UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

EDUARDO SIMÕES LOPES GASTAL

Real-Time Alpha Matting for Natural Images and Videos

Undergraduate Thesis presented in partial fulfillment
of the requirements for the degree of
Bachelor of Computer Science

Prof. Dr. Manuel Menezes de Oliveira Neto
Advisor

Porto Alegre, November 2009

CIP – CATALOGING-IN-PUBLICATION

Gastal, Eduardo Simões Lopes

Real-Time Alpha Matting for Natural Images and Videos / Eduardo Simões Lopes Gastal. – Porto Alegre, 2009.

57 f.: il.

Trabalho de Conclusão (graduação)/Undergraduate Thesis – Universidade Federal do Rio Grande do Sul. Curso de Bacharelado em Ciência da Computação, Porto Alegre, BR–RS, 2009. Advisor: Manuel Menezes de Oliveira Neto.

1. Alpha matting. 2. Compositing. 3. Image and video processing. I. de Oliveira Neto, Manuel Menezes. II. Título.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

Image matting aims at extracting foreground elements from an image by means of color and opacity (alpha) estimation. While a lot of progress has been made in recent years on improving the accuracy of matting techniques, one common problem persisted: the low speed of matte computation.

This work presents the first real-time matting technique for natural images and videos. The proposed technique is based on the observation that, for small neighborhoods, pixels tend to share similar attributes. Therefore, independently treating each pixel in the unknown regions of a trimap results in a lot of redundant work. We show how this computation can be significantly and safely reduced by means of a careful selection of pairs of background and foreground samples.

Our technique achieves speedups of up to two orders of magnitude compared to previous ones, while producing high-quality alpha mattes. The quality of the presented results has been verified through an independent benchmark. The speed of our technique enables, for the first time, real-time alpha matting of videos, and has the potential to enable a new class of exciting real-time applications.

Keywords: Alpha matting, compositing, image and video processing.

Alpha Matting em Tempo Real para Imagens e Vídeos Naturais

RESUMO

O processo conhecido como alpha matting visa extrair objetos que aparecem no primeiro plano de uma imagem (ou vídeo), separando-os do fundo da mesma. Para tanto, calcula-se, para cada pixel pertencente a um objeto, uma estimativa da cor original e do grau de transparência do objeto naquele ponto. Este problema possui diversas aplicações (onde se destacam edição de imagens e produção cinematográfica), o que levou ao desenvolvimento de muitas técnicas ao longo dos anos. Tais técnicas, particularmente nos últimos anos, conseguem resultados muito bons para extração de objetos de imagens com fundos heterogêneos, comumente chamadas de imagens naturais. Entretanto, uma limitação ainda existia: baixa velocidade de computação.

Este trabalho apresenta a primeira técnica de alpha matting em tempo real para imagens e vídeos naturais. A técnica proposta é baseada na observação que, para pequenas vizinhanças, pixels tendem a compartilhar atributos semelhantes. Logo, tratar independentemente cada pixel pertencente a regiões de incerteza resulta em muito trabalho redundante. Nós mostramos como esta computação pode ser significamente e seguramente reduzida através de uma seleção cuidadosa de pares de cores.

A técnica proposta consegue obter resultados de alta qualidade de maneira 100 vezes mais rápida quando comparada à técnicas anteriores. A qualidade de tais resultados foi verificada através de um “benchmark” independente desenvolvido por terceiros. Devido a sua velocidade nunca antes vista, a técnica proposta permite, pela primeira vez, alpha matting em tempo real de vídeos, e apresenta grande potencial para possibilitar uma nova classe de aplicações que utilizem alpha matting em tempo real.

Palavras-chave: Remoção de Fundo, Alpha Matting, Tempo Real.

1 INTRODUCTION

Extraction and compositing of foreground objects are fundamental image and video editing operations. These operations are extensively used by many applications in the fields of computer vision and computer graphics, where common examples include compositing of images for magazines and newspapers, or even insertion of visual information for television broadcasting. Likewise, it its commonplace in the film industry to extract actors from videos captured in closed studios to insert them in novel locations, either due to feasibility or cost. However, due to the discretization process involved in capturing digital images, pixels located in boundary regions receive light from multiple scene elements. Consequently, boundary pixels p will have *mixed* colors, resulting from a linear combination of colors from all objects covering p. This mixture is weighted by the *relative coverage* of each object in relation to the pixel. The process of estimating the original colors and weights that form the final pixel color is known as *alpha matting*.

The matting problem was mathematically established by Porter and Duff (1984), where they introduced the *alpha channel* (also known as *matte*) as the means to control the linear interpolation of foreground and background colors. Formally, to accurately extract a foreground objects from images and videos, matting techniques need to estimate foreground (F) and background (B) colors for all pixels belonging to an image I, along with opacity (α) values. These values are related by the *compositing Equation* 1.1, where the observed color of pixel I_i is expressed as a linear combination of F_i and B_i, with interpolation parameter α_i:

$$I_i = \alpha_i F_i + (1 - \alpha_i) B_i \tag{1.1}$$

For *natural images*, F and B are not constrained to a particular subset of values. Thus, all variables on the right-hand side of Equation 1.1 are unknown, making the matting problem inherently under-constrained — *i.e.*, for RGB images, 7 unknown variables need to be estimated from 3 known values.

Due to this highly ill-posed nature of the matting problem, most existing approaches require additional constraints in the form of user input, either as *trimaps* or *scribbles* (described in Chapter 2). This user-supplied information identifies pixels for which the opacity value α_i is known to be 1 or 0, i.e. *known foreground* and *known background* pixels, respectively. The remaining unconstrained pixels are marked as *unknown*. The goal of a digital matting algorithm is then to compute the values of α_i, F_i, and B_i for all pixels labeled as unknown. Finally, the foreground can be seamlessly composed onto a novel background by replacing the original background B with a new background B' in Equation 1.1.

Existing matting techniques can be classified according to the underlying method used for solving the matte (WANG; COHEN, 2008), which can be based on *sampling*, *pixel affinities* or a combination of the two (detailed in Section 2.2). Most recent matting algorithms (SINGARAJU; ROTHER; RHEMANN, 2009; LEVIN; RAV-ACHA; LISCHINSKI, 2008; WANG; COHEN, 2007; LEVIN; LISCHINSKI; WEISS, 2008; RHEMANN; ROTHER; GELAUTZ, 2008) fall in one of the last two categories, where local affinities are employed in optimization steps for solving or refining the matte. This usually requires the solution of large linear systems, and the size of these linear systems is directly proportional to the percentage of unknown pixels in I. As a result, the dimensions of such linear systems can get quite sizable. Furthermore, these optimization procedures solve for α independently of F and B, thus requiring an additional step for reconstructing F and, if necessary, also B. Consequently, state-of-the-art techniques take from several seconds to minutes to generate alpha mattes for typical images (with about 1 Megapixels).

The slow processing speed of recent matting methods makes the matte creation process a tedious task. Long offline computations have also prevented the use of natural scenes in real-time matting applications, such as live video broadcasting.

This work presents the first real-time matting technique for natural images and videos. The proposed approach is based on the key observation that pixels in a small neighborhood tend to have highly similar values for (α, F, B) triplets. Thus, a significant amount of computation used to obtain the matte for neighboring pixels is in fact redundant and can be safely eliminated. This work shows how to avoid such unnecessary computations by carefully distributing the work over neighboring pixels, which will then share their results. Since the operations performed by the pixels are now complementary, they can be performed independently and in parallel on modern GPUs. As a result, the proposed approach can generate high-quality mattes up to 100 times faster than previous techniques. The quality of these results have been confirmed by the independent image-matting benchmark by Rhemann *et al.* (2009). According to this benchmark, the proposed real-time technique ranked second among the current state-of-the-art techniques. Note, however, that our technique is up to two orders of magnitude faster, allowing for real-time alpha matting. Additionally, the proposed technique can be extended with an optimization step at some performance cost, resulting in the best results ever achieved in Rhemann *et al.*'s benchmark. This extended technique ranks first among all techniques (Section 6.1.2).

The main contribution of this work is the introduction of a new objective function for identifying good pairs of background and foreground samples (Equation 4.11). This new function takes into account spatial, photometric and probabilistic information extracted from the image. Such a function allows the presented approach to achieve high-quality results while still operating on a considerably small discrete search space.

Due to its speed, the proposed technique has the potential to enable new and exciting real-time applications that have not been previously possible. This work illustrates such potential (Chapter 6) by showing the first real-time alpha matting demonstration for natural-scene videos (Section 6.3.1), and by providing real-time feedback to users during interactive alpha-matting extraction sessions (Section 6.3.2).

Figure 1.1 shows an example of an alpha matte extracted with the proposed technique for a challenging example taken from the training dataset provided by Rhemann *et al.* (2009). The image on the top-left shows the original image, while the image on the top-right shows its corresponding trimap (provided in the dataset). The extracted alpha matte is shown on the bottom-left, and was computed in 0.043 seconds. The image on the

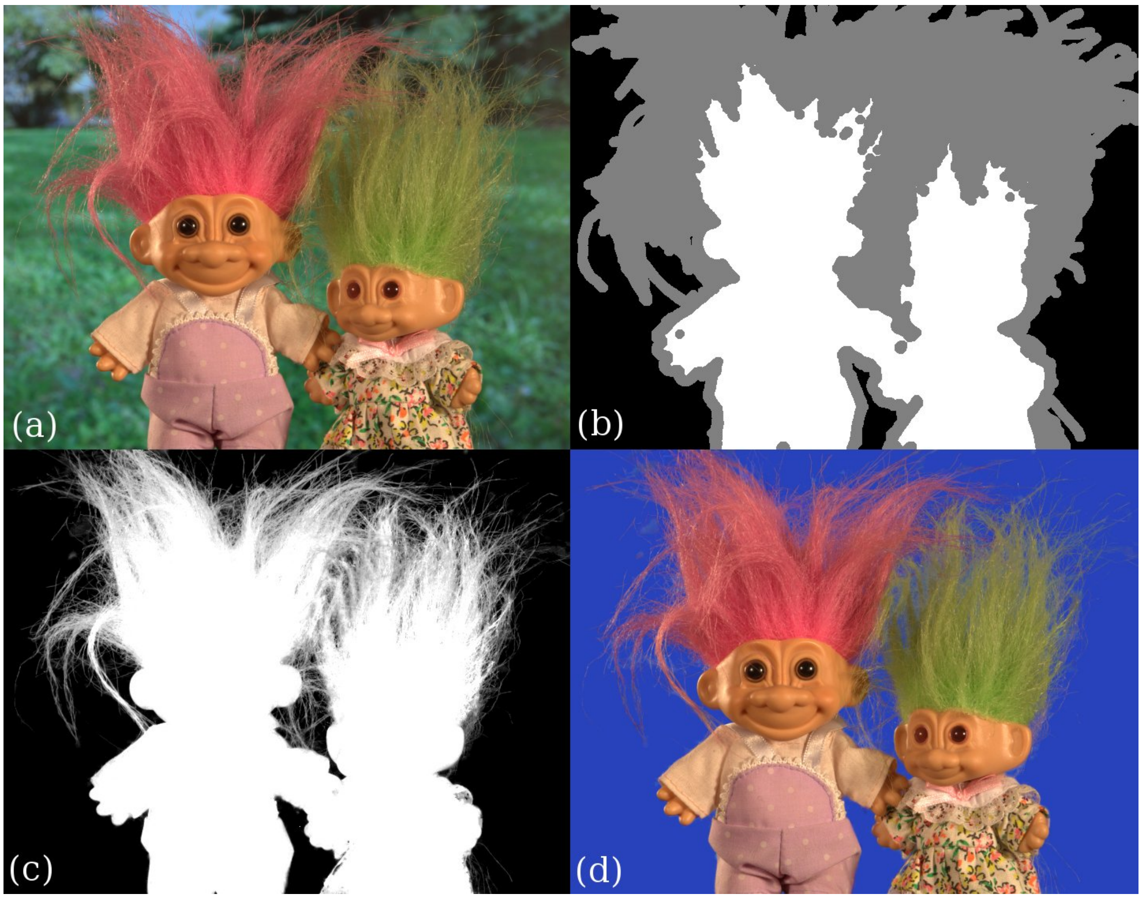


Figure 1.1: Example of alpha matte extraction and compositing using the proposed technique. (a) Image (800×563 pixels) from the training dataset provided by (RHEMANN et al., 2009). (b) Trimap provided in the dataset, with 40% of unknown pixels (gray represents unknown regions, while black and white represent known background and foreground regions, respectively). (c) Alpha matte computed with the proposed technique in 0.043 seconds. (d) Composite of the extracted foreground on a new background.

bottom-right shows the composite of the extracted foreground on a new background. The original image contains 800×563 pixels, and its trimap contains 40% of unknown pixels.

1.1 Structure of this work

The remaining of this work is organized as follows: Chapter 2 categorizes common approaches to constrain and solve the matting problem; Chapter 3 discusses related work; Chapter 4 presents the proposed technique for real-time alpha matting; Chapter 5 shows how to extend the proposed technique with an optimization step; Chapter 6 compares the results obtained by the proposed technique with the state-of-the-art and illustrates potential applications for real-time alpha matting; Finally, Chapter 7 summarizes this work and lists some paths for future work.

2 SOLVING THE MATTING PROBLEM

To accurately extract foreground objects from images and videos, additional constraints need to be defined. Such constraints limit the valid solutions for Equation 1.1, making the problem of solving the matte simpler and guaranteeing a meaningful final result. Constraints for the matting problem can be divided in three main categories, according to their origin: *user supplied constraints*, *assumption constraints*, and *additional information constraints*.

User supplied constraints are those that arise from user interaction. The most common form of user constraint is the manual classification of some pixels as either belonging to the foreground or belonging to the background. This is equivalent to manually fixing the values of α in Equation 1.1: $\alpha = 1$ for known foreground pixels and $\alpha = 0$ for known background pixels.

Assumption constraints arise from the core concepts used to solve the matting problem. They define the strengths and weaknesses of a technique, and usually differ among techniques. The weaker the assumption, the less general is the associated solution. An example of a weak assumption is the one made in *chroma keying* techniques (VLAHOS, 1964), where the background is assumed to have a constant color, usually green or blue. It is important to note that, if an input image breaks an assumption that constraints the problem, the output result is almost certainly incorrect.

Additional information constraints originate from sources other than the input image. This additional information can be used to reduce the size of the solution space, typicallly resulting in more accurate mattes. Unfortunately, the extra information does not come for free. It usually requires special conditions, such as custom designed capturing devices (MCGUIRE et al., 2005), flash and non-flash image pairs (SUN et al., 2006) or camera arrays (JOSHI; MATUSIK; AVIDAN, 2006). One example of such system can be seen in Figure 2.1.

2.1 The Trimap

One common form of constraint representation, which can originate from any of the previously defined categories, is the *trimap*. A trimap T expresses the segmentation of the input image into three disjoint sets (pixel regions): known foreground (T_f), known background (T_b) and unknown (T_u). This reduces the matting problem to estimating the values of α, F and B for pixels in the unknown region. And example of a trimap is shown in Figure 1.1b.

For a user, accurately specifying a trimap requires significant amounts of effort. To solve this problem, many matting approaches allow the user to specify only a small number of constrained pixels in the form of a few *scribbles* on top of the input image. This

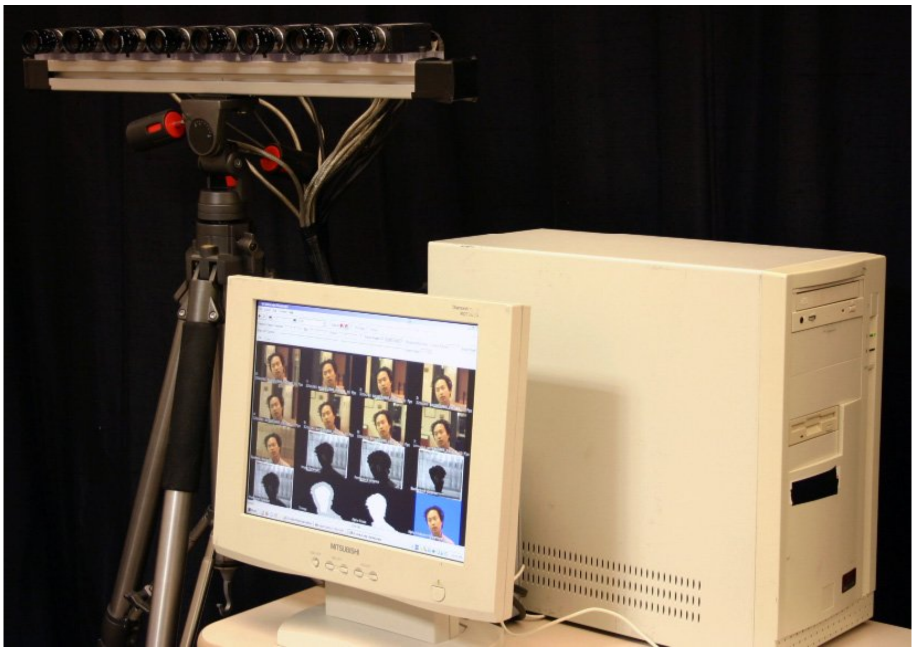


Figure 2.1: Example of specially designed device for constraining the matting problem: custom camera array described in (JOSHI; MATUSIK; AVIDAN, 2006).

significantly reduces the time and effort required from the user, but increases the work done by the matting algorithm, as the majority of the pixels will be marked as unknown. Alternatively, the trimap can be obtained in a more automatic fashion — *e.g.*, from a binary segmentation obtained using known techniques (BAI et al., 2009; BAI; SAPIRO, 2007; SUN et al., 2006; KIM et al., 2004; BOYKOV; KOLMOGOROV, 2003).

It is important to note that the trimap is a major factor influencing the quality of the final matte. The unknown region in the trimap should be as small as possible, meaning more known foreground and background information is available and less unknown variables need to be estimated.

2.2 Categorizing Matting Techniques

According to Wang and Cohen (2008), matting techniques can be classified into three categories according to the underlying method used for solving the matte, which can be based on *sampling*, *pixel affinities* or a combination of the two:

- **Sampling-based approaches** make the assumption that the true foreground and background colors of an unknown pixel can be explicitly estimated by analyzing nearby known pixels (*i.e.*, pixels in the trimap's known regions — T_f or T_b). These analyzed pixels are known as background or foreground *samples*. Once the foreground (F) and background (B) colors are determined, α can be easily calculated from the compositing Equation 1.1.

- **Affinity-based approaches** do not explicitly estimate foreground and background colors. Instead, they model the *matte gradient* across the image lattice by defining various affinities between neighboring pixels. These affinities are always defined in a small pixel neighborhood, where pixel correlations are usually strong.

- **Combined approaches** mix the two methodologies to achieve a good trade-off between accuracy and robustness. Sampling-based approaches work better when faced with distinct foreground and background colors, while affinity-based approaches are relatively insensitive to variations in user inputs, consistently generating smooth mattes.

Approaches involving affinities usually work by minimizing a quadratic energy function, which leads to large and sparse linear systems. Solving such big linear systems requires a considerable amount of computational effort, thus is not applicable to real-time alpha matting. For this reason, the proposed real-time matting technique is a sampling-based approach. Nonetheless, local pixel affinities are explored to guarantee the local smoothness of the matte, as described in Section 4.3.

Additionally, this work also presents a combined matting approach (Chapter 5), obtained by extending the proposed real-time alpha matting algorithm (Chapter 4) with an extra optimization step.

2.3 Evaluating Matting Results: Ground-Truth Mattes

In order to assess the quality of matting algorithms, their outputs (*i.e.,* alpha mattes) have to be compared against a *ground-truth* matte. This ground-truth matte represents the ideal alpha values that should be attained, or, in other words, the *true opacity value* of all pixels.

Ground-truth mattes are usually obtained in an extremely controlled environment by using the *triangulation* technique proposed by Smith and Blinn (1996). They showed that the acquisition of two (or more) images with a constant foreground but with different (and known) backgrounds exceeds all requirements to solve the matting problem. These additional images provide enough information to generate an over-constrained linear system, which can be solved using a least-squares framework.

3 RELATED WORK

3.1 Sampling-based Approaches

The first technique to use sampling for estimating the alpha values of unknown pixels was proposed by Mishima (1994). This technique assumes a blue background and approximates color distributions for the foreground and background colors as two triangular meshes (polyhedras) in color space. The alpha value of an unknown pixel is then computed by calculating its relative position to the two polyhedras. Following this idea, the KnockOut2 (2002) system computes the foreground (background) color for an unknown pixel as a weighted sum of nearby known foreground (background) colors. These values are then used to estimate the alpha value for each unknown pixel.

Earlier systems also include the work of Ruzon and Tomasi (2000), where alpha values are measured along a manifold connecting the boundaries of each object's color distribution. This approach assumes that the unknown region in the trimap is a narrow band around the foreground boundary. This assumption is weak and does not hold even in fairly simple images.

Bayesian matting (CHUANG et al., 2001) models local foreground and background color distributions with spatially-varying sets of Gaussians. This approaches improves on the work of Ruzon and Tomasi (2000) by using a continuously sliding window for neighborhood definition, which marches inward from the foreground and background regions. Nearby computed Fs, Bs and αs are also used to build these color distributions, so that every pixel in the neighbourhood will contribute to the foreground and background Gaussians. The matte is then solved using the *maximum a posteriori* (MAP) technique.

The iterative matting (WANG; COHEN, 2005) approach trains global Gaussian Mixture Models (GMMs) using known foreground and background colors. The alpha value of an unknown pixel is then estimated by sampling from all the Gaussians to cover all the possible foreground colors for the pixel. The Geodesic matting technique (BAI; SAPIRO, 2007) also models global foreground and background color distributions using Gaussian mixtures, but do so in Luv color space. Fast kernel density estimation methods (YANG et al., 2003) are used to reduce the computational complexity of constructing the foreground and background Probability Density Functions (PDFs).

3.2 Affinity-based Approaches

Affinity-based approaches solve for α independent of the estimation of foreground and background colors. The Poisson matting technique (SUN et al., 2004) observes that if the foreground and background colors are locally smooth, the gradient of the matte can be estimated from the gradient of the image

$$\nabla\alpha \approx \frac{1}{F-B}\nabla I \tag{3.1}$$

where $\nabla = (\frac{\partial}{\partial x}, \frac{\partial}{\partial y})$ is the gradient operator. In other words, the matte gradient is proportional to the image gradient. To find the alpha values, $F - B$ is estimated by simply choosing the nearest foreground and background colors for each unknown pixel. The matte is then found by solving for a function (α) whose gradient matches the estimated $\nabla\alpha$. For this, Poisson equations are used, with Dirichlet boundary condition defined by the trimap.

The Random Walks method of Grady *et al.* (GRADY et al., 2005) propagates user constraints to the entire image by minimizing a quadratic cost function. The alpha value for an unknown pixel is estimated as the probability that a random walker starting from this location will reach a pixel in the foreground before striking a pixel in the background (when biased to avoid crossing the foreground-background boundary). These probabilities can be calculated by solving a single system of linear equations.

The Closed-form matting technique from Levin *et al.* (2008) similarly solves the matte by minimizing a cost function derived from careful analysis of the matting problem. The assumption made in this approach is that each foreground and background color is a linear mixture of two colors over a small 3×3 window around each pixel — this is referred to as the *color line model*. Furthermore, Levin *et al.* (2008) showed that F and B can be analytically eliminated from the cost function, yielding a quadratic cost in only α. Rhemann *et al.* (2009) further improved on this work by deriving new closed form expressions for situations where the color line model does not hold.

3.3 Combined Approaches

The recent Robust matting approach of Wang and Cohen (WANG; COHEN, 2007) uses an initial sampling step to adapt an energy function which is then minimized using random walks. The central idea is to collect a large number of foreground and background samples in a neighbourhood close to an unknown pixel p. These samples are considered candidates for estimating the final alpha value of p. Their assumption is that the true foreground and background colors should be close to the ones of some of the collected samples. The authors discuss a way to calculate the confidence of the collected samples for each pixel. This value is used in the optimization procedure, where only information from high confidence pixels is used. This is motivated by the fact that color sampling will not always be reliable for all pixels, thus the alpha value for pixels with low confidence should rely more on the α-propagation induced by the optimization process.

The work of Rhemann *et al.*(RHEMANN; ROTHER; GELAUTZ, 2008) improves on this idea by proposing new weights for the confidence metric. Furthermore, they improve the search for suitable foreground samples by assuming that the foreground object is spatially connected.

3.4 Interactive Matting

Interactive alpha matting of images is typically performed using a two-step iterative process: first, the user refines the needed constraints (*trimap* or *scribbles*), which will then be used for matte generation in a subsequent step. This process is repeated until the user is satisfied with the quality of the matte. As a consequence, any delays between successive evaluations of the first step are enough to make this a time-consuming and tedious task. The system proposed by Wang *et al.* (WANG; AGRAWALA; COHEN, 2007) tries to solve this problem by noting that the user modifies the system constraints in a localized fashion. Therefore, the matte only needs to be (re)computed for a small portion of the image at a time. Nevertheless, the user has the perception of real-time computation, as the matte processing is interleaved with the tracing, by the user, of the foreground element boundary. However, as noted by Rhemann *et al.* (RHEMANN et al., 2008), the disadvantage of this tool is that long or complex boundaries are monotonous and time-consuming to trace.

3.5 Video Matting

For segmentation and matting of videos, Bai and Sapiro (BAI; SAPIRO, 2007) use the geodesic distance — based on the shortest path on a weighted graph — to interactively make soft segmentation and matting of images and videos. The recent work by Bai *et al.* (BAI et al., 2009) uses local classifiers to propagate a user defined segmentation across time. They further extend the work from (LEVIN; LISCHINSKI; WEISS, 2008) by adding a temporal coherence term to the cost function for generating mattes for offline video sequences. None of these techniques, however, are suitable for real-time alpha-matting of videos.

4 REAL-TIME ALPHA MATTING

The proposed method for real-time alpha matting consists of four steps and takes as input an image I (or video sequence) and its corresponding trimap(s). The first step performs some *expansion of known regions*, where “known foreground" and “known background" regions in the trimap are extrapolated into the “unknown" region. The second and third steps take care of *sample selection*, assuring that each pixel in the unknown region selects the best pair of foreground and background samples among the ones available. The final step guarantees the *local smoothness* of the matte while maintaining its distinct features. The following sub-sections present the details of each of these steps.

4.1 Expansion of Known Regions

A trimap T segments an input image (or video frame) into three non-overlapping pixel regions: known foreground (T_f), known background (T_b) and unknown (T_u). The idea behind expanding known regions is to exploit the affinity of neighboring pixels to reduce the size of the unknown region. Thus, let $D_{image}(p,q)$ and $D_{color}(p,q)$ be, respectively, the image-space and color-space distances between two pixels p and q. The expansion process consists of checking for each pixel $p \in T_u$ if there exists a pixel $q \in T_r$ ($r = \{f, b\}$) such that $D_{image}(p,q) \leq k_i$, $D_{color}(p,q) \leq k_c$, and $D_{image}(p,q)$ is minimal for p. In such a case, pixel p is labeled as belonging to region T_r based on its affinity to pixel $q \in T_r$. The value of the parameter k_i depends on the image size (larger images require larger values of k_i). We found that $k_i = 10$ pixels and $k_c = 5/256$ units (measured as Euclidean distance in the RGB color space) produce good results for typical images.

4.2 Sample Selection

For each remaining pixel $p \in T_u$, our goal is to find an (α, F, B) triplet that better models p. For this, a sampling strategy inspired by the work of Wang and Cohen (WANG; COHEN, 2007) is used, but differs from theirs in some fundamental aspects. For any given pixel $p \in T_u$, Wang and Cohen’s idea is to collect a large number of foreground and background samples in a neighborhood around p. Such samples are considered as candidates for estimating the alpha value of p. Their assumption is that the true foreground and background colors should be close to the ones of some of the collected samples. This is a reasonable assumption when the initial sample set is large. For instance, in their approach, Wang and Cohen analyze 400 pairs of foreground and background colors for each pixel p (WANG; COHEN, 2007). Unfortunately, the use of larger sample sets requires a significant amount of computation in order to find good pairs of foreground and background colors. The next sections will show that this computational cost can be significantly reduced by exploiting affinities among neighboring pixels. Furthermore, a new and improved metric for electing the best samples is presented, which takes into account image parameters that were not considered in the sample-selection process described in (WANG; COHEN, 2007).

The proposed approach is based on the fundamental observation that pixels in a small neighborhood often have highly similar values for their true (α, F, B) triplets. From this, it follows that: (1) the initial collection of samples gathered by nearby pixels differ only by a small number of elements; and (2) close-by pixels usually select the same or very similar pairs for their best foreground and background colors. This brings the conclusion that performing the alpha matte computation for each pixel independently of its neighbors results in a large amount of redundant work that can be safely avoided without compromising the quality of the matte. In fact, as demonstrated in Chapter 6, it is possible to achieve speedups of up to two orders of magnitude while still obtaining high-quality results.

To minimize the amount of redundant work while leveraging the high affinity among neighboring pixels, the proposed approach separates the sample-selection procedure in two steps:

1. **Sample Gathering**: at this stage, each pixel $p \in T_u$ selects the best pair from a small set of samples from its neighborhood, gathered in a manner that maximizes the chances that sample sets from neighboring pixels are disjoint;

2. **Sample Refinement**: in this second stage, each pixel $p \in T_u$ analyzes the choices made by its closest neighbors in T_u, and then selects one of these choices as its best pair.

As each pixel is trying to find candidates for both its true foreground and background colors[1], samples must be evaluated in pairs. These *sample-pairs* represent candidates for the true color pair (F, B) of an unknown pixel. Thus, each pixel $p \in T_u$ gathers at most k_g background and k_g foreground samples, resulting in at most k_g^2 tested pairs of background and foreground samples during the *gathering phase*. In the *sample-refinement phase*, each pixel p analyzes the choices (without recombining them) of its (at most) k_r spatially closest pixels also in T_u. Thus, while in practice p performs a total of $k_g^2 + k_r$ pair evaluations, due to the affinity among neighbor pixels, this is roughly equivalent to performing a total of $k_g^2 \times k_r$ pair comparisons. According to our experience, values of $k_g = 4$ and $k_r = 40$ produce very good results. For these values, the actual number of performed pair comparisons is 56 (*i.e.*, 16 + 40), while its net effect approximates a total of 640 (*i.e.*, 16 * 40) comparisons.

Sections 4.2.1 and 4.2.2 present the details of the sample gathering and sample refinement sub-steps.

4.2.1 Sample Gathering

In the sample-gathering stage, each pixel $p \in T_u$ looks for possible foreground and background samples along k_g line segments starting at p. These segments divide the plane of the image into k_g disjoint sectors containing equal planar angles (Figure 4.1). The slope of the first line segment associated to p is defined by an initial orientation $\theta \in \left[0, \frac{\pi}{2}\right]$

[1]We need both F and B to estimate α

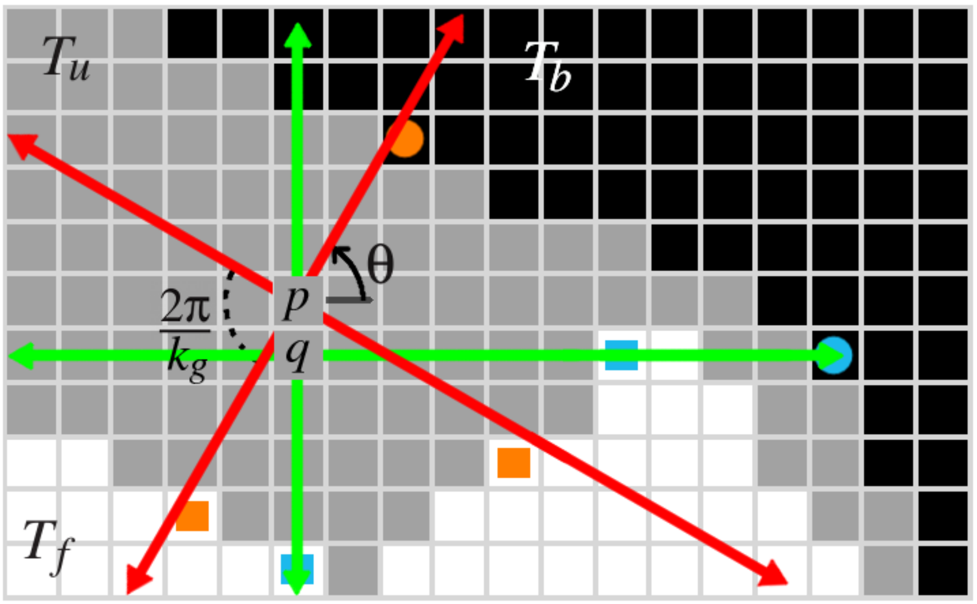


Figure 4.1: The red line segments starting at p define the paths (sets of pixels) for searching for background and foreground pixels. Each segment will contribute at most one foreground and at most one background pixels (the first ones found when moving from p outwards along the line segments). Selected samples for p are marked in orange. Pixel q explores a different path (in green) when searching for its own background and foreground samples (marked in cyan). Foreground samples are designated by squares, background samples by circles.

measured with respect to the horizontal. Such an angle takes a different value for each pixel $q \in T_u$ in a 3×3 window (Figure 4.1). The orientation of the other segments is given by an angular increment given by $\theta_{inc} = \frac{2\pi}{k_g}$. Starting from p and following a particular line segment yields at most one background and at most one foreground sample — the ones closer to p along the segment. Thus, p must find its best pair among, at most, k_g^2 sample pairs.

According to Wang and Cohen (WANG; COHEN, 2007), good pairs of samples (*i.e.*, candidates for background and foreground colors) can explain the color C_p of pixel $p \in T_u$ as a linear combination of themselves. They further argue that the colors of a good pair of samples should also be widely separated in color space and try to enforce this in their objective function. While their observation regarding linear interpolation is intuitively sound according to Equation 1.1, their second statement is oversimplifying and does not address the fundamental issues involved in the computation of a matte. As such, it does not represent a good measure for comparison of candidate pairs. Figure 4.2a illustrates an example where this oversimplification leads to a wrong decision in selecting the best pair of samples.

This work presents a new objective function that combines photometric, spatial, and probabilistic elements to select good quality sample pairs. The proposed approach is the first to comprehensively consider all these aspects. Thus, let $\mathbf{f}_i$ and $\mathbf{b}_j$ be a pair of foreground and background samples, whose colors are F^i and B^j, respectively. Next, we will derive an optimization function (Equation 4.11) for identifying the best sample-pair for each pixel $p \in T_u$. Before describing this final function (Equation 4.11), its required building blocks will be presented.

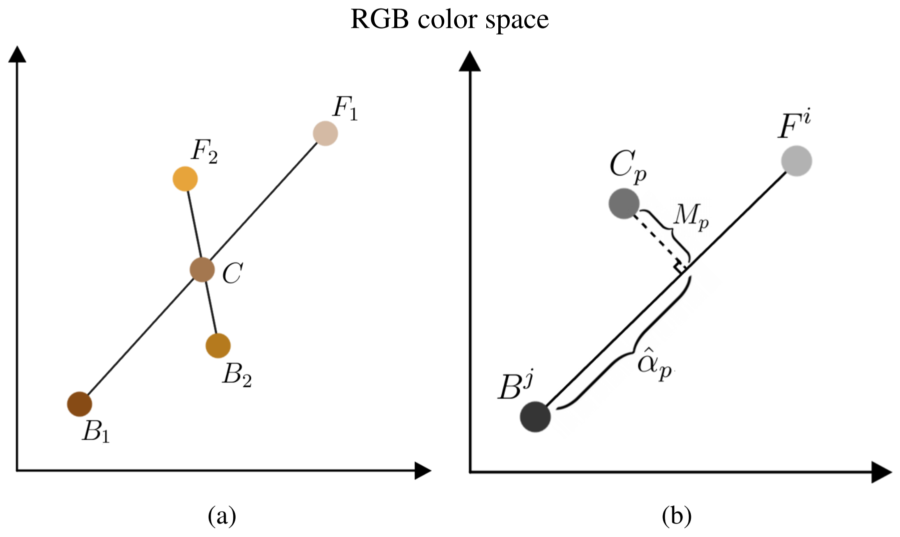


Figure 4.2: (a) Example where the assumption made by Wang and Cohen (2007) — that good pair of samples are widely separated in color space — does not hold. Here, there is no guarantee that the pair (F_1, B_1) more accurately represents, when compared to (F_2, B_2), the true foreground and background colors of C. (b) Illustration of the chromatic distortion M_p modeled by Equation 4.1.

Minimization of chromatic distortion Similar to what has been described by Wang and Cohen (WANG; COHEN, 2007), the proposed approach favors the selection of pairs of foreground and background colors that can model the color of pixel p as a linear combination of themselves. This is modeled by the *chromatic distortion* M_p (Equation 4.1), whose value should be small for a good pair of candidate colors. Unlike their approach, however, distant color pairs are not favored (*i.e.*, they enforce the wide separation of foreground and background colors with an additional denominator in Equation 4.1, dividing it by $\|F^i - B^j\|$, which we do not do).

$$M_p(F^i, B^j) = \left\|C_p - (\hat{\alpha}_p F^i + (1 - \hat{\alpha}_p)B^j)\right\| \tag{4.1}$$

where C_p is the color of p, and $\hat{\alpha}_p$ is the estimated alpha value for p, obtained as the color space projection of C_p onto the line defined by F^i and B^j (Figure 4.2b). Although a small $M_p(F^i, B^j)$ is necessary for accurately representing the alpha value of p, it is not a sufficient condition to elect a good sample pair. For this reason, we propose a new color metric derived from two previously made observations: (i) Levin *et al.* (LEVIN; LISCHINSKI; WEISS, 2008) showed that small pixel neighborhoods tend to form locally linear clusters in color space — this is particularly true for small windows located over image edges (Figures 4.3a and 4.3b); and (ii) Sun *et al.* (SUN et al., 2004) showed that if the foreground and background gradients ∇F and ∇B are relatively small compared with $\nabla\alpha$, than the matte gradient $\nabla\alpha$ is directly proportional to the image gradient ∇I — in other words, color variations in I are the effect of discontinuities in α.

Based on these observations, one concludes that in the unknown region of the trimap — where $\nabla\alpha$ is potentially very large — the locally-linear color variations observed by Levin *et al.* (2008) are primarily caused by variations in α. Thus, all colors from pixels in a small local window are situated along the line spanned by the true foreground and background colors F and B. This means that a good sample pair should also minimize the least squares residual defined by the *neighborhood affinity* term:

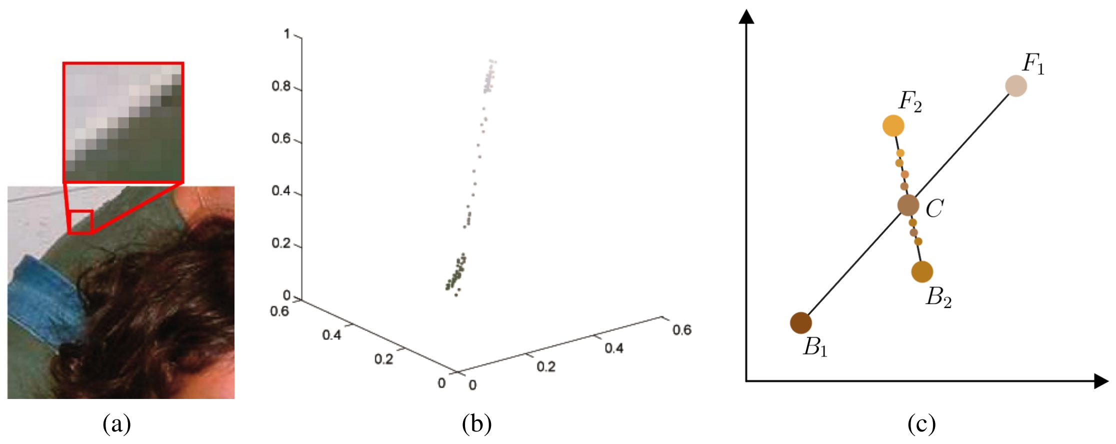


Figure 4.3: (a) Local patch from a real image and the (b) RGB plot of its color distribution. From (LEVIN; LISCHINSKI; WEISS, 2008). (c) How the neighborhood affinity term N from Equation 4.2 helps in the selection of the best sample pair. Colors from pixels in the neighborhood Ω_p are represented by small circles. It can be easily seen that $N(F_2, B_2) < N(F_1, B_1)$, thus the pair (F_2, B_2) more accurately represents the true foreground and background colors of C.

$$N(\mathbf{f}_i, \mathbf{b}_j) = \sum_{q \in \Omega_p} M_q(F^i, B^j)^2 \tag{4.2}$$

where Ω_p is the pixel neighborhood of p, consisting of all pixels in a 3×3 window centered on p, and M_q is the operator defined in Equation 4.1, evaluated at the pixel q. An example is shown in Figure 4.3c.

In addition to color space information, image space statistics should play a key role in identifying good pair of samples. These image parameters were not considered in the sample selection process of (WANG; COHEN, 2007), where only color space metrics were used. Thus, let $D_p(\mathbf{s}) = D_{image}(p, s) = \|\mathbf{s} - p\|$ be the image space distance from a sample s to the current pixel p. We define the *energy* $E_p(\mathbf{s})$ to reach a foreground or background sample s from the current pixel p as the squared path integral of ∇I along the image space line segment L connecting p and s:

$$E_p(\mathbf{s}) \;=\; \int_L \|\nabla I \cdot d\mathbf{r}\|^2 \;=\; \int_p^{\mathbf{s}} \left\| \nabla I \cdot \left(\frac{\mathbf{s} - p}{\|\mathbf{s} - p\|} \right) \right\|^2 \tag{4.3}$$

Notice that the energy E is directly proportional to the projection length of ∇I onto the normalized direction of integration s$-p$. Thus, if the linear path from s to p crosses image regions where $\|\nabla I\|$ is large — *e.g.*, image edges — greater energy will be required to reach s (Figure 4.4).

An estimate of the probabilities of p belonging to the foreground, according to the energy function $E_p(\mathbf{s})$ and image space distance $D_p(\mathbf{s})$, can be obtained as:

$$P_E^f(\mathbf{f}_i, \mathbf{b}_j) \;=\; \frac{E_p(\mathbf{b}_j)}{E_p(\mathbf{f}_i) + E_p(\mathbf{b}_j)} \tag{4.4}$$

$$P_D^f(\mathbf{f}_i, \mathbf{b}_j) \;=\; \frac{D_p(\mathbf{b}_j)}{D_p(\mathbf{f}_i) + D_p(\mathbf{b}_j)} \tag{4.5}$$

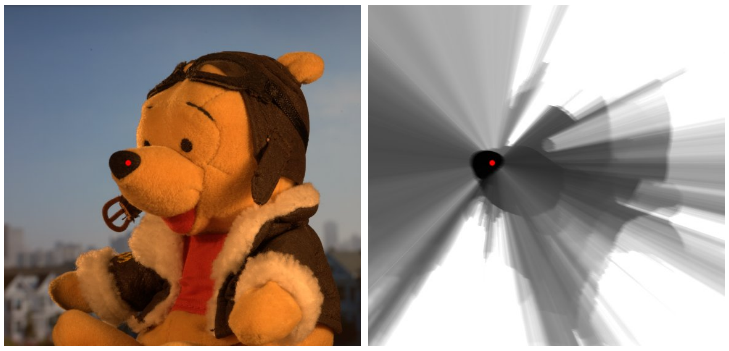


Figure 4.4: Illustration of the energy E required to reach s (marked in red) from all locations in the image. On the right, pixels with colors closer to white require more energy to reach s. Notice how edges on the left image have a direct correspondence to variations in the energy function on the right.

For example, if the energy $E_p(\mathbf{f}_i)$ required to reach the foreground sample $\mathbf{f}_i$ is much lower than $E_p(\mathbf{b}_j)$, $P_E^f(\mathbf{f}_i, \mathbf{b}_j)$ will be close to one — *i.e.*, pixel p has a high probability of belonging to the foreground. This is analogous for P_D^f.

These probabilities are combined using *relevance weights* ω_E and ω_D associated with $E_p(\mathbf{s})$ and $D_p(\mathbf{s})$, respectively. Such weights express the relevance of $E_p(\mathbf{s})$ ($D_p(\mathbf{s})$) relative to the energy (distance) of all observed samples:

$$\begin{aligned}\omega_E(\mathbf{f}_i, \mathbf{b}_j) = exp\ \Bigg\{ & -\frac{E_p(\mathbf{f}_i) - min_i(E_p(\mathbf{f}_i))}{max_i(E_p(\mathbf{f}_i)) - min_i(E_p(\mathbf{f}_i))} \\ & -\frac{E_p(\mathbf{b}_j) - min_j(E_p(\mathbf{b}_j))}{max_j(E_p(\mathbf{b}_j)) - min_j(E_p(\mathbf{b}_j))} \Bigg\}\end{aligned} \tag{4.6}$$

$$\begin{aligned}\omega_D(\mathbf{f}_i, \mathbf{b}_j) = exp\ \Bigg\{ & -\frac{D_p(\mathbf{f}_i) - min_i(D_p(\mathbf{f}_i))}{max_i(D_p(\mathbf{f}_i)) - min_i(D_p(\mathbf{f}_i))} \\ & -\frac{D_p(\mathbf{b}_j) - min_j(D_p(\mathbf{b}_j))}{max_j(D_p(\mathbf{b}_j)) - min_j(D_p(\mathbf{b}_j))} \Bigg\}\end{aligned} \tag{4.7}$$

Finally, let $P_g^f(\mathbf{f}_i, \mathbf{b}_j)$ be the probability of p belonging to the foreground according to the information available in the gathering stage. $P_g^f(\mathbf{f}_i, \mathbf{b}_j)$ is defined as the weighted average of $P_E^f(\mathbf{f}_i, \mathbf{b}_j)$ and $P_D^f(\mathbf{f}_i, \mathbf{b}_j)$:

$$\omega(\mathbf{f}_i, \mathbf{b}_j) = \frac{\omega_E(\mathbf{f}_i, \mathbf{b}_j)}{\omega_E(\mathbf{f}_i, \mathbf{b}_j) + \omega_D(\mathbf{f}_i, \mathbf{b}_j)} \tag{4.8}$$

$$P_g^f(\mathbf{f}_i, \mathbf{b}_j) = \omega(\mathbf{f}_i, \mathbf{b}_j)\ P_E^f(\mathbf{f}_i, \mathbf{b}_j) + (1 - \omega(\mathbf{f}_i, \mathbf{b}_j))\ P_D^f(\mathbf{f}_i, \mathbf{b}_j) \tag{4.9}$$

Intuitively, we want the computed alpha matte value $\hat{\alpha}_p$ (in Equation 4.1) to correlate with the probability $P_g^f(\mathbf{f}_i, \mathbf{b}_j)$ of pixel p belonging to the foreground. This is enforced by minimizing the function $A(\mathbf{f}_i, \mathbf{b}_j)$ (Equation 4.10). Indeed, for a given pair of samples $(\mathbf{f}_i, \mathbf{b}_j)$, when $P_g^f = 0$, $A = \hat{\alpha}_p$. Thus, minimizing A also minimizes the value of $\hat{\alpha}_p$. Likewise, when $P_g^f = 1$, $A = (1 - \hat{\alpha}_p)$, so minimizing A maximizes $\hat{\alpha}_p$. Finally, if $P_g^f = 0.5$, $A = 0.5$, and the value of $\hat{\alpha}_p$ has no effect on the minimization. Function $A(\mathbf{f}_i, \mathbf{b}_j)$ will be later used as one of the terms of Equation 4.11, which identifies good pairs of background and foreground samples for pixel p.

$$A(\mathbf{f}_i, \mathbf{b}_j) = P_g^f(\mathbf{f}_i, \mathbf{b}_j) + (1 - 2P_g^f(\mathbf{f}_i, \mathbf{b}_j))\ \hat{\alpha}_p \tag{4.10}$$

The resulting objective function that combines photometric and spatial affinity, as well as probabilistic information for selecting good pairs of background and foreground samples can be expressed as

$$g(\mathbf{f}_i, \mathbf{b}_j) = N(\mathbf{f}_i, \mathbf{b}_j)\ A(\mathbf{f}_i, \mathbf{b}_j)\ D_p(\mathbf{f}_i)\ D_p(\mathbf{b}_i) \tag{4.11}$$

Here, $N(\mathbf{f}_i, \mathbf{b}_j)$ minimizes chromatic distortion in the 3×3 neighborhood around p. $A(\mathbf{f}_i, \mathbf{b}_j)$ enforces that the computed alpha matte values correlate with the probability of pixel p belonging to the foreground (according to $\mathbf{f}_i$ and $\mathbf{b}_j$). $D_p(\mathbf{f}_i)$ and $D_p(\mathbf{b}_i)$ enforce the spatial affinity criterium: the background and foreground samples should be as close as possible to p. Thus, the best pair of foreground and background samples $(\hat{\mathbf{f}}_p, \hat{\mathbf{b}}_p)$ for pixel p is obtained by evaluating $g(\mathbf{f}_i, \mathbf{b}_j)$ for all possible sample-pairs:

$$(\hat{\mathbf{f}}_p, \hat{\mathbf{b}}_p) = \operatorname{argmin}_{\mathbf{f},\mathbf{b}}\ g(\mathbf{f}_i, \mathbf{b}_j) \tag{4.12}$$

Let (F_g^p, B_g^p) be the corresponding colors of the best pair $(\hat{\mathbf{f}}_p, \hat{\mathbf{b}}_p)$ for pixel p. We then compute two variances σ_f^2 and σ_b^2 as:

$$\begin{aligned} \sigma_f^2 &= \tfrac{1}{N} \textstyle\sum_{q \in \Omega_f} \left\| F_g^p - C_q \right\|^2 \\ \sigma_b^2 &= \tfrac{1}{N} \textstyle\sum_{q \in \Omega_b} \left\| B_g^p - C_q \right\|^2 \end{aligned} \tag{4.13}$$

where Ω_f and Ω_b are $3{\times}3$ pixel neighborhoods centered at $\hat{\mathbf{f}}_p$ and $\hat{\mathbf{b}}_p$, respectively, and $N = 9$. Such variances measure how much the colors of the selected background and foreground samples deviate from their own neighborhoods. Intuitively, the smaller these variances, the bigger the confidence that the selected samples are good representatives for their own neighborhoods, as opposed to being outliers.

The output of the sample gathering stage is a tuple $\tau_p = (F_g^p,\ B_g^p,\ \sigma_f^2,\ \sigma_b^2)$ for each pixel $p \in T_u$.

4.2.2 Sample Refinement

For small values of k_g, the total number of samples analyzed by any given pixel $p \in T_u$ during the sample gathering state is often not enough to reliably estimate either an alpha value or the true foreground and background colors. To address this issue, a more extensive search is performed by sharing the best results obtained by all pixels in T_u.

At this stage of the sample-selection process, each pixel p will compare its own choice of best sample-pair with the choices of its (at most) k_r spatially closest pixels $q \in T_u$. The three tuples with the lowest values of $M_p(F_g^q, B_g^q)$ will then be averaged to create a new tuple $\tilde{\tau}_p = (\tilde{F}_g^p,\ \tilde{B}_g^p,\ \tilde{\sigma}_f^2,\ \tilde{\sigma}_b^2)$ for p. The purpose of this averaging is to reduce the occurrence of noise in the resulting alpha matte. This procedure is supported by the observation that neighbor pixels tend to have similar values of alpha, as well as background and foreground colors (*i.e.*, neighbor pixels tend to present high affinity). Therefore, by averaging the best few values in a given neighborhood, the occurrence of noise is reduced. The confidence measure of this new sample-pair is computed as:

$$f(\tilde{F}_g^p, \tilde{B}_g^p) = exp\left\{ -\lambda\ M_p(\tilde{F}_g^p, \tilde{B}_g^p) \right\} \tag{4.14}$$

where λ[†] models the rate of decrease of f. Thus, the confidence measure modeled by Equation 4.14 decreases fast (but not too fast) as the foreground and background colors

[†] $\lambda = 10$ in our implementation

$\tilde{F}_g^p$ and $\tilde{B}_g^p$ fail to properly model the color C_p of p. The output of the sample-refinement stage for pixel $p \in T_u$ is another tuple $\kappa_p = (F_r^p,\ B_r^p,\ \alpha_r^p,\ f_r^p)$, where:

$$F_r^p = \begin{cases} C_p & \text{if } \left\|C_p - \tilde{F}_g^p\right\|^2 \leq \tilde{\sigma}_f^2 \\ \tilde{F}_g^p & \text{otherwise} \end{cases} \tag{4.15}$$

$$B_r^p = \begin{cases} C_p & \text{if } \left\|C_p - \tilde{B}_g^p\right\|^2 \leq \tilde{\sigma}_b^2 \\ \tilde{B}_g^p & \text{otherwise} \end{cases} \tag{4.16}$$

$$\alpha_r^p = \frac{(C_p - B_r^p) \cdot (F_r^p - B_r^p)}{\|F_r^p - B_r^p\|^2} \tag{4.17}$$

$$f_r^p = f(F_r^p, B_r^p) \tag{4.18}$$

Here, the subscript r represents quantities computed in the sample-refinement stage. The intuition behind the computation of F_r^p is that if the color C_p of pixel p is sufficiently close to the average color $\tilde{F}_g^p$ of the best three foreground samples computed during the gathering stage, then F_r^p should be taken as C_p, thus keeping the original color. The case for B_r^p is similar. The alpha value α_r^p is computed as the relative length of the projection of vector $(C_p - B_r^p)$ onto vector $(F_r^p - B_r^p)$, defined by the computed foreground and background colors. Thus, α_r^p represents the opacity of the foreground sample. Finally, f_r^p expresses the confidence of p in its candidate foreground and background colors F_r^p and B_r^p, modeled by Equation 4.14. For completeness, output tuples for pixels outside the unknown region are also defined; thus, for pixels $v \in T_f \cup T_b$, we have $\kappa_v = (F_r^v = C_v,\ B_r^v = C_v,\ \alpha_r^v,\ f_r^v = 1)$, where:

$$\alpha_r^v = \begin{cases} 0 & \text{if } r \in T_b \\ 1 & \text{if } r \in T_f \end{cases}$$

4.3 Local Smoothness

Although the sample-selection process takes into account affinities among localized groups of pixels, this is not enough to prevent discontinuities in the resulting matte. Thus, an additional step is used to ensure the local smoothness of the final alpha values, while maintaining its distinct features. This is achieved by computing, for each pixel $p \in T_u$, a weighted average of the tuples κ_q of the closest k_l[†] neighbors of p in image space. Such neighbors can come from either T_u, T_f, or T_b. Let Ψ_p be such a neighborhood for pixel p. The weights are defined in such way that details in the matte are preserved.

The final foreground and background colors F^p and B^p of p are computed as:

$$W_c(p,q) = \begin{cases} G\left(D_{image}(p,q)\right)\, |\alpha_r^p - \alpha_r^q|\ f_r^q & \text{if } p \neq q \\ G\left(D_{image}(p,q)\right)\, f_r^q & \text{if } p = q \end{cases} \tag{4.19}$$

$$F^p = \frac{\sum_{q\in\Psi_p}\left[W_c(p,q)\ \alpha_r^q\ F_r^q\right]}{\sum_{q\in\Psi_p}\left[W_c(p,q)\ \alpha_r^q\right]} \tag{4.20}$$

$$B^p = \frac{\sum_{q\in\Psi_p}\left[W_c(p,q)\ (1-\alpha_r^q)\ B_r^q\right]}{\sum_{q\in\Psi_p}\left[W_c(p,q)\ (1-\alpha_r^q)\right]} \tag{4.21}$$

[†]A value of $k_l = 40$ is used in our implementation

where G is a normalized Gaussian function with variance $\sigma^2 = k_l/9\pi$[††] pixels. The weight $W_c(p,q)$ blends the foreground (background) colors of pixels p and q taking into account: (i) *Spatial affinity* – the colors of two pixels that are far apart in image space should not be averaged. This is modeled by the Gaussian function; (ii) *Difference in alpha values* – by minimizing $\|\nabla F\|$ and $\|\nabla B\|$ where the estimated $\|\nabla\hat{\alpha}\|$ is large, we make the final $\nabla\alpha \approx \nabla I$; (iii) *Confidence values* – pixels with low confidence in their foreground and background samples should not propagate their uncertainty.

The term α_r^q multiplying F_r^q in Equation 4.20 denotes the confidence of pixel q in its foreground color F_r^q. This confidence is directly proportional to α_r^q — *e.g.*, if $\alpha_r^q = 0$, q has *zero* confidence in its foreground color F_r^q, as Equation 4.17 yields zero for all values of F_r^q. Similarly, the term $(1-\alpha_r^q)$ multiplying B_r^q in Equation 4.21 denotes that the confidence of pixel q in its background color B_r^q is inversely proportional to α_r^q.

Having F^p and B^p, we can compute the final confidence f^p of pixel p in these final foreground and background colors. To do so, we first define in Equation 4.23 the *mean foreground-background distance* Z (in color space) for the neighborhood Ψ_p. This mean is weighted by $W_z(q)$ (Equation 4.22) which is directly proportional to the confidence f_r^q of q and is maximized for values of $\alpha_r^q = 0.5$ — where the confidence of q in both F_r^q and B_r^q is potentially maximal — while being zero for $\alpha^q = \{0,1\}$. Z will be used next to compute the final confidence f^p.

$$W_z(q) \;=\; f_r^q\;\alpha_r^q\;(1-\alpha_r^q) \tag{4.22}$$

$$Z(\Psi_p) \;=\; \frac{\sum_{q\in\Psi_p}\Big[W_z(q)\;\|F_r^q - B_r^q\|\Big]}{\sum_{q\in\Psi_p} W_z(q)} \tag{4.23}$$

The final confidence f^p of pixel p in its final foreground and background colors F^p and B^p is modeled by Equation 4.24. Here, the first term expresses the ratio of the distance $\|F^p - B^p\|$ to the mean foreground-background distance in the neighborhood Ψ_p (clamped to the range [0,1]). This ratio tries to detect pixels whose final foreground and background colors deviate from those in the neighborhood Ψ_p. The second term is analogous to Equation 4.14.

$$f^p \;=\; min\left(1, \frac{\|F^p - B^p\|}{Z(\Psi_p)}\right)\; exp\Big\{-\lambda\; M_p(F^p, B^p)\Big\} \tag{4.24}$$

Having F^p, B^p and f^p, we can now compute the final alpha value α_p of pixel p. In order to do so, we first define the *low frequency alpha* α_l^p (Equation 4.26) as the weighted average of alpha values in the neighborhood Ψ_p. The weights $W_\alpha(p,q)$ are proportional to the confidence f_r^q of q and inversely proportional to the image space distance of p and q. Additionally, greater weights are given for pixels lying in T_f or T_b (*i.e.*, known pixels).

$$W_\alpha(p,q) \;=\; f_r^q\; G\left(D_{image}(p,q)\right) + \delta(q \notin T_u) \tag{4.25}$$

$$\alpha_l^p \;=\; \frac{\sum_{q\in\Psi_p}\Big[W_\alpha(p,q)\;\alpha_r^q\Big]}{\sum_{q\in\Psi_p} W_\alpha(p,q)} \tag{4.26}$$

[††]The set Ψ_p of the closest k_l pixels to p approximately forms an image space circle with area k_l; thus, k_l can be expressed as $k_l = \pi r^2$. We want the farthest pixels in Ψ_p (with distance of r to p) to have weights close to zero in the Gaussian (*i.e.*, $r = 3\,\sigma$). Thus, $k_l = \pi r^2 = \pi(3\,\sigma)^2$, which solves to $\sigma^2 = k_l/9\pi$.

where δ is a boolean function returning 0 or 1, and G is the Gaussian function from Equation 4.19.

The final alpha value α^p for p is given by Equation 4.27. This equation blends the alpha value computed using F^p and B^p with the low frequency alpha α_l^p, with blending factor defined by the final confidence f^p. Thus, pixels with a low final confidence will accept alpha values from higher-confidence neighbors (modeled by α_l^p) to preserve local smoothness.

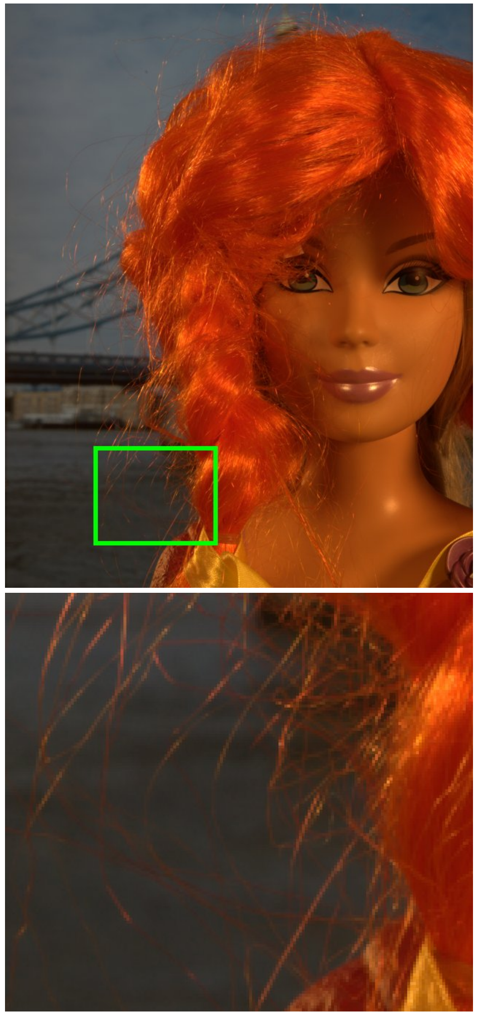
$$\alpha^p \quad = \quad f^p \, \frac{(C_p - B^p) \cdot (F^p - B^p)}{\|F^p - B^p\|^2} + (1 - f^p) \, \alpha_l^p \tag{4.27}$$

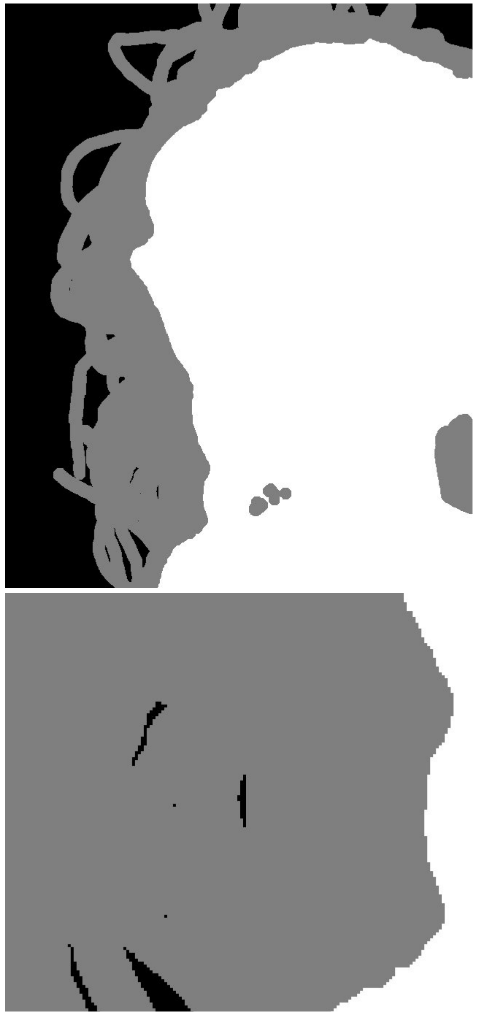
Finally, output of the proposed algorithm for the matting parameters of pixel $p \in T_u$ is given by the tuple $(F^p,\ B^p,\ \alpha^p)$ with an associated confidence value of f^p. For completeness, the matting parameters for pixels outside the unknown region are also defined. Thus, for pixels $q \in T_f$ we have the tuple $(F^q = C_q, B^q = C_q, \alpha^q = 1)$ with confidence $f^q = 1$, and for pixels $w \in T_b$ we have the tuple $(F^w = C_w, B^w = C_w, \alpha^w = 0)$ with confidence $f^w = 1$.

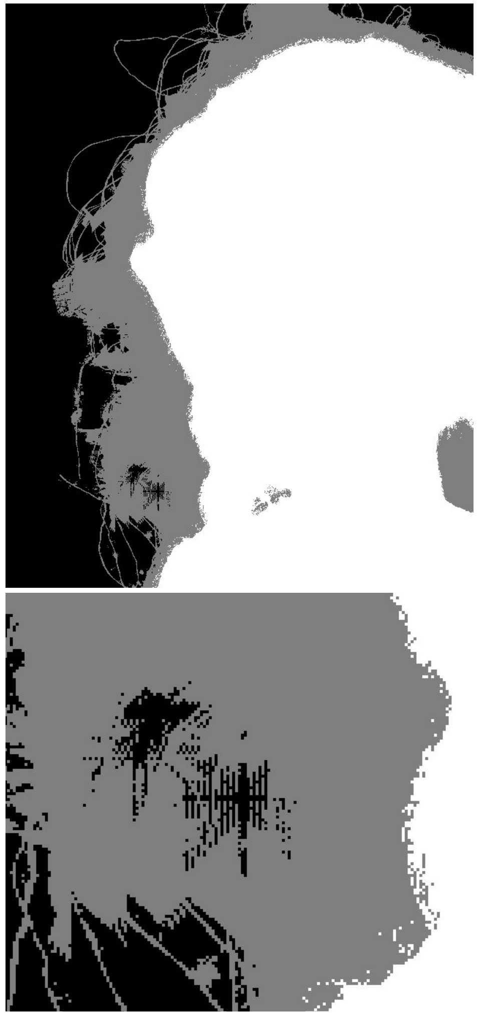
4.4 An Illustrated Example

This section provides a visual summary of the steps involved in the proposed real-time matting technique. Images (a) and (b) show the matting inputs. The first step expands known regions and generates the trimap in (c). The sequences d-f and g-i show the foreground and background colors estimated for each of the Gathering, Refinement and Smoothness steps. Pixels in black in images (d) and (g) did not find any suitable sample-pair in the initial Gathering stage. Notice, however, that in the refinement stage (images (e) and (f)), these pixels have found pairs through the analysis of candidates from their neighborhoods. The sequence j-l shows the estimated alpha values and (m) shows the foreground composed onto a new white background. Note the high-frequency noise in α_r^p (j), which is removed in the smoothness step (l) while preserving details. Images (n) and (o) show the foreground σ_f^2 and background σ_b^2 variances, while images (p) and (q) show the refinement and final confidences f_r^p and f^p.

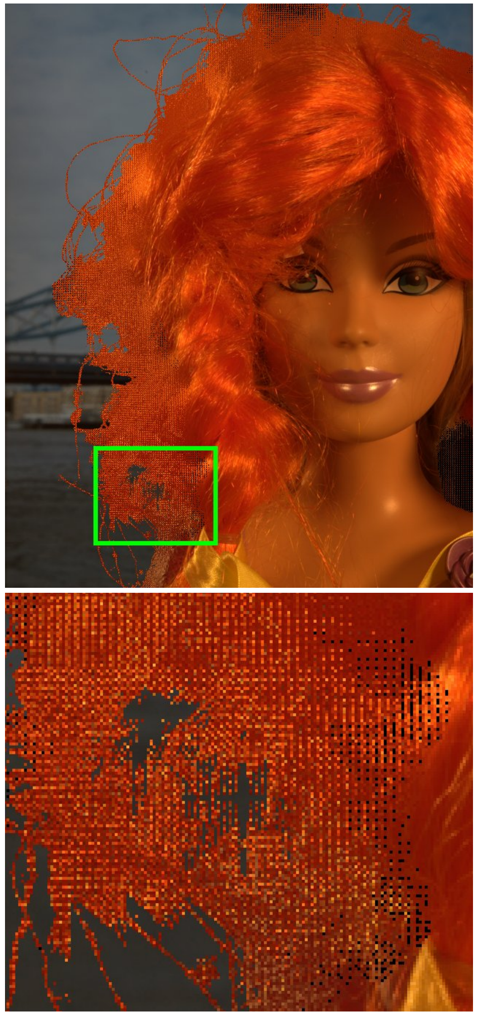
Input and Constraints

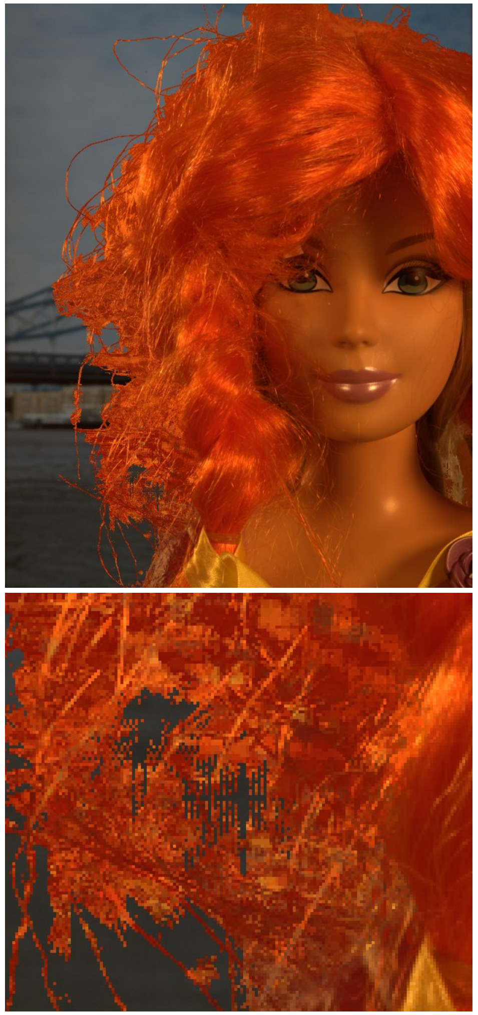

(a) Input image

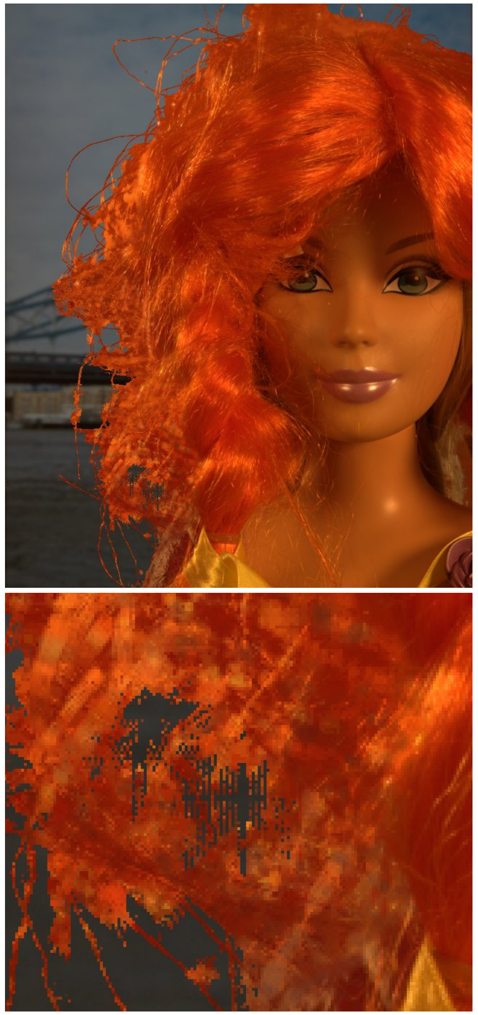

(b) Input trimap


(c) Expanded trimap

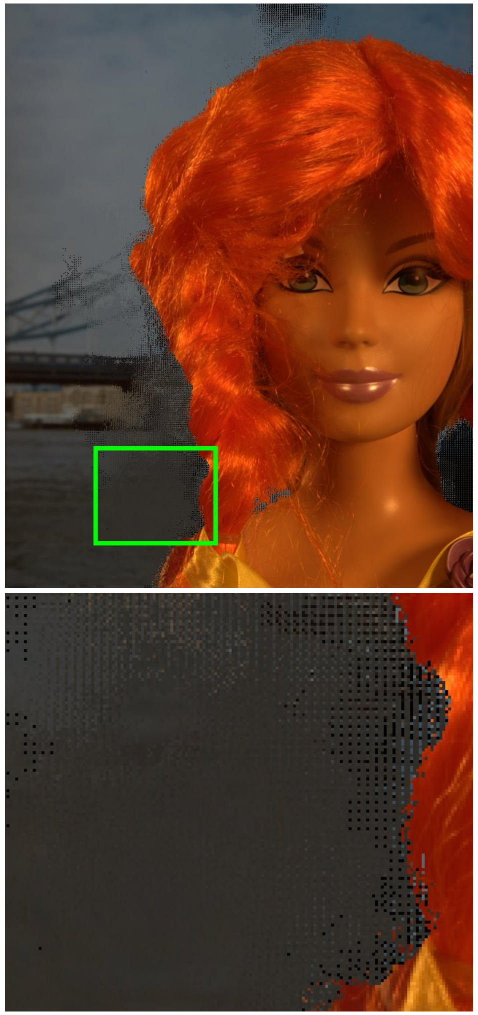
Foreground estimation

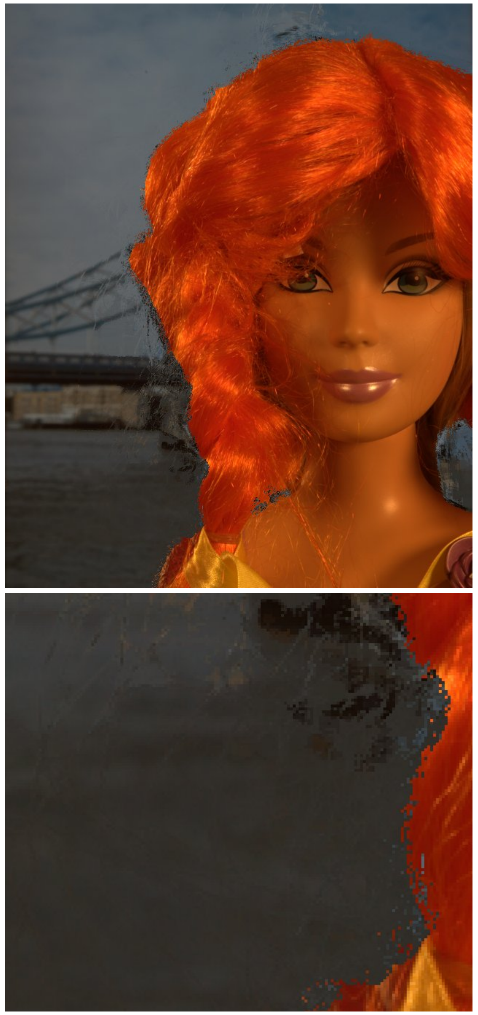

(d) Gathering: F_g^p

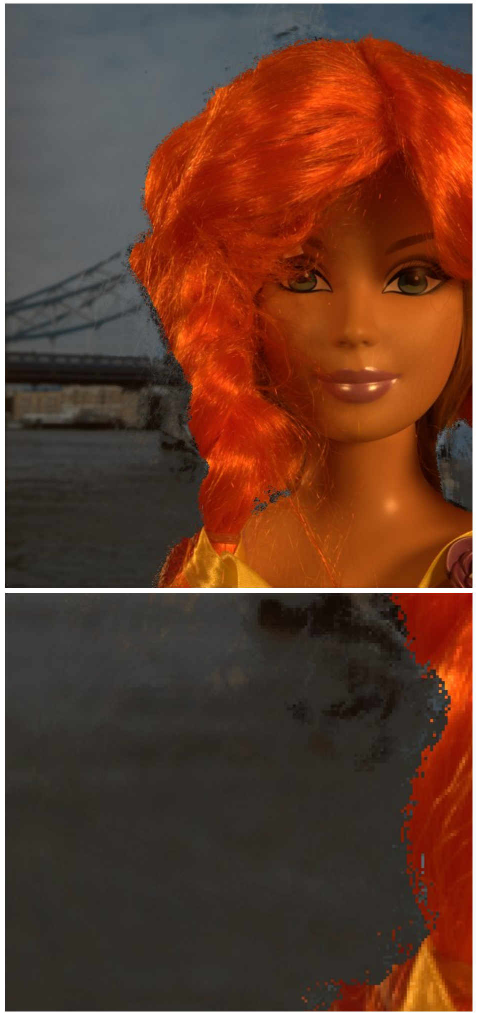

(e) Refinement: F_r^p


(f) Smoothness: final F^p

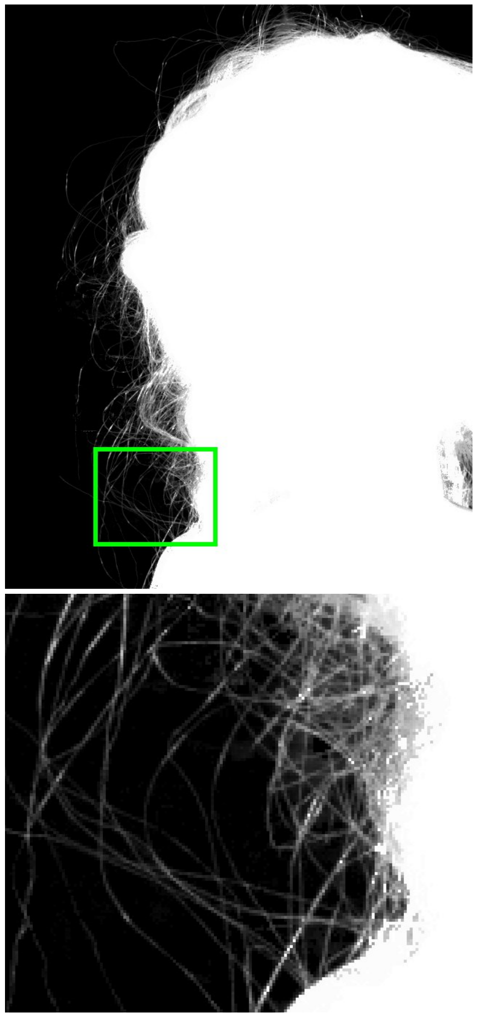
Background estimation

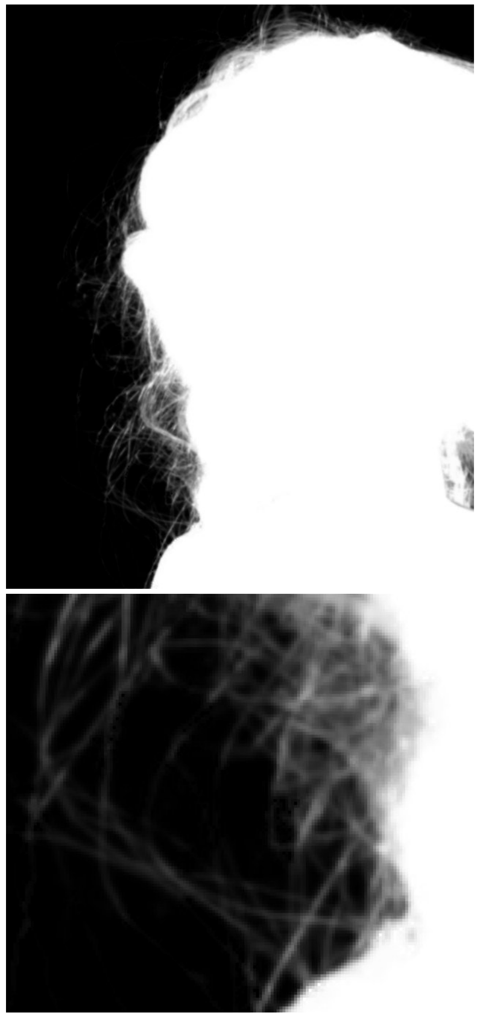

(g) Gathering: B_g^p

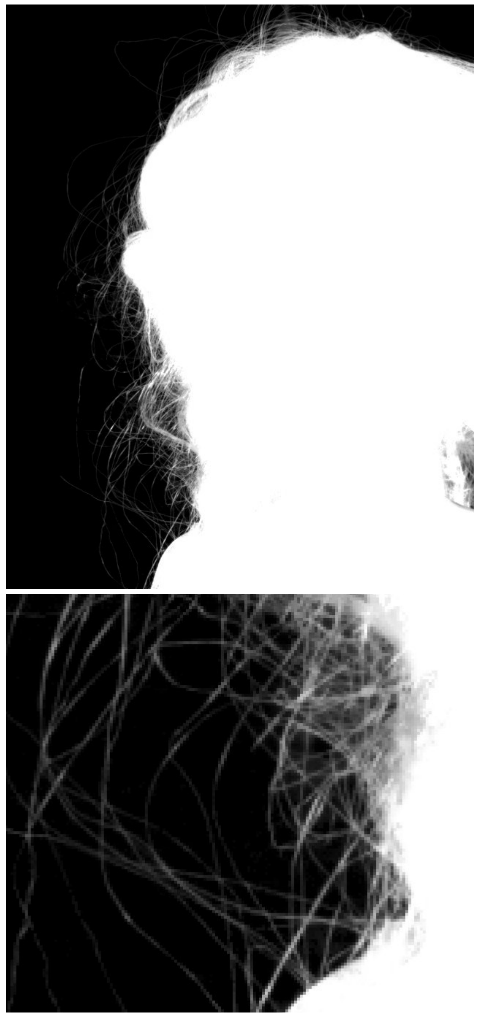

(h) Refinement: B_r^p

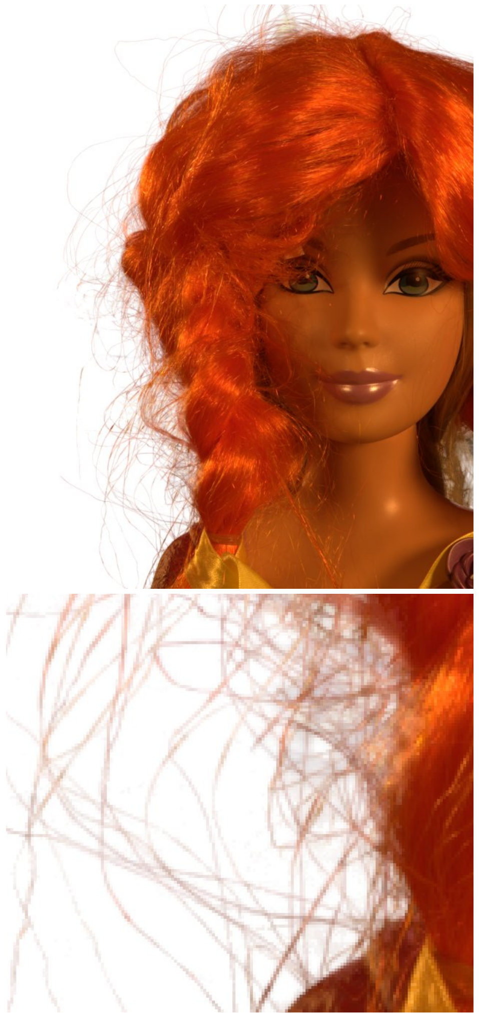

(i) Smoothness: final B^p

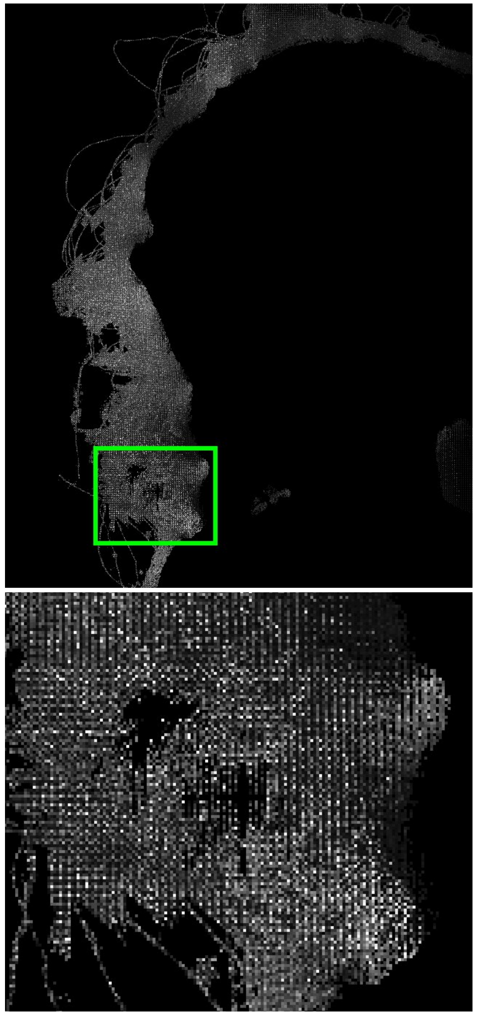
Alpha estimation

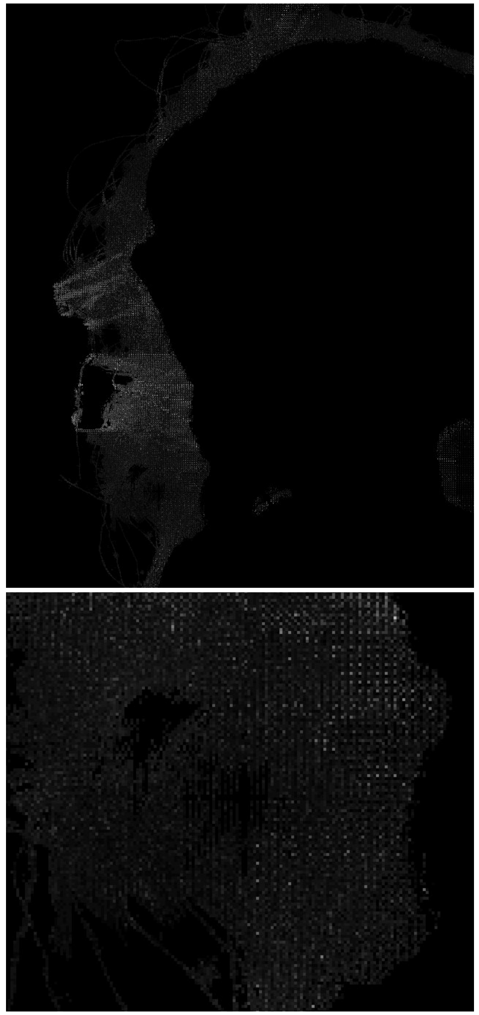

(j) Refinement: α_r^p

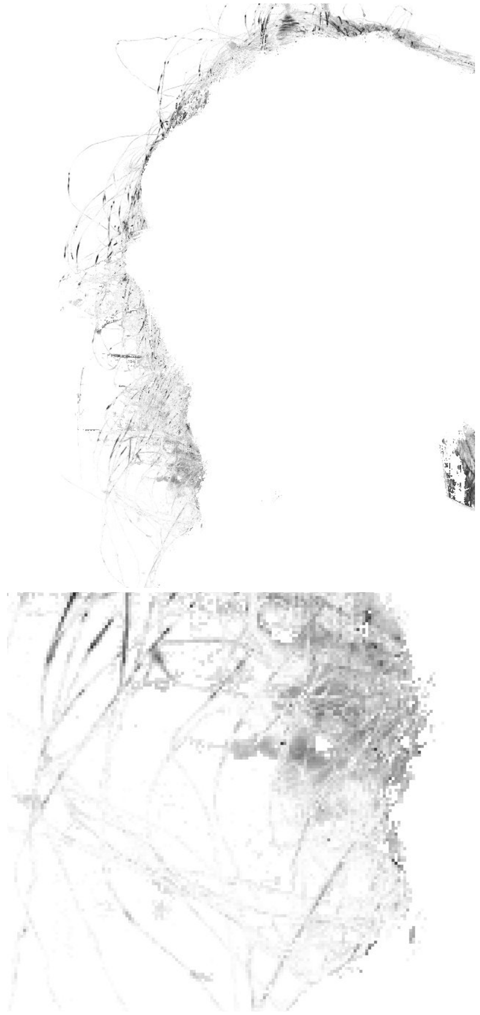

(k) Smoothness: low freq. alpha α_l^p

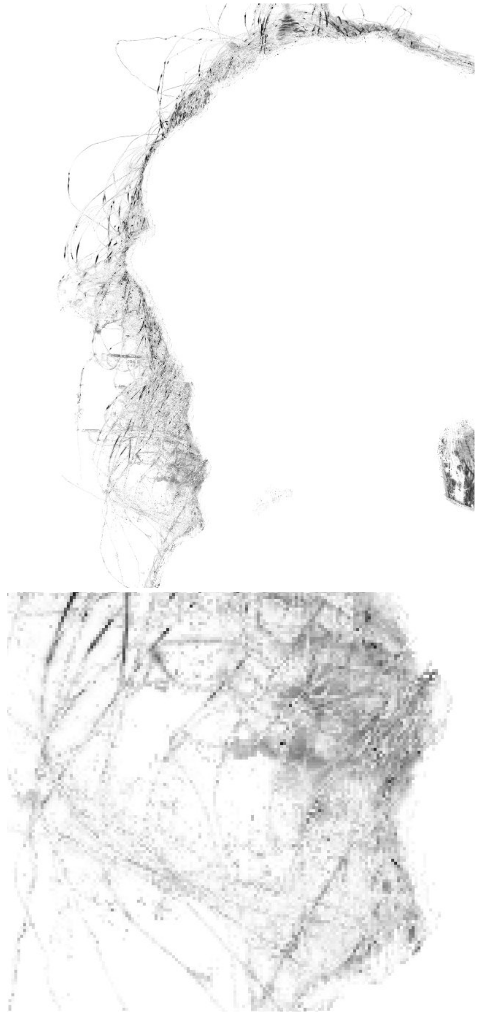

(l) Smoothness: final alpha α^p


(m) Final composite


(n) Gathering: σ_f^2 ($\times 4$)


(o) Gathering: σ_b^2 ($\times 6$)


(p) Refinement: f_r^p


(q) Smoothness: final confidence f^p

5 MATTE OPTIMIZATION

The output of the real-time alpha matting algorithm described in Chapter 4 can produce high-quality results for challenging images, as can be seen in Chapter 6. Nevertheless, some users might want to achieve the best possible final matte, even if this means having higher computation times. This can be achieved by refining the matte obtained in Chapter 4 using an additional optimization step. This step is analogous to the one in (RHEMANN et al., 2008), where the final matte is obtained by minimizing a quadratic cost function in α. This cost function is comprised of a *smoothness term* and a *data term*. Here, we use the same smoothness term as (RHEMANN et al., 2008), but for the data term we use the matting parameters obtained in Chapter 4.

5.1 Smoothness term

The smoothness term uses the *matting Laplacian* L proposed by Levin *et al.* (2008). The matting Laplacian is a matrix characterizing a cost function relating α-values for pixels in small 3×3 pixel windows, and is derived from local smoothness assumptions on foreground and background colors F and B. Furthermore, Levin *et al.* showed that it is possible to analytically eliminate F and B, yielding a quadratic cost function in only α. A detailed derivation of L is out of the scope of this work and the readers are referred to (LEVIN; LISCHINSKI; WEISS, 2008).

5.2 Data term

The data term is defined by the pixel-wise α^p and confidence f^p obtained in Chapter 4. Intuitively, pixels with a high confidence value should respect this estimated α, while pixels with a low confidence value should rely more on the α propagation induced by the optimization process.

5.3 Solving for the matte

Let $\hat{\alpha}^T = [\alpha^0, \ \ldots \ \alpha^p, \ \ldots \ \alpha^n]$ be a vector of all alpha values — obtained in Chapter 4 — for all n pixels in the input image. Let $\hat{\Gamma}$ be a diagonal matrix where each diagonal element γ_p is defined as

$$\gamma_p = \begin{cases} f^p & \text{if } p \in T_u \\ 0 & \text{if } p \in T_f \cup T_b \end{cases} \tag{5.1}$$

The final alpha matte is obtained by solving for

$$\alpha = \text{argmin} \quad \alpha^T L\,\alpha + \lambda\,(\alpha - \hat{\alpha})^T D(\alpha - \hat{\alpha}) + \gamma\,(\alpha - \hat{\alpha})^T \hat{\Gamma}(\alpha - \hat{\alpha}) \tag{5.2}$$

where λ is some large number, $\gamma^{\dagger}$ is a constant which defines the relative weighting between the data and smoothness term, L is the matting Laplacian and D is a diagonal matrix whose diagonal elements are one for pixels in $T_f \cup T_b$ and zero for all other pixels. Equation 5.2 can be broken down as follows:

- **The first term** is the smoothness term. It guarantees the affinity of α values between neighboring pixels.
- **The second term** assures that the trimap's constraints are respected. As λ is large, the final α that minimizes the right-hand-size of Equation 5.2 must match $\hat{\alpha}$ for all constrained pixels (*i.e.*, pixels in $T_f \cup T_b$).
- **The third term** is the data term. It guarantees that pixels with high confidence values will respect the $\hat{\alpha}$ values obtained in Chapter 4.

Equation 5.2 defines a quadratic cost in α, thus the global minimum may be found by differentiating and setting the derivatives to zero. This amounts to solving the following sparse linear system

$$(L + \lambda D + \gamma\hat{\Gamma})\,\alpha = (\lambda D + \gamma\hat{\Gamma})\,\hat{\alpha} \tag{5.3}$$

The final alpha matte is obtained by first evaluating the real-time matting algorithm (described in Chapter 4). The resulting matting parameters are then used to assemble the sparse linear system from Equation 5.3, which is solved using Matlab's "backslash" operator (a direct solver). The matting Laplacian matrix L is obtained using the original implementation of (LEVIN; LISCHINSKI; WEISS, 2008).

$^{\dagger}\gamma = 10^{-1}$ in our implementation

6 RESULTS

The technique described in this work was implemented using C++ and GLSL and was used to process a large number of images and videos. Given that the search space associated with Equation 4.11 is both small and discrete, its minima is computed by evaluating this equation for its corresponding entire search space and by selecting the sample-pair with the smallest value. Since these operations can be performed independently for each pixel $p \in T_u$, we exploit the inherent parallelism of current GPUs to efficiently perform these searches in parallel. All the results reported here were obtained using a 2.8 GHz Quad Core PC with 8 GB of memory and a GeForce GTX 280 with 1024 MB of video memory.

6.1 Matte Error Evaluation

In order to assess the quality of the results obtained by the proposed real-time matting technique, the benchmark provided by Rhemann *et al.* (RHEMANN et al., 2009) is used. It evaluates and compares the accuracy of an image-matting technique against the results produced by the state-of-the-art. Such a benchmark is composed of eight test images publicly available at $www.alphamatting.com$ (Figure 6.1), each accompanied by three trimaps (*small*, *large* and *user*). These images are designed to be challenging representations of natural scenes, containing examples of highly textured backgrounds, as well as images where background and foreground colors cannot be easily differentiated. The ground-truth alpha mattes for each of the test images are used to assess the quality of the results, but are not disclosed to the public. As such, Rhemann *et al.*'s benchmark provides an independent and reliable mechanism for evaluating digital image-matting algorithms.

Figures 6.2 and 6.3 show four tables produced by Rhemann *et al.*'s benchmark. These tables provide two types of error metrics — *sum of absolute differences* (SAD) and *mean squared error* (MSE) — which are obtained by comparing each technique's generated matte with the undisclosed ground-truth matte. Ranking information for each image-trimap pair is also provided along with an "overall rank" (obtained by averaging all raking results for each technique). However, such ranking algorithm is severely oversimplified, resulting in a tendency to overestimate rank positions. For instance, in the MSE table of Figure 6.2, under the "Donkey" image with a "large" trimap, all the top five matting techniques obtain a MSE of 0.4. However, the proposed technique (Improved Sampling Matting (Real-Time)), which also has a MSE of 0.4, gets ranked at the fifth position (rank 5), which has a significant impact in our final "overall rank". This should not happen for such a small error difference[1] relative to the smallest error obtained

[1]equal to 0.09 in the worst case

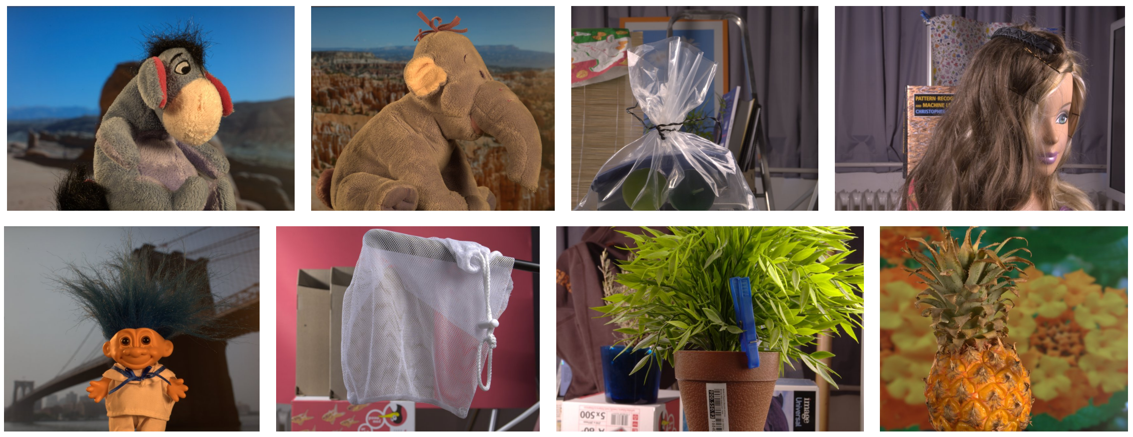


Figure 6.1: All eight images from the test dataset of Rhemann *et al.*'s benchmark. These images are designed as challenging representations of natural scenes. We have examples of: low contrast between foreground and background colors; large unknown regions with transparency; long, thin and transparent hair structures; discontinuities in the foreground object border; and highly textured backgrounds. Ground-truth mattes are not disclosed to the public. From $www.alphamatting.com$.

among all techniques (the second decimal place was not considered significant by the benchmark authors and was not included in the benchmark's tables). Another example where this simplified ranking scheme fails to properly model the relative quality of matting results can be seen in Table 6.1. Here, the difference in error from rank 2 to rank 1 is $0.8 - 0.7 = 0.1$, while the difference in error from rank 3 to rank 2 is $1.2 - 0.8 = 0.4$, which is four times larger. Yet, this important information is not accurately represented in the final ranking. To address these problems we propose a new ranking methodology.

Matting Technique	**Rank**	**MSE**	**MSE diff.**
Improved color matting	1	0.7	-
Improved Sampling Matting	2	0.8	0.1
Robust Matting	3	1.2	0.4

Table 6.1: Another example where the simplified ranking scheme in Rhemann *et al.*'s benchmark fails to properly model the relative quality of matting results. Values obtained from the MSE table of Figure 6.2, under the "Plant" image with a "large" trimap.

6.1.1 Raking by Relative Error

To generate an "overall rank", each technique's error values must be combined in some specified way. However, there is no meaning in directly combining MSE or SAD values, as they were obtained from different input images. To solve this problem, we first transform these errors into a *relative error* form, where they can be meaningfully averaged to assess the overall performance of a matting technique.

The relative error R of a matting technique $\mathcal{M}_k$ for an image $\mathcal{I}$ with a trimap $\mathcal{T}$ is defined as

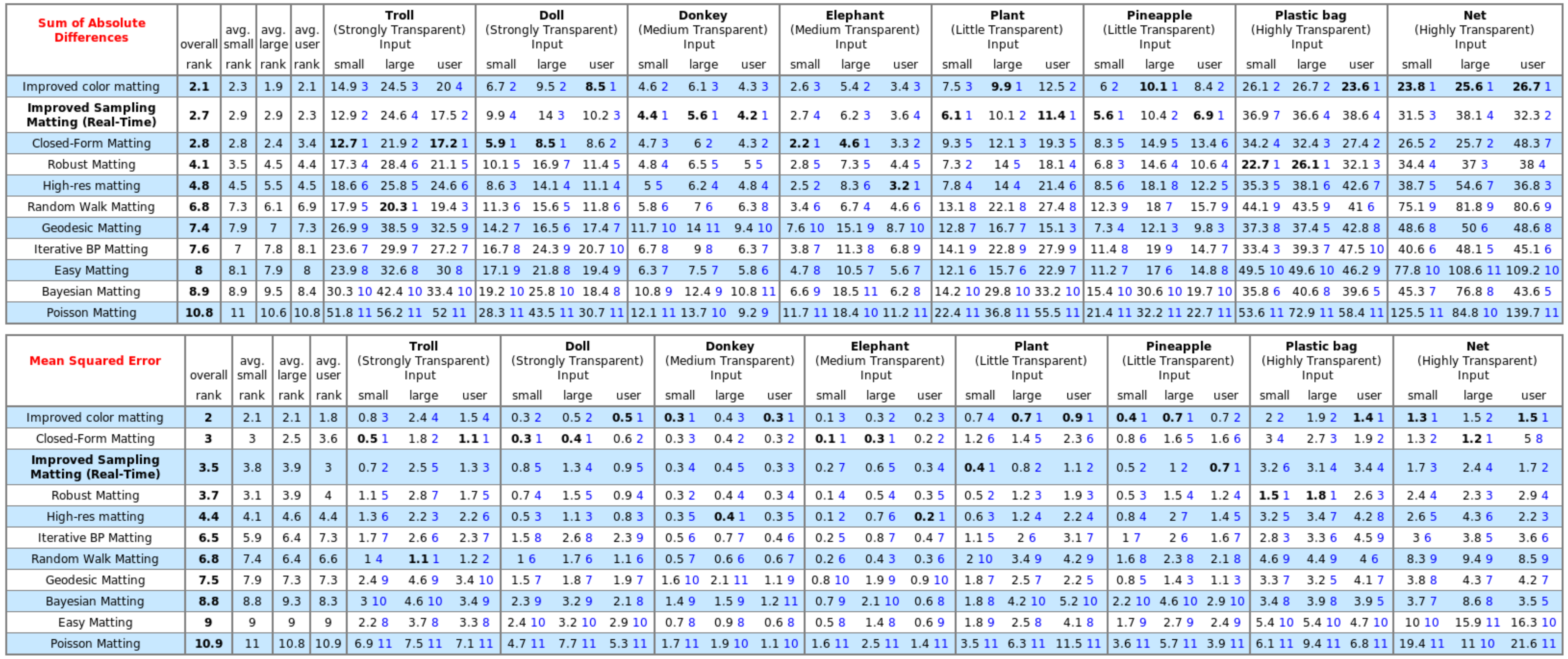
$$R(\mathcal{M}_k, \mathcal{I}, \mathcal{T}) = \frac{error(\mathcal{M}_k, \mathcal{I}, \mathcal{T})}{min_i\left[error(\mathcal{M}_i, \mathcal{I}, \mathcal{T})\right]} \qquad (6.1)$$

where $error(\ldots)$ is the SAD or MSE value obtained from the benchmark tables (Figures 6.2 and 6.3). The relative error $R(\mathcal{M}_k, \mathcal{I}, \mathcal{T})$ accurately expresses the quality of a matting technique $\mathcal{M}_k$ relative to the smallest error obtained among all techniques. By using this methodology, all the top five matting techniques (which have an MSE of 0.4) in Figure 6.2, regarding the "Donkey" image with a "large" trimap, will have a relative error R equal to 1 — *i.e.,* their results have 0% more errors than the best obtained result. More precisely, a technique with a value of $R(\ldots) = r$ produces results with $100 \times (r - 1)\%$ more errors than the best obtained results. Table 6.2 shows this new ranking methodology applied to the example of Table 6.1.

Matting Technique	**Relative Rank R**	**MSE**	**MSE diff.**
Improved color matting	1.00	0.7	-
Improved Sampling Matting	1.14	0.8	0.1
Robust Matting	1.71	1.2	0.4

Table 6.2: The relative error R properly models the quality of matting techniques relative to the smallest error obtained among all techniques.

Benchmark tables for the proposed real-time technique

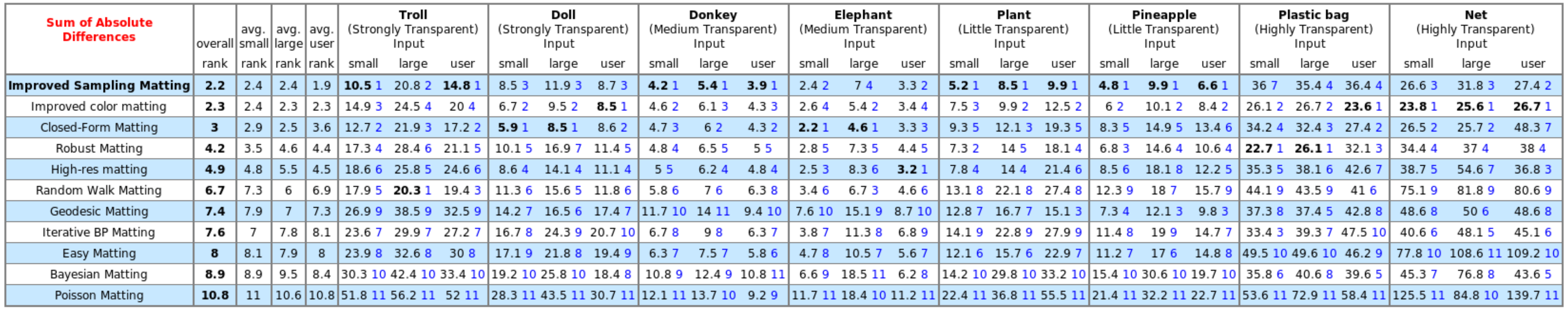


Sum of Absolute Differences	overall	avg. small	avg. large	avg. user	Troll (Strongly Transparent) Input			Doll (Strongly Transparent) Input			Donkey (Medium Transparent) Input			Elephant (Medium Transparent) Input			Plant (Little Transparent) Input			Pineapple (Little Transparent) Input			Plastic bag (Highly Transparent) Input			Net (Highly Transparent) Input		
	rank	rank	rank	rank	small	large	user	small	large	user	small	large	user	small	large	user	small	large	user	small	large	user	small	large	user	small	large	user
Improved color matching	**2.1**	2.3	1.9	2.1	14.9 3	24.5 3	20 4	6.7 2	9.5 2	**8.5** 1	4.6 2	6.1 3	4.3 3	2.6 3	5.4 2	3.4 3	7.5 3	**9.9** 1	12.5 2	6 2	**10.1** 1	8.4 2	26.1 2	26.7 2	**23.6** 1	**23.8** 1	**25.6** 1	**26.7** 1
Improved Sampling Matting (Real-Time)	**2.7**	2.9	2.9	2.3	12.9 2	24.6 4	17.5 2	9.9 4	14 3	10.2 3	**4.4** 1	**5.6** 1	**4.2** 1	2.7 4	6.2 3	3.6 4	**6.1** 1	10.1 2	**11.4** 1	**5.6** 1	10.4 2	**6.9** 1	36.9 7	36.6 4	38.6 4	31.5 3	38.1 4	32.3 2
Closed-Form Matting	**2.8**	2.8	2.4	3.4	**12.7** 1	21.9 2	**17.2** 1	**5.9** 1	**8.5** 1	8.6 2	4.7 3	6 2	4.3 2	**2.2** 1	**4.6** 1	3.3 2	9.3 5	12.1 3	19.3 5	8.3 5	14.9 5	13.4 6	34.2 4	32.4 3	27.4 2	26.5 2	25.7 2	48.3 7
Robust Matting	**4.1**	3.5	4.5	4.4	17.3 4	28.4 6	21.1 5	10.1 5	16.9 7	11.4 5	4.8 4	6.5 5	5 5	2.8 5	7.3 5	4.4 5	7.3 2	14 5	18.1 4	6.8 3	14.6 4	10.6 4	**22.7** 1	**26.1** 1	32.1 3	34.4 4	37 3	38 4
High-res matting	**4.8**	4.5	5.5	4.5	18.6 6	25.8 5	24.6 6	8.6 3	14.1 4	11.1 4	5 5	6.2 4	4.8 4	2.5 2	8.3 6	**3.2** 1	7.8 4	14 4	21.4 6	8.5 6	18.1 8	12.2 5	35.3 5	38.1 6	42.6 7	38.7 5	54.6 7	36.8 3
Random Walk Matting	**6.8**	7.3	6.1	6.9	17.9 5	**20.3** 1	19.4 3	11.3 6	15.6 5	11.8 6	5.8 6	7 6	6.3 8	3.4 6	6.7 4	4.6 6	13.1 8	22.1 8	27.4 8	12.3 9	18 7	15.7 9	44.1 9	43.5 9	41 6	75.1 9	81.8 9	80.6 9
Geodesic Matting	**7.4**	7.9	7	7.3	26.9 9	38.5 9	32.5 9	14.2 7	16.5 6	17.4 7	11.7 10	14 11	9.4 10	7.6 10	15.1 9	8.7 10	12.8 7	16.7 7	15.1 3	7.3 4	12.1 3	9.8 3	37.3 8	37.4 5	42.8 8	48.6 8	50 6	48.6 8
Iterative BP Matting	**7.6**	7	7.8	8.1	23.6 7	29.9 7	27.2 7	16.7 8	24.3 9	20.7 10	6.7 8	9 8	6.3 7	3.8 7	11.3 8	6.8 9	14.1 9	22.8 9	27.9 9	11.4 8	19 9	14.7 7	33.4 3	39.3 7	47.5 10	40.6 6	48.1 5	45.1 6
Easy Matting	**8**	8.1	7.9	8	23.9 8	32.6 8	30 8	17.1 9	21.8 8	19.4 9	6.3 7	7.5 7	5.8 6	4.7 8	10.5 7	5.6 7	12.1 6	15.7 6	22.9 7	11.2 7	17 6	14.8 8	49.5 10	49.6 10	46.2 9	77.8 10	108.6 11	109.2 10
Bayesian Matting	**8.9**	8.9	9.5	8.4	30.3 10	42.4 10	33.4 10	19.2 10	25.8 10	18.4 8	10.8 9	12.4 9	10.8 11	6.6 9	18.5 11	6.2 8	14.2 10	29.8 10	33.2 10	15.4 10	30.6 10	19.7 10	35.8 6	40.6 8	39.6 5	45.3 7	76.8 8	43.6 5
Poisson Matting	**10.8**	11	10.6	10.8	51.8 11	56.2 11	52 11	28.3 11	43.5 11	30.7 11	12.1 11	13.7 10	9.2 9	11.7 11	18.4 10	11.2 11	22.4 11	36.8 11	55.5 11	21.4 11	32.2 11	22.7 11	53.6 11	72.9 11	58.4 11	125.5 11	84.8 10	139.7 11

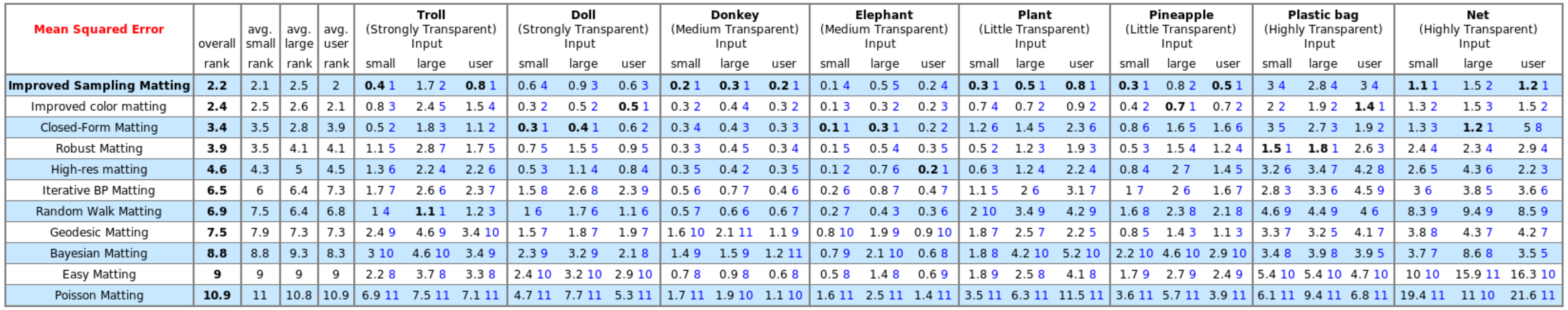
Mean Squared Error	overall	avg. small	avg. large	avg. user	Troll (Strongly Transparent) Input			Doll (Strongly Transparent) Input			Donkey (Medium Transparent) Input			Elephant (Medium Transparent) Input			Plant (Little Transparent) Input			Pineapple (Little Transparent) Input			Plastic bag (Highly Transparent) Input			Net (Highly Transparent) Input		
	rank	rank	rank	rank	small	large	user	small	large	user	small	large	user	small	large	user	small	large	user	small	large	user	small	large	user	small	large	user
Improved color matching	**2**	2.1	2.1	1.8	0.8 3	2.4 4	1.5 4	0.3 2	0.5 2	**0.5** 1	**0.3** 1	0.4 3	**0.3** 1	0.1 3	0.3 2	0.2 3	0.7 4	**0.7** 1	**0.9** 1	**0.4** 1	**0.7** 1	0.7 2	2 2	1.9 2	**1.4** 1	**1.3** 1	1.5 2	**1.5** 1
Closed-Form Matting	**3**	3	2.5	3.6	**0.5** 1	1.8 2	**1.1** 1	**0.3** 1	**0.4** 1	0.6 2	0.3 3	0.4 2	0.3 2	**0.1** 1	**0.3** 1	0.2 2	1.2 6	1.4 5	2.3 6	0.8 6	1.6 5	1.6 6	3 4	2.7 3	1.9 2	1.3 2	**1.2** 1	5 8
Improved Sampling Matting (Real-Time)	**3.5**	3.8	3.9	3	0.7 2	2.5 5	1.3 3	0.8 5	1.3 4	0.9 5	0.3 4	0.4 5	0.3 3	0.2 7	0.6 5	0.3 4	**0.4** 1	0.8 2	1.1 2	0.5 2	1 2	**0.7** 1	3.2 6	3.1 4	3.4 4	1.7 3	2.4 4	1.7 2
Robust Matting	**3.7**	3.1	3.9	4	1.1 5	2.8 7	1.7 5	0.7 4	1.5 5	0.9 4	0.3 2	0.4 4	0.3 4	0.1 4	0.5 4	0.3 5	0.5 2	1.2 3	1.9 3	0.5 3	1.5 4	1.2 4	**1.5** 1	**1.8** 1	2.6 3	2.4 4	2.3 3	2.9 4
High-res matting	**4.4**	4.1	4.6	4.4	1.3 6	2.2 3	2.2 6	0.5 3	1.1 3	0.8 3	0.3 5	**0.4** 1	0.3 5	0.1 2	0.7 6	**0.2** 1	0.6 3	1.2 4	2.2 4	0.8 4	2 7	1.4 5	3.2 5	3.4 7	4.2 8	2.6 5	4.3 6	2.2 3
Iterative BP Matting	**6.5**	5.9	6.4	7.3	1.7 7	2.6 6	2.3 7	1.5 8	2.6 8	2.3 9	0.5 6	0.7 7	0.4 6	0.2 5	0.8 7	0.4 7	1.1 5	2 6	3.1 7	1 7	2 6	1.6 7	2.8 3	3.3 6	4.5 9	3 6	3.8 5	3.6 6
Random Walk Matting	**6.8**	7.4	6.4	6.6	1 4	**1.1** 1	1.2 2	1 6	1.7 6	1.1 6	0.5 7	0.6 6	0.6 7	0.2 6	0.4 3	0.3 6	2 10	3.4 9	4.2 9	1.6 8	2.3 8	2.1 8	4.6 9	4.4 9	4 6	8.3 9	9.4 9	8.5 9
Geodesic Matting	**7.5**	7.9	7.3	7.3	2.4 9	4.6 9	3.4 10	1.5 7	1.8 7	1.9 7	1.6 10	2.1 11	1.1 9	0.8 10	1.9 9	0.9 10	1.8 7	2.5 7	2.2 5	0.8 5	1.4 3	1.1 3	3.3 7	3.2 5	4.1 7	3.8 8	4.3 7	4.2 7
Bayesian Matting	**8.8**	8.8	9.3	8.3	3 10	4.6 10	3.4 9	2.3 9	3.2 9	2.1 8	1.4 9	1.5 9	1.2 11	0.7 9	2.1 10	0.6 8	1.8 8	4.2 10	5.2 10	2.2 10	4.6 10	2.9 10	3.4 8	3.9 8	3.9 5	3.7 7	8.6 8	3.5 5
Easy Matting	**9**	9	9	9	2.2 8	3.7 8	3.3 8	2.4 10	3.2 10	2.9 10	0.7 8	0.9 8	0.6 8	0.5 8	1.4 8	0.6 9	1.8 9	2.5 8	4.1 8	1.7 9	2.7 9	2.4 9	5.4 10	5.4 10	4.7 10	10 10	15.9 11	16.3 10
Poisson Matting	**10.9**	11	10.8	10.9	6.9 11	7.5 11	7.1 11	4.7 11	7.7 11	5.3 11	1.7 11	1.9 10	1.1 10	1.6 11	2.5 11	1.4 11	3.5 11	6.3 11	11.5 11	3.6 11	5.7 11	3.9 11	6.1 11	9.4 11	6.8 11	19.4 11	11 10	21.6 11

Figure 6.2: Results of Rhemann *et al.*'s benchmark for the proposed real-time matting technique (Improved Sampling Matting (Real-Time)). The techniques listed in these tables are: Improved color matting (RHEMANN; ROTHER; GELAUTZ, 2008), Closed-Form Matting (LEVIN; LISCHINSKI; WEISS, 2008), Robust Matting (WANG; COHEN, 2007), High-res matting (RHEMANN et al., 2008), Random Walk Matting (GRADY et al., 2005), Geodesic Matting (BAI; SAPIRO, 2007), Iterative BP Matting (WANG; COHEN, 2005), Easy Matting (GUAN et al., 2006), Bayesian Matting (CHUANG et al., 2001) and Poisson Matting (SUN et al., 2004).

Benchmark tables for the proposed matting technique with the additional optimization step



Sum of Absolute Differences	overall	avg. small	avg. large	avg. user	Troll (Strongly Transparent) Input			Doll (Strongly Transparent) Input			Donkey (Medium Transparent) Input			Elephant (Medium Transparent) Input			Plant (Little Transparent) Input			Pineapple (Little Transparent) Input			Plastic bag (Highly Transparent) Input			Net (Highly Transparent) Input		
	rank	rank	rank	rank	small	large	user	small	large	user	small	large	user	small	large	user	small	large	user	small	large	user	small	large	user	small	large	user
Improved Sampling Matting	**2.2**	2.4	2.4	1.9	**10.5** 1	20.8 2	**14.8** 1	8.5 3	11.9 3	8.7 3	**4.2** 1	**5.4** 1	**3.9** 1	2.4 2	7 4	3.3 2	**5.2** 1	**8.5** 1	**9.9** 1	**4.8** 1	**9.9** 1	**6.6** 1	36 7	35.4 4	36.4 4	26.6 3	31.8 3	27.4 2
Improved color matting	**2.3**	2.4	2.3	2.3	14.9 3	24.5 4	20 4	6.7 2	9.5 2	**8.5** 1	4.6 2	6.1 3	4.3 3	2.6 4	5.4 2	3.4 4	7.5 3	9.9 2	12.5 2	6 2	10.1 2	8.4 2	26.1 2	26.7 2	**23.6** 1	**23.8** 1	**25.6** 1	**26.7** 1
Closed-Form Matting	**3**	2.9	2.5	3.6	12.7 2	21.9 3	17.2 2	**5.9** 1	**8.5** 1	8.6 2	4.7 3	6 2	4.3 2	**2.2** 1	**4.6** 1	3.3 3	9.3 5	12.1 3	19.3 5	8.3 5	14.9 5	13.4 6	34.2 4	32.4 3	27.4 2	26.5 2	25.7 2	48.3 7
Robust Matting	**4.2**	3.5	4.6	4.4	17.3 4	28.4 6	21.1 5	10.1 5	16.9 7	11.4 5	4.8 4	6.5 5	5 5	2.8 5	7.3 5	4.4 5	7.3 2	14 5	18.1 4	6.8 3	14.6 4	10.6 4	**22.7** 1	**26.1** 1	32.1 3	34.4 4	37 4	38 4
High-res matting	**4.9**	4.8	5.5	4.5	18.6 6	25.8 5	24.6 6	8.6 4	14.1 4	11.1 4	5 5	6.2 4	4.8 4	2.5 3	8.3 6	**3.2** 1	7.8 4	14 4	21.4 6	8.5 6	18.1 8	12.2 5	35.3 5	38.1 6	42.6 7	38.7 5	54.6 7	36.8 3
Random Walk Matting	**6.7**	7.3	6	6.9	17.9 5	**20.3** 1	19.4 3	11.3 6	15.6 5	11.8 6	5.8 6	7 6	6.3 8	3.4 6	6.7 3	4.6 6	13.1 8	22.1 8	27.4 8	12.3 9	18 7	15.7 9	44.1 9	43.5 9	41 6	75.1 9	81.8 9	80.6 9
Geodesic Matting	**7.4**	7.9	7	7.3	26.9 9	38.5 9	32.5 9	14.2 7	16.5 6	17.4 7	11.7 10	14 11	9.4 10	7.6 10	15.1 9	8.7 10	12.8 7	16.7 7	15.1 3	7.3 4	12.1 3	9.8 3	37.3 8	37.4 5	42.8 8	48.6 8	50 6	48.6 8
Iterative BP Matting	**7.6**	7	7.8	8.1	23.6 7	29.9 7	27.2 7	16.7 8	24.3 9	20.7 10	6.7 8	9 8	6.3 7	3.8 7	11.3 8	6.8 9	14.1 9	22.8 9	27.9 9	11.4 8	19 9	14.7 7	33.4 3	39.3 7	47.5 10	40.6 6	48.1 5	45.1 6
Easy Matting	**8**	8.1	7.9	8	23.9 8	32.6 8	30 8	17.1 9	21.8 8	19.4 9	6.3 7	7.5 7	5.8 6	4.7 8	10.5 7	5.6 7	12.1 6	15.7 6	22.9 7	11.2 7	17 6	14.8 8	49.5 10	49.6 10	46.2 9	77.8 10	108.6 11	109.2 10
Bayesian Matting	**8.9**	8.9	9.5	8.4	30.3 10	42.4 10	33.4 10	19.2 10	25.8 10	18.4 8	10.8 9	12.4 9	10.8 11	6.6 9	18.5 11	6.2 8	14.2 10	29.8 10	33.2 10	15.4 10	30.6 10	19.7 10	35.8 6	40.6 8	39.6 5	45.3 7	76.8 8	43.6 5
Poisson Matting	**10.8**	11	10.6	10.8	51.8 11	56.2 11	52 11	28.3 11	43.5 11	30.7 11	12.1 11	13.7 10	9.2 9	11.7 11	18.4 10	11.2 11	22.4 11	36.8 11	55.5 11	21.4 11	32.2 11	22.7 11	53.6 11	72.9 11	58.4 11	125.5 11	84.8 10	139.7 11



Mean Squared Error	overall	avg. small	avg. large	avg. user	Troll (Strongly Transparent) Input			Doll (Strongly Transparent) Input			Donkey (Medium Transparent) Input			Elephant (Medium Transparent) Input			Plant (Little Transparent) Input			Pineapple (Little Transparent) Input			Plastic bag (Highly Transparent) Input			Net (Highly Transparent) Input		
	rank	rank	rank	rank	small	large	user	small	large	user	small	large	user	small	large	user	small	large	user	small	large	user	small	large	user	small	large	user
Improved Sampling Matting	**2.2**	2.1	2.5	2	**0.4** 1	1.7 2	**0.8** 1	0.6 4	0.9 3	0.6 3	**0.2** 1	**0.3** 1	**0.2** 1	0.1 4	0.5 5	0.2 4	**0.3** 1	**0.5** 1	**0.8** 1	**0.3** 1	0.8 2	**0.5** 1	3 4	2.8 4	3 4	**1.1** 1	1.5 2	**1.2** 1
Improved color matting	**2.4**	2.5	2.6	2.1	0.8 3	2.4 5	1.5 4	0.3 2	0.5 2	**0.5** 1	0.3 2	0.4 4	0.3 2	0.1 3	0.3 2	0.2 3	0.7 4	0.7 2	0.9 2	0.4 2	**0.7** 1	0.7 2	2 2	1.9 2	**1.4** 1	1.3 2	1.5 3	1.5 2
Closed-Form Matting	**3.4**	3.5	2.8	3.9	0.5 2	1.8 3	1.1 2	**0.3** 1	**0.4** 1	0.6 2	0.3 4	0.4 3	0.3 3	**0.1** 1	**0.3** 1	0.2 2	1.2 6	1.4 5	2.3 6	0.8 6	1.6 5	1.6 6	3 5	2.7 3	1.9 2	1.3 3	**1.2** 1	5 8
Robust Matting	**3.9**	3.5	4.1	4.1	1.1 5	2.8 7	1.7 5	0.7 5	1.5 5	0.9 5	0.3 3	0.4 5	0.3 4	0.1 5	0.5 4	0.3 5	0.5 2	1.2 3	1.9 3	0.5 3	1.5 4	1.2 4	**1.5** 1	**1.8** 1	2.6 3	2.4 4	2.3 4	2.9 4
High-res matting	**4.6**	4.3	5	4.5	1.3 6	2.2 4	2.2 6	0.5 3	1.1 4	0.8 4	0.3 5	0.4 2	0.3 5	0.1 2	0.7 6	**0.2** 1	0.6 3	1.2 4	2.2 4	0.8 4	2 7	1.4 5	3.2 6	3.4 7	4.2 8	2.6 5	4.3 6	2.2 3
Iterative BP Matting	**6.5**	6	6.4	7.3	1.7 7	2.6 6	2.3 7	1.5 8	2.6 8	2.3 9	0.5 6	0.7 7	0.4 6	0.2 6	0.8 7	0.4 7	1.1 5	2 6	3.1 7	1 7	2 6	1.6 7	2.8 3	3.3 6	4.5 9	3 6	3.8 5	3.6 6
Random Walk Matting	**6.9**	7.5	6.4	6.8	1 4	**1.1** 1	1.2 3	1 6	1.7 6	1.1 6	0.5 7	0.6 6	0.6 7	0.2 7	0.4 3	0.3 6	2 10	3.4 9	4.2 9	1.6 8	2.3 8	2.1 8	4.6 9	4.4 9	4 6	8.3 9	9.4 9	8.5 9
Geodesic Matting	**7.5**	7.9	7.3	7.3	2.4 9	4.6 9	3.4 10	1.5 7	1.8 7	1.9 7	1.6 10	2.1 11	1.1 9	0.8 10	1.9 9	0.9 10	1.8 7	2.5 7	2.2 5	0.8 5	1.4 3	1.1 3	3.3 7	3.2 5	4.1 7	3.8 8	4.3 7	4.2 7
Bayesian Matting	**8.8**	8.8	9.3	8.3	3 10	4.6 10	3.4 9	2.3 9	3.2 9	2.1 8	1.4 9	1.5 9	1.2 11	0.7 9	2.1 10	0.6 8	1.8 8	4.2 10	5.2 10	2.2 10	4.6 10	2.9 10	3.4 8	3.9 8	3.9 5	3.7 7	8.6 8	3.5 5
Easy Matting	**9**	9	9	9	2.2 8	3.7 8	3.3 8	2.4 10	3.2 10	2.9 10	0.7 8	0.9 8	0.6 8	0.5 8	1.4 8	0.6 9	1.8 9	2.5 8	4.1 8	1.7 9	2.7 9	2.4 9	5.4 10	5.4 10	4.7 10	10 10	15.9 11	16.3 10
Poisson Matting	**10.9**	11	10.8	10.9	6.9 11	7.5 11	7.1 11	4.7 11	7.7 11	5.3 11	1.7 11	1.9 10	1.1 10	1.6 11	2.5 11	1.4 11	3.5 11	6.3 11	11.5 11	3.6 11	5.7 11	3.9 11	6.1 11	9.4 11	6.8 11	19.4 11	11 10	21.6 11

Figure 6.3: Results of Rhemann *et al.*'s benchmark for the proposed matting technique with the additional optimization step (Improved Sampling Matting). The techniques listed in these tables are: Improved color matting (RHEMANN; ROTHER; GELAUTZ, 2008), Closed-Form Matting (LEVIN; LISCHINSKI; WEISS, 2008), Robust Matting (WANG; COHEN, 2007), High-res matting (RHEMANN et al., 2008), Random Walk Matting (GRADY et al., 2005), Geodesic Matting (BAI; SAPIRO, 2007), Iterative BP Matting (WANG; COHEN, 2005), Easy Matting (GUAN et al., 2006), Bayesian Matting (CHUANG et al., 2001) and Poisson Matting (SUN et al., 2004).

6.1.2 Evaluation of Obtained Results

Table 6.3 summarizes the overall ranking of matting techniques according to the relative error metric defined in Section 6.1.1. The proposed real-time matting technique (*Improved Sampling Matting (Real-Time)*) ranks second. It is, however, up to 100 times faster (as shown in Section 6.2), allowing, for the first time, alpha matting in real-time applications. The proposed matting technique with the additional optimization step (*Improved Sampling Matting*) ranks first, obtaining the best results ever achieved in Rhemann *et al.*'s benchmark. Separate error values for all images in the test dataset can be found in Appendix A.

Matting Technique	Overall Rank	SAD Rank	MSE Rank
Improved Sampling Matting	1.21	1.14	1.28
Improved color matting	1.25	1.15	1.35
Improved Sampling Matting (Real-Time)	1.54	1.27	1.80
Closed-Form Matting	1.56	1.30	1.83
Robust Matting	1.69	1.43	1.95
High-res matting	1.88	1.56	2.21
Iterative BP Matting	2.62	1.97	3.16
Random Walk Matting	2.85	2.08	3.73
Geodesic Matting	3.26	2.13	4.38
Easy Matting	3.91	2.28	5.50
Bayesian Matting	4.05	2.60	5.55
Poisson Matting	7.25	3.84	10.65

Table 6.3 Relative error ranking for image-matting techniques.

These results, however, are worsened by the fact that one of the images in the test dataset breaks the assumption made by the proposed sampling technique: the "plastic bag" image (Figure 6.1, third image from the top-left) contains a completely transparent foreground object; thus, no foreground sample exists that approximates the true foreground color of the object. If we remove this image from the ranking results, the proposed technique has a much smaller relative error (Table 6.4, only the top four techniques are shown).

Matting Technique	Overall Rank	SAD Rank	MSE Rank
Improved Sampling Matting	1.14	1.09	1.19
Improved color matting	1.27	1.16	1.38
Improved Sampling Matting (Real-Time)	1.49	1.23	1.76
Closed-Form Matting	1.57	1.29	1.86

Table 6.4 Relative error ranking for image-matting techniques not considering the "plastic bag" image, which breaks the assumption made by the proposed sampling approach.

Figure 6.4 shows the alpha mattes generated by the proposed real-time technique for some images from the training dataset provided by (RHEMANN et al., 2009). For such dataset, the ground-truth mattes are available and are shown next to our results for comparison.

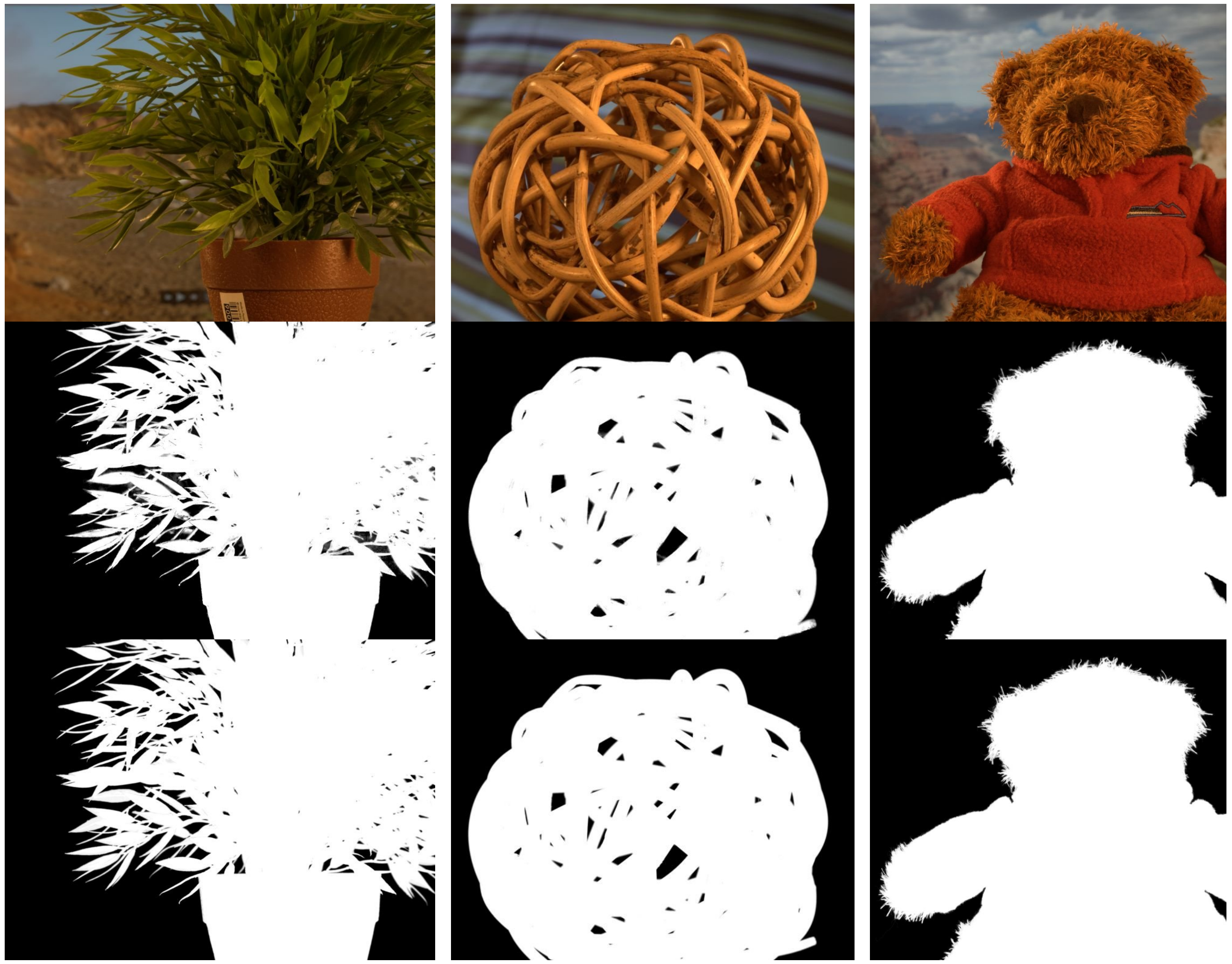


Figure 6.4: Top row: Images from the training dataset provided by (RHEMANN et al., 2009). Center row: Alpha mattes extracted with the proposed real-time technique using the trimaps supplied in the dataset. Bottom row: Ground-truth alpha mattes provided for comparison.

Image	# of pixels	% unknown	Time (sec)
elephant	536,800	16%	0.022
donkey	455,200	17%	0.021
pineapple	481,600	20%	0.027
doll	451,200	25%	0.027
plasticbag	529,600	28%	0.035
plant	425,600	35%	0.033
troll	512,000	40%	0.050
net	496,000	51%	0.063

(a)

Optimization (sec)
1.6
1.6
1.8
2.4
4.3
2.5
5.0
12.5

(b)

Table 6.5: (a) Comparison of the time taken to generate alpha mattes with the proposed real-time technique for the images in Figure 6.1, using the most conservative trimaps (*i.e.*, large). (b) Time taken by the additional optimization step to solve the linear system in Equation 5.3 (using Matlab's direct solver).

Technique	Time (sec)
Closed-Form Matting (2008)	18
Easy Matting (2006)	300
Random Walk Matting (2005)	5
Robust Matting (2007)	50
High-res Matting (2008)	4.5
Fast Matting (2005)	4.5

Table 6.6: Comparison of the time taken to generate alpha mattes with state-of-the art techniques for an image with $392,000$ pixels (0.3 Mpix). From (RHEMANN et al., 2008).

6.2 Performance Evaluation

Since existing techniques are not suitable for real-time applications, performance comparisons considering the time required to compute the alpha matte have been overlooked in many previous publications. The technique proposed in this work, on the other hand, can compute alpha mattes for typical images in real-time. Table 6.5a summarizes the time required by the proposed real-time technique to extract the alpha mattes for the test set of images available from Rhemann *et al.*'s benchmark (RHEMANN et al., 2009) using the most conservative trimaps (*i.e.*, large). For each image, Table 6.5 provides its dimensions, the number of pixels in the unknown region of the trimap, and the time required to compute the matte. For comparison, Table 6.6 shows the matte generation time for six other techniques (image size is $392,000$ pixels, or 0.3 Mpix) — this table was extracted from (RHEMANN et al., 2008), and all times were measured on the same machine. Note that the proposed technique is 100 times faster when compared to the fastest state-of-the-art algorithms.

Additionally, as discussed in Chapter 5, the alpha matte obtained using the proposed real-time technique can be further refined through an extra optimization step. This step involves assembling and solving a sparse set of linear equations, thus requiring a considerable amount of computational effort (Table 6.5b). However, as all state-of-the-art techniques involve solving such a sparse linear system, our results for the extended matting technique have comparable computation times to the ones from current techniques.

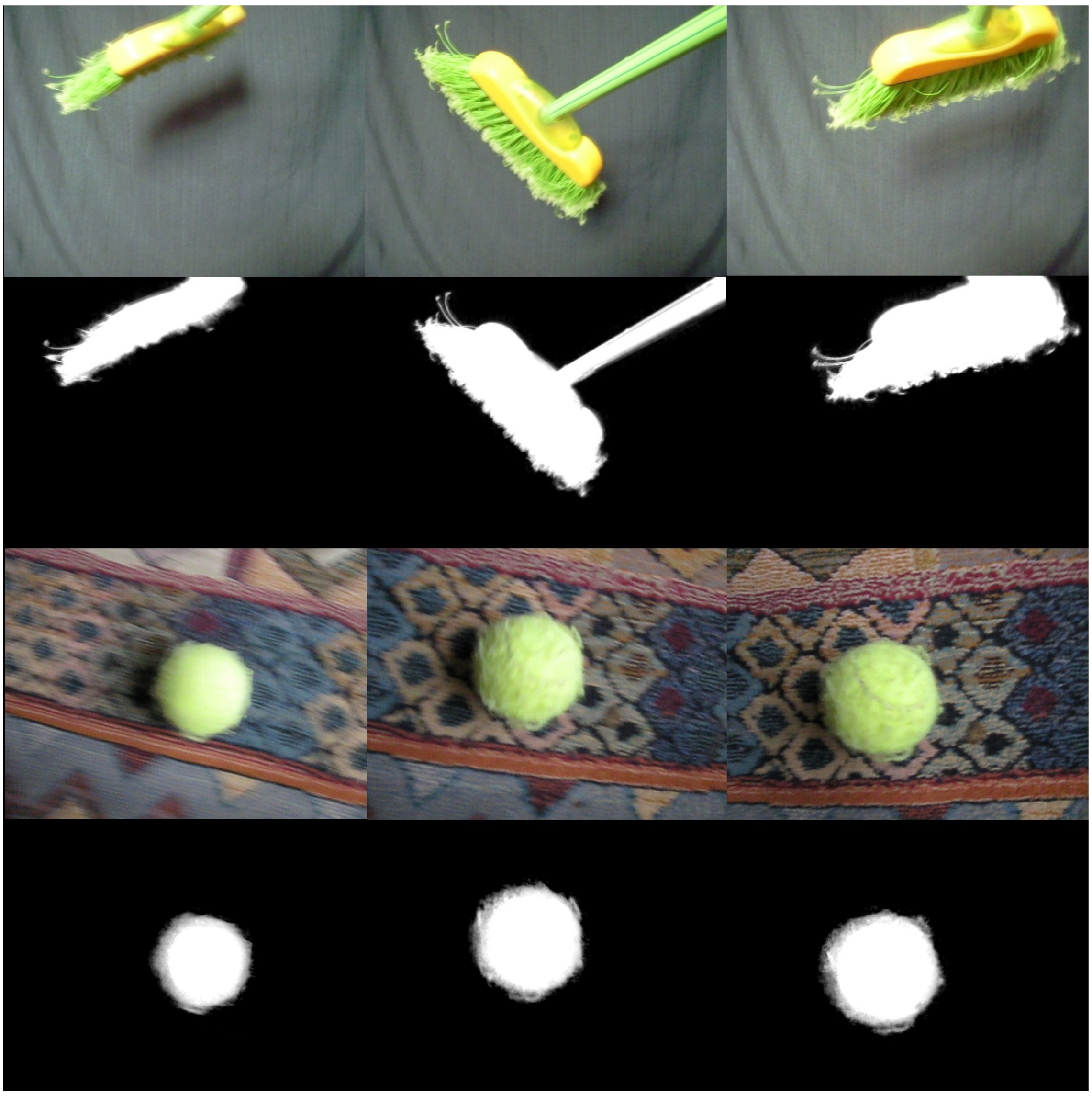


Figure 6.5: Some frames extracted from video sequences processed by the proposed technique for real-time matte extraction. The images on top show the original frames, while the extracted mattes are shown on the bottom.

6.3 Applications of Real-Time Alpha Matting

6.3.1 Video Matting

The proposed method enables the use of alpha matting in real-time applications for the first time. One possible such application is real-time matte generation for natural scene videos. Since the proposed approach uses a trimap as input, it needs to rely on other techniques for providing trimaps for each frame of the video in real-time. Figures 6.5 and 6.6 illustrate such an application for two video sequences. In these examples, the trimaps were created by dilating the boundaries of the binary segmented video frames. Such segmentation was obtained using a real-time background binary segmentation technique[†]

[†]Briefly, background color samples are obtained under several lighting conditions, which are then are used to model (offline) the background color *probability density function* (PDF) using the *kernel density estimation* method. This pre-computed PDF is used for real-time binary background segmentation.

described in Appendix B. Thus, given an input video sequence, the results shown were entirely computed in real-time. This means that the whole sequence of operations comprising binary segmentation, boundary dilation, matte extraction and compositing was performed in real time.

For highly textured backgrounds or greatly transparent foreground pixels, the produced matte might suffer from temporal noise, *i.e.* flickering. One can think of many ideas for increasing the temporal coherence of the matte, such as temporal blending of alpha values based on confidence values, or even selecting candidate samples along the time axis, in addition to the image space. This is an interesting problem that remains to be explored in future work.

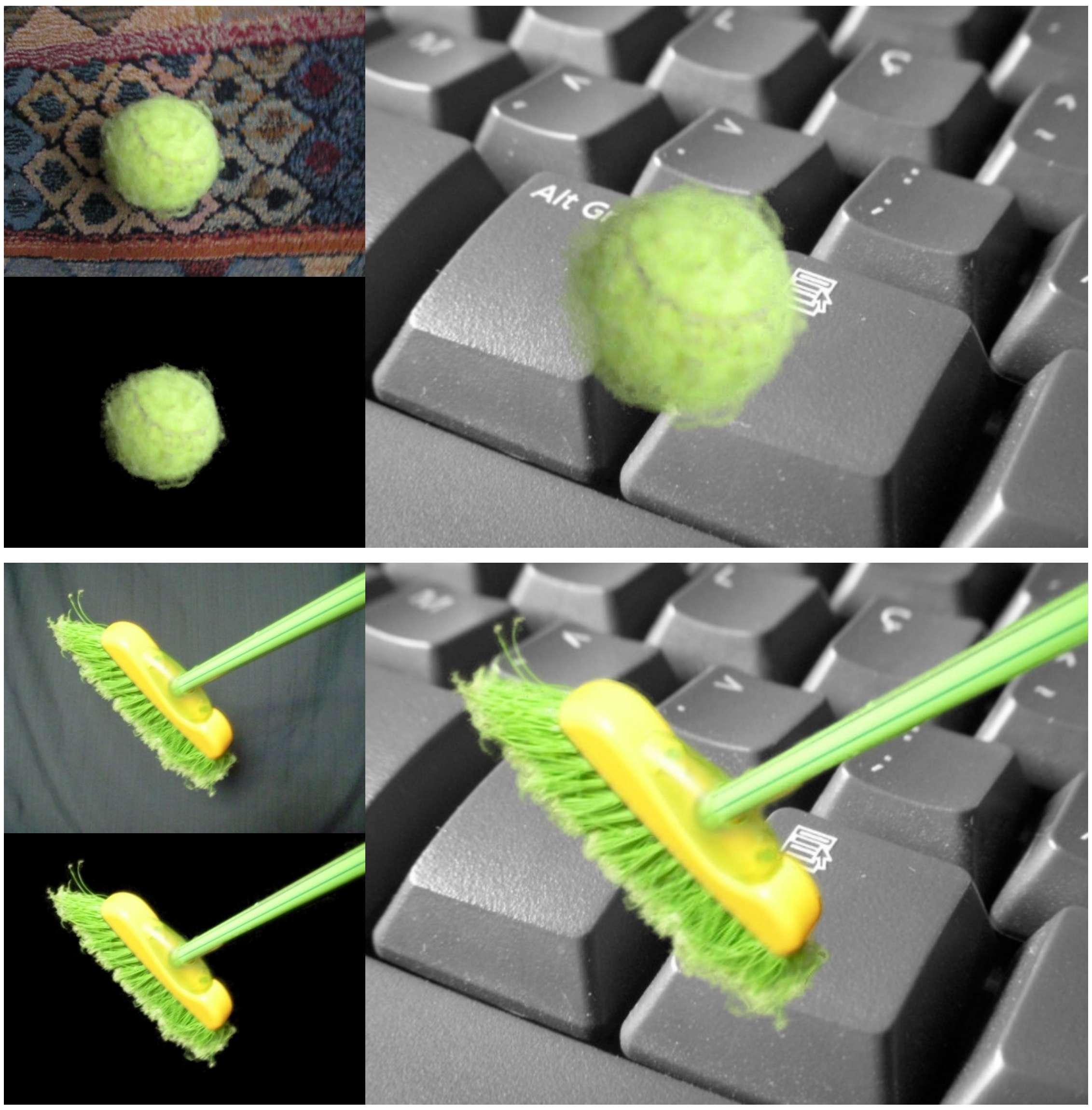


Figure 6.6: Final composite for two frames extracted from video sequences processed by the proposed technique in real-time. The images on the top-left show the original frames, while the images on the bottom-left show the extracted foreground object. On the right, the foreground is shown composited against a new background.

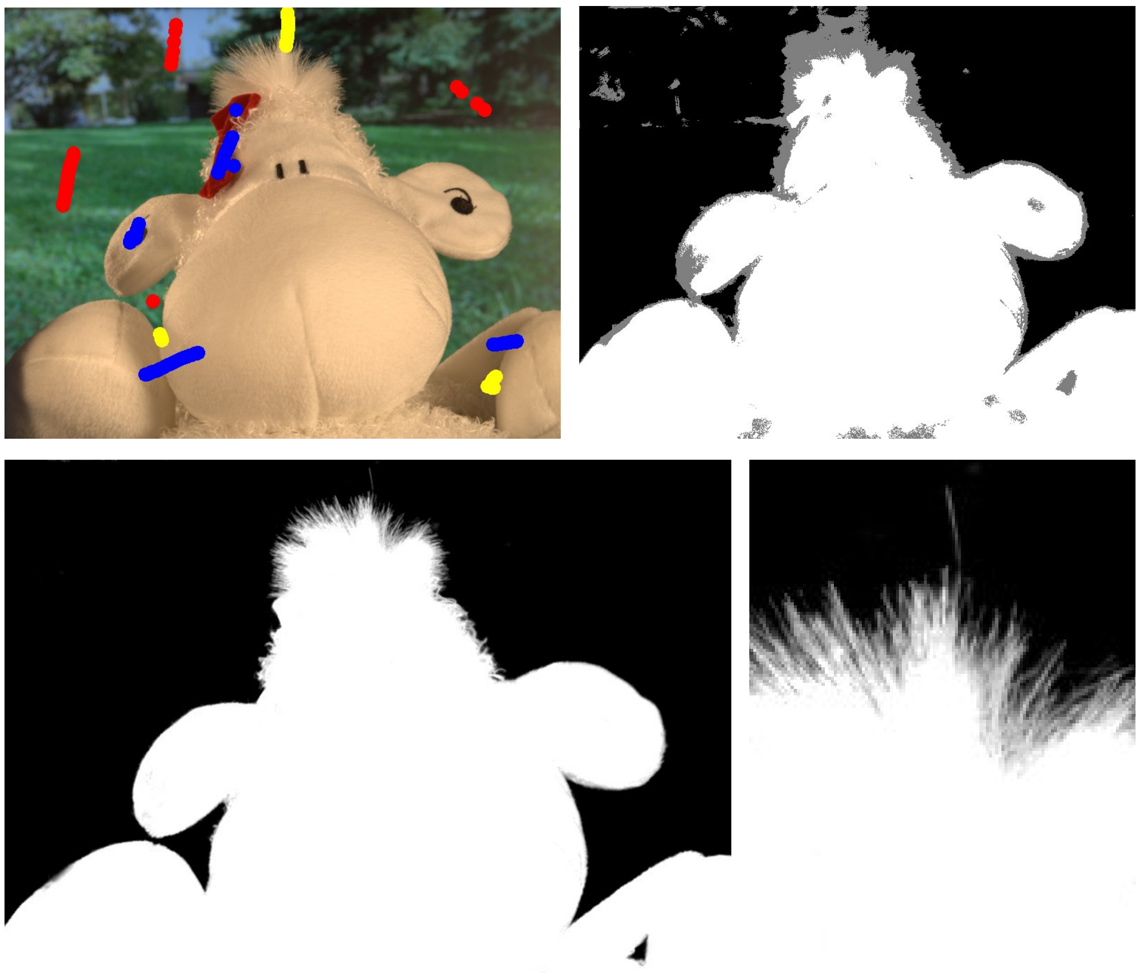


Figure 6.7: Use of the proposed real-time technique for interactive segmentation and composition of images by alpha matting. As the user scribbles over the image (top-left), a trimap is automatically updated, providing instant feedback on the resulting segmentation. The resulting trimap and alpha matte computed from the set of scribbles on the top-left are shown on the top-right and bottom-left images, respectively.

6.3.2 Interactive Alpha Matting

Another benefit of the improved performance of the proposed technique is its ability to provide real-time feedback to users during interactive alpha-matting extraction sessions. We demonstrate this feature using a simple trimap creation interface. Initially, all pixels in the image are labeled as belonging to the unknown region T_u. As one uses small scribbles over the image, a trimap is automatically computed and refined, also providing instant feedback on the resulting matte extraction. The scribbles are propagated using an iterative flood-filling procedure, limited by a simple edge detector. Figure 6.7 illustrates the concept using one of the images of the training dataset. On the top-left, one sees the scribbles superimposed onto the original image. The blue color is a label for foreground pixels, while red and yellow represent background and unknown pixels, respectively. The image on the top-right shows the computed trimap, for which a gray shade indicates uncertainty about whether a pixel belongs to the background (shown in black) or to the foreground (shown in white). The extracted alpha matte is shown on the bottom-left, with a detail crop on the bottom-right.

The speed of the proposed method makes the matte creation process much easier for the user, as there is no delay between input and matte refinement. This considerably reduces the time taken to interactively segment images by alpha matting. The resulting mattes are generated considerably faster for images with complex edges and topologies. Only in the worst case one needs to completely trace the border of the foreground object, which is always needed in the technique described in (WANG; AGRAWALA; COHEN, 2007). Furthermore, the simple trimap generation technique presented here was a proof of concept, thus leaving to be explored the use of more advanced techniques for interactive and real-time trimap generation.

7 CONCLUSIONS AND FUTURE WORK

This work presented the first real-time matting technique for natural images and videos. The proposed technique is based on the observation that pixels in a small neighborhood in image space tend to have highly similar values for (α, F, B) triplets. As such, independently computing such triplets for each pixel in the unknown region in a conventional way tends to result in a lot of redundant work. The amount of required computation can then be significantly and safely reduced by a careful selection of pairs of candidate background and foreground samples. The work required to perform this task can be distributed among neighbor pixels, leading to considerable savings in computation cost. Moreover, the required operations can be performed in parallel, allowing us to exploit the benefits of programmable GPUs.

To obtain these results, this work proposes a new objective function for identifying good pairs of background and foreground samples. Such a function takes into account spatial and photometric, as well as some probabilistic information extracted from the image. This improved objective function allows the proposed approach to achieve high-quality results while still operating on a considerably small discrete search space. The proposed approach can achieve speedups of up to two orders of magnitude compared to previous approaches, while producing highly-accurate alpha mattes. The quality of the generated results was assessed by performing the independently developed benchmark by Rhemann *et al.* (RHEMANN et al., 2009). In such a benchmark, the proposed real-time matting technique ranked second among current techniques. However, the proposed technique is up to 100 times faster than the state-of-the-art, allowing for real-time alpha matting. Additionally, the proposed technique can be extended with an extra optimization step, resulting in the best results ever achieved in Rhemann *et al.*'s benchmark. The extended technique ranks first among all techniques.

We have demonstrated that our technique can provide instant feedback to support interactive extraction of high-quality alpha mattes. Our technique is also fast enough to, for the first time, support alpha-matte computation for videos in real-time, given that the corresponding trimaps are provided. This opens up exciting opportunities for new real-time applications and for improved real-time trimap generation for videos. The works described in (BAI et al., 2009) and (BAI; SAPIRO, 2007) are promising steps toward fast binary segmentation, which can be used for trimap estimation. Finally, the problem of handling temporal coherence for real-time matte remains to be explored.

APPENDIX A RELATIVE ERROR TABLES

Sum of Absolute Differences	**overall rank**	**Troll**			**Doll**			**Donkey**			**Elephant**		
		small	large	user	small	large	user	small	large	user	small	large	user
Improved Sampling Matting	1.14	**1.00**	1.02	**1.00**	1.44	1.40	1.02	**1.00**	**1.00**	**1.00**	1.09	1.52	1.03
Improved color matting	1.15	1.42	1.21	1.35	1.14	1.12	**1.00**	1.10	1.13	1.10	1.18	1.17	1.06
Improved Sampling Matting (Real-Time)	1.27	1.23	1.21	1.18	1.68	1.65	1.20	1.05	1.04	1.08	1.23	1.35	1.13
Closed-Form Matting	1.30	1.21	1.08	1.16	**1.00**	**1.00**	1.01	1.12	1.11	1.10	**1.00**	**1.00**	1.03
Robust Matting	1.43	1.65	1.40	1.43	1.71	1.99	1.34	1.14	1.20	1.28	1.27	1.59	1.38
High-res matting	1.56	1.77	1.27	1.66	1.46	1.66	1.31	1.19	1.15	1.23	1.14	1.80	**1.00**
Random Walk Matting	1.97	1.70	**1.00**	1.31	1.92	1.84	1.39	1.38	1.30	1.62	1.55	1.46	1.44
Iterative BP Matting	2.08	2.25	1.47	1.84	2.83	2.86	2.44	1.60	1.67	1.62	1.73	2.46	2.13
Geodesic Matting	2.13	2.56	1.90	2.20	2.41	1.94	2.05	2.79	2.59	2.41	3.45	3.28	2.72
Easy Matting	2.28	2.28	1.61	2.03	2.90	2.56	2.28	1.50	1.39	1.49	2.14	2.28	1.75
Bayesian Matting	2.60	2.89	2.09	2.26	3.25	3.04	2.16	2.57	2.30	2.77	3.00	4.02	1.94
Poisson Matting	3.84	4.93	2.77	3.51	4.80	5.12	3.61	2.88	2.54	2.36	5.32	4.00	3.50

Sum of Absolute Differences	**overall rank**	**Plant**			**Pineapple**			**Plastic Bag**			**Net**		
		small	large	user	small	large	user	small	large	user	small	large	user
Improved Sampling Matting	1.14	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	1.59	1.36	1.54	1.12	1.24	1.03
Improved color matting	1.15	1.44	1.16	1.26	1.25	1.02	1.27	1.15	1.02	**1.00**	**1.00**	**1.00**	**1.00**
Improved Sampling Matting (Real-Time)	1.27	1.17	1.19	1.15	1.17	1.05	1.05	1.63	1.40	1.64	1.32	1.49	1.21
Closed-Form Matting	1.30	1.79	1.42	1.95	1.73	1.51	2.03	1.51	1.24	1.16	1.11	1.00	1.81
Robust Matting	1.43	1.40	1.65	1.83	1.42	1.47	1.61	**1.00**	**1.00**	1.36	1.45	1.45	1.42
High-res matting	1.56	1.50	1.65	2.16	1.77	1.83	1.85	1.56	1.46	1.81	1.63	2.13	1.38
Random Walk Matting	1.97	2.52	2.60	2.77	2.56	1.82	2.38	1.94	1.67	1.74	3.16	3.20	3.02
Iterative BP Matting	2.08	2.71	2.68	2.82	2.38	1.92	2.23	1.47	1.51	2.01	1.71	1.88	1.69
Geodesic Matting	2.13	2.46	1.96	1.53	1.52	1.22	1.48	1.64	1.43	1.81	2.04	1.95	1.82
Easy Matting	2.28	2.33	1.85	2.31	2.33	1.72	2.24	2.18	1.90	1.96	3.27	4.24	4.09
Bayesian Matting	2.60	2.73	3.51	3.35	3.21	3.09	2.98	1.58	1.56	1.68	1.90	3.00	1.63
Poisson Matting	3.84	4.31	4.33	5.61	4.46	3.25	3.44	2.36	2.79	2.47	5.27	3.31	5.23

Table A.1: SAD Relative error for all images in the test dataset. See section 6.1.2.

Mean Squared Error	**overall rank**	**Troll**			**Doll**			**Donkey**			**Elephant**		
		small	large	user	small	large	user	small	large	user	small	large	user
Improved Sampling Matting	1.28	**1.00**	1.55	**1.00**	2.00	2.25	1.20	**1.00**	**1.00**	**1.00**	**1.00**	1.67	**1.00**
Improved color matting	1.35	2.00	2.18	1.88	**1.00**	1.25	**1.00**	1.50	1.33	1.50	**1.00**	**1.00**	**1.00**
Improved Sampling Matting (Real-Time)	1.80	1.75	2.27	1.63	2.67	3.25	1.80	1.50	1.33	1.50	2.00	2.00	1.50
Closed-Form Matting	1.83	1.25	1.64	1.38	**1.00**	**1.00**	1.20	1.50	1.33	1.50	**1.00**	**1.00**	**1.00**
Robust Matting	1.95	2.75	2.55	2.13	2.33	3.75	1.80	1.50	1.33	1.50	**1.00**	1.67	1.50
High-res matting	2.21	3.25	2.00	2.75	1.67	2.75	1.60	1.50	1.33	1.50	**1.00**	2.33	**1.00**
Iterative BP Matting	3.16	4.25	2.36	2.87	5.00	6.50	4.60	2.50	2.33	2.00	2.00	2.67	2.00
Random Walk Matting	3.73	2.50	**1.00**	1.50	3.33	4.25	2.20	2.50	2.00	3.00	2.00	1.33	1.50
Geodesic Matting	4.38	6.00	4.18	4.25	5.00	4.50	3.80	8.00	7.00	5.50	8.00	6.33	4.50
Bayesian Matting	5.50	7.50	4.18	4.25	7.67	8.00	4.20	7.00	5.00	6.00	7.00	7.00	3.00
Easy Matting	5.55	5.50	3.36	4.12	8.00	8.00	5.80	3.50	3.00	3.00	5.00	4.67	3.00
Poisson Matting	10.65	17.25	6.82	8.87	15.67	19.25	10.60	8.50	6.33	5.50	16.00	8.33	7.00

Mean Squared Error	**overall rank**	**Plant**			**Pineapple**			**Plastic Bag**			**Net**		
		small	large	user	small	large	user	small	large	user	small	large	user
Improved Sampling Matting	1.28	**1.00**	**1.00**	**1.00**	**1.00**	1.14	**1.00**	2.00	1.56	2.14	**1.00**	1.25	**1.00**
Improved color matting	1.35	2.33	1.40	1.13	1.33	**1.00**	1.40	1.33	1.06	**1.00**	1.18	1.25	1.25
Improved Sampling Matting (Real-Time)	1.80	1.33	1.60	1.38	1.67	1.43	1.40	2.13	1.72	2.43	1.55	2.00	1.42
Closed-Form Matting	1.83	4.00	2.80	2.87	2.67	2.29	3.20	2.00	1.50	1.36	1.18	**1.00**	4.17
Robust Matting	1.95	1.67	2.40	2.37	1.67	2.14	2.40	**1.00**	**1.00**	1.86	2.18	1.92	2.42
High-res matting	2.21	2.00	2.40	2.75	2.67	2.86	2.80	2.13	1.89	3.00	2.36	3.58	1.83
Iterative BP Matting	3.16	3.67	4.00	3.88	3.33	2.86	3.20	1.87	1.83	3.21	2.73	3.17	3.00
Random Walk Matting	3.73	6.67	6.80	5.25	5.33	3.29	4.20	3.07	2.44	2.86	7.55	7.83	7.08
Geodesic Matting	4.38	6.00	5.00	2.75	2.67	2.00	2.20	2.20	1.78	2.93	3.45	3.58	3.50
Bayesian Matting	5.50	6.00	8.40	6.50	7.33	6.57	5.80	2.27	2.17	2.79	3.36	7.17	2.92
Easy Matting	5.55	6.00	5.00	5.12	5.67	3.86	4.80	3.60	3.00	3.36	9.09	13.25	13.58
Poisson Matting	10.65	11.67	12.60	14.38	12.00	8.14	7.80	4.07	5.22	4.86	17.64	9.17	18.00

Table A.2: MSE Relative error for all images in the test dataset. See section 6.1.2.

APPENDIX B REAL-TIME BINARY SEGMENTATION UNDER VARIABLE LIGHTING CONDITIONS

A fast binary segmentation is essential to generate trimaps for use in real-time alpha-matting applications. Although various techniques have been proposed for binary segmentation of constant-color and multicolored static backgrounds, these approaches are sensitive to changes in lighting conditions, requiring new calibration and/or changes in their parameter values to adjust to these new conditions. Often, such limitations lend to misclassification of background pixels subject to shading variations, such as shadows cast by foreground objects. We present an improved model for binary segmentation that is robust to changes in lighting conditions without new calibration, can be evaluated in real time and supports moving cameras with multicolored backgrounds.

The technique presented here is divided in two steps: (i) *background modeling*, evaluated offline; and (ii) *background segmentation*, performed by testing all image pixels against the *background model* — computed in the first step — in real-time.

B.1 Background Modeling

The background model is constructed offline by sampling the background under relevant lighting conditions and estimating its *probability density function* B in the RGB color space. Thus, for $x_i = (r_i, g_i, b_i),\ \{r, g, b\} \in [0, 1]$, let $x_1, x_2, \ldots, x_n \sim B$ be an independent and identically-distributed sample of the background population. We estimate B with a *kernel density approximation* as:

$$B(r,g,b) \approx \frac{1}{n}\sum_{i}^{n} G(\|(r,g,b) - b_i\|) \tag{B.1}$$

where $\|\cdot\|$ is the Euclidean norm and G is a gaussian kernel with variance $\sigma^2 = 2 \times 10^{-4}$ (RGB units). The final background model is represented as the 3D isosurface $S(c)$ defined by $B(r, g, b) = c$, where $c = x$ is the maximum possible value for which the isosurface $S(x)$ contains no more than 99.9%[1] of the initial samples b_i. Figure B.1 shows a initial collection of samples from a multicolored background and its corresponding isosurface.

[1]This percentage expresses our confidence in the gathered samples, and needs to be smaller if too much noise is present.

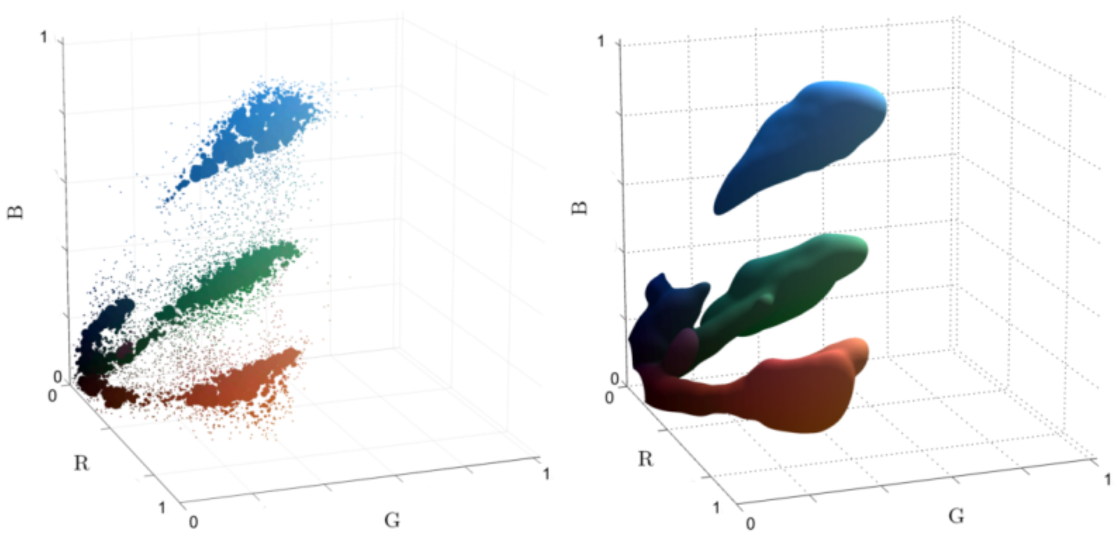


Figure B.1: Example of isosurface for a multicolored background. The initial collection of background samples (left) is accurately modeled by the isosurface S (right).

B.2 Background Segmentation

The value of the binary segmentation mask $M(p)$ for each pixel p in the input image (or video frame) is computed as:

$$M(p) = \begin{cases} 0 & \text{if } C_p \text{ is inside } S(c) \\ 1 & \text{if } C_p \text{ is not inside } S(c) \end{cases} \tag{B.2}$$

where $C_p = (r_p, g_p, b_p)$ is the color of p. In practice, we substitute the test "*is p inside $S(c)$?*" with a table lookup indexed by C_p. For this, the RGB color space is discretized into a $256 \times 256 \times 256$ cube, and the test in Equation B.2 is precomputed into a 3D lookup table H. Thus, the actual computation of the binary mask M is reduced to:

$$M(p) = H(r_p, g_p, b_p) \tag{B.3}$$

B.3 Results

The simplicity in the background model representation makes this a fast and efficient technique. Equation B.3 can be evaluated in parallel for all pixels in the input image, thus is a great candidate for a GPU implementation. We implemented this technique with C++ and GLSL and achieved more than $3,000$ frames per second for images of size 800×600 in a PC with a GeForce 8800GTS 512MB GPU and a 3.2GHz CPU. We note, however, that such an application is CPU-bound due to the extremely small computation done on the GPU.

The real-time binary segmentation technique presented here works for heterogeneous and multicolored backgrounds, while being resistant to lighting variations (without re-calibration) and allowing for a freely moving camera. We evaluate our results against the ones obtained by the technique of Fernandes *et al.* (2006), which works under the same environment conditions. Figure B.2 shows five images (left column) and their corresponding binary segmentation masks generated by Fernandes *et al.*'s technique (center column) and our technique (right column). For all these images, the ideal binary segmentation mask would have in white all (and only) pixels belonging to foreground objects. The first two rows show the robustness of the proposed technique against lighting variations, where the binary masks where generated with the same background model, without re-calibration. The last two rows show heterogeneous background segmentation, where the proposed technique generates binary masks with much less noise while preserving fine details.

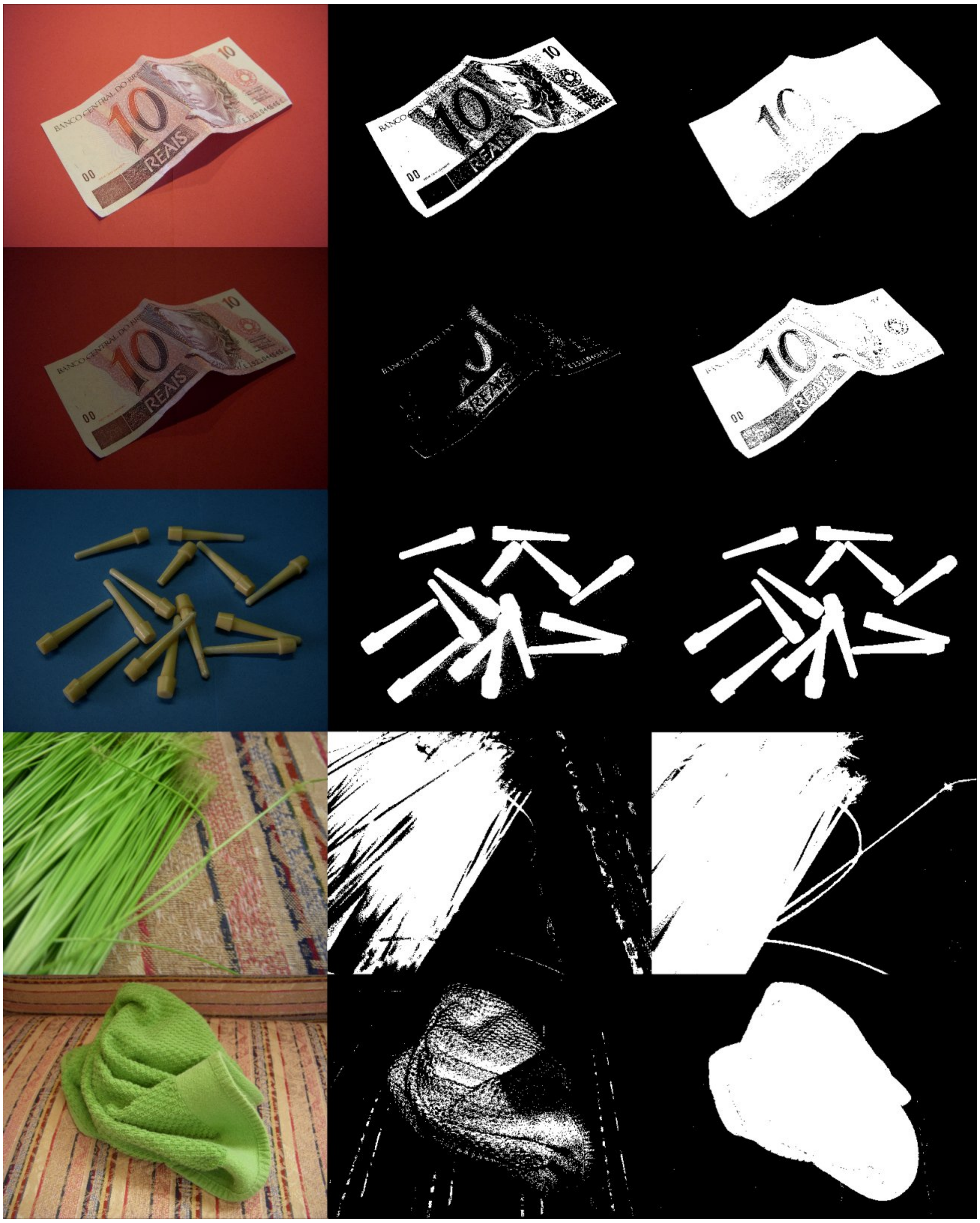


Figure B.2: Evaluation of the results obtained by the proposed binary segmentation technique against the ones from Fernandes *et al.* (2006). Five images are shown (left column) with their corresponding binary segmentation masks generated by Fernandes *et al.*'s technique (center column) and our technique (right column). For all these images, the ideal binary segmentation mask would have in white all (and only) pixels belonging to foreground objects.

REFERENCES

BAI, X.; SAPIRO, G. A Geodesic Framework for Fast Interactive Image and Video Segmentation and Matting. **Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on**, [S.l.], v.1, p.1–8, Oct. 2007.

BAI, X.; WANG, J.; SIMONS, D.; SAPIRO, G. Video SnapCut: robust video object cutout using localized classifiers. **ACM Trans. Graph.**, New York, NY, USA, v.28, n.3, p.1–11, 2009.

BOYKOV, Y.; KOLMOGOROV, V. **Computing geodesics and minimal surfaces via graph cuts**. 2003. 26–33p.

CHUANG, Y.-Y.; CURLESS, B.; SALESIN, D. H.; SZELISKI, R. A Bayesian Approach to Digital Matting. In: IEEE CVPR 2001, 2001. **Proceedings...** IEEE Computer Society, 2001. v.2, p.264–271.

COREL CORPORATION. **Knockout user guide**. 2002.

FERNANDES, L.; OLIVEIRA, M.; SILVA, R. da; CRESPO, G. A fast and accurate approach for computing the dimensions of boxes from single perspective images. **J. of the Brazilian Computer Society**, [S.l.], v.12, n.1, 2006.

GRADY, L.; SCHIWIETZ, T.; AHARON, S.; WESTERMANN, R. Random Walks for Interactive Alpha-Matting. In: FIFTH IASTED INTERNATIONAL CONFERENCE ON VISUALIZATION, IMAGING AND IMAGE PROCESSING, 2005, Benidorm, Spain. **Proceedings...** ACTA Press, 2005. p.423–429.

GUAN, Y.; CHEN, W.; LIANG, X.; DING, Z.; PENG, Q. **Easy matting-a stroke based approach for continuous image matting**. 2006. 567–576p. v.25, n.3.

JOSHI, N.; MATUSIK, W.; AVIDAN, S. **Natural video matting using camera arrays**. 2006. 786p.

JUAN, O.; KERIVEN, R. Trimap segmentation for fast and user-friendly alpha matting. In: IEEE WORKSHOP ON VARIATIONAL, GEOMETRIC & LEVEL SET METHODS, 2005. **Anais...** [S.l.: s.n.], 2005.

KIM, K.; CHALIDABHONGSE, T.; HARWOOD, D.; DAVIS, L. **Background modeling and subtraction by codebook construction**. 2004. v.5.

LEVIN, A.; LISCHINSKI, D.; WEISS, Y. A Closed-Form Solution to Natural Image Matting. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, Los Alamitos, CA, USA, v.30, n.2, p.228–242, 2008.

LEVIN, A.; RAV-ACHA, A.; LISCHINSKI, D. Spectral Matting. **IEEE Trans. Pattern Anal. Mach. Intell.**, Washington, DC, USA, v.30, n.10, p.1699–1712, 2008.

MCGUIRE, M.; MATUSIK, W.; PFISTER, H.; HUGHES, J. F.; DURAND, F. Defocus video matting. **ACM Trans. Graph.**, New York, NY, USA, v.24, n.3, p.567–576, 2005.

MISHIMA, Y. **Soft edge chroma-key generation based upon hexoctahedral color space**. US Patent 5,355,174.

PORTER, T.; DUFF, T. Compositing digital images. **ACM SIGGRAPH Computer Graphics**, [S.l.], v.18, n.3, p.253–259, 1984.

RHEMANN, C.; ROTHER, C.; GELAUTZ, M. Improving Color Modeling for Alpha Matting. In: BRITISH MACHINE VISION CONFERENCE 2008, 2008. **Proceedings...** [S.l.: s.n.], 2008. p.1155–1164. Vortrag: British Machine Vision Conference BMVC 2008, Ledds, UK; 2008-09-01.

RHEMANN, C.; ROTHER, C.; RAV-ACHA, A.; SHARP, T. High resolution matting via interactive trimap segmentation. **Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on**, [S.l.], v.1, p.1–8, June 2008.

RHEMANN, C.; ROTHER, C.; WANG, J.; GELAUTZ, M.; KOHLI, P.; ROTT, P. **A Perceptually Motivated Online Benchmark for Image Matting**. 2009.

RUZON, M.; TOMASI, C. **Alpha estimation in natural images**. 2000. v.1.

SINGARAJU, D.; ROTHER, C.; RHEMANN, C. **New Appearance Models for Natural Image Matting**. 2009.

SMITH, A. R.; BLINN, J. F. Blue screen matting. In: SIGGRAPH '96: PROCEEDINGS OF THE 23RD ANNUAL CONFERENCE ON COMPUTER GRAPHICS AND INTERACTIVE TECHNIQUES, 1996, New York, NY, USA. **Anais...** ACM, 1996. p.259–268.

SUN, J.; JIA, J.; TANG, C.-K.; SHUM, H.-Y. Poisson matting. **ACM Trans. Graph.**, New York, NY, USA, v.23, n.3, p.315–321, 2004.

SUN, J.; LI, Y.; KANG, S.; SHUM, H. Flash matting. **ACM Transactions on Graphics (TOG)**, [S.l.], v.25, n.3, p.778, 2006.

SUN, J.; ZHANG, W.; TANG, X.; SHUM, H. yeung. Background cut. In: IN ECCV, 2006. **Anais...** [S.l.: s.n.], 2006. p.628–641.

VLAHOS, P. **Composite Color Photography**. US Patent 4,007,487.

WANG, J.; AGRAWALA, M.; COHEN, M. F. Soft scissors: an interactive tool for real-time high quality matting. In: SIGGRAPH '07: ACM SIGGRAPH 2007 PAPERS, 2007, New York, NY, USA. **Anais...** ACM, 2007. p.9.

WANG, J.; COHEN, M. **An iterative optimization approach for unified image segmentation and matting**. 2005. v.2.

WANG, J.; COHEN, M. **Image and video matting**: a survey. [S.l.]: Now Pub, 2008.

WANG, J.; COHEN, M. F. Optimized Color Sampling for Robust Matting. **Computer Vision and Pattern Recognition, IEEE Computer Society Conference on**, Los Alamitos, CA, USA, v.0, p.1–8, 2007.

YANG, C.; DURAISWAMI, R.; GUMEROV, N. A.; DAVIS, L. Improved Fast Gauss Transform and Efficient Kernel Density Estimation. In: ICCV '03: PROCEEDINGS OF THE NINTH IEEE INTERNATIONAL CONFERENCE ON COMPUTER VISION, 2003, Washington, DC, USA. **Anais...** IEEE Computer Society, 2003. p.464.