Unsupervised Single Moving Object Detection Based on Coarse-to-Fine Segmentation

Xiaozhou Zhu [1], Xin Song [1] Xiaoqian Chen [1] and Huimin Lu [2]
[1] College of Aerospace Science and Engineering, National University of Defense Technology
Changsha, 410073, P. R. China
[e-mail: work_ranger@163.com, song_xin@139.com, chenxiaoqian@nudt.edu.cn]
[2] College of Mechatronics and Automation, National University of Defense Technology
Changsha, 410073, P. R. China
[e-mail: lhmnew@nudt.edu.cn]
*Corresponding author: Xiaoqian Chen

Received December 10, 2015; revised March 30, 2016; accepted April 27, 2016; published June 30, 2016

Abstract

An efficient and effective unsupervised single moving object detection framework is presented in this paper. Given the sparsely labelled trajectory points, we adopt a coarse-to-fine strategy to detect and segment the foreground from the background. The superpixel level coarse segmentation reduces the complexity of subsequent processing, and the pixel level refinement improves the segmentation accuracy. A distance measurement is devised in the coarse segmentation stage to measure the similarities between generated superpixels, which can then be used for clustering. Moreover, a Quadmap is introduced to facilitate the refinement in the fine segmentation stage. According to the experiments, our algorithm is effective and efficient, and favorable results can be achieved compared with state-of-the-art methods.

Keywords: Moving object detection, motion segmentation, superpixel, spectral clustering

A preliminary version of this paper appeared in IEEE ICIA 2015, August 8-10, Lijiang, China. This version is an extension with the following additions: the targets to detect are extended from flying spacecrafts (rigid bodies) to general moving objects; a new coarse-to-fine framework is proposed: some procedures of the preliminary version are modified to form the superpixel level coarse segmentation stage, and a pixel level fine segmentation stage is introduced; much more experiments on more datasets are performed to compare our approach with state-of-the-art approaches both quantitatively and qualitatively. This research was supported by the National Natural Science Foundation of China (Grant No. 61401487). We express our thanks to University of Freiburg for providing the Motion Segmentation Dataset.

http://dx.doi.org/10.3837/tiis.2016.06.012 **ISSN : 1976-7277**

1. Introduction

Moving object detection aims to detect and segment moving objects from the scene. It has found a great number of real world applications, including navigation, surveillance, recognition, 3D reconstruction and many more. However, it is still full of great challenges due to the changing appearance of the objects and the probable highly dynamic and cluttered background.

Existing literatures for moving object detection/segmentation can be broadly classified into two categories, i.e., background subtraction methods and motion cue based approaches. The former kind of methods take advantages of pixel-level features (Gaussian Mixture Model [1], Codebook [2], etc.) or texture features (Local Binary Patterns [3], Scale Invariant Local Ternary Patterns [4], etc.) to construct an appearance model for the background. Then the constructed background model is subtracted in the new frame to make a comparison. And pixels in the new frame with differences larger than a certain threshold are taken as the detected objects. However, these methods cannot achieve desirable results in scenarios with fast moving camera and highly dynamic background. Besides, Sabhara *et al*. [5] investigated the use of Hu Moments and Zernike Moments in object detection. And thorough surveys on background subtraction methods can be found in [6, 7].

In contrast, the latter kind of algorithms detect moving objects by using different types of motion cues. A kind of widely used motion cues is optical flow computed between adjacent frames. Papazoglou *et al*. [8] first initialized the boundary of the moving object by using optical flow. Pixels within the boundary are labelled as foreground, while pixels out of the boundary are taken as background. They then separate the foreground from the background by iteratively refining the foreground-background labelling. However, when insufficient displacements, occlusions, low-texture regions, etc. occur, the calculation of optical flow may fail.

Another kind of motion cues which gains popularity in recent years is labelled point trajectories. To generate point trajectories, feature points are extracted in the image and then tracked across successive frames. Considering that point trajectories from different objects undergo different motions, they can be classified into different clusters and labelled by using motion segmentation algorithms, including factorization [9], RANSAC [10, 11], spectral clustering [12-14], *etc*. Trajectory points with the same label come from the same object and this cue can be used in detection/segmentation.

In [10] and [11], the authors construct appearance models for the foreground and the background respectively, taking advantage of the sparsely labelled point trajectories. Then the transfer from sparse labelling to per-pixel labelling is achieved by maximizing a posteriori. However, manipulating tens of thousands of pixels directly is computational heavy and takes lots of time.

To circumvent the problem, superpixels can be used as a pre-processing [14-17]. Firstly introduced by *Ren et al.* [18], superpixel is composed of pixels which are similar according to certain criteria, color for example. Dealing with hundreds of superpixels instead of tens of thousands of pixels can drastically reduce the complexity of subsequent processing. One can refer to [19] for more details about different superpixel segmentation algorithms.

In [14], Ochs *et al*. first obtain labelled point trajectories by using the semi-dense method proposed in [20]. A hierarchical image segmentation (HIS) [21] is also conducted to generate hierarchical superpixels segmentation. They then merge these superpixels hierarchically by

using a multi-level variational approach and extend the semi-dense labelling of tracked points to dense labelling of the whole image. This approach is considered to be state-of-the-art due to the promising results acquired. However, the computational burden is heavy and the calculation of HIS [21] is rather memory consuming, which confine its use to off-line applications. Ellis *et al*. [17] propose an online learning algorithm for moving objects segmentation. The sparse labelled point trajectories are used to provide both labelled samples for learning the appearance cue and labelled spatial coordinates for extracting shape-location cue. Then learning and classification are performed at multi-scale superpixel level by using Online Random Forest (ORF), taking advantage of all these cues. However, the boundary of superpixels may not adhere well to the object boundary even if the multi-scale strategy is adopted.

In this work, a novel coarse-to-fine framework for unsupervised detection of single moving object from videos or image sequences is proposed. Our algorithm is composed of a superpixel level coarse segmentation to facilitate subsequent processing as well as to provide a sound initialization, and a pixel level fine segmentation to improve the segmentation accuracy. A distance measurement is devised to measure the similarities between generated superpixels, which is of great significance for clustering in the coarse segmentation stage. Moreover, a Quadmap is introduced to facilitate the refinement in the fine segmentation stage.

The outline of our paper is as follows. We detail the moving object detection method step by step in Section 2. In Section 3 we conduct both quantitative and qualitative experiments. And Section 4 conclude the paper.

2. Our Proposed Algorithm

The framework of our algorithm is shown in **Fig. 1**, which includes following three stages:

(1)Point trajectories generation and motion segmentation;
(2)Superpixel level coarse segmentation;
(3)Pixel level fine segmentation.

Firstly, labelled point trajectories are obtained to be used as the motion cue. Then superpixels are generated and labelled as foreground/background/uncertain, taking advantage of the motion cue. To eliminate uncertain regions, superpixels clustering is conducted according to a tailored similarity measurement, which combines distances in color and texture feature spaces as well as the spatial proximity. After that, a coarse segmentation is performed at the superpixel level by adopting a simple strategy, taking both the results of superpixels labelling and clustering into consideration. Moreover, a Quadmap is constructed according to the coarse segmentation, labelling all the pixels as definitely foreground, probably foreground, definitely background and probably background. Finally, the fine segmentation is achieved by a pixel level refinement with the Quadmap as a good initialization.

2.1 Point Trajectories Generation and Motion Segmentation

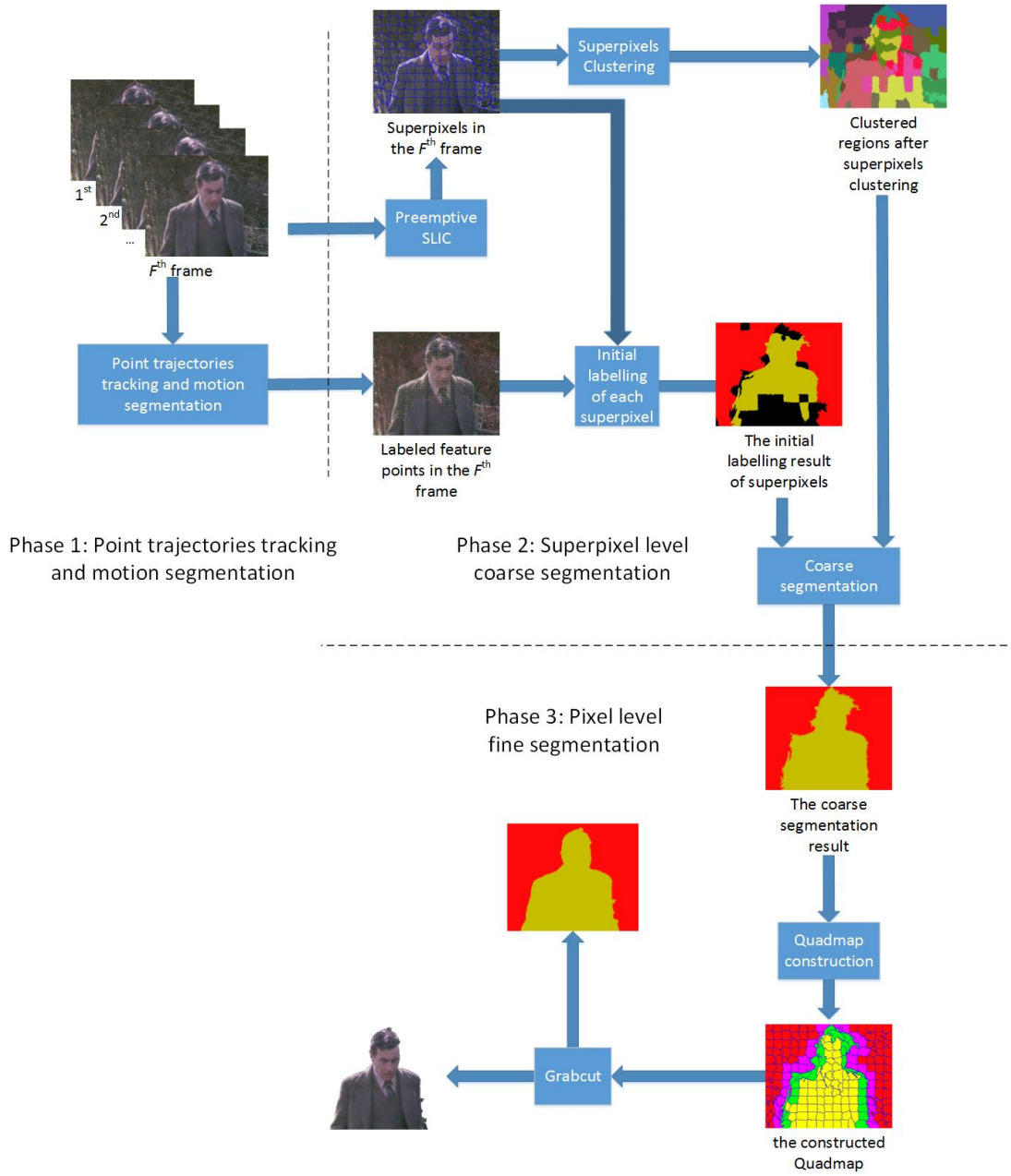


Fig. 1. The framework of our algorithm

We assume that N feature points either from the foreground or the background are tracked in the process. The trajectory of the i^{th} point is represented as $T_i = (u_1^i, v_1^i, ..., u_F^i, v_F^i)^T \in \Re^{2F}$, where (u_f^i, v_f^i) denotes the position of the i^{th} point in the f^{th} image frame. We then group all the N trajectories to build the trajectory matrix T.

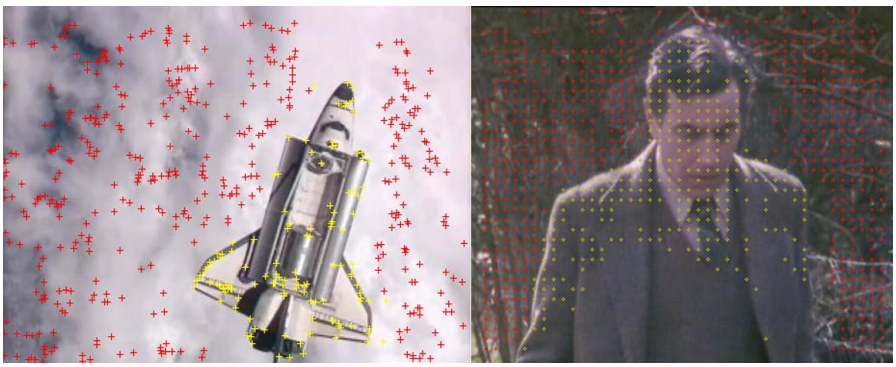
$$T = [T_1, ..., T_i, ..., T_N] = \begin{pmatrix} u_1^1 & \cdots & u_1^N \\ v_1^1 & & v_1^N \\ \vdots & \ddots & \vdots \\ u_F^1 & \cdots & u_F^N \\ v_F^1 & & v_F^N \end{pmatrix}_{2F \times P} \quad (1)$$

Considering that only one moving object occurs in the field of view, trajectories in T are grouped into two clusters by using motion segmentation methods. Trajectories in the same cluster undergo the same motion, and this cue can be used in the subsequent processing.

This stage can be realized by using the KLT + SSC method we proposed in [22], which combines a Kanade-Lucas-Tomasi (KLT) tracker [23] for point trajectories generation and sparse subspace clustering (SSC) [13] for motion segmentation. This method is efficient and performs quite well when dealing with moving rigid bodies. However, the basis of SSC lie in that point trajectories generated by a moving rigid body belong to a subspace whose dimension is at most 4 under the affine camera assumption. And this restricts its application to non-rigid bodies.

Another choice is the semi-dense point trajectories method from [20], where optical flow is used for point tracking and spectral clustering [24-26] is used for motion segmentation. This method is able to deal with both rigid bodies and non-rigid bodies. However, it is time consuming and is more suitable for off-line applications.

One can choose these two or other methods to generate labelled trajectories according to the specific application. In this paper, KLT + SSC method [22] is used when detecting moving rigid bodies for efficiency, the Atlantis spacecraft in **Fig. 2a** for example; while the semi-dense method[20] is adopted when detecting moving non-rigid bodies, the moving person in **Fig. 2b** for example.



(a) (b)

Fig. 2. Labelled points on the F^{th} frame by using different motion segmentation methods, where red points indicate background and yellow points indicate foreground: (a) The result of KLT + SSC method [22]; (b) The result of semi-dense method [20]

As shown in **Fig. 2**, points with the same color come from the same cluster, i.e. foreground or background. Considering the assumption that only one moving object occurs in the field of view, occupying a certain part in the image and surrounded by the background, points more scattered in the image are labelled as background (red points), while points more concentrated are labelled as foreground (yellow points). Here more scattered means that these points are

widely spread and take a larger part of the whole image. While concentrated means that these points are local and take a less part of the whole image.This can be discriminated by firstly finding the smallest rectangle that include the points from one cluster within it and then comparing the area of the two rectangles. And the larger one corresponds to background.

2.2 Superpixel Level Coarse Segmentation

2.2.1 Superpixels Segmentation

Superpixel segmentation is conducted on the frame where the moving object is to be detected with Preemptive SLIC [27], which is a variant of the famous superpixel segmentation method SLIC [28].

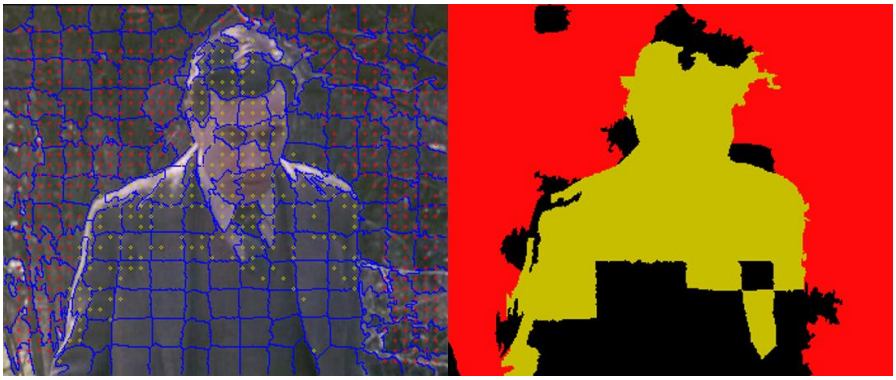
SLIC perform superpixels segmentation by searching locally for similar pixels around the seeded centers initialized. The searching range is confined in a 2S × 2S region to speed up. The similarities are measured according to the pairwise distance $D(p^i, p^j)$ between pixels p^i and p^j defined as [28]

$$\begin{aligned} d_{LAB}^{ij} &= \sqrt{(p_l^i - p_l^j)^2 + (p_a^i - p_a^j)^2 + (p_b^i - p_b^j)^2} \\ d_{spatial}^{ij} &= \sqrt{(p_x^i - p_x^j)^2 + (p_y^i - p_y^j)^2} \\ D(p^i, p^j) &= \sqrt{d_{LAB}{}^2 + (d_{spatial} / S)^2 m^2} \end{aligned} \tag{2}$$

where d_{LAB}^{ij} measures the distance between pixels in the CIELAB color space $[p_l, p_a, p_b]^{\mathrm{T}}$ and $d_{spatial}^{ij}$ measures the spatial proximity by calculating the Euclidean distance of the pixels' coordinates in the image expressed by $[p_x, p_y]^{\mathrm{T}}$. m is the compactness factor used to weight d_{LAB}^{ij} and $d_{spatial}^{ij}$, and the search range S is determined by

$$\sqrt{\frac{\text{Number of pixels}}{\text{Number of superpixels desired}}} \tag{3}$$

Compared with SLIC, a local termination criterion is adopted in Preemptive SLIC. Due to the criterion, searching and updating of clusters which undergo little change are terminated in advance. And thus speed the whole process a lot. **Fig. 3a** shows the superpixels segmentation results along with labelled points in each superpixel.



(a) (b)

Fig. 3. (a) Superpixels segmentation results on the F^{th} frame by using Preemptive SLIC [27], with labelled points in each superpixel; (b) The initial labelling result of superpixels.

2.2.2 Initial Labelling of Each Superpixel

Each superpixel is labelled automatically according to the labelled points within it:
(1)If it only contains red points, then it is labelled as background, shown in red in **Fig. 3b**;
(2)If it only contains yellow points, then it is labelled as foreground, shown in yellow in **Fig. 3b**;
(3)If it contains both red and yellow points, or if it contains no labelled points, then it is labelled as uncertain, shown in black in **Fig. 3b**.

The extracted feature points would be better to be evenly distributed in the first frame and then tracked. It is reasonable to adopt this strategy to make sure that more superpixels would contain labelled points to be used for initial labelling. However, superpixels labelled as uncertain do exist due to tracking failure of feature points, especially when large regions with low texture occur, the sweater area in the bottom of **Fig. 3a** for example. To deal with this problem, we proceed to label the uncertain regions by using superpixels clustering.

2.2.3 Feature Extracting and Superpixels Clustering

We extract color feature and texture feature that complement with each other [29] from each superpixel. One thing need to be mentioned is that the process is applied over the image which contains the superpixels, not over the initially labelled image.

For the color feature, we first transform the representation of pixels from RGB to HSV, which is more robust to changing illumination. Then a color histogram H_c is computed within each superpixel as the color feature, discretizing the hue, saturation, and value channels of HSV color space into 9, 8, and 6 bins. And the Weber's law descriptor (WLD) [30] for image pixels are calculated and normalized into the range [0, 255]. Then a texture histogram H_t of each superpixel is computed as the texture feature.

After feature extraction, superpixels are clustered into k regions by conducting spectral clustering [24-26] on an undirected weighted graph $G=(V,W)$, where V denotes the vertex set with each vertex representing a superpixel, and W denotes the weighted matrix with the entry w_{ij} reflecting the similarity between superpixels S^i and S^j. In order to construct the matrix W, we devise a distance $D(S^i,S^j)$ as follows to measure the proximity between superpixels S^i and S^j [22]:

(1) $d_{color}(S^i,S^j)$ indicates the distance between S^i and S^j in color space by correlating color histogram H_c extracted. $\dim(H_c)$ equals to 432, where $\dim(h)$ denotes the dimensionality of histogram h. And the values of $d_{color}(S^i,S^j)$ range from 0 to 1.

$$d_{color}(S^i,S^j)=(1-correlation(H_c^i,H_c^j))/2 \tag{4}$$

where the correlation between two histograms h^i and h^j is defined as [31]

$$correlation(h^i,h^j)=\frac{\sum_{l=1}^{\dim(h)}(\overline{h^i}-h^{il})(\overline{h^j}-h^{jl})}{\sqrt{\sum_{l=1}^{\dim(h)}(\overline{h^i}-h^{il})^2\sum_{l=1}^{\dim(h)}(\overline{h^j}-h^{jl})^2}} \tag{5}$$

where $\overline{h^i} = (\sum_{l=1}^{\dim(h)} h^{il}) / \dim(h)$.

(2) $d_{texture}(S^i, S^j)$ is the distance between S^i and S^j in texture space, and the computation is the same as $d_{color}(S^i, S^j)$. dim(H_t) equals to 256 and the values of $d_{texture}(S^i, S^j)$ also range from 0 to 1.

$$d_{texture}(S^i, S^j) = (1 - correlation(H_t^i, H_t^j)) / 2 \tag{6}$$

(3) $d_{spatial}(S^i, S^j)$ refers to the city block distance between S^i and S^j

$$d_{spatial}(S^i, S^j) = \| S_x^i - S_x^j \| + \| S_y^i - S_y^j \| \tag{7}$$

where (S_x^i, S_y^i) denotes the coordinate of the center of S^i in the image. $d_{spatial}(S_i, S_j)$ is divided by the maximum value of all the $d_{spatial}(S_i, S_j)$ to be normalized to the range [0, 1].

Then, adaptive weights are used to construct $D(S^i, S^j)$ as follows

$$\begin{aligned} D(S^i, S^j) &= \omega_{color} d_{color}(S^i, S^j) + \omega_{texture} d_{texture}(S^i, S^j) + \omega_{spatial} d_{spatial}(S^i, S^j) \\ \omega_{color} &= \overline{d_{color}} / (\overline{d_{color}} + \overline{d_{texture}} + \overline{d_{spatial}}) \\ \omega_{texture} &= \overline{d_{texture}} / (\overline{d_{color}} + \overline{d_{texture}} + \overline{d_{spatial}}) \\ \omega_{spatial} &= \overline{d_{spatial}} / (\overline{d_{color}} + \overline{d_{texture}} + \overline{d_{spatial}}) \end{aligned} \tag{8}$$

where $\overline{d_{color}}$, $\overline{d_{texture}}$ and $\overline{d_{spatial}}$ is the mean value of all $d_{color}(S^i, S^j)$, $d_{texture}(S^i, S^j)$, and $d_{spatial}(S^i, S^j)$ respectively.

And W can be constructed according to $D(S^i, S^j)$ with each of its element computed as

$$w_{ij} = \exp\left(-D(S^i, S^j)^2 / 2\sigma^2\right) \tag{9}$$

where $\sigma = \max_{i=1}^{n} \min_{j=1}^{n} D(S^i, S^j), i \neq j$.

Fig. 4a shows the clustering result with different colors, and superpixels with the same color belong to the same cluster.

Fig. 4. (a) Clustered regions after superpixels clustering; (b) Coarse segmentation result; (c) The original F^{th} frame

2.2.4 Coarse Segmentation

For each clustered region, the labelled superpixels within vote for the assignment of the uncertain superpixels. We adopt a winner-takes-all strategy:

(1)If the number of superpixels labelled as foreground within the clustered region is more than the number of superpixels labelled as background, then all the uncertain superpixels within the clustered region are labelled as foreground, and vice versa.

(2)If by any chance a draw occurs, i.e. the number of superpixels labelled as foreground equals to the number of superpixels labelled as background, then all the uncertain superpixels within the clustered region are labelled as background.

The result of coarse segmentation is shown in **Fig. 4b**, we can see that the status of uncertain superpixels labelled in 2.2.2 (superpixels in black in **Fig. 3b**) have been updated to foreground (yellow in **Fig. 4b**) or background (red in **Fig. 4b**). Also we can notice that the result is not satisfactory by comparing **Fig. 4b** with **Fig. 4c**. This is caused by the misalignment of superpixels' contour with the true contour of the moving object. To deal with this problem, we proceed to perform pixel level refinement.

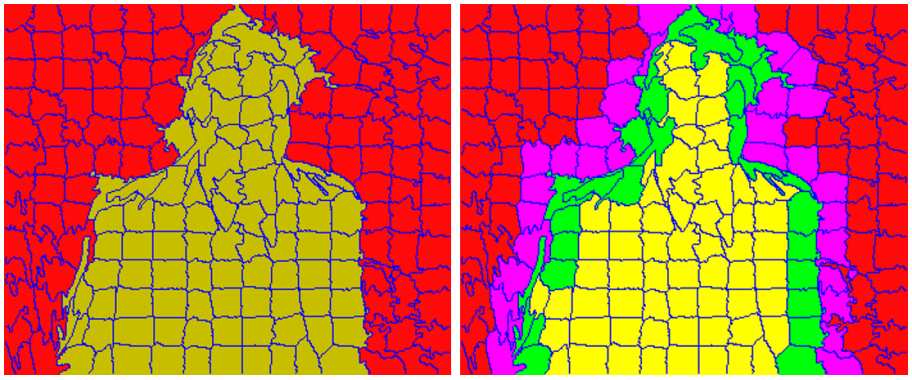
2.3 Pixel Level Fine Segmentation

2.3.1 Quadmap Construction

The Trimap is usually used in image matting, segmenting the input image into three regions: definitely foreground T_f, definitely background T_b and unknown T_u [32]. This three-level pixel map is often generated by user interface, for example manually specifying foreground and background scribbles.

Inspired by the Trimap, we construct the Quadmap automatically by analyzing the 8-connectivity adjacency of labelled superpixels. This is based on the observation that unknown regions only occur in regions where foreground and background are adjoining.

Similar to the Trimap, the Quadmap segments the input image into four regions: definitely foreground D_f, definitely background D_b, probably foreground P_f and probably background P_b.



(a) (b)

Fig. 5. (a) Coarse segmentation of superpixels; (b) The constructed Quadmap

(1)For each superpixel labelled as foreground in coarse segmentation (yellow in **Fig. 5a**), if all its adjacent superpixels are foreground, then it is taken as D_f (yellow in **Fig. 5b**); otherwise it is taken as P_f (green in **Fig. 5b**).

(2)For each superpixel labelled as background in coarse segmentation (red in **Fig. 5a**), if all its adjacent superpixels are background, then it is taken as D_b (red in **Fig. 5b**); otherwise it is taken as P_b (magenta in **Fig. 5b**).

2.3.2 Fine Segmentation

We perform pixel level fine segmentation by using Grabcut [33]. Given the Trimap, Grabcut segments N pixels $z = (z_1 \ldots z_n \ldots z_N)$ in the image into foreground and background. We compare several color spaces including RGB, HSV, CMY, and YUV, and found RGB to be the best color space representation in our case.

The segmentation result is expressed as $\alpha = (\alpha_1 \ldots \alpha_n \ldots \alpha_N)$, where $\alpha_i \in \{0,1\}$ with 0 for background and 1 for foreground. Grabcut finds the optimum segmentation by minimizing the energy function defined as

$$E(\alpha, \theta, z) = U(\alpha, \theta, z) + V(\alpha, z) \tag{10}$$

where $U(\alpha, \theta, \mathrm{z})$ is the data term evaluating the fitness of α to the data z, and $V(\alpha, \mathrm{z})$ is the regularization term encouraging coherence in regions of similar RGB-level. The variables $\theta = (\theta_0, \theta_1)$ describe background and foreground appearance models, where θ_0 and θ_1 are calculated according to pixels labelled as foreground and background by using Gaussian Mixture Models (GMM) respectively.

The minimization can be achieved by alternating optimize over segmentation results α and appearance models θ. T_f and T_b remain unchanged during the process, while T update in each iteration to classify pixels within it to foreground or background.

In our case, the automatically constructed Quadmap is used instead of the interactively labelled Trimap to realize unsupervised segmentation. Moreover, the Quadmap can also provide a better initialization. D_f and D_b remain unchanged, while P_f and P_b update in each iteration. Due to the sound initialization of the Quadmap, good results can be achieved after only 2-3 iterations, as shown in **Fig. 6**.

(a) (b)

Fig. 6. (a) Updated probably foreground and probably background; (b) The detected moving object

3. Experiments and Results

In this section, we first compare our method quantitatively and qualitatively with state-of-the-art methods proposed by Papazoglou *et al*. [8] and Ochs *et al*. [14] by using the Freiburg-Berkeley Motion Segmentation Dataset [14]. Then, our method is applied on image sequences intercepted from a video, in which the Atlantis Space Shuttle is approaching the International Space Station for rendezvous & docking. Our proposed method is implemented in Visual C++ 2012. For Papazoglou *et al*.'s approach [8] and Ochs *et al*.'s method [14], the codes are provided by the authors on their homepage.

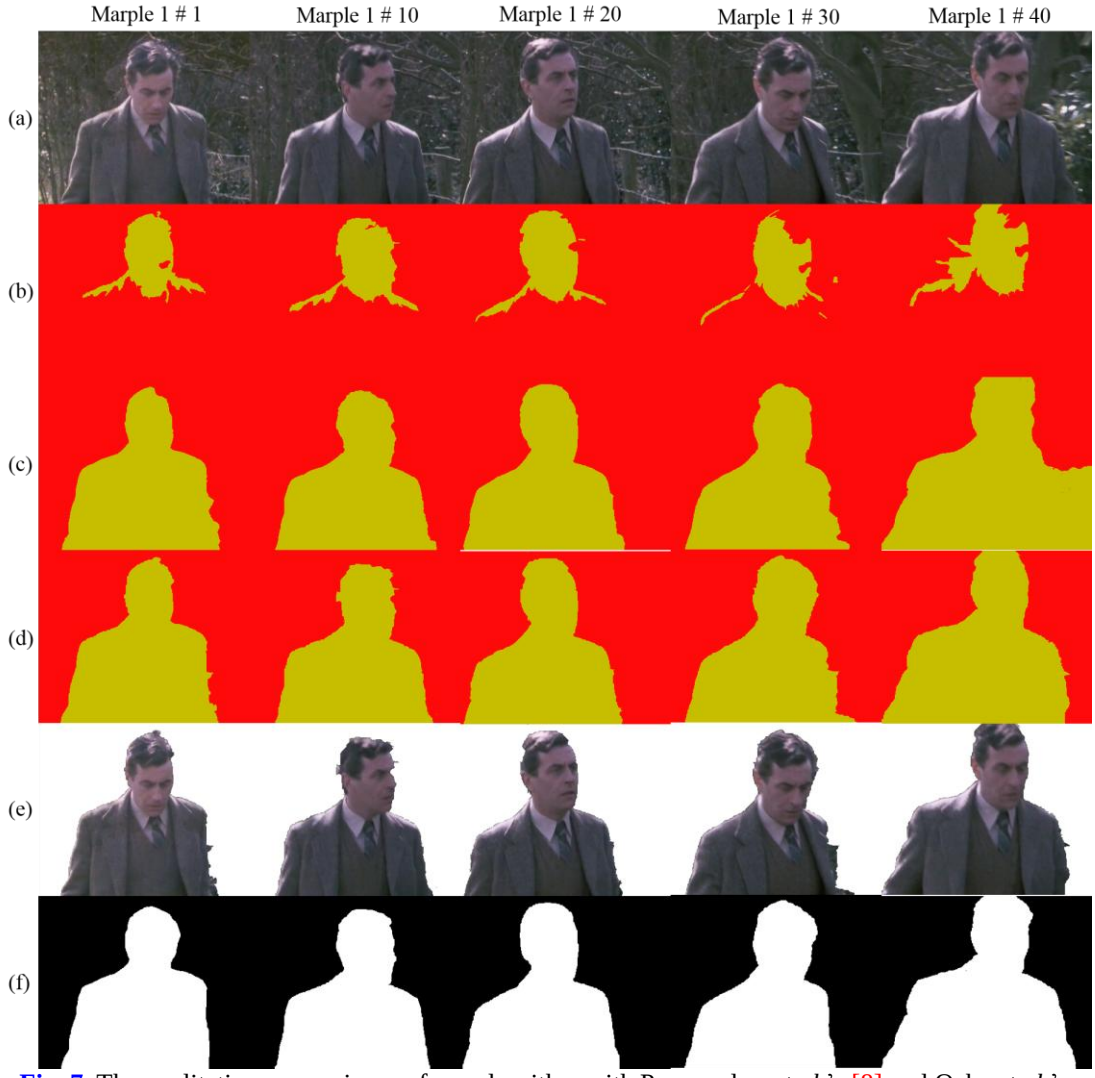


Fig. 7. The qualitative comparisons of our algorithm with Papazoglou *et al.*'s [8] and Ochs *et al.*'s approach [14] on Marple 1 sequence.

3.1 Experiments using the Freiburg-Berkeley Motion Segmentation Dataset [14]

In these experiments, we test our algorithm by using the sequences from the Freiburg-Berkeley Motion Segmentation Dataset [14]. As mentioned above, our aim is to detect the single moving object from videos or image sequences. So we conduct experiments on sequences containing a single moving object in the field of view, which include 28 sequences with 332 annotated ground truth images. These sequences cover typical challenges such as camera motion, occlusion, changing scale and changing illumination, *etc*.

The ground truth of the moving object is annotated every 10 frames in the Marple sequences and every 20 frames in the others. For each sequence, we generate the labelled point trajectories in all the frames by using semi-dense point trajectories method [20], and then detect the moving object in each frame by using our proposed method.

During the experiments, the desired number of superpixels is set to be 200 for the Marple sequences, and the number doubles for the others because the size of these sequences are

approximately 2 times compared with the Marple sequences. And the value of *k* in superpixels clustering is fixed to be 18 empirically. These parameters can be tuned in other cases according to the size of the input images: the larger the images, the greater the desired number of superpixels and *k* can be set.

Fig. 8. The qualitative comparisons of our algorithm with Papazoglou *et al.*'s [8] and Ochs *et al.*'s approach [14] on horses 01 sequence.

3.1.1 Qualitative results

The qualitative comparisons on Marple1 sequence and horses01 sequence are shown in **Fig. 7** and **Fig. 8** respectively: (a) indicates the frames in which the moving object is to be detected; (b) indicates the detection results by using Papazoglou *et al.*'s approach [8], and (c) indicates the detection results by using Ochs *et al.*'s approach [14], where the detected moving objects are shown in yellow; (d) and (e) show the detection results by using our algorithm, where the detected moving objects are also shown in yellow in (d) to compare with results in (b)(c), and (e) shows pixels belonging to the foreground; and (f) indicates the ground truth. Some additional results from other sequences are shown in **Fig. 9** and **Fig. 10**.

We can see that our proposed method can detect the moving objects from the sequences, overcoming challenges such as camera motion, cluttered background, occlusion, changing scale and changing illumination, *etc*. The large area segmentation errors in Marple 1 # 40 and Marple 4 # 30 by Ochs *et al.*'s method [14] caused by the partial mislabelled point trajectories are avoided due to the coarse-to-fine strategy we adopted.

To better explain the advantage of the proposed coarse-to-fine strategy, we take Marple 4 # 30 as an example for illustration. The mislabelled points occur in the neighbouring region of

the hat and the background in the top left of the image. If only coarse segmentation is performed, large area segmentation errors similar with Ochs *et al*.'s method [14] would occur. The reason lies in that the region is relatively textureless and even few mislabelled points would affect the belonging of the whole region. However, due to the following fine segmentation, the potential misclassification would be corrected by firstly marking the belonging of the region as probable to be changed in the Quadmap, and then performing pixel level refinement taking the neighbouring regions whose belonging are definitely assured into consideration.

The poor performance of Papazoglou *et al*.'s method [8] in the Marple sequences and the horses 01 sequence are mainly caused by the heavy dependence on optical flow to find an initial foreground boundary. When the calculation of optical flow is influenced, say by the low-texture sweater regions in the Marple 1 sequence, it cannot achieve good results.

Fig. 9. The qualitative comparisons of our algorithm with Papazoglou *et al*.'s [8] and Ochs *et al*.'s approach [14] on other sequences.

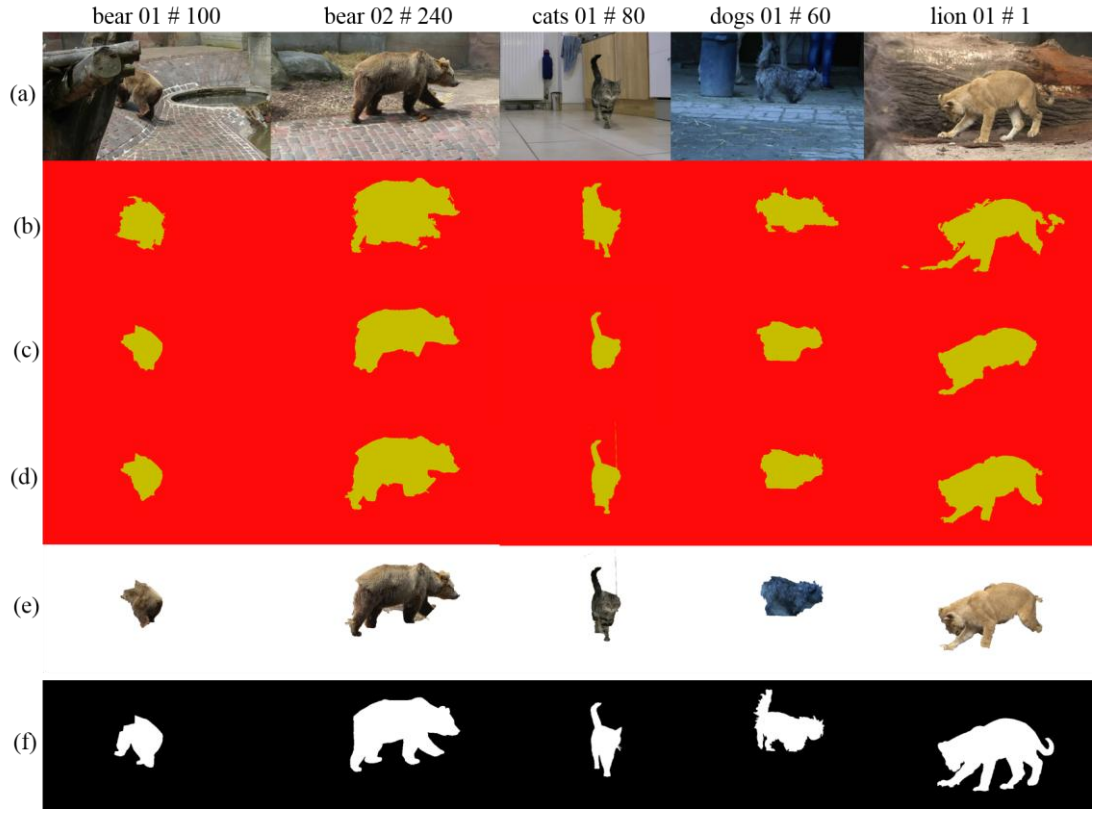


Fig. 10. The qualitative comparisons of our algorithm with Papazoglou *et al.*'s [8] and Ochs *et al.*'s approach [14] on other animal sequence.

3.1.2 Quantitative results

We evaluate our proposed method quantitatively by using three criteria: precision, recall, and *F*-measure [34]. Precision is defined in Equation (11) and used to measure the percentage of the pixels correctly labelled as foreground among all the pixels labelled as foreground

$$precision = \frac{TP}{TP + FP} \tag{11}$$

where *TP* is the number of pixels labelled as foreground correctly, and *FP* is the number of pixels wrongly labelled as foreground.

Recall is defined in Equation (12) and used to evaluate the percentage of the pixels correctly labelled as foreground among all the pixels belonging to the foreground according to the ground truth.

$$recall = \frac{TP}{TP + FN} \tag{12}$$

where false negative (*FN*) indicates the number of pixels belonging to the foreground but wrongly labelled as background.

F-measure takes above mentioned precision and recall into consideration, and is used for measuring accuracy.

$$F - measure = \frac{2 \cdot precision \cdot recall}{precision + recall} \tag{13}$$

Table 1 shows the quantitative evaluation of different methods, including the average precision, average recall, average *F*-measure, the percentage of frames with the *F*-measure > 0.6, and the percentage of frames with the *F*-measure > 0.75. The larger the *F*-measure is, the higher the performance will be. And the *F*-measures achieved by using different methods on these sequences are shown in **Fig. 11**. Where red indicates the results of our method, and green and yellow indicate the result of Ochs *et al*. [14] and Papazoglou *et al.* [8] respectively. As can be seen, our method can achieve favorable results compared with state-of-the-art methods.

However, both our method and Ochs *et al.*'s method [14] use the labelled trajectory points as the motion cue. When the trajectory points labelling failed, for example in rabbits04 sequence the rabbit is too small to contain enough tracked feature points, these methods do not performs well. Thus point trajectories based methods are more suitable to be used in detecting objects that occupy a certain part of the image, but not for detecting tiny objects.

Table 1. Quantitative comparison of different methods

	precision	**recall**	***F*-measure**	***F*-measure>0.6**	***F*-measure>0.75**
Our method	0.745	0.698	0.693	76.2%	69.9%
Papazoglou *et al.*	0.602	0.742	0.614	57.2%	43.4%
Ochs *et al.*	0.767	0.684	0.689	76.8%	66.3%

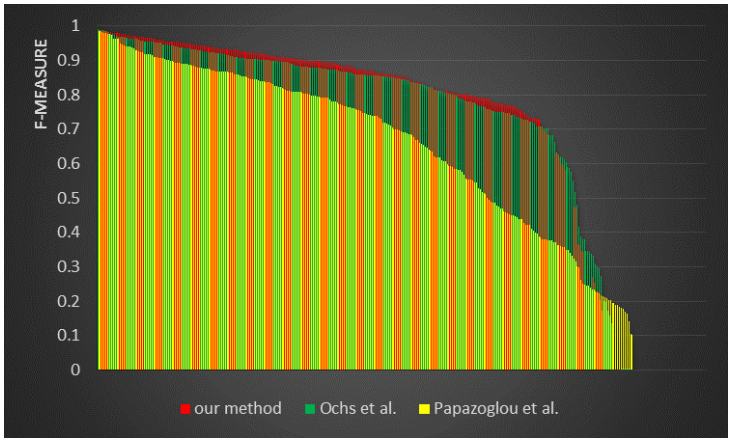


Fig. 11. The *F*-measure of different methods

3.1.3 Running time

In order to test the efficiency, a notebook PC with 2.3GHz CPU and 12GB RAM is used as the experimental platform. Both our method and Ochs *et al.*'s method [14] need the labelled point trajectories as the input.

As to the Marple sequences, some of them are 350×288 in resolution and the others are 450×350 in resolution. It takes 4.2s on average for generating semi-dense point trajectories across 10 successive frames by using the semi-dense point trajectories method [20]. The coarse-to-fine segmentation of our method takes 0.85s per frame on average, and the detailed time cost for superpixels segmentation, feature extracting and superpixels clustering, fine segmentation by using Grabcut, and other procedures are 40ms, 230ms, 460ms, and 120ms, respectively. However, Ochs *et al.*'s method costs 59s on average, and Papazoglou *et al.*'s method [8] consumes 13.4s per frame on average.

And the resolutions of the other sequences include 960×540, 640×400, 640×480, etc. It takes 220s on average for generating semi-dense point trajectories across 20 successive frames

by using the semi-dense point trajectories method [20]. The coarse-to-fine segmentation of our method takes 1.5s per frame on average, and the detailed time cost for superpixels segmentation, feature extracting and superpixels clustering, fine segmentation by using Grabcut, and other procedures are 100ms, 400ms, 850ms, and 150ms, respectively. However, Ochs *et al.*'s method [14] costs 150s on average, and Papazoglou *et al.*'s method [8] consumes 100s per frame on average.

As can been seen from above, the time consumption reduce significantly by using our method due to the coarse-to-fine strategy adopted.

3.2 Experiments using real world video

In these experiments, we test our algorithm by using image sequences intercepted from a video downloaded from the Internet. In the video, the Atlantis Space Shuttle is approaching the International Space Station for rendezvous & docking. As shown in **Fig. 12**, the Atlantis is the moving object to be detected, and the clouds constitute the dynamic background.

Papazoglou *et al.*'s method [8] and semi-dense point trajectories method [20] failed due to the insufficient displacements between adjacent frames to calculate optical flow, which is the prerequisite for both methods. During the experiments, we only perform qualitative comparison between our proposed algorithm and Ochs *et al.*'s [14] due to lack of ground truth. For each sequence, the labelled point trajectories are generated by tracking extracted feature points across 20 frames using the KLT + SSC method we proposed in [22].

The detected Atlantis in each sequence is shown in **Fig. 12**: (a) show the 20th frames of each sequence, in which the flying Atlantis is to be detected; (b) show the detection results by using Ochs *et al.*'s approach [14], where the detected Atlantis is shown in yellow; (c) and (d) show the detection results by using our algorithm, where the detected Atlantis is also shown in yellow in (c) to compare with results in (b), and (d) show pixels belonging to the Atlantis. By comparing results in (b) and (c) we can see that, both Ochs *et al.*'s method [14] and our algorithm are able to detect the maneuvering Atlantis from the dynamic background. However, the empennages in the 2nd and 3rd columns of (b) are wiped out. This is caused by the variational method they used, which tends to smooth out fine details to minimize the global energy function. Our approach outperforms and preserves fine details.

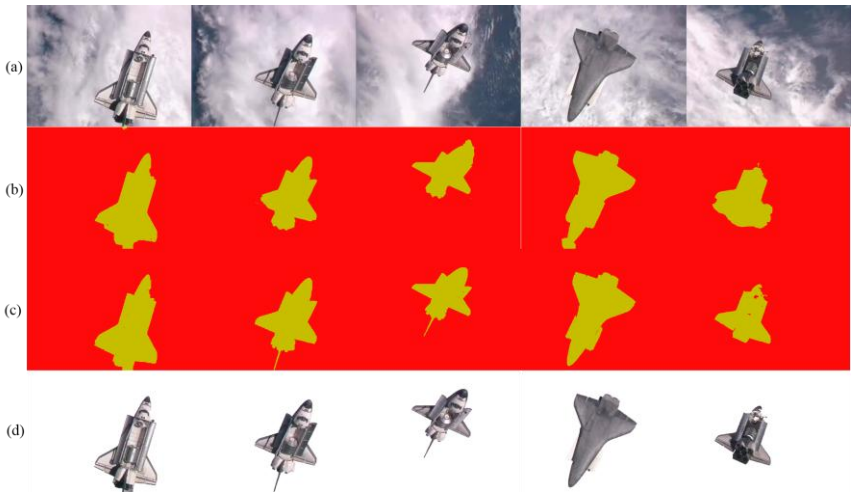


Fig. 12. The qualitative comparison of our algorithm with Ochs *et al.*'s approach [14] on Atlantis sequences.

4. Conclusion and Future Work

A new coarse-to-fine framework for unsupervised segmentation of single moving object from videos or image sequences is proposed in this work, which can be widely used in navigation, surveillance, recognition, 3D reconstruction and many more. For example, after detecting the moving object in the video, object recognition can be performed to infer the category it belongs to.

Our algorithm is composed of a superpixel level coarse segmentation to facilitate subsequent processing as well as to provide a good initialization, and a pixel level fine segmentation to improve the segmentation accuracy. A distance measurement is devised to measure the similarities between generated superpixels, which is of great significance for clustering in the coarse segmentation stage. Moreover, a Quadmap is introduced to facilitate the refinement in the fine segmentation stage. According to the experiments, our algorithm is effective and efficient, and favorable results can be achieved compared with state-of-the-art methods.

And in the future we intend to annotate the ground truth of more images manually to increase the field and the number of testing images, and then perform more experiments to evaluate our method thoroughly. We also plan to extend our work from single moving object detection to multiple moving object detection.

References

[1] C. Stauffer, W. E. L. Grimson, "Adaptive background mixture models for real-time tracking," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2246-2252, June 23-25, 1999. Article (CrossRef Link)

[2] K. Kim, T. H. Chalidabhongse, D. Harwood, and L. Davis, "Background modeling and subtraction by codebook construction," in *Proc. of the IEEE International Conference on Image Processing*, pp. 3061-3064, October 24-27, 2004. Article (CrossRef Link)

[3] M. Heikkilä, M. Pietikäinen, "A texture-based method for modeling the background and detecting moving objects," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 28, No. 4, pp. 657-662, May, 2006. Article (CrossRef Link)

[4] S. Z. Li, M. Pietikainen, V. Kellokumpu, G. Zhao, and S. Liao, "Modeling pixel process with scale invariant local patterns for background subtraction in complex scenes," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1301-1306, June 13-18, 2010. Article (CrossRef Link)

[5] R. K. Sabhara, C. P. Lee and K. M. Lim, "Comparative Study of Hu Moments and Zernike Moments in Object Recognition," *Smart Computing Review*, Vol. 3, No. 3, pp. 166-173, June 2013. Article (CrossRef Link)

[6] S. Brutzer, B. Höferlin, G. Heidemann, "Evaluation of background subtraction techniques for video surveillance," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1937-1944, June 20-25, 2011. Article (CrossRef Link)

[7] O. Barnich, and M. V. Droogenbroeck, "Vibe: A universal background subtraction algorithm for video sequences," *IEEE Transactions on Image Processing*, Vol. 20, No. 6, pp. 1709-1724, June, 2011. Article (CrossRef Link)

[8] A. Papazoglou, V. Ferrari, "Fast object segmentation in unconstrained video," in *Proc. of the IEEE International Conference on Computer Vision*, pp. 1777-1784, December 1-8, 2013. Article (CrossRef Link)

[9] C. Tomasi, and T. Kanade, "Shape and motion from image streams under orthography: A factorization method," *International Journal of Computer Vision*, Vol. 9, No. 2, pp. 137-154, January, 1992. Article (CrossRef Link)

[10] Y. Sheikh, O. Javed, and T. Kanade, "Background subtraction for freely moving cameras," in *Proc. of the IEEE International Conference on Computer Vision*, pp. 1219-1225, September 27 - October 4, 2009. Article (CrossRef Link)
[11] A. Petit, *Robust visual detection and tracking of complex objects: Applications to space autonomous rendezvous and proximity operations*, Université de Rennes 1, Rennes, France, 2013. Article (CrossRef Link)
[12] L. Zappella; X. Lladó; E. Provenzi and J. Salvi, "Enhanced local subspace affinity for feature-based motion segmentation," *Pattern Recognition*. Vol. 44, No. 2, pp. 454-470, February, 2011. Article (CrossRef Link)
[13] E. Elhamifar, R. Vidal, "Sparse subspace clustering: Algorithm, theory, and applications," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 35, No. 11, pp. 2765-2781, March, 2013. Article (CrossRef Link)
[14] P. Ochs, J. Malik, and T.Brox, "Segmentation of moving objects by long term video analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 36, No. 6, pp. 1187-1200, 2014. Article (CrossRef Link)
[15] F. Galasso, M. Keuper, T. Brox, and B. Schiele, "Spectral graph reduction for efficient image and streaming video segmentation," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 49-56, June 23-28, 2014. Article (CrossRef Link)
[16] F. Galasso, R. Cipolla, and B. Schiele, "Video segmentation with superpixels," in *Proc. of the Asian Conference on Computer Vision*, pp. 760-774, November 5-9, 2012. Article (CrossRef Link)
[17] L. Ellis, and V. Zografos, "Online learning for fast segmentation of moving objects," in *Proc. of the Asian Conference on Computer Vision*, pp. 52-65, November 5-9, 2012. Article (CrossRef Link)
[18] X. Ren, and J. Malik, "Learning a classification model for segmentation," in *Proc. of the IEEE International Conference on Computer Vision*, pp. 10-17, October 14-17, 2003. Article (CrossRef Link)
[19] D. Stutz, *Superpixel segmentation using depth information*, RWTH Aachen University, Aachen, Germany, 2014. Article (CrossRef Link)
[20] T. Brox, and J. Malik, "Object segmentation by long term analysis of point trajectories," in *Proc. of the European Conference on Computer Vision*, pp. 282-295, September 5-11, 2010. Article (CrossRef Link)
[21] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 33, No. 5, pp. 898-916, 2011. Article (CrossRef Link)
[22] X. Z. Zhu, X. Song, X. Q. Chen, and H. M. Lu, "Flying spacecraft detection with the earth as the background based on superpixels clustering," in *Proc. of the IEEE International Conference on Information and Automation*, pp. 518-523, August 8-10, 2015. Article (CrossRef Link)
[23] J. Shi, and C. Tomasi, "Good features to track," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 593-600, June 21-23, 1994. Article (CrossRef Link)
[24] A. Y. Ng, M. I. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," in *Proc. of Advances in Neural Information Processing Systems*, pp. 849-856, December 3-8, 2001. Article (CrossRef Link)
[25] U.v. Luxburg, "A tutorial on spectral clustering," *Statistics and Computing*, Vol. 17, No. 4, pp. 395-416, 2007. Article (CrossRef Link)
[26] J. Shi, and J. Malik, "Normalized cuts and image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 22, No. 8, pp. 888-905, 2000. Article (CrossRef Link)
[27] P. Neubert, and P. Protzel, "Compact watershed and preemptive SLIC: On improving trade-offs of superpixel segmentation algorithms," in *Proc. of the IEEE International Conference on Pattern Recognition*, pp. 996-1001, August 24-28, 2014. Article (CrossRef Link)
[28] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, "SLIC superpixels compared to state-of-the-art superpixel methods," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 34, No. 11, pp. 2274-2282, May, 2012. Article (CrossRef Link)

[29] H. Lu, L. Jiang, and A. Zell, "Long range traversable region detection based on superpixels clustering for mobile robots," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, September 28 - October 2, 2015. Article (CrossRef Link)
[30] J. Chen, S. Shan, C. He, G. Zhao, M. Pietikäinen, X. Chen, and W. Gao, "WLD: A robust local image descriptor," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 32, No. 9, pp. 1705-1720, September, 2010. Article (CrossRef Link)
[31] S. H. Cha, "Comprehensive survey on distance/similarity measures between probability density functions," *International Journal of Mathematical Models & Methods in Appliedences*, Vol. 1, No. 4, pp. 300-307, January, 2007. Article (CrossRef Link)
[32] J. Wang, and M. F. Cohen, "Image and video matting: A survey," *Foundations and Trends in Computer Graphics and Vision*, Vol. 3, No. 2, pp. 97-175, 2007. Article (CrossRef Link)
[33] C. Rother, V. Kolmogorov, and A. Blake, "Grabcut: Interactive foreground extraction using iterated graph cuts," *ACM Transactions on Graphics*, Vol. 23, No. 3, pp. 309-314, 2004. Article (CrossRef Link)
[34] D. L. Olson, and D. Delen, *Advanced data mining techniques*, 1st Edition, Springer, Berlin, 2008. Article (CrossRef Link)

Xiaozhou Zhu received the M.S degree in control science and engineering from National University of Defense Technology, China in 2012. He is currently a Ph.D student in the College of Aerospace Science and Engineering at National University of Defense Technology. His research interests include visual tracking, and robot vision.

Xin Song is currently an associate researcher in the College of Aerospace Science and Engineering at National University of Defense Technology, China. His research interests include robot vision, and star tracker.

Xiaoqian Chen is currently a professor in the College of Aerospace Science and Engineering at National University of Defense Technology, China. His recent research interests include aeronautical engineering, optimization, robot vision, *etc*.

Huimin Lu received the Ph.D. degree in control science and engineering from the National University of Defense Technology, China in 2010. He is currently an associate professor in the College of Mechatronics and Automation at National University of Defense Technology. His recent research interests include robot vision, omnidirectional vision, and robot soccer.