VIDEO MATTING USING MOTION EXTENDED GRABCUT

David Corrigan[1], Simon Robinson[2] and Anil Kokaram[1]

[1] Sigmedia Group, Trinity College Dublin, Ireland. Email: {corrigad, akokaram}@tcd.ie
[2] The Foundry, London, UK. Email: sam@thefoundry.co.uk

Keywords: Foreground Segmentation, Compositing, Parametric Data Modelling, Global and Local Motion Estimation, Digital Cinema Post Production.

Abstract

GrabCut is perhaps the most powerful semi-automatic algorithm for matting presented to date. In its existing form, it is not suitable for video object segmentation. This paper considers major extensions that make it suitable for this purpose. A method for initialising matting without user intervention is presented, followed by a more robust data model using a Mean Shift algorithm to control model complexity. In addition, normalised motion information as well as colour is used to form joint colour and motion feature vectors. This improves the robustness of the mattes in the presence of colour camouflage and decreases the user intervention required for a successful result. Comparison between GrabCut and the proposed Motion Extended GrabCut (MxGrabCut), shows the improvement for video matting.

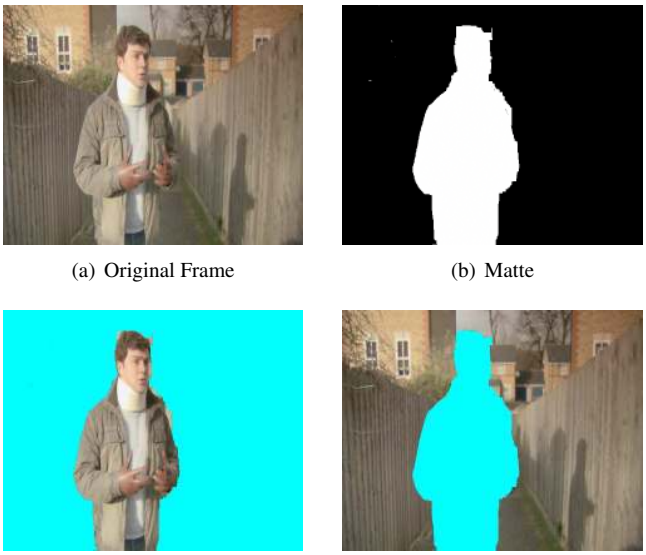
1 Introduction

The segmentation of images into foreground and background layers is a topic that has attracted much attention in the vision community. It is an important task in digital cinema post-production where the typical use is to "pull a matte" from a scene and to composite it onto a new background. A matte, $l(\mathbf{x})$, describes the membership of each pixel, $\mathbf{x}$, to the foreground and background layers and can be described by the equation

$$\mathbf{C}(\mathbf{x}) = l(\mathbf{x})\mathbf{C}_{fg}(\mathbf{x}) + (1 - l(\mathbf{x}))\mathbf{C}_{bg}(\mathbf{x}) \qquad (1)$$

where, $\mathbf{C}(\mathbf{x})$ is the colour value of the observed image at that site and where $\mathbf{C}_{bg}(\mathbf{x})$ and $\mathbf{C}_{fg}(\mathbf{x})$ are the colour values of the foreground and background layers. In general, the matte $l(\mathbf{x})$ can take any value between 0 and 1, although often values are restricted to 0 and 1 which implies that each pixel must belong wholly to one of the layers. Fig. 1 shows an example of a frame divided into foreground and background layers.

Foreground Segmentation has use in a number of applications, including global motion estimation algorithms [21, 9, 19] and image coding [28, 14] as well as compositing for motion-picture post-production[15, 5, 4, 26]. The precise definition of what constitutes the foreground and background layers varies according to the application, the most universal definition being that the foreground layer is composed of objects which are moving and that the background layer is composed of static objects (See Fig. 2).



(a) Original Frame (b) Matte

(c) Foreground Layer (d) Background Layer

Figure 1: This figure shows the separation of the image (a) into foreground and background layers, (c) and (d), using a matte generated by MxGrabCut (b).

1.1 Related Work

Generating mattes of sufficient quality for automatic compositing has proven a difficult task. High quality mattes are required for convincing results when a foreground layer is composited onto a different background. Sub-pixel accuracy is required for the mattes, as pixels at layer boundaries can contain information from both layers. The fall-back solution in the industry has been to use manual rotoscoping to cut out foreground objects from a scene or to solve the constrained segmentation problem of blue or green screen keying.

There has been some research into interactive rotoscoping algorithms as a lower cost alternative to manual rotoscoping. One approach is to make the user place points along or near the layer boundary, with the precise layer boundary being found using an edge tracking algorithm (*e.g.* [23, 20]). However, most research has focused on a region-based solution to the matting problem which estimates mattes directly using a limited amount of user interaction.

The most successful region-based algorithms have tended to

Figure 2: These images show four consecutive frames from a sequence where the camera is tracking the quad-bike. Although the quad-bike is static relative to the camera, it is considered to be the foreground as it is the only object that is moving relative to the ground.

approach the problem in a Bayesian fashion. Given the colour data $\mathbf{C}(\mathbf{x})$, the desired matte, $l(\mathbf{x})$, is the one that finds the *maximum-a-posteriori* value of the probability distribution $p(l(\mathbf{x})|\mathbf{C}(\mathbf{x}))$. Factorising the posterior in a Bayesian fashion yields

$$p(l(\mathbf{x})|\mathbf{C}(\mathbf{x})) \propto p_d(\mathbf{C}(\mathbf{x})|l(\mathbf{x})) \times p_s(l(\mathbf{x})|L) \qquad (2)$$

where $p_d(.)$ is the data likelihood and $p_s(.)$ is a prior on the value of l at a site $\mathbf{x}$ given the label values L in the neighbourhood of $\mathbf{x}$. The segmented image $l(\mathbf{x})$ is considered to be a *Markov Random Field*.

A Bayesian solution to the matting problem was presented in the Bayes Matting algorithm of Chuang *et al.* [5, 4]. Mattes are estimated to sub-pixel accuracy by allowing non-binary values for the matte, $l(\mathbf{x})$, in Eq. 1. In the Bayesian matting algorithm, the likelihood is estimated from Gaussian Mixture Models (GMMs) (*i.e.* a weighted sum of multi-variate gaussian distributions) of the foreground and background layers. The same prior value is used for all possible values $l(\mathbf{x})$, implying this term is effectively ignored. Bayes Matting requires the user to provide a trimap, which defines some regions of the image as either being definitely foreground or definitely background. A small band of pixels at the layer boundary is not assigned to either layer and the Bayes Matting algorithm estimates the value of $l(\mathbf{x})$ at these pixels.

Constructing the trimaps used in Bayes Matting requires a large amount of user interaction. Authors have attempted to reduce the amount of user interaction by solving the binary segmentation problem. In the binary problem, pixels belong wholly to the foreground or background layers. The GrabCut algorithm or Rother *et al.* [25] is one of the more well known approaches. GrabCut merely requires the user to draw a box around the foreground object. It again uses GMMs of the foreground and background layers to construct the likelihood. However, it also introduces a prior $p_s(l(\mathbf{x})|L)$ which enforces a spatial smoothness constraint on the matte. A further innovation of GrabCut is that it employs an iterative approach by recursively solving for $l(\mathbf{x})$ and using that solution to refine the GMMs of the foreground and background.

The common drawbacks of the segmentation algorithms discussed above are that they make no use of temporal information and they require user intervention. Recognising this, Kokaram *et al.* [15] proposed an automated matting algorithm that incorporates temporal sequence data in the form of global motion compensated Displaced Frame Differences (DFDs) and estimates the mattes using a Bayesian framework. Although temporal information is included, the there no explicit inclusion in the framework of the colour or texture information in the current frame. Another important algorithm was proposed by Criminisi *et al.* in [7] which adds motion likelihood and temporal prior terms to the Bayesian framework. However, the algorithm is limited in application to sequences with a static camera and requires a manually labelled ground truth to train some of the terms used in the Bayesian framework.

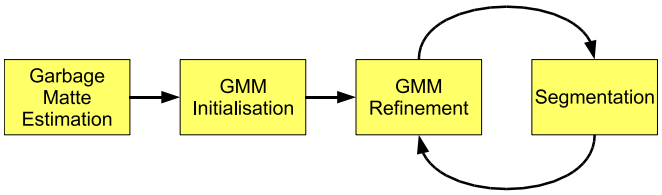


Figure 3: This flowchart outlines the operation of MxGrabCut.

1.2 Motion Extended GrabCut

The proposed algorithm extends on the GrabCut algorithm, resulting in an algorithm that produces high quality mattes without the need for any user intervention or a ground truth. First of all, user interaction is eliminated by estimating an initial garbage matte using the frame intensity difference between the frame to be segmented and its neighbours. This garbage matte allows a GMM to be trained for each the foreground and background layers of the image. The algorithm then adopts an iterative approach similar to GrabCut, with the GMMs being recursively refined at each iteration (See Fig. 3).

Two key improvements have been made to the iterative segmentation framework compared to the one used in GrabCut. Firstly, the GMM for each layer is extended so that they a model a feature vector containing both the RGB values and forward and backward motion values for each pixel. This allows any dependence between colour and motion to be modelled. In total there are 7 data values for each pixel,

3 RGB values and two each for motion with respect to the backward frame and motion with respect to the forward frame. The other key improvement is the use of Mean Shift [10, 6] to automatically choose the number of components in the GMM. Because of this, the GMM should better match the complexity of the data distribution than if the number of components was fixed by the user.

The following sections introduce the MxGrabCut algorithm, and in particular describe the data modelling technique for the foreground and background layers. This is followed by a summary of the results obtained from the algorithm and a discussion which also describes some directions for future research.

2 Data Requirements

The proposed iterative segmentation algorithm requires both colour and motion data. The colour data is simply provided by the image data in the RGB colour space (*i.e.* $\mathbf{C}(\mathbf{x})$). The motion data can be usually derived from the motion vector fields produced by a motion estimator such as [17]. In this case, the motion values are given by the horizontal and vertical vector components of the motion with respect to to the previous frame and also the next frame.

An alternative to performing motion estimation, proposed in [7], is to use spatial and temporal derivatives of the image sequence intensity function. This avoids the need for computationally costly motion estimation, and is reasonable since these derivatives are central to gradient-based motion estimation algorithms [17, 13]. The 4 motion values for each pixel are now given by the forward and backward intensity difference and the horizontal and vertical spatial difference where these terms are defined as

$$\begin{aligned}\Delta_{n,n-1}(\mathbf{x}) &= I_n(\mathbf{x}) - I_{n-1}(\mathbf{x}) \\ \Delta_{n,n+1}(\mathbf{x}) &= I_n(\mathbf{x}) - I_{n+1}(\mathbf{x}) \\ I_x(\mathbf{x}) &= 0.5\big(I_n(x+1,y) - I_n(x-1,y)\big) \\ I_y(\mathbf{x}) &= 0.5\big(I_n(x,y+1) - I_n(x,y-1)\big)\end{aligned} \tag{3}$$

respectively. In Eq. 3, $I_n(\mathbf{x})$ is the intensity of a pixel $\mathbf{x}$ in frame n and $\mathbf{x}$ represents a point (x, y) in the Cartesian co-ordinate space. MxGrabCut can be adapted to work with either form of motion information, although the temporal and spatial derivatives are used in the examples shown later.

3 Normalisation of Motion and Colour

An important consideration in constructing the joint colour/motion feature vector is to normalise the motion values with respect to the colour values. The normalisation should ensure that a given difference between values in a motion component has the same significance as the same difference between values in a colour component. The adopted approach is to force the ranges of the colour and motion components to be the same. This is more straightforward if the temporal and spatial derivatives are used for motion. Then, the theoretical upper and lower bounds of each term can be deduced and the range of the data is adjusted so that it is the same as the colour values. However, if motion vectors are used, the theoretical range can not be deduced. Consequently, the range is estimated by finding the maximum and minimum values in the motion vector fields. In both cases, the new motion value m is given by

$$m = 255 \times \frac{m - m_{min}}{m_{max} - m_{min}} \tag{4}$$

where the range of colour values is 0 to 255.

4 Garbage Matte Generation

The garbage matte in the proposed algorithm is designed to replace the user drawn box in GrabCut. Its function is to give a rough indication of where the foreground layer is, and this facilitates the creation of the data models for both layers. The garbage matte estimation takes inspiration from the algorithm outlined in [15]. The key idea is that pixels which obey the global motion model are part of the background layer. An affine model for global motion is described by

$$\begin{aligned}I_n(\mathbf{x}) &\approx I'_{n-1}(\mathbf{x}) \\ &\approx I_{n-1}(A\mathbf{x} + \mathbf{d})\end{aligned} \tag{5}$$

which states that the intensity values of the current frame, I_n, can be approximated by a motion-compensated neighbouring frame, I'_{n-1}. In this model, A is a 2×2 matrix $\mathbf{d}$ describes the translation. Since pixels in the foreground layer will not obey this motion model, the difference between the current frame and motion compensated neighbour will be large. The two DFDs Δ_b^g and Δ_f^g are estimated according to

$$\begin{aligned}\Delta_b^g(\mathbf{x}) &= I_n(\mathbf{x}) - I_{n-1}(A_b\mathbf{x} + \mathbf{d}_b) \\ \Delta_f^g(\mathbf{x}) &= I_n(\mathbf{x}) - I_{n+1}(A_f\mathbf{x} + \mathbf{d}_f)\end{aligned} \tag{6}$$

where A_b and $\mathbf{d}_b$ are the global motion estimates for the backward image pair while A_f, $\mathbf{d}_f$ are the estimates for the forward image pair. The garbage matte, $l_g(\mathbf{x})$, is then given by

$$l_g(\mathbf{x}) = \begin{cases} 1 & |\Delta_b(\mathbf{x})| > \delta \text{ AND } |\Delta_f(\mathbf{x})| > \delta \\ 0 & \text{Otherwise} \end{cases}. \tag{7}$$

Typically, a value of 20 is used for the threshold δ. The global motion parameters (A_b, $\mathbf{d}_b$, A_f, $\mathbf{d}_f$) are estimated using the F_Align plug-in provided in the Furnace suite of plug-ins [11]. Several published alternatives could be also used, including [21, 9, 16].

As a final stage, the matte is eroded and subsequently dilated. The erosion operation is intended to remove isolated false alarms, while the subsequent dilation operation ensures that as much as possible of the foreground object is contained in the foreground layer of the garbage matte. In the proposed algorithm, the structuring element is larger in the dilation operation than the erosion operation, which implies the the dilation is more significant than the erosion. An example of a garbage matte is shown in Fig. 4.

(a) Original Frame

(b) Garbage Matte

Figure 4: This figure shows an example of a garbage matte produced by MxGrabCut.

5 Data Modelling

As has been stated in Section 1.2, GMMs are used to model the colour and motion data distribution (*i.e.* it defines the Probability Density Function (PDF)) for the foreground and background. In previous matting algorithms, either histograms [2, 7] or GMMs [5, 26, 25] have tended to be used. Since the proposed algorithm adds motion to the model, using histograms is impractical as even a relatively coarse histogram with 20 bins for each of the 7 dimensions (20^7 bins in total) would have too many bins to populated sufficiently. GMMs allow a large amount of data to be represented by a small number of parameters and present a more general model for the data.

For the purposes of the proposed algorithm, each image is said to contain a set of feature vectors $\mathcal{Z}$, where each vector $\mathbf{z} \in \mathcal{Z}$ contains the 7 colour and motion values for one of the pixels in the image. Then, the PDF for the foreground and background layers ($p_{fg}(\mathbf{z})$ and $p_{bg}(\mathbf{z})$ respectively) is described by a GMM as follows

$$p_{fg}(\mathbf{z}) = \sum_k \pi_{fg}(k) G(\mathbf{z}; \vec{\mu}_{fg}(k), R_{fg}(k))$$
$$p_{bg}(\mathbf{z}) = \sum_k \pi_{bg}(k) G(\mathbf{z}; \vec{\mu}_{bg}(k), R_{bg}(k)). \quad (8)$$

In the GMM, each component of the mixture k has a mean $\vec{\mu}(k)$ and a covariance matrix $R(k)$ as well as a weight $\pi(k)$ which is normalised such that

$$\sum_k \pi(k) = 1. \quad (9)$$

Finally, the form of a multivariate Gaussian distribution $G(\mathbf{z}; \vec{\mu}, R)$ is

$$G(\mathbf{z}; \vec{\mu}, R) = \|R\|^{-\frac{1}{2}} \times \exp\Big\{-\frac{1}{2}(\mathbf{z}-\vec{\mu})^T R^{-1}(\mathbf{z}-\vec{\mu})\Big\}. \quad (10)$$

In the proposed algorithm, the foreground and background GMMs are created once the garbage matte, $l_g(\mathbf{x})$, is known (Fig. 3). As a first step, the image pixels are partitioned into two sets according to which layer they belong to in the garbage matte. The GMMs are then created from these sets.

5.1 GMM Initialisation using Mean Shift

In past segmentation algorithms that used GMMs [5, 26, 25], the number of components was chosen arbitrarily the user, with the parameters for the initial GMM components calculated using the algorithm of Orchard and Bouman [22]. Obviously, choosing an arbitrary number of components means that the GMM may not reflect the true order of complexity of the data. In the algorithm outlined here, the number of components in a GMM is identified by applying the Mean Shift [6] algorithm to the corresponding data set. Mean Shift identifies the modes or local maxima of the Kernel Density Estimate (KDE) of the given data set. It also provides a segmentation of the feature vectors. This segmentation is used to assign each feature vector to a component in a GMM[1]. In MxGrabCut, a uniform 7-D spherical kernel is used for mean shift. The radius of the sphere is known as the bandwidth and is the only additional parameter supplied to the algorithm apart from the data. The bandwidth effectively controls the smoothness of the KDE. High bandwidth values result in a smoother KDE which leads to fewer components in the GMMs. A typical value for the bandwidth is 40.

From a practical perspective, the weakness of Mean Shift is its computational complexity which is $\mathcal{O}(n^2)$.[2] Much research has focused on improving the efficiency of mean shift algorithm and here the Path Assigned Mean Shift algorithm [24] is used. To further improve the efficiency of the process, the image and garbage mattes are downsampled horizontally and vertically by a factor of 4. Once the data sets have been divided into components, the parameters of the GMMs (*i.e.* weight, mean and covariance of each component) can be estimated directly from the data of each component. The weights are given by the fraction of points in the set that belong to each component.

5.2 Optimising GMM Parameters using the EM algorithm

From these initial models, the parameters can be refined using the Expectation-Maximisation (EM) algorithm[8]. Using the current set of GMM parameters, the expected component $\hat{k}(\mathbf{z})$ for each point $\mathbf{z}$ in the appropriate partitioned set is estimated (the E-Step). This corresponds to the Maximum Likelihood (ML) component according to

$$\hat{k}(\mathbf{z}) = \arg\max_k \Big\{\pi(k) G(\mathbf{z}; \vec{\mu}(k), R(k))\Big\}. \quad (11)$$

The second stage of the EM algorithm (the maximization stage or M-step) involves finding the maximum likelihood parameter values given the expected component values from the E-Step. For a GMM, this merely involves re-estimating the mean and covariance matrix of the points that belong to each expected component. The weight for each component is again given by the fraction of points in the partitioned set belonging to the expected component.

[1] Each local maximum is associated with a GMM component

[2] n represents the number of Pixels.

EM is used in two of the stages in MxGrabCut (Fig. 3). Firstly, it is used in the GMM Initialisation stage to refine the GMMs after they have been created from the mean shift procedure. EM is repeated a number of times until successive estimates of the GMM parameters converge. This is limited in practice to 20 iterations. It is also used in the GMM refinement stage, where a single EM iteration is performed to update the GMM parameters after each segmentation iteration.

6 Segmentation

The segmentation stage of the proposed algorithm extends the Bayesian framework in Eq. 2 by including the motion data. Given the full set colour and motion feature vectors $\mathcal{Z}(\mathbf{x})$, the posterior PDF $p(l(\mathbf{x})|\mathcal{Z}(\mathbf{x}))$ now factorises as

$$p(l(\mathbf{x})|\mathcal{Z}(\mathbf{x})) \propto p_d(\mathcal{Z}(\mathbf{x})|l(\mathbf{x})) \times p_s(l(\mathbf{x})|L). \quad (12)$$

As in Bayes Matting and GrabCut, the *Data Likelihood* in this algorithm is derived from the GMMs for the foreground and background. Given foreground and background GMMs $p_{bg}(\mathbf{z})$ and $p_{fg}(\mathbf{z})$, the data likelihood is defined as

$$p_d(\mathcal{Z}(\mathbf{x})|l(\mathbf{x})) = \begin{cases} p_{bg}(\mathcal{Z}(\mathbf{x})) & \text{for } l(\mathbf{x}) = 0 \\ p_{fg}(\mathcal{Z}(\mathbf{x})) & \text{for } l(\mathbf{x}) = 1. \end{cases} \quad (13)$$

This implies that for each pixel in the image the likelihood for each label is given by the value of the PDF of the appropriate GMM.

Like GrabCut, the *Prior* probability enforces a spatial smoothness prior on $l(\mathbf{x})$. The segmented image $l(\mathbf{x})$ is considered to be a Gibbs Random Field. It is in the form of a contrast dependent prior [18] and is expressed as

$$p_s(l(\mathbf{x})|L) = \exp\left\{-\Lambda_s \sum_{y \in \mathcal{N}_s(\mathbf{x})} U(\mathbf{x},\mathbf{y})|l(\mathbf{x}) - l(\mathbf{y})|\right\} \quad (14)$$

where $N_s(\mathbf{x})$ is a spatial neighbourhood of $\mathbf{x}$. $U(\mathbf{x},\mathbf{y})$ is the contrast dependent energy term which is defined as

$$U(\mathbf{x},\mathbf{y}) = \|\mathbf{x}-\mathbf{y}\|^{-1} \exp\{-\beta\|\mathbf{C}(\mathbf{x}) - \mathbf{C}(\mathbf{y})\|^2\} \quad (15)$$

where $\|\mathbf{x}-\mathbf{y}\|$ is the euclidean distance between the two neighbours and

$$\beta = \left(2\left\langle \left\|\mathbf{C}(\mathbf{x}) - \mathbf{C}(\mathbf{y})\right\|^2 \right\rangle\right)^{-1}. \quad (16)$$

In this context, $\langle . \rangle$, denotes expectation of the enclosed term over the entire image. Λ_s is a tuning parameter which, when large, increases spatial smoothness of the matte. A value of 50 is used following the suggestion in [25].

The prior encourages spatial smoothness since the probability is decreased whenever two neighbours $\mathbf{x}$ and $\mathbf{y}$ are different. The degree of smoothness is dictated by the contrast dependent energy $U(\mathbf{x},\mathbf{y})$, which is designed to enforce smoothness when the contrast is low [25, 2]. In MxGrabCut, the RGB contrast is chosen since the perceived foreground/background boundaries occur at image edges rather than motion discontinuities.

6.1 Solving for $l(\mathbf{x})$

The desired solution $\hat{l}(\mathbf{x})$ is the matte that maximises the joint posterior PDF over every pixel. In equation form, this is expressed as

$$\hat{l}(\mathbf{x}) = \arg\max_{l(\mathbf{x})} \prod_{\mathbf{x}} p(l(\mathbf{x})|\mathcal{Z}(\mathbf{x})). \quad (17)$$

It becomes more convenient to express the problem as an energy minimisation problem by taking the negative log of the posterior. The solution now becomes

$$\hat{l}(\mathbf{x}) = \arg\min_{l(\mathbf{x})} \sum_{\mathbf{x}} (-\ln\{p(l(\mathbf{x})|\mathcal{Z}(\mathbf{x}))\}). \quad (18)$$

There are numerous suitable methods to solve for $\hat{l}(\mathbf{x})$. In MxGrabCut, the max-flow Graph Cuts approach proposed in [2, 3] is used. This energy minimisation algorithm estimates the global minimum energy for a two label problem in an efficient manner, unlike other methods such as Iterated Conditional Modes [1] and Simulated Annealing [12]. Furthermore, the framework outlined in [2] allows hard constraints to be placed on $\hat{l}(\mathbf{x})$ by manipulating the likelihood function. However, in the segmentation stage of the proposed algorithm, no such constraints are placed on the matte.

6.2 Iterative Segmentation

A segmentation stage is performed each time the GMMs are refined. Once the new matte $l(\mathbf{x})$ is calculated, it is then used to repartition the data set $\mathcal{Z}$. When the next GMM refinement operation is executed, the data GMMs are trained on a more accurate partition. With each refinement, the *maximum-a-posteriori* estimate of the joint PDF of the matte and GMMs, $P(l(\mathbf{x}), p_{fg}(\mathbf{z}), p_{bg}(\mathbf{z})|\mathcal{Z}(\mathbf{x}))$, converges to a local maximum of the PDF. Iterations should continue until the estimate of matte and GMM parameters have converged. In practice, the number of iterations is capped, typically to 10 iterations in the examples shown in this paper.

7 Results and Discussion

Figures 5 and 6 show examples of mattes taken from scenes with either a static background or a background undergoing an affine motion and Fig. 7 shows the improved matte obtained when using the motion as well as colour to train the GMMs. In the best case examples, the result is comparable in quality to a manually drawn matte, with a tight fit of the matte to the object image edge. However, in many examples, there is a mislabelling around the boundary of the matte, resulting in parts of the background being included in the foreground layer. Any mislabelling is typically due to an insufficient garbage matte or because the GMMs do not realistically model or foreground and background. The mattes can be improved in a semi-automatic manner, as outlined in [25, 2], by allowing the user to mark the mislabelled regions.

A binary segmentation of a frame is not sufficient for compositing applications as it does not model the regions of

(a) Original Images

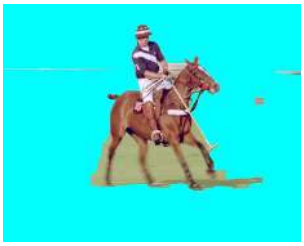
(b) Extracted Foreground Layers

Figure 5: This figure shows examples of the foreground layers extracted by MxGrabCut from two sequences in which there is no camera motion. In the first sequence, both the actor and its shadow are detected as foreground. Although the detected layer boundary generally agrees with the image edge, there is some mislabelling around the layer boundary in each example.

(a) Original Images

(b) Extracted Foreground Layers

Figure 6: This figure shows results of MxGrabCut applied to sequences in which there is camera motion. The detected foreground/background boundary closely follows the true path, although again there is some mislabelling of background as foreground.



(a) Foreground Layer using Motion and Colour

(b) Foreground Layer using only Colour

(c) Foreground Layer using Motion and Colour

(d) Foreground Layer using only Colour

Figure 7: The above images show two example of how using GMMs of the joint colour/motion feature vectors improves the robustness of the algorithm compared to using colour alone. When motion is used, the rate of mislabelling is dramatically reduced. In both examples, all other parameters are the same and the same garbage matte is used.

transparency that are typically found around object edges. Other algorithms, such as the the Bayes Matting algorithm of [4, 5], allow for continuous label values over the range 0 to 1. Both Kokaram *et al.* [15] and Rother *et al.* [25] outline how binary mattes can be used to estimate these non-binary mattes by generating a trimap from the binary matte. In the example shown in Fig. 8, a trimap is generated by performing a small erosion and dilation on the binary matte. The non-binary matte is generated from the trimap using the Bayes Matting technique [4, 5].

The above examples show that good mattes can be obtained when the background motion is affine. The affine and related motion models assume that the background is planar, and thus do not allow for parallax in the background (See Fig. 9). Fig. 10 shows that the algorithm fails in a sequence with parallax in the background. Because of the parallax, the motion in the background can not be compensated accurately, which leads to a failure in the garbage mate estimation. Furthermore, the parallax clutters the motion field, making it more difficult to distinguish between foreground and background motion. Thus, even when a user drawn garbage matte is used, the algorithm produces a poor quality matte.

It is obvious that a better method for modelling such motion is needed. For example, in [27] an algorithm has been proposed to extract multiple layers from shots containing parallax. However, an ambiguity still exists between foreground objects that are moving independently of the background and objects that lie at various depths in the background. A potential solution presents itself when a scene is captured using multiple cameras. The extra views of the foreground object at each time instance allows the depth map of the scene to be extracted. Depth can then be used as an additional feature for segmentation alongside colour and motion hence improving the ability to pull mattes. The applicability of such solutions to digital cinema post-production will increase in line with the growth of 3D cinema.

8 Final Comments

This paper has introduced a new algorithm for automatically segmenting frames into foreground and background layers. The algorithm shows how a garbage matte can be generated automatically, removing the need for user-intervention in the segmentation process. Furthermore, it has been shown that including motion information improves the robustness of the mattes and that the complexity of the GMMs can be chosen adaptively using Mean Shift. There are a number of leads for future research in this area. The problem of foreground segmentation in sequences containing parallax has already been discussed. Other possible extensions could be to reduce the dimensionality of the data space to save on computational complexity or to propagate mattes and GMMs from neighbouring frames to improve temporal consistency of the mattes throughout the sequence.

Acknowledgements

The work presented in this paper was part funded by grants from the EU Marie Curie Transfer of Knowledge project Axiom and Adobe Systems Incorporated.

References

[1] J. E. Besag. On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society B*, 48:259–302, June 1986.

[2] Y. Boykov and M.-P. Jolly. Interactive graph cuts for optimal boundary & region segmentation in n-d images. In *IEEE Internation Conference on Computer Vision*, 2001.

[3] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Transactions on pattern Analysis and Machine Intelligence (PAMI)*, 26:1124–1137, 2004.

[4] Y.-Y. Chuang, A. Agarwala, B. Curless, D. H. Salesin, and R. Szeliski. Video matting of complex scenes. *ACM Transactions on Graphics*, 21(3):243–248, July 2002. Sepcial Issue of the SIGGRAPH 2002 Proceedings.

[5] Y.-Y. Chuang, B. Curless, D. Salesin, and R. Szeliski. A bayesian approach to digital matting. In *IEEE Conference*

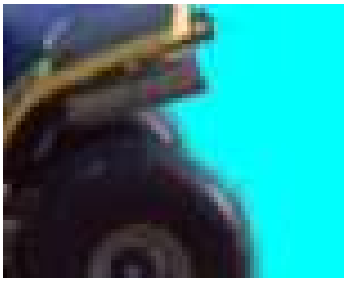
(a) Extracted Foreground using Binary Matte

(b) Definite Foreground Region

(c) Definite Background Region

(d) Extracted Foreground using Non-Binary Matte

(e) Zoom of Binary Foreground Layer



(f) Zoom of Non-Binary Foreground Layer

Figure 8: This figure shows how the binary mattes produced by MxGrabCut can be used to generate non-binary mattes. The binary matte is used to define regions of definite foreground and background (shown as layers in (b) and (c)) by eroding and dilating the matte respectively. A non-binary matte (d) can then be estimated by the Bayes Matting algorithm. (e) and (f) show that the non-binary matte blends with the cyan background in a more visually pleasing manner.

on Computer Vision and Pattern Recognition (CVPR), volume 2, pages 264–271, 2001.

[6] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:603–619, May 2002.

[7] A. Criminisi, G. Cross, A. Blake, and V. Kolmogorov. Bilayer segmentation of live video. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 53–60, 2006.

[8] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society B*, 39:1–38, 1977.

[9] F. Dufaux and J. Konrad. Efficient, robust and fast global motion estimation for video coding. *IEEE Transactions on Image Processing*, 9, 2000.

[10] K. Fukunaga and L. D. Hostetler. The estimation of a density function, with applications in pattern recognition. *IEEE Transactions on Information Theory*, 21:32–40, 1975.

[11] *Furnace User-Guide*. http://www.thefoundry.co.uk/FoundryFileServer/Furnace4.0v4_Nuke.pdf.

[12] S. Geman and D. Geman. Stochastic, relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and machine Intelligence*, 6:721–741, November 1984.

[13] B. K. P. Horn and B. G. Schunk. Determining optical flow. *Artificial Intelligence*, 17, 1981.

[14] N. Jojic and B. Frey. Learning flexible sprites in video layers. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2001.

[15] A. Kokaram, B. Collis, and S. Robinson. Practical motion based video matting. In *proceedings of the IEE European Conference on Visual Media Production (CVMP'05)*, pages 130–136, London, UK, November 2005.

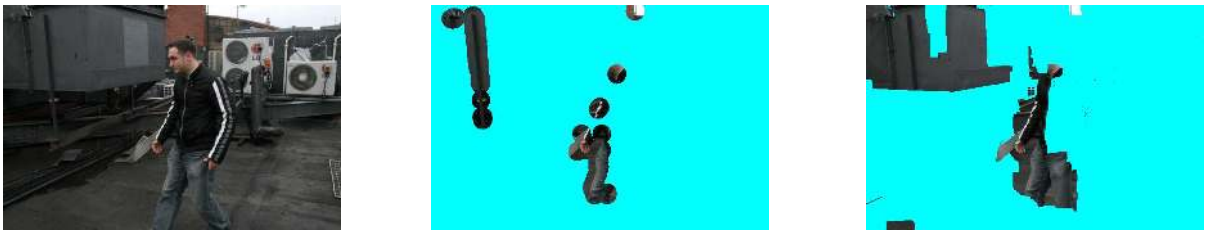
[16] A. Kokaram and P. Delacourt. A new global motion estimation algorithm and its application to retrieval in sports events. In *IEEE International Workshop on Multimedia Signal Processing*, Cannes, France, 2001.

[17] Anil Kokaram. *Motion Picture Restoration: Digital Algorithms for Artefact Suppression in Degraded Motion Picture Film and Video*, chapter 2. Springer Verlag, 1998.

[18] J. Konrad and E. Dubois. Bayesian estimation of motion vector fields. *IEEE trans.on Pattern Analysis and Machine Intelligence*, 14:910–927, September 1992.

[19] C. D. Kuglin and D. C. Hines. The phase correlation image alignment method. In *IEEE International*

Figure 9: These images show 4 consecutive frames from a sequence in the camera motion causes parallax in the background. This can be seen from the motion of the grey tank on the left over the brick fall in the far background.



(a) Original Frame (b) Garbage Matte (c) Estimated foreground

(d) Manually drawn garbage matte (e) Foreground layer estimated from the manually drawn garbage matte

Figure 10: This figure shows the difficulties of matting in sequences containing parallax. In this example the foreground layer produced by MxGrabCut (c) is poor because the estimated garbage matte is also poor(b). However, even when a user-drawn garbage matte is used (d), the estimated matte (e) is still of low quality.

Conference on Cybernetics and society, pages 163–165, 1975.

[20] E. N. Mortensen and W. A. Barrett. Intelligent scissors for image composition. In *ACM SIGGRAPH*, pages 191–198, 1995.

[21] J. M. Odobez and P. Bouthemy. Robust multiresolution estimation of parametric motion models. *Journal of Visual Communication and Image Representation*, 6, 1995.

[22] M. Orchard and C. Boumann. Colour quantisation of images. *IEEE Transactions on Signal Processing*, 39:2677–2690, December 1991.

[23] P. Perez, A. Blake, and M. Gangnet. Jetstream: Probabilistic contour extraction with particles. In *IEEE International Conference on Computer Vision*, volume 2, pages 524–531, July 2001.

[24] A. Pooransingh, C.-A. Radix, and A. Kokaram. The path assigned mean shift algorithm: A new fast mean shift implementation for colour image segmentation. In *IEEE International Conference on Image Processing*, 2008.

[25] C. Rother, V. Kolmogorov, and A. Blake. Grabcut - interactive foreground extraction using iterated graph cuts. In *ACM SIGGRAPH Conference*, pages 309–314, 2004.

[26] M. Ruzon and C. Tomasi. Alpha estimation in natural images. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 18–25, 2000.

[27] P.H.S. Torr, R. Szeliski, and P. Anandan. An integrated bayesian approach to layer extraction from image sequences. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23:297–303, March 2001.

[28] J. Y. A. Wang and E. H. Adelson. Representing moving images with layers. *IEEE Transactions on Image Processing*, 3:625–638, September 1994.