ISPRS Journal of Photogrammetry and Remote Sensing 113 (2016) 1–16

Contents lists available at ScienceDirect

ISPRS Journal of Photogrammetry and Remote Sensing

journal homepage: www.elsevier.com/locate/isprsjprs

Optimal seamline detection for multiple image mosaicking via graph cuts

Li Li [a], Jian Yao [a,*], Xiaohu Lu [a], Jinge Tu [a], Jie Shan [b]

[a] School of Remote Sensing and Information Engineering, Wuhan University, Wuhan, Hubei, PR China
[b] School of Civil Engineering, Purdue University, 550 Stadium Mall, West Lafayette, IN 47907, USA

ARTICLE INFO

Article history:
Received 27 February 2015
Received in revised form 19 December 2015
Accepted 22 December 2015

Keywords:
Multiple image mosaicking
Seamline detection
Graph cuts
Image parallax
Street-view panorama
Aerial images

ABSTRACT

While mosaicking images, especially captured from the scenes of large depth differences with respective to cameras at varying locations, the detection of seamlines within overlap regions is a key issue for creating seamless and pleasant image mosaics. In this paper, we propose a novel algorithm to efficiently detect optimal seamlines for mosaicking aerial images captured from different viewpoints and for mosaicking street-view panoramic images without a precisely common center in a graph cuts energy minimization framework. To effectively ensure that the seamlines are optimally detected in the laterally continuous regions with high image similarity and low object dislocation to magnificently conceal the parallax between images, we fuse the information of image color, gradient magnitude, and texture complexity into the data and smooth energy terms in graph cuts. Different from the traditional *frame-to-frame optimization* for sequentially detecting seamlines for mosaicking multiple images, our method applies a novel *multi-frame joint optimization* strategy to find seamlines within multi-overlapped images at one time. In addition, we propose simple but effective strategies to semi-automatically guide the seamlines by exploiting simple human–computer interaction strongly constraining the image regions that the seamlines will or won't pass through, which is often ignored by many existing seamline detection methods. Experimental results on a large set of aerial, oblique and street-view panoramic images show that the proposed method is capable of creating high-quality seamlines for multiple image mosaicking, while not crossing majority of visually obvious foreground objects and most of overlap regions with low image similarity to effectively conceal the image parallax at different extents.

1. Introduction

Image mosaicking is an important and classical problem in the fields of photogrammetry (Pan et al., 2009, 2014; Chon et al., 2010; Yu et al., 2012; Wan et al., 2013; Pan and Wang, 2011; Mills and McLeod, 2013; Du et al., 2008), remote sensing (Kerschner, 2001; Soille, 2006; Yang et al., 2011; Helmer and Ruefenacht, 2005) and computer vision (Gracias et al., 2009; Agarwala et al., 2004; Uyttendaele et al., 2001; Levin et al., 2004; Xiong and Pulli, 2010; Brown and Lowe, 2007), which is used to merge a set of images geometrically aligned as precisely as possible into a single composite image as seamlessly as possible. In ideally static scenes in which both the photometric inconsistencies and the geometric misalignments are not existed or not obviously visible in overlap regions, the mosaicked image looks perfect only when the geometric distance criterion is used. However, in some cases, especially for street-view panoramic images or oblique photographs, there exist photometric inconsistencies to different extents in overlap regions between images due to illumination variations and different exposure settings, which can be solved very well by a series of color correction, smoothing transition (Levin et al., 2004; Xiong and Pulli, 2009) and image blending (Perez et al., 2003; Prados et al., 2014; Szeliski et al., 2011; Allene et al., 2008) techniques trying to conceal stitching artifacts by smoothing color differences between input images. In addition, since input images are captured without a precisely common projection center from the scenes with large depth differences with respective to cameras at the same time (e.g., street-view panoramic images captured from a mobile vehicle) or captured with different viewpoints from scenes at different times by a single camera (e.g., aerial or oblique images captured from an airborne platform), such these images cannot be precisely aligned in geometry. This results in that the geometric positions of

* Corresponding author.
E-mail addresses: jian.yao@whu.edu.cn (J. Yao), jshan@purdue.edu (J. Shan).
URL: http://cvrs.whu.edu.cn/ (J. Yao).

http://dx.doi.org/10.1016/j.isprsjprs.2015.12.007
0924-2716/

corresponding pixels from different images may be different. The image blending techniques can deal with the color differences along the seamlines but can't handle the obvious parallax caused by geometric misalignment. The efficient way to solve this problem is to detect the optimal seamlines avoiding crossing majority of visually obvious objects and most of overlap regions with low image similarity and large object dislocation. If the seamlines and stitching artifacts are still visible due to color differences, the image blending technique can be further applied to solve it easily. This paper focuses on optimal seamline detection despite the presence of large geometric misalignments for creating visually pleasant image mosaics.

Optimal seamline detection methods search for the seamlines in overlap regions between images where their intensity or gradient differences are minimal. Based on the optimally detected seamlines, multiple aligned images can be mosaicked into a composite image in which the obvious parallax caused by image misalignments can be magnificently concealed. Many methods (Kerschner, 2001; Pan et al., 2009; Chon et al., 2010; Yu et al., 2012; Wan et al., 2013; Pan and Wang, 2011; Mills and McLeod, 2013; Soille, 2006) regarded the optimal seamline detection as an energy optimization problem and solved it by minimizing a special energy function defined to represent the difference between the original images along the seamlines. For these methods, the key ideas concentrate on how to define effective energy functions and how to guarantee the optimality of the solution. The energy functions are often defined by considering color, gradient and texture, and are optimized via different optimization algorithms, e.g., snake model (Kass and Witkin, 1988), Dijkstra's algorithm (Dijkstra, 1959), dynamic programming (Bellman, 1957), and graph cuts (Boykov et al., 2001; Boykov and Kolmogorov, 2004).

Kerschner (2001) proposed an automated seamline detection method using twin snake based on the energy function defined on the similarity of color and texture. The energy optimization started from two snakes on the opposite borders of the overlap region, which moved closer during the energy minimization process, and the optimal seamline is found when two twin snakes coincided. But this algorithm requires a high computation cost and can't completely overcome the local minimum problem. Besides the snake model, the Dijkstra's algorithm was also popularly used for detecting the optimal seamlines. Chon et al. (2010) designed a novel objective function to evaluate the mismatching between two images based on the normalized cross correlation (NCC). Their proposed method first determined the desired level of maximum difference along the seamline and then applied the Dijkstra's algorithm to find the best seamline with the minimal objective function. Compared with the simple Dijkstra's algorithm, their method could find a longer seamline with less pixels with high energy costs. However, using the Dijkstra's algorithm for seamline detection is time consuming as well as needs to give the start and end points of the seamline. Wan et al. (2013) proposed a vector road-based seamline determination algorithm ensuring that the seamline follows the centerlines of wide streets and avoids to cross foreground objects. This algorithm firstly built a weighted graph by overlapping the extracted skeleton of the overlap regions and vector roads, and then found the lowest cost path between two intersections of adjacent image polygons by the Floyd–Warshall algorithm (Floyd, 1962). However, roads may not be completely and automatically extracted from aerial or satellite images. In addition, in the urban areas, the roads are often occluded by buildings and trees, and occupied by moving vehicles or pedestrians. Therefore, it is not the best choice for seamline detection when roads cannot be accurately figured out. Another alternative method is the dynamic-programming-based optimal seamline detection strategy, which is less time-consuming than the Dijkstra's algorithm. Yu et al. (2012) proposed to combine the pixel-based similarity defined by color, edge and texture information with the region-based saliency map based on human attention model to guide the optimal seamline searching process of the dynamic programming (DP) algorithm. However, the DP algorithm also needs to know the start and end points of the seamline. Compared with the above-mentioned energy optimization algorithms, the graph cuts (Boykov et al., 2001; Kolmogorov and Zabin, 2004) can effectively find the global solution with a less computation cost and doesn't require the start and end points of the seamline.

Nowadays, graph cuts has been popularly applied in many fields of computer vision, such as image segmentation (Rother et al., 2004; Boykov and Funka-Lea, 2006; Boykov and Jolly, 2001), stereo matching (Kolmogorov and Zabih, 2001; Hong and Chen, 2004; Wang and Lim, 2011) and recently in image mosaicking (Gracias et al., 2009; Kwatra et al., 2003; Agarwala et al., 2004). For example, Rother et al. (2004) provided a powerful semi-automatic algorithm for foreground object extraction based on iterated graph cuts, named as "GrabCut". Kolmogorov and Zabih (2001) presented an energy minimization formulation of the correspondence problem with occlusions, and provided a fast approximation algorithm based on graph cuts. In the filed of image mosaicking, Kwatra et al. (2003) first applied the graph cuts algorithm to find the seamline for image and video synthesis. Their proposed method defined an energy function based on the difference of color intensities and gradient magnitudes along horizontal and vertical directions and used graph cuts to find optimal seamlines between each patch region. Agarwala et al. (2004) provided a framework to easily create a single composite image by using graph cuts to choose good seamlines within the constituent images, which needs an intuitive human–computer interaction for defining local and global objectives. Gracias et al. (2009) combined the watershed segmentation and the graph cuts algorithm to detect the optimal seamlines. Their algorithm began with creating a set of watershed segments on the difference image of overlap regions followed by finding the solution via graph cuts between those segments instead of the entire set of pixels. However, it only considered the intensity difference when computing the cost of each pixel. As a result, this difference image has lost some necessary information for image segmentation.

All the existing graph-cuts-based optimal seamline detection algorithms only focused on panoramic images and have not been tested on aerial images. In this paper, we formulate a unified graph cuts minimization framework for mosaicking aerial or street-view panoramic images with geometrical misalignments. To obtain high-quality seamlines, not only both color intensities and gradient magnitudes are considered in the energy cost of each pixel but also the texture complexity inspired by the HOG (Histogram of Oriented Gradient) feature descriptors (Dalal and Triggs, 2005) was integrated into the used energy cost function. This then can suppress the large image differences in specific regions such as roads and woodlands suitable to be passed through by seamlines. To quickly find optimal seamlines among multiple images for mosaicking, the traditional strategy, named as the *frame-to-frame optimization*, is to select one image as the reference one to which other images are sequentially merged based on the optimal seamline between the current composite image and the newly introduced one. To effectively find optimal seamlines from multiple aligned images with multi-overlapped regions, we propose to directly apply the multi-label graph cuts energy minimization optimization on all the images to find the optimal solution, which is called as the *multi-frame joint optimization* strategy. In addition, to guarantee that the seamlines won't pass through poor image regions with obvious parallax, we provide simple but effective strategies to semi-automatically guide the seamlines by exploiting a human–computer interaction strongly constraining the image

regions that the seamlines will or won't pass through, which is often ignored by many existing seamline detection methods.

The remaining part of this paper is organized as follows. Section 2 introduces the graph cuts energy minimization framework for finding optimal seamline between two aerial or street-view panoramic images, consisting of a brief introduction of graph cuts, the basic energy definitions, and the labeling optimization. Two strategies for optimal seamline detection from multiple images are detailed in Section 3. The human–computer interaction strategies for semi-automatically finding optimal seamlines are presented in Section 4. Experimental results on challenging street-view panoramic and aerial images are presented in Section 5 followed by the conclusions drawn in Section 6.

2. Two-image seamline detection

In overlap regions between two images, we compute the energy cost of each pixel by considering the color intensities, gradient magnitudes and texture complexities. We regard each pixel as a node in the graph and assume that each node has four cardinal neighbors in the 4-neighborhood. The link between two adjacent nodes is regarded as an edge in the graph and its weight cost is defined as the sum of energy costs of these two nodes. The graph cuts algorithm is used to associate the label of each pixel to one of the input source images with the minimum energy cost.

2.1. Graph cuts

Before giving a detailed description of our algorithm, we briefly review the graph cuts optimization algorithm. In computer vision, many problems are naturally stated in terms of energy minimization and can be regarded as a labeling optimization problem. The graph cuts method introduced by Boykov et al. (2001) has been quickly used in many applications of computer vision. The basic idea is to first construct a weighted graph where each edge weight cost represents the value of the corresponding energy cost function, and then to find the minimum cut in this graph based on the max-flow or min-cut algorithm (Boykov and Kolmogorov, 2004). Let $\mathcal{P}$ be the set of all elements (i.e., nodes in the graph), $\mathcal{N}(\mathcal{P})$ be the set of all element pairs in the neighborhood (i.e., edges in the graph), and $\mathcal{L}$ be the set of all labels. The objective is to find a labeling map f assigning a label $f_p \in \mathcal{L}$ to each element $p \in \mathcal{P}$ by minimizing the following energy function (Boykov et al., 2001):

$$E(f) = \sum_{p\in\mathcal{P}} D_p(f_p) + \sum_{(p,q)\in\mathcal{N}(\mathcal{P})} V_{p,q}(f_p, f_q), \tag{1}$$

where $D_p(f_p)$ denotes the cost of assigning the label f_p to the element p and $V_{p,q}(f_p, f_q)$ defines the cost of assigning the labels f_p and f_q to the element pair p and q, respectively, which are often called as *the data energy term* and *the smooth energy term*, respectively. If the labels f_p and f_q are the same, the value of the smooth cost $V_{p,q}(f_p, f_q)$ would be equal to 0.

2.2. Energy definition

We assume that all input source images have been geometrically aligned as precisely as possible. However, there always exist geometric misalignments between these images at different extents due to that they are captured from the scenes of large depth differences by a single camera with different viewpoints or by multiple cameras at varying locations. To describe the definition of the energy cost function clearly, we first consider the simple case for a composite image $\mathcal{I}$ from two images $\mathbf{I}_p$ and $\mathbf{I}_q$ with an overlap. The energy cost $C(\mathbf{x})$ of the pixel $\mathbf{x} = (x, y)$ in $\mathcal{I}$ is comprised of three terms: the color difference term $C_c(\mathbf{x})$, the gradient magnitude term $C_g(\mathbf{x})$ and the texture complexity term $C_t(\mathbf{x})$, i.e.,

$$C(\mathbf{x}) = \big(C_c(\mathbf{x}) + C_g(\mathbf{x})\big) \times C_t(\mathbf{x}). \tag{2}$$

The color difference is often applied in the energy cost function to detect the optimal seamlines for image mosaicking. However, in our algorithm, the color difference for the pixel $\mathbf{x}$ of $\mathcal{I}$ in overlap regions is computed in the HSV (Hue, Saturation, Value) color space rather than in the popularly used RGB color space or the grayscale space, which is defined as:

$$C_c(\mathbf{x}) = w_v |V_p(\mathbf{x}) - V_q(\mathbf{x})| + (1 - w_v)|S_p(\mathbf{x}) - S_q(\mathbf{x})|, \tag{3}$$

where $V_p(\mathbf{x})$ and $S_p(\mathbf{x})$ denote the intensity values of V and S channels of $\mathbf{x}$ in $\mathbf{I}_p$, respectively, and there are the same meanings for $V_q(\mathbf{x})$ and $S_q(\mathbf{x})$. The weight coefficient $w_v \in [0, 1]$ is used to balance the influence of the differences at the V and S channels, which was set as $w_v = 0.95$ in this paper.

The image parallaxes within the overlap regions with large gradient magnitudes are more salient in vision. To ensure that the seamlines can round such these image regions, we introduce another energy term $C_g(\mathbf{x})$ for the pixel $\mathbf{x}$ based on the gradient magnitudes as well as their absolute differences, i.e.,

$$\begin{aligned} C_g(\mathbf{x}) = \frac{1}{4}\Big(&\big|G_p^x(\mathbf{x})\big| + \big|G_q^x(\mathbf{x})\big| + \big|G_p^y(\mathbf{x})\big| + \big|G_q^y(\mathbf{x})\big|\Big) \\ &+ \big|G_p^x(\mathbf{x}) - G_q^x(\mathbf{x})\big| + \big|G_p^y(\mathbf{x}) - G_q^y(\mathbf{x})\big|, \end{aligned} \tag{4}$$

where $G_p^x(\mathbf{x})$ and $G_p^y(\mathbf{x})$ denote the horizontal and vertical gradient magnitudes of $\mathbf{x}$ in $\mathbf{I}_p$, respectively, which are calculated using the Sobel operator in the grayscale space, and there are the same meanings for $G_q^x(\mathbf{x})$ and $G_q^y(\mathbf{x})$.

Based on the above defined two energy costs in term of color differences and gradient magnitudes, the optimally found seamlines should ensure that the differences along the seamlines are minimal and try to avoid passing through visually obvious foreground objects. However, sometimes, some specific regions such as roads, sky and woodlands are more suitable to be located in the seamlines as the image differences in these regions are not easy to be observed although both color differences and gradient magnitudes may be large. To solve this problem, we propose a new texture complexity measurement to distinguish those regions stemming from the HOG feature descriptors (Dalal and Triggs, 2005), which has been widely and successfully used in computer vision and image processing for the purpose of object detection, such as human detection (Dalal and Triggs, 2005; Zhu et al., 2006). The gradient orientation $O(\mathbf{x})$ of the pixel $\mathbf{x}$ is computed firstly as $O(\mathbf{x}) = \arctan\big(G^y(\mathbf{x})/G^x(\mathbf{x})\big)$ where $G^y(\mathbf{x})$ and $G^x(\mathbf{x})$ denote the gradient magnitude values of $\mathbf{x}$ in the vertical and horizontal directions, respectively. All the gradient orientations are converted into the range of $[0, 2\pi]$. Then, we compute the histogram of oriented gradients $\mathcal{H}(\mathbf{x})$ comprised of B bins over the $k \times k$ size window region $\mathcal{N}_{k\times k}(\mathbf{x})$ centered at the pixel $\mathbf{x}$ in an image $\mathbf{I}$ where $B = 12$ and $k = 11$ were used in this paper. Based on the histogram of oriented gradients, the texture complexity at the pixel $\mathbf{x}$ is defined as:

$$\begin{aligned} \Gamma(\mathbf{x}) &= 1 - \frac{\gamma + \sum_{b=1}^{B} \min\big(H_b(\mathbf{x}), \overline{H}(\mathbf{x})\big)}{\gamma + \sum_{b=1}^{B} H_b(\mathbf{x})} \\ &= \frac{\sum_{b=1}^{B} H_b(\mathbf{x}) - \sum_{b=1}^{B} \min\big(H_b(\mathbf{x}), \overline{H}(\mathbf{x})\big)}{\gamma + \sum_{b=1}^{B} H_b(\mathbf{x})}, \end{aligned} \tag{5}$$

where $H_b(\mathbf{x})$ denotes the frequency of the b-th bin in $\mathcal{H}(\mathbf{x})$ and $\overline{H}(\mathbf{x})$ represents the mean of frequencies of all bins, i.e., $\overline{H}(\mathbf{x}) = \frac{1}{B}\sum_{b=1}^{B} H_b(\mathbf{x})$. The parameter γ represents a predefined posi-

Fig. 1. An original aerial image (Left) and its normalized texture complexity maps with different values of δ ($\delta = 0, 4, 8$ from the second column to the last) where the lighter regions indicate higher texture complexities.

Fig. 2. The four typical regions (Top) in street-view panoramic images and their corresponding normalized texture complexity maps (Bottom) where the lighter regions indicate higher texture complexities.

tive constant avoiding the divide overflow in the case of $\overline{H}(\mathbf{x}) = 0$ and suppressing producing a high texture complexity value at the image regions with low gradient magnitudes but consistent gradient orientations, which is defined as:

$$\gamma = 4k^2 \times \delta, \tag{6}$$

where the constant "4" is the normalization factor of gradient magnitudes based on the Sobel operator, k denotes the window size used for calculating HOG, and δ is a user-controlled variable. An example illustrating the benefit of using γ for computing the texture complexity in Eq. (5) is presented in Fig. 1 where the image regions of roads and water surfaces in an aerial image are gradually suppressed with low texture complexity values with the increasing of the value of δ. Fig. 2 shows the normalized texture complexity maps of four typical regions (buildings, vehicles, roads, and trees) in street-view panoramic images with $\delta = 8$ used if not specifically stated in this paper. Obviously, if the pixel $\mathbf{x}$ is located in the local region with poor texture like sky or strongly repetitive patterns like roads or woodlands, the frequencies of different bins in the histogram is approximately equal, so $\Gamma(\mathbf{x})$ is small and close to 0. In contrast, $\Gamma(\mathbf{x})$ is large and close to 1 if the frequencies of few bins are high and the rest are low.

Based on the definition of the proposed texture complexity, the texture complexity cost term $C_t(\mathbf{x})$ for the pixel $\mathbf{x}$ in overlap regions of $\mathcal{I}$ is defined as:

$$C_t(\mathbf{x}) = \Gamma_p(\mathbf{x}) + \Gamma_q(\mathbf{x}), \tag{7}$$

where $\Gamma_p(\mathbf{x})$ and $\Gamma_q(\mathbf{x})$ represent the texture complexity values of $\mathbf{x}$ in $\mathbf{I}_p$ and $\mathbf{I}_q$, respectively. In this way, we can apply $C_t(\mathbf{x})$ to constrain the color differences and gradient magnitudes in the regions with poor texture or strongly repetitive patterns without affecting other foreground object regions with rich and strong edges as used in Eq. (2).

2.3. Labeling optimization

We formulate the optimal seamline detection as an energy minimization problem and use graph cuts to find the solution, as the two-label graph cuts optimization example shown in Fig. 3. The energy cost $E(\mathcal{I})$ of the composite image $\mathcal{I}$ from two geometrically aligned images $\mathbf{I}_p$ and $\mathbf{I}_q$ is comprised of the data energy term $E_{data}(\mathcal{I})$ and the smooth energy term $E_{smooth}(\mathcal{I})$, i.e.,

$$E(\mathcal{I}) = E_{data}(\mathcal{I}) + E_{smooth}(\mathcal{I}). \tag{8}$$

The data energy term $E_{data}(\mathcal{I})$ represents all energy costs for associating each pixel in one of two input images $\mathbf{I}_p$ and $\mathbf{I}_q$, i.e.,

$$E_{data}(\mathcal{I}) = E_{data}(\mathbf{I}_p) + E_{data}(\mathbf{I}_q), \tag{9}$$

where $E_{data}(\mathbf{I}_i), i = p$ or q, denotes the sum of energy costs of assigning pixels in the composite image $\mathcal{I}$ with the label $\mathcal{L}(\mathbf{I}_i)$ of the image $\mathbf{I}_i$, i.e.,

$$E_{data}(\mathbf{I}_i) = \sum_{\mathbf{x}\in\mathcal{I}} D_l^i(\mathbf{x}), \tag{10}$$

where $\mathbf{x}$ denotes one pixel belonging to $\mathcal{I}$ and $D_l^i(\mathbf{x})$ represents the energy cost of assigning $\mathbf{x}$ with the label $\mathcal{L}(\mathbf{I}_i)$ of the image $\mathbf{I}_i$, which is defined as $D_l^i(\mathbf{x}) = 0$ if $\mathbf{x} \in \mathbf{I}_i$, i.e., the labels $\mathcal{L}(\mathbf{x}) = \mathcal{L}(\mathbf{I}_i)$, otherwise $D_l^i(\mathbf{x}) = \infty$ if $\mathbf{x} \notin \mathbf{I}_i$, i.e., the labels $\mathcal{L}(\mathbf{x}) \neq \mathcal{L}(\mathbf{I}_i)$. According to the above definition, for each pixel $\mathbf{x}$ in $\mathcal{I}$, its data energy cost only depends on whether it is inside the valid region of one image.

In Section 2.2, we have introduced how to compute the energy cost for each pixel on the image pair $\mathbf{I}_p$ and $\mathbf{I}_q$ with an overlap. The smooth energy item is used to punish assigning different labels to adjacent pixels. So, we define the smooth energy cost $E_{smooth}(\mathbf{x}, \mathbf{y})$ for two adjacent pixels $\mathbf{x}$ and $\mathbf{y}$ in the composite image $\mathcal{I}$ as the sum of their individual energy costs as follow:

$$E_{smooth}(\mathbf{x}, \mathbf{y}) = C(\mathbf{x}) + C(\mathbf{y}), \tag{11}$$

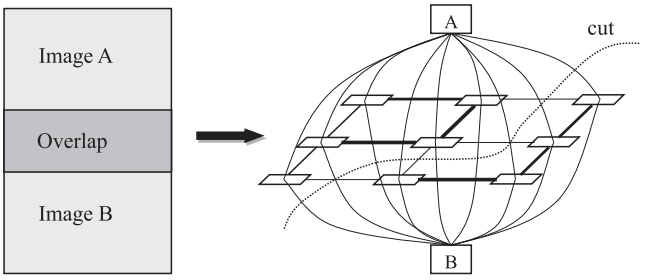

Fig. 3. An illustration example of detecting optimal seamline between two images via graph cuts. The thickness of lines between adjacent pixels represents the value of the energy cost and the "cut" denotes the minimum cut, which means the optimal seamline.

where $C(\mathbf{x})$ and $C(\mathbf{y})$ are the energy costs of the pixels $\mathbf{x}$ and $\mathbf{y}$ defined in Eq. (2). So, the smooth energy term $E_{smooth}(\mathcal{I})$ can be defined as:

$$E_{smooth}(\mathcal{I}) = \sum_{(\mathbf{x},\mathbf{y})\in\mathcal{N}(\mathcal{I})} \sigma(\mathcal{L}(\mathbf{x}),\mathcal{L}(\mathbf{y})) \times E_{smooth}(\mathbf{x},\mathbf{y}), \tag{12}$$

where $\mathcal{N}(\mathcal{I})$ denotes the set of all adjacent pixel pairs in the 4-neighborhood, and $\mathcal{L}(\mathbf{x})$ and $\mathcal{L}(\mathbf{y})$ represent the labels of the adjacent pixels $\mathbf{x}$ and $\mathbf{y}$, respectively. The function $\sigma(\mathcal{L}(\mathbf{x}),\mathcal{L}(\mathbf{y}))$ is a binary logic function defined as $\sigma(\mathcal{L}(\mathbf{x}),\mathcal{L}(\mathbf{y})) = 0$ if $\mathcal{L}(\mathbf{x}) = \mathcal{L}(\mathbf{y})$ otherwise $\sigma(\mathcal{L}(\mathbf{x}),\mathcal{L}(\mathbf{y})) = 1$ if $\mathcal{L}(\mathbf{x}) \neq \mathcal{L}(\mathbf{y})$.

Now, all energy functions have been defined as those used in the standard graph cuts energy minimization framework described in Section 2.1. Therefore, we can directly apply the graph cuts algorithm to find the optimal solution of the seamline in the overlap image region between two geometrically aligned images. To speed up the optimization process, we can decrease the number of elements used in graph cuts by only considering all the pixels in the overlap regions between images as the valid elements in the graph because only the labels of those pixels in overlap regions need to be optimally determined while the labels of other pixels can be directly set as the labels of their own corresponding images.

3. Multi-image seamline detection

The optimal seamline detection algorithm via graph cuts for a pair of images with an overlap has been clearly stated in Section 2. In practical applications, we need to create an image mosaic from multiple images. If there does not exist any overlap region formed by more than two images, i.e., all the overlap image regions are two-overlapped, the seamlines can be independently and separately found via graph cuts from the pairs of overlapped images. However, there often exist multi-overlapped regions in multiple street-view panoramic images blended into a 360°-view panorama and in multi-strip aerial images mosaicked into a composite wide-angle image. In this paper, we propose a novel multi-frame joint optimization strategy to effectively deal with such this case of multi-overlapped images via multi-label graph cuts, as illustrated in Fig. 4(b).

3.1. Frame-to-frame optimization

To efficiently find optimal seamlines among multiple geometrically aligned images for mosaicking, the traditional strategy is to select one image as the reference one to which other images are sequentially merged based on the optimal seamline between the current composite image and the newly introduced one. In this traditional optimization strategy, the seamline is always optimally found just between two images, named as the *frame-to-frame optimization* strategy. Many optimal seamline detection algorithms (Gracias et al., 2009; Xiong and Pulli, 2010; Wan et al., 2013) apply this strategy when dealing with multiple images. An illustration example is shown in Fig. 4(a) where there are three input images **A**, **B** and **C** overlapped by each other. It first finds the optimal seamline between the images **A** and **B**, which is used to create a composite image **AB**, and then finds the optimal seamline between the newly introduced image **C** and the previously composite image **AB**. The key advantage of using this traditional optimization strategy is that only one overlap region between two images are considered in the graph cuts optimization, which can result in a low computation cost due to few number of elements and only two labels in each optimization (Gracias et al., 2009). Furthermore, it can be sped up in parallel. For example, six individual seamlines will be detected from six camera views for creating a 360°-view panorama, as described in experimental results reported in Section 5 by the frame-to-frame optimization strategy, which are comprised of five vertical seamlines from five adjacent horizontal camera view pairs and one horizontal seamline between the top camera view and the horizontal camera ones. First those five vertical seamlines can be separately detected in parallel, which are used to create a composite image from horizontal five camera views. Then the last horizontal seamline can be found between the top camera view and the horizontal composite image. However, the disadvantage of this traditional optimization strategy is that the seamlines in multi-overlapped regions can't be optimally detected in the global optimization sense.

3.2. Multi-frame joint optimization

To more effectively find globally optimal seamlines from multiple aligned images with multi-overlapped regions, we

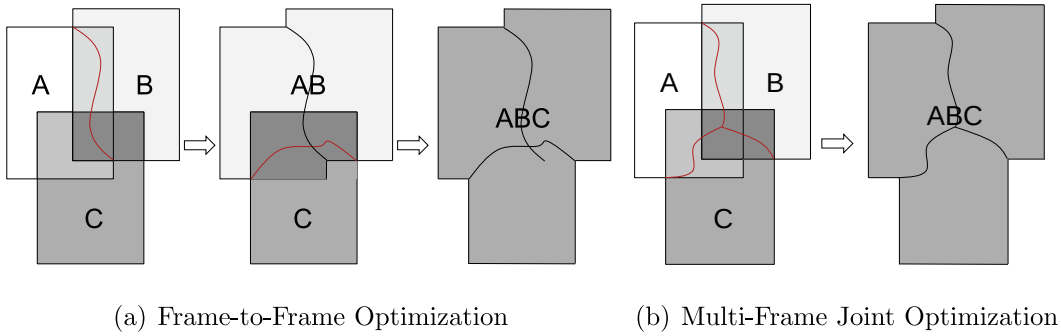

Fig. 4. An illustration of two different optimization strategies for detecting optimal seamlines from multi-overlapped regions among three geometrically aligned images **A**, **B** and **C**: (a) the traditional frame-to-frame optimization strategy; (b) the multi-frame joint optimization strategy. The red contours represent the currently optimized seamlines while the black ones denote the seamlines already found. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

propose to directly apply the multi-label graph cuts energy minimization optimization on all the images at one time to find the optimal solution, which is called as the *multi-frame joint optimization* strategy. An illustration example of this optimization strategy is shown in Fig. 4(b) where the seamlines among three images overlapped by each other are optimized via multi-label graph cuts at one time. Compared with the traditional optimization strategy, in which the detected seamlines may be varied while the different composite sequence is applied, the multi-frame joint optimization strategy can directly obtain the final image mosaic based on the seamlines globally optimized at one time. But, it needs a higher computation cost due to that the labels of multi-overlapped image regions will be optimized as one of multiple labels, instead of one of two labels in the frame-to-frame optimization, and it is difficult for implementing an efficient parallel optimization for mosaicking a large set of images.

To apply the multi-label graph cuts for finding the seamlines from a set of N geometrically aligned images $\{\mathbf{I}_i\}_{i=1}^{N}$ at one time, we need to accordingly modify the definitions of the energy functions. The data energy term will be simply modified as:

$$E_{data}(\mathcal{I}) = \sum_{i=1}^{N} E_{data}(\mathbf{I}_i), \tag{13}$$

where $E_{data}(\mathbf{I}_i)$ is computed as the same in Eq. (10). The smooth energy cost of two adjacent pixels $\mathbf{x}$ and $\mathbf{y}$ in the final composite image $\mathcal{I}$ is re-defined as:

$$E_{smooth}(\mathbf{x},\mathbf{y}) = \max_{(\mathbf{I}_p,\mathbf{I}_q)\in\mathcal{P}(\mathcal{I})} E^{(p,q)}_{smooth}(\mathbf{x},\mathbf{y}), \tag{14}$$

where $\mathcal{P}(\mathcal{I})$ represents the set of all pairs of images with overlap regions, $(\mathbf{I}_p,\mathbf{I}_q)$ denotes a pair of images with an overlap, and $E^{(p,q)}_{smooth}(\mathbf{x},\mathbf{y})$ denotes the smooth energy cost of the pixel pair $(\mathbf{x},\mathbf{y})$ with respective to the image pair $(\mathbf{I}_p,\mathbf{I}_q)$ which is calculated as the same in Eq. (11). The total smooth energy term of the whole composite image $\mathcal{I}$ from multiple images is still calculated as the same in Eq. (12). With these energy modifications, all the seamlines can be optimally found via multi-label graph cuts at one time.

4. Seamline detection via human–computer interaction

Inevitably, like any other seamline detection algorithm, ours cannot guarantee that the detected seamlines could meet all users' requirements in all the overlap regions. Thus, a simple human–computer interaction is needed as a complement. To make end-users flexibly and quickly guide the seamlines passing through the regions they expect, we design the following human–computer interaction ways.

Firstly, the end-users can clearly mark some image regions in the final composite image that the seamlines won't pass through, as illustrated in Fig. 5(a). It can be solved by modifying the smooth energy cost of each adjacent pixel pair $(\mathbf{x},\mathbf{y})$ in those image regions as $E_{smooth}(\mathbf{x},\mathbf{y}) = \infty$.

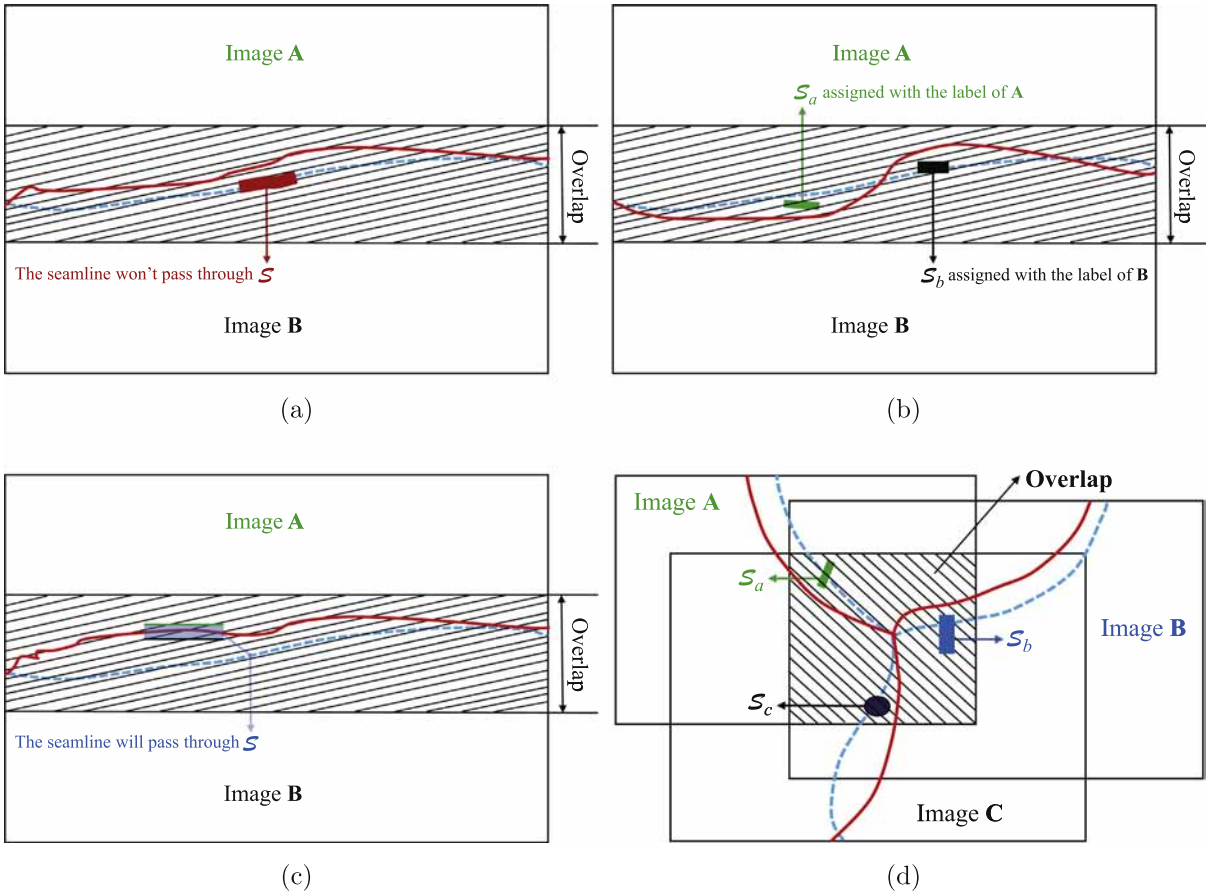

Fig. 5. Illustrative examples for human–computer interaction: (a) the seamline won't pass through the marked image region $\mathcal{S}$; (b) the labels of pixels in $\mathcal{S}_a$ and $\mathcal{S}_b$ are manually assigned with the labels $\mathcal{L}(\mathbf{A})$ and $\mathcal{L}(\mathbf{B})$ of the images **A** and **B**, respectively; (c) the seamline will horizontally pass through the marked image region $\mathcal{S}$; (d) the labels of pixels in $\mathcal{S}_a, \mathcal{S}_b$ and $\mathcal{S}_c$ are manually assigned with the labels $\mathcal{L}(\mathbf{A}), \mathcal{L}(\mathbf{B})$ and $\mathcal{L}(\mathbf{C})$ of the images **A**, **B** and **C**, respectively. The blue dash curves in blue represent the original detected seamlines while the red solid curves denote the optimized seamlines based on the human–computer interaction. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

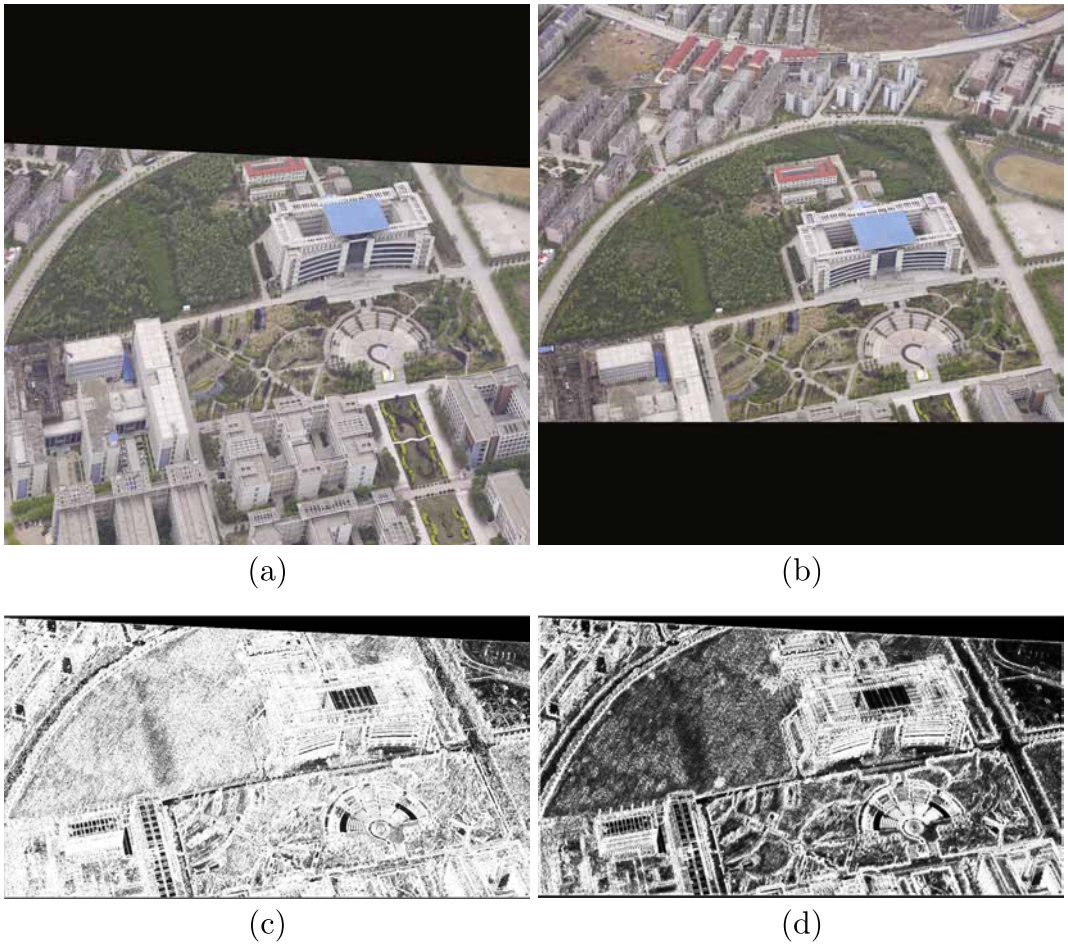

Fig. 6. The two geometrically aligned UAV oblique images in (a)–(b) and the normalized energy cost maps for their overlap region without the use of the texture complexity energy term in (c) and with its use in (d) where the brighter regions indicate higher energy costs, i.e., the larger image differences.

Secondly, the end-users can assign the pixels in some image regions with known image labels, as illustrated in Fig. 5(b) for two-overlapped images and in Fig. 5(d) for three-overlapped images, which can be solved by modifying the data energy costs of those pixels as follows. Let $\mathcal{S} = \{\mathbf{x}_k\}_{k=1}^{K}$ be the set of K selected pixels and $\mathcal{L}(\mathbf{x}_k)$ be the manually assigned label for the selected pixel $\mathbf{x}_k \in \mathcal{S}$. The data energy cost of each selected pixel $\mathbf{x}_k \in \mathcal{S}$ is modified as:

$$D_l^i(\mathbf{x}_k) = \begin{cases} 0, & \mathcal{L}(\mathbf{x}_k) = \mathcal{L}(\mathbf{I}_i), \mathbf{x}_k \in \mathbf{I}_i, \\ \infty, & \mathcal{L}(\mathbf{x}_k) \neq \mathcal{L}(\mathbf{I}_i), \mathbf{x}_k \notin \mathbf{I}_i, \end{cases} \quad i = 1, 2, \ldots, N. \tag{15}$$

With such these modifications, the graph cuts optimization is applied again to obtain the expected seamlines.

Sometimes, the end-users expect that the seamlines will pass through some specific region, like roads or green lands in urban aerial image mosaicking, as illustrated in Fig. 5(c). To achieve this goal, we can assign two specifically different image labels for the left and right boundary points while expected to be passed through vertically, or for the up and down boundary points while expected to be passed through horizontally.

5. Experiments and evaluation

Extensive experiments on representative street-view panoramic, aerial, and oblique images were conducted to comprehensively evaluate the performance of our proposed optimal seamline detection algorithm for creating pleasant image mosaics. To greatly speed up the optimization, we simply down-scaled the input images with a scale factor $S = 0.2$ for computing the energy costs in graph cuts while the detected seamlines were drawn in the original scaled images. Our algorithm was implemented with C++ under Windows and tested in a computer with an Intel Core i7-4770 at 3.4 GHz and the 16 GB RAM memory.

5.1. Texture complexity energy

In this section, we conducted two experiments on two groups of oblique images, captured by the Unmanned Aerial Vehicle (UAV) platform, to prove the effectiveness and superiority of our proposed texture complexity energy term described in Section 2.2. Each group is comprised of three images with a size of 7160×5406 pixels. These input oblique images were first geometrically aligned as precisely as possible into a common coordinate system with the homographic model. Fig. 6 shows two geometrically aligned images (see Fig. 6(a) and (b)) and their normalized energy cost maps for the overlap image region, computed without the use of the texture complexity energy term in Eq. (2) (i.e., $C_t(\mathbf{x}) = 1$) (see Fig. 6(c)) and with its use (see Fig. 6(d)). From Fig. 6(d), we observed that the energy costs of some specific image regions covered by roads and woodlands were greatly decreased when the texture complexity energy term was considered, which has achieved our expected objective.

The seamline detection results without and with the use of the texture complexity energy term in the first group of three oblique

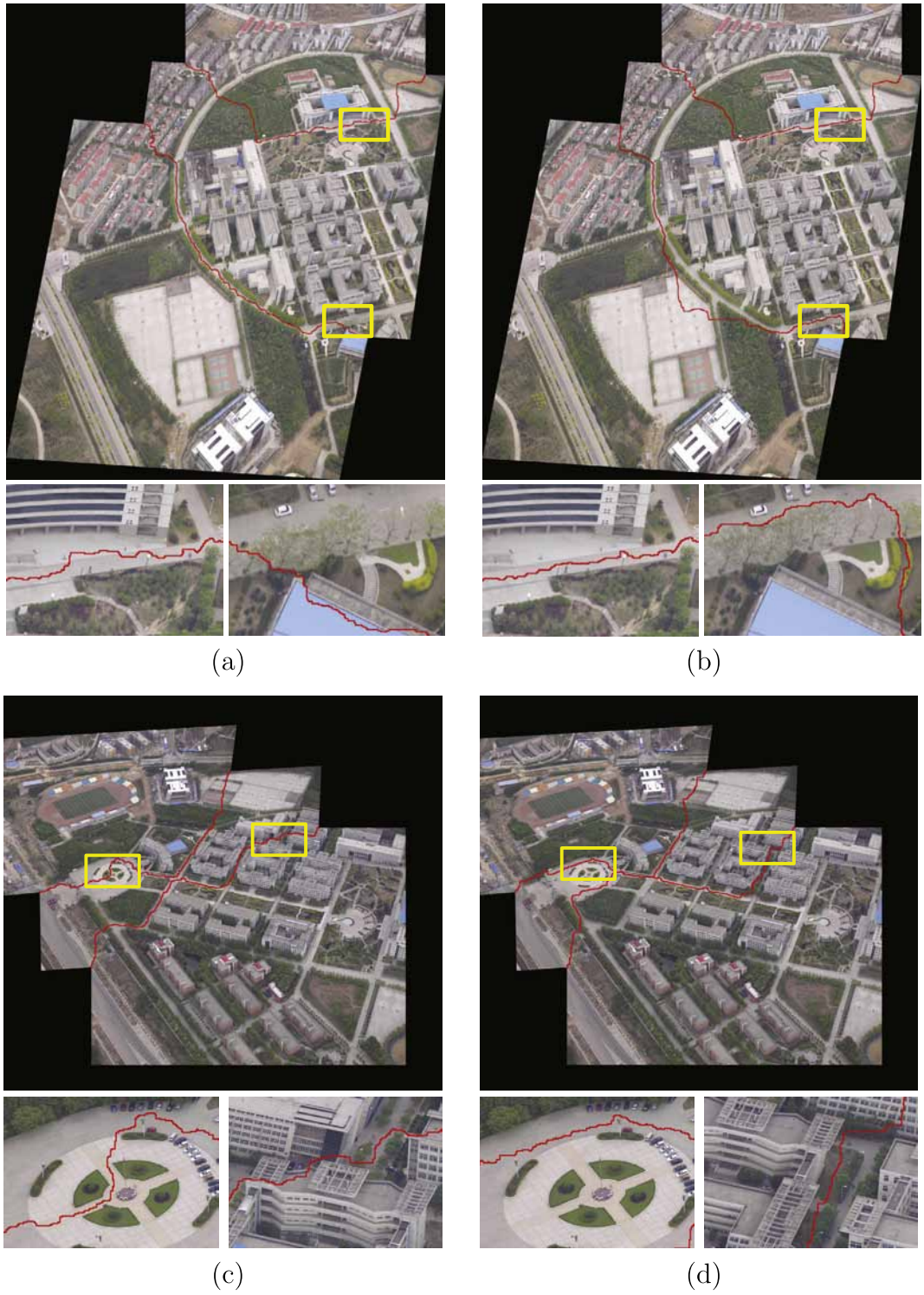

Fig. 7. Visual comparison of the seamline detection results in two groups of three geometrically aligned oblique images when the texture complexity energy term was used (Right: (b) and (d)) or not (Left: (a) and (c)).

images are shown in Fig. 7(a) and (b). From the whole seamline detection results and especially the detailed local regions shown in Fig. 7(a) and (b), we observed that the seamline, as shown in Fig. 7(b), detected with the use of the texture complexity energy term is much better than the seamline, as shown in Fig. 7(a), detected without its use. Noticeably, the seamline successfully rounded some buildings as expected when the texture complexity energy term was considered, which more possibly constrains the seamline to pass through image regions covered by roads and woodlands. While only considering the color difference and gradient magnitude energy terms, the seamline passed through some buildings instead of the nearby grassland, as shown in Fig. 7(a), due to that the color differences and gradient magnitudes in these image regions covered by the buildings and the nearby grassland are relatively large in the same level. In addition, with the use of the texture complexity energy term, the seamline in the image

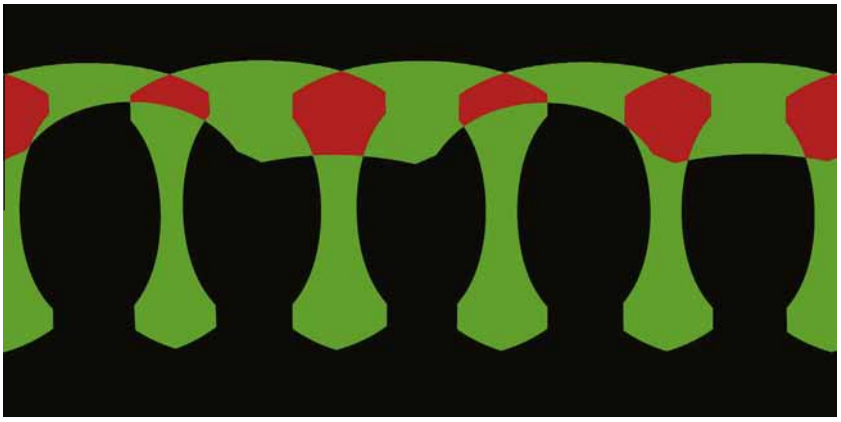

Fig. 8. The image overlap regions of six geometrically aligned and warped images in the 360° street-view panoramic view where the black, the green and the red represent the no-overlapped, two-overlapped, multi-overlapped image regions, respectively. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

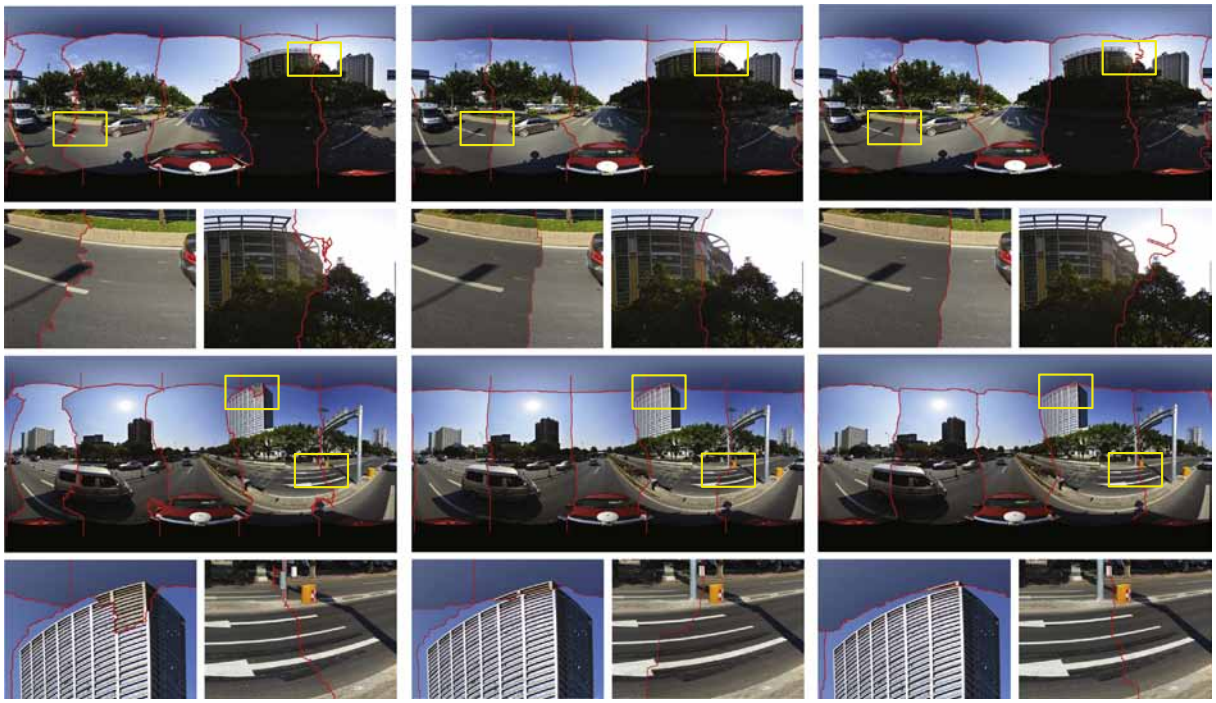

Fig. 9. The seamline detection results via *Enblend* (Left), the traditional frame-to-frame optimization (Middle) and the multi-frame joint optimization (Right).

regions covered by roads was better detected out. In the aspect of computational cost, without the use of the texture complexity energy term, our algorithm took around 86.25 s in the above experiment consisting of all the elapsed times in energy computation and graph cuts optimization. However, with its use, our algorithm only took around 62.59 s, although the energy computation time has increased due to the use of texture complexity term. This is mainly because the optimal solution can be more easily found via graph cuts when the texture complexity energy term is used.

The experimental results on another group of three geometrically aligned oblique images are presented in Fig. 7(c) and (d) and the similar conclusions can be drawn. The computational times of our algorithm without the use of the texture complexity and with its use are 34.78 and 32.69 s, respectively.

5.2. Multi-frame joint optimization

To evaluate the performance of our proposed multi-frame joint optimization strategy for multi-overlapped images, we selected two sets of street-view panoramic images for testing. A 360°-view panorama was obtained by mosaicking six images captured by an integrated multi-camera equipment with six Nikon D7100 cameras of 24 million pixels with wide-angle lenses mounted on a mobile vehicle platform. Six camera images were warped into a common coordinate system with the image size of $12,000 \times 6000$ pixels by a reliable geometrical alignment. As shown in Fig. 8, the overlap image regions in six street-view panoramic images warped from original camera ones are comprised of two-overlapped and multi-overlapped regions.

The comparison results on two groups of street-view panoramic images with the open-sourced software *Enblend*[1] which also applied graph cuts to find the optimal seamlines, the traditional frame-to-frame optimization strategy and the proposed multi-frame joint one are shown in Fig. 9. Via the traditional frame-to-frame optimization strategy, in total six individual seamlines were sequentially detected out, which are comprised of

[1] Available at http://enblend.sourceforge.net/.

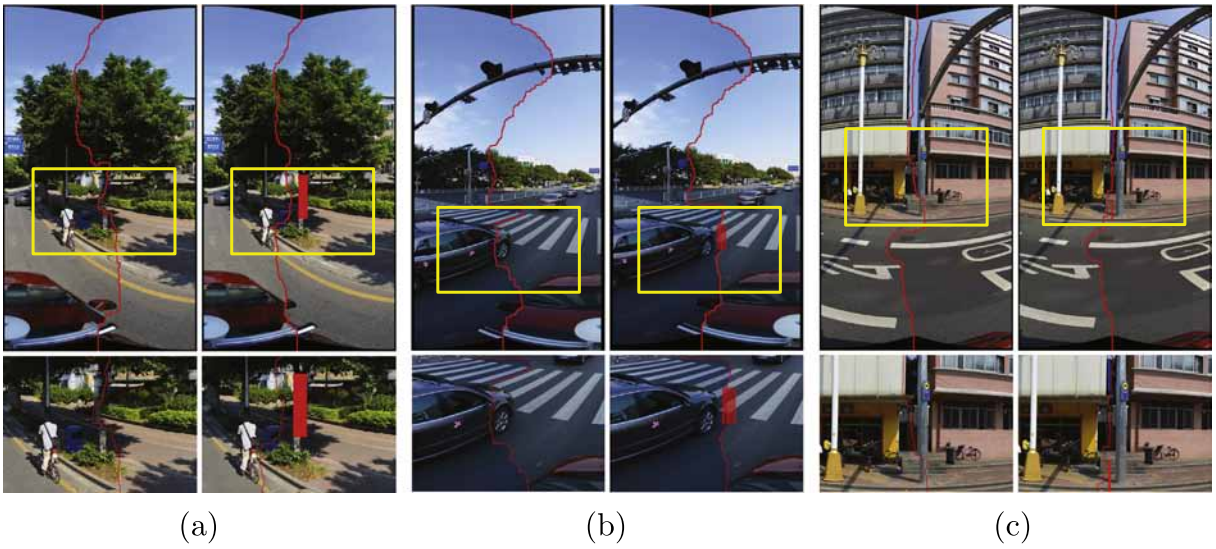

Fig. 10. Illustrative examples of semi-automatic seamline detection via human–computer interaction on three image pairs: (a) the seamline rounding the manually marked red box region; (b) the seamline passing through the manually marked red box region; (c) the seamline with known image labels for the manually marked regions, the blue box region is labeled as the left image and the red box region is labeled as the right image. The original seamline detection results are shown in the left and the re-optimized ones are shown in the right. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

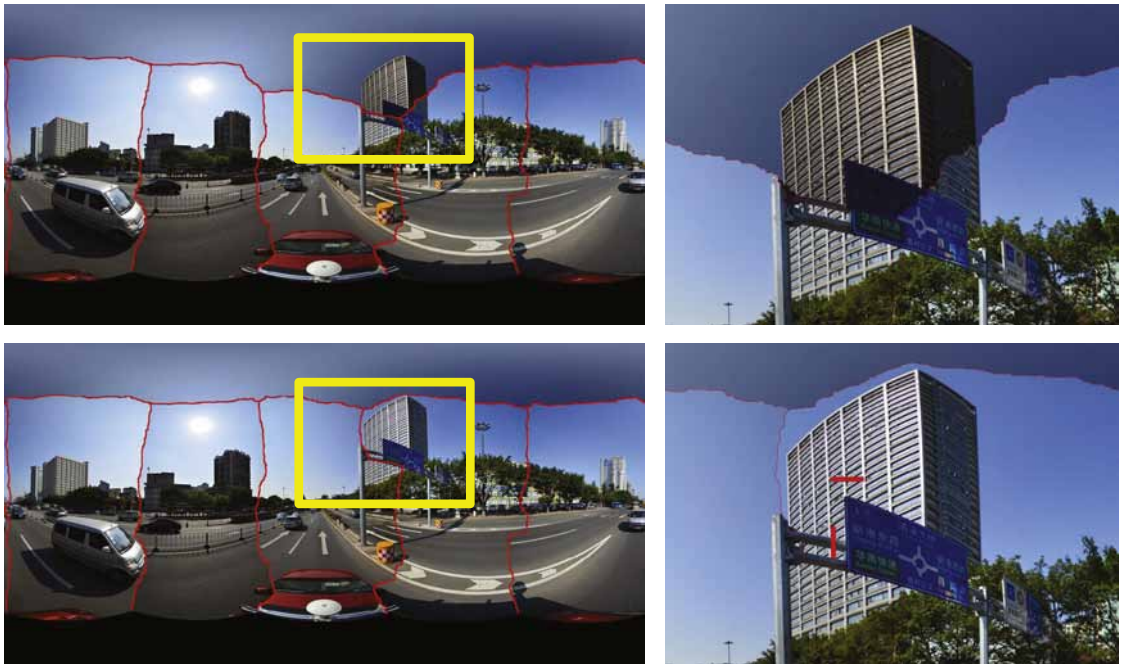

Fig. 11. The original seamline detection results (Top) and the re-optimized ones (Bottom) via simple human–computer interaction on multi-overlapped images.

five vertical seamlines independently found from five pairs of images in the horizontal direction and one horizontal seamline between the top camera image and the bottom composite image from five horizontal images, and *Enblend* also applied this strategy. In the multi-frame optimization strategy, all the seamlines were found at one time. From the results shown in Fig. 9, we observed that our proposed algorithms significantly outperformed the *Enblend*. At the same time, the seamlines produced by the multi-frame joint optimization are more reasonable than ones generated by the traditional frame-to-frame optimization due to that we have considered all image information in all overlap regions at one time, especially in multi-overlapped regions.

In the aspect of optimization efficiency, the multi-frame joint optimization in the above experiment averagely took 38.56 s on two groups of images, which is more than double of the elapsed time, 15.15 s on average, via the frame-to-frame optimization. If parallel optimization is applied, the elapsed running time via the frame-to-frame optimization can be greatly decreased.

5.3. Human–machine interaction

To illustrate how to guarantee that the seamlines will pass through or round some specific regions we expect by introducing simple human–computer interaction operations, we tested our proposed semi-automatic detection strategies on several groups of street-view panoramic images with two-overlapped and multi-overlapped regions. The original seamline detection results and the re-optimized ones under the guide of simple human–computer

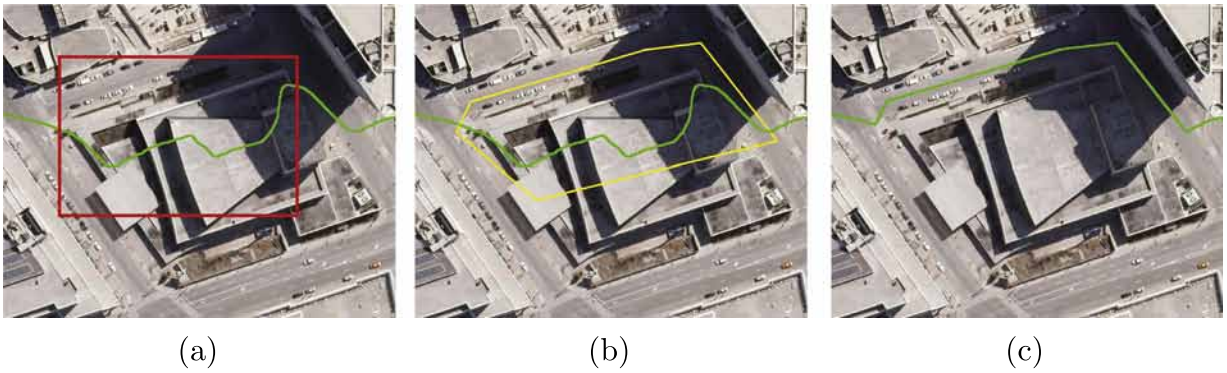

Fig. 12. An illustrative example of seamline editor via *OrthoVista*: (a) the original seamline detected by *OrthoVista*, which passes through the building in the red box; (b) the manually drawn seam polygon in yellow; (c) the newly edited seamline rounding the building along the boundary of the manually drawn seam polygon. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

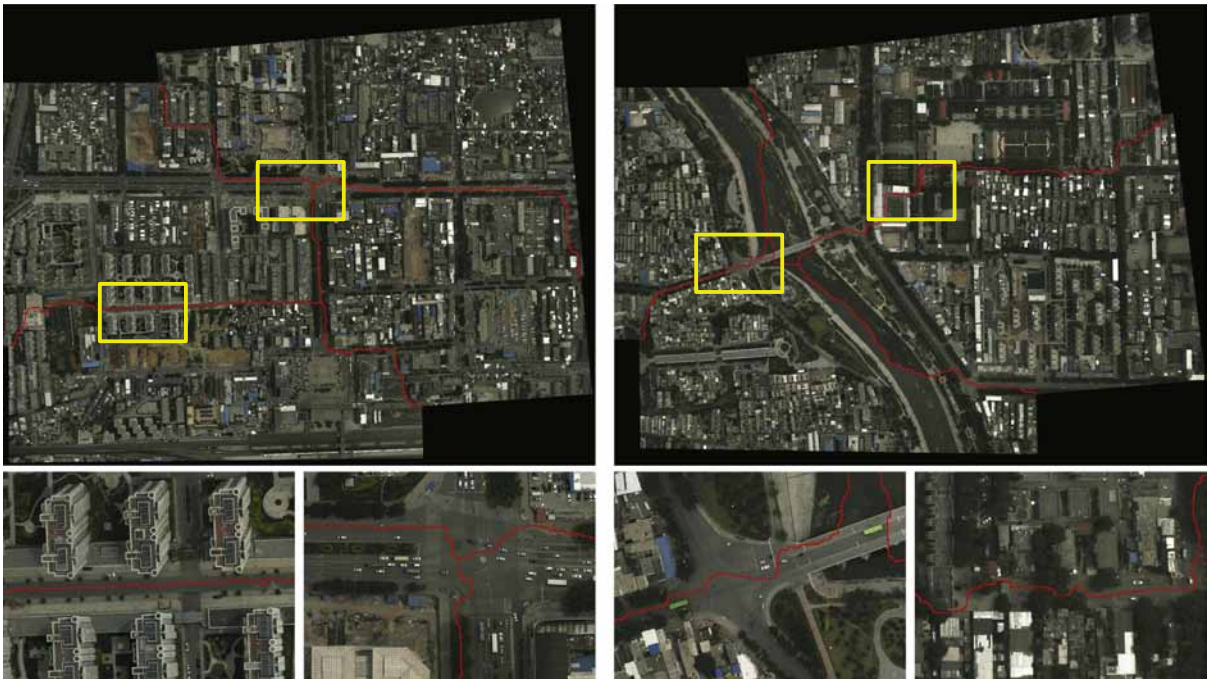

Fig. 13. The seamline results of two groups of aerial images captured in Anyang, China. Each group consists of four images selected from two adjacent strips.

interaction operations within two-overlapped regions of three image pairs are shown in Fig. 10. In Fig. 10(a), the original seamline passes through the tree trunk and the end-user manually marked a red box region on this tree trunk making the re-optimized seamline can round it based on the modified smooth energy costs for the adjacent pixels in the marked box region as described in Section 4. In Fig. 10(b), the original seamline cuts off the rear of a car due to the severe geometrical misalignment existed in the image regions close to the cameras whose projection centers are not precisely the same. In this example, the end-user strictly constrained the seamline passing through the red box manually marked on the roads to completely avoid cutting off the rear. In Fig. 10(c), the original seamline gets through the building and the pole, which was adjusted well by manually assigning the known image labels for the selected regions. A human–computer interaction illustrative example for multi-overlapped images via the multi-frame joint optimization is presented in Fig. 11 where the original seamlines pass through the buildings with obviously visible parallaxes, which were successfully rounded by manually labeling two red rectangle regions as a part of the right-down image and re-performing the multi-label graph cuts optimization based on the modified data energy costs for the pixels in the selected regions.

As we known, there exist many commercial tools for manually editing the seamlines, for example, the popularly used software *Trimble-Inpho OrthoVista*.[2] Fig. 12 presents an illustrative example of seamline editor via *OrthoVista*, from which we observed that the original seamline detected by *OrthoVista* passes through the building in the red box, as shown in Fig. 12(a). We used *Orthovista* to manually edit it by drawing a seam polygon in this region which represents the new seamline, as shown in Fig. 12(b). At last, the new edited seamline were achieved, which successfully rounds the building along the boundary of the manually drawn polygon, as shown in Fig. 12(c). This seamline editor method needs users to draw the new seamline aided by the seam polygon for each failed region, which is a

[2] Available at http://www.trimble.com/.

Table 1
Statistical results of the seamlines found in the Anyang aerial images.

	Seamline 1 (Fig. 13(a))	Seamline 2 (Fig. 13(b))	#Total
#Pixels	57,668	61,138	118,806
Numbers of obvious objects passed through	10 buildings	3 buildings and 1 bridge	13 buildings and 1 bridge
Visible seam length (in pixels)	1064	267	1331
Times (s)	106.47	111.28	217.75

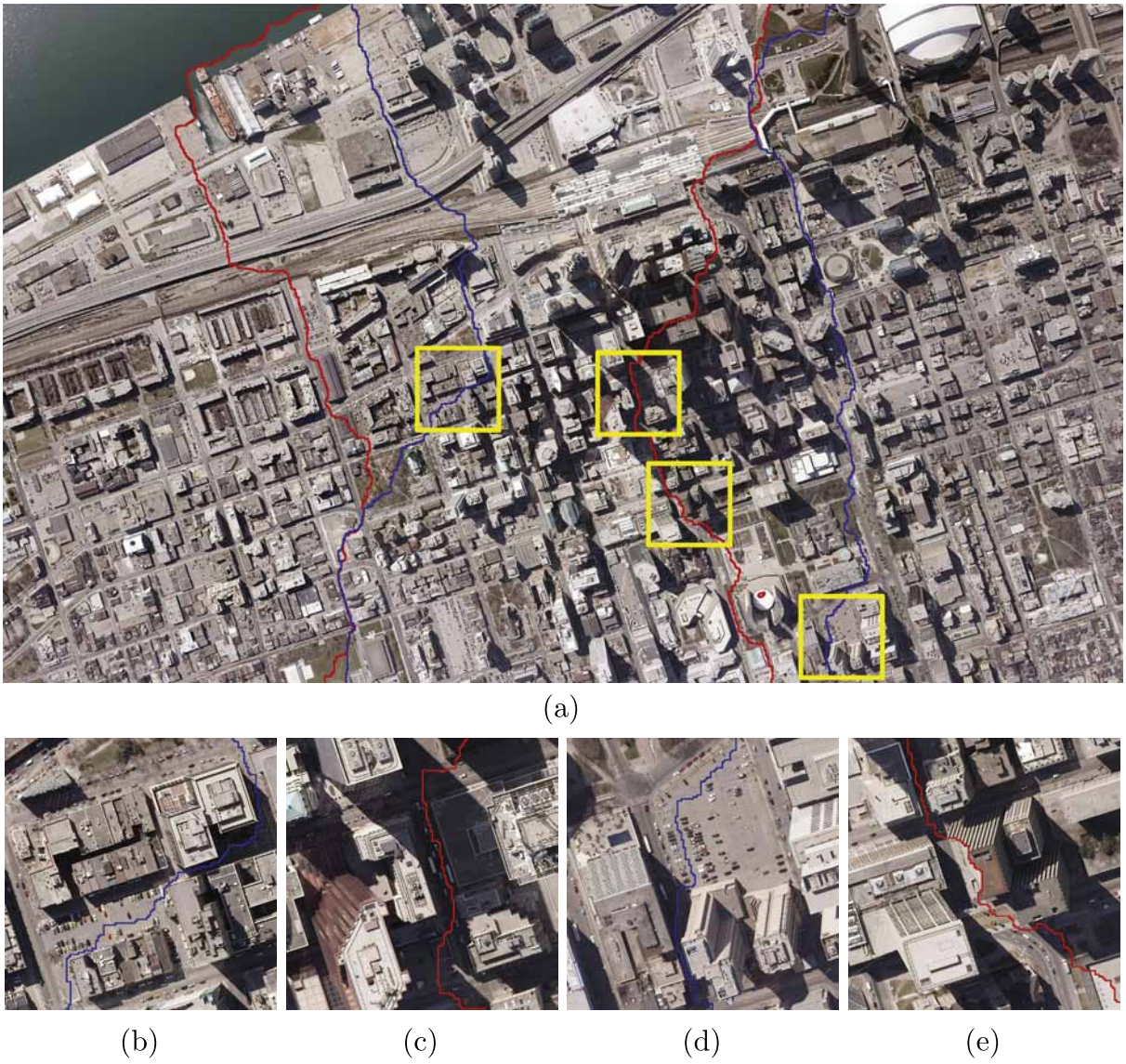

Fig. 14. The seamline detection results of five sequential aerial images of Toronto, Canada, and four detailed local image regions.

tedious work for the users. However, in our proposed human–computer interaction strategies, the users don't need to draw the new seamline for each failed region, they only need to tell our algorithm some necessary informations by simply marking some regions that the seamlines will or won't pass through in the failed regions and the remaining works will be done by our algorithm automatically. For example, if there exist many cars in the road, users of *OrthoVista* need to carefully draw the seam polygon to avoid passing through those cars, but in our proposed algorithm, users just need to tell our algorithm that the seamline should pass through this road, and the cars will be rounded automatically. In addition, our algorithm provides three different ways to modify the seamlines, which are more flexible than the seamline editor method used in *OrthoVista*. For the undesired seamline presented in Fig. 12(a), we have several more simple methods to solve it according to the strategies presented in Section 5.3. Firstly, the building regions can be marked as the region that the seamline won't pass through, and then our algorithm can automatically find another more better seamline avoiding pass through this building. Secondly, we can label the pixels in this building as one image, so the whole building will comes from only one image and the seamline won't pass through it. Lastly, we also can simply let the seamline pass through the neighboring road by marking the road as the region that the seamline will pass through. Of course, those ways can be combined arbitrarily to

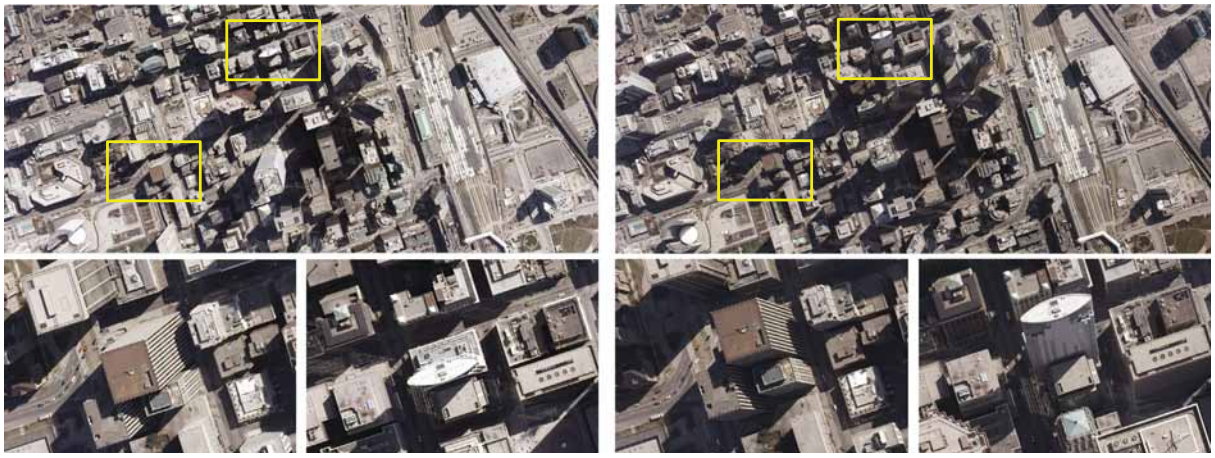

Fig. 15. The overlap regions of $\mathbf{A}_3$ and $\mathbf{A}_4$ in Fig. 14 and especially the detailed local regions including the corresponding input image regions of Fig. 14(e).

Table 2
Statistical results of the seamlines found in the sequential five Toronto aerial images.

	Seamline 1 ($\mathbf{A}_1, \mathbf{A}_2$)	Seamline 2 ($\mathbf{A}_2, \mathbf{A}_3$)	Seamline 3 ($\mathbf{A}_3, \mathbf{A}_4$)	Seamline 4 ($\mathbf{A}_4, \mathbf{A}_5$)	#Total
#Pixels	18,266	16,738	17,870	16,784	69,656
Numbers of obvious objects passed through	2 buildings and 2 cars	1 building and 3 cars	4 buildings and 1 car	1 building	8 buildings and 6 cars
Visible seam length (in pixels)	74	122	319	45	515
Times (s)	12.45	16.82	13.57	14.41	57.24

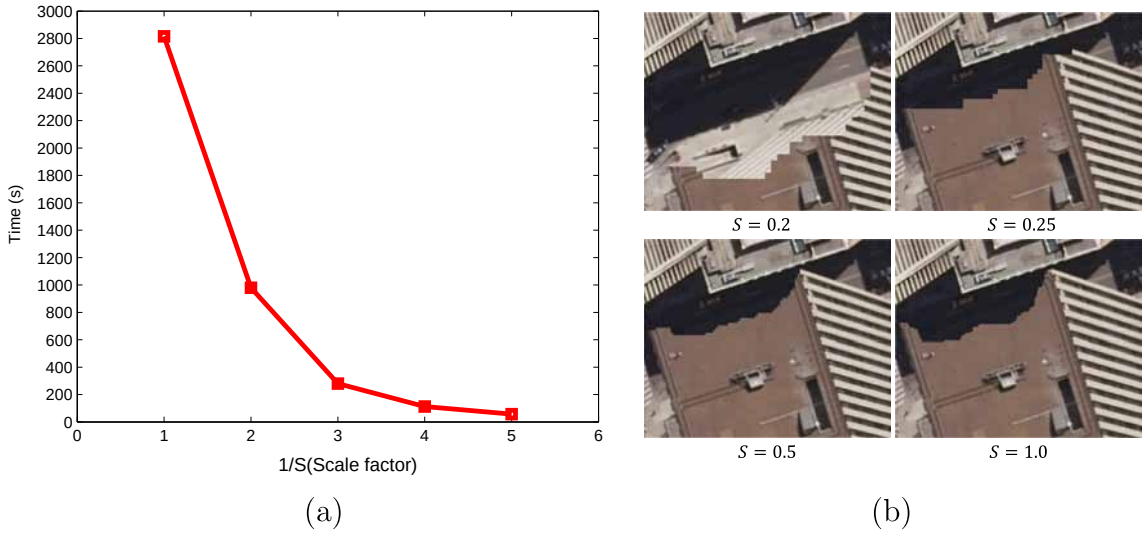

Fig. 16. Optimization performance with different down-scaling factors on five Toronto aerial images: (a) the optimization computational time curve; (b) the mosaicking results based on the detected seamlines in the image region shown in Fig. 14(e).

modify the seamline more flexibly and effectively. In addition, it is very difficulty for *OrthoVista* to manually edit the seamlines in multi-overlapped regions however it can be easily achieved by our proposed human–machine interaction strategies.

5.4. Seamline detection on aerial images

To prove that our algorithm can handle different types of images, we selected two different types of aerial images captured from different urban areas for testing. The first type of aerial images consists of two groups of aerial ones captured in Anyang, China. Each group is comprised of four images with a size of 9334×6000 pixels selected from two adjacent strips. The seamline detection results via the multi-frame joint optimization are shown in Fig. 13, from which we observed that almost all the seamlines follow roads and green lands while avoid passing through buildings, cars and pedestrians at the same time, which can magnificently conceal the parallaxes caused by the geometrical misalignments in different extents. The detailed statistical results of this experiment on Anyang aerial images are listed in Table 1. The second row in Table 1 presents the lengths of the corresponding seamlines. The numbers of obvious objects passed through by seamlines are presented in the third row. Considering that the lengths of seamlines measured by different people may be different, to reduce the influence of subjectivity, the lengths of visible seams with obvious parallax in the fourth row in Table 1 were

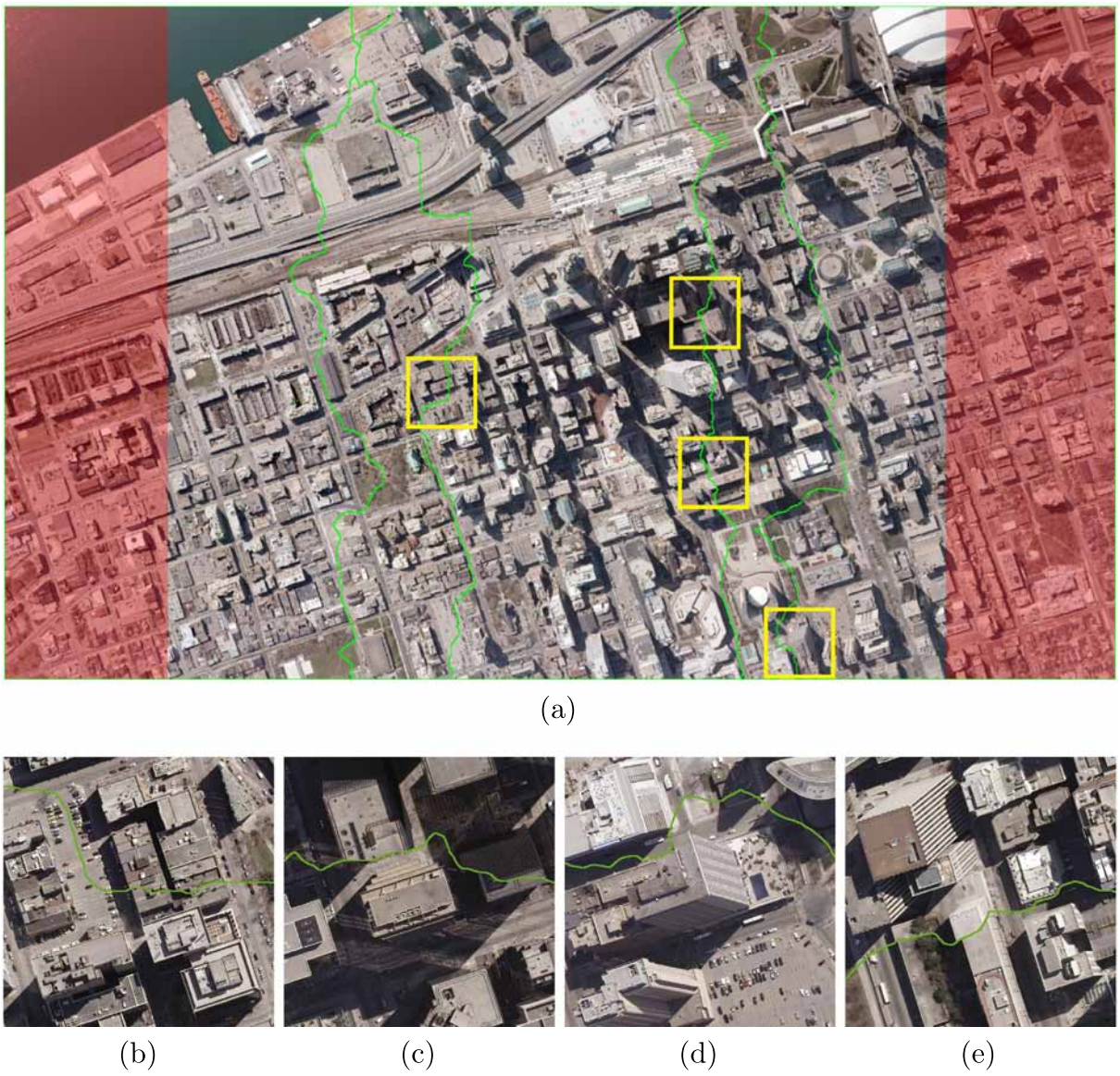

Fig. 17. The seamlines of five Toronto images generated by OrthoVista and four detailed local image regions. The red regions in both ends of the mosaicked image indicate the no-overlapped regions, which were generated by OrthoVista. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

measured by five different people and then averaged. The last row shows the computational time for finding the optimal seamline via our proposed algorithm. From Table 1, we found that there exist 10 buildings crossed by the seamlines as shown in Fig. 13(a), but most of them are located in the endpoint regions of the seamlines whose endpoints were formed by the overlapping intersection, which is difficult to bypass in most algorithms.

The second type of aerial images is comprised of five images with a size of 5750×3750 pixels in Toronto, Canada provided by ISPRS,[3] as shown in Fig. 14. These five images were stitched via the frame-to-frame optimization, which are denoted as $\mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3, \mathbf{A}_4$ and $\mathbf{A}_5$ from left to right in Fig. 14(a). Although the buildings in those regions are very tall, our algorithm also could find the high-quality seamlines avoiding passing through them, as shown in Fig. 14(b)–(d). However, we also observed that one seamline cannot successfully round a tall building as shown in Fig. 14(e). From corresponding input image regions shown in Fig. 15, we found that the roads have been almost occluded by the nearby buildings as well as others roads, so there does not exist any better choice for the seamline. The detailed statistical results of this experiment on five Toronto aerial images are listed in Table 2. Furthermore, Fig. 16 shows the optimization computation times and the seamline detection results in the failed image region shown in Fig. 14(e) with the different down-scaling factors. From Fig. 16(a), we observed that the elapsed times will be greatly decreased with the decreasing of the down-scaling factor S. From Fig. 16(b), we observed that the seamlines are improved but still cannot completely avoid passing through that tall building.

We also compared our seamlines detection results on Toronto aerial images with *OrthoVista*, which is a popularly used commercial software to generate orthoimage. The seamlines generated by *OrthoVista* are shown in Fig. 17. By comparing the seamline detection results of our proposed algorithm and OrthoVista, we found that our algorithm obviously outperforms *OrthoVista*. From Table 2, we found that the seamlines detected by our algorithm pass

[3] Available at http://www2.isprs.org/commissions/comm3/wg4/tests.html.

through 8 buildings and 6 cars at all, but 26 buildings and 47 cars were crossed by the seamlines detected by *OrthoVista* as shown in Fig. 17. The computational times of *OrthoVista* and our algorithm are 63 s and 57.24 s, respectively, which are almost the same.

6. Conclusions

In this paper, we proposed to efficiently detect globally optimal seamlines for mosaicking a set of geometrically aligned street-view panoramic or aerial images into a visual-appealing and informative wide-angle composite image in a graph cuts energy minimization framework. The contributions in this paper are summarized as follows:

- We formulated the optimal seamline detection as a unified graph cuts energy minimization problem. Experimental results have demonstrated its suitability for mosaicking different types of images, including street-view panoramic, aerial and oblique ones, respectively.
- We proposed to integrate a texture complexity energy term into the final energy costs in graph cuts. This energy term for each pixel is calculated based on the HOG vector of the small image region centered at that pixel. Experimental results showed that the texture complexity energy term can guide the seamline more possibly passing through some specific regions such as roads, sky, woodlands, and green lands, in which the parallaxes caused by geometrical misalignments are more easily concealed in vision. In addition, the other two energy terms based on the color differences and the gradient magnitudes are also combined in our energy functions to ensure to find the high-quality seamlines.
- We proposed a multi-frame joint optimization via multi-label graph cuts to deal with multi-overlapped images. Compared with the traditional frame-to-frame optimization strategy, our proposed strategy is more reasonable and effective but needs a higher computation cost and a significantly larger memory.
- We proposed simple but effective human–computer interaction (HCI) strategies for the end-users to semi-automatically detect the seamlines. Representative experiments on several groups of street-view panoramic images had proved its effectiveness and simplicity. Sometimes, such those HCI operations are very necessary for mosaicking a large set of images, especially aerial or oblique ones, into a pleasant composite image because some seamlines are possibly located in poor regions we don't expect.

Nevertheless, the proposed algorithm may be improved in the future in the following ways. First, the parallel optimization strategy is expected to be developed to efficiently detect seamlines from a large set of images via multi-frame joint optimization, which should improve the efficiency at the prerequisite of ensuring the high quality of seamlines. Second, the superpixel segmentation can be introduced to greatly improve the optimization efficiency by decreasing the number of elements in graph cuts. Third, the scene understanding or parsing can be applied in some particular image data. For example, the roads can be detected out for guiding the seamlines. At last but not least, the seamline network optimization framework (Pan et al., 2009; Mills and McLeod, 2013; Chen et al., 2014) can be combined with our algorithm to produce a complete image mosaic from a large set of images automatically.

Acknowledgements

This work was partially supported by the National Natural Science Foundation of China (Project No. 41571436), the Hubei Province Science and Technology Support Program, China (Project No. 2015BAA027), the National Natural Science Foundation of China (Project No. 41271431), and the National Basic Research Program of China (Project No. 2012CB719904).

References

Agarwala, A., Dontcheva, M., Agrawala, M., Drucker, S., Colburn, A., Curless, B., Salesin, D., Cohen, M., 2004. Interactive digital photomontage. ACM Trans. Graph. (TOG) 23 (3), 294–302.
Allene, C., Pons, J., Keriven, R., 2008. Seamless image-based texture atlases using multi-band blending. In: IEEE International Conference on Pattern Recognition (ICPR).
Bellman, R., 1957. Dynamic Programming. Princeton University Press, Princeton, NJ.
Boykov, Y., Funka-Lea, G., 2006. Graph cuts and efficient ND image segmentation. Int. J. Comput. Vis. 70 (2), 109–131.
Boykov, Y., Jolly, M.-P., 2001. Interactive graph cuts for optimal boundary & region segmentation of objects in N-D images. In: IEEE International Conference on Computer Vision (ICCV).
Boykov, Y., Kolmogorov, V., 2004. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. IEEE Trans. Pattern Anal. Mach. Intell. 26 (9), 1124–1137.
Boykov, Y., Veksler, O., Zabih, R., 2001. Fast approximate energy minimization via graph cuts. IEEE Trans. Pattern Anal. Mach. Intell. 23 (11), 1222–1239.
Brown, M., Lowe, D.G., 2007. Automatic panoramic image stitching using invariant features. Int. J. Comput. Vis. 74 (1), 59–73.
Chen, Q., Sun, M., Hu, X., Zhang, Z., 2014. Automatic seamline network generation for urban orthophoto mosaicking with the use of a digital surface model. Remote Sens. 6 (12), 12334–12359.
Chon, J., Kim, H., Lin, C.-S., 2010. Seam-line determination for image mosaicking: a technique minimizing the maximum local mismatch and the global cost. ISPRS J. Photogramm. Remote Sens. 65 (1), 86–92.
Dalal, N., Triggs, B., 2005. Histograms of oriented gradients for human detection. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR).
Dijkstra, E.W., 1959. A note on two problems in connexion with graphs. Numer. Math. 1 (1), 269–271.
Du, Q., Raksuntorn, N., Orduyilmaz, A., Bruce, L.M., 2008. Automatic registration and mosaicking for airborne multispectral image sequences. Photogramm. Eng. Remote Sens. 74 (2), 169–181.
Floyd, R., 1962. Algorithm 97: shortest path. Commun. ACM 5 (6), 345.
Gracias, N., Mahoor, M., Negahdaripour, S., Gleason, A., 2009. Fast image blending using watersheds and graph cuts. Image Vis. Comput. 27 (5), 597–607.
Helmer, E., Ruefenacht, B., 2005. Cloud-free satellite image mosaics with regression trees and histogram matching. Photogramm. Eng. Remote Sens. 71 (9), 1079–1089.
Hong, L., Chen, G., 2004. Segment-based stereo matching using graph cuts. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR).
Kass, M., Witkin, A., 1988. Snakes: active contour models. Int. J. Comput. Vis. 1 (4), 321–331.
Kerschner, M., 2001. Seamline detection in colour orthoimage mosaicking by use of twin snakes. ISPRS J. Photogramm. Remote Sens. 56 (1), 53–64.
Kolmogorov, V., Zabih, R., 2001. Computing visual correspondence with occlusions using graph cuts. In: IEEE International Conference on Computer Vision (ICCV).
Kolmogorov, V., Zabin, R., 2004. What energy functions can be minimized via graph cuts? IEEE Trans. Pattern Anal. Mach. Intell. 26 (2), 147–159.
Kwatra, V., Schödl, A., Essa, I., Turk, G., Bobick, A., 2003. Graphcut textures image and video synthesis using graph cuts. ACM Trans. Graph. (TOG) 22 (3), 277–286.
Levin, A., Zomet, A., Peleg, S., Weiss, Y., 2004. Seamless image stitching in the gradient domain. In: European Conference on Computer Vision (ECCV). Springer.
Mills, S., McLeod, P., 2013. Global seamline networks for orthomosaic generation via local search. ISPRS J. Photogramm. Remote Sens. 75 (1), 101–111.
Pan, J., Wang, M., 2011. A seam-line optimized method based on difference image and gradient image. In: 19th International Conference on Geoinformatics.
Pan, J., Wang, M., Li, D., Li, J., 2009. Automatic generation of seamline network using area Voronoi diagrams with overlap. IEEE Trans. Geosci. Remote Sens. 47 (6), 1737–1744.
Pan, J., Zhou, Q., Wang, M., 2014. Seamline determination based on segmentation for urban image mosaicking. IEEE Trans. Geosci. Remote Sens. Lett. 11 (8), 1335–1339.
Perez, P., Gangnet, M., Blake, A., 2003. Poisson image editing. ACM Trans. Graph. (TOG) 22 (3), 313–318.
Prados, R., Garcia, R., Neumann, L., 2014. Image Blending Techniques and Their Application in Underwater Mosaicing. Springer.
Rother, C., Kolmogorov, V., Blake, A., 2004. Grabcut: interactive foreground extraction using iterated graph cuts. ACM Trans. Graph. (TOG) 23 (3), 309–314.
Soille, P., 2006. Morphological image compositing. IEEE Trans. Pattern Anal. Mach. Intell. 28 (5), 673–683.
Szeliski, R., Uyttendaele, M., Steedly, D., 2011. Fast Poisson blending using multi-splines. In: IEEE International Conference on Computational Photography.
Uyttendaele, M., Eden, A., Skeliski, R., 2001. Eliminating ghosting and exposure artifacts in image mosaics. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR).

Wan, Y., Wang, D., Xiao, J., Lai, X., Xu, J., 2013. Automatic determination of seamlines for aerial image mosaicking based on vector roads alone. ISPRS J. Photogramm. Remote Sens. 76, 1–10.

Wang, D., Lim, K.B., 2011. Obtaining depth map from segment-based stereo matching using graph cuts. J. Vis. Commun. Image Represent. 22 (4), 325–331.

Xiong, Y., Pulli, K., 2009. Mask based image blending and its applications on mobile devices. In: Sixth International Symposium on Multispectral Image Processing and Pattern Recognition.

Xiong, Y., Pulli, K., 2010. Fast panorama stitching for high-quality panoramic images on mobile phones. IEEE Trans. Consum. Electron. 56 (2), 298–306.

Yang, Y., Gao, Y., Li, H., Han, Y., 2011. An algorithm for remote sensing image mosaic based on valid area. In: International Symposium on Image and Data Fusion (ISIDF).

Yu, L., Holden, E.-J., Dentith, M.C., Zhang, H., 2012. Towards the automatic selection of optimal seam line locations when merging optical remote-sensing images. Int. J. Remote Sens. 33 (4), 1000–1014.

Zhu, Q., Yeh, M.-C., Cheng, K.-T., Avidan, S., 2006. Fast human detection using a cascade of histograms of oriented gradients. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR).