Feature-Based Efficient Moving Object Detection for Low-Altitude Aerial Platforms

K. Berker Logoglu¹, Hazal Lezki¹, M. Kerim Yucel¹,², Ahu Ozturk¹, Alper Kucukkomurler¹, Batuhan Karagoz¹, Aykut Erdem², and Erkut Erdem²

¹STM Defense Technologies and Trade Inc, Ankara, Turkey.
²Computer Vision Lab, Department of Computer Engineering, Hacettepe University, Ankara, Turkey.
¹{blogoglu,hlezki,myucel,iaozturk,alper.kucukkomurler,batuhan.karagoz}@stm.com.tr
²{aykut,erkut}@cs.hacettepe.edu.tr

Abstract

Moving Object Detection is one of the integral tasks for aerial reconnaissance and surveillance applications. Despite the problem’s rising potential due to increasing availability of unmanned aerial vehicles, moving object detection suffers from a lack of widely-accepted, correctly labelled dataset that would facilitate a robust evaluation of the techniques published by the community. Towards this end, we compile a new dataset by manually annotating several sequences from VIVID and UAV123 datasets for moving object detection. We also propose a feature-based, efficient pipeline that is optimized for near real-time performance on GPU-based embedded SoMs (system on module). We evaluate our pipeline on this extended dataset for low altitude moving object detection. Ground-truth annotations are made publicly available to the community to foster further research in moving object detection field.

1. Introduction

In line with the recent trend of Unmanned Aerial Vehicle (UAV) usage in civilian and military sectors, miniature UAVs became more and more affordable and thus accessible for everyone. In addition to its vast military deployment in various armed forces in the world, civilian applications of UAVs also emerged swiftly; industrial inspection, agriculture, mapping, transport, cinematography and numerous indoor applications are examples for UAV usage in various fields.

The primary issue of UAV-based computer vision (CV) applications is the platform itself; it is not stable, it tends to have sudden movements, it is exposed to weather conditions, it generates non-homogeneous data (scale, angle, rotation, depth, etc.) and most importantly, it is inherently limited in computational resources. All these difficulties and constraints further the complexity of conventional vision problems, such as object tracking, object detection, object classification and prominently, moving object detection.

Moving object detection is a well-studied problem in controlled environments where ego-motion is not present [31]. Introduction of unconstrained ego-motion, however, transforms moving object detection into a much harder problem. Ego-motion estimation and compensation, which exploits image alignment techniques, is performed to address moving object detection in such scenarios [1]. In Wide Area Motion Imagery (WAMI), these techniques could suffice due to comparably low effect of motion parallax [25]. However, for low-altitude UAV scenarios, motion parallax has a far more detrimental effect [32].

Our contributions are primarily as follows: First, we compile a dataset comprised of sequences from VIVID [6] and UAV123 [18] datasets. We carefully pick sequences with various scenarios; motion parallax, altitude variation, viewpoint variation, presence of multiple objects, varying object sizes and speeds are considered. We hand-annotate the sequences specifically for moving object detection. Second, we present our analysis on this dataset using an efficient, feature-based pipeline which we optimize for near real-time performance on embedded GPU-based SoMs. The annotations are also made publicly available on https://github.com/LAMODDATASET/LAMOD.

The remainder of this paper is organized as follows: related work is reviewed in Section 2. The proposed moving
object detection framework is explained in detail in Section 3. Experimental results as well as details on our extended dataset are given in Section 4. Our conclusions and future works are outlined in Section 5.

2. Related Work

Moving object detection task has been an active area of research within the CV community for a couple of decades. Earlier studies suggest simple background subtraction to segment foreground (object) and background in videos [8]. Among other techniques, spatial differencing [19] has been a prominent one. Temporal differencing, however, does not work properly in slow motion, when moving objects are small compared to the overall visual and if the objects have smooth texture. Statistical methods, where background is adaptively learned, successfully addresses these issues. Such techniques include the use of mixture of Gaussians, Eigen backgrounds [13], stereo [9] and motion-layer [4] approaches. For a general overview of the literature, readers are referred to [24] and [13].

The majority of above-mentioned algorithms, however, are not devised to address the presence of ego-motion, therefore fail to successfully operate in such cases. Towards this end, image alignment based algorithms have emerged. The principle idea is to align the images before performing frame differencing; affine or perspective transformation matrix is used to warp the images. Assuming accurate alignment, frame differencing ideally provides the moving regions in the sequence with the help of morphological operations and connected component analysis [12]. Such techniques can be unified as “feature-based” techniques, where various feature keypoints/descriptors can be used to match images for warping. Another way of addressing the pertinent problem is “motion-based” techniques, where motion-layers [4] and optical flow are used to detect moving regions. Stereo vision [26], flow vector segmentation [20] based on orientation have found use in the field as well. Such algorithms can be considered as good fit for WAMI applications where high altitude bird-eye view images are concerned. When planar surface assumption does not hold in the scene (i.e. low altitude UAV footage), false positives due to motion parallax become more severe.

Parallax handling is inherently a costly operation; it requires camera calibration and compliance to various constraints. In [14], epipolar constraint along with a “plane-parallax” based structure consistency is implemented to distinguish moving regions from parallax-induced false positives. Another work reported in [21] compare epipolar line directions with optical flow orientation to accurately identify moving objects. Work reported in [15] handles parallax via enforcing epipolar and flow-bound constraints with the help of camera pose estimation via Parallel Tracking and Mapping (PTAM). The authors of [7] handle parallax via a parametrized epipolar constraint without the need of motion registration. In [29], a “three-view distance constraint” along with epipolar constraint is used to facilitate moving object detection.

Handling parallax does not necessarily require compliance to epipolar constraints, as shown in various studies. [20] and [5] use optical flow and artificial flow to infer moving regions while handling parallax implicitly. Another study presented in [23] follows an image warping based scheme where moving regions are clustered into moving objects and parallax is handled via object size priors and Kalman-filter based persistency check. There are studies where other priors, such as classifiers for moving object to be detected, are used but this is out of our work’s scope as we do not concentrate on specific classes of objects [22] [28].

Due to operational requirements, moving object detection generally needs to work in real-time. Performing the resource-heavy calculations on a dedicated server is inherently limited by the quality of the data link between the server and the platform. The alternative is to perform the calculations on the UAV itself using a dedicated embedded resource, which is non-trivial from an engineering perspective. There have been several attempts on speeding up moving object detection such as GPU implementations reported in [30] and [3]. For mobility, FPGA has also been used extensively [16], [27]. To the best of our knowledge, however, there has not been an attempt on moving object detection with parallax handling using on-platform embedded resources with near real-time performance.

3. Our Approach

The pipeline of our approach is shown in Figure 1. We propose a feature-based, efficient pipeline mostly implemented on GPU. We have divided our approach to four steps for ease of understanding: pre-processing, ego-motion compensation, moving object detection and parallax-filtering.

3.1. Pre-processing

In the pre-processing step, the input frames are downsampled for faster processing. The downsampled frames are pushed to a frame-buffer whose size varies depending on number of frame difference to be used. The number of frame difference depends on height and the minimum desired motion to be detected; as the frame difference increases, smaller motion can be detected. At each acquired frame, keypoints and corresponding descriptors (for our case; SURF [2]) are calculated. Lastly, the descriptors are matched to previous frame to calculate homography and fundamental matrix. An important step is the filtering of the keypoints and the corresponding descriptors based on the (candidate) moving objects found in the previous frame. Without filtering, these unwanted keypoints
have a detrimental effect on homography calculation and results in inferior perspective warping. One such example is shown in Figure 2, where the bottom left image shows the result of inferior perspective warping and the bottom right image shows accurate warping.

Figure 2. Effect of filtering keypoints on moving objects (from previous detection) before calculating the current homography matrix. Top to bottom: extracted keypoints, thresholded two frame difference and final results are given. Left and right column images show the results without and with the proposed keypoint filtering loop, respectively.

3.2. Ego-motion compensation

Based on the descriptor matching, firstly homography matrix is calculated. RANSAC is used to reject outliers in the process. Secondly, using the calculated homography matrix, previous frame is warped onto the current one using perspective transformation.

3.3. Moving object detection

Since the camera motion is compensated in the previous step, basically two-frame differencing gives the moving pixels. In our approach, we are using three-frame differencing method which is an improved version of two-frame differencing in the sense that it removes noise more effectively. We basically apply logical AND operation for two adjacent two-frame difference results. After obtaining the moving pixels, various morphological operations are applied to filter noise and unite points belonging to same object. Finally, connected component analysis is applied to extract individual object bounding boxes.

3.4. Parallax removal with geometric constraints

The final step is the removal of objects that are falsely detected as moving due to parallax effect. At this step, instead of pixel wise decisions, we make object wise decisions for efficiency. We use the keypoints already extracted on each frame thus we only make decisions for the candidate objects on which keypoints exists. We basically apply two geometric constraints that is proposed in the literature.
[14, 15]; epipolar constraint and flow-vector bound (FVB) constraint.

3.4.1 Epipolar Constraint

Let \( I_{t1} \) and \( I_{t2} \) be two images of a scene acquired by the same camera \( c \) at time \( t1 \) and \( t2 \) respectively from different positions in space. Additionally, let \( P \) denote a point in the scene, \( i \) denote different scene points, \( p_{t1} \) be its projection in \( I_{t1} \), \( P' \) denote the new location of \( P \) at \( t2 \) and \( p_{t2} \) be its projection on the image \( I_{t2} \). There exists a unique fundamental matrix \( F_{12}^{12} \) relating the two images where \( I_{t1} \) is the reference, which satisfies

\[
p_{t2}^i \, F_{12}^{12} \, p_{t1}^i = 0,
\]

for all corresponding points \( p_{t1}^i \) and \( p_{t2}^j \). If \( P \) is a static point, it satisfies \( e_{l1}^i = F_{12}^{12} \, p_{t1}^i \) and \( e_{l2}^j = F_{12}^{12} \, p_{t2}^j \) where \( e_{l1}^i \) and \( e_{l2}^j \) are epipolar lines corresponding to \( p_{t1}^i \) and \( p_{t2}^j \) respectively. Thus, we can say that if \( P' \) is a static point, it should lie on the corresponding epipolar lines as shown in Figure 3. On the contrary, if a point is moving, it does not satisfy the aforementioned epipolar constraint as shown in Figure 4 with an exceptional case; when the point moves along the epipolar lines, i.e. when the camera and point/object move along the same direction as shown in Figure 5.

To classify a feature point as static or dynamic we use the perpendicular distance of the feature point to the corresponding epipolar line as suggested in [15]. Additional to previous works, before applying the epipolar constraint, to calculate fundamental matrix, we use the already extracted keypoints, descriptors and the matches, but with a key difference; instead of using the full set of keypoints or the ones filtered by the candidate moving objects (static + moving) as we did in Section 3.1, this time we filter out the keypoints/descriptors based on the previous final moving object detection result where we expect to detect only the real moving objects. Since the moving objects do not belong to the actual 3D scene, we expect them to hurt the fundamental matrix, thus the epipolar line calculation.

In Figure 6, two different scenes with parallax are shown. It can clearly be seen that the keypoints, which are on static objects falsely detected as moving, move along the corresponding epipolar lines and thus can not be filtered with epipolar constraints. On the other hand, keypoints on real (non-degenerately) moving objects (such as the car in the first row) are not filtered.

3.4.2 Flow-Vector Bound Constraint

To handle the degenerate motion cases when the camera and point/object move along the same direction, flow-vector bound (FVB) constraint is used as suggested in [17, 15]. FVB constraint restricts the motion of static points along the epipolar lines.

For any static point, when a camera follows an ob-
subject/object, assuming the camera translates by $t$, the movement of the point is given by [17]

$$p_{t2} = p_{t1} + \frac{Kt}{z},$$

(2)

where $z$ is the depth of the scene and $K$ is the intrinsic matrix of the camera. Since we are dealing with already captured videos in this study and we do not have the camera parameters, we can use the following

$$Kt = e_{t2}^1,$$

(3)

where $e_{t2}^1$ is the epipole location on $I_{t2}$. Because of the lack of camera matrix and and the camera translation amount, we only have the ability to calculate relative depth between matched points ($z^i_t$) as suggested in [10, 14]

$$z^i_t = \frac{(H_{t1}^2 p_{t1}^i \times p_{t2}^i)^T (p_{t2}^i \times e_{t2}^1)}{|p_{t2}^i \times e_{t2}^1|^2}.$$  

(4)

Using Equations 2, 3 and 4 we can compute the displacement bounds along the epipolar line, $d_{min}$ and $d_{max}$. If a point’s displacement is not between $d_{min}$ and $d_{max}$, it is more likely to be a point on a moving object. Since we also have the point pair matches (in our case, keypoint matches), we also check the deviation between the matched location and result of Equation 2. An example degenerate motion and the result of applying FVB constraint in shown in Figure 7.

4. Experiments

4.1. Dataset

Even though a handful of moving object detection studies have been published, the field still suffers from the lack of a content-rich and accurately labelled dataset. A general trend in moving object detection literature, especially for low-altitude UAVs, is the use of custom datasets compiled by respective authors. Best to our knowledge, however, neither these datasets nor their ground-truth labels are shared with the community in general.

UAV footage datasets have emerged in recent years due to the increased availability of the platforms, however, they are generally labelled for object tracking purposes [18]. This means only a select number of objects present are labelled. Moreover, tracked objects may or may not be moving, therefore such labels are not reliable for moving object detection evaluation.

LAMOD Dataset. In order to address said challenges, we compile our own ground-truths for a combined
Figure 7. Example of degenerate motion and filtering of static points by FVB constraint. From top-left to bottom-right, the detected motion areas, the corresponding keypoint-displacement and epipolar lines on those areas, result without applying FVB constraint and final result with FVB constraint applied are shown in each image.

Table 1. Execution time without parallax filtering.

<table>
<thead>
<tr>
<th>Video Resolution</th>
<th>TX1</th>
<th>TX2</th>
</tr>
</thead>
<tbody>
<tr>
<td>640x360</td>
<td>110 ms</td>
<td>79 ms</td>
</tr>
<tr>
<td>640x480</td>
<td>115 ms</td>
<td>85 ms</td>
</tr>
<tr>
<td>1280x720</td>
<td>350 ms</td>
<td>250 ms</td>
</tr>
</tbody>
</table>

Table 2. Execution time with parallax filtering.

<table>
<thead>
<tr>
<th>Video Resolution</th>
<th>TX1</th>
<th>TX2</th>
</tr>
</thead>
<tbody>
<tr>
<td>640x360</td>
<td>163 ms</td>
<td>135 ms</td>
</tr>
<tr>
<td>640x480</td>
<td>175 ms</td>
<td>140 ms</td>
</tr>
<tr>
<td>1280x720</td>
<td>450 ms</td>
<td>350 ms</td>
</tr>
</tbody>
</table>

4.2. Results

Following the annotation of the extended dataset, we evaluate our pipeline with and without parallax handling. We tune some hyper-parameters with respect to each video to see the extent of this pipeline with optimal tuning. Parameters that are tuned are the number of frames to be differenced and morphology operations. These parameters could have been adaptively changed if external sensory (i.e. IMU) data were present, but for sequences in LAMOD dataset, they are not available. It must be noted that the pipeline does not make use of split/merge handling or shadow suppression, therefore readers are reminded that these values are produced to be used as benchmarks only.

We use NVidia Jetson TX1/TX2 modules in our experiments. Execution times are shown in Table 1 and 2.

For evaluation metrics, we use precision/recall and F-Score, where we take a minimum of 50% overlap to be a correct detection.
4.2.1 Without parallax handling

Precision/recall and F-Score values without parallax handling are shown in Table 3. A figure representing the steps of the pipeline is shown in Figure 9.

It is apparent that for egtest01 results are the best among others. This is expected as this video does not have parallax, viewpoint/altitude variation or any sudden movements. The movement speed is minimal as well. For egtest02, a reduction in recall is obvious as severe occlusion exists, as well as sudden movements. egtest04 shows the detrimental effect of viewpoint variation and out-of-focus effect, which degrades the gradients and thus the feature extraction quality. Egtest05 has the highest amount of occlusion, although the stability of number of object numbers and altitude helps retain an adequate result. Redteam sequence has the lowest resolution, and even though a single object is present, the effect of minor parallax degrades the result quality.

Rest of the videos, car2-3-4-6, are harder as altitude variation, parallax and sudden movements are more frequent. With careful tuning, car2 still manages to keep a good quality, however the others have severely degraded results, especially in recall. Car4 has parallax, occlusion and various object sizes/movement speeds. Car6 and car10 sequences, on the other hand, produce the worst results as severe parallax (degrades precision) coupled with occlusion and varying object sizes (small objects degrade recall) have apparent detrimental effects.

4.2.2 With parallax handling

As mentioned in the previous section, one of the most severe parallax cases are observed in car6, car10 and car1 sequences. Therefore, we evaluate only these videos with our proposed parallax handling scheme. A representative figure of the flow is shown in Figure 10. Results are shown in Table 4.

We evaluate our technique using the segments of the videos where parallax is quite severe. As can be seen from the results, we observe solid increase in F-Scores in each video. Parallax primarily degrades precision due to false positives, and accordingly, parallax handling increases the precision the most. As parallax handling eliminates candidate moving objects, it is apparent that some true positives are eliminated as well, thus the slightly lowered recall.

5. Conclusions and Future Work

Inspired by the lack of a widely-accepted and carefully annotated dataset for moving object detection from low altitude UAV imagery, we present our hand-annotated dataset.
Table 4. Precision-recall and F-Score values with parallax handling. PH stands for parallax handling.

<table>
<thead>
<tr>
<th>Sequence</th>
<th>car1 (first 1020 frames)</th>
<th>car6 (first 700 frames)</th>
<th>car10 (first 750 frames)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>without PH</td>
<td>with PH</td>
<td>without PH</td>
</tr>
<tr>
<td><strong>Precision</strong></td>
<td>0.55</td>
<td>0.70</td>
<td>0.12</td>
</tr>
<tr>
<td><strong>Recall</strong></td>
<td>0.51</td>
<td>0.50</td>
<td>0.71</td>
</tr>
<tr>
<td><strong>F-Score</strong></td>
<td>0.52</td>
<td>0.58</td>
<td>0.20</td>
</tr>
</tbody>
</table>

Figure 9. Moving object detection result on VIVID egtest01. Previous frame, current frame, matched features, warped image, three frame difference, morphology result, enclosing rectangles/keypoint mask, blended pixel-wise result and result vs. ground-truth are shown from top left to bottom right respectively.

Figure 10. Parallax handling result on car6 sequence. From top left to bottom right; pixel-wise results without parallax handling, epipolar lines for keypoints present on moving object candidates, final results before parallax handling and after parallax handling.

We carefully pick the sequences to facilitate the inclusion of a variety of scenarios and effects, such as motion parallax, occlusion and altitude/viewpoint/object size variation.

We propose a feature-based, lightweight pipeline that draws its strength from object-wise decisions while eliminating parallax-induced false positives. This pipeline makes use of two separate feedback loops to realise accurate homography and fundamental matrix calculation. We evaluate the proposed pipeline for scenarios with and without parallax, and observe favourable quantitative results with near-real time performance on embedded GPU-based SoMs. Our quantitative results will serve as a future benchmark for the community.

As future work, we plan to use three frame geometric constraints instead of two to better handle degenerate motion cases [29]. We plan to use our own camera, allowing us to calibrate and obtain camera parameters, thus using essential matrix instead of fundamental matrix. We believe this will lead to more robust and accurate calculation of epipolar lines. Additionally, we plan to use the IMU measurements to dynamically adapt the system parameters that depend on altitude. Migrating the remaining parts of the pipeline to GPUs is also considered for faster execution. Lastly, our work on LAMOD dataset will be updated with new ground-truths and sequence-wise labels (i.e. presence of occlusion, motion parallax, etc.) along with benchmark results of existing techniques.

LAMOD which extends the ground-truth annotations of VIVID and UAV123 datasets for moving object detection.
References


