Cross Anisotropic Cost Volume Filtering for Segmentation

Vladislav Kramarev, Oliver Demetz, Christopher Schroers, and Joachim Weickert

Mathematical Image Analysis Group,
Faculty of Mathematics and Computer Science,
Campus E1.7, Saarland University, 66044 Saarbrücken, Germany
{kramarev,demetz,schroers,weickert}@mia.uni-saarland.de

Abstract. We study an advanced method for supervised multi-label image segmentation. To this end, we adopt a classic framework which recently has been revitalised by Rhemann et al. (2011). Instead of the usual global energy minimisation step, it relies on a mere evaluation of a cost function for every solution label, which is followed by a spatial smoothing step of these costs. While Rhemann et al. concentrate on efficiency, the goal of this paper is to equip the general framework with sophisticated subcomponents in order to develop a high-quality method for multi-label image segmentation: First, we present a substantially improved cost computation scheme which incorporates texture descriptors, as well as an automatic feature selection strategy. This leads to a high-dimensional feature space, from which we extract the label costs using a support vector machine. Second, we present a novel anisotropic diffusion scheme for the filtering step. In this PDE-based process, the smoothing of the cost volume is steered along the structures of the previously computed feature space. Experiments on widely used image databases show that our scheme produces segmentations of clearly superior quality.

1 Introduction

Segmentation is one of the classical problems in image analysis. For the last four decades, researchers have been developing a wide variety of different approaches to this problem. In this paper, we consider a special instance of the segmentation problem, so-called *supervised* segmentation. In this setting, we assume for every class to be given an exemplary and reliable region in the image. In the field of supervised segmentation, energy-based methods are most common today, where the sought segmentation is the minimiser of a suitable cost function. Such a function usually consists of at least two terms: A fidelity term, which relates the unknown to the image, and a regularity term that implements prior knowledge about the solution. The computation of a minimiser often renders itself very difficult: First, in most cases the fidelity term cannot be solved directly for the unknown which makes linearisation or relaxation steps necessary. Second, the regularisation term usually couples the solution globally and by that makes a pointwise minimisation impossible.

A framework which avoids the aforementioned difficulties was first introduced by Scharstein and Szeliski [1] and recently revisited by Rhemann et al. [2]. This so-called *Cost Volume Filtering* (CVF) framework describes a very versatile and general three-step procedure to find a good, spatially smooth configuration in a discrete solution space. In contrast to energy minimisation methods, the only requirement is an evaluable cost function. However, for certain other computer vision problems such as optic flow, the discreteness of the solution space constitutes a problem. Nevertheless, the framework is perfectly suited for the multi-class image segmentation problem. With this paper, we propose an advanced method for this task using the CVF framework.

Our Contribution. Although the general applicability of CVF to (binary) image segmentation has been shown in [2], this paper presents substantial improvements to this concept, yielding state-of-the-art results while sticking to the general framework. In detail, our contributions are twofold:

1. We improve the elementary cost computation in [2] by incorporating colour and texture information, leading to a high-dimensional feature space. In order to make the final cost calculation feasible and efficient, we propose an elaborate feature selection scheme which selects the most relevant features automatically. In this comparatively low-dimensional input space, we finally train a support vector machine with Gaussian kernels to compute the cost value.
2. Regarding the filtering stage, we propose the usage of a novel anisotropic diffusion filter which is steered by the feature space. Moreover, we embed the anisotropic smoothing into an interpolation scheme as known in the context of PDE-based image inpainting [3–5] and compression [6].

Right from the beginning, we want to stress that the main goal of our work is highest segmentation quality instead of lowest computation time, where [2] is focussed on. Nevertheless, also our method can be accelerated drastically by porting it to the GPU if necessary.

Related Work. The field of energy-based segmentation methods can be systematically split into discrete and continuous methods. Among the latter, the seminal model by Mumford and Shah [7] marked the starting point of many successful segmentation methods, e.g. [8, 9]. In [10], Brox et al. have shown in an unsupervised setting that incorporating motion information can improve the segmentation performance. Concerning discrete segmentation methods, graph cut methods [11–13] have become very popular in the last decade. In [14], Rother et al. demonstrate an interactive segmentation method and introduce a large image database with ground truth labelling data, which serves as a commonly used benchmark today. The work by Lellmann et al. [15] is situated in a similar setting as ours but focusses on a relaxation of the cost term, and is supplemented by a regularity term. Martin et al. also utilise color and texture descriptors [16] for boundary detection. Concerning the filtering step, Scharstein and Szeliski [1] propose so-called non-uniform diffusion to filter the costs. Yoon and Kweon [17] employ locally adaptive support weights which show close relationship to the bilateral filter [18]. In [2], Rhemann uses the guided image filter [19] as an approximation of the joint bilateral filter [20]. However, we advocate anisotropic PDE-based diffusion processes [21] to smooth the cost slices. Different strategies for PDE-based interpolation are studied in [3, 4, 6].

Paper Organisation. In the following section, we give an introductory explanation of the cost volume filtering framework. In Section 3 we give detailed explanations of our contributions: After discussing the cost computation scheme in Section 3.1, we subsequently introduce an anisotropic and supervised cost filtering strategy in Section 3.2. Our experiments in Section 4 show that the proposed segmentation method performs well in practice and competes with the state-of-the-art. We conclude the paper in Section 5.

2 Cost Volume Filtering

The *Cost Volume Filtering* (CVF) framework can be seen as a very general and versatile procedure comprising the following steps:

1. **Cost Computation.** First, the cost volume $f : \Omega \times \mathcal{L} \to [0, 1]$ is computed. In practice, a fidelity term is evaluated for every pixel of the image domain Ω and each possible label ℓ of the finite label space $\mathcal{L}$. The choice of this term depends on the application.
2. **Cost Smoothing.** In the second stage, the cost slices undergo a smoothing step. The most important property of this filtering step is that there is no direct coupling between the different label slices, i.e. a 2-D filter is applied to each cost slice separately. However, usually the smoothing is guided by the underlying image, which reflects the assumption that segment boundaries and image edges coincide.
3. **Minimisation.** The final step is to compute the pixelwise minimum of the cost volume and take this label as the result r:

$$r(x, y) = \operatorname{argmin}_{\ell \in \mathcal{L}} f_{\text{smoothed}}(x, y, \ell). \tag{1}$$

This simple stepwise structure of CVF offers several advantages and disadvantages. In principle, any cost function can be used, because there are no requirements such as differentiability, convexity, linearity or even positivity. The filtering stage also offers many degrees of freedom; almost any scalar-valued smoothing method can be used, and the smoothing steps can be easily parallelised since there is no coupling between different labels. Finally, the minimisation is an efficient pointwise $\mathcal{O}(|\mathcal{L}|)$ operation.

On the other hand, unfortunately no energy function is known for CVF up to date, so almost no theoretical properties can be proven. Moreover, especially for continuous problems such as optic flow, the need for a finite solution space

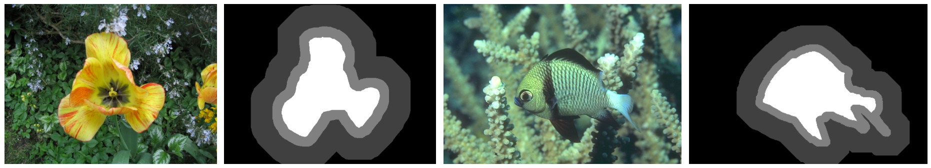

Fig. 1. Example images. While the flower can be well described by its colour statistics, obviously the fish exhibits very similar colours as the background. The second and last images show the associated given trimaps. Source: [14].

constitutes a severe restriction. However, the image segmentation problem is inherently finite and thus ideally suited for CVF.

3 Extensions for Supervised Segmentation

In the following sections, we consider the task of partitioning a colour image such that each component of the partition is assigned to one of $n \geq 2$ classes. Moreover, we are in a supervised segmentation setting, i.e. for each label $\ell \in \mathcal{L} = \{1, \ldots, n\}$ we are given an image region $\mathcal{T}_\ell \subset \Omega$ which definitely belongs to this class label. Figure 1 shows two exemplary images along with given user input. In these so-called trimaps, white and dark grey represent the user input for foreground and background, respectively. The black regions are not considered. Hence, the task is to classify each of the light grey pixels in the boundary region between fore- and background.

3.1 Cost Computation

The application of CVF to image segmentation is already addressed in [2], where the authors propose to compute the costs using colour statistics of the RGB channels. Apparently, this is the first solution which comes to mind. However, it exhibits several drawbacks in practice: The choice of the red, green and blue values as feature is redundant and not invariant against e.g. shadows or shading. In many cases, colour information is not a sufficiently relevant feature; see e.g. Figure 1. Moreover, the computation of the trivariate RGB histogram poses another problem: in order to cover all bins of the histogram fairly, a large number of samples from the training region is needed. In an interactive segmentation setting however, it can happen that size of the user input is very small. Hence, an overfit to the colour statistics in the training region can be the result.

Our method circumvents these problems in the following way: In a first step, we compute a large pool of features. Subsequently, we identify the most relevant ones using filter and wrapper methods [22]. Finally, the actual costs are determined by a support vector machine (SVM) with Gaussian kernels. The following paragraphs discuss these three steps in detail.

Available Features. We propose a pool of features which comprises a variety of colour and texture descriptors. Besides the red, green and blue channel, we also consider the hue, saturation, and value (HSV) representation. In the category of differential features, we compute the Gaussian smoothed gradient magnitude, first order derivatives in horizontal and vertical direction as well as the Laplacian magnitude averaged over 3 colour channels. Additionally, we incorporate the variance, skewness, kurtosis and entropy of the local histogram of disk shaped neighbourhood of different radii. Such multi-scale descriptors have also shown their usefulness in [23, 24]. Additionally, we compute co-occurrence matrices for 16 different offset vectors in every pixel and include the quantities contrast and homogeneity [25] in our feature vector. In total, the mentioned concepts amount to a descriptor space of dimension 80.

Feature Selection. Although computationally very expensive, it would be possible to train an SVM directly in this high-dimensional feature space. However, usually only a few features are relevant for one particular image. Thus, also in terms of the classification performance it is a bad choice to always incorporate all features in the SVM. The goal of this paragraph is to discuss how we select the most relevant features in order to get a discriminative and at the same time low-dimensional feature space that our SVM will be trained in.

To be able to estimate the relevance of a certain set of features, we randomly divide the user input into a training and a validation set. This allows the application of the so-called wrapper method [22], which learns the SVM using the training set and assesses its classification performance on the validation set. This heuristic allows us estimate the relevance of a set of features just from the user input. Since the application of this strategy to all elements of the power set of the pool of features is computationally intractable, we consider colour and texture separately. The red, green and blue channels have the highest spatial resolution, hence we avoid to discard these in practice. The remaining colour features are selected by applying the wrapper method.

To find the most relevant texture features efficiently, we first compute the Fisher score [26] of every feature. Next, we filter the top 5 features and apply the wrapper method once more. A schematic overview of our strategy is depicted in Figure 2.

Cost Evaluation. After having selected the relevant features, we train a support vector machine (SVM) [27] with Gaussian kernels in a regression setting [28] in order to compute the costs. The final training can of course be performed incorporating all user input; a validation set is not necessary anymore. However, we can speed up the training phase by just randomly selecting 50% of the pixels without a significant impairment of accuracy. As for feature selection, we randomly split the input into training and validation sets to test different kernel and soft margin parameters in a grid search fashion [28].

A similar cost computation scheme also using colour and texture features can be found in [29]. However, the authors only considered local patch-based statis-

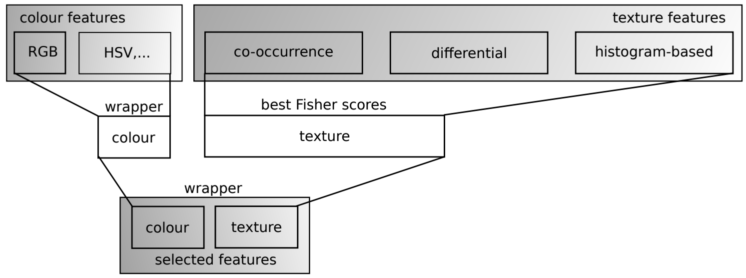

Fig. 2. Proposed feature selection strategy.

tics as features and omitted the important intermediate feature selection step. In [29], the resulting segmentation was computed in a graph cut optimisation scheme with a spatial regularity prior.

The extension to the multilabel case $|\mathcal{L}| > 2$ is straightforward: For each label ℓ, we perform the entire feature selection procedure and train the SVM in a one-vs-all strategy, i.e. during the computation for label ℓ, the user input of all other labels is included to describe the negative class.

3.2 Anisotropic Diffusion Filtering

Once the cost volume has been computed, the properties and the behaviour of the filter are crucial for the final segmentation result. This filtering clearly stands in a close relationship to the regularisation term of energy-based methods: By choosing one particular regularisation term, the resulting smoothing process is determined automatically via the associated minimality conditions of the energy. Thus, for purely energy-based methods the choice of filtering processes is restricted. For CVF however, the cost smoothing filter can be chosen directly and without any restrictions.

In [2], the authors advertise the guided image filter [19] and rely on its efficiency and parallelisability. We instead propose to use *cross edge enhancing diffusion* (Cross-EED) as an extension of *edge enhancing diffusion* (EED) [21]. Both processes perform anisotropic diffusion and are described by the parabolic evolution equation

$$\partial_t u_\ell = \operatorname{div}\left(\boldsymbol{D}\, \boldsymbol{\nabla} u_\ell\right), \quad u_\ell(x,y,0) = f(x,y,\ell), \tag{2}$$

where $\boldsymbol{D} \in \mathbb{R}^{2\times 2}$ is the symmetric positive definite diffusion tensor, and $\boldsymbol{\nabla} = (\partial_x, \partial_y)^\top$ denotes the spatial gradient operator. Each slice of the computed cost volume $f(x,y,\ell)$ is embedded into a pseudo-temporal evolution $u_\ell(x,y,t)$ as its initial state at time $t = 0$. With progressing evolution time the amount of smoothing increases, and the resulting filtered cost slice is finally extracted at the stopping time t_{stop}:

$$f_{\text{smoothed}}(x,y,\ell) = u_\ell(x,y,t_{\text{stop}}). \tag{3}$$

Alignment in Feature Space. The difference between EED and Cross-EED is how the diffusion tensor is computed. For EED, the diffusion tensor is derived from the evolving signal itself. In case of cost volume filtering, this signal would coincide with the cost slices. More precisely, the eigenvalues of the tensor product of the pre-smoothed signal gradient are reweighed:

$$\boldsymbol{D}(\boldsymbol{\nabla} u) := g(\boldsymbol{\nabla} u_\sigma \boldsymbol{\nabla} u_\sigma^\top), \tag{4}$$

where $\boldsymbol{\nabla} u_\sigma := K_\sigma * \boldsymbol{\nabla} u$, and K_σ denotes a two-dimensional Gaussian kernel with standard deviation σ. The anisotropic behaviour of this scheme is due to the positive, strictly monotonically decreasing function $g : \mathbb{R} \to \mathbb{R}^+$ that is applied to the eigenvalues of its argument. With EED, the signal is subject to a highly nonlinear evolution, which is known to smooth along edges in the cost volume but not across them [21].

The motivation for Cross-EED is that if a feature is considered to be relevant for the cost computation, then the spatial structures of this feature should also contain relevant information to steer the smoothing process. Thus, we propose to compute the diffusion tensor in the previously determined feature space. Assuming that the feature selection stage finally selected k features, we align the diffusion along the spatial structures in this feature descriptor $\boldsymbol{h} : \Omega \to \mathbb{R}^k$ as follows

$$\boldsymbol{D} := g\left(K_\rho * \sum_{j=1}^{k} \boldsymbol{\nabla} h_{j,\sigma} \boldsymbol{\nabla} h_{j,\sigma}^\top\right). \tag{5}$$

Additionally, we introduce an outer integration scale ρ in the latter equation, which leads to a coherence enhancing effect. This effect is known to tend to artistically and artificially-looking results for natural images. However, for cost filtering it has shown to be quite beneficial, due to its ability to fill holes and close small gaps in the cost slices [21].

Note that the resulting anisotropic evolution is linear, since the diffusion tensor is constant in time t and does not depend on the evolving cost volume.

Supervised Smoothing. Up to now, the proposed cost smoothing takes as input the computed costs as well as the selected features, but disregards the information contained in the user input. Assuming that this auxiliary information is reliable, we alter the PDE from (2) into a scheme which interpolates the given data as follows [5]: Each slice of the cost volume undergoes an evolution where the user input serves as Dirichlet data and is kept fixed. In the other areas where no pre-segmentation is given, the computed costs serve as initialisation and are subject to the Cross-EED smoothing operator. This behaviour is realised by the PDE

$$\begin{aligned} \partial_t u_\ell &= m \cdot (c_\ell - u_\ell) + (1-m) \cdot \operatorname{div}\left(\boldsymbol{D}\, \boldsymbol{\nabla} u_\ell\right), \\ u_\ell(x,y,0) &= m \cdot c_\ell + (1-m) \cdot f(x,y,\ell). \end{aligned} \tag{6}$$

Let us explain the expressions m and c_ℓ. The mask function $m : \Omega \to [0,1]$ switches between the diffusion process ($m = 0$) and the given information ($m = 1$).

It is realised as the indicator function $\mathbb{1}$ of the union of all training regions

$$m = \mathbb{1}_{\mathcal{T}}, \qquad \mathcal{T} := \mathcal{T}_1 \cup \ldots \cup \mathcal{T}_n. \tag{7}$$

The costs we prescribe in these areas are denoted by $c_\ell : \Omega \to \{0,1\}$ and are minimal ($c_\ell = 0$) for pixels that belong to the training data and have the same label, and maximal ($c_\ell = 1$) for pixels belonging to a different label:

$$c_\ell = 1 - \mathbb{1}_{\mathcal{T}_\ell}. \tag{8}$$

This scheme is closely related to so-called inpainting schemes, which are known from the context of PDE-based image compression [6]. Note however, we are not computing the steady state of equation (6). Instead, our evolution is initialised with the computed costs and stopped at a certain time t_{stop}.

3.3 Iterative Application

Of course, it is possible to iterate the described framework. In particular, such an iterative minimisation constitutes an essential component of the GrabCut method [14]. In our setting, we update the user input regions $\mathcal{T}_\ell$ after each filtering stage. To this end, for every label ℓ we compute the 10% and 90% cost quantiles $q_\ell^{0.1}$ and $q_\ell^{0.9}$, respectively. Then each unclassified pixel is assigned to the user input of class ℓ, if its filtered costs satisfy:

$$f_{\text{smoothed}}(x, y, \ell) < q_\ell^{0.1} \qquad \text{and} \qquad \forall_{\ell' \neq \ell} : \quad f_{\text{smoothed}}(x, y, \ell') > q_{\ell'}^{0.9}.$$

Thus, we consider a pixel reliable, if it has low costs for one label, and high costs for all other labels, and update $\mathcal{T}_\ell$ accordingly. Subsequently, we retrain the SVM using this new input and obtain refined costs. The following supervised smoothing step profits of the update in two aspects: Besides the new costs to be filtered, also the improved presegmentation is exploited for the mask m as well as the prescribed costs c_ℓ, cf. Equations (7) and (8).

4 Experiments

In order to show the performance of our method, we use the publicly available segmentation benchmark of Rother et al. [14]. Although the stages of our method introduce several free parameters, most of them can either be determined automatically or kept fixed for all images. Moreover, we apply an affine rescaling of the costs and features to the range $[0, 1]$ before filtering, which also eases the parameter choice. For the cost filtering stage, we choose the Charbonnier diffusivity function $g(s^2) = 1/\sqrt{1 + s^2/\lambda^2}$ and the constant set of parameters $(\sigma, t_{\text{stop}}, \rho, \lambda) = (0.5, 5000, 0.5, 0.01)$ for all images.

Most of the running time of our sequential CPU implementation is spent in the multiple training phases of the SVM. Depending on the class overlap in feature space, the overall computation time varies between 10 seconds and a few

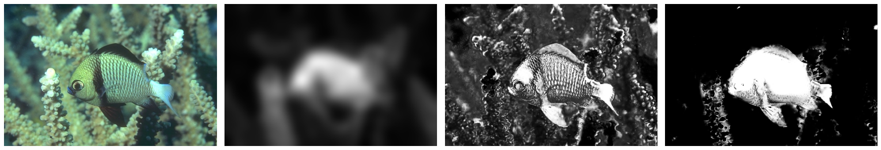

Fig. 3. Importance of texture features for cost computation. **From left to right:** **(a)** Input RGB image. **(b)** Gaussian smoothed Laplacian magnitude (σ=12). **(c)** Cost slice for background only using RGB features. Dark values indicate low costs. **(d)** Costs incorporating the feature from (b).

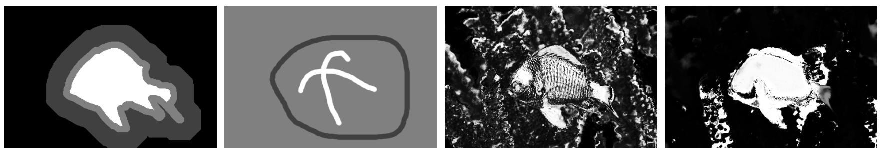

Fig. 4. Impact of reducing the user input. **From left to right:** **(a)** GrabCut input used to compute the costs in Fig. 3 (c) and (d). **(b)** Manually drawn smaller user input. **(c)** Cost slice using input (b) and only RGB features. **(d)** Costs computed from input (b) with texture feature.

minutes. On average, our method takes one minute per image of size 480×320 on a standard PC. Typically, a GPU implementation should be 40 times faster since all components are well paralellisable.

In our first experiment, we illustrate the usefulness of texture information. Figure 3 shows the image of a fish, which has almost the same colour distribution as its surrounding. In RGB space, the fish cannot be discriminated from the background (cf. Fig. 3(c)). On the contrary, by incorporating texture information, the fish can be distinguished: The raw number of misclassified pixels after cost computation drops from 19% to 6% just by incorporating texture features. In Figure 4, we examine the dependency of our method on the size of the user input. While the given user input of the GrabCut benchmark covers relatively large portions of the objects, we use a very sparse self-made trimap for comparison. The costs in Figure 4 (c) and (d) show that our method works almost as good with such more realistic user inputs and does not rely on a large number of feature points. In this special case of very small user input, the usage of texture information has shown to be extraordinarily beneficial.

The second experiment compares the proposed PDE-based smoothing against the filters in literature. The results in Figure 5 show that our anisotropic cost volume filtering clearly outperforms the guided image filter. Its anisotropic behaviour, especially in combination with the coherence enhancing effect, is perfectly suited to preserve small important details such as the feet, tail and wings, which cannot be preserved using the other filters.

In Figure 6, we illustrate how iterating the framework improves the segmentation using the difficult kangaroo image. The given input regions expand towards

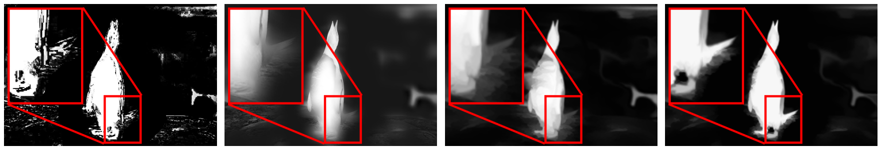

Fig. 5. Results of different smoothing filters. **From left to right: (a)** Input background cost slice for the penguin image. **(b)** Guided image filter. **(c)** Cross-EED without user input interpolation. **(d)** Cross-EED with interpolation.

Fig. 6. Iterating the framework. **From left to right: (a)** Input kangaroo image. **(b)** Initial segmentation (error 7.2%). **(c)** User input. White depicts original user input, red visualises the updated input after the first iteration. **(d)** Final segmentation after second iteration (error 3.9%).

the true object boundaries and the resulting segmentation includes previously undetectable parts of the kangaroo. Within 3 iterations, the average error on the whole GrabCut benchmark decreases from 4.7% to 4.0%.

In our fourth experiment, we segment an image into three classes (Fig. 7). In this example, the bright spots in the background make it impossible to discriminate the leopard without texture information. However, our feature selection strategy indentifies a suitable texture feature, and a highly accurate segmentation is possible.

Finally, Table 1 quantifies the performance of our segmentation method. To this end, we compute the average percentage of misclassified pixels in the unclassified region [13] over the whole set of 50 benchmark images from [14]. Note that we use a fixed set of parameters for all images. Compared to the method of Rhemann et al., our iterated approach reduces the error by 35%.

5 Conclusion

We improve the segmentation framework of Rhemann et al. [2] in several aspects. First, we incorporate texture information and present an elaborate cost computation and feature selection scheme. Additionally, we propose an anisotropic and

Table 1. Quantitative error comparison.

	CVF [2]	Grabcut [14]	Ours just RGB	Ours with texture	Ours iterated
Error unfiltered	-	-	10.0 %	10.2 %	6.8 %
Average Error	6.2 %	5.3 %	4.8 %	4.7 %	4.0 %

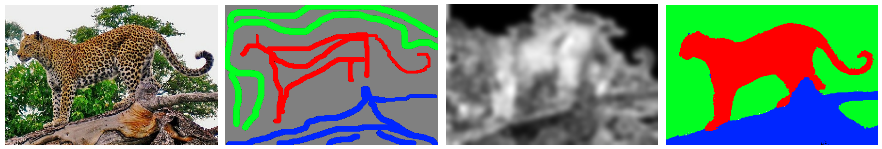

Fig. 7. Multi class segmentation example. **From left to right: (a)** Input image. **(b)** User input. **(c)** Automatically selected texture feature. **(d)** Final segmentation.

supervised cost smoothing scheme that fully exploits the given user input. This smoothing process is steered by structures in the feature space and alignes the segmentation with them. By iterating the framework, we are able to outperform the state-of-the-art on the GrabCut benchmark.

In our ongoing research, we focus on improving the texture descriptors further. Besides Gabor or wavelet features, we are also interested in the potential of preprocessing these features with e.g. our anisotropic diffusion.

Acknowledgements. Funding by the Cluster of Excellence *Multimodal Computing and Interaction* is gratefully acknowledged.

References

1. Scharstein, D., Szeliski, R.: Stereo matching with non-linear diffusion. In: Proc. Conf. on Computer Vision and Pattern Recognition, IEEE (1996) 343–350
2. Rhemann, C., Hosni, A., Bleyer, M., Rother, C., Gelautz, M.: Fast cost-volume filtering for visual correspondence and beyond. In: Proc. Conference on Computer Vision and Pattern Recognition, IEEE (2011) 3017–3024
3. Masnou, S., Morel, J.M.: Level lines based disocclusion. In: Proc. International Conference on Image Processing. Volume 3. (1998) 259–263
4. Bertalmío, M., Sapiro, G., Caselles, V., Ballester, C.: Image inpainting. In: Proc. SIGGRAPH, ACM (2000) 417–424
5. Weickert, J., Welk, M.: Tensor field interpolation with PDEs. In Weickert, J., Hagen, H., eds.: Visualization and Processing of Tensor Fields. Springer (2006) 315–325
6. Schmaltz, C., Weickert, J., Bruhn, A.: Beating the quality of JPEG 2000 with anisotropic diffusion. In Denzler, J., Notni, G., Süße, H., eds.: Pattern Recognition. Volume 5748 of Lecture Notes in Computer Science. Springer (2009) 452–461
7. Mumford, D., Shah, J.: Boundary detection by minimizing functionals, I. In: Proc. Conference Computer Vision and Pattern Recognition, IEEE (1985) 22–26
8. Koepfler, G., Lopez, C., Morel, J.M.: A multiscale algorithm for image segmentation by variational method. SIAM Journal Numerical Analysis **31** (1994) 282–299
9. Chan, T.F., Vese, L.A.: Active contours without edges. Transactions on Image Processing **10** (2001) 266–277
10. Brox, T., Rousson, M., Deriche, R., Weickert, J.: Colour, texture, and motion in level set based segmentation and tracking. Image Vision Computing **28** (2010) 376–390

11. Boykov, Y., Veksler, O., Zabih, R.: Fast approximate energy minimization via graph cuts. Transactions on Pattern Analysis and Machine Intelligence **23** (2001) 1222–1239
12. Boykov, Y., Jolly, M.P.: Interactive graph cuts for optimal boundary and region segmentation of objects in n 1-D images. In: Proc. International Conference on Computer Vision, IEEE (2001) 105–112
13. Blake, A., Rother, C., Brown, M., Pérez, P., Torr, P.H.S.: Interactive image segmentation using an adaptive GMMRF model. In Pajdla, T., Matas, J., eds.: Proc. European Conference on Computer Vision, Springer (2004) 428–441
14. Rother, C., Kolmogorov, V., Blake, A.: "GrabCut": interactive foreground extraction using iterated graph cuts. In: Proc. SIGGRAPH, ACM (2004) 309–314
15. Lellmann, J., Becker, F., Schnörr, C.: Convex optimization for multi-class image labeling with a novel family of total variation based regularizers. In: Proc. Tenth International Conference on Computer Vision, IEEE (2009) 646–653
16. Martin, D., Fowlkes, C., Malik, J.: Learning to detect natural image boundaries using local brightness, color, and texture cues. Transactions on Pattern Analysis and Machine Intelligence **26** (2004) 530–549
17. Yoon, K.J., Kweon, I.S.: Adaptive support-weight approach for correspondence search. Trans. on Pattern Analysis and Machine Intelligence **28** (2006) 650–656
18. Tomasi, C., Manduchi, R.: Bilateral filtering for gray and color images. In: Proc. International Conference on Computer Vision. (1998) 839–846
19. He, K., Sun, J., Tang, X.: Guided image filtering. In Daniilidis, K., Maragos, P., Paragios, N., eds.: Proc. European Conference on Computer Vision. Volume 6311 of Lecture Notes in Computer Science., Springer (2010) 1–14
20. Paris, S., Durand, F.: A fast approximation of the bilateral filter using a signal processing approach. In: Proc. European Conference on Computer Vision. Volume 4 of Lecture Notes in Computer Science., Springer (2006) 568–580
21. Weickert, J.: Anisotropic Diffusion in Image Processing. Teubner, Stuttgart (1998)
22. Kohavi, R., John, G.H.: Wrappers for feature subset selection. Artificial Intelligence **97** (1997) 273–324
23. Lowe, D.L.: Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision **60** (2004) 91–110
24. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In Schmid, C., Soatto, S., Tomasi, C., eds.: Proc. Conference on Computer Vision and Pattern Recognition. Volume 2. (2005) 886–893
25. Haralick, R.M., Shanmugam, K., Dinstein, I.: Textural features for image classification. Transactions of the Systems, Man and Cybernetics (1973) 610–621
26. Duda, R.O., Stork, D.G., Hart, P.E.: Pattern Classification. 2nd edn. Wiley (2000)
27. Vapnik, V.: The nature of statistical learning theory. Statistics for engineering and information science. Springer (2000)
28. Schölkopf, B., Smola, A.: Learning with Kernels. MIT Press, Cambridge (2002)
29. Duchenne, O., Audibert, J.Y.: Fast interactive segmentation using color and textural information. Technical Report 06-26, CERTIS, ParisTech (2006)