Efficient Road Detection and Tracking for Unmanned Aerial Vehicle

Hailing Zhou, *Member, IEEE*, Hui Kong, Lei Wei, *Member, IEEE*, Douglas Creighton, *Member, IEEE*, and Saeid Nahavandi, *Senior Member, IEEE*

***Abstract*—An unmanned aerial vehicle (UAV) has many applications in a variety of fields. Detection and tracking of a specific road in UAV videos play an important role in automatic UAV navigation, traffic monitoring, and ground–vehicle tracking, and also is very helpful for constructing road networks for modeling and simulation. In this paper, an efficient road detection and tracking framework in UAV videos is proposed. In particular, a graph-cut–based detection approach is given to accurately extract a specified road region during the initialization stage and in the middle of tracking process, and a fast homography-based road-tracking scheme is developed to automatically track road areas. The high efficiency of our framework is attributed to two aspects: the road detection is performed only when it is necessary and most work in locating the road is rapidly done via very fast homography-based tracking. Experiments are conducted on UAV videos of real road scenes we captured and downloaded from the Internet. The promising results indicate the effectiveness of our proposed framework, with the precision of 98.4% and processing 34 frames per second for 1046 $\times$ 595 videos on average.**

***Index Terms*—GraphCut algorithm, homography, road detection, road tracking, unmanned aerial vehicle (UAV).**

I. Introduction

UNMANNED aerial vehicles (UAVs) have been widely used in many fields, particularly in transportation. The major applications include security surveillance, traffic monitoring, inspection of road construction, and survey of traffic, river, coastline, pipeline, etc. Relevant research can be traced back to the 2000s in the transportation departments of the Ohio [1], Florida [2], Georgia [3], and California [4] states within the United States. They use UAVs in autonomous navigation to follow roads/rivers, oil-gas pipeline inspection, and traffic parameters measurements. UAVs equipped with cameras are viewed as a kind of low-cost platform that can provide efficient data acquisition mechanisms for intelligent transport systems. With the increasing use of vehicles and their demands on traffic management, this kind of platform becomes more and more popular. Conventional traffic data collection [5] relying on fixed infrastructure is only limited to a local region and, thus, it is expensive and labor intensive to monitor traffic activities across broad areas. In comparison, UAV has advantages, including: (1) there is a low cost to monitor over long distances; (2) it is flexible for flying across broad spatial and temporal scales; and (3) it is capable of carrying various types of sensors to collect abundant data. To collect information for the transportation system, it is important to know where the roads are in UAV videos. Knowledge of road areas can provide users the regions of interest for further navigation, detection and data collection procedures, benefiting their efficiencies and accuracies.

In the literature of road detection and tracking, most approaches use the color (texture) and/or structure (geometry) properties of roads. Among them, the combination of road color and boundary information have achieved more robust and accurate results than using only one of them in road detection, as shown in the work [6], [7]. Therefore, we are in more favor of using both types of information. Because real time is required in many UAV-based applications, our major target is how to effectively combine both types of information for road detection/tracking in an efficient way. Intuitively, there are two rules to make one integrated framework efficient. First, each component of the framework should be fast. Second, if one component is faster than the others in achieving the same purpose, it would better make use of the fastest component as much as possible.

In this paper, we follow the aforementioned two rules to make our framework fast. Specifically, our framework includes two components: road detection and road tracking. In road detection, we propose to utilize the GraphCut algorithm [8] because of its efficiency and powerful segmentation performance in 2-D color images. In road tracking, we propose a fast road tracking approach. There are two facts that spur us to implement road tracking. First, although GraphCut is very efficient, it still cannot achieve a real-time performance when the UAV image resolution is high enough (as in our work), and performing road detection frame by frame is not time efficient. Second, road appearance usually does not abruptly change in video; therefore, road tracking can make full use of continuous spatial–temporal information of roads in videos and thus can quickly infer road areas from previous results.

In road tracking, we aim to track the road border structure between two consecutive frames. In a computer vision society, most developed tracking techniques, such as meanshift [9], particle filter [10], and optical flow [11], are appearance-based methods. They are for specific object class, e.g., face, car, or pedestrian, where objects share common features. Although

Manuscript received January 7, 2014; revised April 17, 2014; accepted June 9, 2014. This project was supported by the Centre for Intelligent System Research under the Collaborative Research Networks (CRN) project funding. The Associate Editor for this paper was D. Fernandez-Llorca.

H. Zhou is with the Centre for Intelligent Systems Research, Burwood, VIC 3125, Australia (e-mail: hailing.zhou@hotmail.com).

H. Kong is with Massachusetts Institute of Technology, Cambridge, MA 02139 USA (e-mail: huikong@mit.edu).

L. Wei, D. Creighton, and S. Nahavandi are with the Centre for Intelligent Systems Research, Burwood, VIC 3125, Australia (e-mail: lei.wei@deakin.edu.au; douglas.creighton@deakin.edu.au; saeid.nahavandi@deakin.edu.au).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TITS.2014.2331353

some contour-based methods, such as snake [12] or curve fitting [13] approaches, have shown promising road detection and tracking performance for applications to unmanned ground vehicles (UGV), they largely depend on the extracted road border (or markings) and vanishing points of the road, and might not be easily adaptable to UAV applications because the road boundary or markings are usually not salient enough to be detected due to the altitude of UAVs. In addition, these approaches are too computationally complex to be real time.

Because of the aforementioned difficulties in adapting existent tracking approaches to track a road in UAV videos, we develop a novel tracking technique based on homography alignment. Homography is a transformation that can be used to align one image plane to another when the moving camera is capturing images of a planar scene. Generally, the road region in our application can be well approximated by a plane, and therefore, homography can be applicable to our road images. As aforementioned, we aim at making our framework efficient. We thus develop a fast homography estimation approach for road tracking, where the efficiency in homography estimation is attributed to three factors: (1) the FAST corner detector [14] is used to find key points in each road frame. (2) The Kanade–Lucas–Tomasi (KLT) tracker [15] is applied to establish a correspondence between the two sets of FAST corners in two consecutive frames. (3) A context-aware homography estimation approach is given where only the corresponding FAST corners in the road neighbors are used with random sample consensus (RANSAC) estimator.

Based on our homography-alignment scheme, if we know where the road area is in a previous frame, we can quickly locate the corresponding road region in current frame. Fig. 1 illustrates the idea of road tracking based on homography alignment, where the road region in (a) captured at time t is known *a priori* and the corresponding road region in frame $t+1$ is tracked through mapping the road of (a) to (b) based on the estimated homography. In a similar way, we can also track the corresponding road area in frame $t+2$ based on the located road region in frame $t+1$, and so on so forth. Thus, we can achieve the purpose of tracking road area in the next a few frames. However, the tracking strategy based on the homography alignment is not complete because the road region mapped from previous frame to current one does not cover the whole road area but only a large portion of it, see the area delineated by the red lines in Fig. 1(b). This is due to the fact that only the road area in the overlapping part (of the two aligned frames) can be tracked, whereas the road area in the nonoverlapping part still needs to be detected. To solve this problem, we propose an online GraphCut scheme to detect the road area in the nonoverlapping part. Since the nonoverlapping part only covers a very small portion of the image, the detection of road area in the nonoverlapping part is very fast. Note that our homography-alignment-based approach is much faster than GraphCut in locating the same road area. Therefore, the use of our fast homography-based approach to track most of the road region in each frame can save considerable computation than purely using GraphCut segmentation.

The contributions of this paper are listed as follows: (1) We propose a real-time framework for the detection and tracking of a specific road in low- and mid-altitude UAV videos. To our best knowledge, this is the first work that has ever been proposed to introduce a tracking technique to speed up the localization of road in UAV videos. (2) We combine a fast homography-alignment-based tracking and an online GraphCut detection in an effective way to locate road region in each frame. The fast homography-alignment approach, which is context aware, is given to track most part of the road region and the online GraphCut detection to detect the rest trivial road area in every frame. (3) To correct the tracking error caused by the drift problem in successive homography alignment, we give an online drift-correction scheme, where a comparison is first made between the color distribution of the tracked road area in current frame and the one obtained from the tracked road regions in some previous frames, and a decision can be made based on the comparison whether it is necessary to switch from tracking to detection. It should be noted that our proposed technique is not just limited to road detection and tracking. It can be also applicable to river, pipeline, or coastline detection and tracking in UAV videos.

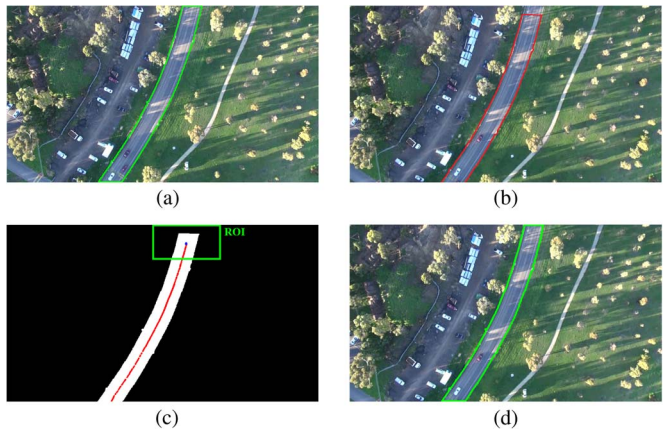

Fig. 1. Road tracking by homography alignment, where the two frames (a) and (b) are not consecutive and they are selected to illustrate the clear gap. (a) Road detection result, where the contour of the road region is shown by green. (b) Road alignment result by homography transformation. (c) Road median axis and the predicted region of interest (ROI) (note that the road area to be detected in the nonoverlapping region falls within the predicted ROI). (d) Road detection in ROI and final road tracking result.

The rest of this paper is organized as follows. Section II reviews some most related works. Section III provides the introduction of the hardware configuration. Section IV describes in details on how to use an offline GraphCut to initialize the road region and how to adjust GraphCut adaptively for road detection in the middle of tracking and drift-error correction. The proposed fast homography-alignment-based road tracking technique is presented in Section V. Section VI discusses experimental results in qualitative and quantitative ways, followed by conclusion in Section VII.

II. Related Work

Road detection and tracking in UAVs, particularly low- and mid-altitude UAVs is our focus in this paper, which can be used for autonomous navigation [4], [16], inspection [17], [18], traffic surveillance and monitoring [1], [20], [21]. A monocular

color camera is often equipped in this area, where UAVs usually fly up to 500 m. The camera can clearly capture each vehicle on the ground and also has large spatial view on traffic areas. The other research line in UAV-based road detection uses satellite or high-altitude UAVs [22]–[24], which aims to identify road network, including many junctions and roundabouts from an image. High-resolution cameras are generally utilized in the high-altitude UAV applications, where cameras are usually 1000 m away from ground. The third types of road detection works use ego-vehicles with onboard cameras for advanced driver assistance systems or UGVs autonomous navigation. A substantial amount of work [7], [25]–[29] have been done in this area. Since the focus of this paper is on road detection and tracking using low-/mid-altitude UAVs, we only give a review of the most related works in this area.

In general, region color distributions or/and boundary structures are probably the most important information utilized for road detection. In [4], they proposed to learn road color distributions using Gaussian mixture models (GMMs) from given sample images, and then determine road pixels in each frame by checking the probabilities of pixels that fit the GMMs. Learning both color and gradient information from a sample image is proposed in [17]. Gaussian and gamma distributions are used to represent color and gradient models. In [16], they learn structures from a sample image. Vanishing points are calculated by detecting pairs of line segments, and used to rectify the image in order to obtain rectified horizontal scans. Road boundary are then identified by finding large intensity changes in the cross-section profile of each horizontal scan. In [30], the clustering technique based on prior hue and texture information is used to classify each image pixel into target and background, and then boundary lines are fitted to refine the desired region. In [18], [19], a simple intensity thresholding technique is used to obtain initial road regions, followed by refinements of local line segment detections, where the assumption is that roads intensities are very different from neighborhood regions and roads can be approximated locally by linear line segments. Note that control algorithms of steering UAV in real time are also developed in [4], [16], which are not in the scope of this paper. Our objective is a fast real-time road tracking approach in UAV videos. Our future work will investigate how to use the tracked road information for parsing scenes in UAV videos.

III. Hardware Configuration

The UAV used in this paper is configured as shown in Fig. 2, which includes several basic components: an octocopter, ground monitors and controllers, batteries and chargers, a camera and camera stabilization support. The octocopter is equipped with a global positioning system (GPS) and an altitude controller, which can fly up to 400 m within a 250 m radius. The fly can be autonomous by aid of programmed GPS waypoints or interactively by remote controls from users on the ground. The camera is carried at the bottom of the UAV, which is full HD in the progressive mode. The UAV is capable of tilting the camera to make it look straight down or with arbitrary tilted angles with regard to the ground. Some UAV images are shown in Fig. 3, which were captured in different days and at different places with varying altitudes. The road colors vary significantly, as well as the background.

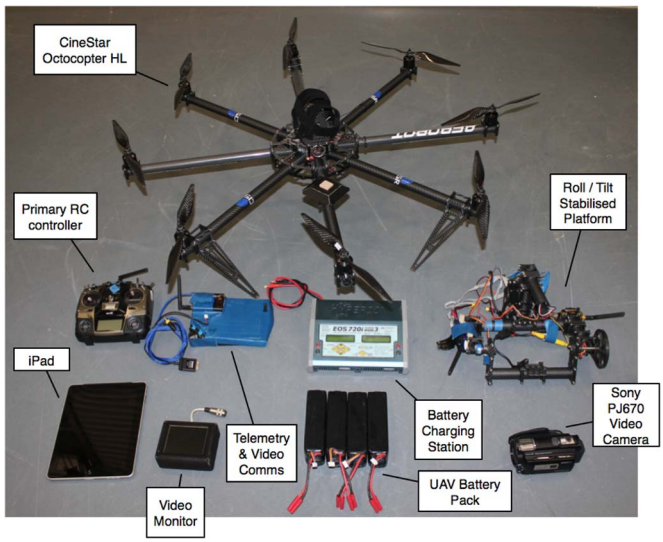

Fig. 2. Hardware configuration of UAV. (The identification of any company and commercial product does not imply endorsement or recommendation by any of the authors or their institutions.)

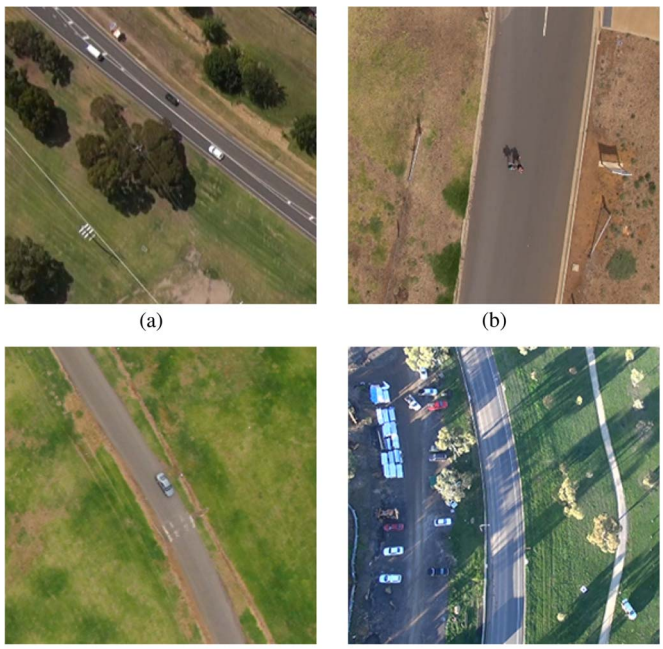

Fig. 3. Different scenarios with diverse road colors, illuminations, and background scenes.

IV. Road Detection Using GraphCut

First, we introduce a GraphCut-based road detection method, where the GMMs are used to model image color distributions, and structure tensors are employed to capture image edge features. To accelerate the performance, we perform road detection on downsampled images as in [7].

A. Overview of Applying GraphCut to Road Detection

With GraphCut, we view our road detection as a binary labeling problem as in [31] and [32]. An image I is formed by a set of pixels $I = \{v_{xy} : (x, y) \in \Omega\}$. For a color image, v_{xy} is an RGB vector, and for a gray image, v_{xy} is an intensity value. We use (x, y) to denote the position of a pixel, and $\Omega \subset R^2$ is the image domain. Let $U = \{u_{xy}, (x, y) \in \Omega\}$ be a set of labels, where $u_{xy} \in \{0, 1\}$. u_{xy} is assigned to the pixel at the position (x, y), which can be either 0 for nonroad or 1 for road. U defines the road areas where pixel labels are 1.

The task for road detection is to find U minimizing the following Gibbs energy function:

$$E(U) = E_c(U) + \gamma E_s(U). \tag{1}$$

The energy function consists of a color term E_c and a structure term E_s balanced by a tradeoff factor γ, $\gamma > 0$. We find that it works well in our experiments when γ is set to a typical value of 50. The E_c defines how pixel color/intensity fits into road/nonroad color models in order to penalize color/intensity difference between pixels and the color models. The color models can be represented by histograms [32] or GMMs [31]. The $E_c(U)$ is defined as

$$E_c(U) = \sum_{(x,y)\in\Omega} -\log \mathcal{P}(v_{xy}, u_{xy}) \tag{2}$$

where $\mathcal{P}(v_{xy}, u_{xy})$ denotes the probability of a pixel v_{xy} matching the color model of road $(u_{xy}=1)$ or nonroad $(u_{xy}=0)$.

The $E_s(U)$ is defined as follows to penalize color/intensity difference between neighboring pixels

$$E_s(U) = \sum_{(xy,ij)\in\xi} |u_{xy} - u_{ij}| e^{-\beta\|v_{xy}-v_{ij}\|} \tag{3}$$

where ξ is the set of 8-neighboring pixel pairs. We $|\cdot|$ to denote the absolute value, allowing us to capture color difference only along the segmentation boundary. $\|\cdot\|$ is the L^2-norm. The parameter β is to control the smoothness and preciseness of the segmentation boundary, which is generally chosen to be as follows according to [31]:

$$\beta = \frac{1}{2E\left(\|v_{xy} - v_{ij}\|^2\right)} \tag{4}$$

where $E(\cdot)$ denotes the expectation value over an image.

A weighted graph is then constructed, where the cost on each graph edge is defined based on the terms E_c and E_s. The minimization of the Gibbs function becomes to find a cut with the minimum cost to partition the graph into two, which is solved by a min-cut/max-flow algorithm [32]. An iterative minimization scheme in [31], [33] can further improve accuracy.

Next, we give detailed descriptions of the modeling of the color distributions using GMMs, image structure features capturing, and road detection using GraphCut.

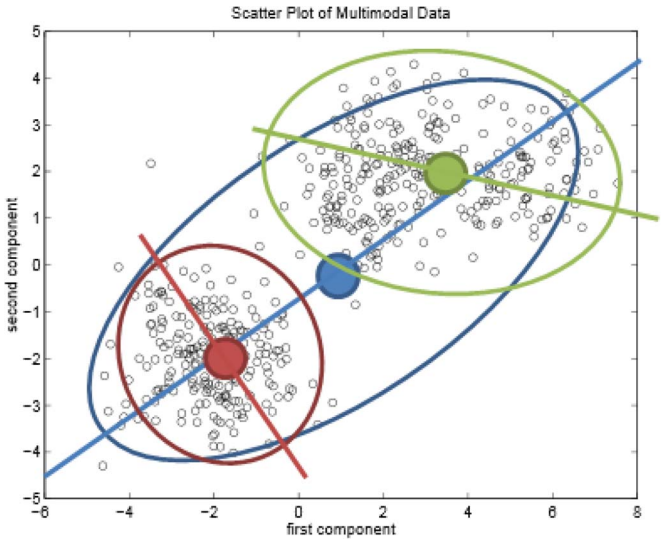

Fig. 4. Illustration of the Orchard–Bouman splitting procedure on 2-D data: In the first iteration, the whole data set (blue oval) is split into two (green and red ovals) after projecting data onto the direction of the eigenvector (blue line) that corresponds to the largest eigenvalue of the covariance of all data. If the largest eigenvalue of the covariance of the data within the green oval is larger than that of the red oval, the data within the green oval is split into two groups next. Otherwise, the data within the red oval is split next. This process is repeated until the desired number of clusters is reached.

B. Off-Line (Static) GMMs for Road/Nonroad Color Modelling

Road and nonroad pixels are first collected from sample images for learning road/nonroad color distributions. We select dozens of frames from UAV videos as the sample images, and scratch several strokes in each frame using green and red colors to specify road and nonroad pixels, respectively, as shown in Fig. 5(a). Two Gaussian mixture models (GMMs) GMM_0 with K_0 components and GMM_1 with K_1 components are used to represent the nonroad and road color distributions. In this paper, we choose K_0 as 3 and K_1 as 5, which work well in our experiments.

The next is to create K_0 components for GMM_0 and K_1 components for GMM_1. This requires to partition nonroad pixels into K_0 clusters, and road pixels into K_1 ones. There are a large number of methods for data clustering [34]. In this paper, the Orchard and Boumand binary splitting algorithm [35] is employed. It performs statistical analysis on data to fast produce solutions, where in each iteration, the cluster with larger variance is split into two until the desired number of clusters is reached. Empirically, we found that the Orchard and Boumand approach is twice faster than Kmeans in creating GMMs. Fig. 4 gives an example of the Orchard and Boumand splitting process on 2-D data. Algorithm 1 presents the procedure of GMM creation.

Based on the clustering results, a GMM (for either road or nonroad) can be represented by using K triplets $\text{GMM} = \{(\mu_1, \Sigma_1, w_1), (\mu_2, \Sigma_2, w_2), \ldots, (\mu_K, \Sigma_K, w_K)\}$, where μ_i and Σ_i are the mean color value and the 3×3 covariance matrix of a Gaussian component, respectively. The w_i denotes the weight of each Gaussian component. We set w_i as the number of pixels in the ith component over the total number of (road or nonroad) pixels.

Algorithm 1 The creation of a GMM

Input: The number of components K and a set of road (or nonroad) pixels Γ.
Output: K components $C_1, \ldots, C_K$.
1. $C_m = \Gamma$, $m = 1$. C_m is a cluster with the largest eigenvalue.
2. For $(k = 2, \ldots, K)$
   2.1 For C_m, calculate the mean value μ_m, the covariance matrix Σ_m, the largest eigenvalue λ_m and the corre sponding eigenvector v_m of Σ_m.
   2.2 Split C_m into two sets, $C_k = \{c \in C_m : v_m c \leq v_m \mu_m\}$, and $C_m = C_m - C_k$.
   2.3 For $(i = 1, \ldots, k)$
      2.3.1 For C_i, calculate the covariance matrix Σ_i, the largest eigenvalue λ_i of Σ_i.
      2.3.2 If $(\lambda_i > \lambda_m)$, $m = i$.

C. GraphCut Detection Based on Static GMMs

With the color distribution models GMM_0 and GMM_1, E_c in (2) can be calculated using

$$E_c(U) = \sum_{(x,y)\in\Omega} -\log \mathcal{P}_{\text{GMM}}(v_{xy}, u_{xy}) \tag{5}$$

where $\mathcal{P}_{\text{GMM}}(v_{xy}, 0) = \mathcal{P}_{\text{GMM}_0}(v_{xy})$ and $\mathcal{P}_{\text{GMM}}(v_{xy}, 1) = \mathcal{P}_{\text{GMM}_1}(v_{xy})$, representing the probabilities of the pixel color v_{xy} fitting the nonroad model GMM_0 and the road model GMM_1. $\mathcal{P}_{\text{GMM}_0}(v_{xy})$ and $\mathcal{P}_{\text{GMM}_1}(v_{xy})$ are defined using

$$\sum_{k=1}^{K}\left[w_k \frac{1}{\sqrt{\det(\Sigma_k)}} e^{\left(-\frac{1}{2}(v_{xy}-\mu_k)^T \Sigma_k^{-1}(v_{xy}-\mu_k)\right)}\right]. \tag{6}$$

The definition of E_s in (3) relies on a constant β and the color difference. We notice that when $\beta = 0$ the segmentation boundary will be smooth everywhere; when $\beta > 0$ the smoothness of the segmentation boundary will be relaxed in regions with high contrast (i.e., high color difference $\|v_{xy} - v_{ij}\|^2$). With the definition of (4), β is fixed over an image. However, a UAV image often contains regions with varying contrasts, where the fixed β is not sufficient to adapt E_s to the different regions. Inspired by the work in [36], we use local geometric structure information to adaptively adjust β over the image so that E_s favors the segmentation boundary along image structure features (i.e., road borders)

$$E_s(U) = \sum_{(xy,ij)\in\xi} |u_{xy} - u_{ij}| e^{-st(v_{xy}, v_{ij})} \tag{7}$$

where $st(\cdot)$ is defined by incorporating both color difference and geometric structures as

$$st(v_{xy}, v_{ij}) = \kappa \frac{s(v_{xy}, v_{ij}) + s(v_{ij}, v_{xy})}{2} \tag{8}$$

$$s(v_{xy}, v_{ij}) = \frac{\lambda_+(x,y)^2 (i-x, j-y) S(x,y) (i-x, j-y)^T}{\lambda_-(x,y) + \epsilon\,(\lambda_+(x,y) - \lambda_-(x,y))} \tag{9}$$

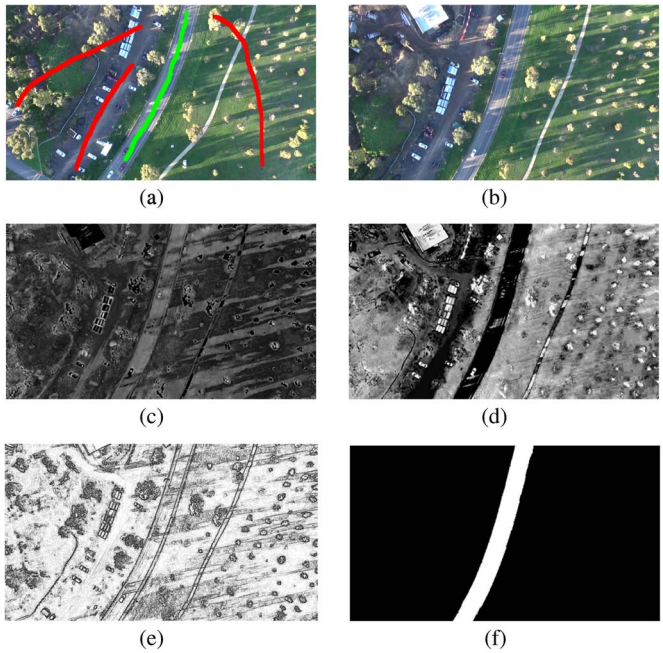

Fig. 5. Road detection. (a) Sample image with strokes examples for collecting road (green) and nonroad (red) pixels. (b) Video frame. (c) and (d) Probability maps P_{GMM_1} and P_{GMM_0} of pixels in (b) fitting the road and nonroad GMMs, respectively, calculated based on (6). (e) the illustrations of e^{-st} on ξ. (f): the result of GraphCut road segmentation for (b).

The calculation of (8) is to make st symmetric with respect to pixels at (x, y) and (i, j), where κ is a global parameter. In the definition of (9), the term $(i-x, j-y)S(x,y)(i-x, j-y)^T$ is approximately $\|v_{xy} - v_{ij}\|^2$ measuring the difference of color, and the other terms are used to give a pixel with large values of λ_+ and λ_+/λ_- a big weight of being on the segmentation boundary, referred to [36]. ϵ is a small number introduced to avoid the case that λ_+/λ_- is too big. $\lambda_+(x, y)$, and $\lambda_-(x, y)$ are the maximal and minimal eigenvalues of the structure tensor S at (x, y), seeing (10). With these definitions, the more similar two colors are (or the smaller λ_+ and λ_+/λ_- are), the larger E_s, and thus less likely to be the segmentation boundary.

Fig. 5 illustrates E_c and E_s visually. The segmentation result is to assign the label "1" to the pixels with low P_{GMM_0} and high P_{GMM_1} values and the segmentation boundary goes along image structures (i.e., low e^{-st} values). Morphology operations, including erosion and dilation are performed to remove noises and fill holes. Contour analysis is applied to find large connected regions, which are the final road segmentation. Since the GraphCut road detection described in this section is based on a static-GMM model of road colors, we call it the GraphCut detection approach based on static GMMs.

1) Structure Tensor: To easily interpret the structure tensor S, the color image I is viewed as a differentiable function $I(x, y) : \Omega \to R^3$. A 2×2 symmetric matrix defined in (10) with 3×3 window size is the structure tensor [36], which indicates local geometric structures of the image

$$S = \begin{bmatrix} I_x^T I_x & I_x^T I_y \\ I_y^T I_x & I_y^T I_y \end{bmatrix} \tag{10}$$

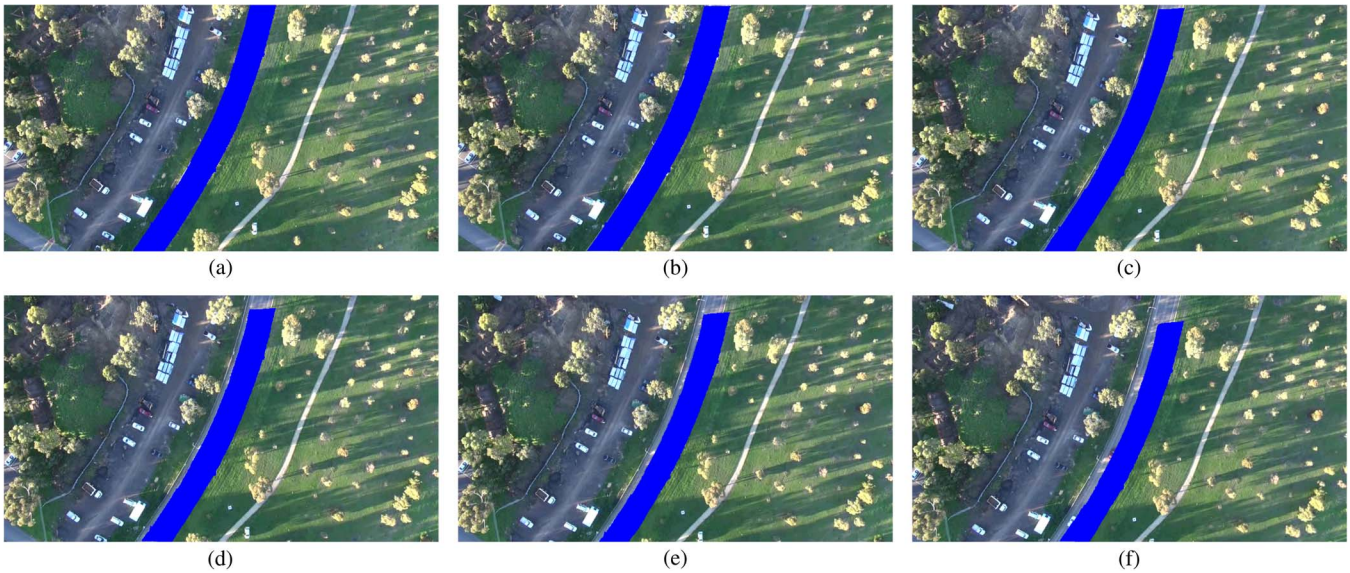

Fig. 6. Illustration of the drift problem in homography-alignment-based road tracking. (a) Initialized road region in the first frame. (b) to (f) Mapping the road region of (a) ***to the 20th, 40th, 60th, 80th, and 100th frames***, respectively, by the transformation derived from the accumulative homographies. The drift problem in road tracking is more serious with more accumulations. To correct the drift error in tracking, we automatically locate the frame where it is necessary to switch from homography-alignment-based tracking to GraphCut detection for road area segmentation.

2) Image Downsampling: For an image with N pixels, the GraphCut algorithm has an average computational complexity of $O(NlogN)$, which is the bottleneck of our proposed system. To reduce the processing time, we perform road detection on a downsampled image, where the image resolution is reduced to a half by applying the bicubic interpolation algorithm.

D. Online GraphCut Detection Based on Dynamic GMMs

We need the online GraphCut detection in two aspects: the first one is to detect road area in the nonoverlapping region of the two aligned frames, as shown in Fig. 1(b) and (c). The second one is that we need to switch from road tracking to road detection when the accumulated drift error in the homography-alignment-based road tracking is too large, as shown in Fig. 6. For both cases, the aforementioned described GraphCut detection based on static GMMs will not be able to adapt to the dynamic change of road scenes due to shadows, road reconstruction, varying background, etc.

Therefore, we propose to update GMMs online. New road and nonroad pixels are automatically collected at intervals (40 frames) based on successful road tracking results. The successful results can differentiate road from nonroad areas. In these areas, the erosion morphology operations are applied to produce center parts of each area, and the pixels with high probabilities are selected for updating corresponding GMMs. After that we use the selected road and nonroad pixels to update GMM_0 and GMM_1 to make GMMs adaptive to the evolution of image frames. The GMM update is quickly achieved by using the Gaussian mixture clustering algorithm instead of rerunning the algorithm described in Algorithm 1, where each pixel is grouped into one of the existing K GMM components that produce the pixel with the largest probability. This scheme is described in Algorithm 2. We call the GMMs with this update procedure as the dynamic GMMs. Correspondingly, we call the GraphCut detection with the dynamic GMMs update as the online GraphCut.

Algorithm 2 The procedure of updating a GMM

Input: A set of new road (or nonroad) pixels Γ^{new}, and GMM components $C_1^{\text{old}}, \ldots, C_K^{\text{old}}$: (μ_i, Σ_i), $i = 1, \ldots, K$.
Output: New GMM components $C_1^{\text{new}}, \ldots, C_K^{\text{new}}$.

1. $C_1^{\text{new}} = \ldots = C_K^{\text{new}} = \text{NULL}$.
2. $l_{\max} = -1$, $l_{\max}$ records the largest likelihood value.
3. For each pixel c_i in Γ^{new}
   3.1 $l_{\max} = -1$.
   3.2 For $(j = 1, \ldots, K)$
      3.2.1 Calculate the likelihood value l_{ji} of the component C_j^{old} producing the pixel c_i: $l_{ji} = (1/\sqrt{\det(\Sigma_j)})(1/2)[(c_i - \mu_j)^T \Sigma_j^{-1}(c_i - \mu_j)]$
      3.2.2 If $(l_{ji} > l_{\max})$, $l_{\max} = l_{ji}$ and $m = j$.
   3.3 $C_m^{\text{new}} = C_m^{new} \bigcup \{c_i\}$

V. Homography-Alignment-Based Road Tracking

In this section, we give the details on how to achieve a fast road tracking based on homography alignment. We also propose a solution to correct the drift problem caused by accumulation error in homography estimation, and give criteria on assessing tracking results. Fig. 7 shows the pipelines of road tracking.

A. Fast Homography Estimation

We treat the ground scene as a plane π in the view of UAVs. Fig. 8 gives an illustration of the scene. For a 3-D point P_π, it has the coordinate p in I_t when we project P_π on the image I_t

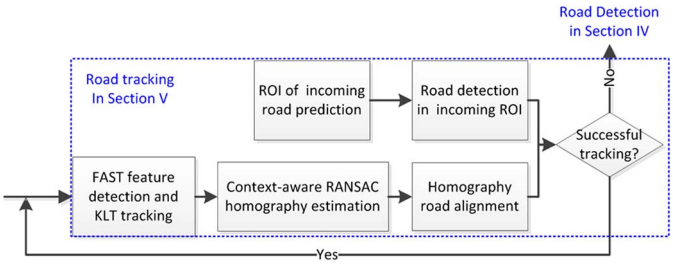

Fig. 7. Flowchart of road tracking.

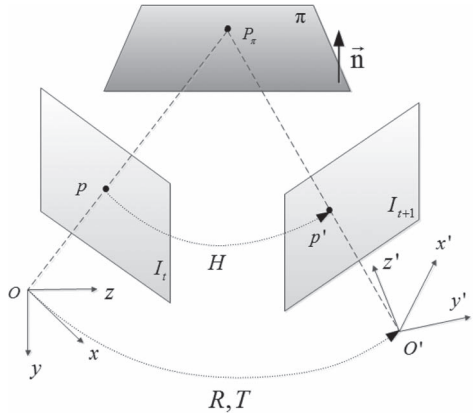

Fig. 8. Geometry homography transformation.

in the camera coordinate system (O, x, y, z), and the coordinate p' in I_{t+1} by projecting P_π on I_{t+1} in (O', x', y', z'). The two camera coordinate systems are transformed by the rotation R and translation T. Suppose that the plane π is with a normal vector $\mathbf{n}$. Homography matrix H explained by using R, T, and $\mathbf{n}$ is able to match the image points of the plane π between the two images I_t and I_{t+1} [37]. H is a 3×3 matrix with 8 freedoms: 3 for the rotation, 3 for the translation and 2 for the normal vector to the plane. Given $M \geq 4$ noncollinear corresponding point pairs (p_i, p_i') $(i = 1, \ldots M)$, H can be calculated. With H, we can track road regions in the current frame (i.e., I_{t+1}) by geometrically mapping the road regions in previous frame (i.e., I_t) to I_{t+1}. Fig. 1(a) and (b) illustrates the mapping of road region from a previous frame to current one based on homography alignment.

To estimate H between two images, the most common way is firstly to detect sets of interest points in two frames, then find correspondence between the two sets of interest points through correlation-based matching, and finally estimate the homography matrix based on matched point pairs via a robust approach such as RANSAC [38]. Generally, the homography estimation in this way is computationally complex when the image resolution is as high as in our application, where the two computationally high parts are the point (or corner) matching and the RANSAC steps.

Therefore, we propose to speed up these two steps to make our homography-alignment-based tracking fast. To speed up the procedure of establishing correspondence between two sets of corners, we first apply the FAST approach [14] to detect interest points in each frame. Compared with some other well-known features such as SIFT [39], SURF [40], Harris [41], FAST

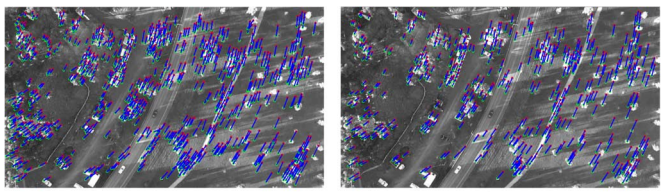

Fig. 9. Point pairs found by using tracking (a) and matching (b) techniques. Red points are detected in the previous frame. Green points are the tracked and matched points in the current frame. The point pairs are connected with blue lines.

can achieve the best balance between accuracy and efficiency for our purpose. Next, the KLT tracker is applied to the set of detected FAST corners in each frame so that we can get the predicted FAST corners in the following frame. In other words, the KLT tracker establishes correspondence between the detected FAST corners in current frame and the predicted ones in the following frame through optical flow technique. Note that the procedure of establishing corner correspondence here is different from the conventional way in establishing the correspondence of corners. In the conventional (most common) way, the corners are detected in each frame and correlation is used to establish the matching between the two sets of corners of two frames. It might work in more general cases, where the two frames are perhaps not close to each other, but the correlation is time consuming due to its high computational complexity. In contrast, our application is a special case, where the two frames to be aligned is consecutive, and the motions of corners are small, which can just satisfy the condition of the KLT tracker. Fig. 9(a) and (b) shows point pairs that are found by using tracking and matching techniques, respectively. The similar points movements (i.e., magnitudes and directions) in Fig. 9(a) and (b) indicates the effectiveness of the tracking technique.

Once we find the corresponding FAST corners of two consecutive frames through KLT, next, we use them to estimate the homography between current frame and the following one. Specifically, we propose a fast homogaphy alignment process. Since our purpose of using homography is to align the road region in two consecutive frames as accurately as possible, we propose a context-aware homography estimation approach, where only the corresponding FAST corners in the local neighborhood of road (a rectangular region which is obtained by expanding one quarter of the width of the road bounding box to left and right, respectively) are used within the RANSAC process. This is motivated by the observation that the homography that is estimated by RANSAC based on a local sampling of FAST-corner pairs can more accurately align the two images at this local region than the homography that is estimated by the common RANSAC (i.e., the random sampling is distributed across the whole image region). We illustrate the context-aware homography estimation scheme in Fig. 10 and show its advantage over the common RANSAC (shown in Fig. 11).

B. Road Tracking

The road area is tracked in a current frame by transforming the road region of a previous frame with the homography

Fig. 10. Illustration of the context-aware homography estimation for local region alignment. **The first row**: the first two images show the two consecutive frames (the previous and current ones), and the third image shows the tracked road region in the previous image. **The second row**: the first image shows the tracked FAST corners (by KLT) in the current frame. The second image shows the detected FAST corners in the current frame, but only in the local neighborhood of the tracking road region. The third image shows the largest number of inliers by RANSAC where samples of the FAST-corner pairs are drawn only from the shown local neighborhood. **The third row**: The first image shows the aligned previous frame. The second image shows the difference image between the previous frame and the current one, and the last image shows the difference between the aligned previous frame and the current one.

matrix H. Suppose the road contour in the previous frame is C_t. The road contour tracked in the current frame is then $C_{t+1}^H = HC_t$. The transformed contour achieves the road tracking in the current frame. Fig. 1(b) shows an example.

Compared with a previous frame, there are incoming regions with incoming road areas in a current frame. The incoming road area R_{t+1}^{in} need to be extracted and then combined with the alignment result R_{t+1}^H as the final road area $R_{t+1} = R_{t+1}^{in} \bigcup R_{t+1}^H$ in a current frame, shown as in Fig. 1. Obviously, R_{t+1}^H is obtained from C_{t+1}^H, and C_{t+1} is then the contour of R_{t+1}.

As UAVs often fly forward, the incoming road should appear near the top of the aligned road. We thus define the ROI centered by the top point of the median axis of C_{t+1}^H. Refer to Fig. 1(c) for an illustration. The median axis is obtained by performing distance map calculation, skeleton, and noise removing. After that, the algorithm in Section IV is applied in ROI.

1) Correction of Drift Error in Tracking: As shown in Fig. 6, the drift errors tend to be very small in the first several dozen frames, and become larger with more incoming frames. The drift errors arise due to two factors: the first one is the inaccuracy in homography estimation. The second one is the accumulation of round errors even when homography estimation is correct, because the transformation between two consecutive frames is slight and thus, the calculation is sensitive to round error. Large drift errors could arise with the accumulation of very small ones.

The large errors cause the aligned road area drifts far from the real road location. We propose a solution to deal with it. For the drift error caused by round error, considering that the drift errors are very small in the initial several dozen frames, the road regions in the first 25 frames are still tracked based on homography alignment between two consecutive frames. But for the subsequent frames, we do the road tracking by estimating the transformation between the current frame and the last tenth frame instead of the previous one, as shown in Figs. 6 and 17. The reason is that, RANSAC uses a small fixed threshold to control the inliers for the optimal transformation estimation. When the transformation between two consecutive frames is too small, RANSAC tends to include some false inliers for transformation estimation. Therefore, we prefer to estimate the transformation between two frames that are z frames apart. We can learn the "z" based on the median magnitude of the motion vectors shown in Fig. 9. For our application, z is set to 10.

In addition, to avoid large drift error, each tracking result is evaluated according to a criteria given in the following, so that if the tracking is unsuccessful, online GraphCut detection is applied instead. Specifically, we propose to compare the corresponding RGB histogram of the current tracking result with an average RGB histogram obtained from the tracking results of the last 10 frames. If the difference is above a threshold, it means that the tracking result drifts too far from the real road area. The histograms are built with 5 bins for each color channel. In each bin, it records the percentage of the pixel number in the bin over the total number of pixels. Moreover,

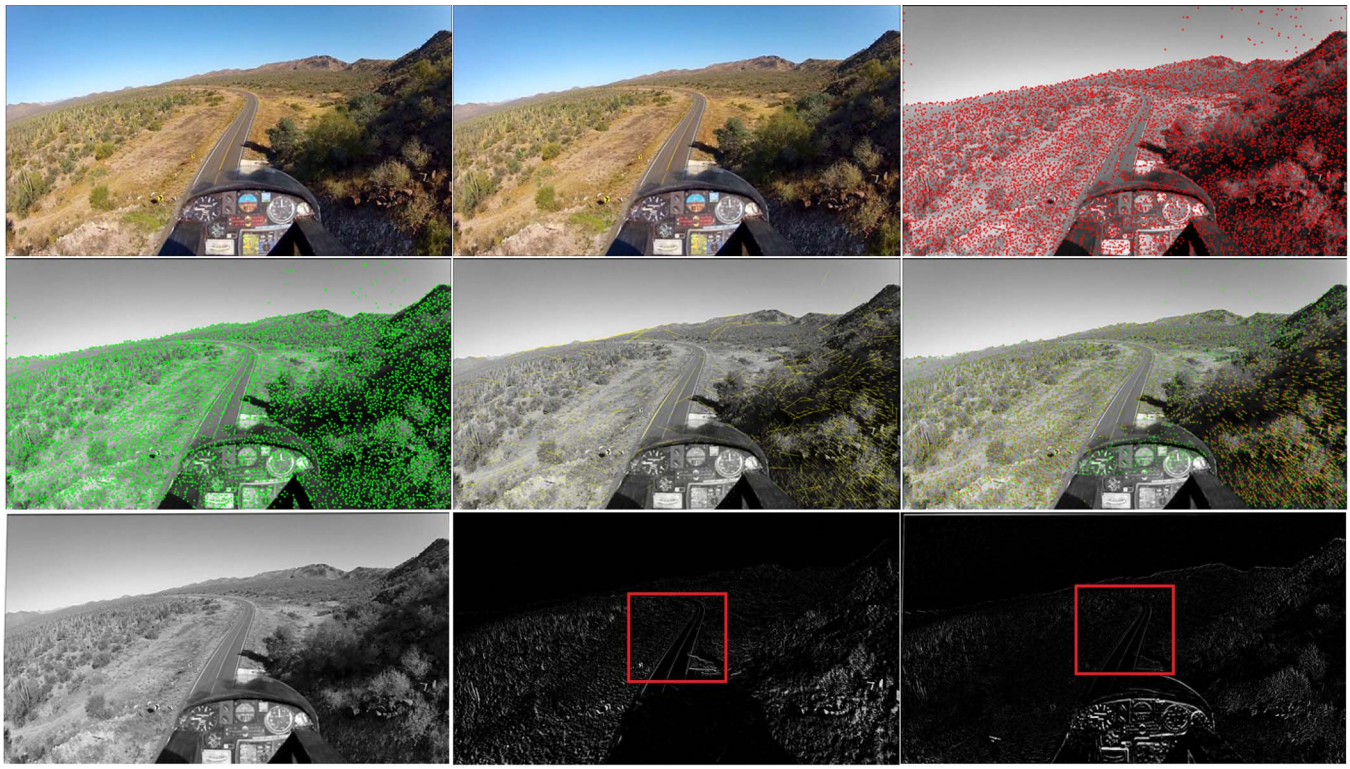

Fig. 11. Illustration of the homography alignment based on the conventional RANSAC. **The first row**: the first two images show the two consecutive frames (the previous and current ones), and the third image shows the detected FAST corners in the previous frame. **The second row**: the first image shows the FAST corners detected in the current frame. The second image shows the initial correspondence of the FAST corners by the correlation-based matching. The third image shows the largest number of inliers by RANSAC. **The third row**: The first image shows the aligned previous frame. The second image shows the difference image between the previous frame and the current one, and the last image shows the difference between the aligned previous frame and the current one. By checking the difference image in Fig. 10, the alignment in road area is much more accurate in our context-aware scheme.

large accumulative error in tracking may also cause obviously zigzag contours, as shown in Fig. 18. We check each road contour. If the change of contour length is above a threshold, tracking results are treated as unsuccessful.

VI. Experimental Results

Experiments are mainly conducted on image sequences acquired using our UAV that flew in different sessions near our center in Australia. The other image sequences downloaded from the Internet are also used for more evaluation on different scenarios. In the test data set, there are 2760 images from six videos with different resolutions, including 1280×720, 1244×748, 1024×576, and 848×480. Due to the limitation of used UAV flight, we could not capture videos of unpaved roads. Roads used in our experiments are paved. Diverse situations such as slow/fast UAV movements, low/high-altitude flying, existence of a lot of shadows, and large variations on image scenes are included in the data set. The colors and shapes of road are different from each other. The data set can be downloaded from https://sites.google.com/site/hailingzhouwei/.

In this paper, quantitative evaluations are provided by comparing the results with ground truth pixelwisely using two measurements: 1) precision $Q = TP/(TP + FP)$, which is the percentage of correctly classified road pixels over the total detected road pixels; and 2) error rate $ER = (FP + FN)/(TP + FN)$, which is the percentage of wrong classified image pixels over the ground-truth road area. TP, FP, and FN, respectively, are true positive, false positive, and false negative. Results with high Q and low ER are desired. The ground truth road areas are manually specified on 400 images evenly sampled from our test videos. All the experiments are conducted using C++ implementation on Intel i7-2600K 3.40-GHz CPU without parallelization and code optimization.

There are some parameters introduced in our method. The values chosen in our experiments are described as follows. The λ in (1) is suggested to be 50 as in [36]. The κ in (8) is chosen to be $EP(s(v_{xy}, v_{ij}) + s(v_{ij}, v_{xy}))$. The ϵ in (9) is 0.1. The K_0 and K_1 are set to be 5 and 3 for nonroad and road GMMs. The size of downsampled images in GraphCut detection is the half of original images. The interval of updating GMM is 40 frames. The size of the incoming ROI is $w \times h$ with $h = 60$ and $w = 6d$, where d is the average distance of points on the median axis to the contour. In the tracking evaluation for drift-error correction, RGB histograms with 5 bins for each channel are defined, the threshold for the average histogram difference set to 12%, and the threshold for the change of contour length is 15% of the previous contour.

A. Road Detection

To evaluate the proposed road detection technique, we perform road detection frame by frame without tracking. It should be noted that in our approach, road detection is first run on the

TABLE I
PERFORMANCE EVALUATION OF THE PROPOSED DETECTION ALGORITHM

	GMM based	GraphCut	GraphCut with structure tensor
Q	94.80%	98.93%	99.45%
ER	5.78%	3.95%	3.77%
T	0.142s	0.271s	0.283s

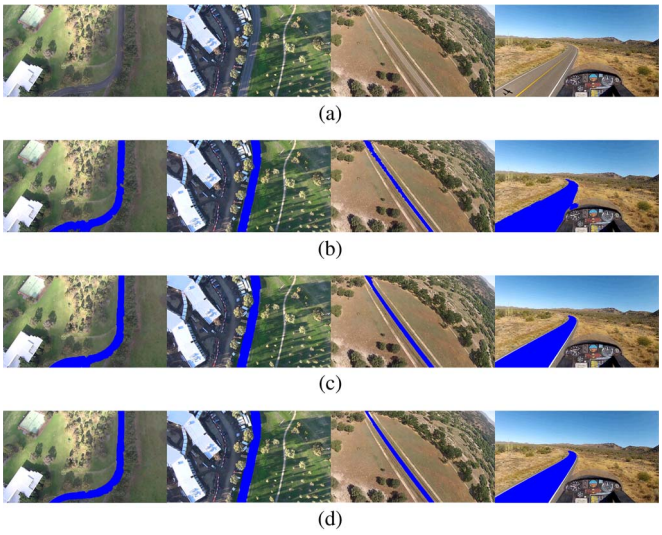

Fig. 12. Comparison on road detection using GMM, Graphcut without and with structure tensor. Note that compared with the GMM method, GraphCut-based methods can guarantee smooth boundaries, and the use of structure tensor can assist GraphCut to align segmentation boundaries with real road borders. It should be also noted that for the image (the last column) with clear road borders, GraphCut without structure tensor can produce almost the same result as the one with structure tensor due to the very high value of $\|v_{xy} - v_{ij}\|$ in (9).

image downsampled to a half resolution and then the final result is obtained by scaling the intermediate result to the full-size images.

The first experiment is to evaluate the technical components of the proposed detection approach: GMM modeling, structure tensor, and GraphCut. We compare their performances to road detection in terms of average error rates ER, precision Q and times T. Table I shows the comparison results. The visual comparisons are given in Fig. 12. Note that the term of GMM based detection in Table I comes from [4] that detects road based on the probability map of (6). From the results, we can see that our proposed approach can produce more accurate results and smooth segmentation boundary along road borders can be obtained, although the detection is slower than the GMM-based method. Since our road tracking is highly dependent on the detection result, in this stage the accuracy of detection is more important than efficiency.

The second experiment is on the number of GMM components. Different numbers for K_0 and K_1 are tested in road detection. The results are shown in Fig. 13, where statistics of average detection error rates and timings (i.e., ER and T) is shown for comparison. The result indicates that $K_0 = 5$ and $K_1 = 3$ achieve the best balance between efficiency and accuracy.

Fig. 14 compares the performances of the static GraphCut and online GraphCut in road detection. It can be seen that due

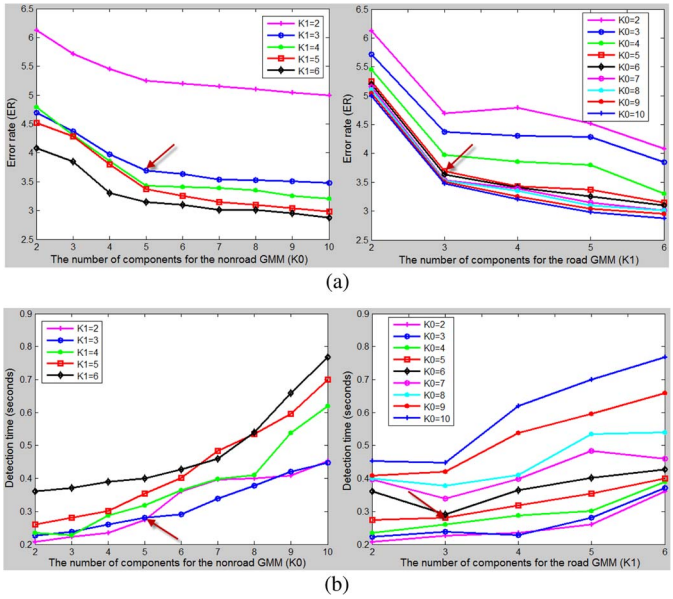

Fig. 13. Statistics of detection accuracy [error rate (a) and times in seconds (b)] using a different number of GMM components.

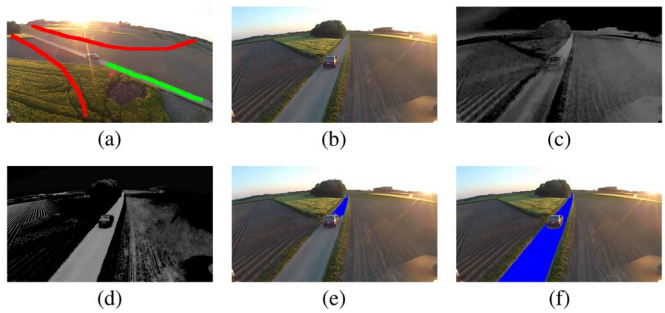

Fig. 14. Static and online GraphCut. (a) Road and nonroad pixels collected from a training frame. (b) Testing 400th frame. (c) and (d) Color modeling using static and dynamic GMMs, and the map show the probabilities of pixels belonging to road calculated by using (6), (e), (f) the detection results based on the static and online GraphCut.

to flight, colors of road, and background scene change a lot. As the dynamic GMM adaptively updates the model using new road and nonroad pixels, it ensures more robust performance. Road detection on the original and half-sized is evaluated in Fig. 15. The quantitative results indicate that the reduction of image size does not change much accuracy of road detection and does speed-up road detection. Fig. 16 shows more road detection results on various images.

B. Road Tracking

Road tracking is evaluated on video sequences when the detected roads are given. The first experiment is on the homography-alignment-based tracking schemes. We compare our tracking approach with the traditional mechanism using FAST detection in images, correlation-based matching and homography estimation based on all matched point pairs. The corresponding times T, precisions Q and average error rates ER are compared in Table II, as well as the time for each step. To calculate the Q and ER, the homography-based alignment

Fig. 15. Comparison of detections on the original and half-sized images. Left: original testing images with the sizes of 1024 × 576 (top) and 1280 × 720 (bottom); Middle: detection results on the original size with $Q = 98.8\%$, $E = 5.21\%$, $T = 1.16$ s (top) and $Q = 99.7\%$, $E = 3.28\%$, $T = 2.43$ s (bottom); Right: detection on the half-size images and then scaled to the full-sized ones with $Q = 98.1\%$, $E = 5.22\%$, $T = 0.27$ s (top) and $Q = 99.1\%$, $E = 3.88\%$, $T = 0.42$ s (bottom).

Fig. 16. Road detection in different scenarios captured by our UAV.

TABLE II
PERFORMANCE EVALUATION OF THE PROPOSED HOMOGRAPHY ESTIMATION ALGORITHM

	Traditional method			Our approach		
Steps	FAST	Matching	Global RANSAC	FAST	KLT	context-aware RANSAC
Times	0.008s	1.149s	0.047s	0.004s	0.003s	0.004s
T	1.204s			0.011s		
Q	94.25%			98.41%		
ER	6.22%			4.39%		

is combined with the online GraphCut detection in ROI as the tracking result and then the tracking result is pixelwisely compared with the groundtruth. From the result in Table II, we can see that our homography-alignment-based technique is much faster and more accurate. The reason is the introduction of the KLT tracker and context-aware RANSAC. The further experiment on the context-aware homography estimation scheme is given in Figs. 10 and 11. We can see that the alignment of road regions based on our context-aware homography-alignment-based tracking is more accurate than the one obtained by the traditional homography estimation. The reason can be due to the fact that some objects such as b ushes, tress, or highland (hills) play a nontrivial role in homography estimation through the common RANSAC. However, these high objects cannot be treated as lying in the same plane with road areas, and are not desirable for the estimation of an optimal homography to accurately align road regions only. In addition, our context-aware homography estimation scheme is much faster than the common RANSAC one, and the computational time is less than 10%.

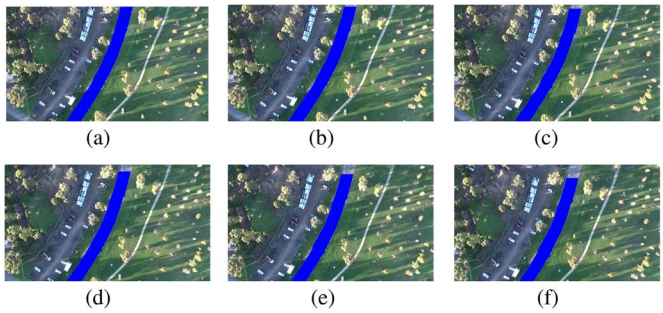

Fig. 17. Correction of the drift problem shown in Fig. 6.

We also experiment different features for homography estimation, including SIFT, Harris, and SURF. Amongst them, SURF has been widely used for an accurate homography estimation. However, it takes 0.142 s per frame for the detection of SURF features. FAST has the best overall performance (as studied in [14]), where the FAST feature detection only takes 0.008 s per frame.

Fig. 17 shows homography-alignment road tracking on frames with 10 intervals in-between, in comparison with Fig. 6. With the reasonable interval, homography estimation becomes more robust to calculation (round) error. Thus, road tracking can be run successfully on more frames without drift-error correction. For the consecutive frame-based estimation, road tracking often fails after 10–30 frames, and for the interval-based estimation, road tracking fails after 100–300 frames.

C. Road Detection and Tracking

The proposed technique of road detection and tracking is experimented as a whole in this section. The technique is tested on UAV videos acquired at different altitudes and fly speeds (For our UAV, we fly with absolute attitudes of 130–170 m and ground speeds of 5–10 m/s). Some results are shown in Fig. 18. More sequences can be found from the supplementary file. Accurate road areas are provided in our results. The switch between road detection and tracking only happens when there are too much changes of color distributions or/and contour lengths of tracked roads, and it can successfully solve the drift and zigzag problems.

Table III lists the running time of previous approaches on road detection and tracking. The work of Frew *et al.* [4] is implemented by us using static GMM on original video frames with the 848 × 480 resolution. The other statistics are obtained from the papers. Our proposed approach achieves the fastest road detection and tracking. We test 2760 frames in total with the resolution of 1046 × 595 on average. The mean detection time is 0.283s and tracking time is 0.028 s per frame. There are 16 frames, where road detection is required, including the initial detection and switching from unsuccessful tracking. The average running time of our approach is 0.029 s per frame.

VII. CONCLUSION

In this paper, a novel approach for road detection and tracking in UAV videos has been proposed. We utilize the static GraphCut segmentation to extract initial road areas, and

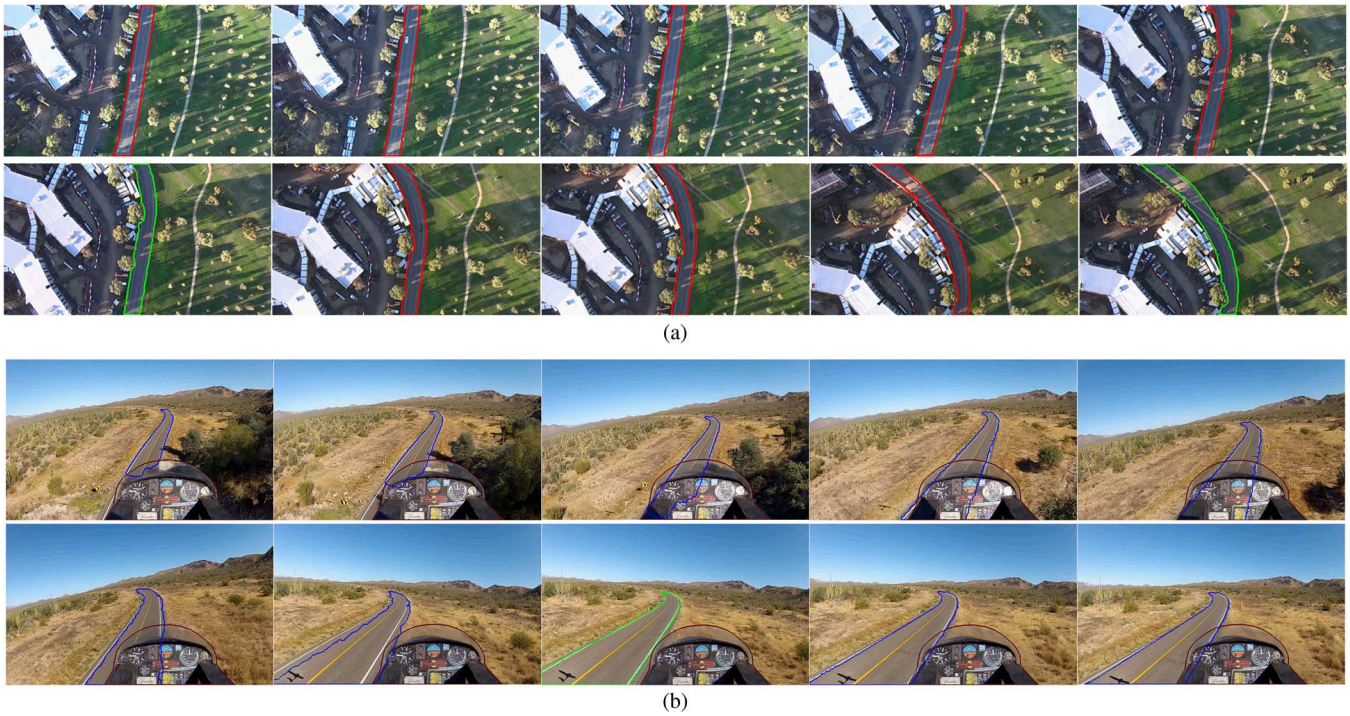

Fig. 18. Road detection and tracking results on different UAV videos. Green road contours are detection results that are switched from tracking. (a) Video captured in a high altitude and medium movements. (b) Video captured in a low altitude and very fast movements.

TABLE III
PERFORMANCE COMPARISON OF ROAD DETECTION AND TRACKING TECHNIQUES

Methods	Running time	Image resolution
Wang *et al* [12]	4s	240×256
Kim *et al* [25]	0.06s	352×240
Guo *et al* [29]	0.125s	320×240
Alvarez *et al* [28]	0.6s	640×480
Siogkas *et al* [7]	0.35s	640×480
Frew *et al* [4]	0.35s	848×480
Our approach	0.029s	1046×595

then track road areas in subsequent frames by combining a fast context-aware homography-alignment road tracker and an online GraphCut approach for road detection in incoming ROIs. Fast road detection and tracking is achieved in our proposed method. Efficiency and effectiveness of the proposed tracking technique are demonstrated in our experiments.

Based on our experiments, we notice that drift error and zigzag contour problems more often happen in UAVs at low altitudes and in high speeds. The reason is that (1) there are more high objects that are not in the same road plane when UAVs flying at low altitude, which affect the homography estimation; (2) the condition of KLT tracker (the linearization of the optimization equation through Taylor expansion at very local region) is not satisfied when UAVs fly very fast. The proposed context-ware homography estimation solves the problem from (1), but it is still difficult for urban areas with high buildings along road sides. Switching from road tracking to detection can address the problem from (2), but it would be much slower since detection is more computationally complex than tracking. Solving the drift problem is interesting work in the future. More intelligent methods similar to key frame-based [42] or incremental structure-from-motion [43] can be investigated.

ACKNOWLEDGMENT

The authors would like to thank M. Fielding for flying UAV.

REFERENCES

[1] B. Coifman, M. McCord, M. Mishalani, and K. Redmill, "Surface transportation surveillance from unmanned aerial vehicles," in *Proc. 83rd Annu. Meet. Transp. Res. Board*, 2004, pp. 1–9.
[2] [Online]. Available: http://www.list.ufl.edu/ATSS.htm
[3] [Online]. Available: http://soliton.ae.gatech.edu/people/ejohnson/
[4] E. Frew *et al.*, "Vision-based road-following using a small autonomous aircraft," in *Proc. IEEE Aerosp. Conf.*, Mar. 2004, vol. 5, pp. 3006–3015.
[5] H. Zhou, D. Creighton, L. Wei, D. Y. Gao, and S. Nahavandi, "Video driven traffic modeling," in *Proc. IEEE/ASME Int. Conf. Adv. Intell. Mechatronics*, Jul. 2013, pp. 506–511.
[6] Y. He, H. Wang, and B. Zhang, "Color-based road detection in urban traffic scenes," *IEEE Trans. Intell. Transp. Syst.*, vol. 5, no. 4, pp. 309–318, Dec. 2004.
[7] G. K. Siogkas and E. S. Dermatas, "Random-walker monocular road detection in adverse conditions using automated spatiotemporal seed selection," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 2, pp. 527–538, Jun. 2013.
[8] Y. Boykov, O. Veksler, and R. Zabih, "Fast approximate energy minimization via graph cuts," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 23, no. 11, pp. 1222–1239, Nov. 2001.
[9] D. Comaniciu, V. Ramesh, and P. Meer, "Real-time tracking of non-rigid objects using mean shift," in *Proc. IEEE Comput. Vis. Pattern Recog.*, 2000, pp. 142–149.
[10] K. Nummiaro, E. Koller-Meier, and L. Van Gool, "An adaptive color-based particle filter," *Image Vis. Comput.*, vol. 21, no. 1, pp. 99–110, Jan. 2003.
[11] B. K. P. and B. G. Schunck, "Determining optical flow," *Artif. Intell.*, vol. 17, no. 1, pp. 185–203, Aug. 1981.
[12] Y. Wang, E. K. Teoh, and D. Shen, "Lane detection and tracking using B-Snake," *Image Vis. Comput.*, vol. 22, no. 4, pp. 269–280, Apr. 2004.
[13] Y. Wang, D. Shen, and E. K. Teoh, "Lane detection using spline model," *Pattern Recog. Lett.*, vol. 21, no. 8, pp. 677–689, Jul. 2000.
[14] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," in *Proc. Eur. Conf. Comput. Vis.*, 2006, pp. 430–443.
[15] C. Tomasi and T. Kanade, "Detection and tracking of point features," in *School of Computer Science*. Pittsburgh, PA, USA: Carnegie Mellon Univ., 1991.

[16] S. Rathinam, Z. Kim, and R. Sengupta, "Vision-based monitoring of locally linear structures using an unmanned aerial vehicle 1," *J. Infrastructure Syst.*, vol. 14, no. 1, pp. 52–63, Mar. 2008.

[17] S. Rathinam *et al.*, "Autonomous searching and tracking of a river using an UAV," in *Proc. IEEE Am. Control Conf.*, Jul. 2007, pp. 359–364.

[18] Y. Lin and S. Saripalli, "Road detection from aerial imagery," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2012, pp. 3588–3593.

[19] Y. Lin and S. Saripalli, "Road detection and tracking from aerial desert imagery," *J. Intell. Robot. Syst.*, vol. 65, no. 1–4, pp. 345–359, Jan. 2012.

[20] A. Angel, M. Hickman, P. Mirchandani, and D. Chandnani, "Methods of analyzing traffic imagery collected from aerial platforms," *IEEE Trans. Intell. Transp. Syst.*, vol. 4, no. 2, pp. 99–107, Jun. 2003.

[21] B. Coifman, M. McCord, R. G. Mishalani, M. Iswalt, and Y. Ji, "Roadway traffic monitoring from an unmanned aerial vehicle," *Proc. Inst. Elect. Eng.- Trans. Intell. Transp. Syst.*, vol. 153, no. 1, pp. 11–20, Mar. 2006.

[22] A. P. DalPoz, R. A. B. Gallis, J. F. C. daSilva, and E. F. O. Martins, "Object-space road extraction in rural aeras using stereoscopic aerial images," *IEEE Geosci. Remote Sens. Lett.*, vol. 9, no. 4, pp. 654–658, Jul. 2012.

[23] J. Hu, A. Razdan, J. C. Femiani, M. Cui, and P. Wonka, "Road network extraction and intersection detection from aerial images by tracking road footprints," *IEEE Trans. Geosci. Remote Sens.*, vol. 45, no. 12, pp. 4144–4157, Dec. 2007.

[24] C. Unsalan and B. Sirmacek, "Road network detection using probabilistic and graph theoretical methods," *IEEE Trans. Geosci. Remote Sens.*, vol. 50, no. 11, pp. 4441–4453, Nov. 2012.

[25] Z. W. Kim, "Robust lane detection and tracking in challenging scenarios," *IEEE Trans. Intell. Transp. Syst.*, vol. 9, no. 1, pp. 16–26, Mar. 2008.

[26] H. Kong, J.-Y. Audibert, and J. Ponce, "Vanishing point detection for road detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, Jun. 2009, vol. 2, pp. 96–103.

[27] H. Kong, J. Y. Audibert, and J. Ponce, "General road detection from a single image," *IEEE Trans. Image Process.*, vol. 19, no. 8, pp. 2211–2220, Aug. 2010.

[28] J. M. Alvarez and A. M. Lopez, "Road detection based on illuminant invariance," *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 1, pp. 184–191, Mar. 2011.

[29] C. Guo, S. Mita, and D. McAllester, "Robust road detection and tracking in challenging scenarios based on Markov random fields with unsupervised learning," *IEEE Trans. Intell. Transp. Syst.*, vol. 13, no. 3, pp. 1338–1354, Sep. 2012.

[30] A. Xu and G. Dudek, "A vision-based boundary following framework for aerial vehicles," in *Proc. IEEE Int. Conf. Intell. Robots Syst.*, Oct. 2010, pp. 81–86.

[31] C. Rother, V. Kolmogorov, and A. Blake, "GrabCut-interactive foreground extraction using iterated GraphCuts," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 309–314, Aug. 2004.

[32] Y. Boykov and M. Jolly, "Interactive graphcuts for optimal boundary and region segmentation of objects in n-d images," in *Proc. Int. Conf. Comput. Vis.*, 2001, pp. 105–112.

[33] J. F. Talbot, Implementing Grabcut. [Online]. Available: http://www.justintalbot.com/

[34] S. J. Sangwine and R. E. N. Horne, *The Color Image Processing Handbook*. New York, NY, USA: Springer-Verlag, 1998.

[35] M. T. Orchard and C. A. Bouman, "Color quantization of images," *IEEE Trans. Signal Process.*, vol. 39, no. 12, pp. 2677–2690, Dec. 1991.

[36] H. Zhou, J. Zheng, and L. Wei, "Texture aware image segmentation using GraphCuts and active contours," *Pattern Recog.*, vol. 46, no. 6, pp. 1719–1733, Jun. 2013.

[37] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, vol. 2. Cambridge, U.K.: Cambridge Univ. Press, 2000.

[38] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, Jun. 1981.

[39] H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool, "SURF: Speeded up robust features," *Comput. Vis. Image Underst.*, vol. 110, no. 3, pp. 346–359, Jun. 2008.

[40] D. G. Lowe, "Object recognition from local scale-invariant features," in *Proc. IEEE Int. Conf. Comput. Vis.*, 1999, vol. 2, pp. 1150–1157.

[41] C. Harris and M. Stephens, "A combined corner and edge detector," in *Proc. Alvey Vis. Conf.*, 1988, pp. 147–151.

[42] A. Rahimi, L. Morency, and T. Darrell, "Reducing drift in differential tracking," *Comput. Vis. Image Understanding*, vol. 109, no. 2, pp. 97–111, Feb. 2008.

[43] C. Wu, "Towards Linear-Time Incremental Structure from Motion," in *Proc. IEEE Int. Conf. 3-D Vis.*, Jun./Jul. 2013, pp. 127–134.

Hailing Zhou (M'13) received the B.Eng. degree (with honors) in computer science from Xidian University, Xian, China, in 2006 and the Ph.D. degree from Nanyang Technological University (NTU), Singapore, in 2012.

In 2012 she was a Project Officer working on object-based image editing with NTU. She is currently a Research Fellow with the Centre for Intelligent Systems Research, Burwood, Australia. Her main areas of research are computer vision, image processing, human–computer interaction, and computer graphics.

Hui Kong received the Ph.D. degree from Nanyang Technological University, Singapore, in 2007.

From 2008 to 2009 he was a Postdoctoral Researcher with the Willow Team, École Normale Supérieure and INRIA, Paris, France. From 2010 to 2011 he was a Research Scientist working on medical image analysis with Ohio State University Medical Center, Columbus, OH, USA. He is currently a Research Scientist with Massachusetts Institute of Technology, Cambridge, MA, USA, working on computer vision-based field scanning projects. His research interests include computer vision, image processing, and machine learning.

Lei Wei (M'13) received the B.Eng. degree in computer science from Tianjin University, Tianjin, China, and the Ph.D. from Nanyang Technological University (NTU), Singapore, in 2006 and 2011, respectively.

He was a Research Associate with NTU before he started as a Postdoctoral Research Fellow with the Centre for Intelligent Systems Research, Burwood, Australia. His research interests include haptic rendering, haptic collision detection, networked haptics, medical haptics, human–computer interaction, and image processing.

Douglas Creighton (M'10) received the B.Eng. degree (with honors) in systems engineering and the B.Sc. degree in physics from Australian National University, Canberra, Australia, in 1997 and the Ph.D. degree from Deakin University, Burwood, Australia, in 2004.

He is an Associate Professor and Deputy Director of the Centre for Intelligent Systems Research, Deakin University, Burwood. His research focus is on algorithms and methodologies to improve estimation and performance of complex systems operating under uncertainty, variability, and continuous change.

Saeid Nahavandi (SM'07) received the Ph.D. from Durham University, Durham, U.K.

He is an Alfred Deakin Professor, Chair of Engineering, and the Director of the Center for Intelligent Systems Research at Deakin University, Burwood, Australia. He has published over 450 papers in various international journals and conferences. His research interests include modeling of complex systems, robotics, and haptics. He is the Co-Editor-in-Chief of IEEE SYSTEMS JOURNAL and an Editor (South Pacific Region) of *International Journal of Intelligent Automation and Soft Computing*. He is a Fellow of Engineers Australia and the Institution of Engineering and Technology.