Vis Comput (2011) 27: 707–716
DOI 10.1007/s00371-011-0588-5

ORIGINAL ARTICLE

Robust image segmentation against complex color distribution

Fan Zhong · Xueying Qin · Qunsheng Peng

Published online: 22 April 2011

Abstract Color distribution is the most effective cue that is widely adopted in previous interactive image segmentation methods. However, it also may introduce additional errors in some situations, for example, when the foreground and background have similar colors. To address this problem, this paper proposes a novel method to learn the segmentation likelihoods. The proposed method is designed for high reliability, for which purpose it may choose to discard some unreliable likelihoods that may cause segmentation error. The reliability of likelihoods is estimated in a few Expectation–Maximization iterations. In each iteration, a novel multi-class transductive learning algorithm, namely, the Constrained Mapping, is proposed to learn likelihoods and identify unreliable likelihoods simultaneously. The resulting likelihoods then can be used as the input of any segmentation methods to improve their robustness. Experiments show that the proposed method is an effective way to improve both segmentation quality and efficiency, especially when the input image has complex color distribution.

Keywords Segmentation · Likelihoods · Color distribution · Learning

F. Zhong · X. Qin (✉)
School of Computer Science and Technology, Shandong University, Jinan, China
e-mail: xyqin@cad.zju.edu.cn

F. Zhong
e-mail: zhongfan@cad.zju.edu.cn

Q. Peng
State Key Lab. of CAD&CG, Zhejiang University, Hangzhou, China
e-mail: peng@cad.zju.edu.cn

1 Introduction

Image segmentation has long been an active yet challenging research topic, and is found to be a fundamental problem in a wide variety of applications, including image editing, medical image processing, object recognition, etc.

Interactive image segmentation is the most flexible way to get what the user wants from input image. In recent years there have been proposed a lot of interactive segmentation methods. The popular methods can be categorized as follows: (1) *Graph Cut* (*GC*) based methods [3, 4, 17, 21]. Graph Cut [6] is an efficient way to minimize the energy function of Markov Random Fields (MRFs), which is the basic form of many image segmentation methods (see Sect. 2.1). In the past decade it has always been the most popular framework of image segmentation, for bilayer segmentation it can get the global optimal solution efficiently [6], and for multi-layer segmentation it also can get the local minimum by iteratively applying the binary cuts [4]. (2) *Random Walks* (*RW*) based methods [10, 11, 19], which view image segmentation as a diffusing process on a weighted graph, with each node corresponds to a pixel and the weight of each edge relates to the color difference of the two connected pixels, the segmentation result then can be obtained by solving a sparse linear system. Compared with GC, RW is less efficient; however, it can produce soft segmentation, which is desirable in some situations. (3) *Geodesic* based methods [1, 7, 12, 16, 23], which classify pixels mainly by their geodesic distance to the foreground and background seed pixels, the geodesic distance is calculated based on image gradients. By using the fast geodesic distance transform, these methods can be extremely fast, and thus are suitable for video segmentation.

Although the above methods are different in many aspects, the cues they use to solve for the segmentation are

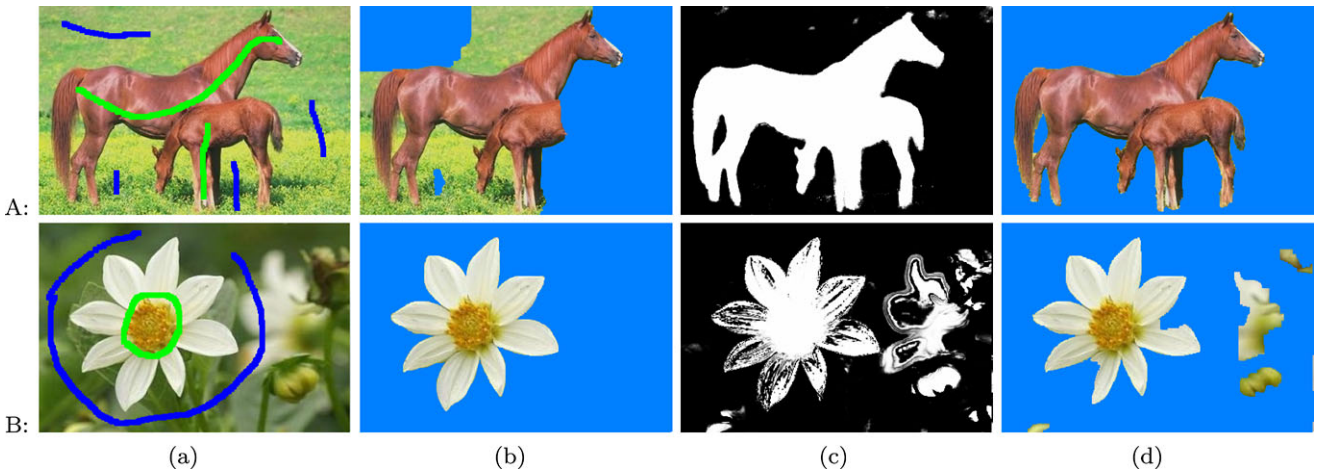

Fig. 1 The learnt likelihoods can result in either better (A) or worse (B) segmentation result. (**a**) Input image and user strokes. (**b**) Results produced without using any learnt likelihoods. (**c**) Likelihoods map obtained with GMMs. (**d**) Results produced with the likelihoods in (**c**)

common, and mainly contain: (1) constraints of seed pixels, whose states are manually marked by the user; (2) the learnt likelihoods. The likelihood of each unknown pixel measures its probability to belong to foreground and background, and is typically estimated based on the color distribution of foreground and background, which can be trained with the colors of seed pixels as training data. Figure 1(c) shows the examples of likelihoods map. (3) The priors about the segmentation, a good segmentation is typically assumed to be smooth and contrast-sensitive [3], which is helpful to suppress noise and make the segmentation boundary align to image edges.

Intuitively, more cues provide us with more knowledge about the segmentation, and thus should produce better segmentation. However, this is not always true. It has been proven that using the smooth prior may cause long thin objects to be cut off [21], and using the contrast-sensitive prior may result in missegmentation along the undesirable strong image edges [20]. So far there has no special attention paid to the learnt likelihoods. Previous works usually adopt Gaussian Mixture Models (GMMs) to model the color distribution of the foreground and background, but never discuss its side-effect. In fact, as demonstrated in Fig. 1B, the likelihoods learnt though GMMs may contain a lot of errors, which would cause the result to be even worse than that produced without using any learnt likelihoods.

Generally, because there is no learning algorithm that can achieve zero error rate, the learnt likelihoods unavoidably contain some error, i.e. *false likelihoods*. A small quantity of false likelihoods would not cause any problem because their effects can be suppressed by using priors. However, if large quantity of false likelihoods are involved as cues, missegmentation would occur, which often happens to input images of complex color distribution. For example, in Fig. 1B some parts of the foreground and background have similar color, in which case there is no classifier can guarantee the accuracy of the learnt likelihoods. Therefore, the learnt likelihoods are not always trustable, and instead of using all of them as the segmentation cue, only those of high reliability should be used.

Notice that the reliability is not the probability of the predicted class because the latter can be seriously biased due to imperfect inductive biases [14]. A common misunderstanding about the probability is that if a feature is undistinguishable between two classes (e.g. a color appearing in both foreground and background), a classifier would assign it nearly equal probabilities to be the two classes. This is not true for most classifiers, including GMMs, k-NN and SVM. Figure 1B shows you the case of GMMs. Section 2.3 will further discuss about these classifiers in more details. Using these classifiers to learn the likelihoods may introduce a lot of segmentation errors, as demonstrated in Fig. 1B.

In this paper we propose a novel method to learn the segmentation likelihoods. Unlike previous learning algorithms, our method is designed for high reliability but not for high prediction rate, and it may answer "unknown" in extremely difficult cases in order to avoid making a mistake (producing false likelihoods). In this way it can be guaranteed that using learnt likelihoods would not introduce additional errors (the case in Fig. 1B). Compared with previous works on image segmentation, which achieve improvements mainly by introducing novel solving method or additional cues, we are devoted to a more fundamental problem, that is, the way to obtain reliable likelihoods, which is of great significance because no segmentation method can work well with likelihoods that contain a lot of errors. Our method can be applied to any method that utilize the learnt likelihoods, and can help them to get better result with less user interaction. It also makes the system more predictable and more con-

trollable, and it may ease the user's pain to tune parameters, especially for input images of complex color distribution.

2 Segmentation likelihoods

In this section we will first briefly introduce a general segmentation framework which serves as the basic form of wide variety of segmentation methods, then discuss some properties of the learnt likelihoods and show the importance of learning reliable likelihoods.

2.1 A general segmentation framework

Let us denote by $\mathbf{x} = (x_1, \ldots, x_i, \ldots, x_N)$ an array of colors that represents the input image, with i the index of pixels. $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_i, \ldots, \alpha_N)$ is the corresponding segmentation result, where $\alpha_i \in \{0, 1\}$ is the state of the ith pixel. For soft segmentation and matting, α_i can also take continuous values between 0 and 1.

The Gibbs energy formulation of image segmentation then can be expressed as the posterior of $\boldsymbol{\alpha}$:

$$p(\boldsymbol{\alpha}|\mathbf{x}) = \frac{1}{p(\mathbf{x})} p(\mathbf{x}|\boldsymbol{\alpha}) p(\boldsymbol{\alpha}) \tag{1}$$

In terms of the Bayes theorem, $p(\mathbf{x})$ is the normalizing constant, $p(\mathbf{x}|\boldsymbol{\alpha})$ is the likelihoods, and $p(\boldsymbol{\alpha})$ is the priors. The above equation can be factored with respect to pixels:

$$p(\boldsymbol{\alpha}|\mathbf{x}) = \frac{1}{p(\mathbf{x})} \prod_i P(\alpha_i) \prod_{(i,j)} P(\alpha_i, \alpha_j) \prod_{(i,j,k)} \cdots \tag{2}$$

where $P(\alpha_i) = p(x_i|\alpha_i)p(\alpha_i)$ is the brief representation of the unary terms, $P(\alpha_i, \alpha_j)$ is the binary terms. The factorization can be greatly simplified by supposing the Markov property, which assumes that only neighboring pixels are directly related, therefore:

$$\begin{aligned} p(\boldsymbol{\alpha}|\mathbf{x}) &\approx \frac{1}{p(\mathbf{x})} \prod_i P(\alpha_i) \prod_{(i,j)\in\mathfrak{N}} P(\alpha_i, \alpha_j) \\ &= \frac{1}{Z} \exp\{-E(\boldsymbol{\alpha})\} \end{aligned} \tag{3}$$

where

$$E(\boldsymbol{\alpha}) = \sum_i D(\alpha_i) + \lambda \sum_{(i,j)\in\mathfrak{N}} S(\alpha_i, \alpha_j) \tag{4}$$

in which $\mathfrak{N}$ is the set of all neighborhoods. $E(\boldsymbol{\alpha})$ is the energy function that can be minimized to get $\boldsymbol{\alpha}$. $D(\alpha_i)$ and $S(\alpha_i, \alpha_j)$ are the two types of energy terms that relate to the likelihoods and priors, and are usually called data and smooth terms, respectively, as the cues they encode indicate.

In interactive image segmentation, the states of the seed pixels are known, which provides us with another type of constraint, namely, the *seed constraint*. Adding seed constraint into (4) is straightforward:

$$\begin{aligned} E(\boldsymbol{\alpha}) = c \sum_{i\in\mathcal{Q}} |\alpha_i - \dot{\alpha}_i|^2 + \sum_i D(\alpha_i) \\ + \lambda \sum_{(i,j)\in\mathfrak{N}} S(\alpha_i, \alpha_j) \end{aligned} \tag{5}$$

where $\mathcal{Q}$ is the set of seeds, $\dot{\alpha}_i$ are their true states, c is a large constant number. The above equation is essential the same as (4) because the seed terms can be merged with the data terms (both are unary). Here we separate them apart just for clarity when to discuss their effects.

Equation (5) is the basic form of many segmentation methods, including the popular GraphCuts and Random Walks based methods. The Geodesic-based methods also can be formulated in this way by some extensions [7].

2.2 Effect of the learnt likelihoods

The learnt likelihoods are not always necessary, for example, in [11] and [19] the likelihoods are not involved, i.e. the data terms in (5) are constant. However, in most situations using the learnt likelihoods as cues is helpful to reduce user interaction [10], especially when segmenting objects of complex topology is in order, or when the input image is highly textured (as the case in Fig. 1A).

In fact, the learnt likelihoods acts the same as seed constraints, except that it is estimated through learning and thus may contain errors. For two-layer segmentation, the likelihood of the ith pixel can be represented as its probabilities to be the foreground ($p(x_i|F)$) and background ($p(x_i|B)$), which can be used to estimate the state of the ith pixel:

$$\hat{\alpha}_i = \frac{p(x_i|F)}{p(x_i|F) + p(x_i|B)} \tag{6}$$

$\hat{\alpha}_i$ can also be regarded as the observed value of $\dot{\alpha}_i$ (the true state of the ith pixel). Now that $\hat{\alpha}_i$ is estimated through learning, error is unavoidable. The likelihood of the ith pixel is said to be *false* if $|\hat{\alpha}_i - \dot{\alpha}_i| > 0.5$. For example, the likelihood of a background pixel is false implies $\hat{\alpha}_i > 0.5$, and subsequently $p(x_i|F) > p(x_i|B)$, which obviously contradict the truth.

As demonstrated in Fig. 1B, the learnt likelihoods can cause missegmentation if much of them are false. Figure 1B(d) is the likelihoods map visualized through the estimated states $\hat{\alpha}_i$, from which one can easily find out a lot of false likelihoods, and it is these false likelihoods cause the missegmentation in Fig. 1B(e). Although for this example a better result can be obtained by emphasizing smooth priors (increasing λ), the bad effect of large area of dense

false likelihoods is hard to be eliminated in this way. More importantly, as we mentioned in the introduction, over-emphasizing smooth priors may cause long thin objects to be cut off. Consequently, we often find it very hard to set the parameter λ. In fact, when a large quantity of false likelihoods is present, one can never get accurate segmentation by solely tuning parameters of the energy equation.

2.3 Trivial likelihoods vs. false likelihoods

Recall that $p(x_i|F) = p(x_i|B)$ (or $\hat{\alpha}_i = 0.5$) indicates the ith pixel has equal probability to be both foreground and background, this type of likelihoods would not take effect to segmentation result, so we call them *trivial likelihoods*. In Fig. 1(b) the likelihoods are not used, which is equivalent to assign all pixels trivial likelihoods. In Fig. 1B, the result produced without the learnt likelihoods is much better than that produced with. This observation tells us: (1) *having trivial likelihoods is better than false likelihoods*. With trivial likelihoods we gain nothing, but with false likelihoods we gain negative. False likelihoods can only make things worse. (2) *more likelihoods does not mean better result, even false likelihoods are not considered*. As in the case of Fig. 1B, sometimes using seed constraints and the contrast-sensitive priors is enough to get very accurate segmentation, in which case adding the likelihoods is of little help. Even the likelihoods is necessary, it is usually not necessary everywhere.

Therefore, instead of seeking the most true likelihoods, the learning algorithm should seek the least false likelihoods by setting some likelihoods to be trivial. Unfortunately, the popular learning algorithms, including GMMs, k-NN, and SVM, are all designed for highest average prediction rate (most true likelihoods). We refer to this type of learning algorithms as *gainful learning*. Consider the case when two classes overlap in feature space, which corresponds to the case when the foreground and background have similar colors, in the overlapped regions a gainful learning algorithm would prefer the class that occurs more often, which would definitely cause the other class be misclassified. According to our above analysis, in such a difficult case it is better to faithfully answer "unknown" in order to eliminate the risk of producing false likelihoods. If a learning algorithm behaves like this, it is called *faithful learning*. A faithful learning algorithm should never make a decision unless it is confident of the answer, so it is more trustworthy and more suitable for learning the segmentation likelihoods.

3 Learning reliable likelihoods

A gainful learning algorithm tends to make a mistake in two cases, the first case is when some parts of the foreground and background have similar colors (Fig. 1B); the second case is when user interaction is not enough to cover every cluster of colors in the image, as demonstrated in Fig. 2. Unlike the first case, the second case is not easy to be observed. Figure 2 is an example just for illustration. In the first case, different classes overlap in feature space and are hard to be distinguished. If a feature falls into overlapped regions, it is said to be *ambiguous*. In the second case, the number of training data is not enough to cover the whole feature space, and in uncovered regions the learning algorithm can only give predictions by guessing. If a feature falls into uncovered regions, it is said to be *unconstrained*. Therefore, if the feature of a pixel is ambiguous or unconstrained, the estimated likelihood is likely to be false, and to identify false likelihoods is equivalent to identify ambiguous and unconstrained features.

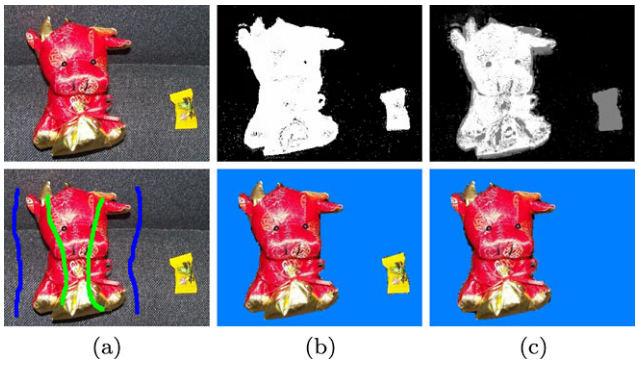

Fig. 2 Missegmentation due to uncovered cluster of colors. (**a**) Input image and strokes. *The yellow background object* is covered by neither foreground nor background brushes. (**b**) Result (*bottom*) got with likelihoods (*top*) learnt with GMMs. *The yellow object* is mistaken as foreground because it is closer to the doll than to the background in color space (although both are very far). (**c**) Result got with likelihoods learnt with our method. *The yellow object* are assigned trivial likelihoods, and then can be correctly segmented by coherence

3.1 From gainful learning to faithful learning

The result of a m-class learning algorithm can be expressed as m-class classifier $\mathcal{P} = f(\mathcal{X}|\mathcal{S})$, where $\mathcal{X}$ and $\mathcal{S}$ are the input and training data sets, respectively. For image segmentation $\mathcal{X}$ is known and finite: $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_i, \ldots, \mathbf{x}_N\}$, where i is the index of pixels. Both $\mathcal{X}$ and $\mathcal{S}$ contain m classes of features, $\mathcal{X} = \bigcup_{k=1,\ldots,m} \mathcal{X}_k$, $\mathcal{S} = \bigcup_{k=1,\ldots,m} \mathcal{S}_k$, with $\mathcal{X}_k$ and $\mathcal{S}_k$ the subset of features of the kth class, $\mathcal{S}_k \subset \mathcal{X}_k$. $\mathcal{P} = \{\mathbf{p}_1, \ldots, \mathbf{p}_i, \ldots, \mathbf{p}_N\}$, $\mathbf{p}_i \in \Re^m$ is the probability vector of $\mathbf{x}_i$, with $\mathbf{p}_i^{(k)}$ the probability of the kth class, so $\mathbf{x}_i$ is predicted to belong to the kth class if $\mathbf{p}_i^{(k)}$ is maximum.

The case of faithful learning is a little different because now besides the original m classes, an additional class is necessary in order to answer "unknown". We call this special class the *trivial class*, and use zero to index it. Now the probability vector is $m + 1$ dimensional, $k =$

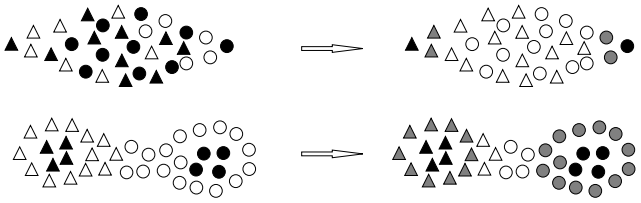

Fig. 3 Two cases of ambiguous features. *The left* is input, with *black filled shapes* the training data. *The right* is output, and *unfilled shapes* are labeled as ambiguous. *Top*: training data of different classes are overlapped. *Bottom*: training data of different classes are connected through test data

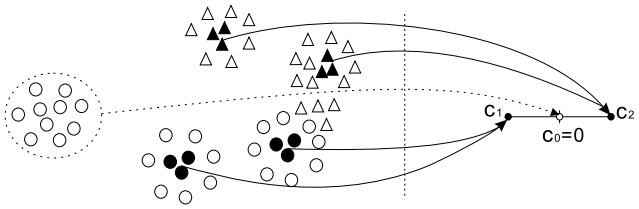

Fig. 4 The 2-class Constrained Mapping: the input data (*left*) are mapped to the intermediate space (*right*) under the constraints of the training data (*filled*). The features contained *in the large circle* are unconstrained, and are mapped to $\mathbf{c}_0 = \mathbf{0}$

$0, 1, \ldots, m$, and the corresponding classifier is represented as $\tilde{\mathcal{P}} = \tilde{f}(\mathcal{X}|\mathcal{S})$, with $\tilde{\mathcal{P}} \subset \Re^{m+1}$.

With the above notations f can be regarded as the special case of $\tilde{f}$ with $\tilde{\mathbf{p}}_i^{(0)} \equiv 0$. Interestingly, $\tilde{f}$ can also be converted to f simply by introducing the training data of the trivial class. Denoting by $\tilde{\mathcal{S}}$ the extended training data set with the trivial class' training data set $\tilde{\mathcal{S}}_0$ nonempty, then $\tilde{f}(\mathcal{X}|\mathcal{S})$ is equivalent to $f(\mathcal{X}|\tilde{\mathcal{S}})$, and can be learnt with any multi-class gainful learning algorithm, e.g. GMMs. However, since $\tilde{\mathcal{S}}_0$ cannot be drawn directly from seed pixels as other classes, we must construct it by other way.

If $\mathcal{X}$ is labeled, ambiguous features can be easily found out according to the density of the non-trivial classes. Specifically, $\forall \mathbf{x}_i \in \mathcal{X}$, if there exist at least two non-trivial classes both have non-zero density at $\mathbf{x}_i$, $\mathbf{x}_i$ can be considered to be ambiguous. The problem now is that $\mathcal{X}$ is not labeled, so we must learn the labeling of $\mathcal{X}$ and construct $\tilde{\mathcal{S}}_0$ at the same time, which can be done in the following EM process:

E-step: updating the training data set $\tilde{\mathcal{S}}$ according to current labeling (or $\tilde{\mathcal{P}}$). $\forall \mathbf{x}_i \in \mathcal{X}$, if there exist at least two *non-trivial* classes both have non-zero density at $\mathbf{x}_i$, first remove $\mathbf{x}_i$ from $\tilde{\mathcal{S}}_j$ if $\mathbf{x}_i \in \tilde{\mathcal{S}}_j$, $j \neq 0$, then add it into $\tilde{\mathcal{S}}_0$.

M-step: re-classifying $\mathcal{X}$ with respect to current extended training data set $\tilde{\mathcal{S}}$, that is, redo $\tilde{\mathcal{P}} = f(\mathcal{X}|\tilde{\mathcal{S}})$.

The iteration can start from M-step by simply setting $\tilde{\mathbf{p}}_i^{(0)} = 0$, $\forall i$. It stops until no more ambiguous features can be found. Figure 3 shows two typical cases of input data that may result in ambiguous features. Notice that in the first case some training features are cast to ambiguous features. This is reasonable because these colors occur both in foreground and background. Unlike the first case of ambiguous features, which can be found out in the first iteration, the second case of ambiguous features can only be found out in the second or third iteration because the training data of the two classes are not overlapped; however, after the first iteration there must be some ambiguous features emerge from the features connecting the training data.

So far we have considered only ambiguous features, unconstrained features cannot be identified with the above method. In addition, the classifier f is also not defined yet. In the next section we will present a novel learning algorithm that can address both problems in an elegant way.

3.2 Constrained mapping

The classifier f can be regarded as a map $f : \mathcal{X} \to \mathcal{P}$, without loss of generality, it can be decomposed into two consecutive maps $f^\dagger : \mathcal{X} \to \mathcal{Y}$, $f^\ddagger : \mathcal{Y} \to \mathcal{P}$, with $\mathcal{Y} \subset \Re^n$ the *intermediate space*, and $\mathcal{Y} = \{\mathbf{y}_1, \ldots, \mathbf{y}_i, \ldots, \mathbf{y}_N\}$. Let $\mathcal{C} = \{\mathbf{c}_1, \ldots, \mathbf{c}_k, \ldots, \mathbf{c}_m\}$ be a set of points in $\Re^n$. Our basic idea is simple: to choose $f^\dagger$, so that $\|\mathbf{y}_i - \mathbf{c}_k\| < \|\mathbf{y}_i - \mathbf{c}_{k'}\|$, $\forall k' \neq k$, if $\mathbf{x}_i$ belongs to the kth class. $\mathbf{c}_k$ is called the *center* of the kth class in the intermediate space. Note that $f^\dagger$ itself is a classifier that coincide with f, so $f^\ddagger$ is used only to convert the distance in the intermediate space to probability, and is easy to be determined once $f^\dagger$ is given.

Let us assign each pair of features $\mathbf{x}_i, \mathbf{x}_j$ a weight w_{ij}, which measures their similarity. Larger w_{ij} means $\mathbf{x}_i$ and $\mathbf{x}_j$ are more likely to belong to the same class, then in order to make $f^\dagger$ meet our requirement, $\mathbf{y}_i$ and $\mathbf{y}_j$ should stay closer in $\Re^n$, so we can force them to have the following relationship:

$$\mathbf{y}_i = \frac{\sum_j w_{ji}\mathbf{y}_j}{\sum_j w_{ji}} \quad j = 1, \ldots, i-1, i+1, \ldots, N \tag{7}$$

which means $\mathbf{y}_i$ is the weight sum of all $\mathbf{y}_j$, $j \neq i$, with w_{ji} as the weights. The larger w_{ji} would pull $\mathbf{y}_i$ closer to $\mathbf{y}_j$, just as we have expected. On the other hand, if $\mathbf{x}_i$ is a training feature, $\mathbf{y}_i$ should be the center of the corresponding class:

$$\mathbf{y}_i = \mathbf{c}_k \quad \text{if } \mathbf{x}_i \in \mathcal{S}_k \tag{8}$$

Now $\mathcal{Y}$ can be obtained by solving the system of linear equations (7) under the constraints of (8). We call this method *constrained mapping* (CM) in that it maps the input data directly into the intermediate space under the constraints provided by the training data, as illustrated in Fig. 4.

The centers Although in theory CM can work if only no two centers are the same, the choice of the centers does influence its accuracy. Consider the case when $m = 3$ (3-class

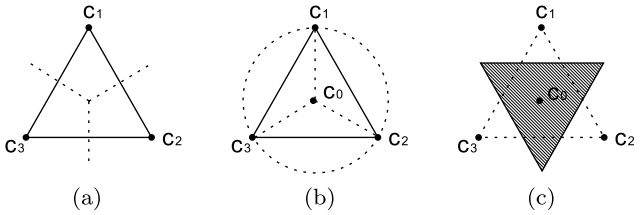

Fig. 5 The centers of the 3-class Constrained Mapping

CM), and an ambiguous feature $\mathbf{x}_i$ that has equal probability to be all of the three classes, then the ideal position for $\mathbf{y}_i$ should have equal distance to the three centers $\mathbf{c}_1$, $\mathbf{c}_2$ and $\mathbf{c}_3$. However, you can easily verify that such point is not exist if the intermediate space is one dimensional.

Generally, since ambiguous features may occur between any two classes, in order to ensure that every ambiguous feature can find a proper position in the intermediate space, the occupying region of every class in the intermediate space should be neighbor to the occupying region of every other class. In addition, classes should be treated equally if we have no reason to prefer any of them. Therefore, we propose to *choose the centers* $\mathbf{c}_1, \ldots, \mathbf{c}_i, \ldots, \mathbf{c}_m$ *as the vertices of a regular m-polyhedral in space* $\Re^{m-1}$. For $m = 2$, it is a line segment; for $m = 3$, it is an equilateral triangle (Fig. 5(a)); for $m = 4$, it is a regular tetrahedral, etc.

The trivial class An advantage of CM is that it provides us a natural way to incorporate the trivial class. Denoting by $\mathbf{c}_0$ the center of the trivial class. In view of ambiguous features, $\mathbf{c}_0$ should have equal distance to other centers, and thus should be the circumcenter of the chosen regular m-polyhedral (Fig. 5(b)). Figure 5(c) shows the occupying region of the trivial class (the hatched region). Notice that according to (7), $\mathbf{y}_i$ always fall in the convex hull formed by $\mathbf{c}_1, \ldots, \mathbf{c}_m$, so from Fig. 5(c) you can see the non-trivial classes are separated apart by the trivial class.

We still have two unsolved problems; one is how to compute the weights, the other is how to identify unconstrained features. In fact this two problems are closely related. We use the Gaussian kernel as the weighting function:

$$w_{ij} = e^{-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}} \tag{9}$$

When the distance of $\mathbf{x}_i$ and $\mathbf{x}_j$ is much larger than σ, w_{ij} is very small and can be truncated to zero. This simple operation brings us two great benefits: first, the linear system becomes much sparser and thus can be solved much easier; second, the unreliable "weak connections" between features are disconnected. Weak connections are troublesome because they are not able to faithfully reflect user intention, as demonstrated in Fig. 2. Recall that unconstrained features are those features that fall into uncovered regions, which implies they are far from training data, and by disconnecting weak connections, unconstrained features are isolated from training data, and then solving (7) would map them all to the trivial solution $\mathbf{0}$ due to the missing of the constraints of (8), as illustrated in Fig. 4. So in order to merge unconstrained features into the trivial class, we need only to *truncate the small weights to zero and then properly choose the centers so that* $\mathbf{c}_0 = \mathbf{0}$.

Connections with previous methods Equation (7) can be written in matrix form:

$$LY = \mathbf{0} \tag{10}$$

where $Y = (\mathbf{y}_1, \ldots, \mathbf{y}_i, \ldots, \mathbf{y}_N)^T$ is a matrix of unknowns. Since Y can be solved column by column, we can temporally suppose it is a column vector. L is a $N \times N$ matrix defined as follows:

$$L_{ij} = \begin{cases} \sum_{j'=1,\ldots,i-1,i+1,\ldots,N} w_{j'i} & \text{if } i = j \\ -w_{ji} & \text{otherwise} \end{cases} \tag{11}$$

L is widely known as the Laplace–Beltrami operator. It has appeared in circuit theory [8], spectral clustering [15], manifold learning [2, 13] and image segmentation [11].

Among all of those works the one most close to CM is [13], which is a transductive learning algorithm attempts to minimize the following objective function:

$$\underset{Y}{\text{argmin}} \quad Y^T LY + c(Y - Y_0)^T D(Y - Y_0) \tag{12}$$

$$\text{s.t.} \quad Y^T Y = \mathbf{1} \quad \text{and} \quad Y^T \mathbf{1} = \mathbf{0} \tag{13}$$

where Y_0 represents the value of the labeled data; D is a $N \times N$ diagonal matrix, $D_{ii} = 1$ if $\mathbf{x}_i$ is labeled, $D_{ii} = 0$ otherwise; c is a large constant number. In [13], the above problem is treated as a constrained eigenvalue problem [9], and its optimal solution can be approximated through the leading eigenvectors of L. This approach is very similar to [2], and both are inherited from unsupervised ratio-cuts [18]. In unsupervised case, $D = \mathbf{0}$, so in order to fix the solution, (13) must be introduced to remove arbitrary scaling (by $Y^T Y = \mathbf{1}$) and the trivial solution (by $Y^T \mathbf{1} = \mathbf{0}$). In supervised case we need no longer to do like this because now Y is uniquely determined by (12). Taking derivative to (12) with respect to Y and setting it to zero results:

$$(L + cD)Y = cDY_0 \tag{14}$$

One can easily verify that to solve the above linear system is equivalent to solve (7) under the constraint of (8), so for two-class learning without considering the trivial class, CM is equivalent to [13] if the latter uses all eigenvectors. However, CM is much faster than [13] because to solve the linear system (14) is much faster than to search the eigenvectors of L (a few seconds against several minutes).

3.3 Implementation details

Naive implementation of the algorithm proposed in this section may result in poor performance. In order to determine whether a feature is ambiguous or not, we need to estimate the kernel density of the non-trivial classes, which is costly in high dimensional space. In this paper we use RGB color of pixels as their features, and define density at each position as the number of samples contained in the $9 \times 9 \times 9$ cube centered at it, then the density can be computed efficiently by extending the integral image [22] to 3D color space, which is straightforward.

There are two ways to truncate the weights, one is to truncate all weights below a given threshold, the other is to keep only the largest k weights. We test both ways and find they have no large difference in accuracy, while the latter way is more efficient because it can be greatly accelerated through kd-tree. We also find that CM is not sensitive to the choice of k, in our implementation we choose $k = 15$. Notice that k is much smaller than the number of features, so the property of (7) is just locally preserved. Globally preservation of (7) can be achieved by using large k, but we found it brings us nothing except large computational cost.

After getting $\mathbf{y}_i$, we can compute the probability vector of $\mathbf{x}_i$ (to determine $f^{\ddagger}$). Specifically, let $d_{ik} = \frac{1}{\|\mathbf{y}_i - \mathbf{c}_k\|}$, $k = 0, 1, \ldots, m$. $\mathbf{x}_i$ is trivial if $d_{i0} > d_{ik}, \forall k \neq 0$. The probability vector $\mathbf{p}_i$ then can be computed as:

$$\mathbf{p}_i^{(k)} = \begin{cases} 1/m & \text{if } x_i \text{ is trivial} \\ \frac{d_{ik}}{\sum_{k \neq 0} d_{ik}} & \text{otherwise} \end{cases} \tag{15}$$

If $\mathbf{x}_i$ is trivial, we simply set $\mathbf{p}_i^{(k)} \equiv 1/m$ so that it has equal probability to be all classes, which means the corresponding pixels are assigned trivial likelihoods, just as we have expected.

4 Experimental results

In experiments we use the proposed method to learn the likelihoods of unlabeled pixels. The priors model is chosen to be the contrast-sensitive Ising prior [3]. The final segmentation result is obtained by minimizing (5) with the binary graph-cuts [5].

In order to evaluate the quality of the learnt likelihoods, we first normalize it so that $p(x_i|F) + p(x_i|B) = 1, \forall i$, then compute the gain, error and loss as follows:

$$\begin{aligned} \text{gain} &= \frac{1}{N} \sum_{i \in \mathcal{T}} \big| p(x_i|F) - p(x_i|B) \big| \\ \text{error} &= \frac{1}{N} \sum_{i \in \mathcal{F}} \big| p(x_i|F) - p(x_i|B) \big| \\ \text{loss} &= 1 - \text{gain} - \text{error} \end{aligned} \tag{16}$$

Table 1 Gain (%), error (%) and computation time (seconds) of different learning algorithms (gain/error(time)). In brackets is the number of different colors contained in each image. CM* and FL* quantize input image into 4096 colors with median-cut. k-NN is accelerated through kd-tree

	Figure 1A [56018]	Figure 1B [33865]	Figure 6A [50180]
GMM	96.1/1.74(0.44)	82.5/10.2(0.29)	54.2/9.09(0.33)
k-NN	96.4/1.37(1.42)	82.7/9.61(2.16)	74.9/6.37(1.86)
SVM	94.9/1.53(0.97)	80.4/11.3(7.06)	77.5/5.09(41.5)
CM	92.9/0.80(5.61)	81.5/9.05(4.45)	77.0/4.75(3.36)
CM*	93.1/0.82(0.51)	80.9/9.05(0.52)	57.6/3.15(0.48)
FL	91.3/0.34(8.67)	69.6/0.65(7.17)	48.7/0.78(9.91)
FL*	92.0/0.39(0.88)	70.1/0.91(0.85)	50.9/1.85(0.88)

where $\mathcal{T}$ and $\mathcal{F}$ are the sets of pixels with true and false likelihoods, respectively, N is the number of pixels. Let us denote by FL the proposed faithful learning algorithm (CM is just one step of FL). Table 1 lists the gain and the error rates of different methods as well as their computation time, from which you can see that CM achieves lower error rates than GMMs, k-NN and SVM. Since CM needs to solve a large linear system, it is not as fast as GMMs and k-NN. Fortunately, observing that the size of the linear system is equal to the number of different colors, so CM can be greatly accelerated by quantizing the input image with less colors. CM* is a version of CM that quantizes input image with 4096 colors. The accuracy of CM* is comparable with CM, but CM* is much faster. Compared with CM , FL further reduces the error rates to a great extent. Since FL can converge in only two or three iterations, at the same time the matrix L needs only to be computed once, FL costs only a little more time than CM.

Figure 6A is an example of complex color distribution. For such images it is very hard to cover every cluster of colors with brush, in which case k-NN would produce a lot of false likelihoods in the uncovered regions, while our method performs much better by introducing some trivial likelihoods. Both Figs. 6B and 7 are challenging examples of ambiguous colors. Our method assigns only to those pixels of unambiguous colors nontrivial likelihoods, so it produces better results than both k-NN (which introduces too many false likelihoods) and the method that does not make use of any learnt likelihoods (which introduces no true likelihoods). Table 2 lists the error rates of the segmentation results obtained with the likelihoods learnt through k-NN and our method.

The missegmentation as in Fig. 6A(d) can be eliminated by increasing coherency. However, in this way the long thin objects may be cut off. Even if this would not happen, to tune the parameters is very tedious. In fact the error is due to false likelihoods but not the lacking of coherency, so it is unwise to reduce error by increasing coherency. Figure 6B shows

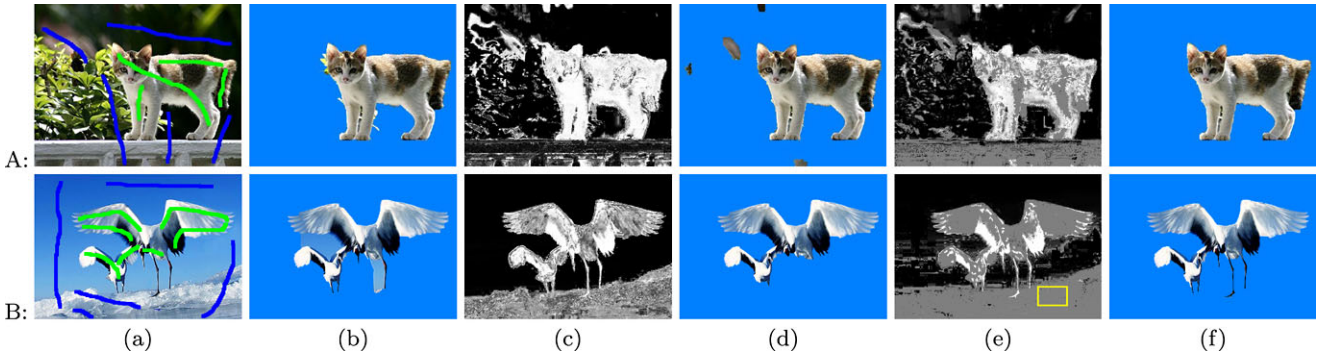

Fig. 6 Segmentation results. (**a**) Input image and strokes. (**b**) Segmentation result without using the learnt likelihoods. (**c**), (**d**) The likelihoods learnt with k-NN and the corresponding segmentation result. (**e**), (**f**) The likelihoods learnt with our method and the corresponding segmentation result. In (**e**), the pixels with intensity 127 (as the pixels in the rectangle) are assigned trivial likelihoods

Table 2 Error rates (%) of the final segmentation produced with likelihoods learnt through k-NN and our method

	Figure 1A	Figure 1B	Figure 6A	Figure 6B	Figure 7
k-NN	0.65	5.88	2.28	1.30	11.8
FL	0.69	0.96	0.72	0.83	0.86

that our method is helpful to solve this problem. Since most false likelihoods are eliminated by the proposed learning method, using weak coherency is enough to get a good result (Fig. 6B(f)), otherwise strong coherency is necessary in order to counteract the effect of false likelihoods (Fig. 6B(d)).

In [10], the authors showed that using the learnt likelihoods is helpful to segment objects of many disconnected parts. Figure 8A illustrates this case. However, if our purpose is to get only one of these parts, say, the largest flower in Fig. 8, using the likelihoods learnt with previous learning algorithms may cause problems because now the foreground and background have nearly the same color distribution. With our method one can deal with both cases easily. In the first case it can greatly reduce interaction (Fig. 8A), in the second case the user needs only to mark the largest flower as foreground and one of other flowers as background, then all flowers would be assigned trivial likelihoods and can be correctly segmented (Fig. 8B). Note that in the second case, the likelihoods of the background regions other than the flowers still take effect.

Besides Graph Cut, we also tested Random Walks [10]. The likelihoods learnt with our method again produced much better results when the input image has a complex color distribution (Fig. 9). In fact, we found that when the learnt likelihoods is involved as the segmentation cues, it would dominate the overall effect of the segmentation result. The missegmentation produced by GC and RW even tends to appear at the same place, both in regions of many false likelihoods.

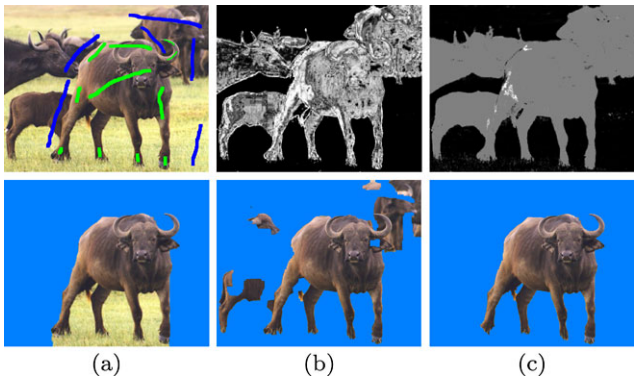

Fig. 7 A challenging example of ambiguous colors. (**a**) Input and the result without using the learnt likelihoods. (**b**) The likelihoods learnt with k-NN and the final segmentation result. (**c**) The likelihoods learnt with the proposed method and the final segmentation result

Fig. 8 Segmenting object of many disconnected parts (*top*) or only one of the parts (*bottom*), with the likelihoods learnt with the proposed method both cases can be handled easily. The middle column shows the learnt likelihoods in both cases

5 Conclusion

This paper proposed a novel method to learn the segmentation likelihoods. The proposed method can be combined with wide variety of segmentation methods to improve their robustness against complex color distribution. We demonstrated that the likelihoods learning algorithm should seek

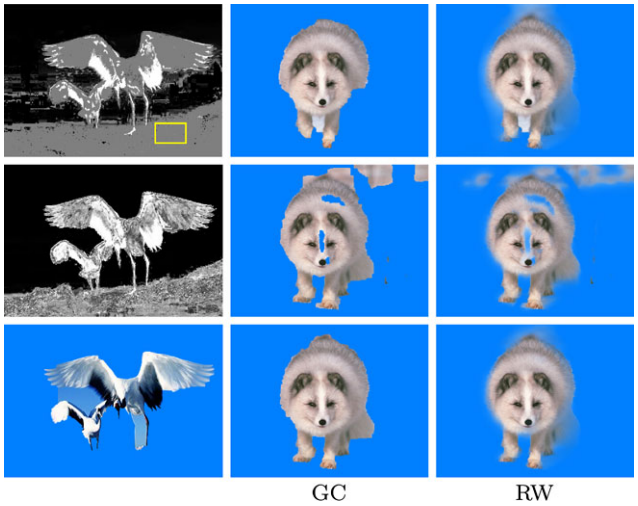

Fig. 9 Applying the proposed method to both Graph Cuts (GC) and Random Walks (RW) based segmentation. *From top to bottom*: input and the results without using the likelihoods, results of k-NN, results of our method. *From left to right*: input and the learnt likelihoods, results of GC, results of RW

for high reliability but not for high prediction rate, then proposed a faithful learning algorithm, which makes decisions only when it is confident of the answer. We introduced a density-based method that can be used to identify ambiguous features in a few EM iterations. In each iteration, a novel transductive learning algorithm, namely, the Constrained Mapping, is used to do a classification and identify unconstrained features.

Both qualitative and quantitative evaluation to the proposed method can convince us that: (1) CM produces less error than previous popular learning algorithms; (2) the proposed faithful learning algorithm can greatly reduce the quantity of false likelihoods; (3) reducing the quantity of false likelihoods is helpful to get better segmentation with less user interaction, at the same time can make the system more predictable and more controllable.

A limitation of our work is that it considers only the likelihoods learnt based on color distribution. We believe that by considering all cues simultaneously, one can further achieve improvements for more challenging examples.

Acknowledgements This work is supported by 973 program of China (No. 2009CB320802), Natural Science Foundation of China (No. U1035004, No. 60870003), and Natural Science Fund for Distinguished Young Scholars of Shandong Province (No. JQ200920). The authors would also like to thank the reviewers for their insightful comments and suggestions.

References

1. Bai, X., Sapiro, G.: A geodesic framework for fast interactive image and video segmentation and matting. In: Proceedings of International Conference on Computer Vision (ICCV), pp. 1–8 (2007)
2. Belkin, M., Niyogi, P.: Semi-supervised learning on manifolds. Mach. Learn. **56**, 209–239 (2004)
3. Blake, A., Rother, C., Brown, M., Perez, P., Torr, P.: Interactive image segmentation using an adaptive GMMRF model. In: Proceeding of European Conference on Computer Vision (ECCV), pp. 428–441 (2004)
4. Boykov, Y.Y., Jolly, M.P.: Interactive graph cuts for optimal boundary & region segmentation of objects in n-d images. In: Proceedings of International Conference on Computer Vision (ICCV), pp. 105–112 (2001)
5. Boykov, Y., Kolmogorov, V.: An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. IEEE Trans. Pattern Anal. Mach. Intell. **26**, 359–374 (2004)
6. Boykov, Y., Veksler, O., Zabih, R.: Efficient approximate energy minimization via graph cuts. IEEE Trans. Pattern Anal. Mach. Intell. **20**(12), 1222–1239 (2001)
7. Criminisi, A., Sharp, T., Blake, A.: Geos: geodesic image segmentation. In: Proceeding of European Conference on Computer Vision (ECCV), pp. 99–112 (2008)
8. Doyle, P.G., Snell, J.L.: Random walks and electric networks. Carus Math. Monogr. **22** (1984)
9. Gander, W., Golub, G., Matt, U.: A constrained eigenvalue problem. Linear Algebra Appl. 815–839 (1989)
10. Grady, L.: Multilabel random walker image segmentation using prior models. In: Proceedings of Computer Vision and Pattern Recognition (CVPR), pp. 763–770 (2005)
11. Grady, L.: Random walks for image segmentation. IEEE Trans. Pattern Anal. Mach. Intell. **28**(11), 1768–1783 (2006)
12. Gulshany, V., Rotherz, C., Criminisiz, A., Blakez, A., Zisserman, A.: Geodesic star convexity for interactive image segmentation. In: Proceedings of Computer Vision and Pattern Recognition (CVPR), pp. 3129–3136 (2010)
13. Joachims, T.: Transductive learning via spectral graph partitioning. In: International Conference on Machine Learning (ICML), pp. 290–297 (2003)
14. Kukar, M., Kononenko, I.: Reliable classifications with machine learning. In: International Conference on Machine Learning (ICML), pp. 219–231 (2002)
15. Ng, Y.A., Jordan, M.I., Weiss, Y.: On spectral clustering: analysis and an algorithm. In: Advances in Neural Information Processing Systems (NIPS), pp. 849–856 (2002)
16. Price, B.L., Morse, B., Cohen, S.: Geodesic graph cut for interactive image segmentation. In: Proceedings of Computer Vision and Pattern Recognition (CVPR), pp. 3161–3168 (2010)
17. Rother, C., Kolmogorov, V., Blake, A.: Grabcut Interactive foreground extraction using iterated graph cuts. In: Proceedings of ACM SIGGRAPH, pp. 309–314 (2004)
18. Shi, J., Malik, J.: Normalized cuts and image segmentation. IEEE Trans. Pattern Anal. Mach. Intell. **22**(8), 888–905 (2000)
19. Sinop, A.K., Grady, L.: A seeded image segmentation framework unifying graph cuts and random walker which yields a new algorithm. In: Proceedings of International Conference on Computer Vision (ICCV), pp. 1–8 (2007)
20. Sun, J., Zhang, W., Tang, X., Shum, H.Y.: Background cut. In: Proceeding of European Conference on Computer Vision (ECCV), pp. 628–641 (2006)
21. Vicente, S., Kolmogorov, V., Rother, C.: Graph cut based image segmentation with connectivity priors. In: Proceedings of Computer Vision and Pattern Recognition (CVPR), pp. 1–8 (2008)
22. Viola, P., Jones, M.: Robust real-time object detection. Int. J. Comput. Vis. **57**, 137–154 (2004)
23. Zheng, Y., Kambhamettu, C., Yu, J., Bauer, T., Steiner, K.: Fuzzymatte: a computationally efficient scheme for interactive matting. In: Proceedings of Computer Vision and Pattern Recognition (CVPR), pp. 1–8 (2008)

Fan Zhong is a lecturer of School of Computer Science and Technology, Shandong University. He received his Ph.D. from Zhejiang University in 2010. His research interests include image and video editing, computer vision.

Xueying Qin is currently a professor of School of Computer Science and Technology, Shandong University. She received her Ph.D. from Hiroshima University of Japan in 2001, and M.S. and B.S. from Zhejiang University and Peking University in 1991 and 1988, respectively. Her main research interests are augmented reality, video-based analyzing and rendering, and photo-realistic rendering. Her main goal is to build an augmented world by vision-based methods with photo-realistic quality, especially for outdoor scenes.

Qunsheng Peng is a Professor in the State Key Lab of CAD& CG at Zhejiang University. His research interests include realistic image synthesis, virtual reality, scientific visualization and biological computing, etc. He graduated from Beijing Mechanical College in 1970 and received a Ph.D. from School of Computing Studies, University of East Anglia, in 1983. He is currently the chairman of the Professional Committee of CAD and Graphics, China Computer Federation and serves as a member of the editorial boards of several international and domestic journals.