Color image segmentation using PDE-based regularization and watersheds

Erdem Yörük and Ceyhun B. Akgül

***Abstract*—In this paper, we propose an extension of watershed segmentation to color images. PDE-based regularization is considered prior to the extraction of watershed contours in order to avoid the inherent oversegmentation problem. Several PDEs are implemented and tested in terms of their efficiency as a preprocessing block of watershed segmentation. We have also implemented Vincent and Soille's fast watersheds algorithm in concatenation with region merging to get a true segmentation of color images. The results prove to be satisfactory.**

***Index Terms*—Color image segmentation, PDE-based regularization, Watershed transformation, Region merging.**

I. INTRODUCTION

Image segmentation is a fundamental preprocessing step in image analysis and classification. Applications vary from medical image analysis to content-based image retrieval systems. The task is basically to discriminate the regions in an image in such a way that each region is homogeneous in itself and distinct from the remainder. Many techniques are present in the literature, they are more or less successful in their specific domain but no one has a perfect generalization ability [4]. In this work, we present the implementation of a watershed segmentation of color images with region merging. Prior to watershed segmentation, PDE based regularization of the image is performed as preprocessing.

Watershed algorithm is a morphological technique for gray-scale images that yields an edge map without disconnected edges in contrast to gradient-based edge detectors like Sobel operator, or in Canny to some extent[1]. Once such an edge map is found, a region merging scheme can be applied to obtain a true segmentation of the image. The problem associated with watersheds is the one of oversegmentation. Due to insubstantial minima in the squared gradient-map of the image landscape, many irrelevant regions arise as the outcome of segmentation. Weickert proposed a method to fix this inconvenience by PDE-based regularization of gray scale images[1]. He uses an effective implementation of Catté's nonlinear diffusion filter by Additive Operator Splitting(AOS) [2] so that the possibly noisy image exhibits flat regions inside the edges, while the latter are preserved. Such a regularized image would constitute a better input to watershed technique with region merging.

In this work, it is aimed to perform an efficient and fast segmentation task for vector valued images. For this purpose, different approaches using PDE-based regularization and watershed segmentation have been proposed in the literature for both for scalar and vector valued inputs[1]. Basically two of them are investigated and adopted in this paper: Partial Differential Equations and Watershed algorithms as they are reviewed in [3] and [1], respectively.

Concentrating ourselves on these two methods, a compromise of both is preferred followed by a region merging procedure, despite the fact that any one of them can be implemented seperately to perform the whole segmentation task alone. PDE method which uses diffusion filters is suitable for image denoising and coherence enhancement, but in the case of total segmentation it is not the best choice if low computational load and high convergence rate is desired. On the other hand, watershed algorithm which works relatively faster, suffers from the limitation that many irrelevant minima cause an oversegmentation. So, as Weickert proposes in [1] an efficient combination of both is aimed where they are used in succesive parts of the whole process, compensating each other's shortcomings: First a PDE based regularization is implemented to obtain regular regions of segment candidates suitable for the next step where watersheds and region merging are applied to obtain the final result without the risk of oversegmentation. Here, the procedure offered by Weickert is for scalar images; as an innovation, we extend it to vector valued ones. However that would require the definition of an edge geometry common to each channel allowing us to apply a common vector PDE, since using separate scalar PDEs on each component would be useless resulting in different diffusion patterns and falsely smoothed edges when blended[3].

The paper is organized as follows: in Section II, we present the theory and practical considerations concerning PDE-based regularization of color images. In Section III, we review Vincent et al.'s approach to watershed segmentation based on immersion simulations[5]. We also give an algorithmic formulation of the procedure. This section concludes with the discussion of two different region-merging schemes that may follow watershed segmentation. In section IV, we provide the results of the proposed approach applied on both synthetic and real-life color images.

Erdem Yörük is a M. Sc.student and a research assistant at Electrical Engineering Dept., Boğaziçi University, Istanbul, Turkey.

Ceyhun B. Akgül is a M. Sc.student at Electrical Engineering Dept., Boğaziçi University, Istanbul, Turkey.

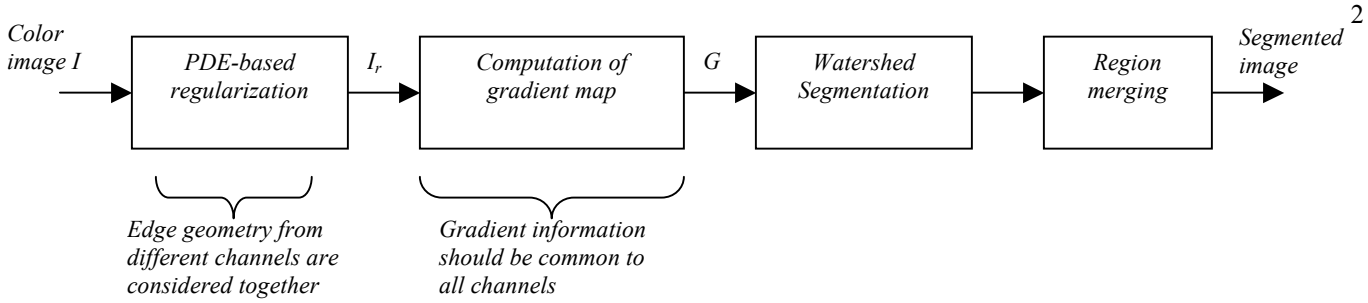

Fig. 1. Block diagram of the proposed approach.

II. PDE-BASED REGULARIZATION

A. Background

Restoration and regularization of noisy and blurred data have always been a particular area of image processing and many algorithms have been proposed to solve this problem. For this purpose, diffusion filters based on partial differential equations are quite remarkable in terms of their efficiencies to regularize images while preserving discontinuities of edges, which is however not the case in linear filtering schemes. The first work initiating the wide study on diffusion filters was the anisotropic diffusion PDE algorithm proposed by Perona and Malik, which was aimed to smooth grayvalued images while preserving edges. Then, the unification of many different anisotropic regularization PDEs acting on scalar images, was possible with the formulation of the Φ-function within a common variational framework: Consider a spatial 2D-domain denoted by Ω with Neumann boundary conditions on ∂Ω. Then a noisy scalar image I_0 can be regularized by minimizing the following Φ-functional:

$$E(I) = \int_\Omega \left[\frac{\alpha}{2}(I - I_0)^2 + \Phi(\|\nabla I\|) \right] d\Omega \qquad (1)$$

Here the fixed parameter α > 0 prevents the solution to be too different from the original image I_0, with $\Phi : \Re \to \Re$ being an increasing function and controlling the regularizaiton behavior. The minimization of E(*I*) is then performed by the following PDE evolution:

$$\frac{\partial I}{\partial t} = \alpha(I - I_0) + div\left(\frac{\Phi'(\|\nabla I\|)}{\|\nabla I\|} \nabla I \right) \qquad (2)$$

With this common framework, many scalar regularization schemes proposed in the current literature, can be expressed by finding the corresponding Φ-function. Basically, such a regularization PDE should adapt its diffusion behavior to the local geometry of the image, defined by the edge indicators and edge orientations. Including these, (2) becomes:

$$\begin{aligned} \frac{\partial I}{\partial t} &= \alpha(I - I_0) + \Phi''(\|\nabla I\|) I_{\eta\eta} + \frac{\Phi'(\|\nabla I\|)}{\|\nabla I\|} I_{\xi\xi} \\ &= \alpha(I - I_0) + c_\eta(\|\nabla I\|) I_{\eta\eta} + c_\xi(\|\nabla I\|) I_{\xi\xi} \end{aligned} \qquad (3)$$

where $I_{\eta\eta} = \boldsymbol{\eta}^T \mathbf{H} \boldsymbol{\eta}$ and $I_{\xi\xi} = \boldsymbol{\xi}^T \mathbf{H} \boldsymbol{\xi}$ are the second spatial derivatives of I in the directions of the gradient $\boldsymbol{\eta} = \nabla I / \|\nabla I\|$ and its orthogonal $\boldsymbol{\xi} = \boldsymbol{\eta}^\perp$, with **H** denoting the hessian of *I*. According to these definitions, on an image discontinuity, namely on an edge, we have a diffusion along $\boldsymbol{\eta}$ (normal to the edge) weighted with:

$$c_\eta(\|\nabla I\|) = \Phi''(\|\nabla I\|) \qquad (4)$$

and a diffusion along $\boldsymbol{\xi}$ (tangential to the edge) weighted with:

$$c_\xi(\|\nabla I\|) = \frac{\Phi'(\|\nabla I\|)}{\|\nabla I\|} \qquad (5)$$

Here different choices on Φ lead to different diffusion patterns, with the following constraints on c_η and c_ξ:

1) c_η and c_ξ must be positive to avoid inverse diffusion along $\boldsymbol{\eta}$ and $\boldsymbol{\xi}$.
2) When the local geometry is flat with no edges ($\|\nabla I\| \to 0$), the diffusion should be isotropic, with no preferred diffusion directions since $\boldsymbol{\eta}$ and $\boldsymbol{\xi}$ do not represent significant orientations in this case: $c_\eta \cong c_\xi = \beta > 0$ then $\partial I / \partial t = \beta(I_{\xi\xi} + I_{\eta\eta}) = \beta \nabla I$
3) On high gradient regions ($\|\nabla I\| >> 0$), where the current point may be located on an edge, the diffusion should be done

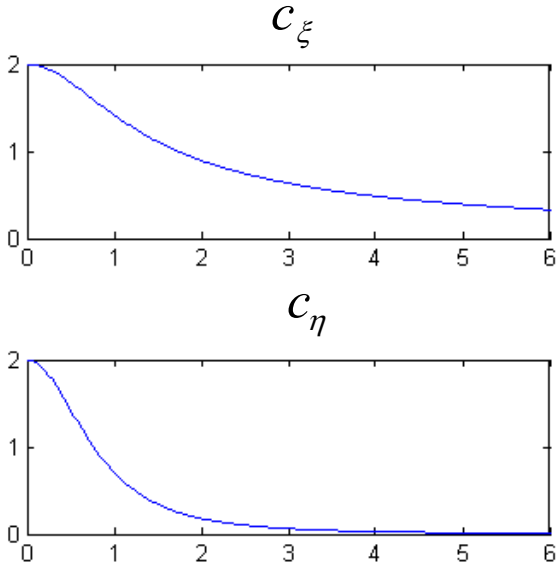

Fig. 2. The decreasing positive functions c_ξ and c_η.

only along the tangential edge direction $\boldsymbol{\xi}$, to preserve it: $c_\xi >> c_\eta$ and $c_\eta \cong 0$ then $\partial I / \partial t = c_\xi I_{\xi\xi}$.

As an example the hypersurface function defined in [10]:

$$\Phi(s) = 2\sqrt{1+s^2} - 2 \tag{6}$$

satisfies these properties and is used in some of the implementations performed in this paper and c_ξ and c_η look like as shown in Fig. 2.

B. Vector Local Geometry

As mentioned above, a regularization process should adapt its diffusion behavior to the local geometry of the image, defined by the edge indicators and edge orientations. For the scalar case, such attributes are given by $\|\nabla I\|$ and by the orientation basis ($\boldsymbol{\eta}, \boldsymbol{\xi}$), respectively. Extending this approach to vector valued images **I**, we have to define equivalent geometric attributes, taking the coupling between vector channels I_i into consideration. Applying separate scalar PDEs to each of these components would not be a remedy, since each channel would diffuse with different local geometries $\|\nabla I_i\|$ and ($\boldsymbol{\eta}_i, \boldsymbol{\xi}_i$), leading to a blended image with falsely smoothed edges.

For a vector valued image **I**, the local geometry with gadient norm *N* and direction $\boldsymbol{\theta}_+$, should be "as common as possible" to all channels. One approach would be to reduce the dimensionality of the vector image to a scalar one by evaluating the luminance. If $f(\mathbf{I}) = \sum_{i=1}^{M} I_i^2$ denotes the luminance function of a given *M*-channel image **I**, gradient norm and direction will be $N = \|\nabla f(\mathbf{I})\|$ and $\boldsymbol{\theta}_+ = \nabla f(\mathbf{I}) / \|\nabla f(\mathbf{I})\|$. However, the luminance function would not be able to detect iso-luminance contours.

A possible remedy, proposed by Di Zenzo in [1] considers a multivalued image **I** as a 2D→*M*D (*M* = 3 for color images) vector field and looks for the local variations of the norm $\|d\mathbf{I}\|^2$, mainly given by the variation matrix **G**. If we denote by $\mathbf{X}=(x,y)^T$, we get $\|d\mathbf{I}\|^2 = d\mathbf{X}^T \mathbf{G} d\mathbf{X}$, where $\mathbf{G} = \sum_{i=1}^{M} \nabla I_i \nabla I_i^T$ with ∇I_i denoting the gradient in the i[th] image channel.

For color images **I** = (*R,G,B*) the symmetric and semipositive matrix **G** is then:

$$\mathbf{G} = \begin{bmatrix} R_x^2 + G_x^2 + B_x^2 & R_x R_y + G_x G_y + B_x B_y \\ R_x R_y + G_x G_y + B_x B_y & R_y^2 + G_y^2 + B_y^2 \end{bmatrix} \tag{7}$$

The positive eigenvalues $\lambda_{+/-}$ of **G** are the maximum and the minimum of $\|d\mathbf{I}\|^2$ and the orthogonal eigenvectors $\boldsymbol{\theta}_+$ and $\boldsymbol{\theta}_-$ are the corresponding variation orientations:

$$\lambda_{+/-} = \frac{g_{11} + g_{22} \pm \sqrt{\Delta}}{2} \tag{8}$$

and

$$\boldsymbol{\theta}_{+/-} // \begin{bmatrix} 2g_{12} \\ g_{22} - g_{11} \pm \sqrt{\Delta} \end{bmatrix} \tag{9}$$

where $\Delta = (g_{11} - g_{22})^2 + 4g_{12}^2$. Considering this Di Zenzo metric an appropriate choice for the vector image-gradient, the gradient norm will be:

$$N = \sqrt{\lambda_+ + \lambda_-} = \sqrt{trace(\mathbf{G})} = \\ = \sqrt{\sum_{i=1}^{M} \|\nabla I_i\|^2} = \sqrt{R_x^2 + G_x^2 + B_x^2 + R_y^2 + G_y^2 + B_y^2} \tag{10}$$

Here, Di Zenzo's approach considers the gradient as a random vector with its realizations in each channel and computes its covariance matrix **G**. Then, the larger eigenvalue λ_+ of **G** will correspond to its eigenvector with maximum variation, $\boldsymbol{\theta}_+$, which is assumed to be vector-image gradient, common to each channel. With the same reasoning, the second eigenvector $\boldsymbol{\theta}_-$ will be taken orthogonal to this gradient direction.

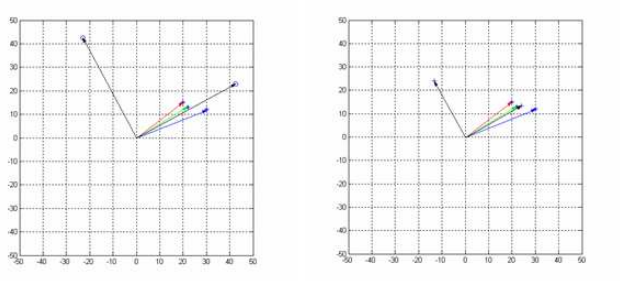

Fig 3. Vector-image gradient directions and gradient orthogonals found with Di Zenzo metric(left) and simple vector addition method (right; red, green and blue arrows represent the corresponding channel gradients)

An alternative and simpler definition for vector-gradient direction would be:

$$\boldsymbol{\theta}_+ \,//\, \sum_{i=1}^{M} \nabla I_i = \begin{bmatrix} R_x + G_x + B_x \\ R_y + G_y + B_y \end{bmatrix}, \; \boldsymbol{\theta}_- = \boldsymbol{\theta}_+^{\perp} \tag{11}$$

where it is found simply by vector addition of channel gradients. Along with Di Zenzo's, this metric is also used in this paper. Both choices detect vector valued edges and corners in a good way, and they are easy to compute. For a given edge region where each channel exhibits slightly different gradient direction the overall gradient directions according to both metric will be as illustrated in Fig. 3.

C. Vector PDE

Along with the ones rewieved in [3], we implemented various diffusion filters, comparing their results in Section IV. In all of them, N and $\boldsymbol{\theta}_+$ (with $\boldsymbol{\theta}_- = \boldsymbol{\theta}_+^{\perp}$) denote the vector-image gradient norm and direction, respectively, found by the vector local geometries defined above. In the first two of the following the diffusion equation is of the form:

$$\frac{\partial I_i}{\partial t} = div(g(N)\nabla I_i) \tag{12}$$

with g(N) some diffusivity function which is decreasing in N. The subsequent three constitute tensor diffusion PDEs of the general form:

$$\frac{\partial I_i}{\partial t} = div(\mathbf{D}\nabla I_i) \tag{13}$$

with a specific symmetric and diagonalizable diffusion tensor **D**, where its eigenvalues are the diffusion weights along corresponding eigenvectors. Note that in both of them g(N) and **D** contain local geometry information common to all channels, whereas gradient ∇I_i and time derivative $\partial I_i / \partial t$ are specific to channel i. The reflecting boundary conditions are also satisfied by constraining $\partial_n I_i / \partial\Omega = 0$ where **n** denotes the normal to the image boundary $\partial\Omega$, i.e. diffusion normal to the image boundary is forbidden.

1) PDE of Catté et al.: The nonlinear diffusion filter of Catté et al. with the structure given in (10), uses N_σ instead of N, where N_σ is the gradient norm defined by (8), obtained from a Gaussian smoothed version of **I**. The diffusivity function, g is defined by:

$$g(s) = \begin{cases} 1 & (s = 0), \\ 1 - \exp\left(\dfrac{-3.315}{(s/\mu)^8}\right) & (s \neq 0). \end{cases} \tag{14}$$

For such rapidly decreasing diffusivities, smoothing on both sides of an edge is much stronger than smoothing across it. This selective smoothing process prefers intraregional smoothing to interregional blurring. The factor 3.315 ensures that the flux $\Phi(s) = sg(s)$ is increasing for $|s| \le \mu$ and decreasing for $|s| > \mu$. Thus, μ is a contrast parameter seperating low contrast regions with forward diffusion from high contrast locations where backward diffusion may enhance edges.

2) PDE of Perona and Malik: The anisotropic PDE of Perona and Malik defined by (10) employs the diffusivity:

$$g(s) = \exp\left(-\frac{s^2}{k^2}\right) \tag{15}$$

which has the same effect on edges as in Catté's PDE, where k is again some contrast parameter.

3) Edge enhancing diffusion: This tensor diffusion PDE as defined in (11), uses the following diffusion tensor:

$$\mathbf{D} = \begin{bmatrix} \mathbf{u} & \mathbf{v} \end{bmatrix} \begin{bmatrix} g(N_\sigma) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{u}^T \\ \mathbf{v}^T \end{bmatrix} \tag{16}$$

with eigenvalues $\lambda_1 = g(N_\sigma)$ and $\lambda_2 = 1$, and corresponding eigenvectors $\mathbf{u} = \boldsymbol{\theta}_+ \,//\, \nabla \mathbf{I}_\sigma$ and $\mathbf{v} = \mathbf{u}^{\perp} = \boldsymbol{\theta}_-$. For diffusivity g, the function defined in (13) is taken. As in the case of Catté, N_σ is the gradient norm of the smoothed version of **I,** found by (8). With this tensor, on high gradient regions where $N_\sigma >> 0$, we have $g(N_\sigma) << 1$, resulting in a very small diffusion along **u** (normal to the edge), and a diffusion with weight 1 along **v** (tangential to the edge), thus enhancing the edges. On flat regions where $N_\sigma \to 0$, we have $g(N_\sigma) \to 1$, resulting in an isotropic diffusion, equally weighted by 1 along **u** and **v**.

4) Coherence enhancing diffusion: This tensor diffusion PDE as defined in (11), uses the following diffusion tensor:

$$\mathbf{D} = \begin{bmatrix} \mathbf{u} & \mathbf{v} \end{bmatrix} \begin{bmatrix} \beta & 0 \\ 0 & \beta + (1-\beta)\exp\left(\frac{-C}{(\lambda_+ - \lambda_-)^2}\right) \end{bmatrix} \begin{bmatrix} \mathbf{u}^T \\ \mathbf{v}^T \end{bmatrix} \quad (17)$$

with eigenvalues $\lambda_1 = \beta$ and $\lambda_2 = \beta + (1-\beta)\exp(-C/(\lambda_+ - \lambda_-)^2)$, and corresponding eigen-vectors $\mathbf{u} = \boldsymbol{\theta}_+$ and $\mathbf{v} = \mathbf{u}^\perp = \boldsymbol{\theta}_-$ as in the previous case. λ_+ and λ_- are the eigenvalues of Di Zenzo matrix **G**, defined in (6). According to this, on high gradient regions where $\lambda_+ >> \lambda_-$, we have $\exp(-C/(\lambda_+ - \lambda_-)^2) \to 1$, and hence $\beta + (1-\beta)\exp(-C/(\lambda_+ - \lambda_-)^2) \to 1$. Thus, in those regions we have small diffusion weghted by $0 < \beta < 1$ along **u** (normal to the edge), and a diffusion with weight 1 along **v** (tangential to the edge). On flat regions where $\lambda_+ \cong \lambda_-$, we have $\exp(-C/(\lambda_+ - \lambda_-)^2) \to 0$, and hence $\beta + (1-\beta)\exp(-C/(\lambda_+ - \lambda_-)^2) \to \beta$, resulting in an isotropic diffusion, equally weighted by β along **u** and **v**. Taking β small, provides small diffusion orthogonal to the edges but also suppresses the isotropic smoothing behavior in flat regions.

5) Beltrami flow: With a different approach Sochen and Kimmel found a particular case of the coherence enhancing diffusion PDE, denoted by Beltrami flow. The tensor of this PDE has the following form:

$$\mathbf{D} = \sqrt{\det(\mathbf{Id} + \mathbf{G})}(\mathbf{Id} + \mathbf{G})^{-1} \quad (18)$$

with eigenvalues $\lambda_1 = \sqrt{(1+\lambda_-)/(1+\lambda_+)}$ and $\lambda_2 = \sqrt{(1+\lambda_+)/(1+\lambda_-)}$, and corresponding eigenvectors $\mathbf{u} = \boldsymbol{\theta}_+$ and $\mathbf{v} = \mathbf{u}^\perp = \boldsymbol{\theta}_-$. **G** is Di Zenzo matrix found by (5) with its eigenvalues λ_+ and λ_-,and **Id** is a 2×2 identity matrix. In order to keep the diffusion small in regions with dense structure, the diffusion equation of (11) is weighted with an extra term $(\det(\mathbf{Id} + \mathbf{G}))^{-1/2}$ resulting in:

$$\frac{\partial I_i}{\partial t} = \frac{1}{\sqrt{\det(\mathbf{Id} + \mathbf{G})}} div(\sqrt{\det(\mathbf{Id} + \mathbf{G})}(\mathbf{Id} + \mathbf{G})^{-1} \nabla I_i) \quad (19)$$

Here, on high gradient regions where $\lambda_+ >> \lambda_-$, we have $\lambda_1 \cong 0$, hence the diffusion is mainly done along **v** (tangential to the edge). On flat regions where $\lambda_+ \cong \lambda_-$, we have $\lambda_1 \cong \lambda_2 \cong 1$, resulting in isotropic diffusion.

6) Deriche's PDE: The diffusion PDE proposed by Tschumperlé and Deriche in [3], has the general form of (3):

$$\frac{\partial I}{\partial t} = \alpha(\mathbf{I} - \mathbf{I}_0) + c_{\theta_+}(N)\mathbf{I}_{\theta_+\theta_+} + c_{\theta_-}(N)\mathbf{I}_{\theta_-\theta_-} \quad (20)$$

where c_{θ_+} and c_{θ_-} are decreasing functions of their arguments defined by (4), (5) and (6) and plotted in Fig. 3**.** This equation has the property to adapt its smoothing behavior to the local geometry of the image and it performs a coherence restoration process.

III. WATERSHED SEGMENTATION WITH REGION MERGING

A. Background

In watershed segmentation, the gradient image is viewed as a topographic relief that posses peaks and valleys. Fig. 4 is an illustration of a simple grayscale map as a topographical surface. From this perspective, a watershed algorithm can be described in view of simulating flooding water that fills up the valleys or merge in vaste plateaus. Two well known rigorous approaches to watershed segmentation are based on:

1) Rain-falling simulation
2) Immersion simulation

In this work, we adopted the immersion simulation approach which originates from Vincent and Soille's work[5]. The concept of immersion simulation can be described as follows. Holes are pierced at the minima of the surface and the whole surface is slowly immersed in water. The water rises in through these holes and gets collected in the catchment basins. When the water from one basin starts to merge into an adjacent one, a dam is built to prevent this overflow. The dams or *watershed lines* separate the catchment basins from one another and correspond to the boundaries in the image. The power of watershed segmentation resides in that it always produces closed contours. The watershed lines when properly located enclose different regions which stand for a true segmentation of the original image. In most of the cases, there are too many subtle regions that should be merged by following some application specific criteria. The PDE-based

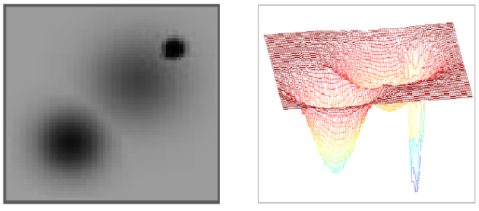

Fig. 4. A grayscale map and its topographical 3D representation.

regularization is expected to reduce the number of regions

after watershed segmentation, i.e prevent oversegmentation. Furthermore, it has been pointed out that even if a regularization is performed prior to gradient computation, the number of regions resulting from watershed segmentation is far from being acceptable[5]. Accordingly, a region merging procedure is more than necessary.

B. Vincent and Soille's Fast Watershed Algorithm

Vincent and Soille viewed the problem as a region-growing problem by starting with a possibly disconnected region of the minimum gray level, then growing this region with some morphological operations. The watershed lines happen afterwards to be the complement of the final region which is indeed the set of catchment basins. The process for finding this final region is summarized next.

Let $h_{\min}, \ldots, h_{\max}$ be the discrete gray-levels present in a gradient image I. $X_{h_{\max}}$ is the aforementioned final region and can be obtained recursively by

(i) $X_{h_{\min}} = T_{h_{\min}}(I)$ = {Pixels with gray-levels less than or equal to $h_{\min}$}
(ii) X_{h+1} = {Pixels that belong to a regional minimum at level $h+1$} $\cup$ {The geodesic influence zone of X_h inside $T_{h+1}(I)$}
(iii) Continue until the maximum discrete gray-level $h_{\max}$ is reached

A regional minimum is a connected plateau of pixels with a unique value from which it is impossible to reach a pixel of lower value without having to climb. On the other hand, the geodesic influnce zone is best explained by an illustrative figure as in Fig. 5 where there is a disconnected set **B** of three connected components such that **B**={$\mathbf{B}_i$, i=1,2,3} inside a point set **A**. Accordingly, the geodesic influence zone of $\mathbf{B}_i$ in **A** is the set of points in **A** that are closer to $\mathbf{B}_i$ than to $\mathbf{B}_j$ with j different than i. The closeness is measured with respect to geodesic distance defined as the length of the shorthest path, that is totally included in **A**, between a pixel outside a connected component $\mathbf{B}_i$ and the latter. The geodesic influence zone of **B** in **A**, in its turn, happens to be the union of the

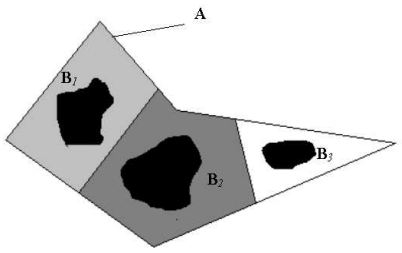

Fig. 5. The geodesic influnce zones of $\mathbf{B}_1$ to $\mathbf{B}_3$ in the set **A** are shown in shades of gray.

geodesic influnce zones of each of the connected components $\mathbf{B}_i$. The boundaries between the shaded regions of Fig. 5 thus constitute the set of points that do not belong to any of the geodesic influence zones. Now let us elaborate this illustrative example from the perspective of the procedure given above. Assume that all the pixel locations in the image I are sorted in the ascending order of their gray values and we have direct access to them. We start with $X_{h_{\min}}$ which is equivalent to the disconnected set **B** of Fig. 5. Following the same analogy, the set $T_{h_{\min}}(I)$ corresponds to **A**. Given these two regions, $X_{h_{\min}}$ and $T_{h_{\min+1}}(I)$, we try to discover the new set of catchment basins that will form $X_{h_{\min+1}}$ together with the old set of catchment basins $X_{h_{\min}}$. Obviously, all the regional minima associated with level $h_{\min+1}$ are among the catchment basins at level $h_{\min+1}$ and this constitutes the left term of the union in step (ii). Now, the remainder of $T_{h_{\min+1}}(I)$, that is different from the regional minima at level $h_{\min+1}$ and $X_{h_{\min}}$, have value $h_{\min+1}$ and belongs either to the set of catchment basins at level $h_{\min}$ or not. Hence by computing the geodesic influence zone of $X_{h_{\min}}$ inside $T_{hmin+1}(I)$, we can assign the part of the latter that are associated with the set catchment basins at level $h_{\min}$, to $X_{h_{\min+1}}$ and exclude the set of pixels that does not belong to any of the catchment basins. This set is indeed the set of watershed lines at level $h_{\min+1}$. Since we can proceed similarly for all discrete gray-level pairs h_i and h_{i+1} as suggested in step (ii), when the gray level $h_{\max}$ is reached we end up with the complement of the watershed lines, i.e. $X_{h_{\max}}$.

C. Implementation of Vincent and Soille's Algorithm

The watershed lines that are obtained using the aforementioned procedure are often made of disconnected lines. Moreover, they happen to be very thick whenever they are equally distant from the catchment basins they separate. Vincent and Soille's algorithm gets rid of these shortcomings by incorporating the labeling of each discovered regional minima and their associated pixels during the iterations. The end product of this approach is a labeled image with a possibly disconnected set of watershed pixels that are assigned to the label "zero" as a practical implementation hint. Each label corresponds to a catchment basin. Using this output image, one can either link the watershed pixels to obtain an image of contours or remove them to obtain a real tesselation of the original image into its different catchment basins. In order to link the watershed pixels, it suffices to give value "zero" to each pixel that has in its neighborhood a pixel with a smaller label. A morphological thinning is necessary afterwards in order to avoid the image be filled with watershed pixels. On the other hand, to remove the watershed lines, one should assign a watershed pixel to the label of one of its neighbors. In case of thick watershed lines where watershed pixels with no labeled neighbors occur, the procedure can be repeated until no more change is possible.

Let us now describe the labeling phase from the perspective of practical implementation. For a given gray level h, suppose that all regional minima and hence their associated catchment basins are discovered and the corresponding pixels have been given a unique label. The pixels that belong to the next level h+1 can be accessed directly due to the initial sorting of pixels.

Now, those pixels among them which have an already labeled pixel in their neighborhood are put into a queue so that the first element that is put into the queue can be first extracted. A specific pixel in the queue should be given the label of the closest catchment basin, i.e. the geodesic influence zone of which includes the pixel(this achieves the right-hand side of the union in step (ii) of the algorithm). After this, there only remain the pixels that belong to minima at level h+1 and that should be scanned to be given new labels.

The computational ease of Vincent and Soille's algorithm is due to its two aspects:

1) Initial sorting of pixels in the ascending order of their gray values so that each pixel can be directly accessed during the labeling phase (usually called the flooding step)
2) The use of *first-in-first-out* data structure in order to keep track of the pixels being tested by computing the geodesic influence zones of the existing catchment basins.

Without initial sorting of pixels, the whole image should be scanned for finding out the pixels that belong to each of the gray levels. On the other hand, in case first-in-first-out data structure is not used, the entire image should be scanned for changing just one pixel, a fact which is impractical from all perspectives. These two aspects together make the computation of watersheds of a typical 150×150 image takes approximately one second on a standard Pentium III 677 MHz PC.

One property of Vincent and Soille's algorithm is that it works with discrete gray levels. Accordingly, the values of the gradient image should be cast to integers. It is claimed that this fact may lead to inaccuracies. However, we think that this property have the effect of producing less segments, which is desirable from the perspective of oversegmentation.

B. Region Merging

PDE-based regularization creates almost piecewise constant areas, however in the presence of small variations within these areas, the watershed segmentation produces irrelevant segments that are not even sought after. In order to avoid such problems, a region merging procedure should follow watershed segmentation.

For grayscale images, the region merging is relatively simple: neighboring regions that do not differ by more than a specified contrast value can be merged into each other. However, for the case of color images as in ours, the situation is more complicated and worth explaining. We have found two different approaches useful for the purpose of region merging.

In one approach, we have borrowed from the field of agglomerative clustering. Each region is considered by a cluster characterized by the mean RGB vector of the pixels that belong to the region. That is each region has a vector attribute computed from the regularized image. Afterwards, we compute the distances between these vectors and merge two regions whose associated mean RGB vectors are closest two each other among all pair of regions. We proceed until only one region remains. The end result of these steps is a hierarchical tree or a *dendrogram* that consists of many upsidedown U shape lines connecting nodes in a hierarchical tree. The height of each U is the distance between the two regions to be connected at that time. If we cut the tree at a certain level and keep only the nodes of the tree that survive, we end up with a set of labels associated with each surviving node. The childs of such a node happen to be the regions that are to be given the label of that node. Obviously, this approach does not consider region adjacency, that is two regions that are far apart might be assigned to the same label. This subtlety can be overcome by a connected component procedure for labeled images so that disconnected regions of the same label can be assigned to different labels. Assignment of small irrelevant segments to larger regions can be carried in conjunction with this connected component procedure. This aproach has the important benefit that the number of segments does not have to be predetermined.

In the other approach, the output of watershed segmentation is first scanned to mark neighborhood relationships and to take note of region areas. Starting from the smallest region, a current region is assigned to the neighboring region that is closest in terms of the distance between mean RGB vectors. The procedure is iterated until a predetermined number of regions is reached.

We have observed that both region merging procedure operate equally well in reducing the number of segments to an acceptable value after watershed segmentation.

IV. EXPERIMENTS

We have tested the resulting method on two different types of images: synthetic color images with sharp edges and flat regions, and real color images. We have devised a variant of testing the synthetic images by adding white Gaussian noise of different variance levels into each of the channels. The performance of diffusion filtering is measured by the SNR values of the resulting PDE-regularized images, which is related to the variance of the pixels residing in flat regions. Ideally, the variance of those pixels should go to zero in the regularized version of the image since noise removal is a property of the diffusion filter. On the other hand, edge-preserving property is compared visually. For watershed segmentation, we have found useful and illustrative to show the watershed contours in the abscence and presence of PDE-based regularization. Real color images are tested in an unsupervised manner by just providing the cutting level of the tree described in Section III-D or the number of final segments for region merging phase. The results are evaluated based on semantic relevance of detected segments.

A. Experiments with Synthetic Images

As mentioned in Section II-C, eight different diffusion filters are implemented for regularization. Their performances are tested, first by computing the SNR value at each channel of the output given by:

$$SNR_i = 10\log_{10}\left(\frac{\mathrm{var}(I_i)}{\mathrm{var}(I_i - I_{pde_i})}\right) \qquad (21)$$

where I_i and I_{pde_i} denote the i^{th} channel of the original and diffused noisy image, respectively. Those values are listed in Table 1. For better comparison, each filter is observed under equal conditions where the contrast parameter k (if available), the number of iterations T and the stepsize Δt are taken 100, 20 and 0.25, respectively. It is clear that each diffusion filter exhibits certain improvement with respect to initial noise level. However, performance is significantly large for edge enhancing diffusion filters and for the anisotropic PDEs of Catté and Perona & Malik, whereas versions of coherence enhancing PDE show very little SNR improvement for 20 iterations, due to their slow convergences. Basically, for the last four, T should be increased to the order of 100 to observe significant diffusion behavior. If not mentioned as "alternative metric" which is a simple addition of channel gradients, all filters employ Di Zenzo's local geometry and results show that both metric exhibit more or less the same performance in terms of SNR. As a second measure, the output images, depicted in Fig. 6, are also inspected visually, whether their corresponding PDEs resulted in a displacement or blurring of the edges. Like their performances in terms of SNR, the edge enhancing diffusion filters and PDEs of Catté and Perona & Malik outperform the others. Blurring of the edges is especially significant in coherence enhancement filter and in Deriche's method which is an undesired result especially for the next step of watersheds. In the case of coherence enhancement & coherence restoration diffusion, such a blurring may be avoided by decreasing β, which however will reduce the isotropic diffusion behavior in flat regions, making the process more noise sensitive and resulting in a lower SNR value. In terms of their remarkable overall performance, edge enhancing filters is preferred as a regularization tool for the watershed algorithm.

Table 1. SNR values of at the output of diffusion filters

Image	SNR in R channel (dB)	SNR in G channel (dB)	SNR in B channel (dB)
Noisy image	7.51	8.13	7.21
Catté et al. $k = 100$, $T = 20$ $\Delta t = 0.25$	19.23	19.75	18.49
Perona & Malik $k = 100$, $T = 20$ $\Delta t = 0.25$	18.95	19.49	18.23
Edge enhancing $k = 100$, $T = 20$ $\Delta t = 0.25$	21.08	21.63	19.63
Edge enhancing (with alternative metric) $k = 100$, $T = 20$ $\Delta t = 0.25$	22.07	22.34	19.00
Coherence enhancing $k = 100$, $\beta = 0.1$ $T = 20$ $\Delta t = 0.25$	10.32	10.79	9.46
Beltrami flow T=20, $\Delta t = 0.25$	7.53	8.15	7.23
Deriche et al. $\alpha = 0.001$, $T = 20$ $\Delta t = 0.25$	8.48	9.09	8.16

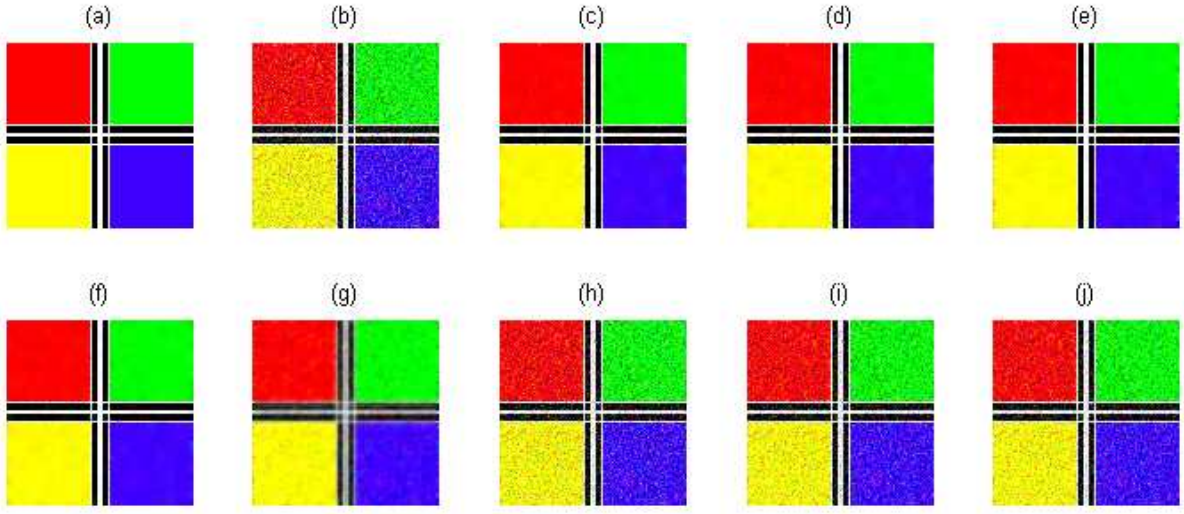

Fig. 6. (a) Original color image, (b) Noisy image, (c) Catté diffusion, (d) Perona & Malik diffusion, (e) Edge enhancing diffusion with Di Zenzo metric, (f) Edge enhancing diffusion with alternative metric, (g) Coherence enhancing diffusion, (h) Beltrami flow, (i) Deriche diffusion with Di Zenzo metric, (j) Deriche diffusion with alternative metric

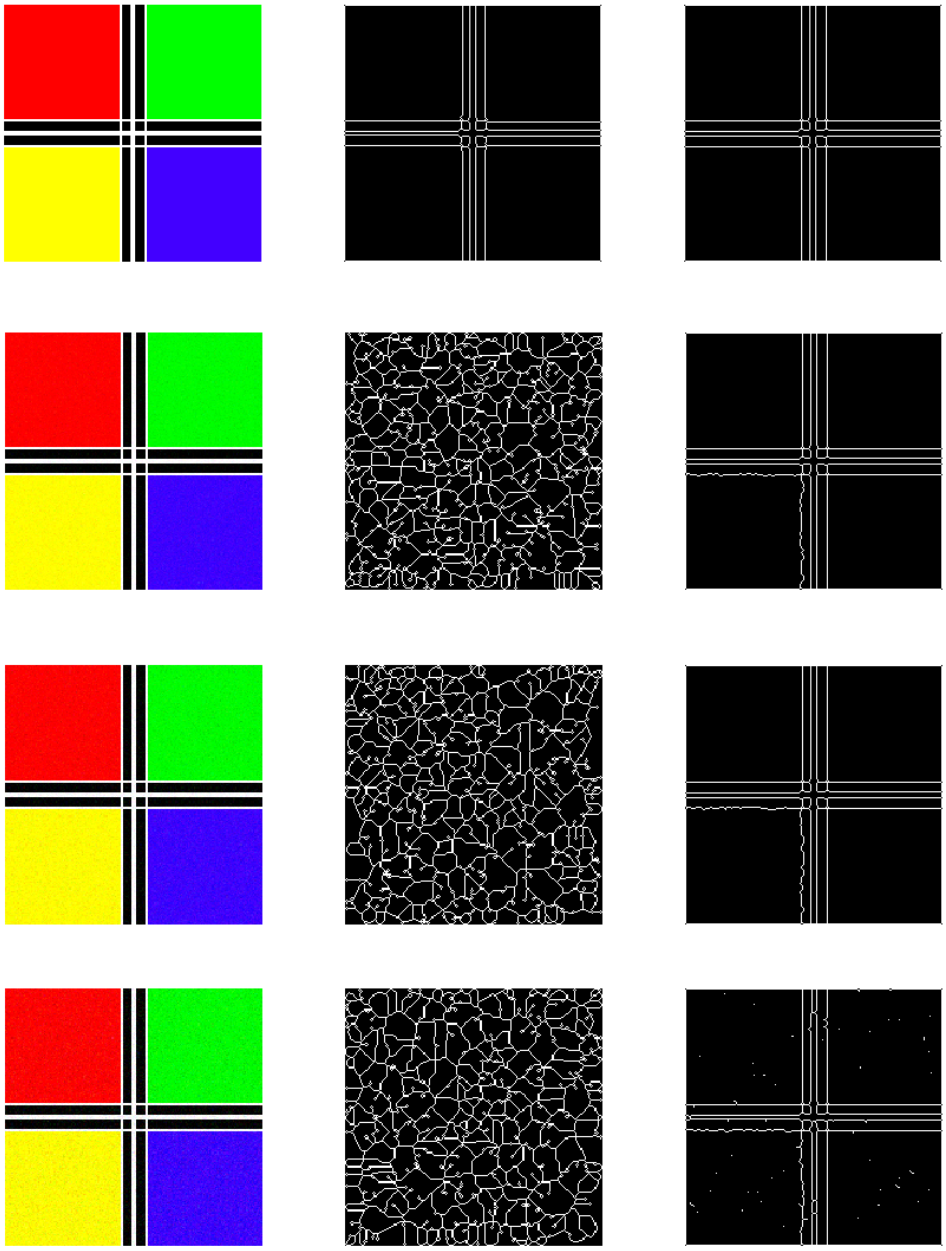

Fig. 7. A synthetic image and its noisy versions with noise variance 64, 128 and 256 (first column); watershed contours extracted without prior regularization (second column); watershed contours extracted with prior regularization (third column).

In Fig. 7, the results of watershed segmentation of a synthetic image and its noisy versions are shown. The image is designed so that there are four flat regions separated by sharp, solid lines. The noise variances are 64, 128 and 256 respectively. In the second column of Fig. 7, we see watershed segmentation results when no regularization is performed a priori, in the third column results after regularization are shown. In the case of no noise, the result is satisfactory and equivalent for both cases as expected. However, as the noise level increases, the inherent oversegmentation problem of watershed segmentation arises when no regularization is performed. There are too many irrelevant segments inside the detected contours and their semantic quality are debatable. On the other hand when a regularization is carried a priori, we end up with watershed contours incomparably better than the ones of the former case.

In a second experiment with synthetic images, we have used an image with again flat regions with curved, thin contours and investigated the accuracy of watershed segmentation in the presence of of additive Gaussian noise. In order to illustrate the effectiveness of PDE-based regularization we also provide the results of direct application of watershed segmentation in Fig. 8. The first row of Fig. 8 corresponds to no noise case. The watershed contours are superimpoed with the original image in the middle and right columns, where we show the results with no regularization and after regularization respectively. For this image too, we again experimented with noise variances 64, 128 and 256. We only show the result of the most severe case, i. e. noise variance is 256, since even in this case the watershed segmentation after regularization yields accurate contours with some occasional isolated pixels that can easily be removed by post-processing(Fig. 8 second row, right). Again when no regularization is performed, there are too many subtle segments(Fig. 8 second row, middle).

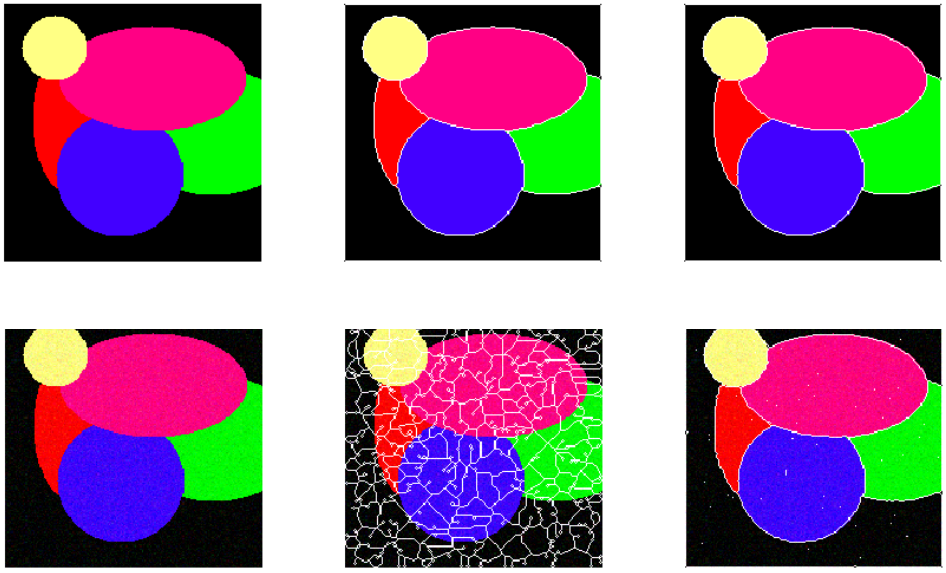

Fig. 8. A synthetic image and it noisy version with noise variance 256 (first column); watershed contours extracted without prior regularization shown superimposed with the original image (second column); watershed contours extracted with prior regularization superimposed with the original image (third column).

B. *Experiments with Real Images*

We have tested our color image segmentation algorithm on several real images. The results concerning two of them are shown in Figs. 9("Boots" image) and 10("Lady" image) step by step. For both image, we have applied PDE-based regularization based on edge-enhancing diffusion filter with contrast parameter k=30, number of iterations 10 and step size Δt=0.25. The original image, its regularized version and the common gradient map computed from the regularized image are shown in the top row of Figs. 9 and 10. Observe that high gradient areas are well located. In the second row, watershed contours at the end of each step are shown. The leftmost one is the output of watershed segmentation applied to regularized image. This image of contours is to be processed in order to have the watersheds get connected as described in Section II-C. Linked watershed lines are shown in the middle. They enclose a number of semantically irrelevant or very small segments that should be merged into each other as described in section II-D. The number of insignificant segments is more emphasized in "Boots" image than in "Lady" image. The rightmost columns of the second rows in Figs. 9 and 10, exhibit the watershed contours obtained after region merging. Finally, the images of third columns reflect the end result of our color image segmentation algorithm, segment boundaries are superimosed with the original images. For both cases, segment boundaries are fully connected and divide the image into a set of labeled regions that form a real tesselation of the original image. Furthermore, detected segments are very similar to those that can be visually extracted. This shows that all the relevant segments are kept by our segmentation algorithm.

V. Discussion and Conclusion

In this work, we have proposed an extension to a well founded segmentation technique, i.e. watershed segmentation which is originally proposed for grayscale images, in order to achieve fast and accurate segmentation of color images. Furthermore, we have explored possibilities of regularizing color images based on nonlinear diffusion PDEs using a gradient information which is common to all channels.

Resulting implementations of vector PDEs prove to be very efficient in reducing the oversegmentation problem inherent to watershed technique. Furthermore, our Matlab code processes an image of size 150×150 in approximately 2 seconds on a Pentium III 677 MHz PC for 10 iterations of diffusion. Our C implementation of watershed transformation is based on Vincent & Soille's fast watersheds and takes less than 1 second for the same type of images on the same PC. We observed that the results of watershed segmentation were satisfactorily accurate for PDE-regularized color images. Moreover, the watershed technique with efficient post-processing(linking and thinning the watershed lines) yield connected segment boundaries, hence the net effect of the overall procedure is a true tesselation of the color image under study. In addition to these steps, the use of a region merging procedure is necessary to suppress semantically irrelevant segments. Actually, the time bottleneck of our algorithm is the region merging phase implemented in Matlab. According to the number of regions that watershed segmentation produces, region merging lasts from 3 to 6 seconds or even more. In total, our algorithm can fullly automaticly, except the cut-level of the hierarchical tree at the region merging phase, segment a typical color image in approximately 10 seconds. If we think of carrying the implementation of regularization into C language, we will end up with a very fast and satisfactorily accurate color image segmentation scheme.

Fig. 9. “Boots” image, its regularized version and the gradient common to all channels (first row); watershed lines at the output of Vincent and Soille’s algorithm, watershed lines after post-processing , segment boundaries after region merging (second row); final segmentation result: segment boundaries superimposed with the original image (third row, middle).

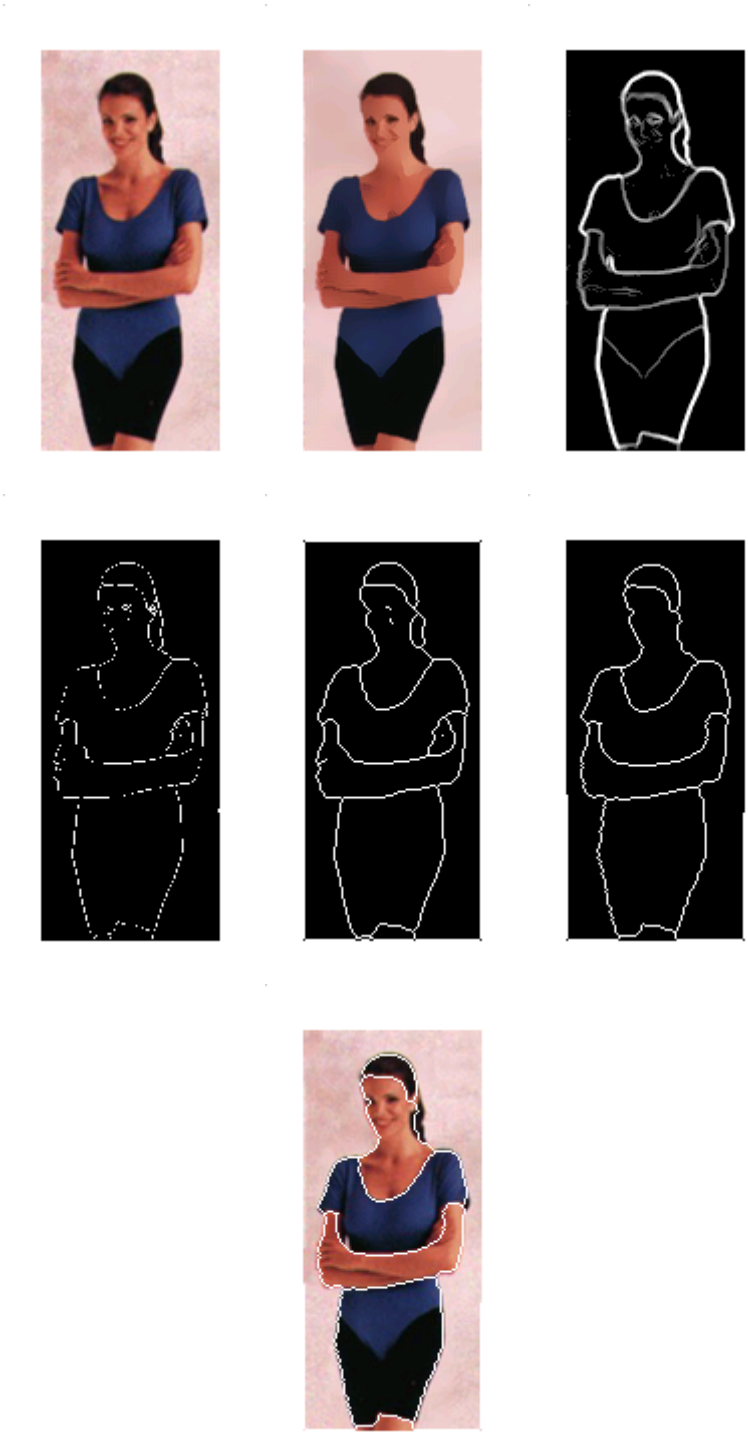

Fig. 10. “Lady” image, its regularzied version and the gradient common to all channels (first row); watershed lines at the output of Vincent and Soille’s algorithm, watershed lines after post-processing , segment boundaries after region merging (second row); final segmentation result: segment boundaries superimposed with the original image (third row, middle).

REFERENCES

[1] J. Weickert, “Efficient image segmentation using partial differential equation and morphology”, *Pattern Recognition*, vol. 34, pp. 1813-1824, 2001.

[2] J. Weickert, “Applications of nonlinear diffusion in image processing and computer vision”, *Acta Math. Univ. Comenianae vol. 25*, 1(2001), pp. 33–50.

[3] D. Tschumperlé, R. Deriche, “Diffusion PDEs on vector valued images”, *IEEE Signal Processing Magazine, vol. 19, no. 5, pp.16-25*, September 2002.

[4] N.K. Pal, S.K. Pal, “A review on image segmentation techniques”, Pattern Recognition, vol. 26, pp. 1277-1294, 1993.

[5] L. Vincent, P. Soille, “Watersheds in Digital Spaces: An Efficient Algorithm Based on Immersion Simulations”, *IEEE Trans. On PAMI,* vol. 13, no. 6, June 1991.