Improved GrabCut Segmentation via GMM Optimisation

Daniel Chen, Brenden Chen, George Mamic, Clinton Fookes, Sridha Sridharan
Image and Video Research Laboratory
Queensland University of Technology
GPO Box 2434, 2 George St.
Brisbane, Queensland 4001
{*daniel.chen*, *chong.chen*, *c.fookes*, *s.sridharan*}@qut.edu.au

Abstract

Semi-automatic segmentation of still images has vast and varied practical applications. Recently, an approach "GrabCut" has managed to successfully build upon earlier approaches based on colour and gradient information in order to address the problem of efficient extraction of a foreground object in a complex environment. In this paper, we extend the GrabCut algorithm further by applying an unsupervised algorithm for modelling the Gaussian Mixtures that are used to define the foreground and background in the segmentation algorithm. We show examples where the optimisation of the GrabCut framework leads to further improvements in performance.

1 Introduction

Foreground image segmentation is the task of identifying an object from the background. Existing approaches rely on colour information, edge information or region connectivity where the overall aim is to achieve accurate segmentation with minimal user interaction.

A recent method of using Graph Cuts by Boykov and Jolly [4] combines both colour and edge information. The technique was made iterative by Rother *et al.* [9] and reduced the amount of user interaction required. This process named "GrabCut" requires the user to draw a rough bounding box around the foreground object in an image. This box roughly splits the image into background and foreground regions allowing the colours in each region to be statistically modeled using Gaussian Mixture Models (GMMs). This paper discusses two modifications that can be made to the GrabCut process to improve segmentation performance.

The original GrabCut algorithm uses a fixed number of Gaussians in each GMM, however it is shown in this paper that the number of Gaussians used to model the foreground and background can have a significant effect on segmentation performance. The addition of the CLUSTER algorithm [3] analyses the foreground and background regions prior to segmentation and estimates the optimal number of Gaussians needed in each GMM in order to best model each region.

GrabCut also utilizes edge information to identify the border between foreground objects and background. This is done by analysing the gradient (change in color) between two neighboring pixels. Altering the algorithm to construct the background GMM using only background pixels in the bounding box in subsequent iterations gives a more accurate model of background colors around the selected object. This effectively focuses the method on the region of interest by removing isolated background colors from the GMM that are not spatially close to the foreground object.

These modifications to GrabCut are tested on real-world images used by the GrabCut authors and are available on their website [1]. This dataset contains both ground truth data and bounding box locations, allowing for consistent benchmarking and quantitative performance evaluation of the proposed method. Test cases using images from the Temple dataset [10] are also included along with discussions of these results.

2 Background

The ability to accurately extract a foreground object from an image is desirable functionality for any consumer level graphics package. It also has applications in the realm of computer based intelligent surveillance where recognition or tracking may need to be performed on foreground objects such as people or vehicles. The ability to extract these moving objects from a dynamically changing background is a crucial step.

The most rudimentary form of digital image segmentation requires the user to manually specify each foreground

pixel individually. Although highly accurate, it is both slow and tedious for what is a cognitively simple task. More intelligent methods of object segmentation fall into two categories: border-based and region-based methods [6].

Border-based techniques such as intelligent scissors [7] require the user to specify seed points around the border of the object. The algorithm treats the image as a graph where each pixel is a node and assigns each path with a cost based on the colour gradient between the two pixels. The task of defining the border around the foreground object can be done by computing the minimum cost path between seed points. Although simpler than manually selecting pixels, the method still requires a significant amount of user input. This is especially true in high resolution images where a large number of seed points are required to accurately draw around the object.

More recent region-based techniques improved the task of segmentation by reducing the amount of user interaction required. Intelligent Paint [8] and the well known Magic Wand tool in Photoshop are examples where the user is only required to specify a given region in the foreground object. From this region, colour statistics are computed and connected neighboring regions that display similar statistics are segmented out. A newer method by Blake *et al.* [2] require the user to draw a rough thick border around the object which essentially establishes three regions, a definite foreground, a definite background and a border region that contain both. This is known as the Trimap from which colour statistics for each region are modeled using Gaussian Mixture Markov Random Field (GMMRF).

A popular technique known as Graph Cut [4] builds on these earlier works by combining both border and region information to improve segmentation performance. The method requires the user to specify both a foreground and background region from which colour statistics can be modeled using GMMs. Like Intelligent Scissors, a graph is set up where every pixel is represented by a node in the graph. The nodes are linked to its neighbouring pixels, with the edge connecting them given a cost weighted by the difference in the pixels' colour values. Two other nodes, called the source and sink node, represent the foreground and background. These nodes are each connected to every pixel node. A cost is also applied to these edges according to the probability of the connected pixel's colour occurring in the foreground/background distribution. Once the graph is set up, segmentation is done by performing a graph cut using the min-cut/max-flow algorithm [5] which attempts to optimally separate the source and sink nodes (foreground and background pixels) from each other. A method named Lazy Snapping [6] adds a further step where the initial segmentation boundary produced by the Graph Cut can be manipulated by the user through draggable vertices allowing for a more accurate border.

GrabCut [9] improves on Graph Cut by further reducing the amount of user interaction through provision of a simple bounding box around the foreground object. The bounding box splits the image into two regions, a definite background region and a mixed background/foreground region from which two GMMs are created to model the background and foreground. Unlike Graph Cut the foreground is not explicitly marked and as such the foreground GMM will be inaccurate due to the presence of background pixels in the mixed region. To overcome this Graph Cut is performed iteratively, since after the first iteration some background pixels in the mixed region will become correctly classified allowing the GMMs to be updated for next iteration. This process repeats until the segmentation converges and no more changes occur.

3 Image Segmentation by Optimised GrabCut

This section describes the optimised GrabCut segmentation algorithm.

3.1 RGB Data Models

The input image to the algorithm is an RGB image, Y, composed of n pixels. The user then selects a foreground region F and the remainder of the image is assigned to the background B. Each region is then modelled using GMM's with K subclasses, for which the following parameters are required to specify the k^{th} subclass,

$$\begin{aligned}\pi_k &= \text{the weight assigned to pixel of subclass } k,\\ \mu_k &= \text{the mean vector of the subclass},\\ R_k &= \text{the covariance matrix of the subclass.}\end{aligned}$$

Hence, if K is the total number of subclasses then we can use the notation π, μ and R to denote the sets ${\pi_k}_{k=1}^{K}$, ${\mu_k}_{k=1}^{K}$ and ${R_k}_{k=1}^{K}$. The complete set of parameters that are then required to define the GMM are K and $\Sigma = (\pi, \mu, R)$. The probability density function for pixel y_n for which the random variable $X_n = k$ (denoting the number of subclasses) is given by,

$$p_{y_n|x_n}(y_n|k,\Sigma) = \frac{1}{(2\pi)^{M}/2}|R_k|^{-1/2}exp{-}0.5(y_n-\mu_k)'R_k^{-1}(y_n-\mu_k). \tag{1}$$

As we don't know the subclass that each of the pixels belongs to we define a conditional probability as follows,

$$p_{y_n}(y_n|\Sigma) = \sum_{k=1}^{K}\pi_k p_{y_n|x_n}(y_n|k,\Sigma). \tag{2}$$

From this we can then calculate the log of the probability for Y as follows,

$$\log p_y(y|K,\Sigma) = \sum_{n=1}^{N} \log(p_{y_n}(y_n|\Sigma). \quad (3)$$

The objective is then to estimate K and $\Sigma = (\pi, \mu, R)$. This can be done using the minimum description length (MDL) criterion as applied to the equation,

$$MDL(K,\Sigma) = \sum_{n=1}^{N} \log(p_{y_n}(y_n|\Sigma) + 0.5L\log(NM). \quad (4)$$

The major difference between the MDL and other methods such as EM and AIC is the dependence on the penalty term on the total number of data values NM which in practice prevents overfitting of the data model. An effective algorithm for the solution of the parameter set using the MDL criterion was developed by Bouman *et al.* [3] and may be summarised as follows:

1. Initialise the number of subclasses K_0, for the optimisation.
2. For all $K > 1$,
   - Initialise $\Sigma = (1/K_0, y_n, \frac{1}{N}\sum_{n=1}^{N} y_n y_n^t)$,
   - Apply an iterative EM algorithm until the change in MDL(K,Σ) is less than a threshold,
   - Record Σ and MDL(K,Σ),
   - Reduce number of clusters.
3. Choose K and Σ_K which minimise the MDL.

The Gibbs energy for segmentation was formulated by [4] and can be written as follows,

$$E(\alpha, k, \Sigma, y_n) = \sum p_{y_n}(y_n|\Sigma) + V(\alpha, y), \quad (5)$$

where $\alpha_n \in \{0, 1\}$ assigns to each pixel a unique GMM component either from the background or the foreground model. The smoothness term is taken to be the same as the term that was used in the original GrabCut algorithm and is as follows,

$$V(\alpha, y) = \gamma \sum_{(m,n)\in C} [\alpha_n \neq \alpha_m] \exp(-\beta\|z_m - z_n\|^2), \quad (6)$$

where, the constant $\sigma = 50$ after using a training set set and optimising against ground truth [9]. The constant $\beta = (2E((z_m - z_n)^2))^{-1}$ was chosen by [4], where $E(.)$ is the expectation operator.

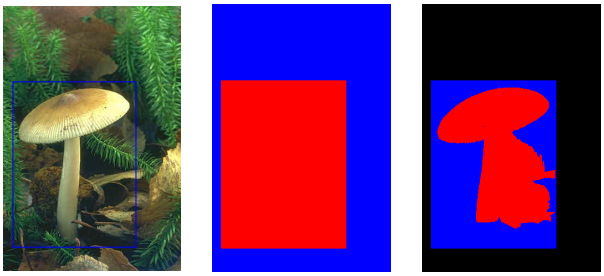


Figure 1. Background optimisation procedure. Red is foreground, blue is background. Left: Original image with selection box. Centre: Initial labelling. Right: After 1st segmentation.

3.2 Adaptive Iterative Energy Segmentation

The energy minimisation that is used is based upon the iterative procedure that was presented in [9] *et al.* and the algorithm is as follows:

1. User selects foreground region of image, hence initalising F and B.
2. Initalise $\alpha_n = 0n \in B$ and $\alpha_n = 1n \in F$.
3. Calculate optimal GMMs for foreground and background.
4. Estimate segmentation using mincut to solve for $\min_k E(\alpha, k, \Sigma, y_n)$.
5. Reassign B and F according to the results of the segmentation, limiting B to only pixels within the original selection window.
6. For $\delta E(\alpha, k, \Sigma, y_n) > \epsilon$,
   - Calculate optimal GMMs for foreground and background.
   - Estimate segmentation using mincut to solve for $\min_k E(\alpha, k, \Sigma, y_n)$.
   - Reassign B and F according to the results of the segmentation.

The key differences between the iterative energy segmentation algorithm proposed here and that of [9] are twofold. Firstly, with each iteration the number of GMMs is varied based upon the MDL driven optimisation strategy described in the previous sub-section. This feature removes the variations that may be induced on the segmentation due to the arbitrary setting of the K parameter by the user.

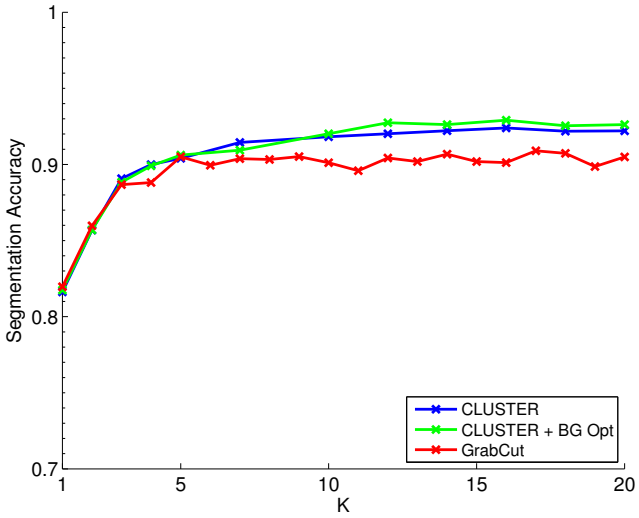


Figure 2. Change of segmentation accuracy with respect to K, the no. of sub-classes.

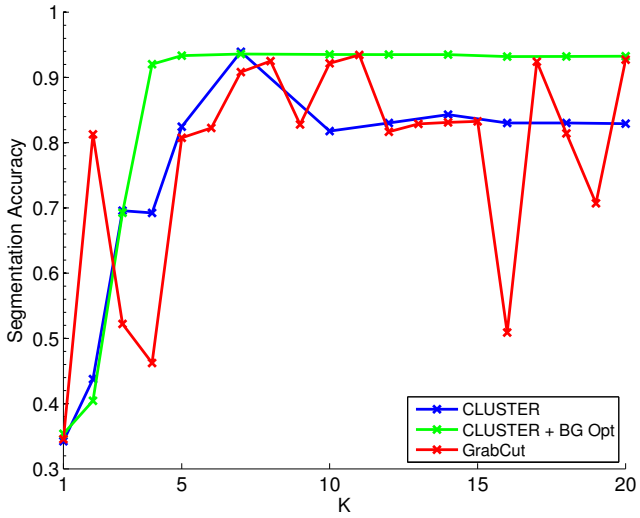


Figure 3. Change in segmentation accuracy with varying K for image 24077.jpg.

Secondly, following the first pass of the segmentation, the background B is constrained to only be the pixels in the original foreground selection window, where as the original algorithm considers the entire image. This provides a more localised modelling of the background colour distribution, theoretically providing a better segmentation result. As a consequence, this also reduces the total number of pixels which are considered in the iterative energy segmentation and produces considerable improvements in the speed of the MDL optimisation. This optimisation procedure is illustrated in Figure 1.

4 Results and Discussion

The real-world dataset [1] is used to evaluate the performance of the proposed algorithm. The dataset contains 49 images of a wide range of objects in scenes of varying complexity. The dataset also includes bounding box coordinates and ground truth data, although two of the supplied bounding boxes were poorly chosen and these images were subsequently ignored in our tests. Ground truth data allows different segmentation schemes to be benchmarked using scores based on the percentage of pixels inside the bounding box that are correctly classified. The results are shown in Figure 2.

The graph shows the average segmentation accuracy over the 47 images with respect to the number of GMM components used. It can be seen that the alterations to the original algorithm provides a small, but clear improvement.

The terms used in the figures are as follows. 'CLUSTER' refers to the MDL optimisation described in Section 3. 'BG Opt' refers to the background optimisation scheme applied to subsequent iterations. It should be noted that K with respect to the CLUSTER algorithm refers to the set upper limit allowed for the value of K, and that the actual number of components used in the foreground/background modelling dynamically varies between each iteration. A poor selection of K can have a significant detrimental impact on the segmentation result.

Rother *et al.* [9] stated to use 5 components in the GMMs and this can be verified from the results shown. It can be seen that the segmentation accuracy plateaus after $K = 5$. It needs to be stressed, however, that the results shown are averaged over all the images in the database. An image can have more or less components as its optimal value of K. Increasing K can result in a worse segmentation as over fitting occurs when modelling the colour distributions. An example of this can clearly be seen in Figure 3. As the CLUSTER algorithm attempts to estimate the optimal number of Gaussian components, its results do not vary as much from changing the value of K. This allows an arbitrary value of K with a sufficient upper bound to be safely set. Some example segmentaiton results from this dataset can be seen in figures 5 and 4.

The Temple dataset was also used to evaluate the algorithm. This dataset contains 233 images from varying views of a scaled model of "Temple of the Dioskouroi". The model exists over a uniform background and thus is quite easy to segment accurately. Holes exists in the model however, and it is found that the original GrabCut has difficulty correctly segmenting these holes. Ignoring the holes, it is found that $K = 5$ is able to produce a clean segmentation

Figure 4. Segmentation examples. Top: Original image + bounding box. Middle: Grabcut Bottom: CLUSTER + BG Opt.

of the temple in most images, bar some minor blemishes. $K > 5$ provides small benefits as there is little room for improvement. Having $K < 5$ however, begins to introduce severe artifacts into the segmentation result. This is demonstrated in Figure 6.

The reason for the difficulty in detecting the holes in these images is not immediately apparent. The holes, like the majority of the background, is black while the foreground object is significantly brighter. When viewed under a different light, however, the problem becomes clear. Figure 7 shows a histogram equalised greyscale example image, colour-mapped for clarity. It can be seen that the intensities within the holes are different to the rest of the background. As all pixels within the user selected boundary are initialised as foreground, these intensity values which are rare in the initial background model, would not be classified as background.

No ground truth are available for the segmentation of this dataset. As the segmentation of the boundary has been deemed to be accurate ($K >= 5$), quantitative assessment of the algorithms' performance will be based on its ability to detect the holes. The table in Figure 8 shows the results.

Tests were run through all 233 images using the GrabCut algorithm with $K = 2, 3, 4, 5, 7, 10$ and both the GMM optimisation and GMM optimisation with background optimisation with a maximum K of 7. The holes are classified into two types, 'major' and 'minor', according to their sizes. Holes are classified as 'minor' when it only appears as a narrow strip. This is illustrated in Figure 9a. A false detection is when an area (usually in shadow) is incorrectly segmented as a hole, as shown in Figure 9b.

From the results, it can be seen that the hole detection rate increases as K decreases. This can be explained in the modelling of the colourspace. If K is large, components of the GMM can be used to directly represent the colours within the holes, resulting in the situation described earlier. However, if K is small, there may be insufficient components available to model just these colours itself, and will need to be represented with the black that occurs in the background by a single Gaussian. This allows the holes to be segmented as background in subsequent iterations of the algorithm.

While having a small value of K (< 5) is able to better detect holes, it is likely not worth the cost segmentation errors that may occur. The CLUSTER algorithm, however, is able to provide better hole detection while maintaining clean segmentation with the maximum Gaussian mixture number set at 7. This is most likely due to its ability to dynamically vary the number of Gaussians in both the foreground and background models independently according to the MDL based criterion.

The main improvement is achieved when the background optimisation is included. By constraining the background model to only the pixels within the user selection on all but the initial iteration, a better representation of the local colour distribution around the object is achieved. From Fig-

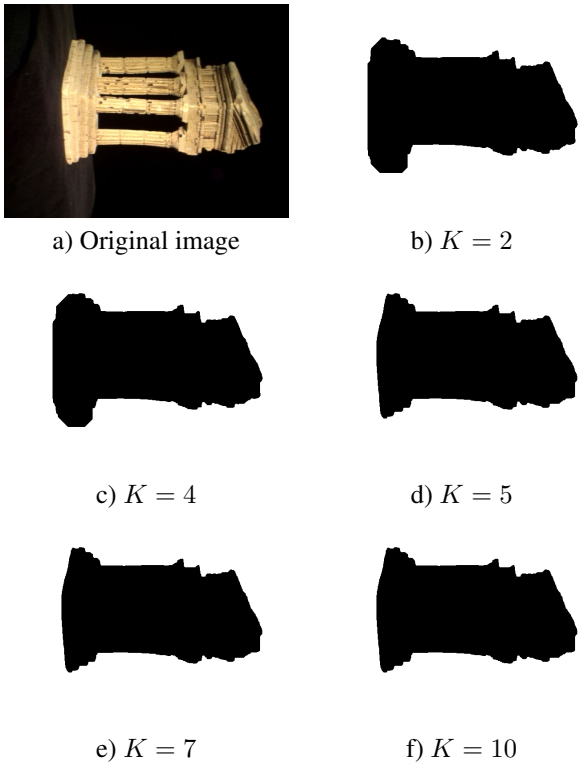
189080.jpg 208001.jpg grave.jpg

Figure 5. Segmentation examples. Top: Original image + bounding box. Middle: Grabcut Bottom: CLUSTER + BG Opt.

ure 7, it can be seen that the background colours near the object are more similar to those in the holes than the rest of the image, pushing the weightings of the pixels within the holes toward that of the background. Figure 10 shows the segmentation results of a test image.

5 Conclusion

This paper describes two modifications to the GrabCut object segmentation algorithm that improve segmentation accuracy without increasing user interaction. The CLUSTER optimization technique is shown to improve segmentation performance and stability by removing the negative effects caused by using an unsuitable value of K. The addition of background GMM optimization helps GrabCut focus on the region of interest around the border of the foreground object. This is shown to significantly improve segmentation performance when the foreground object contains holes. It is important to keep in mind that the original GrabCut algorithm is a two step process of an initial automatic segmentation followed by manual user touchups (if necessary). By improving the accuracy of the initial segmentation via the two GMM optimization techniques the reliance on manual touchups can be decreased or even completely eliminated in some images.



a) Original image b) $K = 2$ c) $K = 4$ d) $K = 5$ e) $K = 7$ f) $K = 10$

Figure 6. Temple segmentation (holes filled).

6 Acknowledgments

This project was supported by the Australian Government Department of the Prime Minister and Cabinet.

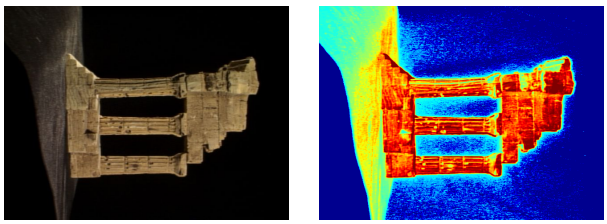


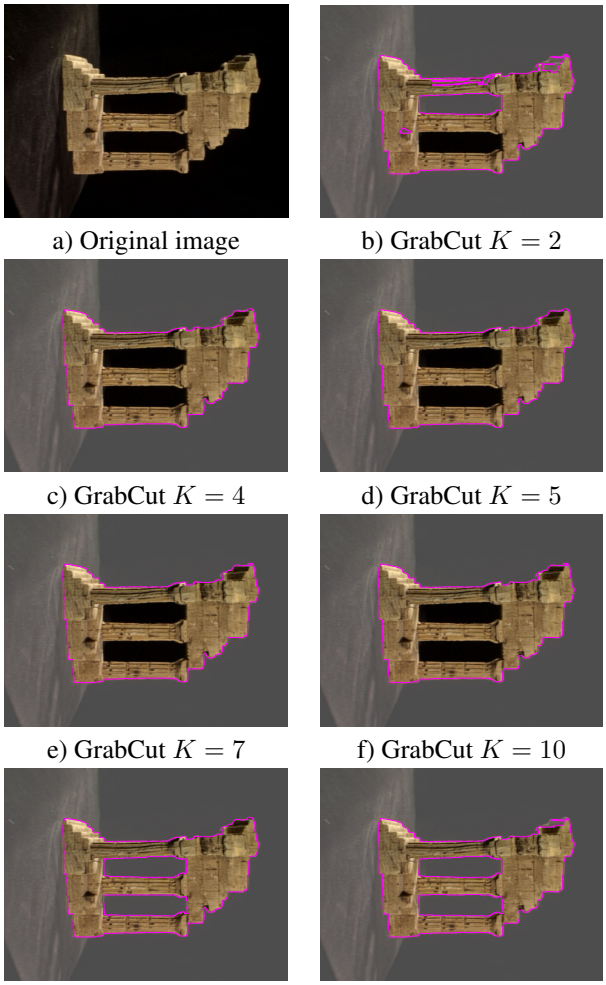
Figure 7. Intensity map.

	Major	Minor	False
Ground Truth	279	75	-
GrabCut $K = 2$	135	15	12
GrabCut $K = 3$	144	13	0
GrabCut $K = 4$	139	8	0
GrabCut $K = 5$	128	4	0
GrabCut $K = 7$	116	5	0
GrabCut $K = 10$	98	0	0
CLUSTER	155	29	1
CLUSTER + BG Opt	239	39	0

Figure 8. Temple segmentation results.

a) 1 'major' and 2 'minor' b) 1 false detect

Figure 9. Hole classification.



a) Original image b) GrabCut $K = 2$

c) GrabCut $K = 4$ d) GrabCut $K = 5$

e) GrabCut $K = 7$ f) GrabCut $K = 10$

e) CLUSTER f) CLUSTER + BG Opt

Figure 10. Temple segmentation example.

References

[1] http://research.microsoft.com/vision/cambridge/i3l/segmentation/GrabCut.htm.

[2] A. Blake, C. Rother, M. Brown, P. Perez, and P. Torr. Interactive image segmentation using an adaptive gmmrf model. In *8th European Conference on Computer Vision*, 2004.

[3] C. A. Bouman. Cluster: An unsupervised algorithm for modeling gaussian mixtures. Technical report, Prude University, July 2005.

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

[5] 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*, 26(9):1124–1137, 2004.

[6] Y. Li, J. Sun, C.-K. Tang, and H.-Y. Shum. Lazy snapping. In *SIGGRAPH '04: ACM SIGGRAPH 2004 Papers*, pages 303–308, 2004.

[7] E. N. Mortensen and W. A. Barrett. Intelligent scissors for image composition. In *SIGGRAPH '95: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pages 191–198. ACM, 1995.

[8] L. J. Reese and W. A. Barrett. Image editing with intelligent paint. In *Proceedings of Eurographics 2002*, pages 714–723, 2002.

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

[10] S. M. Seitz, B. Curless, J. Diebel, D. A. Scharstein, and R. A. Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 1, pages 519–528, 2006.