these computations proceed in floating point format. As such, the compression reduces the dynamic range, but does not introduce quantization artifacts.

¹Luminance is computed from RGB signals using $L = 0.2126 + 0.7152 + 0.0722$ as specified by the International Telecommunication Union ITU Rec 709.BT [ITU90].

4.2.2 Center-surround analysis

Earlier techniques to detect whether a region of pixels is in focus or not, rely on local variance estimates [WLG01, WPG02]. This is the approach we follow as well by noting that the variability of a local region of pixels with respect to a somewhat larger region of pixels is estimated by subtracting their averages. Thus, a difference between two Gaussian blurred images yields a local estimate of variability for each pixel. A Gaussian response R is achieved by convolving an image L' with a Gaussian filter kernel F_σ :

$$F_\sigma(x, y) = \frac{1}{\pi\sigma^2} \exp\left(-\frac{x^2 + y^2}{\sigma^2}\right)$$

$$R_\sigma(x, y) = L'(x, y) \otimes F_\sigma(x, y)$$

with $\otimes$ the convolution operator, and σ specifying the amount of blur. By computing two such blurred images with different sized kernels, and subtracting the larger from the smaller, we obtain a measure of local variability $V_{\sigma_1\sigma_2}$:

$$V_{\sigma_1\sigma_2}(x, y) = \frac{R_{\sigma_1}(x, y) - R_{\sigma_2}(x, y)}{R_{\sigma_2}(x, y)}$$

We choose the difference between the two Gaussians to be $\sigma_2 = \sigma_1 + 1$, which, although less common than a doubling of the kernel size, largely avoids the feature tracking problem that is common to many scale-space problems [Wit83]. The division by R_{σ_2} is included to locally normalize the result. Since the sigmoidal compression is akin to taking the logarithm for the mid-range of luminance values, the difference of Gaussians essentially computes a measure of local contrast.

The above computation is a crude and simplified approximation of the receptive fields measured for ganglion cells in the retina (although for our purposes sufficiently powerful) [Kuf53, Mar82, Dow87, Pal99]. Its sensitivity to contrast effectively encodes edges, which are transmitted to the brain. In uniform regions, the two quantities subtracted are identical, and no signal is generated. Such an encoding is therefore efficient, since it is only sensitive to changes in signal. A possible

reason for this encoding is to overcome the communication bottleneck that is created by the optic nerve.

The human visual system is thought to operate at different scales, i.e. the above center-surround computation may be repeated for increasingly large σ . Such a stack of analyzers may measure variability at different scales, each scale usually being twice as large as the previous. We typically use $\sigma_1 \approx 0.35$ for the smallest scale. The local variability V is then summed over all scales n :

$$V = \sum_{i=1}^n |V_{\sigma_i \sigma_{i+1}}(x, y)|$$

We take the absolute value at each scale because for our application we are not interested in whether the surround has a higher or lower value than the center or vice-versa; either difference is equally important.

For the problem of alpha matte creation, a small adjustment of this procedure allows the comparison between two complementary images. This adjustment is a deviation from how the human visual system works, and is made to cater to our specific problem at hand. To classify a pixel as belonging to the object being segmented, the local variability in the foreground image V^{fg} should be higher than the variability in the background image V^{bg} . Thus, the above equation may be refined as follows:

$$V' = \sum_{i=1}^n \frac{|V_{\sigma_i \sigma_{i+1}}^{\text{fg}}(x, y)| - |V_{\sigma_i \sigma_{i+1}}^{\text{bg}}(x, y)|}{|V_{\sigma_i \sigma_{i+1}}^{\text{fg}}(x, y)| + |V_{\sigma_i \sigma_{i+1}}^{\text{bg}}(x, y)| + \epsilon} \quad (4.1)$$

The constant $\epsilon = 10^{-2}$ is included to avoid division by zero. For pixels that belong clearly to the object, V' will be larger than 0. Background pixels are classified as $V' < 0$. Undecided pixels, for instance in uniform regions, yield a value of $V' \approx 0$. The division was introduced to amplify the system's sensitivity in near-uniform regions. This allows the correct classification of edge pixels in nearly uniform regions, albeit at the cost of mis-classification of other edges. Dependent on the images being segmented, this division may be omitted, although we have found the extra division to yield a useful signal for the subsequent filling-in process.

For a step function, acting as a foreground image, and a Gaussian blurred step function ($\sigma = 1$ pixel), acting as an out-of-focus background image, we plot the numerator of (4.1) for different scales in Figure 4.3 (left). Similarly, Figure 4.3 (right) shows the full division of (4.1). An important observation is that the influence of the step does not fan out significantly for larger scales. This is due to our choice of scales, which is one pixel larger for each subsequent scale ($\sigma_{i+1} = \sigma_i + 1$). This implies that in this formulation features track well across scales, and the typical problem associated with scale-space solutions of edges shifting according to scale does not apply here [Wit83]. Were we to choose a doubling of the filter kernel size at each scale, then we would have to solve the feature tracking problem in our application. Figure 4.4 shows that for real images, features can also be reliably tracked across scales.

This has the implication that, for our purposes, a multi-scale approach does not yield many advantages over a single scale approach. Edges that are present at the smallest scale, are also present at higher scales, although there they are detected with a smaller amplitude. Thus, although the user is free to choose the number of scales n in this computation, we have found that in general only one scale is sufficient, i.e. we choose $n = 1$.

An important observation about the signal V' is that it is predominantly positive for pixels belonging to the object, and predominantly negative for pixels belonging to the background. It can thus be seen as an edge signal with its sign indicating foreground or background, although textured regions also cause appropriate foreground and background signals. Low contrast areas are encoded with (near) zero values. For a single scale, and for both formulations of (4.1), i.e. numerator only and full division, the signal V' is shown in Figure 4.5. While positive and negative values correctly indicate foreground and background pixels, there are large regions of undecided (gray) pixels which need to be filled-in.

4.2.3 Filling-in

The edge signal V that is transmitted to the brain (or V' used for our alpha matte pulling application), may be thought of as forming the basis for further processing. One of the tasks faced by the human visual system is reconstruction of the original signal, for instance to form an idea of brightness. Integrating the edge signal into a reconstructed representation of the scene, is called filling-in.

This process is under-constrained, since many different ways to reconstruct a scene from edge information are possible. It is thought that the human visual system makes specific assumptions about the world to enable a reasonable reconstruction. For instance, it may be assumed that the luminance distribution between sharp discontinuities is more or less uniform. This leads to a computational model of filling in which employs a (reaction-) diffusion scheme to propagate edge information into the interior of uniform regions [NPH01].

A different assumption may be that the real world exhibits the well-known $1/f$ image statistic [Rud97a, Rud97b]. Given that low spatial frequencies are attenuated in the edge signal, but are not completely absent, a signal may be reconstructed by boosting low frequencies according to this $1/f$ statistic [DB03].

Although either approach could yield results pointing generally in the right direction, we use the diffusion-based solution, where the diffusion algorithm is given by [PM90, Wei97]:

$$B_{x,y}^0 = V'(x, y)$$

$$B_{x,y}^{t+1} = B_{x,y}^t + \frac{\lambda}{\| (N_{u,v}) \|} \sum_{(u,v) \in N_{x,y}} (B_{u,v}^t - B_{x,y}^t) g(P_{u,v}, P_{x,y})$$

To guarantee convergence, λ should be less than or equal to one. In our experiments, we set λ equal to one. This equation iterates $t_{\max}$ times, and at each iteration each pixel receives some energy from its immediate neighbors. The amount received depends on the diffusivity parameter

$g()$. Thus, to make a diffusion equation compute a useful quantity, care should be taken to design an appropriate diffusivity. In our case, we wish to simulate a filling-in process. As there is noise in our data, we would like diffusion to occur across pixels with different signs, as well as those with the same sign. To guide the diffusion process, we use another discriminating feature: color. If there is a large color difference between two adjacent pixels, then it is probable that they belong to different objects (that are possibly differently focused). The diffusivity parameter is kept small for these cases. The parameter is large for diffusion across pixels that have similar color values. The following function satisfies these requirements:

$$g(P_{u,v}, P_{x,y}) = \frac{\exp \frac{(L_{u,v} - L_{x,y})^2}{2\sigma_L^2} \frac{(\alpha_{u,v} - \alpha_{x,y})^2}{2\sigma_\alpha^2} \frac{(\beta_{u,v} - \beta_{x,y})^2}{2\sigma_\beta^2}}{\sigma_L \sigma_\alpha \sigma_\beta (2\Pi)^{3/2}} \quad (4.2)$$

L , α , and β are the pixel values in $L\alpha\beta$ colorspace. The parameters $\sigma_L, \sigma_\alpha, \sigma_\beta$ are all kept 1. Diffusion for 200 iterations ($t_{\max} = 200$) yields an image as shown in Figure 4.6. In this image, gray signifies zero (undecided) pixels, whereas lighter values indicate foreground and darker pixels indicate background. At least for this image, the foreground is clearly differentiated from the background, thus requiring only minor post-processing to create an alpha matte estimate. In Section 4.3 we show further results.

4.2.4 Clean-up

The filling-in process creates results that are nearly good enough to serve as initial alpha matte estimates. The image may be thresholded to create a hard (binary) alpha matte, although small regions will be mis-classified. The borders in the image where background and foreground meet can be set to 0.5 to indicate areas where the alpha matte needs to be fine-tuned. To do this, an algorithm is required that allows for fractional alpha values in the case of objects with fuzzy boundaries.

This conversion to gray will also eliminate small mis-classified regions, which can be correctly classified in the second algorithm.

4.3 Results

Several foreground and background images as well as their results after applying the diffusion algorithm are shown in Figure 4.7. We used a Nikon D2H to capture these images. This camera allows auto-bracketing with nine different exposures, each spaced one f-stop apart. The focusing mechanism of this camera allows us to focus on different parts of the scene without physically moving the camera.

No specific alignment between each pair of images was applied, although the images were taken using a tripod. The parameters to the various components of our algorithm were all set to the default values as indicated in Section 4.2.

These results indicate that a pair of differently focused images affords a reasonable degree of robustness. The giraffe image is perhaps less convincing than the others, but this can be ascribed to the presence of glass surfaces which complicates the analysis.

The computation time of our algorithm is in the order of half a minute for a 600×400 image. This timing result is obtained on an Apple iBook with an 800 MHz G3 processor. While this does not afford interactive or real-time processing, we believe that the speed of our algorithm is sufficient to be practical.

Finally, Figure 4.8 shows the results after thresholding and removing small clusters of connected pixels. The glass surfaces in the scene with the giraffe statue present a difficult case, as many of the reflections are in focus, and are therefore classified as foreground.

4.4 Conclusions

Humans have no trouble detecting depth-of-field in images. Automatic segmentation using depth of field within a single image is possible by assessing local variance. The novelty of our work lies in the exploration of relative local variance between pairs of images which exhibit different depth of field. For such input, we have created a robust object segmentation algorithm. This allows an accurate relative assessment of local variance, which may be accomplished with image processing techniques that are similar to the sequence of processing found in the human visual system.

Starting with unquantized high dynamic range data, our algorithm applies non-linear response compression, followed by center-surround processing to detect areas of high local variance. Undecided regions are filled-in using a diffusion process. After that, only straightforward standard computer vision techniques are required to clean-up the results.

This work can be directly applied to the creation of alpha mattes. In addition, whenever an object needs to be segmented from its background, it can be photographed twice. The subsequent analysis is automatic. In our opinion the extra work required to take a second photograph is easily offset by the fact that no user interaction is required during alpha extraction.

The alpha mattes that we obtain from this algorithm are not accurate enough to be directly used as input into the image-based material editing methods. We have developed another algorithm that fine-tunes our matte estimate to generate a matte that is suitable as input into the material editing framework. We present the second algorithm in the next chapter.

Figure 4.1: *Input to our algorithm. The top image is focused on the object, whereas the bottom image is focused on the background.*

Figure 4.2: Comparison of the logarithm and sigmoid operators.

Figure 4.3: Plots of the variability computed for different scales. The numerator of (4.1) is used in the left plot, whereas the full division of (4.1) is plotted on the right. To construct V' , one would have sum over all scales in either plot.

Figure 4.4: Plots of the variability computed for different scales. The numerator of (4.1) is used in the left plot, whereas the full division of (4.1) is plotted on the right. To construct V' , one would have sum over all scales in either plot. These plots are for the middle scan line of the two images shown in Figure 4.1.

Figure 4.5: Signal V' for a single scale. The top image shows the difference between foreground and background image without the division, and the bottom images shows the same with the division in (4.1). Black encodes negative values, white encodes positive values and gray indicates zero, i.e. no difference between foreground and background.

Figure 4.6: *Result after running the diffusion algorithm for 200 iterations, i.e. B^{200} . Foreground and background regions are white and black, and undecided regions are encoded as gray. The top image shows the result for V' without the division, and the bottom image is for V' with the division. In either case, a single scale was used ($n = 1$).*

Figure 4.7: *From left to right: foreground image, background image, results after applying sigmoidal compression, center-surround processing and filling-in.*

Figure 4.8: *Thresholded results, with minimal clean-up applied. Luminance and color cues could be applied to further improve the results. The full division in (4.1) was used. The input images are depicted in Figures 4.1 and 4.7.*

CHAPTER 5

ALPHA MATTE GENERATION REVISITED

The goal of digital matting approaches, including the one presented in the previous chapter, is to enable the compositing of an object from an image, into a second image. An object may be composited into a new image if its color, as well as opacity at every pixel is known. Opacity for each pixel is defined as a value between 0 and 1, where 0 specifies that the object is completely transparent (or absent) at the pixel, and 1 indicates that the object is entirely opaque at the pixel. Once these values have been computed, the object may be placed in the new image by using the compositing equation:

$$I_{i,c} = \alpha_i F_{i,c} + (1 - \alpha_i) B_{i,c} \quad (5.1)$$

Here, F represents the computed color of the object, α represents its computed opacity, and B is the second image to which the object is being added. When this equation is evaluated for all pixels i , and for all channels c , the result is a new image I , in which the object has been placed into the new image.

All digital matting techniques can be explained in terms of finding the variables on the right hand side of the above equation (α_i , $F_{i,c}$, $B_{i,c}$), given $I_{i,c}$, which is the original image of the object. Thus, for each pixel, there are 7 unknown variables, and only 3 known values. The problem is therefore underconstrained, and a difficult one to address.

Many techniques have been proposed to estimate the unknown variables. A simplifying assumption that may be made by these techniques, is that the object which is to be composited into another image, does not have fuzzy or blurred boundaries. This implies that the alpha matte that will be computed for the object will be composed entirely of 0s and 1s, and will not have any fractional values, (that is, will be a binary matte). Given this assumption, the problem is reduced to

finding the value of a single variable per pixel, α_i . Once the binary matte has been created, finding the color of each pixel in the object is trivial: it is simply the corresponding image color.

Most techniques that compute binary mattes rely on user-specified alpha values for some of the pixels [BJ01, LST04, RKB04]. The rest of the pixels are segmented into regions of 0s and 1s on the basis of these given alpha values. Other techniques may rely on a different kind of input from the user. For instance, our approach in the previous chapter requires a second, differently focused image as input.

While many objects in images have sharp, opaque boundaries, others such as hair, fur, and glass do not. Objects that were in motion at the time of image capture also tend to have blurred boundaries. Ideal alpha mattes for such objects have fractional alpha values in some regions, and can therefore not be represented well by binary maps. In such cases, other techniques are required to compute the matte and color for the object. A variety of such techniques have been proposed to address this problem [CCS01, SJT04, MMP05, LLY06]. These rely on some initial classification by the user as well, and this classification is typically in the form of a trimap.

By computing values for all of the 7 unknown variables (α_i , $F_{i,c}$, $B_{i,c}$), these techniques are capable of generating reasonably accurate alpha mattes and color for most objects that exhibit transparency. There is one class of objects, however, that even these techniques are unable to generate satisfactory mattes for. This class includes all objects that are selectively transparent, that is, allow light of only certain wavelengths to pass through. For instance, all colored transparent objects are selectively transparent. Wavelengths that are perceived as red by the human eye, will pass through a red glass, while other wavelengths will not pass through. This implies that the object is transparent ($\alpha = 0$) for certain colors (red in this case), and opaque ($\alpha = 1$) for others. The standard form of the compositing equation (Equation 5.1) does not allow multiple values of α for the same pixel, and therefore it needs to be re-evaluated to allow reasonable compositing of

such objects. We propose a revised form of the compositing equation:

$$I_{i,c} = \alpha_{i,c} F_{i,c} + (1 - \alpha_{i,c}) B_{i,c} \quad (5.2)$$

Here, each channel has its own alpha value. Therefore, for the red glass, α_R may be 1, while α_G and α_B will be 0. This equation incorporates scenes which include selectively transparent objects, as well as those that were represented well by Equation 5.1. Note that for objects that are not selectively transparent, α_R , α_G , and α_B will all have the same value. This form of the composite equation has 9 unknown variables that need to be computed, as 3 alpha values per pixel need to be computed now.

In this chapter, we present an algorithm to compute all 9 unknown variables per pixel ($\alpha_{i,c}$, $F_{i,c}$, $B_{i,c}$) in the modified composite equation. We are thus able to successfully composite objects such as colored glass into new images. Along with the original image, our algorithm takes a trimap as input. We have generated our trimaps automatically using the algorithm described in the previous chapter. The remainder of this chapter is organised as follows: We discuss previous work in Section 5.1, followed by the details of our algorithm in Section 5.2. We then present results and conclusions in Section 5.3.

5.1 Previous Work

Several techniques to find the opacity and color of an object require the image to be captured in a controlled environment. Blue screen matting, which was among the first to be used, is an example of such techniques. This method requires a constant colored background, so that $B_{i,c}$ is known in the compositing equation [SB96]. The requirement reduces the number of unknowns to 4, and adding simple, ad hoc constraints can make the problem tractable. Apart from the inconvenient requirement of a specific background, this technique can fail on fairly simple images, and needs

an expert to tune its parameters [CCS01]. Another technique that relies on a constant background was proposed by Mishima [Mis93].

In certain circumstances, if the (possibly varying) background is known, then an alpha matte may be generated by applying an arbitrary threshold on the difference between the pixel values when the object is present, and when it is not. Applying a threshold results in the jagged appearance of the derived mattes. This artifact may be reduced by blurring, but errors in the matte are not removed.

Natural image matting techniques generate mattes of objects that are present in relatively arbitrary images. Several of these generate binary mattes [BJ01, LST04, RKB04]. These approaches translate simple user specified constraints into the min-cut problem. Solving this gives a reasonable segmentation of the image into foreground and background. Among these, while Rother et al do find fractional values for pixels in a thin strip at the border of the object, they cannot have fractional alpha values in large portions of the object.

Recently, Levin et al [LLY06] have proposed a method of obtaining a quadratic cost function in alpha. This enables them to solve for the unknown alpha variables by solving a sparse set of equations. While their method generates solutions in very little time, they do not bound the values that their variables may take, and often have alpha mattes with values outside the range $[0,1]$.

Other techniques that do not make the assumption of the matte being binary, often involve iterative optimization of cost functions [CCS01, SJT04, MMP05]. These techniques typically add constraints to make the problem over-constrained. For example, they may assume that the unknown variables vary smoothly across the image. While these techniques are capable of producing very realistic results for many images, they lack the ability to produce alpha values for selectively transparent objects, as mentioned earlier.

Furthermore, all current approaches¹, attempt to solve the composite equation from standard images, we propose the use of HDR images for alpha matte estimation. We note that the composite equation is physically invalid for color values that have undergone transformation under the radiometric response functions of most commercially available cameras. For the following radiance composite equation of a single channel,

$$I = \alpha F + (1 - \alpha)B, \quad (5.3)$$

I is the radiance value that enters the camera from the scene. If I is transformed by f (where f is the camera's response function), then $f(I) = f(\alpha F + (1 - \alpha)B) \neq \alpha f(F) + (1 - \alpha)f(B)$ when f is nonlinear. Thus, if intensity values available to the algorithm have undergone transformation then the traditional composite equation is invalid. In contrast, if we use high dynamic range images in which the camera's response function has been accounted for, the composite equation becomes an accurate model of image formation, potentially leading to better estimation.

5.2 Algorithm

The objective of the algorithm is to take as input the color information contained in the image along with some prespecified values of the unknowns, and produce as output the foreground F , background B and alpha values for all pixels. Since there are a greater number of unknowns than knowns it is useful at the outset to outline the constraints at our disposal, which can aid in making the estimation process tractable:

- **Composite Equation:** Primarily, estimation of F , B , and α must satisfy the composite equation. Therefore, if these estimated values are placed in the composite equation, they should generate the known image values. In a given image I of dimensions X and Y , the

¹that we are aware of.

number of known image values are $3XY$.

- **Regularization:** An assumption of regularity can also be used to constrain estimation. The pixel values typically vary smoothly so that values within a small neighborhood are similar.
- **Constrained Estimation:** Natural bounds exist for each of the unknown variables, which can be used to constrain the solution. As mentioned earlier, all variables F , B , and α , are non-negative, and α has a maximum value of 1. In addition, the user supplied information also acts as strong constraints on the final estimate.

In this section, we describe how these constraints can be folded into the estimation process in an intuitive and principled manner. Emphasis is placed on formulating the problem *linearly* since well studied routines exist for solving such problem types.

5.2.1 Composite equation

The composite equation is traditionally written in its convex form as,

$$I = \alpha F + (1 - \alpha)B. \quad (5.4)$$

The equation is bilinear in the unknowns, but through judicious change in variables the estimation can be made linear and therefore easier to compute. Equation 5.4 can be expanded into

$$I = \alpha F - \alpha B + B. \quad (5.5)$$

Instead of optimizing over α , F and B for each pixel, for each channel, we replace the variables and compute the interaction terms αF and αB and the background B . Over all n pixels, a linear

system of equations, $\mathbf{A}_c \mathbf{x} = \mathbf{b}_c$, can be constructed and solved directly from image information. Explicitly this system is,

$$\begin{bmatrix} 1 & -1 & 1 & & & \\ & & & \ddots & & \\ & & & & 1 & -1 & 1 \end{bmatrix} \begin{bmatrix} \alpha_{1,c} F_{1,c} \\ \alpha B_{1,c} \\ B_{1,c} \\ \vdots \\ \alpha_{n,c} F_{n,c} \\ \alpha B_{n,c} \\ B_{n,c} \end{bmatrix} = \begin{bmatrix} I_{1,c} \\ \vdots \\ I_{n,c} \end{bmatrix}.$$

which finds the values of the unknowns that minimize Equation 5.6.

$$J_c(B, \alpha F, \alpha B) = \sum_{i,c} (I_{i,c} - \alpha_{i,c} F_{i,c} + \alpha_{i,c} B_{i,c} - B_{i,c})^2. \quad (5.6)$$

From the estimated values of αF , αB and B , the values of α and F can then be uniquely recovered,

$$\alpha = \frac{\alpha B}{B} \quad (5.7)$$

$$F = \frac{\alpha F}{\alpha}. \quad (5.8)$$

Thus, representing our unknowns in this manner allows us to evaluate a cost function that is linear with respect to the new variables. Of course, at this point, the linear system will be underconstrained since the composite equation does not uniquely determine values of the unknowns. In order to make the system solvable, we introduce the remaining constraints into the system.

5.2.2 Regularization

To regularize the solution we enforce a smoothness constraint on neighboring values of the unknowns. Thus in addition to J_c , the composite cost function, we wish to minimize J_s , the smoothness cost function, where

$$J_s(B, \alpha F, \alpha B) = \sum_{i,c} \sum_{j \in N(i)} [(B_i - B_j)^2 + (\alpha B_i - \alpha B_j)^2 + (\alpha F_i - \alpha F_j)^2]$$

where $N(i)$ is the neighborhood of i , e.g. a 4-neighborhood or an 8-neighborhood (as used in this work), and consists of all pixels adjacent to pixel i . This time a homogeneous system of equations can be constructed enforce this smoothness constraint, $\mathbf{A}_s \mathbf{x} = \mathbf{0}$, where each row in $\mathbf{A}_s$ enforces one smoothness constraint between unknowns. For instance, to enforce the smoothness constraint between B_i and B_j , the element in that row of $\mathbf{A}_s$ corresponding to B_i would be set to 1, while the element corresponding to B_j would be set to -1. Similarly, a row is added corresponding to each smoothness constraint, for each pixel, for every channel.

5.2.3 Constrained estimation

So far, we have incorporated the composite equation and smoothness constraints into the system. The prespecified values on the variables have not been taken into account yet and neither have the bounds on the unknowns. These are manifested as linear constraints on the linear system of equations described in the previous subsection. Non-negativity constraints on all F values and B values, along with the lower and upper bounds of 0 and 1 on all α s can be defined using matrix inequalities. Lower bounds are directly applicable since αF , αB and B must all be greater than or equal to zero. A maximum value of 1 for α implies that $\alpha B \leq B$ since if α is 1, $\alpha B = B$, and for

all other values of α , $\alpha B < B$. Thus for a single channel of a pixel,

$$\begin{bmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & -1 & 1 & \end{bmatrix} \begin{bmatrix} \alpha_{i,c} F_{i,c} \\ \alpha B_{i,c} \\ B_{1,c} \end{bmatrix} \geq \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

or simply $\mathbf{D}\mathbf{x} \geq \mathbf{0}$.

5.2.4 Minimizing the objective function

The final objective function can now be defined, satisfying the composite equation and smoothness relationships, while enforcing the bounds on the unknowns and the constraints provided by the prespecified input. The final objective function over the values of αB , αF , and B , is $J = aJ_c + bJ_s$ or in expanded form,

$$\begin{aligned} J(B, \alpha F, \alpha B) = & \sum_{i \in I, c \in C} [a(I_{i,c} - \alpha_{i,c} F_{i,c} + \alpha_{i,c} B_{i,c} - B_{i,c})^2] \\ & + b \sum_{j \in N(i)} [(B_{i,c} - B_{j,c})^2 + (\alpha B_{i,c} - \alpha B_{j,c})^2 + (\alpha F_{i,c} - \alpha F_{j,c})^2] \end{aligned} \quad (5.9)$$

subject to the linear equality and inequality constraints of Section 5.2.3. The first term corresponds to the composite equation and the second term the smoothness constraint, ensuring neighboring values have similar estimates. The weights a and b determine the relative importance of the composite equation and the smoothness constraints. Given this objective function, a linear system of equations can be created, such that its solution will minimize this function, and therefore solve the alpha matte extraction problem. We need to construct matrices in the following system of equations:

$$\mathbf{A}\mathbf{x} = \mathbf{b} \quad (5.10)$$

where $\mathbf{x}$ is a vector of the unknown variables. This system is illustrated in Figure 5.1. The top part of $\mathbf{A}$, $\mathbf{A}_c$ contains data related to the composite equation. For an image with dimensions X and Y , the number of known values (and therefore equations in the system) are $3XY$. The bottom part of $\mathbf{A}$, $\mathbf{A}_s$, contains data related to the smoothness constraints. An example of a filled sparse matrix $\mathbf{A}$ is given in Figure 5.2. Solving a linear least squares problem subject to linear constraints is a standard quadratic programming problem [GMW81], and can be solved using a number of approaches. In particular, for problems, such as the one being considered in our work, where the constraints are a mixture of bounds and linear constraints, active set approaches are generally considered the method of choice. We employ the `lsqlin` function in Matlab, which applies an active set approach to solve the constrained system of equations.

Figure 5.1: *System of Equations for Alpha Matte Extraction. This system does not yet incorporate the weights mentioned earlier. Note that $\mathbf{A}$ is a sparse matrix.*

To give a unique solution, the number of equations should exceed the number of unknown variables. The number of unknowns are $9XY$. The number of equations based on the composite

Figure 5.2: An example of a filled, sparse matrix A . Elements with values 0, -1, and 1, are shown in gray, black, and white respectively.

equation are $3XY$, while the number of equations based on smoothness constraints are $9(4XY - 3X - 3Y + 2)$, as computed in Figure 5.3. The above system is overconstrained, and therefore gives

Figure 5.3: Smoothness constraints are shown in green in this figure, for a 3×3 image. The number of constraints shown here are multiplied by 9 as they are present between each of the variables αF , αB , and B , in the 3 channels.

a unique solution, so long as the number of constraints are greater than the number of unknowns

$$9XY \leq 3XY - 9(4XY - 3X - 3Y + 2) \quad (5.11)$$

After some manipulation of this expression, it can be seen that for a square image (i.e. $X=Y$), the system is underconstrained only if both X and Y are equal to 1.

Thus far, the contribution of weights in the cost function has not been considered. The weighted linear system can be expressed as:

$$W\mathbf{Ax} = W\mathbf{b} \quad (5.12)$$

where W is a square, diagonal matrix. Each diagonal element contains a weight for the corresponding equation in the system. In its simplest form, W consists of weights a and b from Equation 5.10. All the composite equations are given weight a while the smoothness constraint equations are given weight b .

These weights are unable to incorporate situations where smoothness constraints between certain pixels might be more important than others. For instance, we only want there to be smoothness in computed variables across pixels, if those pixels have similar color. For instance, we do not want our variables to be smooth across sharp edges, as the edge might be due to change in the alpha matte. We might want to have our variables change across this edge as well. We therefore assign individual weights to each smoothness constraint equation in the system, based on the color difference between the two pixels. These weights are computed using Equation 4.2 from Chapter 4. Once computed, they are multiplied with the initial weights (comprising of a and b). The combined weights give our system the ability to abruptly change variables across space, a feature that is often lacking in matte extraction techniques.

5.3 Results

While the foreground and background images (mentioned in the previous chapter) contain differently colored objects with a lot of texture and well-defined boundaries, the algorithm discussed in

the last chapter is sufficient to generate an accurate binary matte allowing objects to be placed in other images. Figure 5.4 shows this ideal case. The two input images were generated by adding noise to both foreground and background, and then blurring them for the background and foreground images respectively. Clearly, the binary map extracted using the technique presented in the last chapter is sufficient here, and no post processing is required to improve the result.

Figure 5.4: *The top row shows two differently focused images of the same object. Focus information is used to obtain images in the second row. The left image is the output of the algorithm before diffusion, and right image is the output after only 60 iterations of diffusion. A threshold may easily be applied to this image to generate a binary matte, as shown in the left image in the third row. The right image shows how the object may now be placed into a new environment by using the matte.*

Most scenes do not exhibit so much texture, and therefore the initial matte estimate may not be accurate enough, even with diffusion, to produce an accurate matte. Take the images in Figure 5.5,

for instance. The input images were generated in a manner similar to those in Figure 5.4, but the amount of noise was reduced. The matte that was used to place the object onto the foreground and background images was blurred as well. This implies that the computed alpha matte should have fractional values - another feature that is not handled by the technique proposed in the previous chapter. The resulting matte is not accurate enough to allow a realistic placement of the object into a new background.

Figure 5.5: *The top row shows two differently focused images of the same object. Focus information is used to obtain images in the second row. The left image is the output of the algorithm before diffusion, and right image is the output after 260 iterations of diffusion. A threshold is difficult to apply to this data to generate an accurate binary matte. An attempt is shown in the left image in the third row. The right image shows discrepancies when the object is placed into a new environment by using the matte.*

In such cases, a trimap may be generated instead of a binary matte. Our alpha matte extraction technique can be used to convert this trimap to an accurate alpha matte. We generated a trimap from the binary matte shown in Figure 5.5, and show it along with our results in Figure 5.6. The results obtained by using the technique proposed by Levin et al [LLY06] had to be modified before use. As there are no bounds for their solution, the matte that was generated for this particular figure had minimum and maximum values of -11.861 and 8.221 respectively. Values above 1 and below 0 had to be clamped to 1 and 0 respectively, before the image at the bottom right of Figure 5.6 could be generated.

As the dimensionality of our problem is very large (9 dimensions per pixel), the size of the image that we can currently compute a matte for, is restricted to a 10×10 image (900 dimensions). For even slightly larger images, the optimization function runs out of memory on our machine, which has 1GB of RAM. As a result, we are currently restricted to working with trimaps with a very small unknown region, even though the algorithm could theoretically work with scribbles as user input. An input image is divided into 10×10 regions, for which the mattes are computed separately. Hardware restrictions also imply that extensive experimentation and results remain outside our scope for the time being.

In this chapter, we have presented a novel way of solving the alpha matting problem. Together with the initial trimap, which can be generated using the algorithm in the previous chapter, our set up can automatically generate accurate alpha mattes with fractional values.

Figure 5.6: *Using the two differently focused images shown in the previous figure, a trimap may be generated using the method discussed in the last chapter. This trimap is the left image in the top row. The image on the right is the alpha matte that is generated using this trimap. The middle row shows that the object may now be placed more realistically into a new scene, or the matte may be used to create an image that is focused correctly both in the foreground and the background. The last row shows a results obtained with the method proposed by Levin et al.*

CHAPTER 6

MATERIAL EDITING

In this chapter, we detail the image-based material editing techniques that we have developed. Specifically, we discuss algorithms to change the material of a specified object in the image to a transparent, a translucent, or a textured material, or any material that may be represented by a bidirectional reflectance distribution function. To do this in a physically correct way would involve accurately estimating the lighting, and 3D geometry of the object from the image; a task that not even state-of-the-art computer vision can achieve to a high degree of precision. Instead of a physically accurate solution, our goal is to generate material transforms that appear plausible to the observer.

Image-based material editing, as proposed in this chapter, incorporates several components for which solutions already exist. We will discuss prior work in each section where relevant. However, it should be noted that our choices for particular solutions are to maximize the ensemble result (i.e., produce a plausible edit to an object’s material appearance), rather than in all cases use state-of-the-art components.

In particular, we emphasize that our proposed solution frequently incorporates simpler algorithms, while maintaining a compelling ensemble result. A user of our work may substitute more advanced solutions at will. In this chapter, however, we show that several algorithms each with minimal complexity, may together produce convincing image edits.

6.1 Problem Definition

For the purpose of manipulating images, insight into image formation may be gained from studying the rendering equation [Kaj86]:

$$L(x, y) = L_e + \int_{\Omega} f_r(\Theta_i, \Theta_o) L_i(\Theta_i) \cos \Theta_i d\omega_i \quad (6.1)$$

For non-emitting objects, we can set $L_e = 0$. Thus, the light L focused onto a pixel (x, y) is formed by taking the corresponding nearest surface point in the scene, and constructing a hemisphere Ω around the surface normal associated with this point. Then, light from all directions Θ_i covered by Ω is reflected towards the pixel by an amount governed by the reflectance function f_r .

We are interested in manipulating pixel values $L(x, y)$ such that objects appear to be made of different materials. Thus, we aim to take an existing image L and transform it into a different image L' such that:

$$L'(x, y) = \int_{\Omega} f'_r(\Theta_i, \Theta_o) L_i(\Theta_i) \cos \Theta_i d\omega_i \quad (6.2)$$

However, in the above two equations there are several unknowns: the orientation of the hemisphere Ω for each point on the object, the reflectance function f_r , and the incident lighting L_i are not available. All we know are the pixel values L and the desired reflectance function f_r . Thus, computing L' is not possible without making further assumptions and simplifications. The key point of this work is that we will show how to simplify the problem such that although the physics of light is not strictly followed, the results remain plausible. We achieve this by exploiting limitations of human vision in a consistent manner.

First, for each pixel belonging to the object we wish to alter, we require an approximation of Ω , or equivalently the surface normal $\mathbf{n}$. Details of both the problem and our solution are presented in Section 6.3.

Let's assume we have an approximation to Ω and we have called it Ω' . To replace an object's reflectance properties, for each pixel we would need to know the incident lighting L_i coming from all directions covered by Ω' . It would be possible to capture an environment map using multiple photographs (perhaps using fish-eye lenses), or by employing a reflecting sphere, as used in image-based lighting applications [Deb02]. With a single photograph, we have accurate background lighting for some directions, namely those visible in the image, but not others.

It is possible to approximate the incident lighting for all remaining directions on the basis of the partial knowledge we have by means of the background in the input image. Dependent on the nature of the desired reflectance function f'_r , the approximation needs to be more or less precise. In particular, if f'_r is relatively diffuse, the approximation can be relatively coarse [RH01]. In addition, human perception is such that relatively coarse approximations are sometimes tolerated. For instance, Ostrovsky et al [OCS05] have found that deliberate inconsistencies in the direction of illumination are very difficult to detect. Other experiments have confirmed that humans cannot in general estimate illumination reliably [nP05a, nP05b]. This implies that inaccuracies in illumination are likely to go unnoticed — a feature we actively exploit in our work.

The unknown values of L_i can be classified into two groups: values that are outside the image, for instance behind the photographer, and values coming from behind the object we wish to alter. For these two cases we adopt different solutions because the number of missing values is different in each case, and therefore the level of approximation can be different. In either case, however, we end up with a set of incident lighting values L'_i , as discussed further in Section 6.4.

After computing approximate values for the object's surface shape as well as the incident lighting, a new reflectance function f'_r can be chosen and a new image may be rendered:

$$L'(x, y) = \int_{\Omega'} f'_r(\Theta_i, \Theta_o) L'_i(\Theta_i) \cos \Theta_i d\omega_i \quad (6.3)$$

Finally, the types of material changes we are currently interested in involve retexturing of objects, increasing or decreasing specularities, adding transparency and translucency, as well as replacing an object's BRDF. Our work on transforming materials on the basis of a single input image is novel, with the exception of object texturing [FH04].

6.2 Overview

The general approach we adopt is first to acquire a high dynamic range (HDR) photograph of the object of interest by concatenating multiple exposures [DM97b]. Most of the techniques we present also work to some extent with standard (low dynamic range) images. However, HDR input yields more robust results. The gradients used to estimate Ω are smoother if derived from HDR images; highlight identification is also facilitated. Importantly, estimates of L_i can be directly used to light the object, i.e. the background pixels are suitable for techniques akin to image-based lighting [Deb02, RWP05b].

Second, we create an alpha matte to separate the object from the background. Sophisticated techniques have recently become available to assist the user in this task [LST04, RKB04], or go towards automatic alpha matte extraction [MMP05, RK05]. We have outlined our own technique in Chapters 4 and 5.

The pixels forming the object will be altered to fit the object with a new material. For each of these pixels, a value of Ω' will first be estimated. The remainder of the pixels will be used to estimate values of L'_i . By aligning our image processing with the object's boundaries, the illusion of a different material may be created.

In the remainder of this chapter, we first discuss the approaches we have found useful for estimating Ω' and L_i (Sections 6.3 and 6.4), followed by the various material transforms we have achieved (Section 6.5). We discuss our results in Section 6.6.

6.3 Object Shape (Ω')

A hemisphere of directions Ω' can be estimated from pixels belonging to the object by first estimating the distance between the viewer and the object, i.e. the depth. Gradients in depth are then used to compute a surface normal $\mathbf{n}$ for each pixel. Each of these steps are discussed in the following subsections.

6.3.1 Depth recovery

Computing a depth map from a single image is the classic shape-from-shading problem. Unfortunately, as this problem is severely under-constrained, good solutions for arbitrary images do not exist [ZTC99]. However, specific solutions have been developed for specific cases, while manual user guidance may sometimes aid depth recovery as well [Kan98, IMT99, OCD01].

Experimental evidence suggests that human perception of object shape may be affected by the direction of the light source in the scene. For example, shapes are seen to be elongated in the direction of the light source [KnC96]. Also, lower luminance values are frequently interpreted as more distant surface locations [LB00]. Under diffuse lighting, this is a reasonable default assumption made by shape-from-shading algorithms [LZ94], and one we follow as well.

Reflections are distorted as a function of surface curvature, and in particular the second derivatives of surface heights determine the appearance of reflections [FTA04]. As second derivatives

are unaffected by affine transformations, global inaccuracies in depth recovery (such as shear) would not necessarily affect the perception of reflections on objects. Therefore, the consistency between neighboring positions on an object is more important than the absolute value of the depth-map to produce a reasonable impression of object shape (as has been suggested by perception researchers [TR89]).

We therefore aim to derive a locally-consistent depth-map from the object’s luminance distribution, with higher luminance values specifying the parts of the object closer to the observer (i.e. following the dark-is-deep paradigm). Our approach consists of several steps. First, we compute the luminance of each pixel [ITU90]:

$$L(x, y) = 0.213 R(x, y) + 0.715 G(x, y) + 0.072 B(x, y) \quad (6.4)$$

An initial crude depth map $d(x, y)$ may be computed by simply setting $d(x, y) = L(x, y)$. However, there are several problems with this approach. First, an object may be textured, leading to errors in the depth map. Second, the shape implied by the depth values may bulge in the direction of any light sources, as argued above. Third, highlights may cause significant distortions in the depth map — a problem addressed in Section 6.5.1.

The luminance variation due to shading is useful as an indicator of shape, whereas treating textured areas as valid depth information is only useful for special effects (such as making an object appear embossed). The discontinuities in luminance due to silhouettes ought to be preserved.

While disentangling luminance variations due to several different processes is once more an under-constrained problem, in practice shading, textures and patterns often occur at different frequencies and/or luminance levels. We therefore require a filter that is tunable to spatial scales as

Figure 6.1: *Bilateral filtering applied to the object. The kernel size in the intensity domain was varied in this image sequence, while the spatial filter was kept constant.*

well as luminances. The bilateral filter directly fits these requirements [TM98]:

$$d(x, y) = \frac{\sum_u \sum_v b(x, y, u, v) L_s(x - u, y - v)}{\sum_u \sum_v b(x, y, u, v)} \quad (6.5)$$

$$b(x, y, u, v) = G_{\text{spatial}} \left(\sqrt{(x - u)^2 + (y - v)^2}, \sigma_{\text{spatial}} \right) \times \\ G_L (L_s(x - u, y - v) - L_s(x, y), \sigma_L) \quad (6.6)$$

The bilateral filter kernel b is composed of two Gaussian filter kernels, G_{spatial} and G_L with kernel widths of σ_{spatial} and σ_L , affording user control over the recovered object shape. For instance, σ_L determines how much texture remains visible in the depth map, as shown in Figure 6.1. For larger values of this parameter, small details are increasingly suppressed while the overall round shape is emphasized. Further, control over surface indentations is afforded by σ_{spatial} , as shown in Figure 6.2. Dependent on the specific shape of the object different parameter settings may thus yield satisfactory depth maps.

The bilateral filter is typically used in the logarithmic domain. While $\log L$ is a better behaved quantity than L , the range of values produced by taking the logarithm is unconstrained. A closer model of early human vision involves employing sigmoidal compression, which produces behavior similar to log luminance in the mid-range of values, but additionally has a range of output values that is strictly limited between 0 and 1:

$$L_s(x, y) = \frac{L^n(x, y)}{L^n(x, y) + \sigma^n} \quad (6.7)$$

Figure 6.2: *The value of $\sigma_{spatial}$ controls the smoothness of the depth-map. Here, its value is doubled from 1 to 2, 4 and 8.*

The exponent n is set to 0.73 by default, a value common in modeling for instance photoreceptor responses [NR66]. The semi-saturation constant determines which luminance values are mapped in near-logarithmic fashion. A common and sensible value is to chose σ equal to the log average luminance of the object's pixel values:

$$\sigma = \exp \left(\frac{1}{N} \sum_{x,y} \log (L(x,y)) \right) \quad (6.8)$$

Compressing the input L as above helps make the results more robust, and additionally facilitates the choice of values for σ_L .

By setting the depth of pixels not belonging to the object to zero, we allow the bilateral filter to smooth over the silhouette edges. This is in essence equivalent to 'shape inflation'. Since the contrast at the silhouettes (with the background) is increased by setting the background to zero, the filter will convolve values either side of the silhouette, dependent on the value of σ_L . This will show up as a larger depth for pixels near the silhouette, which is plausible for most non-flat objects

as it gives the impression of shape curvature at the surface boundaries. This feature is shared by models produced by level-set approaches [Wil98].

6.3.2 Gradients and surface normals

The recovered depth $d(x, y)$ can now be used to estimate local gradients and surface normals. However, due to the sigmoidal compression applied earlier, the range of values of d is strictly limited. This means that gradients near silhouette edges are too small. We therefore invert the sigmoidal compression function of (6.7) first:

$$d'(x, y) = \left(-\frac{\sigma^n d(x, y)}{d(x, y) - 1} \right)^{1/n} \quad (6.9)$$

In two dimensions, the gradient field is $\nabla(x, y)$, which can now be defined in terms of neighboring depth values:

$$\nabla(x, y) = (d'(x + 1, y) - d'(x, y), d'(x, y + 1) - d'(x, y)) \quad (6.10)$$

$$= (\nabla_x, \nabla_y) \quad (6.11)$$

As luminance values were used to derive these gradients, we cannot expect that the range of gradient values is directly commensurate with the shape of the object. In addition, there is no reason to believe that the conversion from luminance values to gradient values should be linear. We have empirically determined a non-linear spline function which reshapes the gradient field to boost small and large gradients, attenuating intermediate gradients, as shown in Figure 6.3. In addition the amount of reshaping can be adjusted by recursively applying this function (also shown in this figure). Typically, we apply this function between one and four times, dependent on the input data.

The resulting gradient field can then be used to compute 3D surface normals, or may be applied directly to warp textures, as shown in Section 6.5. We compute 3D surface normals using the

Figure 6.3: *A family of gradient reshaping functions.*

following gradient vectors:

$$\mathbf{g}_x = [1, 0, \nabla_x]^T \quad (6.12)$$

$$\mathbf{g}_y = [0, 1, \nabla_y]^T \quad (6.13)$$

After normalization of $\mathbf{g}_x$ and $\mathbf{g}_y$, the surface normal is given by the cross product:

$$\mathbf{n} = \frac{\mathbf{g}_x \times \mathbf{g}_y}{\|\mathbf{g}_x \times \mathbf{g}_y\|} \quad (6.14)$$

6.4 Incident Light (L_i)

Given an (approximated) representation of 3D shape, from (6.3) we see that a successful manipulation of an object's material requires knowledge of the incident lighting L_i ¹. Given a single photograph, all the background pixels together provide information for a subset of the directions we are interested in. For all other directions, namely those behind the object, and those outside the image, we will have to substitute an approximation L'_i .

¹In this and subsequent sections, we will retain the notation L_i when discussing incident lighting. However, it should be noted that while this symbol is commonly used to denote luminance values, we will use the same symbol to indicate RGB triplets wherever this does not lead to confusion.

Our material edits can be grouped into full 3D simulations, as well as 2D remapping functions. For the 3D simulations, we need a value L'_i for every possible direction. For the 2D remapping schemes, we may restrict ourselves to a set of directions that correspond to existing image pixels. However, in both cases an impression of what lies behind the object is desirable. Solutions for both problems are outlined in the following subsections.

6.4.1 Light from behind the object

To determine what light may come from behind the object, we may employ an image inpainting algorithm [BSC00, DCY03, SYJ05]. The input to such algorithms is typically the image itself as well as an alpha matte which defines the hole that needs to be replaced with a new background. Algorithms to create suitable alpha mattes are described in Chapters 4 and 5. The result is an image with the gap filled with plausible background pixels.

While the more advanced algorithms may yield more plausible results, after applying them to (6.3) the quality of the inpainting algorithm is largely masked due to further processing, contorting and warping. In the following, we demonstrate that for this particular application the inpainted region only needs to capture the correct image statistics, which is a weaker requirement than those met by full inpainting algorithms. This affords a much simpler algorithm, while still yielding satisfactory results.

An important observation is that for the background to be useful as a complex light source, its precise configuration is unimportant. However, the statistical properties of the inpainted pixels ought to be similar to the remainder of the environment². In particular the image's power spectral slope should be preserved [Rud97b, Sch98, TO03, RSA04], as well as its color composition.

²This is not generally the case for inpainting algorithms, but merely a consequence of our specific application.

Figure 6.4: *Removing the elephant from the original image (left) is shown in the middle. The masking effect of further processing is demonstrated with the simulation of transparency (right).*

The most straightforward approach that would achieve these goals is to copy large blocks of pixels from the background into the gap formed by removing the object. This approach would minimize the introduction of new edges, and thus preserve the aforementioned image statistics. To minimize the introduction of artifacts along the gap's boundary, we copy pixels both from the left and from the right parts of the image, and blend them according to how much they have been displaced.

To begin, we find the leftmost and rightmost extent of the gap's boundary, and call the corresponding x -coordinates $x_{\min}$ and $x_{\max}$. The value assigned to all pixels inside the gap is then L'_i :

$$L'_i(x, y) = w_1 L(2x_{\min} - x, y) + w_2 L(2x_{\max} - x, y) \quad (6.15)$$

$$w_1 = \frac{x - x_{\min}}{x_{\max} - x_{\min}} \quad (6.16)$$

$$w_2 = \frac{x_{\max} - x}{x_{\max} - x_{\min}} \quad (6.17)$$

In case the x -coordinates index outside the image's boundaries, we use wrap-around within the regions $[0, x_{\min}]$ and $[x_{\max}, X_{\max})$ (with $X_{\max}$ being the image's horizontal resolution). The effect of this operation is given in Figure 6.4, showing the original image (left), the result (middle), and a simulation of transparency to demonstrate the masking effect of further processing.

6.4.2 Reconstructing a 3D environment

The image with the filled-in background can be used directly in 2D mapping operations as outlined in Section 6.5. For mapping a BRDF onto an object, it is necessary to reconstruct a full spherical 3D environment. In traditional image-based lighting approaches, photographs of mirrored spheres or the output of scanning panoramic cameras are mapped to a spherical environment [Deb02]. In our case, an approximate mapping can be constructed between the hole-filled photograph and the desired environment.

The plane of the image in three-dimensional space may be thought of as partitioning the environment into two half spaces. Our environment map is created by extruding a circular part of the background image to form a hemisphere behind the image plane, and again to form a hemisphere in front of the plane, as illustrated in Figure 6.5. This leads to the following computation of direction vectors for given pixel coordinates.

The center of the circle $[x_c \ y_c \ 0]^T$ is given by the middle of the image, and this circle is made as large as possible, i.e. the radius R is $\min(x_c, y_c)$. Next, we construct for each pixel within this circle, a normalized coordinate pair $[x_n \ y_n]$:

$$[x_n, y_n]^T = \left[\frac{x - x_c}{R}, \frac{y - y_c}{R} \right]^T \quad (6.18)$$

A pixel $[x_n \ y_n]^T$ can now be mapped to a direction vector $\mathbf{d}$ with:

$$\mathbf{d} = \left[x_n, y_n, 1 - \sqrt{(x_n)^2 + (y_n)^2} \right]^T \quad (6.19)$$

These direction vectors are equivalent to the incoming and outgoing directions Θ_i and Θ_o in (6.3). This means that for every desired direction, a corresponding index into the hole-filled image can be computed for the purpose of solving (6.3).

While this mapping distorts the environment leading to physically inaccurate lighting, the distortion is locally consistent, and therefore largely unobjectionable to the human visual system. As

Figure 6.5: *The background image with the object removed is used to create an HDR environment map. First a circle is cut from the middle of the image. This is then placed in the image plane, and then extruded to become half the environment. The image is also pulled out in the opposite direction to form the other half of the environment.*

such, we have recovered both plausible surface normals for the object itself, as well as plausible lighting to be used to relight the object. We therefore have all the preliminaries necessary to perform material transformations.

6.5 Material Transforms

Editing an object's material properties can take several different forms, depending on the desired target material. For instance, increasing or decreasing the gloss on an object would require only a straightforward luminance remapping, whereas replacing the BRDF of an object would mean a full evaluation of (6.3). Retexturing an object as well as creating transparent and translucent objects require techniques that sit somewhere in between these extremes in terms of computational com-

Figure 6.6: *Test images used in this chapter.*

plexity. Hence, we discuss each of these material transforms separately in the following sections. Results are shown throughout this section, using input images depicted in Figure 6.6.

6.5.1 Glossy surfaces

For objects that already exhibit some amount of glossiness, an exponential remapping of luminance values may increase or decrease the perception of gloss [FJB04]. To increase specularity, we have found the following mapping to yield acceptable results:

$$L' = \begin{cases} L_{\min} + (L_{\max} - L_{\min}) \left(\alpha \frac{L - L_{\min}}{L_{\max} - L_{\min}} \right)^{\beta} & L > L_{\min} \\ L & L \leq L_{\min} \end{cases} \quad (6.20)$$

Figure 6.7: *The luminance values of the statue (left) were remapped to create a more specular (middle) and more diffuse (right) image.*

where $L_{\max}$ is the maximum luminance of all object pixels, and $L_{\min}$ is the minimum luminance that can still be considered part of the highlight. This is a user parameter, which may alternatively be estimated with the highlight detection algorithm presented in Section 6.5.4. To steer the amount of specularity, the user parameters d and p are available. These are typically set to $p = 20$ and $d = 0.05$.

To make an object appear more diffuse, we use the following remapping:

$$L' = \begin{cases} L_{\text{hmax}} + (L_{\min} - L_{\text{hmax}}) \left(\alpha \frac{L - L_{\text{hmax}}}{L_{\max} - L_{\text{hmax}}} \right)^{\beta} & L > L_{\text{hmax}} \\ L & L \leq L_{\text{hmax}} \end{cases} \quad (6.21)$$

with p typically set to 0.05. The value of L_{hmax} is computed by taking the luminance associated with the peak of the histogram.

Examples of both remappings are shown in Figure 6.7. For high-gloss objects, the latter remapping may also be useful as a pre-processing step before recovering Ω' (Section 6.3). This would permit the shape recovery to be less sensitive to highlights.

6.5.2 BRDF replacement

A more general approach to material replacement in images is to carry out a full simulation based on Equation (6.3). With estimates of surface normals $\mathbf{n}$ available for each point on the object, and having constructed a mapping from a rectangular image to a full 3D environment to provide values for L'_i , we can evaluate (6.3) directly. While our approach follows common practice in image-based lighting [Deb02], the remainder of this section is devoted to detailing our specific implementation.

We assume that the world coordinate system is aligned with the image such that the image plane is formed by the x and y axis, and the z axis is pointing out of the image towards the viewer. The direction vectors $\mathbf{d}$ and surface normals $\mathbf{n}$ are given with respect to this coordinate system, and so is the view vector, which we assume to be $\mathbf{v} = [0 \ 0 \ 1]^T$.

To sample the environment for each point on the surface of the object, a fast and efficient sampling scheme is required. One of the most efficient ways to generate sampling points for our environment is afforded by Ostromoukhov's importance sampling technique [ODJ04]. Using this algorithm as a preprocess, we typically find around 400 or fewer sampling coordinates $[x_n \ y_n]^T$ in our background image, leading to an equal number of values of L'_i . Each of these sampling points has an associated 3D direction vector $\mathbf{d}$, computed with Equation (6.19). For each point on the object, we sample only those importance samples for which $\mathbf{d} \cdot \mathbf{n} > 0$, i.e. those samples that lie within the hemisphere Ω' .

A common convention is to evaluate BRDFs in a coordinate system which is aligned with the surface normal. This is achieved by rotating the view vector over an angle $\alpha = \mathbf{n} \cdot \mathbf{v}$ around an axis $\mathbf{a} = \mathbf{n} \times \mathbf{v}$. For each sample point in the environment we wish to evaluate, its associated direction vector $\mathbf{d}$ is rotated over the same angle, giving $\mathbf{d}_\alpha$. Quaternion rotation should be used to avoid gimbal lock [HFK94].

Figure 6.8: *Example of mapping a BRDF representing nickel onto a SIGGRAPH mug.*

Although our application will work with any BRDF model, we have chosen Matusik et al’s measured BRDFs [MPB03] since these were captured in high dynamic range, and are therefore directly compatible with our system. Thus, the equation we evaluate to fit an object with a new material is:

$$L'(x, y) = \sum_{\{\mathbf{d} | \mathbf{d} \cdot \mathbf{n} > 0\}} f_{\text{Matusik}}(\mathbf{d}_\alpha, \mathbf{v}_\alpha) L'_i(\mathbf{d}) \mathbf{d} \cdot \mathbf{n} \quad (6.22)$$

Some results of this approach are shown in Figures 6.8 and 6.9.

6.5.3 Retexturing

The machinery outlined in Section 6.3 is sufficient to allow objects to be retextured. In particular, the gradient field ∇ can be used to estimate a warping of an arbitrary texture $vectT$, which can then be applied to the object. To steer the amount of apparent curvature, we introduce a linear scale

Figure 6.9: *Examples of mapping arbitrary BRDFs onto an object.*

factor s , and compute texture indices $[t_x, t_y]^T$ as follows:

$$t_x = x + s \nabla_x \quad (6.23)$$

$$t_y = y + s \nabla_y \quad (6.24)$$

The indices are taken modulo the texture size and are used to compute the new pixel colors of the object. Assuming a pixel $[x \ y]$ belonging to the object has an RGB color triplet denoted $\mathbf{C}(x, y)$, the new color $\mathbf{C}'(x, y)$ can be derived from its original color $\mathbf{C}(x, y)$ and the texture look-up value $\mathbf{T}(t_x, t_y)$. For extra flexibility, we follow Smith and Blinn [SB96] and add two user parameters, namely a scalar $f \in [0, 1]$, and a new object color $\mathbf{G}$, leading to the matting equation:

$$\mathbf{C}'(x, y) = (1 - f) \mathbf{G} \mathbf{T}(t_x, t_y) + f \mathbf{C}(x, y) \quad (6.25)$$

The parameter f linearly interpolates between the original object color and the texture mapped color. Results for $f = 0$ and $\mathbf{G} = [1 \ 1 \ 1]^T$ are shown in Figure 6.10. This result is included to

Figure 6.10: *A marble texture is applied to the left image, whereas a stone texture is applied to the right image.*

show that our approach is capable of reproducing Fang and Hart’s [FH04]. It also forms the basis for modeling transparency and translucency.

6.5.4 Transparency and translucency

A key factor affecting the appearance of a transparent object is its geometric shape. It determines how light incident from the background is refracted, leading to the characteristic distortions associated with transparent objects. With an appropriate estimation of the object’s shape, such as Ω' (Section 6.3), it should therefore be possible to warp the background image L'_i such that it gives the appearance of a transparent object.

Thus, rather than apply an arbitrary texture to an object, we may use the approach of Section 6.5.3 to create the illusion of transparency. We have found that as long as several conditions are met, convincing results may be obtained. First, the warp needs to be consistent with Ω' , which in turn needs to be at least locally consistent with the actual geometry Ω . Note that this is a weaker requirement than having access to the full volumetric geometry, which would have made

Figure 6.11: A texture suitable for simulating transparency (left, see also middle panel of Figure 1.2, and translucency (right, as well as the right panel of Figure 1.2.)

the problem intractable. Second, the color composition and natural image statistics of the background image should be as outlined in Section 6.4.1. Our basic approach to create transparent and translucent objects is then given by (6.25), whereby $\mathbf{T} = L'_i$ is called the background map.

There are various extensions to this basic solution, leading to different types of transparent materials, from clear glass to frosted glass, to translucency. Each of these involve pre-processing of $\mathbf{T}$. In addition, we have found value in detecting the highlights in the original object, and preserving them. Both topics are discussed in the remainder of this section.

To simulate objects that are not perfectly clear, we convolve the background map $\mathbf{T}(t_x, t_y)$ with a Gaussian filter, as for example shown in Figure 6.11 (left). A much larger filter kernel is used to simulate sub-surface scattering in translucent objects.

The latter type of object additionally receives a color adjustment to simulate wavelength-dependent absorption of light by the material. This adjustment may be achieved by employing a color transfer algorithm [RAG01] whereby the background map is given the same color statistical properties as the object's pixels. A typical treatment to simulate translucent materials is shown in Figure 6.11 (right).

In addition to manipulating $\mathbf{T}(t_x, t_y)$, dark glass may be simulated using an exponential a after normalizing the matting function (6.25):

$$C'_{\text{dark}} = (C'(x, y)/C_{\text{max}})^a \quad (6.26)$$

An exponent a of around $1.2 - 1.4$ produces realistic attenuation. The user parameter f in (6.25) controls the degree to which the object is transformed — $f = 0$ produces a completely transparent object, whereas $f = 1$ yields the original object. In between values model semi-transparent objects.

A final refinement of our transparency/translucency algorithm involves detection and placement of highlights. For objects with relatively sharp highlights, it may be beneficial to retain these in the transparent and translucent versions of the same object, since highlights generally enhance realism [FJB04] and aid the perception of 3D shape [TM83, BB90, BB91].

As we only require detection of highlights on pixels belonging to the object, our highlight detection task is relatively simple. In particular, while highlights are not necessarily the brightest part of a scene, they do tend to be the brightest part of a single object. Furthermore, in conventional imaging, highlights are frequently burned out (“clipped”) due to limitations of the capturing device. By using HDR imaging, highlights may be captured accurately, which also simplifies their detection. Rather than resort to one of several sophisticated schemes [kSK88, Dre94], we therefore propose a simple algorithm.

We will assume that the brightest pixels of the object form the highlights. The histogram of object pixels is analyzed to determine the intensity of the darkest pixel that may still be considered part of the highlight. Intensity histograms of real objects typically show a large peak for the darker pixels, and a long tail forming the highlight pixels (Figure 6.12). We have found that for many objects, the minimum of the histogram’s derivative is a reasonable approximation of the start of the highlight, also shown in Figures 6.12 and 6.13. This method is useful as a starting point, although on occasion the user may wish to manually reduce the size of the highlight somewhat.

Figure 6.12: *Histogram of a high dynamic range image. The derivative is shown in black.*

Once we have detected highlights in the original object, we replace the corresponding pixels in the new image by the pixel values in the original image.

The results of these manipulations are shown in Figure 6.14. Various types of glass may be simulated, including colored and dark glass, as demonstrated in Figure 6.15. Translucency is demonstrated in Figure 6.16.

HDR images also facilitate the recovery of object shape. As the method involves the computation of gradients, it is assumed that quantized low dynamic range data will not be amenable to the recovery of suitable gradients for further processing. Figure 6.17 shows a comparison, where the middle exposure of a sequence of nine images was processed separately. The high dynamic range result shows appropriate curvature in the object, whereas the shape has become flat in the low dynamic range image, and loses the appearance of transparency. Skilled photography, in combination with more forgiving scenes, may yield better results with LDR input.

Figure 6.13: *Using the minimum in the derivative of the histogram, highlight pixels of the left image are detected, and for demonstration purposes painted red in the right image.*

6.6 Conclusions

Given a single high dynamic range photograph, we have presented tools and techniques to change the material properties of objects found in such an image. A key observation enabling dramatic transformations of object appearance is that certain aspects of the interaction between light and matter need to be accurately reproduced, whereas other aspects can be approximated without harm. For instance, we have found that accurate retrieval of the object's surface normals is much more important than the precise refraction of a background through an object. The environment used

Figure 6.14: *Transparency results demonstrated on different objects.*

to light the object needs to have the correct range of colors and intensity, but the reflections and refractions of an object do not have to be positioned with high accuracy.

These observations allow us to produce compelling imagery useful for special effects and digital content creation. We achieve this by carefully reconstructing surface normals based on the luminance distribution of the object. Since this aspect of image-based material editing requires high accuracy, the use of high dynamic range imagery is recommended as this reduces quantization errors.

Our approach thus enables extensive material manipulations using images alone, and does so without the need for expensive equipment, light probes or 3D modeling. In addition to being useful in the special effects community, this work affords exciting opportunities in the study of the human visual system and its perception of materials.

Figure 6.15: *Colored glass (left) and darkened glass (right) simulated with a linear attenuation to achieve color (G in Equation 6.25, and non-linear scaling to simulate dark glass (Equation 6.26). Note that the refractions in these images are different from the ones shown in Figure 6.14 due to a different choice of filter kernel sizes of the bilateral filter, as well as a different choice of scale factor s .*

Figure 6.16: *Translucency applied to various different objects.*

Figure 6.17: *Comparison between HDR and LDR input for a transformation to a glassy material. The top row shows input and output in HDR (tonemapped for display), and the bottom row shows the low dynamic range equivalent. The HDR images were tonemapped with the photographic operator [RSS02a], whereas the LDR images were linearly scaled to fit the display range.*

CHAPTER 7

CONCLUSIONS AND FUTURE WORK

Given a single high dynamic range photograph, we have presented tools and techniques to change the material properties of objects found in such an image. A key observation enabling dramatic transformations of object appearance is that certain aspects of the interaction between light and matter need to be accurately reproduced, whereas other aspects can be approximated without harm. For instance, we have found that accurate retrieval of the object's surface normals is much more important than the precise refraction of a background through an object. The environment used to light the object needs to have the correct range of colors and intensity, but the reflections and refractions of an object do not have to be positioned with high accuracy.

These observations allow us to produce compelling imagery useful for special effects and digital content creation. We achieve this by carefully reconstructing surface normals based on the luminance distribution of the object. Since this aspect of image-based material editing requires high accuracy, the use of high dynamic range imagery is recommended as this reduces quantization errors. Our approach thus enables extensive material manipulations using images alone, and does so without the need for expensive equipment, light probes or 3D modeling. This approach does not introduce significant temporal artifacts if applied to individual frames in a sequence. In addition to being useful in the special effects community, this work affords exciting opportunities in the study of the human visual system and its perception of materials.

As the input into our material editing work is an HDR image, we have also presented a robust method of generating HDR images from multiple exposures. The generated images show significantly diminished noise and ghosting artifacts, provided that the images predominantly consist of accurate pixel values. Removing random noise from images is significantly easier, and generally takes a single iteration of our algorithm to remove. Ghosting invariably takes several iterations.

Our algorithm may be applied to scenes where ghosting occurs in large portions of the scene, as well as scenes in which ghosting occurs in regions with high dynamic range.

To make it easier for the user to specify an alpha matte we have also developed a method to automatically generate an alpha matte, given two high dynamic range images of the object as input. The images are identical to the original image except for the fact that one focuses on the object while the background is blurred, and the other focuses on the background in the scene and the object is blurred. This specific input allows us to compare the details of the scene at each pixel location, and use this information to determine whether the pixel belongs to the object or not.

The output is an approximate binary matte, which may be converted into a trimap, and then fine-tuned, to give a reasonable alpha matte. As the problem is under-constrained, we have used additional information about the scene, such as smoothness across space, user supplied information, and bounds for the unknown variables. Mattes of HDR images are extracted, as the composite equation is physically invalid for the color values in standard images, that have undergone transformation under nonlinear radiometric response functions.

Both parts of the alpha matting algorithm rely on computing similarity of pixels on the basis of color. To select a colorspace that is effective against illumination changes, and is perceptually linear, we have designed an experiment to test the performance of several color spaces. We have found the $L\alpha\beta$ space to be the most suitable color space for our purposes.

We have published most of the work that has been discussed here, and provide a list of publications below:

- Erum Arif Khan, Erik Reinhard, Roland Fleming and Heinrich Buelthoff, 'Image-based Material Editing', ACM Transactions on Graphics (Siggraph), 2006
- Erum Arif Khan, Ahmet Oguz Akyuz, Erik Reinhard, 'Ghost Removal in High Dynamic Range Images', IEEE International Conference on Image Processing, 2006

- Erik Reinhard and Erum Arif Khan, 'Depth-of-field-based Alpha-matte Extraction', Second ACM Symposium on Applied Perception in Graphics and Visualization, 2005.
- Erik Reinhard, Erum Arif Khan, Oguz Akyuz, Roland Fleming and Heinrich Buelthoff, 'Image-based Material Editing', video, Siggraph Animation Theater, 2005
- Erum Arif Khan, Erik Reinhard, Roland Fleming and Heinrich Buelthoff, 'Image-based Material Editing', sketch, Siggraph, 2005
- Erum Arif Khan, Erik Reinhard, Roland Fleming and Heinrich Buelthoff, 'Image-based Material Editing', University of Central Florida Technical Report, CS-TR-05-09, 2005
- Erum Arif Khan and Erik Reinhard, 'Evaluation of Color Spaces for Edge Classification in Outdoor Scenes', IEEE International Conference on Image Processing, 2005
- Erum A Khan and Erik Reinhard, 'A Survey of Color Spaces for Shadow Identification', Abstract, ACM Symposium on Applied Perception in Computer Graphics and Visualization, 2004

One of the more important contributions of our work is to demonstrate that it is possible to significantly approximate the model of a scene, without introducing any noticeable artifacts. It is hoped that in the long term, this demonstration will help steer the direction of future work pertaining to scene approximation. In graphics, where more and more accurate models are being used, our work shows that such accuracy and its associated high computation time may not be necessary for visual realism, and that development of simpler models will considerably decrease image generation time without necessarily sacrificing realism.

Our work opens up a variety of venues that we intend to pursue in future. Currently, the field of image-based material editing is in its initial phase, where we have obtained viewers' subjective responses to images in which materials of objects have been transformed. While most viewers

find the images realistic, some images seem 'more' realistic than others. Also, realistic images of certain material transformations are generated after only a few tries, while other material transformations take many more attempts to generate a realistic looking image. We intend to design experiments that will quantitatively assess the quality of the generated images. We also intend to use our current results in experiments that attempt to better understand the human visual system, by answering questions like: Which specific features can we modify in our environment without perceiving the change? Why do we perceive inaccuracies in some material transformations more easily than in others? Why do we achieve temporal stability for certain material transforms and not others?

Another important area of future work is to enable material transformations that our work is not capable of as yet. So far, we have not transformed the material of an object into one that will change the silhouette (and therefore alpha matte) of the object. For example, making the object look like it is made of hair or fur would necessarily change its silhouette. We also intend to look further into material transformations where the original material is such that the object's luminance values are an inadequate representation of its depth. For instance, changing the material of an object that is made of glass would be impossible with our current approach.

Often, the material transformation made by our algorithm is realistic, but contradictory information in the remainder of the image makes the transformed scene appear less realistic. For instance, when we replace an opaque object by a transparent one, the change should be accompanied by the removal of shadows which were cast by the previously opaque object. We might also expect to see caustics in the scene around the new transparent object. As we do not currently make changes to the remainder of the image, stark shadows appear to be cast by transparent objects and caustics are noticeably absent. Since we are not used to seeing such conflicting cues in our environment, the HVS immediately flags the image as unrealistic. A potential way of improving our results would be to make changes to the background as well, so that no conflicting cues remain.

All the above venues of future work potentially involve significant work and time. There are also several specific developments that we intend to make to our existing work in the short term. For instance, we intend to explore the possibility of using our noise and ghost removal work to remove artifacts caused by misalignment of individual exposures. This will enable our algorithm to generate artifact free HDR images from exposures taken by a moving (possibly hand-held) camera. We also intend to improve results from our alpha matte extraction technique by incorporating texture to optimize the initial trimap. The more informative features we can use to separate the object from the background, the more reliable the final alpha matte will be. Finally, while optimizing our initial estimate of the alpha matte, we intend to increase the window size in which we perform the optimization. This change will make optimization less local, and will therefore improve the accuracy of the final alpha matte.

LIST OF REFERENCES

- [Ade82] E H Adelson. “Saturation and adaptation in the rod system.” *Vision Research*, **22**:1299–1312, 1982.
- [Ade01] E H Adelson. “On Seeing Stuff: The Perception of Materials by Humans and Machines.” *Proceedings of the SPIE, Human Vision and Electronic Imaging*, **4299**:1–12, 2001.
- [APS00] M Ashikhmin, S Premoze, and P Shirley. “A microfacet-based BRDF generator.” *Siggraph*, pp. 65–74, 2000.
- [AS00] M Ashikhmin and P Shirley. “An anisotropic Phong BRDF model.” *Journal of Graphics Tools*, **5**(2):25–32, 2000.
- [ASL05] E H Adelson, L Sharan, and Y Li. “Material Perception.” *CSAIL Research Abstract*, 2005.
- [BB90] A Blake and H. H. Bülthoff. “Does the brain know the physics of specular reflection?” *Nature*, **343**(6254):165–168, 1990.
- [BB91] A Blake and H. H. Bülthoff. “Shape from Specularities: Computation and Psychophysics.” *Philosophical Transactions of the Royal Society (London) Series B*, **331**:237–252, 1991.
- [BJ01] Y Boykov and M-P Jolly. “Interactive graph cuts for optimal boundary and region segmentation of objects in N-D images.” In *Proceedings of IEEE International Conference on Computer Vision*, 2001.
- [BKY99] P N Belhumeur, D J Kriegman, and A. L. Yuille. “The basrelief ambiguity.” In *IJCV*, volume 35, pp. 33–44, 1999.
- [Bog00] L Bogoni. “Extending Dynamic Range of Monochrome and Color Images through Fusion.” In *International Conference on Pattern Recognition*, volume 3, pp. 7–12, 2000.
- [BSC00] Marcelo Bertalmio, Guillermo Sapiro, Vincent Caselles, and Coloma Ballester. “Image inpainting.” In *SIGGRAPH '00: Proceedings of the 27th annual conference on computer graphics and interactive techniques*, pp. 417–424, 2000.
- [CAC02] Yung-Yu Chuang, Aseem Agarwala, Brian Curless, David H. Salesin, and Richard Szeliski. “Video matting of complex scenes.” In *SIGGRAPH '02: Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, pp. 243–248, New York, NY, USA, 2002. ACM Press.
- [CCS01] Y-Y Chuang, B Curless, D Salesin, and R Szeliski. “A Bayesian approach to digital matting.” In *Proceedings of Computer Vision and Pattern Recognition (CVPR 2001)*, volume II, pp. 264–271, 2001.
- [CGP01] Rita Cucchiara, Costantino Grana, Massimo Piccardi, Andrea Prati, and Stefano Sirotti. “Improving Shadow Suppression in Moving Object Detection with HSV Color Information.” *IEEE Intelligent Transportation Systems*, pp. 334–339, 2001.

- [CKS97] Vicent Caselles, Ron Kimmel, and Guillermo Sapiro. “Geodesic Active Contours.” *International Journal of Computer Vision*, **22**(1):61–79, 1997.
- [CL88] Raphael Charit and Murray H Loew. “Complex Shadow Boundary Segmentation Using the Entry-Exit Method.” In *Computer Vision and Pattern Recognition*, 1988.
- [Dan01] K J Dana. “BRDF/ BTF Measurement Device.” In *Eighth International Conference on Computer Vision*, volume 2, p. 460, 2001.
- [DB03] Steven C Dakin and Peter J Bex. “Natural image statistics mediate brightness filling in.” *Proceedings of the Royal Society of London, B*, **270**:2341–2348, 2003.
- [DCY03] Iddo Drori, Daniel Cohen-Or, and Hezy Yeshurun. “Fragment-Based Image Completion.” *ACM Transactions on Graphics*, **22**(3):303–312, 2003.
- [Deb98] Paul E Debevec. “Rendering synthetic objects into real scenes: bridging traditional and image-based graphics with illumination and high dynamic range photography.” In *SIGGRAPH 98 Conference Proceedings*, Annual Conference Series, pp. 45–50, 1998.
- [Deb02] Paul Debevec. “A tutorial on image-based lighting.” *IEEE Computer Graphics and Applications*, **22**(2):26–34, 2002.
- [DF03] Mark S Drew and Graham D Finlayson. “Recovery of Chromaticity Image Free from Shadows via Illumination Invariance.” In *IEEE Workshop on Color and Photometric Methods in Computer Vision, ICCV*, pp. 32–39, 2003.
- [DM97a] P Debevec and J Malik. “Recovering High Dynamic Range Radiance Maps from Photographs.” In *SIGGRAPH 97 Conference Proceedings*, Annual Conference Series, pp. 369–378, August 1997.
- [DM97b] Paul E. Debevec and Jitendra Malik. “Recovering High Dynamic Range Radiance Maps from Photographs.” In *SIGGRAPH 97 Conference Proceedings*, Annual Conference Series, pp. 369–378, August 1997.
- [Dow87] J E Dowling. *The retina: an approachable part of the brain*. Belknap Press, Cambridge, 1987.
- [Dre94] Mark S Drew. “Robust Specularity Detection from a Single Multi-illuminant Color Image.” *Computer Vision, Graphics, and Image Processing. Image Understanding*, **59**(3):320–327, 1994.
- [ED04] Elmar Eisemann and Frdo Durand. “Flash Photography Enhancement Via Intrinsic Relighting.” In *SIGGRAPH*. ACM Press, 2004.
- [EL93] John Ens and Peter Lawrence. “An investigation of methods for determining depth from focus.” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **15**(2):97–108, 1993.
- [FDA01] Roland W Fleming, Ron A Dror, and E H Adelson. “How Humans Determine Reflectance Properties under Unknown Illumination?” *Proceedings of the IEEE workshop on Identifying Objects Across Variations in Lighting: Psychophysics and Computation*, 2001.

- [FH04] Hui Fang and John C Hart. “Textureshop: Texture Synthesis as a Photographic Editing Tool.” *ACM Transactions on Graphics*, **23**(3):354–358, 2004.
- [FJB04] Roland W Fleming, Heinrik Wann Jensen, and Heinrich H Bülthoff. “Perceiving translucent materials.” In *APGV’04: Proceedings of the 1st Symposium on Applied Perception in Graphics and Visualization*, pp. 127–134, 2004.
- [FTA04] Roland W Fleming, A Torralba, and E H Adelson. “Specular reflections and the perception of shape.” *Journal of Vision*, **4**(9):798–820, 2004.
- [FvF90] James Foley, Andries van Dam, Steven Feiner, and John Hughes. *Computer graphics, principles and practice*. Addison-Wesley, 2nd ed. edition, 1990.
- [GMW81] P E Gill, W Murray, and M H Wright. *Practical Optimization*. Academic Press, London, 1981.
- [Goe03] F J Goerlich. “Weighted samples, kernel density estimators and convergence.” In *Empirical Economics*, volume 28, pp. 335–351, 2003.
- [GPS89] D Greig, B Porteous, and A Seheult. “Exact MAP estimation for binary images.” *Journal of the Royal Statistical Society, B*, **51**:271–279, 1989.
- [GS03] Theo Gevers and Harro Stokman. “Classifying Color Edges in Video Into Shadow-Geometry, Highlight, or Material Transitions.” *IEEE Transactions on Multimedia*, **5**, 2003.
- [HF79] D C Hood and M A Finkelstein. “Comparison of changes in sensitivity and sensation: implications for the response-intensity function of the human photopic system.” *Journal of Experimental Psychology: Human Perceptual Performance*, **5**:391–405, 1979.
- [HFK94] J C Hart, G K Francis, and L H Kauffman. “Visualizing Quaternion Rotation.” volume 13, pp. 256–276, 1994.
- [HM82] J A Hanley and B J McNeil. “The meaning and use of the area under the Receiver Operating Characteristic (ROC) curve.” In *Radiology*, volume 143, pp. 29–36, 1982.
- [Hor74] Berthold K P Horn. “Determining Lightness from an Image.” *CVGIP*, **3**:277–299, 1974.
- [Hur86] Anya Hurlbert. “Formal Connections between Lightness Algorithms.” *Journal of the Optical Society of America A*, **3**:1684–1693, 1986.
- [IMT99] T Igarashi, S Matsuoka, and Hidehiko Tanaka. “Teddy: A Sketching Interface for 3D Freeform Design.” In *Proceedings of SIGGRAPH*, 1999.
- [ITU90] ITU. *International Telecommunication Union ITU-R Recommendation BT.709, Basic Parameter Values for the HDTV Standard for the Studio and for International Programme Exchange*. Geneva, 1990. Formerly CCIR Rec. 709.
- [JWL05] K Jacobs, G Ward, and C Loscos. “Automatic HDRI Generation of Dynamic Environments.” In *Sketch, Siggraph 2005, Annual Conference Series*, 2005.

- [Kaj86] James T. Kajiya. “The Rendering Equation.” *ACM Computer Graphics*, **20**(4):143–150, aug 1986. held in Dallas, Texas, August 18–22, 1986.
- [Kan98] S Kang. “Depth-Painting for Image Based Rendering Applications.” Technical report, Microsoft, 1998.
- [KD75] Jochen Kleinschmidt and John E Dowling. “Intracellular recordings from gecko photoreceptors during light and dark adaptation.” *J gen Physiol*, **66**:617–648, 1975.
- [KnC96] J J Koenderink, A J van Doorn, C Christou, and J S Lappin. “Perturbation study of shading in pictures.” *Perception*, **25**:1009–1026, 1996.
- [kSK88] G J klinker, S A Shafer, and T Kanade. “The measurement of highlights in color images.” *International Journal of Computer Vision*, **2**(1):7–32, 1988.
- [Kuf53] S W Kuffler. “Discharge patterns and functional organization of mammalian retina.” *Journal of Neurophysiology*, **16**:37–68, 1953.
- [KUW03] Sing Bing Kang, Matthew Uyttendaele, Simon Winder, and Richard Szeliski. “High dynamic range video.” *ACM Transactions on Graphics*, **22**(3), 2003.
- [LB00] Michael S Langer and Heinrich H Bülthoff. “Depth discrimination from shading under diffuse lighting.” *Perception*, **29**(6):649–660, 2000.
- [LBD03] R. Lawson, H. H. Bülthoff, and S. Dumbell. “Interactions between view changes and shape changes in picture - picture matching.” *Perception*, **32**:1465–1498, 2003.
- [LFT97] Eric Lafortune, S.-C Foo, K Torrance, and D Greenberg. “Non-linear approximation of reflectance functions.” In *Computer Graphics 31, Annual Conference Series*, pp. 117–126, 1997.
- [LL95] D K Lynch and W Livingston. *Color and Light in Nature*. Cambridge University Press, 1995.
- [LLY06] A Levin, D Lischinski, and Weiss Y. “A Closed Form Solution to Natural Image Matting.” In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2006.
- [LST04] Yin Li, Jian Sun, Chi-Keung Tang, and Heung-Yeung Shum. “Lazy snapping.” *ACM Transactions on Graphics*, **23**(3):303–308, 2004.
- [LZ94] M S Langer and S W Zucker. “Shape-from-shading on a cloudy day.” *Journal of the Optical Society of America A*, **11**:467–478, 1994.
- [Mar82] David Marr. *Vision, a computational investigation into the human representation and processing of visual information*. W H Freeman and Company, San Fransisco, 1982.
- [MB95] Eric N. Mortensen and William A. Barrett. “Intelligent scissors for image composition.” In *SIGGRAPH ’95: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pp. 191–198, New York, NY, USA, 1995. ACM Press.

- [MB99] Eric N. Mortensen and William A. Barrett. “Tobogan-based intelligent scissors with a four parameter edge model.” In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pp. 452–458, 1999.
- [Min54] M Minnaert. *Light and Color in the Open Air*. Dover, 1954.
- [Mis93] Y Mishima. “Soft edge chroma-key generation based on hexoctahedral color space.” In *US Patent 5,355,174*, 1993.
- [MML98] A S Mortlock, D M McConnell, and M Lessart. “Automatic head conformance from front and profile images.” In *Proceedings of the IEE Colloquium on Computer Vision for Virtual Human Modelling*, 1998.
- [MMP05] M McGuire, W Matusik, H Pfister, J F Hughes, and F Durand. “Defocus video matting.” *ACM Transactions on Graphics*, **24**, 2005.
- [MN99] T Mitsunaga and S K Nayar. “Radiometric self-calibration.” In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume I, pp. 374–380. IEEE, 1999.
- [MN00] T Mitsunaga and S K Nayar. “High dynamic range imaging: Spatially varying pixel exposures.” In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume I, pp. 472–479. IEEE, 2000.
- [MP95] S Mann and R Picard. “On being ‘undigital’ with digital cameras: Extending dynamic range by combining differently exposed pictures.” *IS&T’s 48th Annual Conference*, pp. 422–428, 1995.
- [MPB03] Wojciech Matusik, Hanspeter Pfister, Matt Brand, and Leonard McMillan. “A data-driven reflectance model.” *ACM Transactions on Graphics*, **22**(3):759–769, 2003.
- [MYT95] Tomoo Mitsunaga, Taku Yokoyama, and Takashi Totsuka. “AutoKey: human assisted key extraction.” In *SIGGRAPH ’95: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pp. 265–272, New York, NY, USA, 1995. ACM Press.
- [nP05a] Susan F te Pas and Sylvia C Pont. “A comparison of material and illumination discrimination performance for real rough, real smooth and computer generated smooth spheres.” In *APGV ’05: Proceedings of the 2nd ACM Symposium on Applied Perception in Graphics and Visualization*, pp. 75–81, 2005.
- [nP05b] Susan F te Pas and Sylvia C Pont. “Estimations of light source direction depend critically on material BRDFs.” *Perception*, **34**:212, 2005. Supplement ECV05, A Coruña.
- [NPH01] Heiko Neumann, Luiz Pessoa, and Thorsten Hansen. “Visual filling-in for computing perceptual surface properties.” *Biological Cybernetics*, **85**:355–369, 2001.
- [NR66] K I Naka and W A H Rushton. “S-potentials from luminosity units in the retina of fish (Cyprinidae).” *J Physiol*, **185**:587–599, 1966.
- [OCD01] Byong Mok Oh, Max Chen, Julie Dorsey, and Frédo Durand. “Image-Based Modeling and Photo Editing.” In *Proceedings ACM SIGGRAPH 2001*, pp. 433–442, 2001.

- [OCS05] Yuri Ostrovsky, Patrick Cavanagh, and Pawan Sinha. “Perceiving illumination inconsistencies.” *Perception*, **34**:1301–1314, 2005.
- [ODJ04] Victor Ostromoukhov, Charles Donohue, and Pierre-Marc Jodoin. “Fast hierarchical importance sampling with blue noise properties.” *ACM Transactions on Graphics*, **23**(3):488–495, 2004.
- [Oli91] J. Oliensis. “Shape from shading as a partially wellconstrained problem.” In *CVGIP: Image Understanding*, volume 54, pp. 163–183, 1991.
- [Oli02] Manuel M Oliveira. “Image-Based Modeling and Rendering Techniques: A Survey.” *RITA - Revista de Informática Teórica e Aplicada*, **IX**:37–66, October 2002.
- [Pal99] Stephen E Palmer. *Vision Science: Photons to Phenomenology*. The MIT Press, Cambridge, Massachusetts, 1999.
- [Par62] E Parzen. “On Estimation of a Probability Density and Mode.” In *Annals of Math. Statistics*, 1962.
- [Pen94] A. Pentland. “Local shading analysis.” In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 6, pp. 170–187, 1994.
- [PK04] J. Park and A. C. Kak. “Multi-peak range imaging for accurate 3d reconstruction of specular objects.” In *Proceedings of the Sixth Asian Conference on Computer Vision*, January 2004.
- [PM90] Pietro Perona and Jitendra Malik. “Scale-space and edge detection using anisotropic diffusion.” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **12**(7):629–639, 1990.
- [PMC01] A Prati, I Mikic, R Cucchiara, and M Trivedi. “Comparative evaluation of moving shadow detection algorithms.” In *IEEE CVPR workshop on Empirical Evaluation Methods in Computer Vision*, Kauai, 2001.
- [PTY00] Sumanta N. Pattanaik, Jack Tumblin, Hector Yee, and Donald P. Greenberg. “Time-Dependent Visual Adaptation for Fast Realistic Display.” In *SIGGRAPH 2000 Conference Proceedings*, pp. 47–54, July 2000.
- [RAG01] Erik Reinhard, Michael Ashikhmin, Bruce Gooch, and Peter Shirley. “Color transfer between images.” *IEEE Computer Graphics and Applications*, **21**:34–41, September/October 2001.
- [RBS99] M Robertson, S Borman, and R Stevenson. “Dynamic Range Improvement Through Multiple Exposures.” In *International Conference on Image Processing*. IEEE, 1999.
- [RCC98a] D L Ruderman, T W Cronin, and C-C Chiao. “Statistics of Cone Responses to Natural Images: Implications for Visual Coding.” In *Journal of the Optical Society of America A*, volume 15, pp. 2036–2045, 1998.
- [RCC98b] D L Ruderman, T W Cronin, and C-C Chiao. “Statistics of Cone Responses to Natural Images: Implications for Visual Coding.” *J. Opt. Soc. Am. A*, **15**(8):2036–2045, 1998.

- [RD05] Erik Reinhard and Kate Devlin. “Dynamic Range Reduction Inspired by Photoreceptor Physiology.” *IEEE Transactions on Visualization and Computer Graphics*, **11**(1):13–24, January/February 2005.
- [RH01] Ravi Ramamoorthi and Pat Hanrahan. “A signal-processing framework for reflection.” *ACM Transactions on Graphics*, **23**(4):1004–1042, 2001.
- [RK05] Erik Reinhard and Erum Khan. “Depth-of-Field-Based Alpha-Matte-Extraction.” In *Second ACM Symposium on Applied Perception in Graphics and Visualization (APGV)*, 2005.
- [RKB04] Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. ““GrabCut” — Interactive foreground extraction using iterated graph cuts.” *ACM Transactions on Graphics*, **23**(3):309–314, 2004.
- [Ros56] M Rosenblatt. “Remarks on Some Nonparametric Estimates of Density Functions.” In *Annals of Math. Statistics*, 1956.
- [RSA04] Erik Reinhard, Peter Shirley, Michael Ashikhmin, and Tom Troscianko. “Second order image statistics for computer graphics.” 2004.
- [RSS02a] E Reinhard, M Stark, P Shirley, and J Ferwerda. “Photographic Tone Reproduction for Digital Images.” volume 21, pp. 267–276, 2002.
- [RSS02b] Erik Reinhard, Michael Stark, Peter Shirley, and Jim Ferwerda. “Photographic Tone Reproduction for Digital Images.” *ACM Transactions on Graphics*, **21**(3):267–276, 2002.
- [RT92] E. Rouy and A. Tourin. “A Viscosity Solutions Approach to Shape-from-Shading.” In *SIAM Num. An.*, volume 29, pp. 867–884, 1992.
- [Rud97a] D L Ruderman. “Origins of Scaling in Natural Images.” *Vision Research*, **37**:3385–3398, 1997.
- [Rud97b] D L Ruderman. “The Statistics of Natural Images.” *Network: Computation in Neural Systems*, **5**(4):517–548, 1997.
- [RWP05a] E Reinhard, G Ward, S Pattanaik, and P Debevec. *High Dynamic Range Imaging: Acquisition, Display, and Image-Based Lighting*. Morgan Kaufmann, 2005.
- [RWP05b] Erik Reinhard, Greg Ward, Sumanta Pattanaik, and Paul Debevec. *High Dynamic Range Imaging: Acquisition, Display and Image-Based Lighting*. Morgan Kaufmann Publishers, San Francisco, 2005.
- [SA05] D J Simons and M S Ambinder. “Change Blindness.” *Current Directions in Psychological Science*, **14**:44, 2005.
- [SB96] A R Smith and J F Blinn. “Blue Screen Matting.” In *SIGGRAPH ’96 Proceedings*, pp. 259–268, 1996.
- [SCE01] Elena Salvador, Andrea Cavallaro, and Touradj Ebrahimi. “Shadow Identification and Classification Using Invariant Color Models.” In *International Conference on Acoustics, Speech, and Signal Processing*, 2001.

- [Sch94] Christophe Schlick. “Quantization Techniques for the Visualization of High Dynamic Range Pictures.” In Peter Shirley, Georgios Sakas, and Stefan Müller, editors, *Photorealistic Rendering Techniques*, pp. 7–20. Springer-Verlag Berlin Heidelberg New York, 1994.
- [Sch98] A van der Schaaf. *Natural image statistics and visual processing*. PhD thesis, Rijksuniversiteit Groningen, The Netherlands, March 1998.
- [SJT04] Jian Sun, Jiaya Jia, Chi-Keung Tang, and Heung-Yeung Shum. “Poisson matting.” *ACM Trans. Graph.*, **23**(3):315–321, 2004.
- [SL97] D J Simons and D T Levin. “Change blindness.” *Trends in Cognitive Science*, **1**:261–267, 1997.
- [SL98] D J Simons and D T Levin. “Failure to detect changes to people in a real-world interaction.” *Psychonomic Bulletin and Review*, **5**:644–649, 1998.
- [SYJ05] Jian Sun, Lu Yuan, Jiaya Jia, and Heung-Yeung Shum. “Image Completion with Structure Propagation.” *ACM Transactions on Graphics*, **24**(3):861–868, 2005.
- [TBP03] Tom Troscianko, Roland Baddeley, Carlos A Párraga, Ute Leonards, and Jolyon Troscianko. “Visual Encoding of Green Leaves in Primate Vision.” *Journal of Vision*, **3**, 2003.
- [THG99] Jack Tumblin, Jessica K. Hodgins, and Brian K. Guenter. “Two methods for display of high contrast images.” *ACM Transactions on Graphics*, **18** (1):56–94, 1999.
- [TKB98] M. J. Tarr, D. Kersten, and H. H. Bühlhoff. “Why the visual recognition system might encode the effects of illumination.” *Vision Research*, **38**:2259–2275, 1998.
- [TM83] J T Todd and E Mingolla. “Perception of surface curvature and direction of illuminant from patterns of shading.” *Journal of Experimental Psychology: Human Perception and Performance*, **9**:583–595, 1983.
- [TM98] C Tomasi and R Manduchi. “Bilateral filtering for gray and color images.” In *Proc. IEEE International Conference on Computer Vision*, pp. 836–846, 1998.
- [TO03] Antonio Torralba and Aude Oliva. “Statistics of natural image categories.” *Network: Comput. Neural Syst.*, **14**:391–412, 2003.
- [TR89] J T Todd and F D Reichel. “Ordinal structure in the visual perception and cognition of smoothly curved surfaces.” *Psychological Review*, **96**:643–657, 1989.
- [UER01] M Uyttendaele, A Eden, and Szeliski R. “Eliminating Ghosting and Exposure Artifacts in Image Mosaics.” In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, p. 509, 2001.
- [VNV03] D Van De Ville, M Nachtegaele, D Ven Der Weken, E E Kerre, W Phillips, and L Ignace. “Noise Reduction by Fuzzy Image Filtering.” In *IEEE Transactions on Fuzzy Systems*, volume 11, 2003.
- [WA94] J Y A Wang and E Adelson. “Representing Moving Images with Layers.” In *IEEE Transactions on Image Processing*, volume 3, pp. 369–378, May 1994.

- [Wan95] Brian A Wandell. *Foundations of Vision*. Sinauer Associates, inc., 1995.
- [War92] G J Ward. “Measuring and Modeling Anisotropic Reflection.” *ACM Computer Graphics*, **26**(2):265–272, jul 1992.
- [War03] Greg Ward. “Fast, robust image registration for compositing high dynamic range photographs from hand-held exposures.” *Journal of Graphics Tools*, **8**(2):17–30, 2003.
- [Wei97] Joachim Weickert. “A review of non-linear diffusion filtering.” In Bart ter Haar Romeny, Luc Florac, Jan Koenderink, and Max Viergever, editors, *Scale-Space Theory in Computer Vision*, pp. 3–28, Berlin, 1997. Springer-Verlag.
- [Wil98] L Williams. “Image Jets, Level Sets, and Silhouettes.”, 1998.
- [Win01] Piotr S Windyga. “Fast Impulsive Noise Removal.” *IEEE Transactions on Image Processing*, **10**(1):173–179, 2001.
- [Wit83] A P Witkin. “Scale-space filtering.” In *Proceedings of the Eighth International Joint Conference on Artificial Intelligence*, 2, pp. 1019–1022, 1983.
- [WLG01] James Z Wang, Jia Li, Robert M Gray, and Gio Wiederhold. “Unsupervised multiresolution segmentation for images with low depth of field.” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **23**(1):85–90, 2001.
- [WPG02] Chee Sun Won, Kyungsuk Pyun, and Robert M Gray. “Automatic object segmentation in images with low depth of field.” In *Proceedings of IEEE International Conference on Image Processing*, pp. III.805 – III.808, 2002.
- [WS82] G Wyszecki and W S Stiles. *Color science: Concepts and methods, quantitative data and formulae*. John Wiley and Sons, New York, 2nd edition, 1982.
- [XP] Ruifeng Xu and Sumanta N Pattanaik. “A Novel Monte Carlo Noise Reduction Operator.” *IEEE Computer Graphics and Applications*, p. 31.
- [ZFG05] Steve Zelinka, Hui Fang, Michael Garland, and John Hart. “Interactive Material Replacement in Photographs.” *Proceedings of Graphics Interface 2005, Canadian Human-Computer Communications Society*, pp. 227–232, 2005.
- [ZG04] Steve Zelinka and Michael Garland. “Jump map-based interactive texture synthesis.” *ACM Transactions on Graphics*, **23**(4):930–962, 2004.
- [ZTC99] Ruo Zhang, Ping-Sing Tsai, James Cryer, and Mubarak Shah. “Shape from shading: a survey.” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **21**(8):690–706, 1999.