STUDY AND IMPLEMENTATION OF IMAGE AND VIDEO MATTING TECHNIQUES

(SIDDHARTH SRIVASTAVA, ADITYA RASTOGI, JASDEEP SINGH KHURANA, SHAILI BHATI)

University School of Information Technology, GGS Indraprastha University, Kashmere Gate, Delhi 110006

Contents

INTRODUCTION **5**
2.1) Binary Segementation and Matting 5
VIDEO MATTING **8**
2.1) Challenges in Video Matting: 8
2.2) Techniques used for Video Matting 9
2.3) Interpolating Trimaps Using Optical Flow 9
2.4) Bayesian Video Matting 10
2.5)Rotoscoping for Trimap Generation 11
TRIMAP GENERATION **14**
3.1)What is trimap? 14
3.2) Trimap Generation Techniques 15
3.3) Background Subtraction Methods 15
3.4) Optical Flow Based Methods 16
3.5) Using Optical Flow to calculate Trimap and Alpha matte 17
MATTING TECHNIQUES **21**
4.1) Sampling Techniques 21
4.1.1) Ruzan and Tomasi 21
4.1.2) Byesian Matiing 23
4.2) Non-Sampling Techniques 216
4.2.1) Poison Matting 216
4.2.2) Spectral Matting 30
COMPOSITING **34**
5.1) Compositing to get final image 35
5.2) Compositing in Videos 37
TEST CASES **38**
MATLAB CODE **43**
FUTURE DIRECTIONS AND CONCLUSIONS **53**
8.1) Limitations of Current Approaches 54
8.2 Future Directions 54
8.3 Learning/Example-based Approaches 55
8.4 More Practical Video Matting Systems 56
REFERENCES **57**

STUDY AND IMPLEMENTATION OF image and video matting techniques

ABSTRACT

Matting is the technique of estimating accurate foreground in images and video.

Both image and video matting are very important in terms of both commercial and technological advances. Various television advertisements, educational videos etc. these days use matting techniques to minimize the cost of re-recording the videos or re-capturing the snapshots, instead there is a need of efficient matting techniques to perform these tasks swiftly. It also gives power to insert new elements in a scene seamlessly or transport an entity into a different environment in order to create novel visual artifacts.

With the recent advances of digital cameras, using matting techniques to create novel composites or facilitate other editing tasks has gained increasing interests from both professionals as well as consumers. Consequently, various matting techniques and systems have been proposed to try to efficiently extract high quality mattes from both still images and video sequences.

The report consists of various matting techniques that have been proposed till date which have relevance in video matting.

Our Aim is to introduce *Video Matting in Real Time Video Conferencing* both for desktop computers and mobile devices such as Cell Phones and Tablets.

The common approaches available till date requires manual intervention and are also computationally expensive, hence, for our application, we needed an approach which is not only independent of human intervention but also is computationally effective to handle Real-Time data.

The difficulties and challenges of video matting are first analyzed, and various ways if combine matting algorithms with other video processing techniques for building efficient video matting systems are reviewed.

Key contributions, advantages as well as limitations of important systems are summarized.

INTRODUCTION

INTRODUCTION

Extracting foreground objects from still images or video sequencesplays an important role in many image and video editing applications.Accuratelyseparating a foreground object from the background involvesdetermining both full and partial pixel coverage, also known as *pulling amatte*, or *digital matting*.

In digital matting, a foreground element is extracted from a background image by estimating a color and opacity for the foreground element at each pixel. The opacity value at each pixel is typically called its alpha, and the opacity image, taken as a whole, is referred to as the alpha matte or key.

The problem was first studied mathematically by Porter and Duff in 1984. They introduced alpha channel as the basis for linear interpolation when rendering foreground over arbitrary background.

Mathematically, the observed image is modeled as:

$$I_z = \alpha_z F_z + (1 - \alpha_z) B_z$$

Where:

$I_z$is the observed image ($z = (x,y)$)

$\alpha_z$is the alpha matte

F_z and $B_z$represent the foreground and background images respectively.

The alpha matte can take any value in [0,1]. If it is 1 then pixel z is called definitely foreground, and if it is 0 then pixel z is called definitely background. In other cases z is called mixed.

So the problem lies is in estimating these values in the images and video frames i.e. estimating the foreground, background and mixed pixels.

The known information for an input image are the three dimensional color vector I_z and the unknown variables are three dimensional F_z and B_z, and the scalar vector α_z.

Hence, matting is an under-constrained problem where 7 unknown variables need to be determined from 3 known variables. This results in requirement of user intervention or prior assumptions on image statisticsfor determining these unknown variables to calculate an accurate matte from the input image. Once the matte is estimated correctly, the background can be replaced by the new background seamlesslyby replacing the new background with B_z in the alpha matte equation.

Most recent methods expect the user to provide a *trimap*as a starting point.Thetrimap is a rough (typically hand-drawn) segmentation of the image into three regions: foreground (shown in white), background (shown in black) and unknown (shown in gray). Given the trimap, these methods typically solve for F, B,and α simultaneously.

BINARY SEGMENTATION VS MATTING

If alpha value is constrained to be 0 or 1, then the problem of matting is reduced to binary image/video segmentation, where each pixel fully belongs to foreground or background. Binary segmentation serves as an initial starting point for video matting techniques. But matting is more complex than the problem of binary segmentation and is more general and hence delivers results which are closer to the user desired results.

Although matting is modelled as a more general problem than binary segmentation, which is theoretically harder to solve, most existing matting algorithms avoid the segmentation problem by having the trimap as another input in addition to the original image. The trimap may be manually specified by the user, or produced by other binary segmentation approaches. The trimap reduces the dimension of the solution space of the matting problem, and leads the matting algorithms to generate user-desired results.

VIDEO MATTING

VIDEO MATTING

VIDEO MATTING is the technique of pulling a matte from a video sequence of a dynamic foreground element against a natural background. Video Matting to some extent requires the techniques used for Image matting but is generally much harder to implement.

2.1)Challenges in Video Matting:

1) Large Data Set: The algorithms must be able to efficiently process large number of pixels in a video sequence as well as they must be fast enough so that interface remains responsive.
2) Temporal Coherence: The algorithms should maintain the temporal coherence among the frames of the video sequence. Applying the image matting techniques directly on each frame of the video, results in temporal inconsistency.
3) Fast motion vs Low temporal resolution: Generally video frames are shot at 30 frames per second, thus the sampling rate is much less than ideal and hence it becomes difficult to deal with fast motions. The algorithm should be able to maintain an inter-frame correspondence in this case.

To alleviate these challenges, most of the video matting algorithms use the following abstract method:

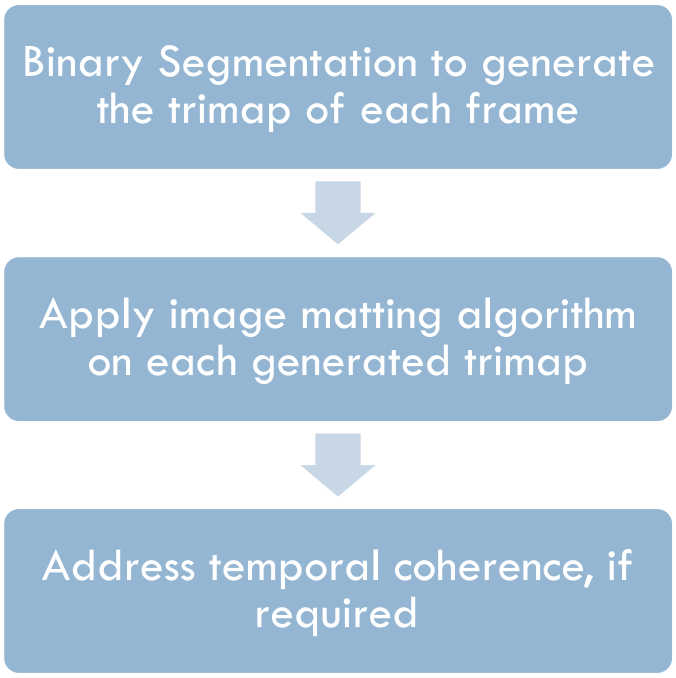

Binary Segmentation to generate the trimap of each frame
Apply image matting algorithm on each generated trimap
Address temporal coherence, if required

FIGURE 1FLOW CHART FOR GENERAL VIDEO MATTING APPROACH

2.2)Techniques used for Video Matting

A number of techniques have been proposed in existing video matting approaches to alleviate the difficulties and leverage the advantages. To deal with large data size, most approaches adopt a two-step framework. In the first step, only binary segmentation is solved to generate a trimap for each frame. Given the trimaps, matting algorithms are then applied in the second step to refine the foreground boundary. Since only binary segmentation is considered in the first step, these approaches can give users rapid response through various user interfaces. Once accuratetrimaps are generated, image matting algorithms can then be applied offline on each frame to generate the final fine mattes. To address the importance of temporal coherence, instead of creating trimaps on video frames independently, most approaches create trimaps in a temporally coherent way, by performing spatio-temporal optimizations. This also allows trimaps to be propagated from a limited number of user defined key frames to the entire sequence, resulting in significantly reduced user input.

2.3) Interpolating Trimaps Using Optical Flow

As a widely used technique for estimating the inter-frame motion at each pixel in a video sequence, optical flow has been used in the Bayesian video matting system for trimap propagation. The basic idea is to ask the user to specify trimaps on a few key frames in the input sequence, and then use optical flow to propagate trimaps to all other frames.

To ensure the trimap propagation is stable, a set of supplemental methods have been proposed.

- In the first step, the system requires the user to specify some ”garbage mattes” to eliminate the foreground on some frames. This allows a dynamic clean background plate to be reconstructed from a composite mosaic of the remaining backgrounds in each frame. Both trimap propagation and matte estimation can leverage the constructed background plate.

- In the second step, bi-directional (forward and backward) flow is computed to guide the user-specified trimaps from keyframes to inter- mediate frames. Specifically, from frame t, a forward flow is computed from the previous keyframetk to t, along with an accumulated errormapEtk . The trimapMtk is then warped to frame t as Mf . A backward flow from the next key frame tk+1 is computed is a similar way, resulting an accumulated error map Etk+1 and a warped trimap Mb . The two warped trimaps are then unified by choosing the assignment with smaller error at each pixel. If the two error maps are large at a particular pixel, the pixel is then either set to be unknown, or to be background if its color is close enough to the color in the background plate computed in the first step.

- In the third step, Bayesian matting is applied on each frame with the propagated trimap. The estimated background plate is also used to both improve the quality of the matte and speed up the matting process

2.4)Bayesian Video Matting

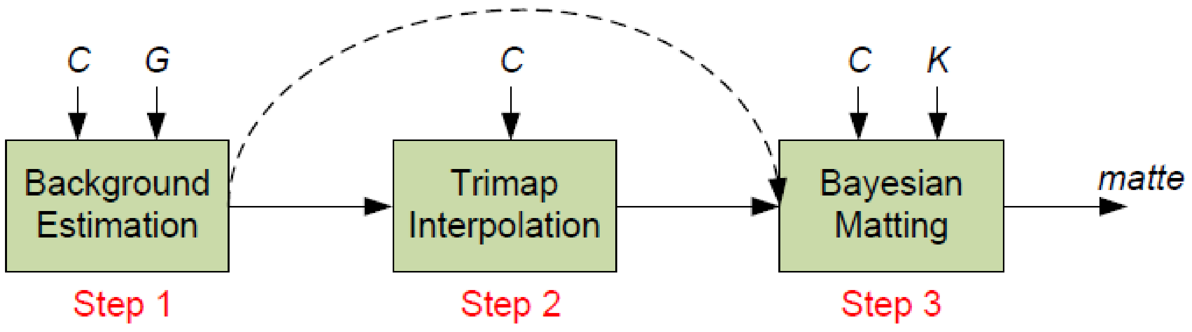


FIGURE 2 BAYESIAN VIDEO MATTING

This is a flow chart of Bayesian Video matting which depends upon the Bayesian Image Matting.Here C represents the input sequence, K is user selected keyframes and G stands for garbage trimaps.The Bayesian video matting works by first asking the user to specify trimaps on a few keyframes on the input sequence. The trimaps are propagated to other frames by applying optical flow.

The steps followed in Bayesian Video Matting are as follows:

(i) The system requires user to specify some garbage mattes to eliminate foreground on some frames. This results in creation of a dynamic clean background plate.

(ii) The bidirectional flow is computed to guide the user-specified trimaps from keyframes to intermediate frames.

(iii) Now, the Bayesian Matting is applied on each frame with the propagated trimap.The estimated background plate is also used to improvise the quality of matte and speed up the matting process.

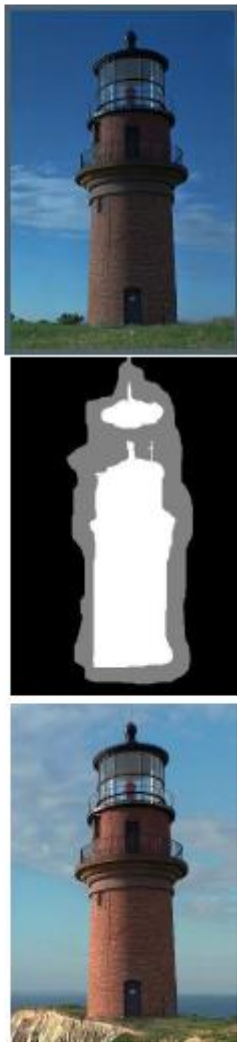


FIGURE 3 BAYESIAN MATTING SAMPLE IMAGES

The above image sequence shows the application of Bayesian Matting. The first image is the original image while the second image is the trimap calculated by interpolating the trimap supplied by the user. The third snapshot shows the lighthouse in the original image but with a different background.

2.5)Rotoscoping for Trimap Generation

Another commonly used production matting technique is rotoscoping. In traditional film production, rotoscoping often refers to the process of manually tracing shapes, performed one frame at a time, through a captured image sequence. Recently, optimization techniques have been introduced into rotoscoping process and a keyframe-based rotoscoping system has been proposed , which significantly reduces the amount of human effort involved for tracking shapes.

The process starts by having the user select two keyframesta andtb ,and draw a set of curves indicating object boundaries on them. Curves are specified by placing control points of a piecewise cubic B´ezier curve with C0 continuity. The curves drawn on ta can be copied to tb and modified to fit the new object position.

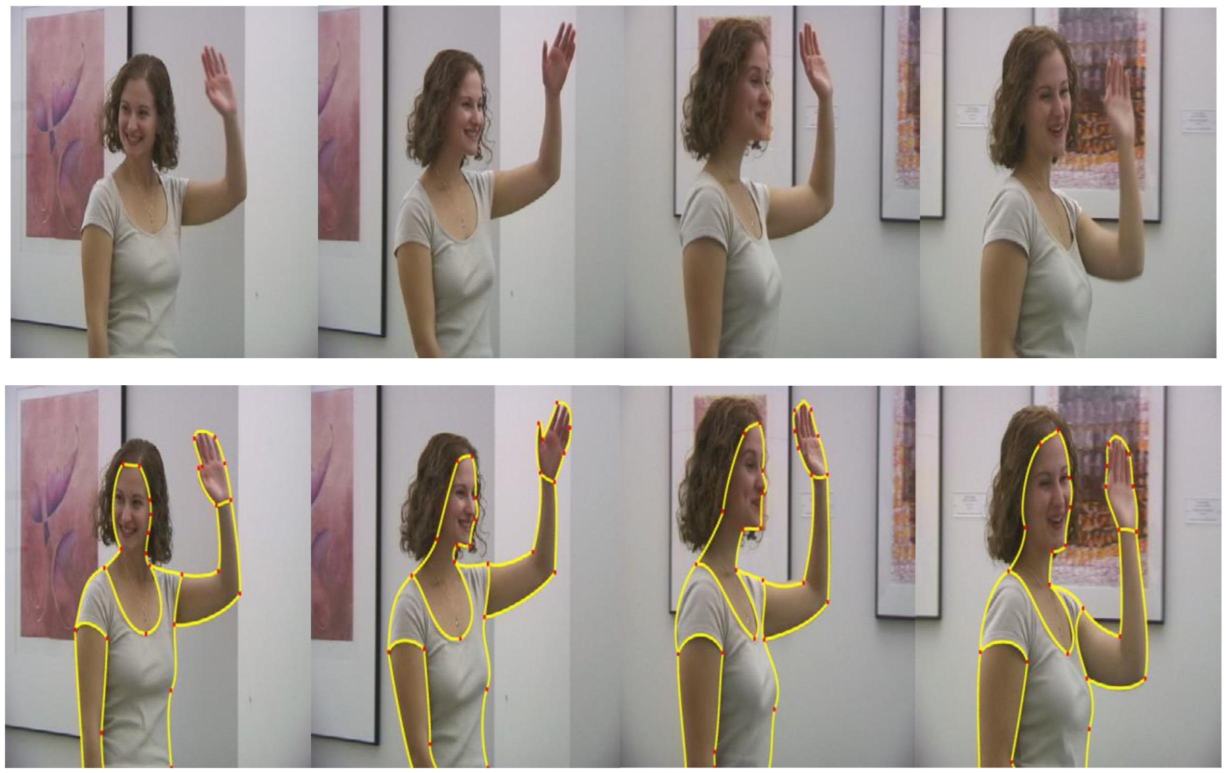


FIGURE 4 EXAMPLES OF ROTOCURVES

TRIMAP GENERATION

TRIMAP GENERATION

The trimap generation is the most important part of the entire process of matting. The Trimap serves as the first guide to accurately computing the final matte that separates the foreground and the background.

One of the important factors effecting the performance of a matting algorithm is how accurate the trimap is. Ideally, the unknown region in the trimap should only cover truly mixed pixels. In other words, the unknown region around the foreground boundary should be as thin as possible to achieve the best possible matting results. the less number of unknown variables need to be estimated, and the more known foreground and background information is available to use. However, accurately specifying a trimap requires significant amounts of user effort and is often undesirable in practice, especially for objects with large semitransparent regions or holes. Thus a big challenge for designing a successful matting algorithm is how to achieve a good trade-off between the accuracy of the matte and the amount of the user effort required.

3.1)What is trimap?

Trimap is in fact is a three level pixel map which contains three types of pixels, definitely foreground, definitely background and unknown pixels. The unknown pixels are those which lie in the region separating the foreground and background objects. The trimap may be manually specified by the user or produced by binary segmentation approaches.

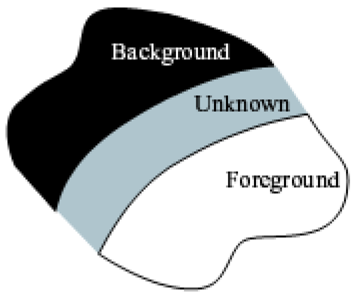


FIGURE 4 TRIMAP

It is worth mentioning that the recently proposed Spectral matting algorithm can automatically extract a matte from an input image without any user input. However, as the authors agreed, the automati

approach has a number of limitations including erroneous results for images with highly-textured backgrounds. Thus in practice, user specified trimaps are typically necessary to achieve high quality mattinresults.

3.2) Trimap Generation Techniques

Manually Generated Trimap

In this technique the unknown region of the trimap is marked by outlining the foreground object manually using rotoscopic curve or any basic polarized brushing technique.

For every keyframe the user has to repeat the same manual procedure of selecting the unknown regions. And hence this technique is computationally ineffective and slow.

Automtically Generated Trimap

In this technique the procedure of generating unknown region in trimap is automated using background subtraction and object tracking algorithms such as optical flow.

3.3) Background Subtraction Methods

A) **Frame difference:**

$| \text{frame}_i – \text{frame}_{i-1} | > \text{Th}$

- The estimated background is just the previous frame
- It evidently works only in particular conditions of objects’ speed and frame rate
- Very sensitive to the threshold Th.

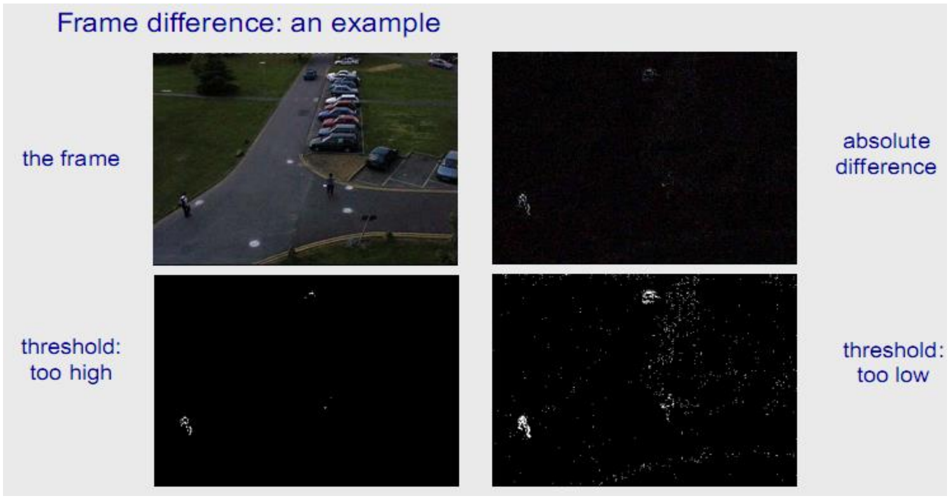


FIGURE 5 FRAME DIFFERENCE

B) Background as the average or the median of the previous n frames:

- rather fast, but very memory consuming: the memory requirement is n * size(frame)

C) Background as the running average:

- $B_i + 1 = \alpha * F_i + (1 - \alpha) * B_i$
- α, the learning rate, is typically 0.05.
- No more memory requirements

D) Mixture of Gaussians

- The model copes also with multimodal background distributions.
- The number of modes is arbitrarily pre-defined.
- All weights ω_i are updated (updated and/or normalised) at every new frame.
- At every new frame, some of the Gaussians "match" the current value (those at a distance < 2.5 σ_i): for them, μ_i, σ_i are updated by the running average.
- The mixture of Gaussians actually models both the foreground and the background: all distributions are ranked according to their ω_i / σ_i and the first ones chosen as "background"

3.4) Optical Flow Based Methods

Optical flow is an approximation of the local image motion based upon local derivatives in a given sequence of images. That is, in 2D it specifies how much each image pixel moves between adjacent images while in 3D in specifies how much each volume voxel moves between adjacent volumes.

The computation of differential optical flow is, essentially, a two-step procedure:

1. measure the spatio-temporal intensity derivatives (which is equivalent to measuring the velocities normal to the local intensity structures) and

2. integrate normal velocities into full velocities, for example, either locally via a least squares calculation or globally via a regularization.

The approach assumes localrigidity. This assumption assures that optical flow actually captures real motions in a scene rather than expansions,contractions, deformations and/or shears of various scene objects.

3.5) Using Optical Flow to calculate Trimap and Alpha matte

In this approach, Instead of matching pixel colors in the pair of frames, we measure how the optical-flow-warped alpha-channel fits the next frame. By minimizing this measure with respect to optical flow fully preserves the structure of the foreground object and prevents temporal incoherence artifacts.

The Algorithm

We start from a known alpha-channel for the first frame. It can be produced by a user using any existing image matting algorithm.Then we try and deform it with a smooth optical flow, until we get the best match for the next frame. We process consequent frames in the same way. The algorithm is due to M. Sindeyev et al. [3].

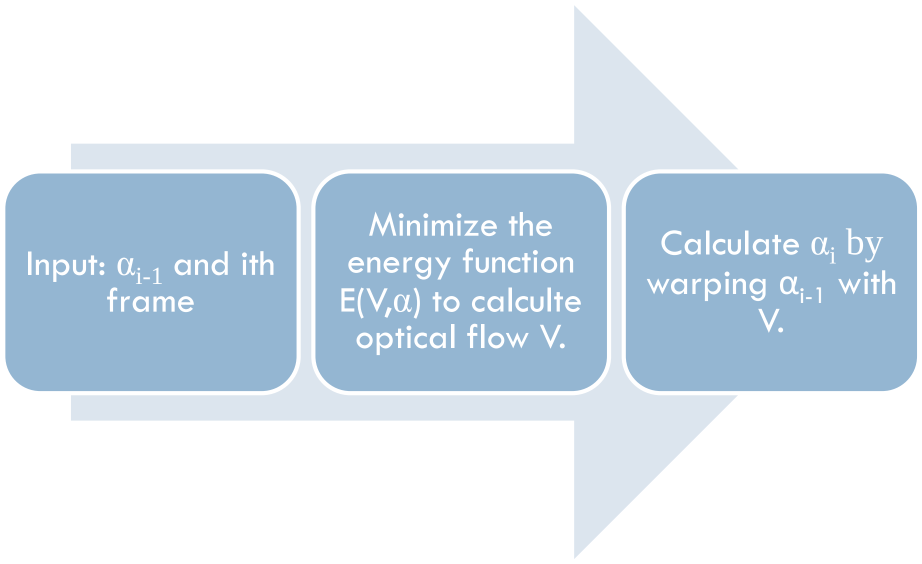

Input: α_{i-1} and ith frame
Minimize the energy function E(V,α) to calculte optical flow V.
Calculate α_i by warping α_{i-1} with V.

FIGURE 6 AUTOMATIC TRIMAP CALCULATION PROCEDURE

- When processing the current frame, we use the following energy function:

$$E(V,\alpha) = E_d(V,\alpha) + \mu E_s(V)$$

  where V is the optical flow, αis the alpha-channel of the previous frame (as a column-vector) andμ is a smoothness parameter.α is fixed.

- We obtain only V by solving:

$$V = \arg v \min E(V,\alpha)$$

- The data term is defined as follows:

$$E_{d1}(V,\alpha) = V(\alpha)^T L V(\alpha).$$

  And

$$E_{d2}(V,\alpha) = \sum_i V(\alpha)_i \left\| C_i - V(C_{prev})_i \right\|^2,$$

  Such that

$$E_d(V,\alpha) = E_{d1}(V,\alpha) + \mu E_{d2}(V,\alpha).$$

  The data term uses the Laplacian from the Closed-form Matting algorithm which is defined as

$$\sum_{k|(i,j)\in w_k} \left(\delta_{ij} - \frac{1}{|w_k|}\left(1 + (I_i - \mu_k)(\Sigma_k + \frac{\varepsilon}{|w_k|} I_3)^{-1}(I_j - \mu_k)\right)\right)$$

  whereΣk is a 3×3 covariance matrix, µk is a 3×1 mean vector of the colors in a window wk,and I3 is the 3×3 identity matrix.

- To regularize the optical flow a simple first-ordersmoothness term is used:

$$E_s(V) = \sum_i \sum_{j\in w(i)} \left\| V_i - V_j \right\|^2$$

  where w(i) is a 3x3 pixel neighborhood of the i-th pixel.

- The expression for energy is minimized to calculate Vx and Vy labeling for each pixel.

- Then an alpha-channel for the current frame is constructed by warping it with the found optical flow V according to he following equation

$$V(I)\big|_{(x,y)} = I(x + V_x(x,y), y + V_y(x,y)),$$

  Where V(I) represents a warping operation in all the above equations.

The result of tracking is refined by applying Bayesian Mat-ting with smoothness to prevent the accumulation of tracking error and interpolation artifacts. Unknown region to be processed is constructed from all pixels where alpha is not equal 0 or 1. The mean value for the alpha at each pixel is taken from the generated alpha map.

The alpha mattes so obtained can either be used as approximations of trimaps or alpha mattes in themselves for the subsequent frames in the video.

MATTE CALCULATION

MATTING TECHNIQUES

There are various image matting techniques which use different criteria for performing the matting which range from evaluating the color range of the pixels in the image to spectral matting techniques. We discuss some popular matting techniques that are used to arrive at the alpha matte for an image or a video frame. We discuss two categories of matting techniques- parametric sampling based non sampling based.

4.1) Sampling Techniques

Ruzan and Tomasi's method: This is one of the most basic techniques of Image matting. This method is a color sampling method. In this method the alpha values are measured along the connecting components of each objects color distribution. This approach assumes that the unknown region is a narrow band around the foreground boundary and the skeleton of the unknown region can be represented as a chain of pixels.

The model construction and alpha estimation procedure is summarized as follows:

(1) Divide the chain of pixels (the skeleton of the unknown region) into intervals by selected *anchor points*;

(2) Centered on each anchor point, define a local spatial window which covers a local unknown region, and a local foreground and background region.

(3) Foreground and background pixels in the local window are used to estimate a foreground and background isotropic (unoriented) Gaussian distribution, or *point mass*, respectively,

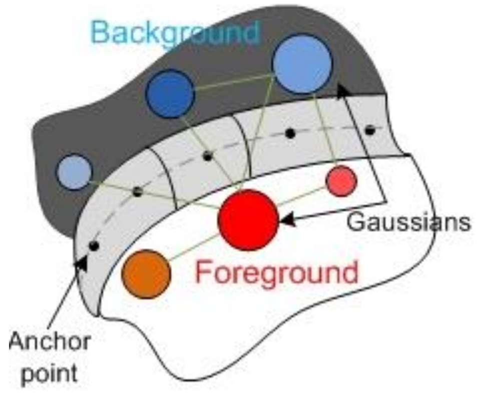


FIGURE 7 (A) MANIFOLD ESTIMATION

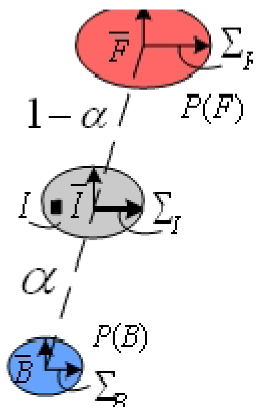


FIGURE 7 (B) MODEL INTERPOLATION (ALPHA ESTIMATION)

(4) Build the manifold by collecting foreground Gaussians with background Gaussians, while rejecting some connections according to certain “intersection” and “angle” criteria, as shown as the lines between point masses.

(5) The observed color of an unknown pixel is modeled as coming from an intermediate distribution between the foreground and background distributions. An intermediate distributions also defined to be a sum of Gaussians, where each Gaussian has linearly interpolated mean and covariance between a foreground and background Gaussian pair, according to an estimated alpha value. The optimal alpha is the one that yields an intermediate distribution for which the observed color has maximum probability.

(6) For an unknown pixel, after its alpha value is estimated, its foreground color is estimated by interpolating the means of foreground and background Gaussian pairs.

Disadvantages:

(i) Selection of anchor points is ad hoc

(ii) Alpha values are computed independently of the pixel values which may result in inconsistencies

Bayesian Matting: Similar to Ruzon and Tomasi's algorithm, this approach also models foreground and background colors as mixtures of Gaussians, but with a number of improvements. Bayesian matting uses a continuously sliding window for neighborhood definition, which marches inward from the fore-ground and background regions. In addition to the use of foreground and background samples to build color distributions, it also uses nearby computed Fs, Bs and αs, so that every pixel in the neighborhood will contribute to the foreground and background Gaussians. The matting problem is formulated in a well-defined Bayesian frame-work and the matte is solved using the maximum a posteriori (MAP) technique.

The model construction and alpha estimation procedure is summarized as follows:

(1) InMAP estimation, we try to find the most likely estimatesfor F, B, and alpha, given the observation C. We express this as a maximization over a probability distribution P and then use Bayes's rule to express the result as the maximization over a sum of log likelihoods:

$$\begin{aligned}&\arg\max_{F,B,\alpha} P(F,B,\alpha \mid C)\\ &= \arg\max_{F,B,\alpha} P(C \mid F,B,\alpha)\, P(F)\, P(B)\, P(\alpha) / P(C)\\ &= \arg\max_{F,B,\alpha} L(C \mid F,B,\alpha) + L(F) + L(B) + L(\alpha)\,,\end{aligned}$$

we drop the P(C) term because it is a constant with respect to the optimization parameters.

(2) We can model the first term by measuring the difference between the observed color and the color that would be predicted by the estimated F, B, and alpha:

$$L(C \mid F,B,\alpha) \;=\; -\|C - \alpha F - (1-\alpha)B\|^2 / \sigma_C^2$$

This log-likelihood models error in the measurement of C and corresponds to a Gaussian probability distribution centered at $C = \alpha F + (1\text{-}\alpha)B$ with standard deviation sigma.

(3) The sptial coherence of the image is used to estimate the foreground term L(F). That is, the color probability distribution using the known and previously estimatedforeground colors within each pixel's neighborhood N is built.To more robustly model the foreground color distribution, we weight the contribution of each nearby pixel i in N according to two separate factors. First, we weight the pixel's contribution by α^2_i , which gives colors of more opaque pixels higher confidence. Second, we use a spatial Gaussian falloff gi with sigma= 8 to stress the contribution of nearby pixels over those that are further away. The combined weight is then wi = $\alpha^2_i g_i$.

(4) Given a set of foreground colors and their corresponding weights, we first partition colors into several clusters using the method of Orchard and Bouman [7]. For each cluster,we calculate the weighted mean color F and the weighted covariance matrix $\sum_F$:

$$\overline{F} = \frac{1}{W} \sum_{i \in N} w_i F_i$$

$$\Sigma_F = \frac{1}{W} \sum_{i \in N} w_i (F_i - \overline{F})(F_i - \overline{F})^T$$

Where $= \sum_{i \epsilon N} \omega i$. The log likelihoods for the foreground L(F) can then be modeled as being derived from an oriented elliptical Gaussian distribution, using the weighted covariance matrix as follows:

$$L(F) = -(F - \overline{F})^T \Sigma_F^{-1} (F - \overline{F}) / 2 .$$

(5) The definition of the log likelihood for the background L(B) depends on which matting problem we are solving. For natural image matting, we use an analogous term to that of the foreground, setting wi to $(1-\alpha_i^2)g_i$in the foreground equations.

(6) We assume that the log likelihood for the opacity L(α) is constant (and thus omitted from the maximization in equation.

:

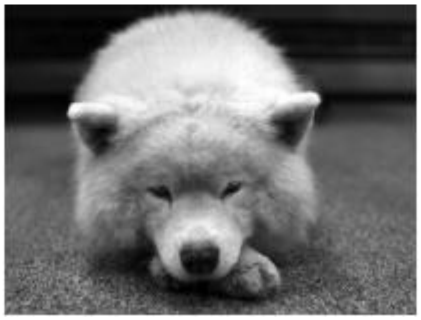


FIGURE 8 ORIGINAL MAGE

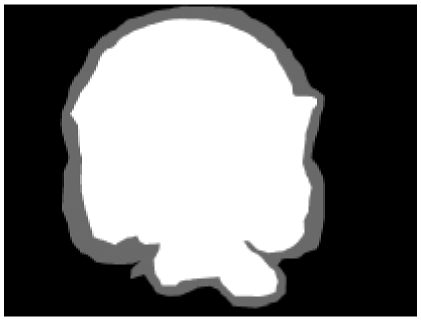


FIGURE 9 TRIMAP

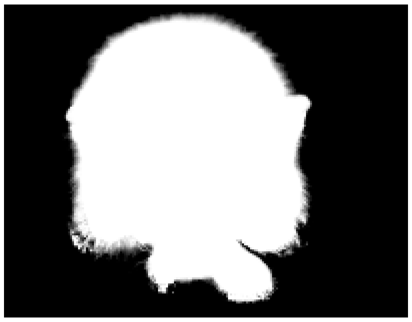


FIGURE 10 BAYESIAN MATTE

Bayesain Matting Results

4.2) Non-Sampling Techniques

Poisson Matting: Poisson Matting estimates the gradient of the matte from image, then reconstructs the matte by applying Poisson's Equation. This reduces the error caused by mis-classification of color samples in a complex scene.It is based on the assumption that intensity change in the foreground and background is smooth.

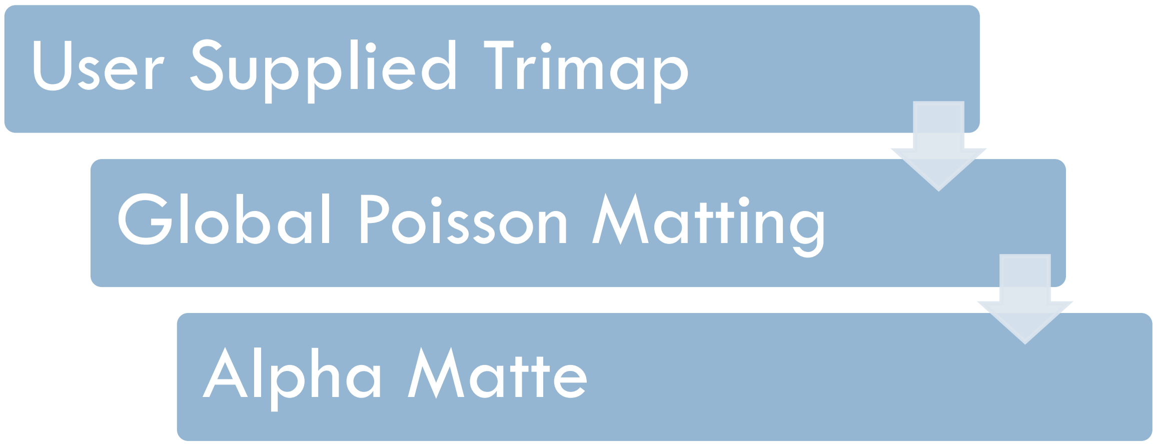


FIGURE 11 THE POISSON MATTING PROCESS

Poisson matting consists of two steps:

(1) First, an approximate gradient field of matte is computed from the input image.In order to get an approximate gradient field of matte, we take the partial derivatives on both sides of the matting equation:

$$\nabla I = (F - B)\nabla\alpha + \alpha\nabla F + (1-\alpha)\nabla B$$

This is the differential form of the matting equation, for R, G and B channels individually. In situations in which foreground F and background B are smooth, i.e., $\alpha\nabla F + (1-\alpha)\nabla B$ is relatively small with respect to $(F - B)\nabla\alpha$, we can get an approximate matte gradient field as follows:

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

(2) As shown in Figure , ΩF , ΩB and Ω are defined as "definitely foreground", " definitely background " and "unknown" regions respectively. For each pixel p =(x, y) in the image, Ip is its intensity,Fp and Bp are the foreground and background intensity respectively.LetNp be the set of its 4 neighbors. $\partial\Omega = \{p \in \Omega F \cup \Omega B \mid Np \cap \Omega =/ 0\}$ is the exterior boundary of Ω.

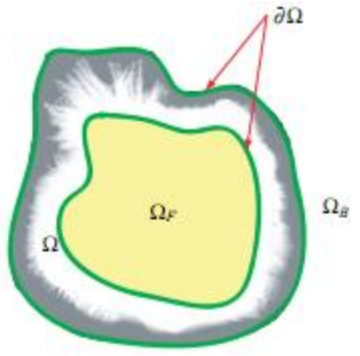


FIGURE 12POISSON MATTING REGIONS

(3) To recover the matte in the unknown region Ω given an approximate (F −B) and image gradient ∇I, we minimize the followingvariational problem

$$\alpha^* = \arg\min_{\alpha} \int\int_{p\in\Omega} ||\nabla\alpha_p - \frac{1}{F_p - B_p}\nabla I_p||^2 dp$$

With the following drichlet boundary conditions $\alpha_{|\partial\Omega} = \alpha`|_{\partial\Omega}$., we define:

$$\hat{\alpha}_p|_{\partial\Omega} = \begin{cases} 1 & p \in \Omega_F \\ 0 & p \in \Omega_B \end{cases}$$

And the associated Laplacian equation is :

$$\Delta\alpha = div(\frac{\nabla I}{F - B})$$

(4) We solve these equations using an iterative optimization process:

- Initially, for each pixel p in Ω, Fp and Bp are approximated by corresponding the nearest foreground pixel in ΩF and background pixel in ΩB. Then, the constructed (F −B) image is smoothed by a Gaussian filter to suppress significant changes due to noise and inaccurate estimation of F and B.

- α is reconstructed by solving Poisson equations using the current (F −B) and ∇I.

- Let Ω+F = {p ∈ Ω|αp > 0.95, Ip ≈ Fp}. The condition αp> 0.95 and Ip ≈ Fp guarantee that the pixels in Ω+F are mostly foreground. Similarly, let Ω+B = {p ∈ Ω|αp < 0.05, Ip ≈Bp}. Here, Fp, Bp and Ip represent the color vectors at pixel p.We update Fp and Bp according to the color of the nearest pixelsin ΩF ∪Ω+F and in ΩB ∪Ω+B , respectively. A Gaussian filter is alsoapplied to smooth (F −B).

- We iterate the above steps 2 and 3 until change in the matting results is sufficiently small or both Ω+ and Ω+ are empty in step 3.In each iteration, the selection of Ω+F and Ω+B has little error, which guarantees that more accurate colors in these two regions are further propagated into less accurate neighboring pixels.

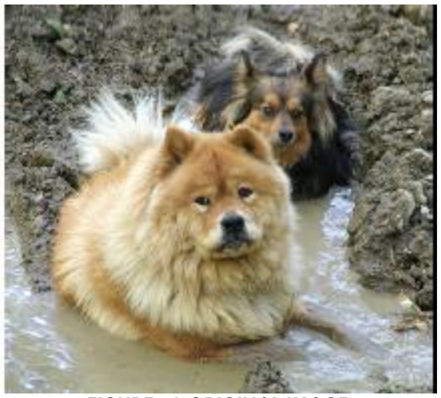


FIGURE 13 ORIGINAL IMAGE

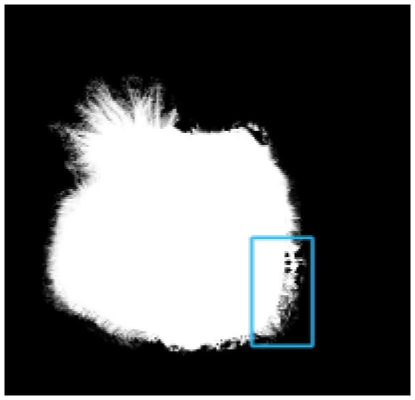


FIGURE 14 BAYESIAN MATTING

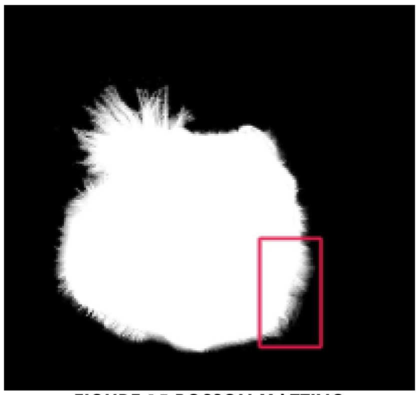


FIGURE 15 POSSON MATTING

Comparing the results of Bayesian and Poisson Matting

Spectral Matting: It is an approach to naturalimage matting that automatically computes a set of fundamental fuzzy matting components from the smallest eigenvectors of a suitably defined Laplacian matrix. Thus, this approach extends spectral segmentation techniques, whose goal is to extract hard segments, to the extraction of soft matting components. These components may then be used as building blocks to easily construct semantically meaningful foreground mattes, either in an unsupervised fashion or based on a small amount of user input.

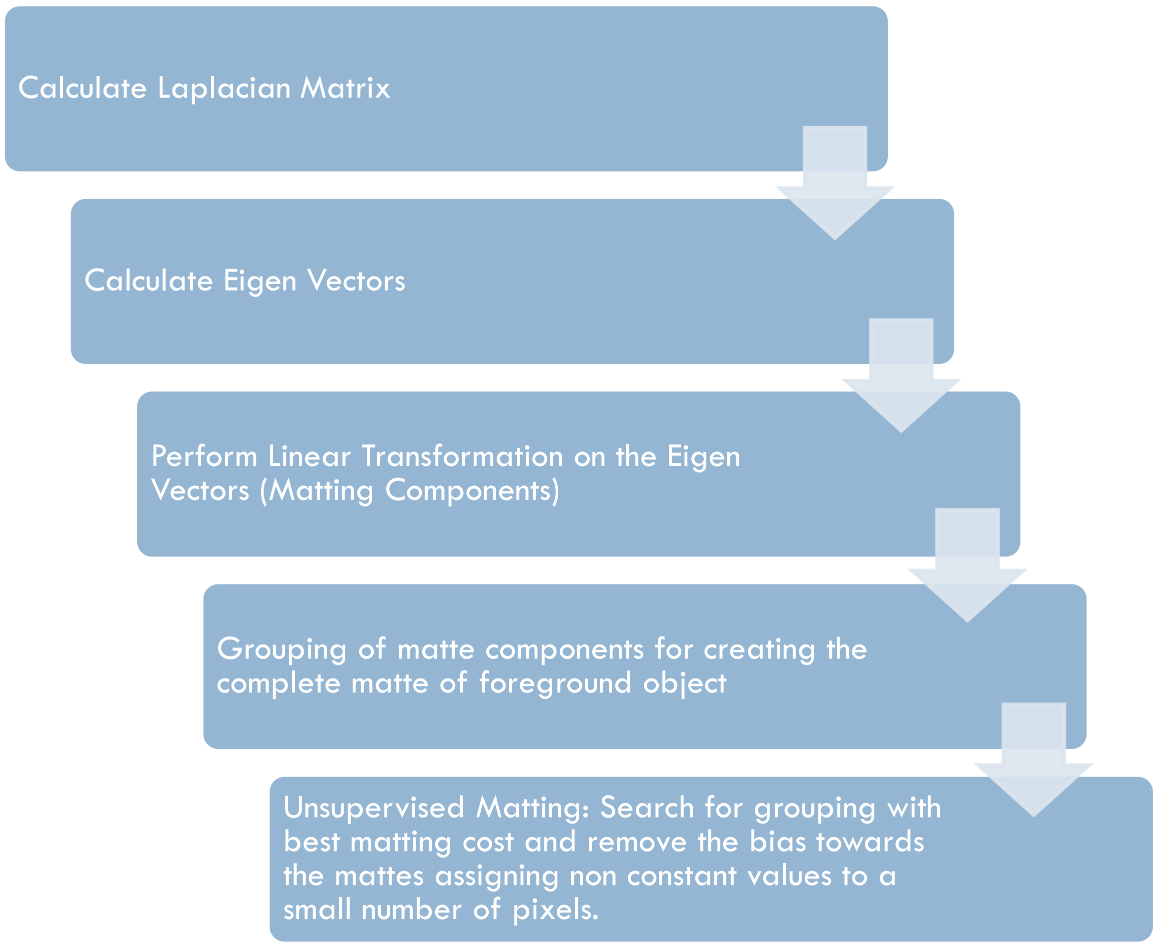

Calculate Laplacian Matrix
Calculate Eigen Vectors
Perform Linear Transformation on the Eigen Vectors (Matting Components)
Grouping of matte components for creating the complete matte of foreground object
Unsupervised Matting: Search for grouping with best matting cost and remove the bias towards the mattes assigning non constant values to a small number of pixels.

FIGURE 16 THE SPECTRAL MATTING PROCESS

Goals of Spectral Matting:

- Automatically extract matting components from an image
- Derive analogy between hard spectral segmentation and matting, and use similar tools.
- Use matting components to automate matte extraction process and suggest new modes of user interaction

(1) Compositing Equation:

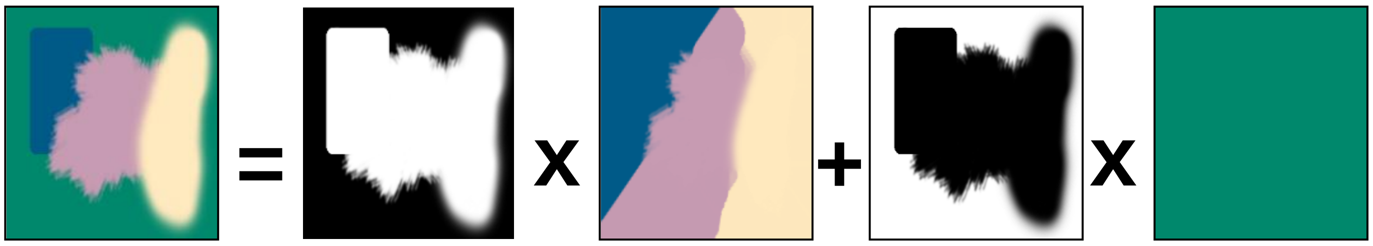
The composite equation is generalized assuming that each pixel is a convex combination of K image layers (like L^1,L^2… etc.)

i.e.

$$I_i = \alpha_i^1 L_i^1 + \alpha_i^2 L_i^2 + \ldots + \alpha_i^K L_i^K$$

The following diagrams demonstrate the difference between the 2-layer compositing equation and the generalized compositing equation:

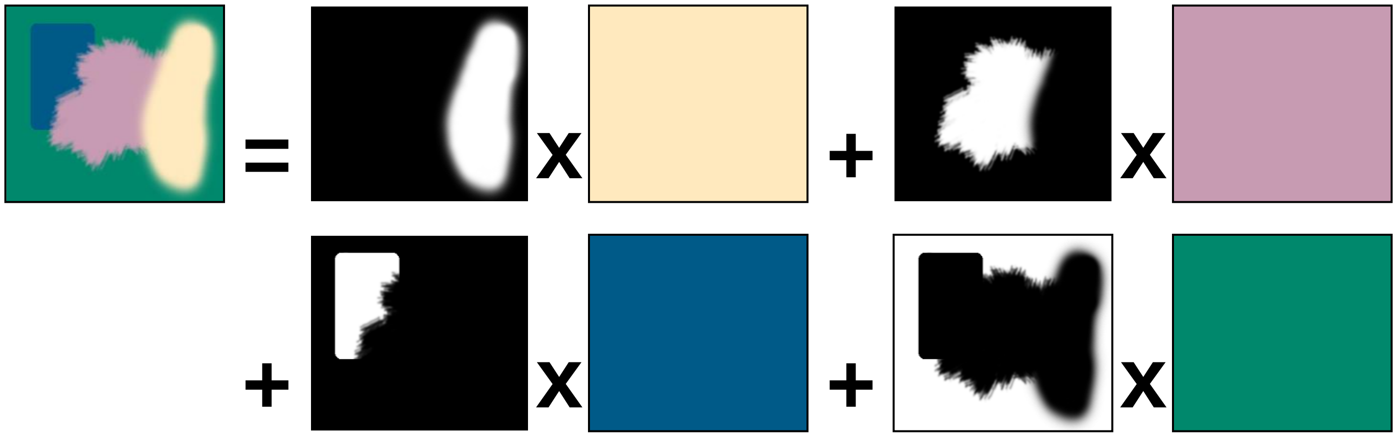
$$I_i = \alpha_i F_i + (1 - \alpha_i) B_i$$



2 LAYER COMPOSITING

The K-Layer compositing works as follows:

$$I_i = \alpha_i^1 L_i^1 + \alpha_i^2 L_i^2 + \ldots + \alpha_i^K L_i^K$$



K LAYER COMPOSITING

(2) Spectral Segmentation:

It consists of analyzing smallest eigenvectors of a graph Laplacian*L*

$$L = D - A$$

Where

D is the Degree Matrix

$$D(i,i) = \sum_j A(i,j)$$

A is the affinity matric (adjacent neighbors)

$$A(i,j) = e^{-\|C_i - C_j\|^2 / \sigma^2}$$

(3) Matting Laplacian:

The further analysis is done by processing the *matting Laplacian* given by:

$$J(\alpha) = \alpha^T L \alpha$$

Where L is a semidefinite sparse matrix given by:

$$L(i, j) \propto \sum_{k|(i,j)\in w_k} -\left(1+(C_i-\mu_k)^T(\Sigma_k+\varepsilon I_3)^{-1}(C_j-\mu_k)\right)$$

(4) Calculating the matting components

In this step a linear transformation (Newton's method) is applied to the eigen vectors to determine the matting components which yields a set of binary vectors.The transformation is applied iteratively on the constrained equation of alpha and this helps in measuring the sparsity of matting component. Point to note in this is that alpha Is not constrained between 0 and 1 but due to sparsity penalty it logically lies in within this range.
The result of the iterative operation depends upon the initial approximation taken for Newton's Method. One of the ways used to get a good initialization approximation is to apply k-means algorithm on the set of smallest eigenvectors of the matting Laplacian and project the indicator vectors of the resulting clusters on to the span of eigen vectors.

(5) Grouping Components
The problem to be solved here is to determine which matte component belong to the foreground as they need to be combined to produce the final matte.
For example, if α^1, α^2, and α^3 belong to the foreground matte then the complete matte is represented as:

$$\alpha^1+\alpha^2+\alpha^3$$

The grouping is done by calculating the cost and correlation between the matting components via L and store them.

Limitation of Spectral Matting :
The number of components need to be set , in case the number of components are too few, the result may not contain the desired matte, and if too many components are specified then the computation becomes complex and it affects the user interaction.

COMPOSITING

5.1 Compositing to get final image

Compositing is the process of digitally assembling multiple images to make a final image, typically for print, motion pictures or screen display. In our project we composite the extracted foreground images with a blank background to create a new image. The extracted foreground image can also be composited with any other background image to give the appearance of a changed background.

The basic operation as described earlier is 'alpha blending' or 'alpha matting', where an opacity value, 'α' is used to control the proportions of two input pixel values that end up a single output pixel.

Consider three pixels;

- a foreground pixel, f
- a background pixel, b
- a composited pixel, c

and

α, the opacity value of the foreground pixel. (α=1 for opaque foreground, α=0 for a completely transparent foreground). A monochrome raster image where the pixel values are to be interpreted as alpha values is known as the matte.

Then, considering all three colour channels, we have:

$c_r = \alpha f_r + (1 - \alpha)\, b_r$

$c_g = \alpha f_g + (1 - \alpha)\, b_g$

$c_b = \alpha f_b + (1 - \alpha)\, b_b$

These equations together give the final composited RGB images.

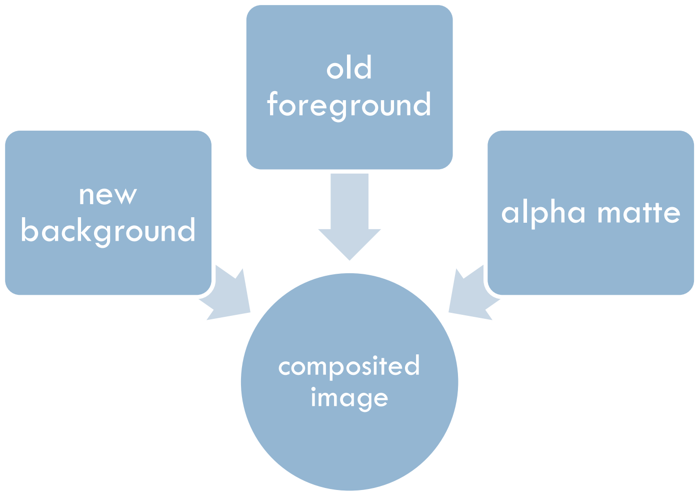
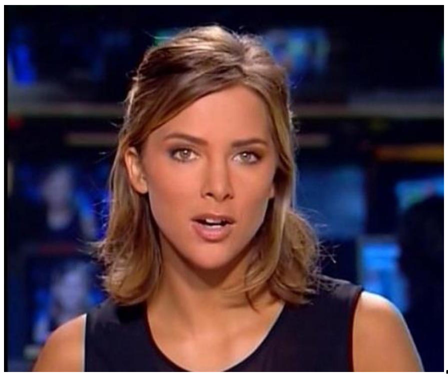


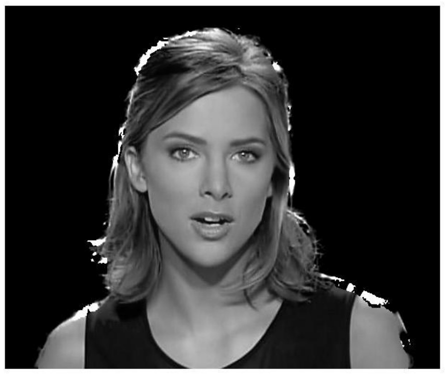
FIGURE 17 COMPOSITING

An Example



X

=



5.2 Compositing in Videos

The naive approach to obtaining composited video frames involves calculating alpha mattes for each of the frames and then performing the compositing operation in each frame. It is easy to see that the process is prohibitively expensiveand also tends to give discontinuous matting between frames.

An alternative approach is to calculate the matte for only certain keyframes and then interpolate between frames to arrive at the mattes for the entire video. This approach significantly reduces the computational expenses, but requires one to select a suitable interpolation mechanism to get the best results.

Also, a simple heuristic to reduce the computation time can be employed by rewriting the compositing equation. If this operation has to be done in real time video this is a good trick to boost performance.

$c_{out} = \alpha f_{in} + (1 - \alpha) b_{in}$

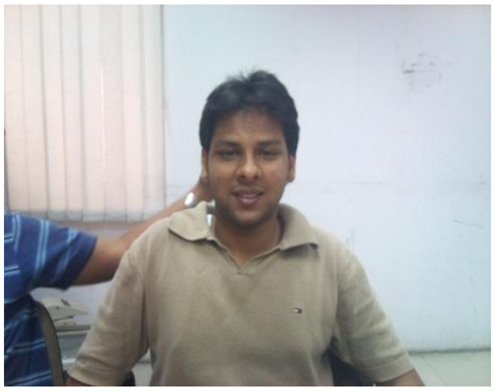
$c_{out} = \alpha f_{in} + b_{in} - \alpha b_{in}$

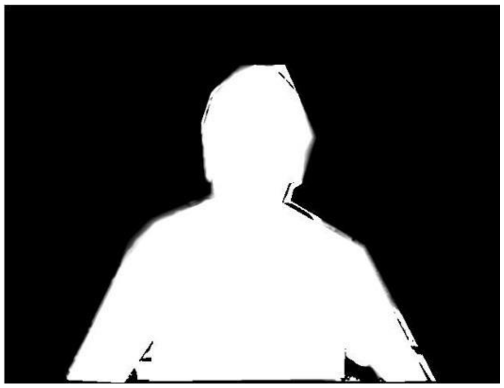
$c_{out} = b_{in} + \alpha (f_{in} - b_{in})$

By simply rewriting the mathematical expression one can save 50% of the multiplications required.

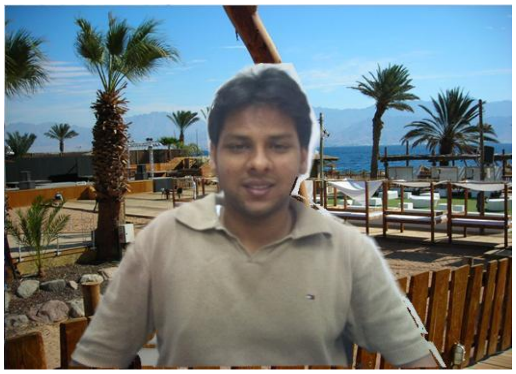
TEST CASES

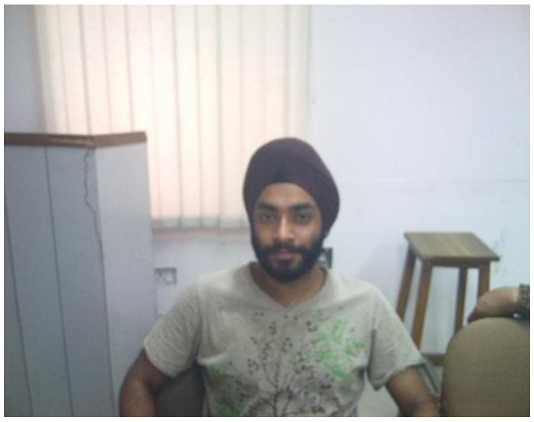
Poisson Matting

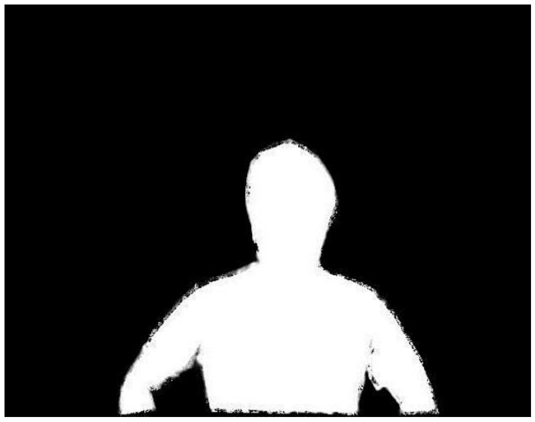

Original Image


Alpha Matte

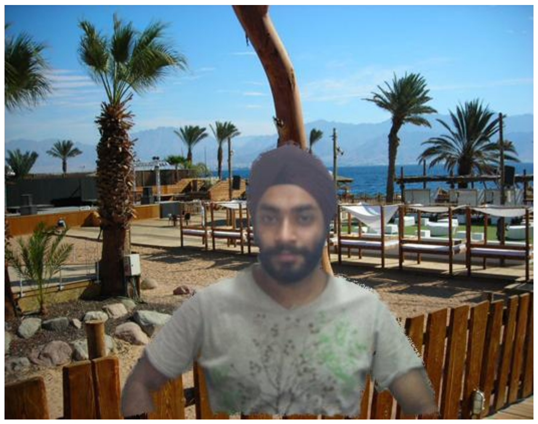
New Background


Composite Image

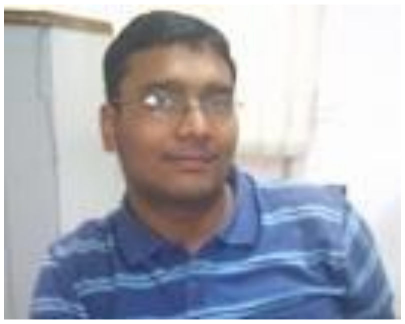

Original Image


Alpha Matte

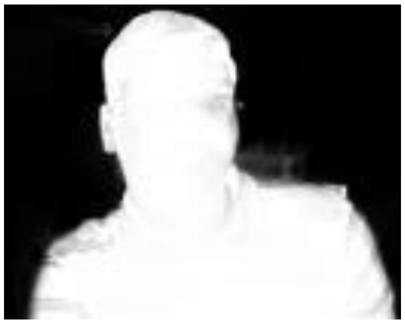
New Background


Composite Image

Spectral Matting

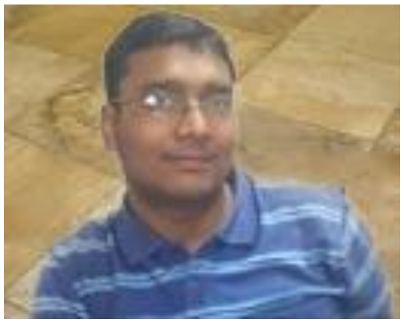


Original Image



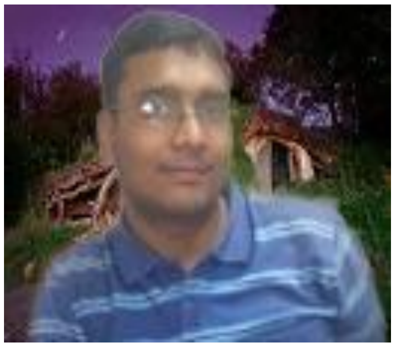
Alpha Matte

New Background 1



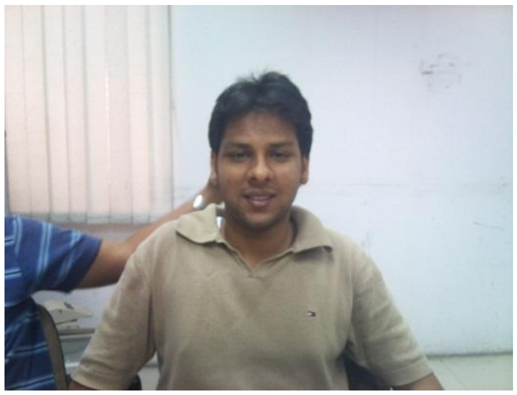
Composite Image 1

New Background 2

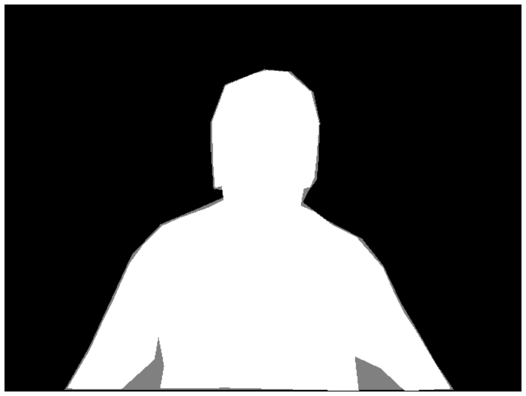


Composite Image 2

Bayesian Matting

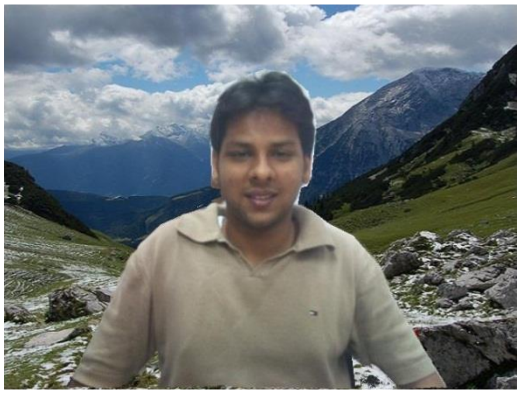


Original Image



Alpha Matte

New Background



Composite Image

MATLAB CODE

poisson_matte.m

```
function [ alpha alpha2 k I] = poisson_matte(w,tol1,tol2)
%POISSON_MATTE Summary of this function goes here
%  I - grayscale image in double format
%  w - SOR param , 1<w2 val = 1.5
%  tol - tolerance for stopping gauss-siedel iteration val = 0.05
%  tol2 - tolerance for comparison of I with Fp and Bpval = 0.01
%  alpha - the final settled matte
%  alpha2 -  the out of range matte

%%reading the test case images--------------
I = imread('testcases/tc2.bmp');
I = rgb2gray(I);
I = im2double(I);
F_mask = imread('./testcases/fg2.bmp');
B_mask = imread('./testcases/bg2.bmp');
O_mask = imread('./testcases/om2.bmp');
F_mask = im2bw(F_mask,0.5);
B_mask = im2bw(B_mask,0.5);
O_mask = im2bw(O_mask,0.5);
%%reading test case images-------------------

k=0;
while(1)
    k = k+1;
%%1.get the disjoint F_mask, B_mask,O_mask from trimap image
%%to be implemented

%%2.compute the F and B images for each R,G,B component
    [F B filt_image] = calculate_fb(F_mask,O_mask,B_mask,I);

%%3.compute D as div(gradient(I)/(filt_image)) -doubt in computation
    D = calculate_d(I,filt_image);

%%4.solve the poisson equations for all pixels in omega to get alpha
[ alpha alpha2] = poisson_solver(O_mask,F_mask,D,100,w,tol1);

%%5. Use alpha matrix to get extension of foreground and background
[ F_maskO_maskB_maskOf_sumOb_sum] =
update_fb(F_mask,O_mask,B_mask,alpha,I,F,B,tol2);

%%6.repeat process until F_mask and B_mask stop growing
if((Of_sum==0)&&(Ob_sum==0))
break;
end
end
```

calculate_fb.m

```
function [F B filt_image] = calculate_fb(F_mask,O_mask,B_mask,I)
%INITIALISE_D this function computes the F and B images
%F_mask - mask for foreground region
%O_mask - mask for omega region
%B_mask - mask for background region
%I - intensity image


[m n] = size(I);

%%initialisations


F=I;
O_test = zeros(m,n,'uint8');
O_test(:) = O_mask(:);
next_f = cell(m*n,1);
new_f  = cell(m*n,1);


%%pixels at boundary of Fg used to claculate initial values for next_f pixels
boundary = bwboundaries(F_mask);
k=1;
for r=1:1:length(boundary)
    b = boundary{r};
for s=1:1:length(b)
next_f{k} = [b(s,1) b(s,2)];
        k=k+1;
end
end


%%foreground dilation to calculate F matrix
l=0;
while((l~=1))                          %%because if l==1 new has not grown
    l=1;
    q=1;
while(~isempty(next_f{q}))
coord = next_f{q};
        q=q+1;
        i = coord(1);
        j = coord(2);
%for each pixel in foreground
%get the safe neighbours
if((i==1)||(i==m)||(j==1)||(j==n))
neighb = safe_neighbors(i,m,j,n);
else
neighb = [i (j-1); (i-1) j; i (j+1); (i+1) j];
end
for k=1:1:4
ni = neighb(k,1);
```

```
nj = neighb(k,2);
if(O_test(ni,nj)==1)
F(ni,nj) = F(i,j);
F_mask(ni,nj)=1;
new_f{l} = [ninj];
                    l = l+1;
O_test(ni,nj)= O_test(ni,nj)+1;
end
end
end
next_f = new_f;
new_f = cell(m*n,1);
end

%%initialisation

B=I;
O_test = zeros(m,n,'uint8');
O_test(:) = O_mask(:);
next_b = cell(m*n,1);
new_b  = cell(m*n,1);

boundary = bwboundaries(B_mask);
k=1;
for r=1:1:length(boundary)
    b = boundary{k};
for s=1:1:length(b)
next_b{k} = [b(s,1) b(s,2)];
        k=k+1;
end
end

l=0;
while((l~=1))                          %%because if l==1 new has not grown
    l=1;
    q=1;
while(~isempty(next_b{q}))
coord = next_b{q};
        q=q+1;
        i = coord(1);
        j = coord(2);
if((i==1)||(i==m)||(j==1)||(j==n))
neighb = safe_neighbors(i,m,j,n);
else
neighb = [i (j-1); (i-1) j; i (j+1); (i+1) j];
end

for k=1:1:4
ni = neighb(k,1);
nj = neighb(k,2);
if(O_test(ni,nj)==1)
B(ni,nj) = B(i,j);
B_mask(ni,nj)=1;
new_b{l} = [ninj];
                    l = l+1;
```

```
O_test(ni,nj)= O_test(ni,nj)+1;
end
end

end
next_b = new_b;
new_b = cell(m*n,1);
end

T = I;
for i=1:1:m
for j=1:1:n
if(O_mask(i,j)==1)
            d = F(i,j)-B(i,j);
if(d>0)
T(i,j) = d;
else
T(i,j) = 0;
end
end
end
end



%%___________gaussian smooth the image_______________________
gfilter = fspecial('gaussian',[3 3], 3);
filt_image = imfilter(T, gfilter, 'replicate');
```

calculate_d.m

```
function [D div filt_image] = calculate_d(I,filt_image)
%CALCULATE_D computes D as div(gradient(I)/(filt_image))
%I - grayscale image in double;
%filt_image filtered image in double


[gxgy] = gradient(I);
%div  = divergence(gx,gy);
%%remove zeros from filt_image before dividing
y = filt_image(filt_image>0);
min_val = min(y(:));
filt_image(filt_image==0)=min_val;
%D = div./filt_image;
gx = gx./filt_image;
gy = gy./filt_image;
div = divergence(gx,gy);
D = div;
end
```

poisson_solver.m

```
function [ alpha alpha2 k l s] = poisson_solver(O_mask,F_mask,D,maxiter,w,tol)
%POISSON_SOLVER This function solves the poisson equations to
%get the final alpha as according to step 2 of the agorithm

%O_mask - binary mask denoting the omega region
%F_mask - binary mask denoting the Foreground region
%D - previously computed values of div(grad(I)/(F-B)) for each pixel
%maxiter - max no of iterations for gauss siedel method
%w - overrelaxation parameter
%tol - acceptable tolerance
%alpha - alpha matte
%alpha2 - alpha matte with out of range vals
%k - no of iterattions on which approximation brraks
% l &s - values to bre compared to check convergence if k==maxiter




[m n] = size(D);
flag = zeros(m,n,'uint8');

a = zeros(m,n,2);
a(:,:,1) = F_mask; %all ones for F pixels
a(:,:,2) = F_mask;

for i=1:1:m
for j=1:1:n
if(O_mask(i,j)==1)
a(i,j,1)=0.5;
end
end
end

%create the list of pixels in omega region beforehand for speedup
O_list = cell(m*n,1);
l=0;
for i=2:1:m-1
for j=2:1:n-1
if(O_mask(i,j)==1)
                l = l+1;
O_list{l} = [i j];
end
end
end

for k=1:1:maxiter
for q=1:1:l
coord = O_list{q};
            i = coord(1);
            j=  coord(2);
```

```
        a(i,j,2) = a(i,j,1)+ 0.25*w*(a(i+1,j,1)+a(i,j+1,1)+a(i,j-1,2)+a(i-1,j,2)-
4*a(i,j,1)-D(i,j));

if(abs(a(i,j,2)-a(i,j,1))<tol)
flag(i,j)=1;
end
end
if(sum(flag(:))==l)
break;
end
a(:,:,1) = a(:,:,2);
end
s = sum(flag(:));

alpha = a(:,:,2);
alpha2 = a(:,:,2);
alpha_min = min(alpha(:));
range = max(alpha(:))-alpha_min;
for i=1:1:m
for j=1:1:n
if(O_mask(i,j)==1)
alpha(i,j) = (alpha(i,j)-alpha_min)/range;
end
end
end
```

safe_neighbors.m

```
function [ neighb ] = safe_neighbors(i,m,j,n)
%SAFE_NEIGHBOURS bounds check to avoid index out of scope
%error
neighb = [i (j-1); (i-1) j; i (j+1); (i+1) j];
if(i==m)
neighb(4,:)= [i j];
end
if(i==1)
neighb(2,:)=[i j];
end
if(j==n)
neighb(3,:)=[i j];
end
if(j==1)
neighb(1,:)=[i j];
end
```

update_fb.m

```
function [ Fm_newOm_newBm_newOf_sumOb_sum ] = update_fb( F_mask,O_mask,B_mask,
alpha, I, F, B, tol)
%UPDATE_FB calcultaes the new F & B mask biy extending the old ones
%according to the criterias specified below
%params - the same as before
%tol - the diff bw F or B value and  value.




[m n] = size(I);
Of_plus = zeros(m,n,'uint8');
Ob_plus = zeros(m,n,'uint8');
for i=1:1:m
for j=1:1:n

if(O_mask(i,j)==1)
if((alpha(i,j)>0.95)&& abs((I(i,j)-F(i,j)))<tol)
%pixel belonging to omega f plus
Of_plus(i,j) = 1;
%remove from omega
O_mask(i,j) = 0;
end
if((alpha(i,j)<0.05)&& abs((I(i,j)-B(i,j)))<tol)
%pixel belonging to omega b plus
Ob_plus(i,j) = 1;
%remove from omega
O_mask(i,j) = 0;
end
end
end
end
end

Fm_new = F_mask | Of_plus;
Bm_new = B_mask | Ob_plus;
temp = Fm_new|Bm_new;
Om_new = ~temp;
Of_sum = sum(Of_plus(:));
Ob_sum = sum(Ob_plus(:));
end
```

FUTURE DIRECTIONS AND CONCLUSION

FUTURE DIRECTIONS AND CONCLUSIONS

The ultimate goal of matting research is to develop intelligent, userfriendly, computationally efficient tools, which can be used to extract high quality mattes wherever the foreground and background are separable to human eyes, both on still images and video sequences.

8.1)Limitations of Current Approaches

Although matting techniques have been largely improved in recent years, there is still a long way to go to achieve this goal.

1) <u>Accuracy</u>

Current matting approaches can achieve good results when the foreground and background are smooth and well-separable in the color space. However, if the foreground and/or background contain(s) highly-textured regions with complex color patterns, existing matting approaches tend to generate noisy results with noticeable artifacts. One could imagine that a more complex background will make the case even worse. Strong color discontinuities within texture patterns may be even stronger than real foreground edges, thus will confuse most of matting algorithms. How to improve the accuracy of matting algorithms against such difficult examples is a open question.

2) <u>Efficiency</u>

Although recently proposed matting algo- rithms tend to generate more accurate results than early approaches, they are generally more expensive to compute. The Soft Scissors sys-tem achieves near-realtime performance on 1000 × 1000 images, but are not able to give instant feedback on larger ones. The mem- ory consumption of matting algorithms also need to reduced in order to deal with gigapixel images. The GPU implementation for the basic Random Walk matting is inspiring, but how to take advantage of GPU computation for other algorithms is still unknown.

8.2 Future Directions

There are a number of directions that one can explore in order to improve current matting algorithms, or build new matting sys-tems that outperform existing ones.

Automatic Evaluation towards Self-Adjusting Mattes

There are some common properties of good mattes extracted from natural images. For instance, the transition from foreground to back- ground are generally smooth, and it is almost impossible to have two neighboring pixels whose alpha values are 1 and 0, or have a region where every pixel has the same alpha value of 0.5. These errors can often be spotted in erroneous mattes. Developing automatic means that can detect erroneous matting re- sults, both globally and locally, can be beneficial to existing matting systems. Most of existing matting algorithms have a number of tunable parameters, which are either fixed internally, or provided to the user to tweak. If the resulting matte can be automatically evaluated, matting algorithms thus can adjust their parameters to find the best parameter combination, which can generate the best possible result.

Automatic evaluation can also help build hybrid matting system by combining a number of techniques together. Different matting algorithms have quite different characteris tics, thus may work well in different situations. By evaluating results generated by different algorithms from the same input, a better one can be produced by combining all of them together using thebest parts of each of them.

8.3 Learning/Example-based Approaches

Existing matting approaches only rely on the current input to gen- erate a result, thus could fail in similar situations over and over again. This gives us the opportunity to use example-based learning approaches to augment matting systems, and give them the ability to learn whatthe correct mattes should be in certain situations. Potentially, a matting system could be customized if the user keeps feeding it certain types of inputs. As more training examples are available one can hope the systems can improve. As a result, the required user efforts for gen- erating good results will be reduced over time.

The idea may sound straightforward, but how to implement such a system remains an open problem. Since foregrounds in different images usually have quite different colors and shapes, the training samples may contain extremely large variances. A compact set of features thus need be extracted which can capture the essential characteristics of the underlying mattes, while ignoring the absolute foreground and back- ground colors in each example.

The newly proposed high resolution matting system has made the first attempt along this direction. A high resolution ground-truth data set is constructed, and a new gradient preserving prior on alpha is developed based on the training data, which is used in the “alpha deblurring” process for improving the results generated by previous approaches. Such a data set is extremely valuable for exploring new example-based matting approaches in the future.

8.4More Practical Video Matting Systems

A number of image matting systems have already been successfully commercialized. Compared with image matting tools, current video matting systems are somewhat less practical. Existing systems either are computationally expensive, or have too many assumptions which may or may not hold in practice. Further- more, it is unclear what is the right user interface for video matting. Keyframe-based interfaces are natural and intuitive, but may not be efficient when large motions present. Volume-based interfaces are very efficient for marking foreground objects in multiple frames, however are less intuitive for normal users. A hybrid interface which combines them together may stand out in the future.

Once a matte has been extracted, what can one do with it? We have barely touched on this topic. The obvious application is compositing the matte and associated image onto a new background, but other applications are possible. We have shown one application for refor- matting images by making the foreground object bigger on the same background. One can also envision much more interesting compositing applications where multiple matted images and video are combined.

This raises other issues such as adjusting the lighting, color tempera- ture, and shadowing to create a seamless composite. We are sure there are other applications we have not thought of. Hopefully this survey will provide a good basis for those wanting to push the state-of-the-art in matting methods and for developing new unforseen applications.

REFERENCES

REFERENCES

1. Jian Sun, JiayaJia, Chie Keung Tang, Heung YeungShum:Poisson Matting
2. Jonathan Finger, Oliver Wang, Video Matting from depth Maps
3. P. Perez, M. Gangnet, and A. Blake, "Poisson Image Editing", SIGGRAPH, 2003.
4. Anat Levin, Alex Rav-Acha, DaniLischinski , Spectral Matting
5. Y. Chuang, B. Curless, D. Salesin, and R. Szeliski, "A Bayesian Approach to Digitial Matting", CVPR, 2001.
6. Image and Video Matting: A Survey Jue Wang and Michael F. Cohen.
7. Y. Chuang, A. Agarwala, B. Curless, D. Salesin, and R. Szeliski. Video matting of complex scenes, 2002.
8. M. Ruzon and C. Tomasi, "Alpha estimation in natural images," in Proceedings of IEEE CVPR, pp. 18–25, 2000.