

Movement Detection Based on Dense Optical Flow for Unmanned Aerial Vehicles

Regular Paper

Josef Maier^{1,*} and Martin Humenberger¹

¹ AIT Austrian Institute of Technology, Vienna, Austria

* Corresponding author E-mail: josef.maier.fl@ait.ac.at

Received 27 Jun 2012; Accepted 27 Aug 2012

DOI: 10.5772/52764

© 2013 Maier and Humenberger; licensee InTech. This is an open access article distributed under the terms of the Creative Commons Attribution License (<http://creativecommons.org/licenses/by/3.0>), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Abstract In this paper we present a novel computer vision-based movement detection algorithm which can be used for applications, such as human detection with unmanned aerial vehicles. The algorithm uses the deviation of all pixels from the anticipated geometry between 2 or more succeeding images to distinguish between moving and static scenes. This assumption is valid because only pixels which correspond to moving objects can violate the epipolar geometry. For the estimation of the fundamental matrix we present a new method for rejecting outliers which has, contrary to RANSAC, a predictable runtime and still delivers reliable results. To determine movement, especially in difficult areas, we introduce a novel local adaptive threshold method, a combined temporal smoothing strategy and further outlier elimination techniques. All this leads to promising results where more than 60% of all moving persons in our own recorded test set have been detected.

Keywords Movement Detection, Object Detection, Detection of Persons, Dense Optical Flow, Unmanned Aerial Vehicle, Epipolar Geometry, Random Sample Consensus, Adaptive Thresholding, Property based Filtering, Dynamic Environment

1. Introduction

In recent years unmanned aerial vehicles (UAVs) have gained increasing relevance in many applications such as video surveillance or field measurement. UAVs are quickly ready for action and can be easily equipped with various sensors such as gas detectors or video cameras. In emergency situations, such as the uncontrolled emission of poisonous gases or fluids, the robust sensing and prediction of possible damage is of the highest relevance to plan rescue missions. The initial schedule at the beginning of the disaster management is a key factor for success or failure. This includes the fast detection and localization of missing persons. Both tasks can be done by UAVs equipped with proper sensory mechanisms. The topic of this paper is the detection of moving (escaping) people based on computer vision with on-board cameras on UAVs.

2. Related Work on Movement and Human Detection

Reliable detection of moving objects from a moving camera is still an open research field. Especially in the field of Simultaneous Localization and Mapping (SLAM) this problem is well-known. Classical approaches of 2D motion segmentation try to separate the image flow into different

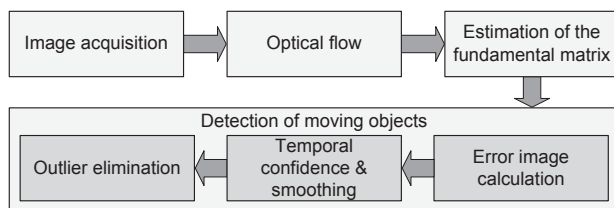


Figure 1. Algorithm workflow of the presented human detection approach.

regions either by looking for flow discontinuities [1], while imposing some regularity conditions [2], or by fitting a mixture of probabilistic models [3, 4]. An approach using a stereo camera was introduced by Solà [5] and relies on finding moving objects on the basis of feature detection. Therefore, Solà considers only objects that enter the image from the image borders and features for which the matching failed as moving objects. A similar approach using only one camera was presented by Migliore et al. [6]. Here, the detection of movement is based on continuously checking the intersections between three viewing rays belonging to the same feature viewed in three different camera poses. For the case that these intersections are not the same during camera motion or it does not exist, the feature is classified as dynamic. To address the uncertainties arising with each measurement, a probabilistic framework called Uncertain Projective Geometry [7] is used. Recently, Perera and Pasqual [8] presented simultaneously with Maier and Ambrosch [9] an approach for recognizing moving objects on the basis of detecting outliers. According to Perera and Pasqual, a moving object is detected if the distance between a found feature and its corresponding epipolar line is above a threshold. As this method uses features, some moving objects might be not detected. Maier and Ambrosch presented a similar approach which relies on dense optical flow data and an adaptive threshold. Another interesting method was introduced by René Vidal et al. [10, 11]. They introduce the Multibody Fundamental Matrix to represent the motion of each object relative to the camera between two frames. Thus, they can evaluate the number of moving objects, their motions and which point correspondence corresponds to which object by an accurate analysis of a formulation of the epipolar constraint extended to the multibody case. A drawback of this method is the high combinatorial complexity.

The remainder of the paper is organized as follows: Section 3 gives an overview of our approach. Section 4 introduces a novel point sampling strategy for selecting proper point correspondences of optical flow, which are further used in Section 5 to estimate the fundamental matrix between 2 temporal succeeding images. Section 6 explains our approach for detecting movement and Section 7 presents the achieved results. Finally, Section 8 concludes the paper and gives an overview of possible further research.

3. Algorithm Workflow

As stated earlier, this section gives a short overview on our approach. Therefore, Figure 1 illustrates the workflow of the movement detection algorithm. It is based on the

calculation of the deviation between each pixel in the actual image and its epipolar line given by its predecessor image. According to this, the first step is the image acquisition of the area by a digital video camera mounted in a UAV. The second step is the calculation of the optical flow to determine the pixel changes of 2 succeeding images. The third step is the estimation of the fundamental matrix with the point correspondences of the optical flow. The fourth step is the movement detection itself where we calculate the sparse error image first. From these error values the local adaptive thresholds for detecting moving objects are calculated. With these and the sparse error image, the dense error image is computed. Areas in this image with error values unequal to zero are candidates for moving objects. In the next step, only areas that were also detected in another 2 temporal succeeding error images are not rejected. In this step it might be the case that areas of real moving objects are rejected and others are erroneously accepted. To counteract this effect, a method to filter the remaining candidates, as well as insert areas that have a high potential to be real moving objects, is used. The standard deviation of the grey values of a detected area in the original image is used to apply a final filtering step on the moving objects. The following sections describe the single steps in detail.

4. Point Sampling on Dense Optical Flow Data

For the correspondence search and computation of the dense optical flow we use the pyramidal implementation of the Lucas Kanade Feature Tracker [12]. It is used to calculate the fundamental matrix F with several point correspondences which should fulfil specific requirements, such as independency. This means that no more than 2

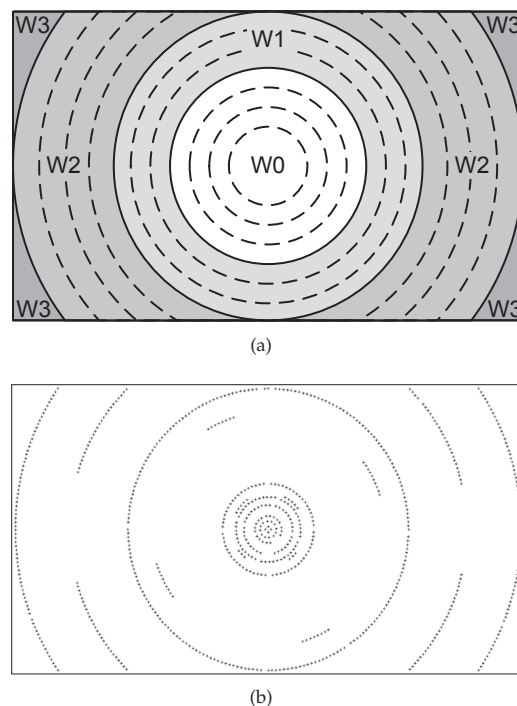


Figure 2. Selection of point correspondences. (a) Different image areas. (b) 700 selected point correspondences on a real image.

points should lie on one line because more points on the line contribute nothing to the linear system of equations of the 8-point algorithm. To fulfil this, points located on circles are selected, indicated by the dashed lines in Figure 2(a). As can be seen in Figure 2(b) on the right side of the outer circle, a small gap between selected point correspondences is present. This is due to the fact that the reliability of the point correspondences is increased by applying a threshold on the distance of a point in the first image to its corresponding point in the second image.

Moreover, the image is divided into 4 areas (cf. Figure 2(a)) [9]. These are denoted as $W0$, $W1$, $W2$ and $W3$. Depending on the amount of radial distortion, correspondences are selected from areas $W0$ to $W2$. This is due to the fact that the effect of radial distortion in the midmost circular area is insignificant for normal perspective, non-catadioptric, and non-wide-angle cameras where the distortion centre is in or near the centre of the image [13, 14]. As in many cases the distortion is not exactly known, we only use point correspondences from area $W0$. The points from $W1$ and $W2$ are only used for calculating the accuracy of the fundamental matrix F in these areas (see Section 5 for more details). Point correspondences from area $W3$ are neither used for calculating the fundamental matrix nor for calculating the accuracy of it in this area because here also little distortion decreases the accuracy significantly. This can be traced back to optical effects that arise near the borders of the aperture and with increasing distance from the optical axis of the lens system.

5. Estimation of the Epipolar Geometry Using a New Robust Method

The fundamental matrix F is computed using

$$\mathbf{x}'^T F \mathbf{x} = 0. \quad (1)$$

and the normalized 8-point algorithm [14, 15] with (homogeneous) point correspondences $\mathbf{x} \leftrightarrow \mathbf{x}'$. The state-of-the-art algorithm to deal with overdetermined systems is RANSAC [14, 16]. The RANSAC algorithm performs well for the case that nearly noise-less data and a small proportion of outliers is present. Otherwise, the RANSAC algorithm for eliminating incorrect correspondences can cause a high computational effort. One reason for this is the calculation of the error (distance between a point and its corresponding epipolar line) for every correspondence over many iterations, particularly when the number of available correspondences is high. To overcome this, we use a novel approach which has a predictable runtime and is suited for our application. To achieve real-time behaviour, other methods like pre-emptive RANSAC [17] or ARRSAC [18] also exist.

First, a set of point correspondences is selected according to the described point sampling in Section 4. This set of point correspondences has to be large enough (e.g., 300 correspondences) to contain at least 50 inliers. Thus, a least-squares solution for the inliers that also gives a good estimate for noisy data can be achieved. From the set of point correspondences a semi-random subset of 8 correspondences is extracted. Semi-random means

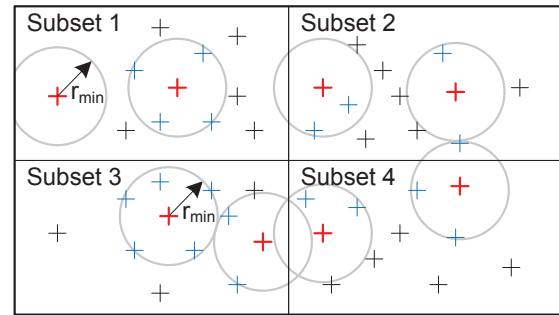


Figure 3. Segmentation of point correspondences into 4 subsets with a minimum distance of r_{min} between them: the red crosses mark the selected, the blue crosses the non-permissible point correspondences and the grey circles indicate the areas where a further selection is prohibited.

avoiding randomly selecting correspondences that are concentrated on a small region in the image. The first point set is split into at least 4 equally sized point sets of different areas of the image, shown in Figure 3. According to this, an equal number of correspondences with a minimum distance of r_{min} between each other must be selected randomly from each of the subsets, cf. Figure 3.

After selecting 8 point correspondences, they are normalized and the fundamental matrix is calculated with

$$\begin{bmatrix} x'_1 x_1 & x'_1 y_1 & x'_1 & y'_1 x_1 & y'_1 y_1 & y'_1 & x_1 & y_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x'_n x_n & x'_n y_n & x'_n & y'_n x_n & y'_n y_n & y'_n & x_n & y_n & 1 \end{bmatrix} \mathbf{f} = \mathbf{0}, \quad (2)$$

where $\mathbf{f}$ holds the entries of F and $\mathbf{x} = (x, y, 1)^T$ are the homogeneous coordinates of the point correspondences in the first and $\mathbf{x}' = (x', y', 1)^T$ in the second temporal succeeding image [14, 15]. For solving Equation 2 with 8 correspondences, the LU factorization is used. The estimation of F is repeated a few times (e.g., 30 times) and the results F_i as well as their homography matrices H_i are saved. Afterwards, each F_i is normalized by a global homography H_g , so that the fundamental matrices F_{g_i} are comparable:

$$F_{g_i} = H_g^{-T} H_i^T F_i H_i H_g^{-1}. \quad (3)$$

The homography H_g is calculated by the following equation:

$$H_g = \begin{bmatrix} s_g & 0 & -\frac{s_g d_x}{2} \\ 0 & s_g & -\frac{s_g d_y}{2} \\ 0 & 0 & 1 \end{bmatrix} \quad \text{with } s_g = \frac{16\sqrt{2}}{2\sqrt{d_x^2 + d_y^2} + d_x + d_y}, \quad (4)$$

where d_x is the width and d_y the height of the image.

Afterwards, a few (e.g., 20) correspondences are semi-randomly (see above) selected from area $W0$ (from the same set that was sampled in the beginning) and a few (e.g., 20) correspondences are semi-randomly selected from the areas $W1$ and $W2$. After normalizing this correspondences with H_g , the errors d_{ij} and the distances from points to epipolar lines $d_{(t-1)ij}$ and d_{tij} are calculated for both, the first (temporal) and the second image $I_{O_{t-1}}$ and I_{O_t} by

$$d_{(t-1)ij} = \frac{\mathbf{x}_j^T \mathbf{l}_{(t-1)ij}}{\sqrt{l_{(t-1)1ij}^2 + l_{(t-1)2ij}^2}} \text{ and } d_{tij} = \frac{\mathbf{x}_j^T \mathbf{l}_{tij}}{\sqrt{l_{t1ij}^2 + l_{t2ij}^2}}, \quad (5)$$

whereas

$$\mathbf{l}_{(t-1)ij} = F_{gi}^T \mathbf{x}'_j \text{ and } \mathbf{l}_{tij} = F_{gi} \mathbf{x}_j \quad (6)$$

with $\mathbf{l}_{\{t-1,t\}ij} = (l_{\{t-1,t\}1ij}, l_{\{t-1,t\}2ij}, 1)^T$. Afterwards, $d_{(t-1)ij}$ and d_{tij} are combined by

$$d_{ij} = d_{(t-1)ij}^2 + d_{tij}^2. \quad (7)$$

These errors d_{ij} are calculated for every fundamental matrix F_{gi} and the selected correspondences $\mathbf{x}_j \leftrightarrow \mathbf{x}'_j$. The errors are partitioned into d_{W0ij} and d_{W12ij} to distinguish between errors in the area $W0$ and errors in the areas $W1$ and $W2$. As there could be very large errors $d_{(t-1)ij}$ and d_{tij} present that would affect the algorithm, a threshold $t_F = 10s_g$ is used in

$$d_{tij} = \begin{cases} t_F + \frac{d_{tij}}{10} & | \quad 10t_F > d_{tij} > t_F \\ 2t_F + \frac{d_{tij}}{10^2} & | \quad 10^2 t_F > d_{tij} \geq 10t_F \\ 3t_F + \frac{d_{tij}}{10^3} & | \quad 10^3 t_F > d_{tij} \geq 10^2 t_F \\ 4t_F & | \quad d_{tij} \geq 10^3 t_F \end{cases} \quad (8)$$

to limit the error values. In the next step, the overall errors

$$d_{W0i} = \sum_j d_{W0ij} \text{ and } d_{W12i} = \sum_j d_{W12ij} \quad (9)$$

for each fundamental matrix F_{gi} are calculated and combined with $d_i = d_{W0i} d_{W12i}$. According to these error values, the fundamental matrix F_{gi} with the smallest error d_i is selected. In addition, the lower and upper quartiles of the corresponding error values d_{W0ij} and d_{W12ij} are rejected. From the remaining error values, the mean values μ_{W0} and μ_{W12} are calculated. Afterwards, if

$$\mu_{W0} < t_{W0} \text{ and } \mu_{W12} < t_{W12}, \quad (10)$$

with the thresholds given as $t_{W0} = 2(1.5s_g)^2$ and $t_{W12} = 2(3s_g)^2$, the error values d_{S_i} are calculated for all selected point correspondences (the large set from which the correspondences were randomly selected to estimate the fundamental matrices F_i) by Equations 5, 6 and 7 using the selected F_{gi} . All correspondences for which their error value exceeds a certain threshold (e.g., $d_{S_i} > 2(0.75s_g)^2$) are marked as outliers and are therefore rejected. From the remaining correspondences (the inliers) a final fundamental matrix F can be calculated using the singular value decomposition (SVD). If Equation 10 does not hold, an overall error of

$$\epsilon_{F_i} = \frac{1}{\log(\mu_{W0}) \cdot \log(\mu_{W12})} \quad (11)$$

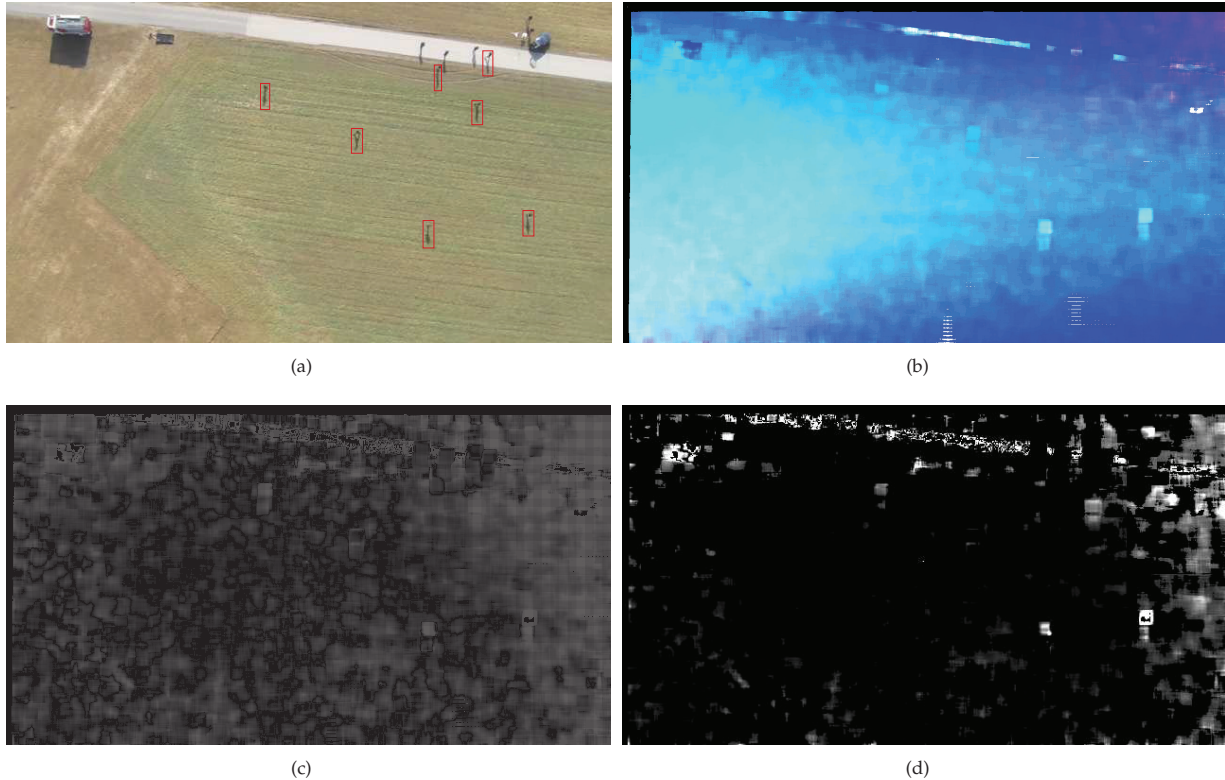


Figure 4. Real-world image with corresponding optical flow and the error images: (a) Camera image, moving persons are marked red. (b) Optical flow results: The colour indicates the amount of displacement. (c) Sparse error image (only the error value of every 4th point correspondence is calculated). The brighter a pixel the larger the error. (d) Error image after applying the local adaptive thresholds.

corresponding to the selected F_{g_i} is calculated and both (ϵ_{F_i} and $F_{F_i} = F_{g_i}$) are stored. Subsequently, the whole procedure is repeated until either a F_{g_i} is found for which Equation 10 holds or if the number of iterations reaches a threshold (e.g., 10 iterations). If the latter is the case, the fundamental matrix F_{F_i} with the smallest error value ϵ_{F_i} is selected. Therefore, a classification into inliers and outliers, and the estimation of the final fundamental matrix F can be performed as described before.

However, one disadvantage of the proposed algorithm, in comparison to RANSAC, is that it might perform slower when the number of point correspondences and the proportion of outliers are very small.

6. Movement Detection

After estimating the fundamental matrix, the epipolar errors, which are the distances between the pixels and their epipolar lines, are calculated. An adaptive threshold has to be calculated that is small enough to recognize slowly moving objects, but is large enough to reject noisy pixels. Afterwards, the dense error image is calculated utilizing local adaptive thresholds and a sparse error image. The next sections describe in detail how these steps are done.

6.1. Epipolar Error Calculation

For this reason, the error value is calculated for every point correspondence $x_i \leftrightarrow x'_i$ marked as inlier during the estimation of the fundamental matrix (see Section 5) using Equations 5, 6 and 7, as well as the final fundamental matrix F . From the resulting error values, the quartiles are calculated and distances that are below the lower and above the upper quartile are rejected. From the remaining distances the arithmetic mean value μ_g and from all inliers the standard deviation σ_g are calculated.

6.1.1. Sparse Error Image and Global Threshold

Experimental tests showed that a good global threshold t_g can be calculated with $t_g = 2\sigma_g + \mu_g$. In many cases, the afore mentioned error is not uniformly distributed over the whole image, especially when distortion is present. As a result, a global threshold might be not the best choice for every situation. Therefore, we are using local adaptive thresholds t_{ij} . To compute these thresholds a global upper threshold $t_{gu} = n_{gu}t_g$ is necessary in the first place. As this threshold is used to exclude correspondences that have a too large error, care has to be taken that t_{gu} is not too small. This strongly depends on the expected maximum pixel deviation caused by the fastest moving object on the

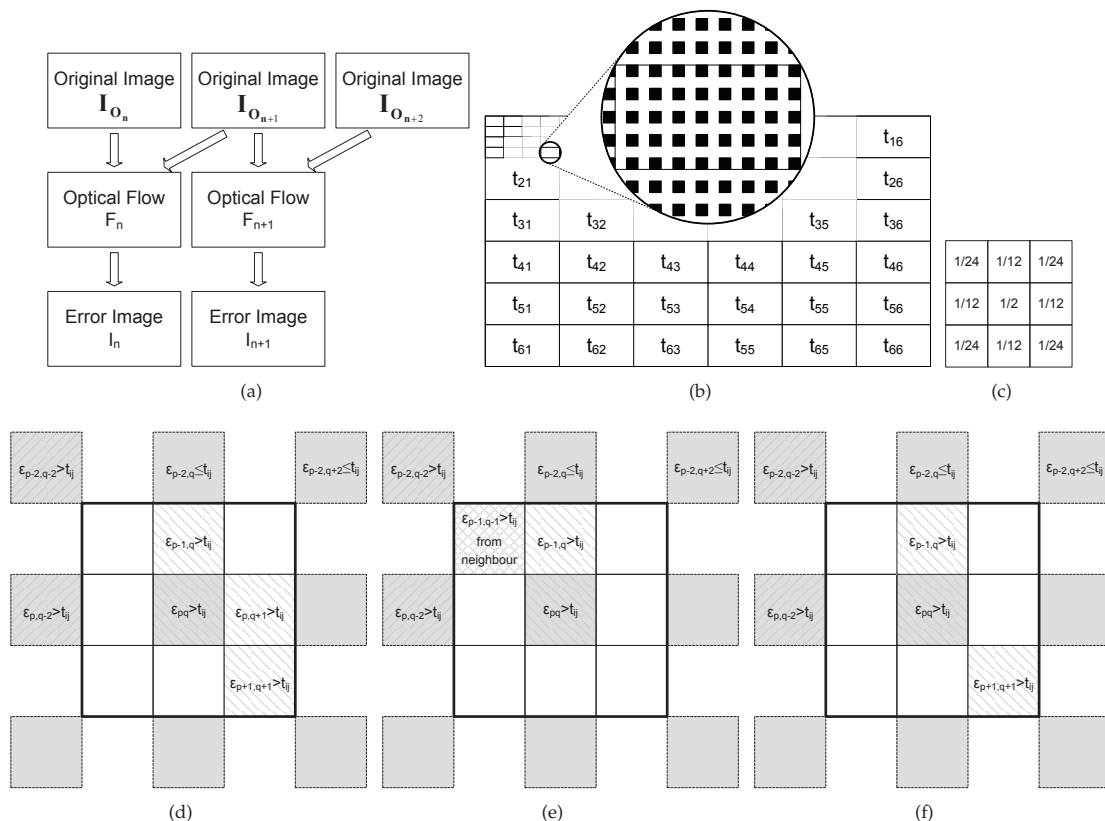


Figure 5. Error image emergence, local adaptive thresholding and error filtering. (a) Emergence of error images. (b) Different areas in the image with their unique adaptive thresholds t_{ij} . (c) Weighting-mask for weighting a local adaptive threshold with its neighbours. (d) The shown error values ($\epsilon_{p-1,q}$, $\epsilon_{p,q+1}$ and $\epsilon_{p+1,q+1}$) around the centre pixel $\epsilon_{p,q}$ (hatched from top left to down right) are accepted because more than 2 error values are above the threshold t_{ij} . (e) The error value $\epsilon_{p-1,q}$ above the centre pixel $\epsilon_{p,q}$ is accepted because an error value $\epsilon_{p-1,q-1}$ (grid in the background) that was calculated from the neighbour-centre-pixel $\epsilon_{p-2,q-2}$ or $\epsilon_{p,q-2}$ (hatched from down left to top right) was found. (f) All the shown error values ($\epsilon_{p-1,q}$, $\epsilon_{p,q}$ and $\epsilon_{p+1,q+1}$) inside the large black rectangle are set to 0 because only 2 error values greater than t_{ij} and no error value calculated from a neighbour-centre-pixel were found.

ground. Moreover, this depends on the altitude of the UAV, the lens and the frame rate of the camera, as well as on the camera resolution. Thus, n_{gu} should be set according to the application. For our application, $n_{gu} = 5$ whereas t_{gu} should not be smaller than 2.25 (otherwise it is set to this value). After calculating t_{gu} , the error values ϵ_{pq} are calculated for every 4th point correspondence in the image (every second row and second column) by Equations 5, 6 and 7. If an error value $\epsilon_{pq} \geq t_{gu}$, ϵ_{pq} is set to 0. From the resulting error values the sparse error image is created, cf. Figure 4(c). The emergence and temporal affiliation of an error image is shown in Figure 5(a).

6.1.2. Local Adaptive Threshold

For calculating the local adaptive thresholds, the sparse error image is divided into 36 areas (6×6), as can be seen in Figure 5(b). Each of these areas is again divided into 16 areas (4×4) which is shown for one of the former areas in the left upper corner of Figure 5(b). As can be seen in the zoomed area of Figure 5(b), each of these afore mentioned small areas contains many error values (indicated as black squares). To calculate a local adaptive threshold, one error value unequal to zero is selected from each of these small areas. If all error values in an area are zero, the value of a neighbouring area is selected. If this value is also not available, the global threshold t_g is used. After selecting 16 error values, the arithmetic mean value μ_{ij} is calculated from these values excluding the values below the lower and above the upper quartile. In addition, the standard deviation σ_{ij} is calculated from these 16 values. Using μ_{ij} and σ_{ij} , the local adaptive threshold can be calculated with

$$t_{ij} = 0.75(\mu_{ij} + 2\sigma_{ij}) + 0.25t_g. \quad (12)$$

After computing all t_{ij} , they are weighted with their neighbouring thresholds using the weighting mask in Figure 5(c).

6.1.3. Dense Error Image

In the next step the remaining error values (that were not computed in the sparse error image) can be calculated. Hence, error values surrounding an already calculated error value in the sparse error image are only computed, if this *centre-pixel* is larger than zero (cf. Figure 5(d)). In addition, more than 2 of these new calculated error values must be above t_{ij} (which belongs to this image area) and below t_{gu} , as can be seen in Figure 5(d). Otherwise, they (including the centre-pixel) are set to zero, which is shown in Figure 5(f). An exception to this is the presence of an already calculated error value on a position surrounding the centre-pixel. According to this, the value of the centre-pixel (and the values at the surrounding positions) is not set to zero if the latter is the case and at least one additional error value is above t_{ij} and below t_{gu} , cf. Figure 5(e). Applying this procedure to the whole error image, connected areas (areas with error values greater than zero) arise in the error image that are candidates for moving objects. An example is shown in Figure 4(d). Furthermore, it can be recognized in this error image that the number of areas is increasing with the distance from the image centre. This is due to the presence of radial

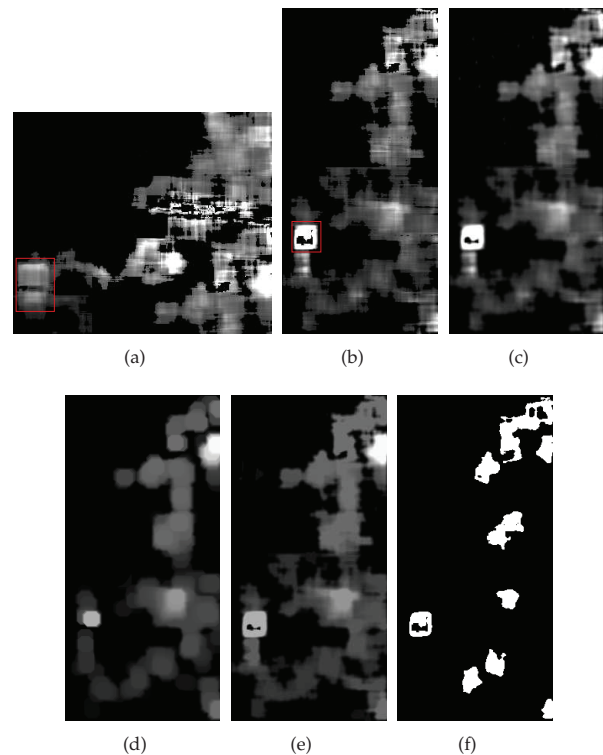


Figure 6. Treatment of large connected areas in error images. (a) Section of an error image with a large connected area including a detected moving object (marked red) which has a similar intensity (error value) than the rest of the area. (b) Section of an error image with a large connected area including a detected moving object (marked red). (c) Spatial filtering of the area using an averaging filter. (d) Morphological opening [19] on the area. (e) Morphological reconstruction [20] of the area using the filtered area and the area with morphological opening. (f) Detected local maxima from the reconstructed area.

distortion in the image. Moreover, the quality of the error image strongly depends on the accuracy of the optical flow. If areas exist in the image where the optical flow cannot be calculated accurately because of not enough differences in the grey values, these areas are detected as moving objects. An example of this is the street and the shadow of the fire truck in Figure 4(a). Thus, incorrectly calculated optical flow for these areas can, besides other artefacts, already be seen in Figure 4(b). The resulting false detected areas can be seen in Figure 4(d). To overcome this problems, further filtering procedures are necessary which are described in the following sections.

As can be seen in Figure 4(d), areas exist that are very small or have an object-unlike shape. Thus, all areas that are too small (e.g., <9 pixels) or that feature a ratio of their side lengths which is larger than 1 : 10 are rejected. Moreover, areas that are too large are processed in a further step. Such an area can be seen on the right side of Figure 4(d) and is shown with larger scale in Figure 6(b). Such an area can be very large (up to the width or height of an image), especially when distortion is present or the optical flow calculation was inaccurate. As these areas can include a moving object, they should not be deleted. In most cases, moving objects inside such a large area feature higher

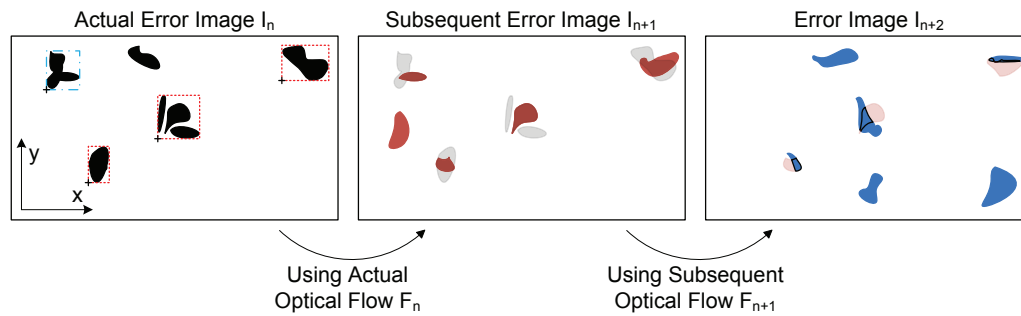


Figure 7. Matching of detected areas in 3 temporal succeeding error images: the red short dashed rectangles mark objects in image I_n with matching objects in images I_{n+1} and I_{n+2} . To illustrate the matching procedure, the (black) areas of image I_n are drawn with transparency in image I_{n+1} , whereas the overlapping areas in image I_{n+1} are drawn with transparency in image I_{n+2} . The overlapping areas of all 3 error images are bordered in black in error image I_{n+2} . The black area in image I_n , which is marked by a blue dotted rectangle, indicates a candidate that was not removed during the last matching (I_{n-1} , I_n and I_{n+1}), but has no corresponding area in image I_{n+2} .

error values than its surrounding. Therefore, we can isolate them by applying image processing on the sections of the error images. Such an extraction of areas with higher local error values is shown from Figures 6(b) to 6(f). In the first place, an averaging filter is applied to the original section of the error image to smooth the error values, as can be seen in Figure 6(c). Next, morphological opening [19] is applied to the filtered image. The result is shown in Figure 6(d). A circle is used as a structuring element whereas its size depends on the application (we use a circle-radius of 10 pixels). Applying morphological reconstruction [20] by utilizing the latter image and the filtered image, a solution like the one shown in Figure 6(e) can be achieved. From this, the local maxima can be calculated, cf. Figure 6(f). As can be seen, the area with the real moving object was extracted correctly. Unfortunately, also false positive areas are extracted. If such an area is very small, large or of an object-unlike shape (described before), it is rejected. This segmentation procedure also works well for smaller differences between the area of an moving object and its surrounding, as for the red marked area in Figure 6(a).

In the next step each detected area is enlarged at the edges by 6 pixels. This increases the probability of finding corresponding areas in the subsequent algorithm step - temporal smoothing (see next section). This is due to the fact that detected areas (of moving objects) in temporal

succeeding error images are in most cases not equal in form and size.

6.2. Search for Matching Error-Areas in Temporal Succeeding Images

After the candidates for moving objects were identified in 3 temporal succeeding error images, they are filtered. Hence, the position of such a candidate in the first temporal error image is reprojected in the second error image using the optical flow data. If a candidate is found in the reprojected area in the second image, the position of the overlapping area is reprojected in the third temporal error image. Afterwards, it is checked again if the reprojected area in the third image includes a candidate. During this procedure, the overlapping area of the candidates must be above a specific threshold. Experimental tests showed that the threshold for the overlapping area in the first 2 images should be set to 20% of the candidate's size of the first image. The overlapping area of all 3 images should be at least 10% of the candidate's size of the first image. This matching procedure is shown in Figure 7.

To achieve a better result during the matching procedure, a further step is introduced preceding the matching procedure. This step extends the detected areas of an actual error image with their corresponding areas (areas with a small distance between their left lower coordinates - see Figure 8) of the previous and subsequent error images.

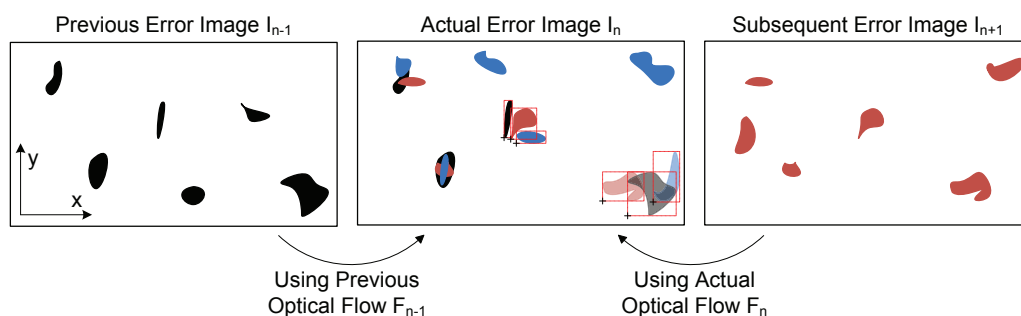


Figure 8. Extension of detected areas in an error image: the black crosses in the left lower corners of the red dashed rectangles indicate the coordinates for calculating the distances between the areas. Areas with small distances between each other are grouped together.

Accordingly, this step can only be carried out if the matching procedure was already performed for the first error image. Therefore, only areas that were not removed during the first matching procedure are extended by corresponding areas of the subsequent error images. Otherwise, the noise (falsely detected areas) would cause an enlargement of incorrectly detected areas. The red short dashed rectangles in Figure 8 mark 2 examples of such corresponding areas. Resulting areas that are too large are removed from the error images I_n and I_{n+1} . This is indicated by the areas in the right lower corner of error image I_n in Figure 8. As can be seen, the resulting error image I_n from Figure 8 is used as input (error image I_n) in Figure 7. Without the extension of the areas, the midmost candidate in Figure 7 would have been rejected.

As some real moving objects are sometimes not detected in an error image as a result of an inaccurate optical flow calculation or (radial) distortion, the temporal matching would fail. This could already be the case if only one area in one error image is missing. Thus, candidates that were detected once in 3 temporal succeeding error images and 4 greyscale images (original images), respectively, are stored for a sequence of 3 error images subsequent to the image where the matching was successful, cf. Figure 9(a). Their coordinates are updated for the succeeding error images by using the optical flow data. As a consequence, they can be seen as candidates for moving objects in the succeeding images, but they are not used within the matching procedure as input. If within this sequence of images a corresponding area is found again, it is

stored for a larger sequence of images (more than 3) and its coordinates are updated for every succeeding error image. The number of sequences depends on the following condition:

$$\zeta = \begin{cases} \frac{c+\bar{c}}{c-\bar{c}} & | \quad c \neq \bar{c} \\ 2\bar{c} & | \quad c = \bar{c}, \end{cases} \quad (13)$$

where c is the number of found corresponding areas and $\bar{c}$ is the number of missing corresponding areas for one candidate starting with the image where the candidate was found again. If $\zeta < 0 \vee \zeta > 10$, the candidate is rejected. Moreover, the candidate is no longer stored if it was detected again in 3 temporal succeeding images. In this case, it is detected during the matching procedure. An example concerning to this procedure is shown in Figure 9(b). As one can imagine, error image I_n in Figure 9(a) is equivalent (except area-extension) to I_{n+1} in Figure 9(b), whereas error image I_n in Figure 9(b) is equivalent to I_{n+2} in Figure 9(a).

For a further processing of the data, only the position (shown as small black crosses in the left lower corners of the rectangles in Figures 7 and 9) and size of the rectangles marking the candidates are of relevance. Thus, for every error image the afore mentioned information is stored for candidates that were detected during the matching procedure, for candidates that were detected up to 3 error images before and for candidates that were found again (see above). On the basis of this data, candidates that are very close to each other are combined and candidates that are too large are rejected.

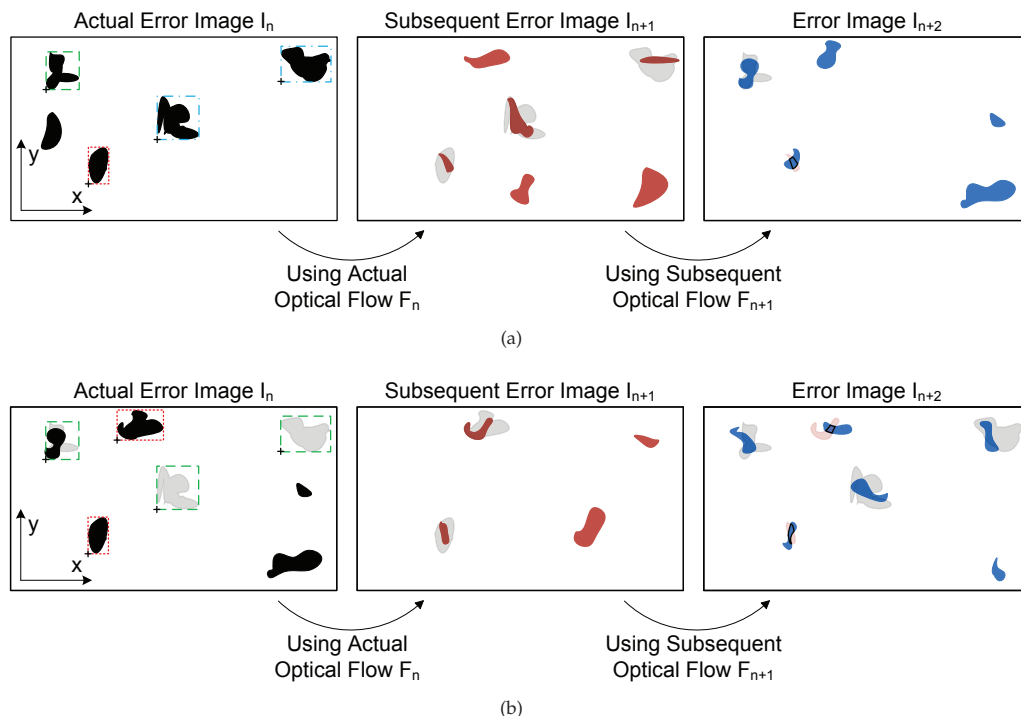


Figure 9. Preventing rejection of candidates for moving objects that were detected only in a few sequences. (a) Storage of candidates for which a further matching fails. These candidates are marked by a blue dotted-dashed rectangle. The green dashed rectangle marks a candidate for which a corresponding area was found again and the red short-dashed rectangle marks a candidate with successful matching. (b) Storage of candidates for which a corresponding area was found again. The 2 areas drawn with transparency in error image I_n indicate the position of the candidates, but they are not part of the error image.

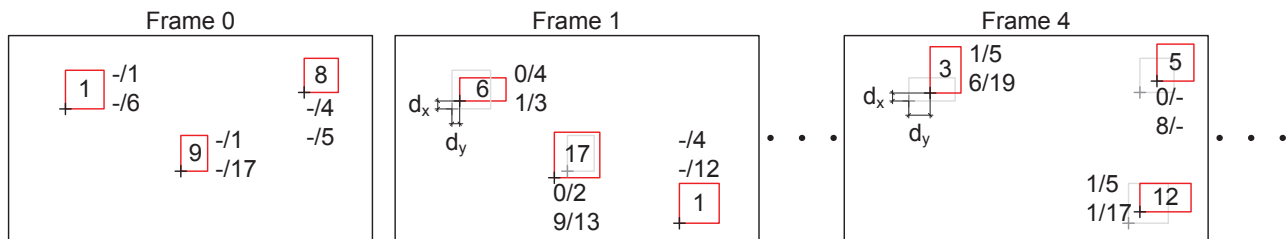


Figure 10. Referencing of candidates originating from different error images: the first line of the text next to each red rectangle indicates the previous and subsequent image number of the corresponding candidate. The second line indicates the number of the respective candidate in the specified image. The grey rectangles denote the positions and sizes of the candidates of the previous images, whereas the coordinates are calculated using the optical flow data. Candidates with a small distance d_x and d_y to each other are referenced.

6.3. Property-Based Filtering and Padding of Moving Object Sequences

In the next step the candidates are filtered according to their frequency of occurrence. Therefore, the positions of corresponding candidates in a previous and a subsequent image are stored for every candidate in an image. As these images sometimes are not temporally located one after another, the position of a candidate is calculated for at most 5 subsequent images by using the optical flow data until a corresponding (small distance between positions) candidate is found. Moreover, the search for a corresponding candidate is stopped and the candidate is removed in every frame if the number of images where it was detected compared with the number of images where it was not detected drops below 40% (starting after 10 computations). Figure 10 shows an example of referencing a few candidates.

For corresponding candidates that were found more than 6 times in different images, but feature some candidates that are not located one after another, candidates are inserted into the images on their corresponding positions where they are missing.

6.4. Elimination of False Positives

In the last step for all candidates the standard deviation of the grey values within the rectangle marking a moving object in the original image is computed. If the standard deviation for an 8-bit grey value image in the relevant

area is below a threshold, the candidate is rejected. This procedure is especially helpful for images where large areas feature a similar grey value (e.g., lawns, acres, etc.). The remaining candidates have a high potential to represent the positions of real moving objects in the images.

7. Results and Discussion

For testing the performance of the whole movement detection algorithm, a camera was mounted on a UAV. From the recorded video stream, a few series of consecutive images with a resolution of 1280x720 pixels were extracted. On each of these image series moving persons were imaged. In the example image series presented in this paper which consists of 108 images and 10 imaged people, only 8 of them are moving (not always at the same time) at different speeds (from slow to fast) in different directions. One of these images is shown in Figure 11, whereas the detected persons are marked with red rectangles. The position and the size of these rectangles corresponds to those described in Sections 6.2 and 6.3.

We tested the algorithm on images with distortion as can be seen in Figure 11(b) and with removed radial distortion, cf. Figure 11(a). Not surprisingly, the results are better for the image series with removed distortion. In this case, the algorithm correctly detected 60.16% of the moving persons. However, 30.52% of the marked areas were wrong. For the images with radial distortion, only 46.62%

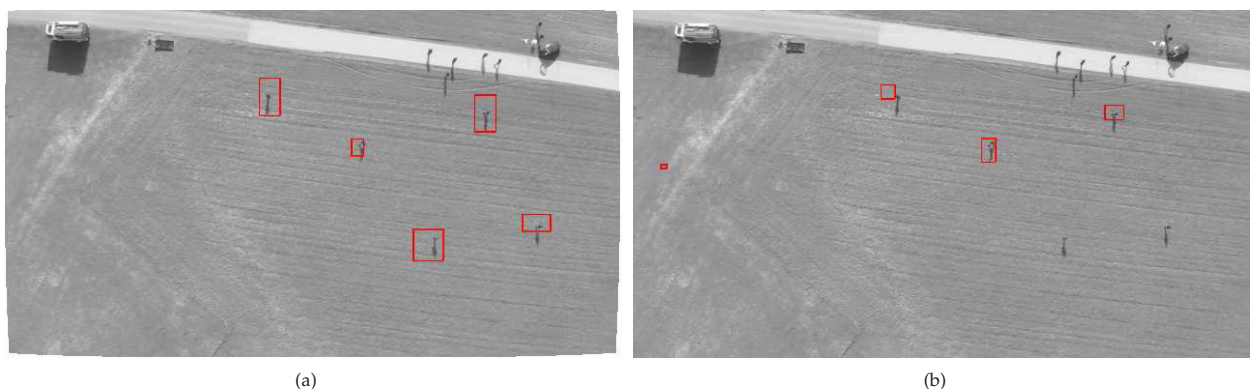


Figure 11. Detected moving persons in a real-world image. (a) Detection of moving persons in an image with removed radial distortion. (b) Detection of moving persons in an image with radial distortion.

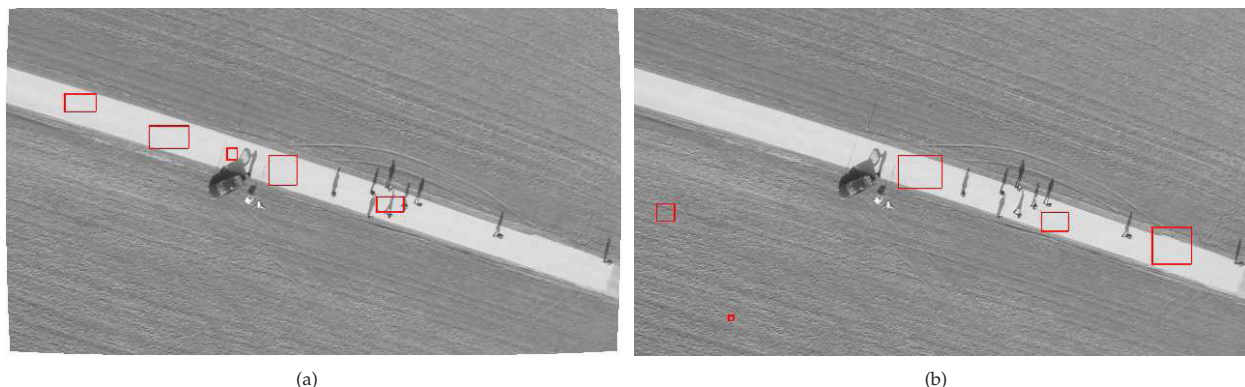


Figure 12. Example images with detected moving persons resulting from a very inaccurate optical flow calculation. (a) Detection of moving persons in an image with removed radial distortion. (b) Detection of moving persons in an image with radial distortion.

were marked correctly and 55.62% of the areas were incorrectly marked. The difference in the correctly marked areas between the images with and without distortion is due to the fact that slowly moving persons which are located at the borders of the images are indistinguishable from the noise resulting from the distortion. Moreover, the matching during the calculation of the optical flow failed for the street and the shadow of the fire truck. Excluding the persons and results from these areas, 65.03% of the moving persons are correctly detected for the images without and 51.39% of them are correctly detected for the images with distortion. Moreover, the proportion of incorrectly marked areas would be 24.31% for the first and 54.69% for the latter case. On the other hand, when the optical flow is very inaccurate in the area where the moving persons should be detected, the detection rate significantly falls. Figure 12 illustrates one image of such an image series. In this image series, which consists out of 179 images and 10 moving (not always at the same time) persons, all persons are moving on the street. As the street features nearly no variations in the grey levels, the matching of the optical flow calculation failed in this area. Thus, the detection rate for correctly detected moving persons falls to 23.64% (falsely marked areas: 73.4%) for the images without (see Figure 12(a)) and down to 15.96% (falsely marked areas: 85.58%) with radial distortion, cf. Figure 12(b). Therefore, we can see that the result of the algorithm strongly depends on the accuracy of the optical flow calculation and on the amount of radial distortion.

8. Conclusion and Future Work

We presented a novel method for movement detection based on a dense optical flow calculation for which we estimated the fundamental matrix by the well-known 8-point algorithm. For a robust estimation of the fundamental matrix we introduced a new method for rejecting outliers. This method provides a predictable run time which is especially important for situations with a large amount of correspondences and many outliers. Moreover, we presented that the detection of moving objects can be performed in the real world by providing local adaptive thresholds and a good approximation of the fundamental matrix. In addition, we showed that the resulting candidates for moving objects can be successfully

filtered by using information of many temporal succeeding images.

The next steps in our research will focus on an optimized implementation of a more sophisticated optical flow algorithm, as well as the realization of the whole movement detection system on an on-board smart camera of a UAV. Moreover, the algorithm will be tested on other scenes (sparse forest, urban areas, ...).

9. Acknowledgements

The research leading to these results was funded by the KIRAS security research programme of the Austrian Ministry for Transport, Innovation and Technology (www.kiras.at).

10. References

- [1] Anselm Spoerri. The Early Detection of Motion Boundaries. Technical report, MIT Artificial Intelligence Laboratory, 1991.
- [2] Michael J. Black and P. Anandan. Robust Dynamic Motion Estimation Over Time. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 1991.
- [3] Allan Jepson and Michael Black. Mixture Models for Optical Flow Computation. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, 1993.
- [4] Yair Weiss. Smoothness in Layers: Motion segmentation using nonparametric mixture estimation. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, pages 520–526, 1997.
- [5] Joan Solà. *Towards Visual Localization, Mapping and Moving Objects Tracking by a Mobile Robot: a Geometric and Probabilistic Approach*. PhD thesis, Institut National Polytechnique de Toulouse, 2007.
- [6] Davide Migliore, Roberto Rigamonti, Daniele Marzorati, Matteo Matteucci, and Domenico G. Sorrenti. Use a Single Camera for Simultaneous Localization And Mapping with Mobile Object Tracking in dynamic environments. In *ICRA Workshop on Safe Navigation in Open and Dynamic*

Environments: Application to Autonomous Vehicles, 2009.

- [7] Stephan Heuel. *Uncertain Projective Geometry*. Springer-Verlag Berlin Heidelberg, 2004.
- [8] Samunda Perera and Ajith Pasqual. Towards Realtime Handheld MonoSLAM in Dynamic Environments. In *Advances in Visual Computing*, volume 6938 of *Lecture Notes in Computer Science*, pages 313–324, 2011.
- [9] Josef Maier and Kristian Ambrosch. Distortion Compensation for Movement Detection Based on Dense Optical Flow. In *Advances in Visual Computing*, volume 6938 of *Lecture Notes in Computer Science*, pages 168–179, 2011.
- [10] René Vidal, Stefano Soatto, Yi Ma, and Shankar Sastry. Segmentation of Dynamic Scenes from the Multibody Fundamental Matrix. In *ECCV Workshop on Vision and Modeling of Dynamic Scenes*, 2002.
- [11] René Vidal, Stefano Soatto, and Shankar Sastry. Two-View Segmentation of Dynamic Scenes from the Multibody Fundamental Matrix. Technical report, UC Berkeley, UCLA, 2002.
- [12] Jean-Yves Bouguet. Pyramidal Implementation of the Lucas Kanade Feature Tracker: Description of the algorithm. Technical report, Intel Corporation, 1999.
- [13] Reg G. Willson and Steven A. Shafer. What is the Center of the Image? In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 1993.
- [14] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in computer vision*. Cambridge University Press, 2nd edition, 2000.
- [15] Richard Hartley. In Defense of the Eight-Point Algorithm. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 19, June 1997.
- [16] Martin a. Fischler and Robert C. Bolles. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Communications of the ACM*, 24(6):381–395, June 1981.
- [17] David Nistér. Preemptive RANSAC for Live Structure and Motion Estimation. In *Proceedings of the Ninth IEEE International Conference on Computer Vision*, volume 1, pages 199–206, 2003.
- [18] Rahul Raguram, Jan-Michael Frahm, and Marc Pollefeys. A Comparative Analysis of RANSAC Techniques Leading to Adaptive Real-Time Random Sample Consensus. In *Proceedings of the European Conference on Computer Vision*, volume 5303 of *Lecture Notes in Computer Science*, pages 500–513, 2008.
- [19] R. C. Gonzalez and R. E. Woods. *Digital Image Processing, Second Edition*. Pearson Education International, 2002.
- [20] Luc Vincent. Morphological Grayscale Reconstruction in Image Analysis: Applications and Efficient Algorithms. *IEEE Transactions on Image Processing*, 2(2):176–201, April 1993.