Trajectories and camera motion compensation in aerial videos

Charles Beumier, Xavier Neyt
CISS department
Royal Military Academy
Brussels, Belgium
charles.beumier@elec.rma.ac.be

***Abstract*—This paper presents a method for trajectory extraction in videos acquired with a slightly moving camera. Trajectories are initialized at Shi-Tomasi [1] feature points and tracked thanks to the Lucas-Kanade [2] algorithm from the openCV library [3]. New feature points are regularly introduced to compensate for track losses and to handle newly appeared objects. A simple and fast method for camera motion compensation has been implemented, using the fact that near static scene points undergo an equal translation between any two images. Local histograms of displacement normally exhibit a clear peak since our application considers scenes with relatively few moving targets. These peaks designate which tracks and thus which points are best to estimate the homographies representing motion between frames of the sequence. Tracking results for pedestrian and vehicles with camera motion compensation are shown and discussed for two test cases with different environment, scenario and different video quality. The usefulness of camera motion compensated trajectories is demonstrated by an example of target classification based on track maximal speed and possible hotspot detection from long track pauses.**

Keywords-surveillance; Unmanned Aerial Vehicle; feature points; trajectories; camera motion compensation

I. INTRODUCTION

The domain of surveillance is in full development on the one hand due to the regular recall from terrorism events and on the other hand thanks to the technological advances in sensors and platforms. In particular, drone (or UAV, Unmanned Aerial Vehicle) applications have literally taken off these last years. Several cues make UAVs appropriate for surveillance. First, the aerial view is in many circumstances advantageous to track people or vehicles and to estimate their position and speed. Secondly, under the condition of flight authorization, the flexibility of a drone allows for a quick deployment and with a reduced risk for human lives. Thirdly, a wide range of drones with different properties is available, from the long endurance and high altitude vehicles suited for persistent surveillance, to the small and agile copters adapted to more dynamic interventions.

This paper describes a method to extract vehicle and people trajectories from aerial videos captured from a UAV with a camera with fast rate but relatively low quality. As a preliminary study, Full HD videos were considered to design a solution for tracking in reasonable processing time. The objective of our research is to address the problem of persistent surveillance through trajectory analysis. We currently deal with rather small images and fast rate, as opposed to the usual wide area imagery captured at low rate for which a survey on moving object detection was recently published [4].

The proposed approach for vehicle and people trajectory extraction will be useful for the detection of Improvised Explosive Devices (IEDs) for route clearance, subject of the IEDDET project [5] of the European Defence Agency program. This project plans to detect IEDs or their possible indicators thanks to several sensors carried by Unmanned Ground Vehicles (UGVs) preceding the convoy. A UAV will capture RGB and thermal images to detect possible suspicious behaviors of vehicles and people in the vicinity of the convoy itinerary.

The next section describes the literature background in tracking and camera motion compensation that motivated our development. Section III details the implementation. Results are given in section IV for two test cases: a road crossing captured by a smartphone in the Full HD video format and a square in front of a chapel in HD Ready resolution. Conclusions and future directions are given in section V.

II. LITERATURE BACKGROUND

As mentioned in the survey of Jalal and Singh [6], there are two opposite approaches for tracking. The first one detects objects or points of interest in an image and associates them to the tracks made of positions detected in previous images. The other approach, also called 'track before detect', consists in initializing tracks and in matching (for instance thanks to image patches or histograms) to follow the underlying object parts from image to image. We preferred this second strategy, avoiding the delicate object detection in images from videos captured by a moving camera.

Taken from another perspective, as presented by Teutsch [7], there are three ways to detect motion once a sequence has been compensated for camera motion: image differencing, background learning and foreground segmentation, and clustering of moving local features. Since camera motion compensation is typically solved with local features, the third way with clustering can be used to compensate for camera motion and detect absolute motion. This is the approach we have followed.

Detecting motion from the analysis of moving feature points has been adopted by many authors. In his thesis

Teutsch [7] applies independent motion detection to compensate camera motion and to generate motion clusters as initial object hypotheses. He suggests a minimum track duration of 5 frames (at 25 Hz) for the tracking of cars. Rodriguez-Canosa et al. [8] compare vectors obtained from tracking with vectors synthesized from camera motion estimated by method [9]. A dynamic object track is initiated if at least five feature points are tracked between two consecutive frames, and if these tracks differ significantly from the synthetic vectors. Kalal, Mikolajczyk and Matas [10] select good tracks by focusing on small position changes of feature points after a forward then backward tracking. In their comparative tests the superior tracking performance of this forward-backward strategy is attributed to the fact that selected feature points are based on more than a single image. We defend the same argument but favor more images by tracking points longer than one or five frames forward. In our application an interval of 5 seconds appeared appropriate to have stable point displacements while keeping a high probability of continuous tracking for many feature points. Weak or wrong feature points are likely to be lost or to become erratic after a few frames.

Many feature point detection algorithms have been designed and some also provide attributes to help matching them between different images. Acronyms such as FAST [11], SIFT [12] or SURF [13] are now common in the literature [7, 14]. We started our development with Shi Tomasi [1] for feature point detection and Lucas Kanade [2] for tracking. We took advantage of the existing implementation of the openCV library. This pair of algorithms, generally referred to as KLT (Kanade-Lucas-Tomasi), satisfies our requirements in term of precision and speed for camera motion compensation and track detection. In the domain of camera motion compensation, KLT was recognized as the most commonly used method by Teutsch [7] in 2014. Bonin-Font [14] also found KLT as the best alternative for their real-time application in robot navigation compared to SIFT, SURF and FAST methods. In their experimental survey, Smeulders et al. [15] compared 19 visual trackers on a large dataset and for many different aspects of difficulty. As one of the oldest approach, KLT is part of the tests and does not demerit in comparison with other methods.

Similarly to some of the abovementioned works, we realize camera motion compensation thanks to the feature points tracked for target detection. Candidates for a good estimation are the feature points related to static scene points. These will be used to find the homography that registers one image to another. We propose a feature point selection mechanism that exploits the similarity of image feature point displacement proper to static scene points lying close to each other. For a limited time interval and thanks to the coherence in consecutive frames, these feature points are very likely to be correctly tracked.

Once compensated for camera motion, the tracks can be analyzed to address the problem of surveillance [16], in search for suspicious behavior or presence in sensitive areas. We propose the maximal speed along tracks to classify targets and the long pauses along these tracks to localize possible key events. Results are presented in the form of a scene image with superimposed tracks with some automatic interpretation, like for video event summarization [17]. The complete system is however planned to be semi-automatic in the sense that an operator will be tasked to confirm the automatically flagged situations by interpreting the related small video segments.

III. IMPLEMENTATION

This section details the implementation of the components present in Fig. 1 and aiming at object tracking, camera motion compensation and trajectory analysis. With the final objective of detecting suspicious behaviors, the system currently highlights target tracks from their maximal speed and localizes long target pauses.

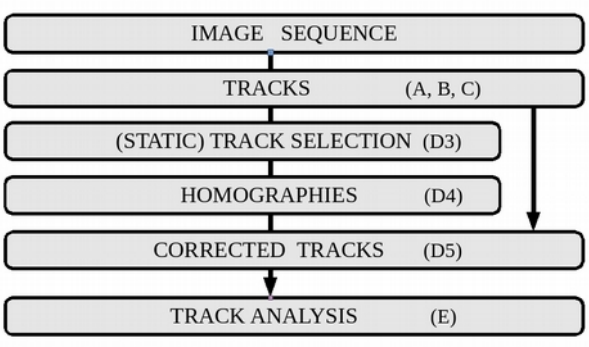

Figure 1. Synoptic of the presented development.

A. *Feature point detection*

Feature points have been detected using the method of Shi Tomasi [1], thanks to the openCV library call named “goodFeaturesToTrack”. It is a corner detector based on the matrix of local gray level derivatives. The quality as corner for a point is assessed through the two eigenvalues of the matrix. The parameter *qualityLevel* controls up to which quality corners are returned, what indirectly determines their quantity. This number may be limited by parameter *maxCorners*. The third argument *minDistance* specifies the minimum distance between returned corners, influencing the maximal density, especially in highly textured areas.

B. *Feature point tracking*

For the set of feature points detected in a first image, corresponding points will be searched for in the consecutive images of the sequence. We used the opencv implementation of the Lucas Kanade algorithm [2]. When specifying a list of points and its corresponding image A, the call returns a list of corresponding points for a specified image B. A status is also returned to mark which of the points were found a poor correspondence.

The algorithm registers the local area around a feature point in image A with image B by using the local gradient in A and the gray difference in A and B [2]. This estimation is refined iteratively and is improved by considering a window of pixels with size *winSize* around the feature point. Since the algorithm looks iteratively for the solution, several

termination criteria have been provided for which we use the default values.

In their method, Lucas and Kanade [2] make a mathematical development for motion estimation which is valid if the displacement is of limited extent. They introduced a multi-level pyramid to ensure this condition of small displacement, starting from the lowest resolution in the pyramid and refining the estimation progressively through the pyramid levels up to the original resolution. We adopted the default value 2 for the argument *maxLevel* meaning that three levels of the pyramid are used (the original scale and the next two lower scales).

C. Feature point re-detection

Tracking points with the method presented so far suffers from two problems. First, some tracks will be lost over time due to image quality, occlusion or change in target appearance. Secondly newly appeared targets will not be tracked. To address these issues, feature points are regularly detected and the ones sufficiently far from any current track point initiate new tracks. The minimum distance of a new feature point with an existing track point was set equal to the parameter *minDistance* of the feature point detector.

The frequency of introducing new tracks results from the compromise between having a fast reaction to the two aforementioned problems and a reasonable processing time. We adopted a value of 1 Hz, meaning that new tracks can be initiated each second. The time required for feature point extraction is roughly similar to the tracking procedure between two consecutive images.

D. Camera motion compensation

Even in the case of a stabilized drone, image acquisition suffers from camera motion, for instance due to the wind. These external influences should be compensated for so that trajectories of moving targets are accurate. Unfortunately for a moving camera, even static scene points may move in the imagery, what complicates camera motion estimation. Fortunately, static object image points close to each other undergo a similar movement.

We have designed a simple and fast method for camera motion compensation that exploits the similarity of static scene point tracks. It is based on the 5 following steps.

1) Division into sub-sequences:

In order to compensate camera motion thanks to feature point tracks, a long sequence is best sliced into sub-sequences. The idea is to limit the time interval so that it is covered by a sufficient number of static point tracks, preferably all over the image. On the contrary the sub-sequence duration should be long enough to discriminate static scene point tracks from slowly moving target ones. In our experiments, we adopted a sequence division into slices of 5 seconds.

2) Spatial division

The effect of camera motion is rarely homogeneous in the images due to the scene relief, the camera rotation and the perspective distortion. However, the displacement of static scene points is very similar in small neighborhoods.

The definition of locality is camera and application dependent. We adopted a division of the image area into an integer number of tiles, horizontally and vertically, so that the tiles are approximately 150x150 pixels. Since the procedure for track selection (see *3*) is very fast, the tile size could be adapted to reach a better selection of static tracks. This was not necessary in our tests so far.

3) Track selection

In order to get representative values of the vector field for the static scene, we propose to look for the most represented bin in the *dx* and *dy* histograms in each image tile. *dx* and *dy* are the components of the total displacement along tracks, between the first and the last frame of a sub-sequence. In our application, the number of static scene points is usually higher than coherent moving target points. The tiles for which this is not true will not impair the approach as long as there remain enough tiles with correct dx, dy estimation. In more difficult situations, this track selection could be enhanced by adding tracks which were selected in previous sub-sequences or by including votes from neighboring tiles, likely to be similar for static points.

The objective of this third step is to deliver a list of tracks normally corresponding to static scene points. For each tile, we select the tracks which populated the maximal bin of the dx and dy histograms, if these contain at least 3 counts. Mention that histogram filling for all tiles only requires one scan of all tracks present in the sub-sequence interval. The separation into two 1-D histograms instead of a 2-D one was motivated by a reduction in memory consumption and computation time. The bins are indeed 1-pixel wide and the displacement after a few seconds might reach a few hundred pixels.

The fact that the list of ‘static’ tracks would contain moving point trajectories is not an issue. The next step is resilient to quite a large ratio of outliers.

4) Homographies

A transformation between two frames of a sequence can be modeled by a homography if the scene points lie in a 3D plane. This condition is likely to hold (at least in approximation) in our application involving vehicles, especially for a flight sensibly higher than the scene relief.

The homography between the first and last frame of the sub-sequence is estimated by the *findHomography* openCV call. As a few moving targets or noisy tracks can pollute the list returned by the previous step, the RANSAC procedure [18] was specified in the call. This is indeed simpler that trying to detect and filter such undesired tracks. Thanks to the obtained homography, the corrected coordinates of the last frame track points are compared to the points of the first frame to decide which of the tracks are coherent with the homography. These form a new list of static scene point trajectories that will be used for camera compensation of the sub-sequence (step 5).

5) Camera compensation

Within a sub-sequence, the homography of each frame is derived from the points of the ‘static’ tracks selected in the previous step, compared to the same track points of the first frame of this sub-sequence. To get the transformations

relatively to the first image of the whole sequence, a simple homography matrix multiplication is required. Since the sequence slices were created with one frame overlap, the last frame homography *hmgrL* of the previous sub-sequence already maps towards the first sequence frame. The current sub-sequence homographies (except for the first frame) is multiplied by this homography *hmgrL*.

Disposing of all the homographies relatively to the first image of the sequence, each trajectory point is corrected. It is a fast operation involving a matrix multiplication that typically concerns one or two thousands points per image. The frames can also be warped by the homographies thanks to the openCV call *warpPerspective* to be registered to the first frame. This is useful to detect moving objects or obtain their shape by frame differencing.

Trajectory compensation for camera motion enhances the detection of slowly moving objects. Speed estimation is also more accurate. However, even after correction, static scene points might exhibit some image movement due to imprecise tracking or due to parallax, where the homography hypothesis of a planar scene does not hold.

E. Trajectory analysis

We propose the following trajectory classification. First, the tracks with small extent, normally corresponding to static scene points are classified as such and no more used (they were helpful for camera motion compensation). Other trajectories either refer to moving targets or to phantom points. Phantom points originate from image features which are incorrectly tracked. They usually exhibit erratic motion, with speed or direction incoherence.

We suggest to filter out phantom tracks thanks to a measure that we call *erraticity*. This measures the high frequency content from the distance between the trajectory curve and a low pass version of it. The low pass curve is obtained from the coordinate average of each point with its direct two neighbors. Our erraticity measure is the root mean square of the euclidean distance between the original trajectory points and its low pass version. It is a value in pixel which increases with irregularities in orientation (curvature) and in point spacing (change in speed) along the track.

For the remaining trajectories (i.e. neither static nor erratic) we propose the maximal speed as a first clue for target classification into vehicles or pedestrians. The speed is measured by pixel distance knowing the inter frame time interval and the rough ground sampling distance. This speed estimation is an approximation knowing that images are not ortho-rectified. However the targeted application concerns a high flight (500m) and a quite vertical point of view, thus limiting the discrepancies in the ground sampling distance. We suggest to measure the speed of a track as the percentile 95 of the speed distribution of its small tracklets with 1 second duration. It represents a value close to the maximal speed on the trajectory, with some tolerance against wrong instantaneous estimations.

As a second clue for trajectory analysis, we propose to look for long pauses in the tracks identified as belonging to moving targets. Stopping cars, loitering people or meeting individuals are examples of activities which may prove to be informative, especially when the location, time or duration details are taken into account.

We have considered the detection of long track pauses by measuring the distance between points along the tracks separated by a time interval of 5 seconds. For a distance lower than 25 cm, the candidate interval is checked to see if its points effectively stay in a small neighborhood. The coordinates of the central point are returned and constitute a possible activity hot spot.

In the followed ‘track before detect’ approach, the vehicles or pedestrians were not tracked as objects but thanks to their feature points. This allows for partial occlusion as long as some features remain visible. Grouping tracks into object has not been done yet. It was also not our intention so far to outline the targets neither to link partial tracks. Since camera compensation is achieved, image differencing can be used for target delineation. The object shape is of course a good candidate for further target classification. Track linking will also be necessary to address the problem of strong occlusion or change in appearance, when the tracks of a target have been interrupted due to tracking failure.

IV. RESULTS

A. Datasets

The method developed for tracking with camera motion compensation was first tested on a sequence captured from about 15 m above a road crossing with several cars, 2 bikes and a few pedestrians. Since the point of view is not vertical a quite large difference in image size exists between close and far objects. The image sequence consists of frames with a time interval of 0.2 s from a Full HD video (1920x1080) at 25 frames per second (fps), captured by a Samsung A5 (2016) smartphone.

For the targeted application in the context of route clearance, video data of a square in front of a chapel was captured from the SCHIEBEL Camcopter with a RGB camera in the HD Ready format (1280x720) at 30 fps.

For both datasets, the RGB pixel data were converted into gray values before being processed for key point detection and tracking.

B. Trajectory extraction and correction

For feature point detection, the parameters were set to get roughly 1000 points per Mpixel. This appeared to be a good compromise between a comfortable density of stable points spread over the images and fast processing. The parameters *qualityLevel* and *minDistance* were set respectively to 0.01 and 7. The tracking parameters *winSize* and *maxLevel* were given their openCV default values (respectively (21,21) and 3). We tested other values and noticed similar results for parameter changes in the [-20%,+20%] interval. We limited to 20 the number of new tracks regularly added to compensate for track losses and handle newly appeared objects.

The Lucas Kanade algorithm successfully tracked most moving objects. This is certainly helped by the favorable

conditions of our application: high frame rate (0.2s), similar appearance from a top view and rare occlusions.

Camera motion compensation was implemented as detailed in section III. The values about sub-sequence duration and space division or histogram bin size were slightly changed without major implication on the correction. The alignment error between the frames warped by their homography and the first frame was limited to a few pixels, depending on the sequences. This is small compared to the trajectory amplitude of moving objects or to the object size in case frame differencing would be used for segmentation.

C. The road crossing test case

Fig. 2 displays the trajectories detected in a short sequence of 4 seconds of the road crossing dataset. For visibility reasons we only show a part of the image. Nearly all tracks are clean, with no erratic parts, attesting the image quality of the video, the appropriateness of the Lucas Kanade approach and of the chosen parameters.

Thanks to their large extent, the trajectories of the moving cars and bicycles represent the first focus of attention in the image although they are much less numerous than the similar and small tracks of static scene points. The impact of camera motion may seem little in this example, but slowly moving pedestrians, such as present in the top right of the figure are hardly discriminated from their uncorrected trajectories.

Figure 2. Trajectories for a part of the road crossing dataset without camera motion compensation.

One beneficial effect of camera motion compensation is visible in Fig. 3 which displays the corrected trajectories of Fig. 2. The vehicle tracks are now rectilinear and more parallel to the road, what is likely to be closer to reality.

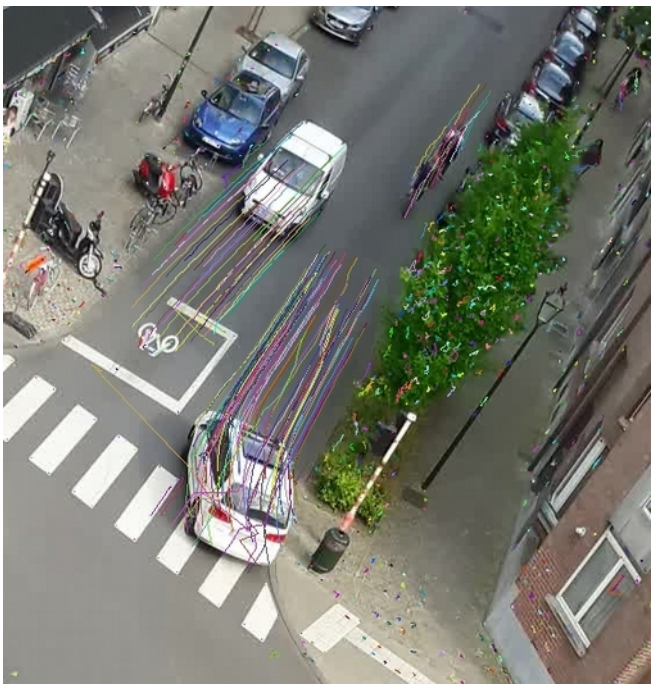

Figure 3. Trajectories of Fig 2 after camera motion compensation.

Also, the trajectories of most ‘static points’ are reduced to very short tracks. This is true for well defined feature points like the corners of the pedestrian crossing white rectangles. This is less the case for points in noisy areas like in the pedestrian pavement, much sacrificed by MPEG compression. Much worse are the cases of feature points in the trees. First they are not purely static, due to the wind, and secondly they are usually weak corners in the difficult texture area made of leaves. A very few examples in the image show the weakness in localization of feature points along some edges. After correction, their path follows the edge. They are probably among the worst feature points returned by the detector, having low corner quality. They are not so well localized in the direction along the edge but were accepted because we desired corners to cover as much of the image as possible for precise camera motion compensation.

Quite trivially after correction, moving objects stand out with their long path. This is obvious for the moving cars and bicycles but the advantage of corrected tracks is clearer for two pedestrians at the top right of the image in Fig. 3. They could have been missed in Fig. 2 since their feature points describe small non-oriented patterns hard to discriminate from the ones of static scene points. Once corrected, these trajectories show a clear orientation with a speed making them good pedestrian candidates.

D. Surveillance for route clearance

Our target application concerns vehicle and people tracking in the context of route clearance. A drone precedes a convoy to detect suspicious activities. In the EDA program IEDDET, a SCHIEBEL Camcopter equipped with a RGB camera will send a continuous MPEG video to the base station. The area location with suspicious activities of individuals or vehicles should be notified to the UGV

responsible to detect IED indicators before the convoy arrives.

In comparison with the first test case based on Full HD video, we faced the following image quality reduction:

- the pixel quantity per image is 56 % smaller and the ground resolution twice as low;
- the image quality depends more on meteorological conditions, as the Camcopter has to be operated at a sufficient altitude (500 m in our case);
- the video may suffer from flow discontinuities due to transmission defects.

The acquisition campaign happened during a dark and rainy day. It was not unusual at the 500m altitude to get an image partially fogged by a cloud. Fig. 4 shows the detected trajectories for 50 seconds of video (250 frames). They were superimposed on the Google image of the area.

The retained scenario contained a dark blue van bringing in and out people, a small truck passing by and a few people close to the chapel and on the opposite side of the square. The adopted top viewpoint offered less perspective effects and a better speed estimation. The apparent camera motion was this time closer to a rotation. This is due to the wind effect on the Camcopter operated in auto pilot mode and trying to keep its focus on a specified area in front of the chapel. The same parameter values were used for trajectory extraction and camera compensation as for the first test case.

The lower image contrast and the lower resolution had a wrong influence on the tracking of feature points, what was noticeable in the amount of erratic tracks. The number of track interruptions also increased, but as a consequence of new situations. First, the vertical observation of a walking person suffers more from the variability of his shape contour, what disturbs tracking. Secondly, there were occlusions caused by a crane deployed in the scene at the bottom of the image. So far video discontinuities or occlusions causing broken tracks were not handled. Their number and consequences were however limited thanks to the multiplicity of tracks attached to each target.

Figure 4. Uncorrected tracks for 50 seconds of video superimposed on the google image of the chapel area.

Fig. 5 displays the trajectories corrected for camera motion compensation. Compared to the first test case we notice a moderately less precise correction, which is explained by the fact that the sequence is twelve times longer and that the imagery is of lower quality. Most static scene points describe a small blob after correction and not a single point. The correction is however effective and sufficient. The corrected moving object tracks become smoother, as expected.

Figure 5. Tracks of Fig. 4 after camera motion compensation.

With the help of the tracks displayed in Fig. 5, we can partly interpret the played scenario. A vehicle stopped in the middle of the image. Two persons left this vehicle. One went towards the image bottom right, the other one went to the trunk. A person left the chapel (carrying a canister) towards the trunk. Another person crossed the road from left to right, passing behind the vehicle. From the beginning till the end, a second vehicle arrived from left, overtook the stopped vehicle and left through the upper side of the image. In the meantime a person was loitering on the left side of the square.

Globally, this second test case showed that the approach, although developed for image videos of better resolution and quality could work on the images of our application. The negative point in the current implementation is the loss of tracks for some of the person feature points, mainly due to the deformation of the outline from the top view. This could be helped with a better illumination or by considering images from a thermal sensor. Also, a blob detector from image differencing is a valid alternative to globally track the ever changing structure of a moving person imaged from the top.

E. Trajectory analysis

For the second test case we applied the trajectory classification suggested in section III on the corrected tracks displayed in Fig. 5. The phantom tracks were detected with a threshold value for erraticity equal to 2 pixels. Erratic tracks are drawn in white in Fig. 6. This figure uses as background the google map image of the chapel area and the magenta color for large differences between the registered last frame and the first frame (showing the moving vehicles and people, if not hidden by the tracks). Erratic tracks are mainly located in areas with poor texture or close to the image borders where tracking is more delicate. Three vehicle trajectories were also identified as erratic. Tracking got confused after the vehicles passed under a crane, just before turning left at the bottom of the figure. Some vehicle feature points have indeed been stuck by the crane edges. This huge deceleration was reported by a high erraticity value.

For the remaining tracks, not related to static scene points and not recognized as phantoms, the maximal speed parameter was measured as explained in section III. The tracks with speed (in pixels per frame interval) above 0, 2, 4, 6, 8, 10, 12 and 14 were drawn respectively with colors dark blue, light blue, cyan, green, yellow, orange, red and dark red. With a ground resolution of 4.5 cm per pixel and a frame interval of 0.2 second we arrive at the speed limits 0, 1.6, 3.2, 4.9, 6.5, 8.1, 9.7, 11.3 km/h and above.

Many very low speed trajectories (in dark or light blue) correspond to another kind of phantom tracks, with sometimes quite large extent. Most originate from feature points located near the image borders where tracking is delicate. Others are feature points drifting along an edge and describing a linear track. Tracks in the vicinity of the image border should simply be omitted.

The remaining tracks, from light blue to dark red, so above 3.2 km/h, correspond to pedestrians and vehicles. The 2 vehicles went sufficiently fast for a while to have tracks drawn in red or dark red, so above 10 km/h. It should be said that vehicles were driving very slowly due to the large occupation of the area with people in exercise. The central vehicle was just stopping, but is still drawn with a color thanks to the speed definition with high percentile. One of the pedestrians was running, stopping then coming back on its way. His tracks are drawn in orange for an approximate maximum speed of 8 km/h, although the average speed is less. A high speed, even of limited duration is relevant to the operator. Another person has green tracks testifying a normal speed of 5 km/h. The person leaving the chapel with light blue tracks was naturally slower as he was carrying a fuel canister. A last person, loitering on the left of the area, has dark blue tracks. It is interesting to see how discriminative for this scenario was a clue as simple as the maximal speed.

The track pauses with a minimum of 5 seconds duration were detected for the moving target tracks. They are displayed as black disks in Fig. 6. They successfully designate the stopped vehicle, the loitering man, the static canister (before it was taken away) and the running man before he moved. Many pauses were detected in the vicinity of the crane since, as explained before, the crane edges trapped vehicle tracks, making them static points. There are also three black disks (on a green and on a yellow track) which are due to parasitic tracks, moving next to a target for a while and finally ‘abandoned’ in a fixed position.

V. Conclusions and Future Work

We have presented in this paper the extraction of compensated trajectories for feature points attached to moving objects. For this, feature points were detected thanks to the Shi Tomasi approach and tracked with the Lucas Kanade tracker of the openCV library. Camera motion was compensated for thanks to static scene points that were selected from the detected tracks which exhibit a similar displacement in image local areas. A homography modeled the camera motion for each image, from the assumption that a large part of the scene is planar.

The implemented tracking method was successful for our urban crossing road test case, even in the presence of trees and houses. For the test case on early warning for route clearance (part of the EDA/IEDDET program), we could classify the compensated tracks between people and vehicles, based on a maximal speed measure. Border and erratic tracks had to be filtered out to avoid outliers. It was even possible with speed to discriminate the walking, the loitering, the carrying and the running individuals. The detection of pauses in moving target tracks has shown its potential for the automatic localization of possible suspicious activities.

In the future, we plan to address the tracking issue of occlusion and the use of frame differencing for target confirmation or detection. The compensated trajectories will have to be further analyzed to highlight specific behaviors such as loitering, stopping or making a U-turn. Target interactions such as people meeting, group splitting or individuals leaving a car are also of importance in order to describe the key events of a video and ask an operator to interpret the situation and flag real suspicious events. In this way, the operator load for video analysis will be reduced from hours to a few seconds or minutes of critical events depending on the details of his mission.

Acknowledgment

We would like to thank the Belgian MoD and in particular the Royal Higher Institute for Defence for supporting this research. Many thanks to the SCHIEBEL’s team who operated the Camcopter and delivered aerial videos and to the Austrian Defence team who played the scenario on the ground.

References

[1] J.Shi and C.Tomasi, “Good features to track,” Proc. IEEE Conference on Computer Vision and Pattern Recognition, June 1994, pp. 593-600.

[2] B. Lucas and T. Kanade, “An iterative image registration technique with an application to stereo vision,” Proc. International Joint Conference on Artificial Intelligence, 1981, pp. 674-679.

[3] Itseez, “The OpenCV reference manual,” release 2.4.12, July 2015, https://opencv.org/releases.html.

[4] L. Sommer, M. Teutsch, T. Schuchert and J. Beyerer, “A survey on moving object detection for wide area motion imagery,” IEEE Winter Conference on Applications of Computer Vision (WACV), 2016, pp. 1-9.

[5] “EDA programme launched to improve IED detection,” https://www.eda.europa.eu/info-hub/press-centre/latest-news/2017/01/12.

[6] A. Jalal and V. Singh, “The state-of-the-Art in visual object tracking,” Informatica, vol.36, 2012, pp. 227-248.

[7] M. Teutsch, “Moving object detection and segmentation for remote aerial video surveillance,” PhD thesis, 2014, http://dx.doi.org/10.5445/KSP/1000044922.

[8] G. Rodriguez-Canosa, S. Thomas, J. Del Cerro, A. Barrientos and B. MacDonald, “A real-time method to detect and track moving objects (DATMO) from unmanned aerial vehicles (UAVs) using a single camera,” Remote Sensing, vol.4, 2012, pp 1090-1111.

[9] G. Klein and D. Murray, “Parallel tracking and mapping for small AR workspaces,” Proc. IEEE and ACM Int. Symp. On Mixed and Augmented Reality (ISMAR 2007), Nov 2007, pp. 225-234.

[10] Z. Kalal, K. Mikolajczyk and J. Matas, “Forward-backward error: automatic detection of tracking failures,” 20th International Conference on Pattern Recognition, Aug 2010, pp. 2756-2759.

[11] E. Rosten and T. Drummond, “Machine learning for high speed corner detection,” Proc. of the European Conference on Computer Vision, May 2006, pp. 430-443.

[12] D. Lowe, “Distinctive image features from scale-invariant keypoints,” International Journal Computer Vision, vol. 60, 2004, pp. 91-110.

[13] H. Bay, T. Tuytelaars and L. Van Gool, “SURF: speeded up robust features,” Proc. 9th European Conference on Computer Vision, May 2006, pp. 404-417.

[14] F. Bonin-Font, A. Ortiz and G. Oliver, “Experimental assessment of different feature tracking strategies for an IPT-based navigation task,” Proc. 7th IFAC Symposium on Intelligent Autonomous Vehicles (IAV), Lecce Italy, Sep 2010, pp. 175-180.

[15] A. Smeulders, D. Chu, R. Cucchiara, S. Calderara, A. Dehghan and M. Shah, “Visual tracking: an experimental survey,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 36, no. 7, Jul 2014, pp. 1442-1467.

[16] W. Hu, T. Tan, L. Wang and S. Maybank, “A Survey on Visual Surveillance of Object Motion and Behaviors,” IEEE Transactions on Systems, Man, and Cybernetics – Part C: Applications and Reviews, vol. 34 no. 3, Aug 2004, pp. 334-351.

[17] H. Trinh, J. Li, S. Miyazawa, J. Moreno and S. Pankanti, “Efficient UAV Video Event Summarization,” Proc. 21st Int. Conf. On Pattern Recognition (ICPR 2012), Nov 2012, pp. 2226-2229.

[18] M. Fischler and R. Bolles, “Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography,” Graphics and Image Processing, vol. 24, No 6, Jun 1981, pp. 381-395.

Figure 6. Tracks of Fig. 5 classified as erratic (white), as static (blue dots) or as targets with different speeds (colors from blue to red). Black dots are track pauses. The magenta color highlights the large pixel differences between the first and the last frame (vehicles, people). The background is the Google image of the area.