Chapter

Object Tracking Using Adapted Optical Flow

Ronaldo Ferreira, Joaquim José de Castro Ferreira and António José Ribeiro Neves

Abstract

The objective of this work is to present an object tracking algorithm developed from the combination of random tree techniques and optical flow adapted in terms of Gaussian curvature. This allows you to define a minimum surface limited by the contour of a two-dimensional image, which must or should not contain a minimum amount of optical flow vector associated with the movement of an object. The random tree will have the purpose of verifying the existence of superfluous vectors of optical flow by discarding them, defining a minimum number of vectors that characterizes the movement of the object. The results obtained were compared with those of the Lucas-Kanade algorithms with and without Gaussian filter, Horn and Schunk and Farneback. The items evaluated were precision and processing time, which made it possible to validate the results, despite the distinct nature between the algorithms. They were like those obtained in Lucas and Kanade with or without Gaussian filter, the Horn and Schunk, and better in relation to Farneback. This work allows analyzing the optical flow over small regions in an optimal way in relation to precision (and computational cost), enabling its application to area, such as cardiology, in the prediction of infarction.

Keywords: Object tracking, vehicle tracking, optical flow, gaussian curvature, random forest

1. Introduction

Object tracking is defined as a problem of estimating the object's trajectory, done by means of a video image. There are several tools for tracking objects and are used in various fields of research, such as computer vision, digital video processing, and autonomous vehicle navigation [1]. With the emergence of high-performance computers, high-resolution cameras, and the growing use of so-called autonomous systems that, in addition to these items, require specialized tracking algorithms, increasingly accurate and robust for automatic video analysis, has currently been the target of numerous research on the development of new object tracking techniques [2, 3].

Object tracking techniques are applicable to motion-based reconnaissance cases [4], automatic surveillance systems [5], pedestrian flow monitoring in crosswalks [6], traffic control [7], and autonomous vehicular navigation [8]. Problems of this type are highly complex due to the characteristics of the object and the environment, generating many variables, which impairs performance and makes the application of tracking algorithms unfeasible to real-world situations. Some approaches seek to resolve this impasse by simplifying the problem, reducing the number of variables [9]. This process, in most cases, does not generate good results [10, 11], making it even more difficult to identify the main attributes to be selected to perform a task [12, 13].

Most of the object tracking problems occur in open environments, so-called uncontrolled [14]. The complexity of these problems has attracted the interest of the scientific community and generated numerous applied research in various fields of research. Current approaches, such as the ones that use convolutional neural networks—CNN, deal well with the high number of variables of these types of problems, providing space–temporal information of the tracked objects, through three-dimensional convolutions [15–17]. This ends up creating an enormous number of learnable parameters, which ends up generating an overfitting [11]. A solution to reduce this number of learnable parameters was combining space–time data, extracted using the optical flow algorithm, used in the Two-Stream technique [18–20]. However, this technique presents good results only for large datasets, showing itself to be inefficient for small datasets [15, 21].

In recent years, research using machine learning has been applied to tracking problems, gaining notoriety due to the excellent results obtained in complex environments and attribute extraction [21–23]. Deep learning stands out among these techniques for presenting excellent results to unsupervised learning problems, [24], object identification [25], semantic segmentation [26]. Random trees are also examples of machine learning techniques, and their excellent results, due to their precision and great capacity to handle a large volume of data and low overfitting tendency [27, 28], and widely used in research areas such as medicine, in the prediction of hereditary diseases [29], agriculture to increase the productivity of a given plantation crop and in astronomy, acting on the improvement of images captured by telescopes, in the spectrum electromagnetic radiation not visible to the human eye [30]. The possibilities of applications, and new trends and research related to machine learning techniques, with particular attention to random trees, allow the development of algorithms that can be combined with existing ones, in the case of optical flow algorithms, (belonging to computational field of view) taken advantage of in this way, the advantages of each [31–33].

Developing an algorithm whose objective is to track objects, using the particular advantages of these techniques in a combined way, justifies creating a tracking algorithm that combines the optical flow technique, adapted in this work in terms of the Gaussian curvature associated with a minimal surface, with a random trees waiting for it to capture on this surface a minimum number of optical flow vectors that characterize the moving object, accurately and with low computational cost, contributing not only in the fields of computational vision but in other branches of science, such as in medicine, it can help in the early identification of infarctions.

2. Related works

Due to the large number of studies related to the technique of object tracking, only a small number surrounding this theme will be addressed. The focus of this project is not to make a thorough study on the state of the art. With this in this item, the main works in the literature, associated with the tracking of objects, will be presented. Among the various approaches used for this context, we highlight those focused on the techniques of optical flow, and others belonging to machine learning, such as those that use identifications of patterns, which allow relating, framing, and justifying the development of this proposal and its importance, through its contribution, to the state of the art.

2.1 Object tracking

Object tracking is defined as a process that allows you to uniquely estimate and associate the movements of objects with consecutive image frames. The objects considered can be from one, the set of pixels belonging to a region of the image. The detection of pixels is done by a motion detector or objects, which allows to locate objects with similar characteristics that move, between consecutive frames.

These characteristics of the object to be tracked are compared with the characteristics of a reference object modeled by a classifier over a limited region of the so-called region of interest frame, where the probability of detection of the object is greater. Thus, according to [33], the detector of traced objects, locate several objects on the different parts of the region of interest and performs the comparison of these objects with the reference object. This process is performed for each frame and each object detected, candidate to be recognized as the greatest possible similarity, to the reference object can be represented, through a set of fixed-size characteristics, extracted from this region containing a set of pixels, which can be represented by a numerical array of data.

Thus, mathematically, the region containing a set of pixels belonging to the regions of the object of interest, where the characteristics that allow to test whether the region of the frame, in which the object to be traced is, is given by:

$$\mathrm{OC_i(t) = OC(t)|\|L(OC(t)) - L(OR(i-1))\| < \varepsilon} \tag{1}$$

where, $L(OC(t))$ is the position (x, y) of the centroid of the candidate object $OC(t)$, $L(OR(i-1))$, is the position of the object traced to the $(i$–$1)$—frame of the video and $0 < \varepsilon \in \mathbb{R}$, is an actual value associated with the size of the region of the object of interest.

According to the works of [34, 35], learning methods are used to adapt the changes of movement and other characteristics such as geometric aspect and appearance of the tracked object. These methods are usually used adaptive tracked object trackers and detectors. The following will be presented other types of object trackers, found in the literature.

According to [36], a classifier can be defined with a f belonging to a family of functions F parameterized by a set of classifier parameters. They form a detector of objects to be tracked which in turn is an integral part of a tracker. A classifier can also be training and thereby generate a set of classification parameters, producing the function f, that allows you to efficiently indicate the classes v_i of the test data x_i from a training set $C_t = \{(x_1, y_1), \dots, (x_n, y_n)\}$. The data is points in *the space of* the characteristics, which can be entropy, the gray level, among others.

The classifier aims to determine the best way to discriminate the data classes, on the space of characteristics. The test data form a set containing the characteristics of the candidate objects, which have not yet been classified. The position of the object to be tracked in the frame is defined as the position corresponding to the highest response of the detector of the object to be tracked on the *ith*-candidate objects. Therefore, the position of the object to be tracked is determined by the position of the *ith*-candidate object, which is most likely to belong to the class of the crawled object, given by the following equation:

$$\mathrm{L(OR(t)) = L\big(argmaxiP\big(yi = COR|OCi(t), PR(t)\big)\big)} \tag{2}$$

$$P(yi = COR|OCi(t), PR(t)) = \begin{cases} \mathrm{1/N, se\|L(OCi\ (t)) - L(OR(t|t-1))\| < \varepsilon} \\ \mathrm{0, otherwise} \end{cases} \tag{3}$$

where the variable *COR* in the equation (3), are the classes of the tracked objects and the candidate objects *OC*, all with equiprobability of occurrence. According to the types of classifiers used object detectors to be tracked, along with the initial detector, it is possible to use some learning technique to train them. One of the ways used is offline training [36] and adjusting the parameters of the classifier before running the tracker.

Offline-trained classifiers are generally employed in object detectors designed to detect all new objects of interest that enter the camera's field of view [37]. The training set C_t, must contain characteristics x_i extracted from the objects to be traced and diverse environmental characteristics. This allows new objects, with varied geometric characteristics and aspects, to be detected more efficiently. As for online training, the adjustment of the parameters of the classified is performed during the tracking process. For online trained classifiers, they are generally used in object detectors to be tracked. Thus, in each frame, the new extracted characteristics are used to adjust the classifiers.

2.1.1 Binary classification

In [38], trackers that use the detection tracking technique deal with object tracking, as a binary classification problem whose goal is to find the best function f that separates the objects to be tracked R, of other objects in the environment. Object tracking seen as a binary classification problem, which is currently one of the subjects that receives the most attention in research in computing vision.

In [39], were developed trackers that used detectors of objects to be tracked, formed by classifiers in committee formed by binary classifiers said weak. For [40], a binary classifier is defined as a classifier, used in problems where the class y_i of a OC_i belongs to the set $Y = \{-1, 1\}$. The negative class $\{-1\}$ refers to the characteristics of the environment and other objects. A positive class {+1} refers to the class of the object to be tracked.

A classifier is said to be weak, when it has a probability of "hitting" a given data class, only slightly higher than a random classifier. The detector of the object to be tracked must separate the crawled object from the other objects and the environment. Its purpose and determine the position of the tracked object, according to the equations (1)–(3). According to [41, 42] each of the *ith* candidate object classes OC_i, it is defined according to Bayesian theory of decision, through minimal classification error. This means that, the decision given by observing, the sign of the difference between $P\big(y_i = COR|OC_i(t), PR(t)\big)$ and $P\big(y_i = CNOR|OC_i(t), PR(t)\big)$, so that the sum of these probabilities is unitary.

2.1.2 Monitoring systems

For [65], the term monitoring system, refers to the process of monitoring and autonomous control, without human intervention. This type of system has the function of detecting, classifying, tracking, analyzing, and interpreting the behavior of objects of interest. In [43, 44], this technique was used combined with statistical techniques for controlling people's access to a specific location. It was also observed the use of intelligent monitoring systems, applied to building, port, or ship security [45, 46].

The functions comprised by a monitoring system are so-called low- and high-level tasks. Among some high-level tasks, we highlight the analysis, interpretation and description of behavior, the recognition of gestures, and the decision between the occurrence or not of a threat. Performing high-level tasks require that for each frame, the system needs to perform low-level tasks, which involve direct manipulation of the image pixels [47–55]. As an example, we highlight the processes of noise elimination, detection of connected components, and obtain information on the location and geometric aspect of the object of interest.

A monitoring system consists of five main components, which are presented in **Figures 9**. Some monitoring systems may not contain all components. The initial detector aims to detect the pixel regions of each frame that have a significant probability of containing an object to be tracked. This detector can be formed by a motion detector that detects all moving objects based on models of objects previously recorded in a database or based on characteristics extracted offline [40, 41]. The information obtained by the initial detector is processed by an image processor], which will have the function of eliminating noise, segmenting, and detecting the connected components.

The regions containing the most relevant pixels are analyzed and then classified as objects of interest by the classifier [49–53]. Objects of interest are modeled and are now called reference objects so that the tracker determines its position frame by frame [54, 55]. The information obtained by the initial detector is processed by an image processor], which will have the function of eliminating noise, segmenting, and detecting the connected components.

A tracker, an integral part of a detector, is defined as a function that allows estimating the position of objects at each consecutive frame, through and defines the region of the object of interest, for each ith object being tracked within a region of interest. This estimation of the movement is performed through the correct association of the captured and tracked objects, to consecutive video frames. The trace often and interpreted as a data binding problem. **Figure 1** shows a schematic of the main components of a monitoring system.

2.2 State of the art in object tracking with optical flow

Several techniques that allow the calculation to have been developed in recent years to calculate the optical flow vector [79]. These methods are grouped according to their main characteristics and the approach used for the calculation of the optical flow. Thus, the differential methods performed in the studies in [55], the methods d and calculation of the optical flow through the frequency domain [45] the phase correlation methods [75], and the method of association between regions [80].

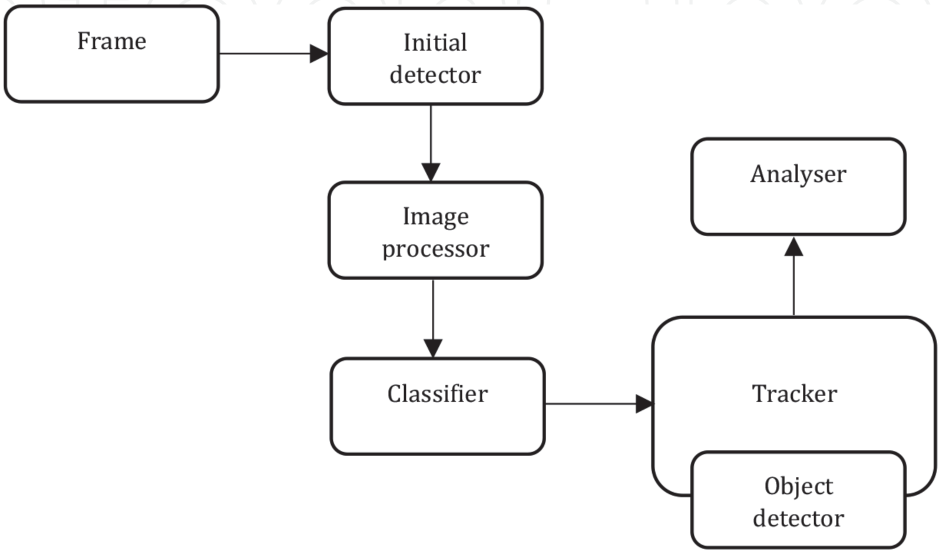

Figure 1.
Main component of a monitoring.

The method proposed in [55], allows the calculation of the optical flow for each point around a neighborhood of pixels. In [56], it is also considered a neighborhood of pixels, but in this case, the calculation of the optical flow is performed geometrically. In the work presented by [57] it is adding of the restrictions of regularization. In [58] turn active compare performance analyses were performed between the various algorithms and optical flow present in the literature.

This technique is considered robust for detaining and tracking moving objects from your images, both those captured by fixed or mobile cameras. This gives this technique, but high computational cost makes most practical applications unfeasible. Thus, to reduce this complexity, techniques of increasing resolutions were adopted in [67]. Also, for the same purpose, we used the techniques of subsampling on some of the pixels belonging to the object of interest to obtain optical flow [51].

Other authors also use a point of interest detector to select the best pixels for tracking and calculate the optical flow on these points [51, 66]. The reduction in the number of points to be tracked is associated with a decrease in computational complexity, so in [51] the points of interest were selected using the FAST algorithm [66].

The method developed by Lucas-Kanade [55], it is a differential method and widely used in the literature and having variations modifications. It allows you to estimate the optical flow for each point (x, y, t) calculating the like transformation $(TA\ (x, t)$, applied to the pixels of a pixel grid, with center in (x, y) by the following function $f(x, t)$, that is:

$$f(x,t) = \min\left(\sum_{x \in (pixel\ grid)} [Q(x,t-1) - (Q(TA(x),t)] * g(x)]\right) \tag{4}$$

where $g(x)$ is a Gaussian smoothing filter centered on x.

New variations of the techniques were being proposed to make the calculation of the optical flow faster and faster. In [59] a tracker was proposed based on the algorithm of [55]. The translation of a point represented by a grid of rectangular sized pixels 25 × 25, was calculated and its validity is evaluated by calculating the SSD[1] in the grid pixels in $Q(t)$ and in $Q(t-1)$. If the SSD is high, the point is dropped and stops being traced.

In [50] objects were detected by subtracting the image from the environment and removed the movement of the camera with the calculation algorithm of the optical flow vector proposed by [55]. In the studies carried out in [60, 61], they showed that the reliability of the estimated optical flow reduced the case of some points of the object of interest whose optical flow cannot be represented by the same matrix given by the related transformation $(TA\ (x, t))$ of the other points. Thus, to improve the robustness of the algorithm of [55, 61] proposed a calculation of the independent optical flow vector for each of the N points belonging to the object of interest selected with the SURF (Speeded Up Robust Features) point detector in the initial frame.

In [61] they also modified Lucas - Kanade's algorithm [55] by inserting the Hessian matrix in the calculation of the value of the variation of the related transformation $(\Delta TA\ (x, t))$. The algorithm allows for more effective tracking when partial occlusions, deformations, and changes in lighting occur, as optical flow is not calculated considering all points of objects of interest.

[1] Actual amount of data rate for actual data recorded.

Already in the proposal presented in [62] was the development of algorithm to detect people in infrared images that combines the information of the value of pixels with a method of motion detection. The algorithm forms a relevant pixel map by applying thresholding segmentation. While the camera is still, an image M is built with the differentiation between frames. If the camera is in motion, M is filled with the pixels obtained by the analysis of the moment of the optical flow calculated by the algorithm of [55]. The map of relevant pixels is replaced by the union between M and the Pixel Map relevant to the first case and the second an interception between M and the pixel map relevant case to compensate for the movement of the camera.

The method for tracking swimmers presented in [45], uses the information of the movement pattern by the optical flow and the appearance of the water that is modeled by a MoG.[2] This allows you to calculate an optical flow vector for each pixel of the video independently of the other, through B which is an array composed of gradients in the directions x and y pixels in a grid of pixels.

In [63], a method was presented that incorporated physical restrictions to the calculation of optical flow. The tracker uses the constraints to extract the moving pixels with a lower failure rate. The calculation can be impaired when occlusions occur or when the environment has low light. The operator defines the physical constraints and selects the points of the OR that are tracked by optical flow. Constraints can be geometric, kinematic, dynamic, of the property of the material that makes up the OR or any other type of restriction.

In [64], the points that are tracked with the optical flow are defined by applying the Canny edge detector on the pixels of the reference pixel map. Pixels that produce a high response to the Canny detector are the selected points.

In [65], optical flow is used as a characteristic for tracking the contour of the object. The contour is shifted in small steps until the position in which the optical flow vectors are homogeneous is found.

In [66], they performed an estimate of the translation and orientation of the reference object by calculating the optical flow of the pixels belonging to its silhouette. The coordinates of the centroid position are defined by minimizing the Hausdorff distance between the mean of the optical flow vectors of the reference object and the candidate object to be chosen as the object of interest.

2.2.1 Optical flow as a function of Gaussian curvature

Optical flow is defined as a dense vector field associated with the movement and apparent velocity of an object, given by the translation of pixels from consecutive frames in an image region. It can be calculated from the brightness restriction, considered constant, from the corresponding pixels in consecutive frames.

Mathematically be a pixel (x, y), associated with a luminous intensity $I(x, y)$, over an image surface or plane, and a time interval and a sequence of frames associated with an apparent offset of the pixel over that image surface or plane. Thus, the rate of variation of light intensity in relation to a time interval, associated with the apparent movement of the pixel, on a surface or plane of the image, being considered practically null can be given by:

$$\frac{dI(x,y)}{dt} = \frac{\partial I(x,y)}{\partial x}\,\frac{d\,x}{dt} + \frac{\partial I(x,y)}{\partial y}\,\frac{d\,y}{dt} + \frac{\partial I(u,v)}{dt} \tag{5}$$

$$= I_x\,u_x + I_y\,v_x + I_t \tag{6}$$

[2] MoG: mixture of Gaussian distributions.

$$\frac{dI(x,y)}{dt} = 0 \Longrightarrow I_x\, u_x + I_y\, v_x + I_t = 0 \tag{7}$$

So that equation (7) is called optical flow restriction and where the terms I_x, I_y, I_t denote the derivatives relative to the brightness intensity relative to the coordinates x, y and time t, and u and v, $(u(x,y), v(x,y))$ are the horizontal and vertical components of a vector representing the optical flow field, for the pixel (x,y) in question.

The number of variables in equation (6) is greater than that of equations, which does not allow estimating components and vector, and determining a single solution for the optical flow restriction equation. With this, Lucas and Kanade proposed a solution to solve this problem. The solution method proposed by them considers the constant flow in a region formed by a set of pixels $N \times N$, so you can write the optical flow restriction equation for each pixel in this region, thus obtaining a systems of equations with 2 variables, that is:

$$\begin{gathered} \mathrm{I_{x1}v_x + I_{y1}v_y + I_{t1} = 0} \\ \mathrm{I_{x2}v_x + I_{y2}v_y + I_{t2} = 0} \\ \vdots \\ \mathrm{I_{xp}v_x + I_{yp}v_y + I_{tp} = 0} \end{gathered} \tag{8}$$

Passing the set of equations given by equation (8) to the matrix form we have:

$$\begin{pmatrix} \mathrm{I_{x1}} & \mathrm{I_{y1}} \\ \vdots & \vdots \\ \mathrm{I_{xp}} & \mathrm{I_{yp}} \end{pmatrix} \begin{pmatrix} \mathrm{v_x} \\ \mathrm{v_y} \end{pmatrix} + \begin{pmatrix} \mathrm{I_{t1}} \\ \vdots \\ \mathrm{I_{tp}} \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix} \tag{9}$$

Using the least squares method, in the system of equations (9) in the form of matricial, the same can be solved. Therefore, the optical flow $v = (v_x, v_y)$ can be estimated for a particular region or window with $N \times N$ pixels, that is:

$$\begin{aligned} &\begin{pmatrix} \mathrm{I_{x1}} & \mathrm{I_{y1}} \\ \vdots & \vdots \\ \mathrm{I_{xp}} & \mathrm{I_{yp}} \end{pmatrix} \begin{pmatrix} \mathrm{I_{x1}} & \mathrm{I_{y1}} \\ \vdots & \vdots \\ \mathrm{I_{xp}} & \mathrm{I_{yp}} \end{pmatrix}^{\mathrm{t}} \begin{pmatrix} \mathrm{v_x} \\ \mathrm{v_y} \end{pmatrix} = - \begin{pmatrix} \mathrm{I_{t1}} \\ \vdots \\ \mathrm{I_{tp}} \end{pmatrix} \begin{pmatrix} \mathrm{I_{x1}} & \mathrm{I_{y1}} \\ \vdots & \vdots \\ \mathrm{I_{xp}} & \mathrm{I_{yp}} \end{pmatrix}^{\mathrm{t}} \\ &\Rightarrow \begin{pmatrix} \mathrm{I_{x1}} & \mathrm{I_{y1}} \\ \vdots & \vdots \\ \mathrm{I_{xp}} & \mathrm{I_{yp}} \end{pmatrix} \begin{pmatrix} \mathrm{I_{x1}} & \mathrm{I_{y1}} \\ \vdots & \vdots \\ \mathrm{I_{xp}} & \mathrm{I_{yp}} \end{pmatrix}^{\mathrm{t}} \left(\begin{pmatrix} \mathrm{I_{x1}} & \mathrm{I_{y1}} \\ \vdots & \vdots \\ \mathrm{I_{xp}} & \mathrm{I_{yp}} \end{pmatrix} \begin{pmatrix} \mathrm{I_{x1}} & \mathrm{I_{y1}} \\ \vdots & \vdots \\ \mathrm{I_{xp}} & \mathrm{I_{yp}} \end{pmatrix}^{\mathrm{t}} \right)^{-1} \begin{pmatrix} \mathrm{v_x} \\ \mathrm{v_y} \end{pmatrix} \\ &= - \begin{pmatrix} I_{t1} \\ \vdots \\ I_{tp} \end{pmatrix} \begin{pmatrix} I_{x1} & I_{y1} \\ \vdots & \vdots \\ I_{xp} & I_{yp} \end{pmatrix}^{t} \left(\begin{pmatrix} I_{x1} & I_{y1} \\ \vdots & \vdots \\ I_{xp} & I_{yp} \end{pmatrix} \begin{pmatrix} I_{x1} & I_{y1} \\ \vdots & \vdots \\ I_{xp} & I_{yp} \end{pmatrix}^{t} \right)^{-1} \end{aligned} \tag{10}$$

Where:

$$\begin{pmatrix} \mathrm{I_{x1}} & \mathrm{I_{y1}} \\ \vdots & \vdots \\ \mathrm{I_{xp}} & \mathrm{I_{yp}} \end{pmatrix} = \mathrm{A_{px2}} = \mathrm{A}$$

Therefore, one has that:

$$\begin{pmatrix} \mathrm{I_{x1}} & \mathrm{I_{y1}} \\ \vdots & \vdots \\ \mathrm{I_{xp}} & \mathrm{I_{yp}} \end{pmatrix} \begin{pmatrix} \mathrm{I_{x1}} & \mathrm{I_{y1}} \\ \vdots & \vdots \\ \mathrm{I_{xp}} & \mathrm{I_{yp}} \end{pmatrix}^{\mathrm{t}} \left(\begin{pmatrix} \mathrm{I_{x1}} & \mathrm{I_{y1}} \\ \vdots & \vdots \\ \mathrm{I_{xp}} & \mathrm{I_{yp}} \end{pmatrix} \begin{pmatrix} \mathrm{I_{x1}} & \mathrm{I_{y1}} \\ \vdots & \vdots \\ \mathrm{I_{xp}} & \mathrm{I_{yp}} \end{pmatrix}^{\mathrm{t}} \right)^{-1} =$$

$$= (A \cdot A^t)(A \cdot A^t)^{-1} = \begin{pmatrix} 1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 1 \end{pmatrix} = Id_{pxp} = Id \qquad (11)$$

Thus:

$$Id = -I_t \cdot A^t (A \cdot A^t)^{-1} \qquad (12)$$

This method has a reduced computational cost to determine optical flow estimation when compared to other methods because it is simple, that is, it is since the region in which the variation of light intensity between pixels is minimal has a size 2×2, contained in a region *N x N*. In this way, the Optical Flow is determined in a region of 2×2, between these two pixels, using only one matrix inversion operation (equation (12)).

To calculate the optical flow over the size region *N x N*, partial derivatives must be calculated in each pixel. However, considering almost null the variation of the intensity of light between pixels, over the region, the small differences in the accumulated intensities of brightness between pixels compromise the accuracy of the Optical Flow in relation to the determination of the actual motion object, that is, it gains in the processing speed and loses precision in the determination of the motion. When deriving equation (5) we have equation (13), that is:

$$\xi_\alpha^2 = a_x \alpha_1 + a_y \alpha_2 + a_3 u^2 + a_4 v^2 + a_5 uv + a_6 u + a_7 v + a_8 \qquad (13)$$

Where the terms $\alpha_x = \frac{\partial v_x}{\partial t}$, $\alpha_y = \frac{\partial v_y}{\partial t}$, are called the components of the acceleration vector, (v_x, v_y) is the components of the velocity vector and the terms $a_1 = I_x = \frac{\partial}{\partial x} I(x,y,t); a_2 = I_y = \frac{\partial}{\partial y} I(x,y,t); a_3 = I_{xx} = \frac{\partial^2}{\partial x^2} I(x,y,t); a_4 = I_{yy} = \frac{\partial^2}{\partial x^2}(x,y,t); a_5 = I_{xy} = \frac{\partial}{\partial xy} I(x,y,t); a_6 = I_{xt} = \frac{\partial}{\partial xt} I(x,y,t);$ $a_7 = I_{yt} = \frac{\partial}{\partial yt} I(x,y,t)$ $a_8 = I_{tt} = \frac{\partial}{\partial t^2} I(x,y,t)$, are the first and second partial derivatives of the $I(x,y,t)$.

In view of the small variations present and accumulated along the vector field associated with the optic flow, which cause an additional error in equation (13), a regularization adjustment was made, given by equation (14):

$$\xi_c^2 = \left(\frac{\partial}{\partial x} v_x(x,y) \right)^2 + \left(\frac{\partial}{\partial y} v_x(x,y) \right)^2 + \left(\frac{\partial}{\partial y} v_y(x,y) \right)^2 + \left(\frac{\partial}{\partial x} v_y(x,y) \right)^2 \qquad (14)$$

Thus, combining equations (13) and (14), the error ξ can be minimize by the equation (15):

$$\iint \left(\xi_{\alpha}^{2} + \alpha^{2}\xi_{c}^{2}\right)\mathrm{d}x\mathrm{d}y \tag{15}$$

where α is the value of the weights required for smoothing the variation of the associated optical flow. So, to get the $v_x = v_x(x, y)$ e $v_y = v_y(x, y)$, thus using the resources of the variational calculation one has:

$$\begin{cases} 2\alpha_3 v_x + \alpha_5 v_y = \alpha^2 \nabla^2 v_x - \mathrm{b}_1 \\ \alpha_5 v_x + 2\alpha_4 v_y = \alpha^2 \nabla^2 v_y - \mathrm{b}_2 \end{cases} \tag{16}$$

where $\nabla^2 v_x$ is the Laplacian of v_x e $\nabla^2 v_y$ is the Laplacian of v_y and the coefficients de b_1, b_2 can be given as:

$$\begin{cases} \mathrm{b}_1 = \dfrac{\partial}{\partial \mathrm{t}}\alpha_1 + \alpha_6 \\ \mathrm{b}_2 = \dfrac{\partial}{\partial \mathrm{t}}\alpha_2 + \alpha_7 \end{cases} \tag{17}$$

and replacing the coefficients $I_x, I_y, I_{xx}, I_{yy}, I_{xy}, I_{xt}, I_{yt}, I_{tt}$ in equation (16), one has:

$$I_{xx} v_x + I_{xy} v_y = \left(\frac{\alpha^2}{3}\right) \nabla 2 v_y - I_{xt} \tag{18}$$

$$I_{xy} v_x + 2I_{xy} v_y = \left(\frac{\alpha^2}{3}\right) \nabla 2 v_x - I_{yt} \tag{19}$$

whereas $\nabla^2 v_x = \overline{v_{x_1 j_k}} - v_{x_1 j_k}$, e $\nabla^2 \mathrm{v_y} = \overline{v_{y_1 j_k}} - v\ {}_{j_1 j_k}$, are the Laplacians of equations (18) and (19), given in their discretized digital forms together with equation (20),

$$\lambda = \left(\frac{\alpha^2}{3}\right) \tag{20}$$

It is possible to reduce the data system by (17), such as:

$$\left[\lambda^2\left(I_{xx} + I_{yy} + \lambda^2\right) + \kappa\right] v_x = \lambda^2\left(I_{yy} + \lambda^2\right) \overline{v_x} - \lambda^2 I_{xy} + \overline{v_x} + c_1 \tag{21}$$

$$\left[\lambda^2\left(I_{xx} + I_{yy} + \lambda^2\right) + \kappa\right] v_x = -\lambda^2\left(I_{yy}\overline{v_x} + \lambda^2\left(I_{yy} + \lambda^2\right)\overline{v_y} + c_2 \tag{22}$$

where the term $\kappa = I_x I_{yy} - {I_{xy}}^2$, it is called *Gaussian curvature* of the surface. And it is also that:

$$\begin{cases} c_1 = I_{xy} I_{yt} - I_{xx}\left(I_{yy} + \lambda^2\right) \\ c_2 = I_{xx} I_{yy} - I_{xx}\left(I_{xx} + \lambda^2\right) \end{cases} \tag{23}$$

Where c_1, c_2 they're real constants.

Therefore, isolating terms v_x, v_y and still replacing c_1 e c_2 in equations (21) and (22) respectively, resulting in equations (24) and (25):

$$\mathrm{v_x} = \overline{\mathrm{v_x}} - \left[\frac{\lambda^2\left(\mathrm{I_{yy}} + \lambda^2\right)\overline{\mathrm{v_x}} - \lambda^2 \mathrm{I_{xy}} + \overline{\mathrm{v_x}} + \mathrm{c_1}}{\left[\lambda^2\left(\mathrm{I_{xx}} + \mathrm{I_{yy}} + \lambda^2\right) + \kappa\right]\mathrm{v_x}}\right] \tag{24}$$

$$\mathrm{v_y} = \overline{\mathrm{v_y}} - \left[\frac{-\lambda^2\left(\mathrm{I_{yy}}\overline{\mathrm{v_x}} + \lambda^2\left(\mathrm{I_{yy}} + \lambda^2\right)\overline{\mathrm{v_y}} + \mathrm{c_2}}{\left[\lambda^2\left(\mathrm{I_{xx}} + \mathrm{I_{yy}} + \lambda^2\right) + \kappa\right]\mathrm{v_x}}\right] \tag{25}$$

The Algorithm 1 is a pseudocode to generate the proposed optical flow vector, through equations (24) and (25) and that allow estimating the speed and position of an object, through a sequence of video images.

Algorithm 1. Adapted optical flow (Gaussian curvature κ).

```
Begin
Input: Image sequence (video)
Output: Vector optic flow generator ($v_x$, $v_y$)
        For I = 1...N do

                Convert images to a gray tone
                Calculate the partial derivatives of 1°and 2°orders of $I(x, y, t)$
                Calculate constants $a_x$, $a_y$, $a_1, \dots, a_8$, $b_1, b_2, \lambda, c_1, c_2$
                Calculate the discretized Laplacians of $\nabla^2 v_x$, $\nabla^2 v_y$
                Calculating Gaussian curvature $\kappa$
                Calculate flow components ($u$, $v$)

        End For

End
```

2.2.2 Random forests

Developed by Breiman [67] in the mid-2000s, and later revised in [68] random trees are considered one of the best-supervised learning methods used in data prediction and classification. Due to its simplicity, low computational cost, great potential to deal with a large volume of data, and still present great accuracy of results, currently this method has become very popular being applied in various fields of science as data science [69]. Bioinformatics, Ecology, in real-life systems and recognition of 3D objects. In recent years, several studies have been conducted with the objective of making the technique more elaborate and seeking new practical applications [70–72].

Many studies were carried out with the aim of narrowing the existing gap between theory and practice can be seen in [73–76]. Among the main components of random tree forests, one can highlight the bagging method [67], and the criterion of classification and regression called cart-*split* [77], which play critical roles.

Bagging (a bootstrap-aggregating contraction) is an aggregation scheme, which generates samples through the bootstrap method, from the original dataset. These methods are nonparametric and belong to the Monte Carlos method class [78], treating the sample as a finite population. Still, these methods are used when the distribution of the target population is not specified, and the sample is the only information available. How in this way a predictor of each sample is constructed, so that the decision is made through an average, and is more effective computational procedures to improve the indexable estimates, especially for large sets of high-dimensional data, where finding a good model in one step is impossible due to the complexity and scale of the problem. As for the cart-split criterion, it originates from the CART program [67], and is used in the construction of individual trees to choose the best cuts perpendicular to the Axes. However, while bagging and the CART division scheme are key elements in the random forest, both are difficult to mathematically analyze and are a very promising field for both theoretical and practical research.

In general, the set of trees is organized in the form of {T_1 (Θ_1), T_2 (Θ_2)… T_i (Θ_i}, where T_B is every tree and Θ_B are bootstrap samples with spare dimensions q x *mtry*, where *mtry* is equal to the number of variables that will be used on each node during the construction of each tree and q is approximately $0, 67 \times (n)$. Each of the trees produces a response $y_{1,i}$ for each of the samples W {$T_1(W) = y_{1,i}$, $T_2(W) = y_{2,i}$, ‥, $T_i(W) = y_{2,B}$} and the mean (regression) or majority vote (classification) of the tree responses will be the final response of the model for each of the samples.

3. Methodology

The methodology employed consisted of combining the optical flow algorithm in terms of Gaussian curvature, developed in this work together with the technique of random forest. The language used for the development of this algorithm was the MATLAB programming language, executed on a 64-bit 8th generation notebook, CORE i7. The input data is a video extension Avi, lasting 5 min of a vehicle and two cyclists, circulating in the vicinity of the beach of Costa Nova, in the locality of Ilhavo, in Aveiro, Portugal. The video was fragmented into a set of frames, analyzed two by two by the algorithm for the generation of the vector field of optical flow. After that, the resulting image associated with the flow and a minimal surface region, given by the Gaussian curvature. Next on this surface, the random trees analyzed which vectors presented important characteristics to characterize in an "optimal" way, the movement of the object. (see **Figure 2**)

After finishing the process of analysis of the movement of the objects, the execution times and accuracy of the results obtained by the proposed algorithm were compared in relation to the algorithms of Lucas Kanade, Horn and Shunck, Farneback and Lucas Kanade with or without Gaussian filter, allowing to validate the results obtained. After that, the implementation of the developed algorimo began.

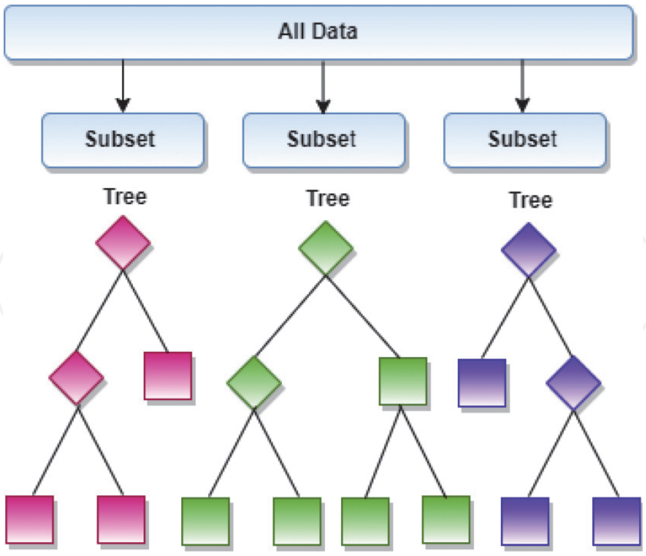

Figure 2.
Representative model of operation of a random forest.

4. Results

Figure 3 shows the vehicle and the two cyclists that were used to collect the image to which the results proposed in this work were obtained so that the choice was random on the right side. A graphical representation of the vector field of

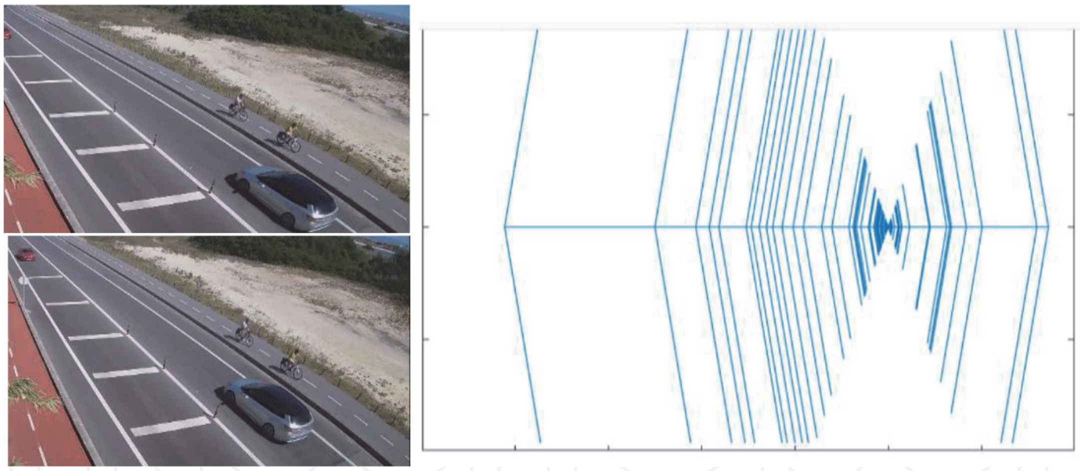

Figure 3.
(a) Left side: vehicle shift between moments (t − 1) and t. (b) Right side: representation of the corresponding optical flow.

optical flow generated by the sequence of two consecutive frames, over 5 minutes of video is shown.

On the right side of **Figure 3**, the optical flow associated with the movement of the vehicle between the time intervals from $(t - 1)$ to t is being represented. Note that the vector representation of this field associated with this flow was performed in such a way that the vectors generated by the field were superimposed in the horizontal direction of the central axis of the figure. Although there were other objects present at the site, that is, two cyclists and a car in the upper left corner, the object of interest considered was the vehicle close to the cyclists. This is shown on the right side of **Figure 3**, by the layout of this horizontal arrangement of vectors, which allows indicating whether the current movement and the predicted movement of the considered object is to the left or to the right.

The region with the highest horizontal vector density in **Figure 3** is located on the left side, in blue. It is also observed that the number of vectors in this region, despite being spaced, starting from the center to the left, is greater in relation to the number of vectors on the right side. It is also possible, through it, to visually evaluate the movement behavior of the considered objects. This region, containing a higher vector density, corresponds to the current direction in which the object is heading and its predicted displacement. It is also possible to observe that this vector density increases towards the left side, passing through the central part, coming from the right, clearly indicating the direction of movement of the object, that is, the object moves to the left. In **Figure 4**, this process can be understood more clearly.

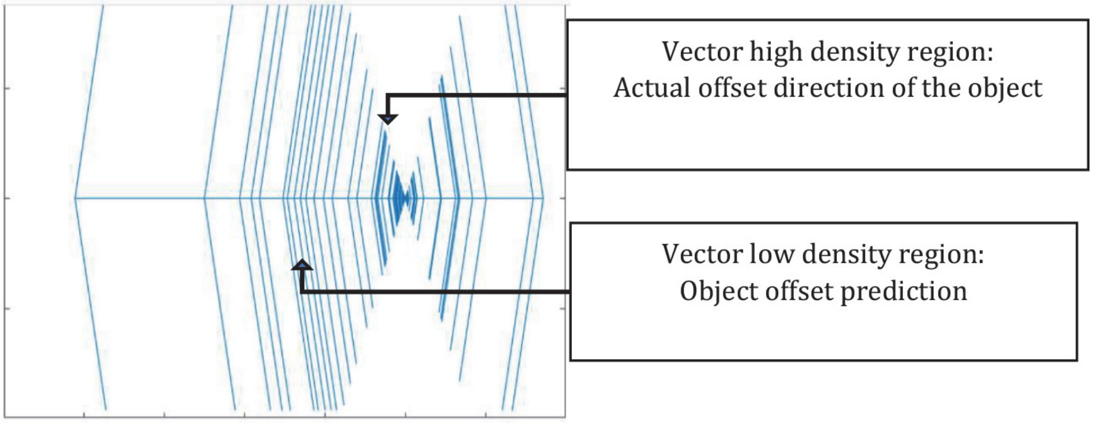

Figure 4.
Prediction and actual displacement of the object obtained through the optical flow.

In a similar way to the one mentioned in **Figure 3**, on the right side of **Figure 5**, the optical flow generated by the displacement of the moving vehicle is represented, between the instants t to $(t + 1)$. It is possible to observe that the vector representation of this field was performed in such a way that its vectors representing this field were also superimposed on the axis in the horizontal direction of the figure, creating a vector density created by this superposition. The object of interest considered remains the vehicle close to the cyclists. As can be seen on the right side of **Figure 3**, the arrangement of the horizontal vectors also allows to indicate the current movement and, if its movement prediction is to the left or to the right.

It is possible to observe a small increase in the vector density to the left, but that has a great influence on the determination of the real and predicted position of the object in the considered time intervals. The Object continues with its actual movement to the left, as well as the predicted movement of the object to the left. However, he showed a slight movement to the left (direction where the cyclists are).

In **Figure 6**, a small variation of the optical flow is observed again in the associated movement between the instants $(t + 1)$ to $(t + 2)$. In this figure, the vehicle is next to the cyclists, both in the opposite direction to the vehicle. The movement of the vehicle continues without great variation in the direction, causing no period for cyclists or other vehicles in the opposite direction to it on the left side.

In **Figure 7**, there was no optical flow variation in the associated movement between the time intervals $(t + 2)$ to $(t + 3)$. In this figure, the vehicle can be seen as it passes the two cyclists. This means that the non-significant variation in the optical

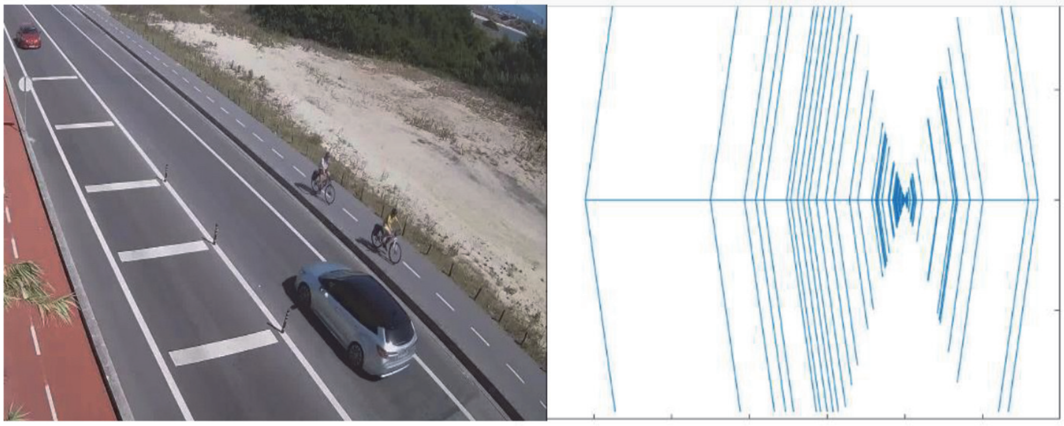

Figure 5.
Object remains on the right side, but with a medium offset to the right and displacement estimate still to the left.

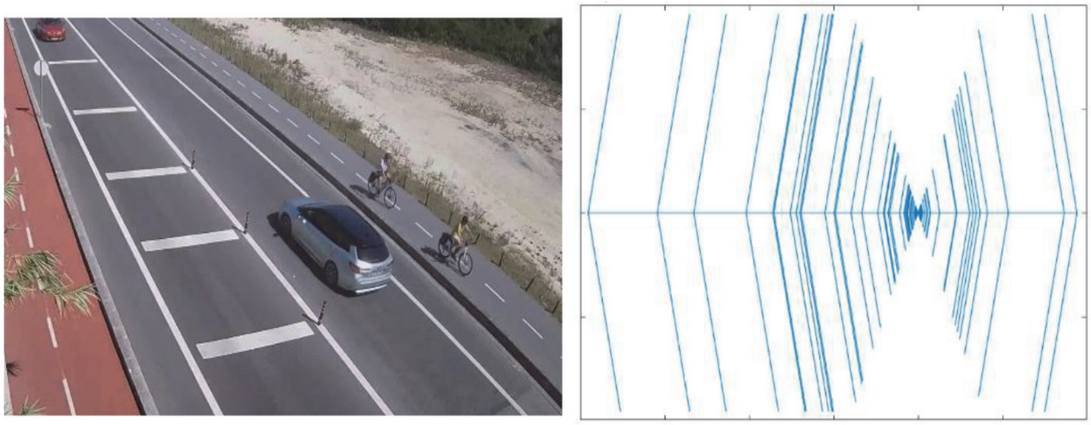

Figure 6.
Object remains on the right side, but with a slight shift to the right and offset estimate to the left.

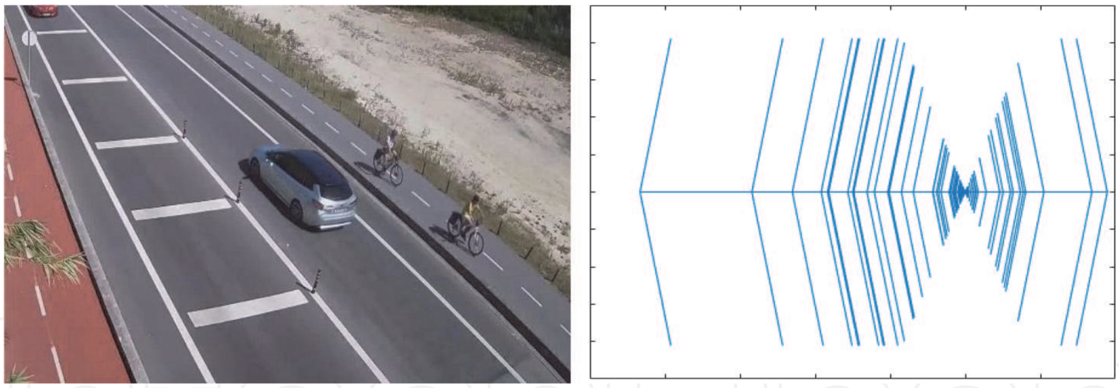

Figure 7.
Object moving and keeping on the left for consecutive frames.

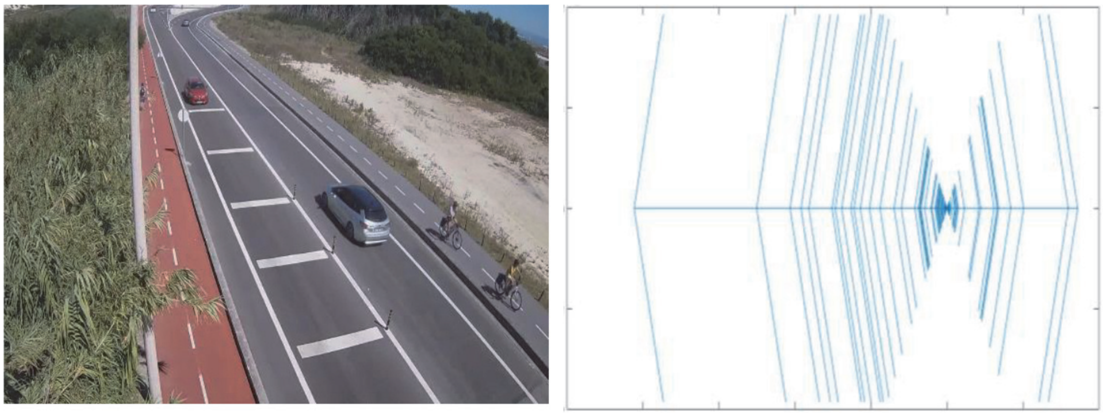

Figure 8.
Object with unchanged offset pattern.

flow vector field, keeping the number of vectors higher on the left side is associated with maintenance in the direction of movement of the object considered, that is, it continues to move on the left side.

In **Figure 8**, the vehicle can be seen completely overtaking the two cyclists and approaching another vehicle in the opposite direction in the upper part of the image (left). The variation of the optical flow vector field remains the same. This indicates that the vehicle continues its trajectory, on the left side to the cyclists, however without posing a danger of collision for the other vehicle in the opposite direction.

5. Analysis and discussing the results

This item will show how the performance evaluation of the proposed algorithm and accuracy was performed in relation to the Algorithms of Luca and Kanade, with or without Gaussian filter, Horn and Schunck, and Farneback.

The algorithm allowed to show on the display in real-time the displacement of the object on the right side and the set of vectors capable of representing the movement of the real-time or accumulated indicating the tendency, in this case, of the direction that the object should perform. This process was carried out in a similar way, using the other algorithms to make it possible to compare them. The behavior of the proposed algorithm and the other will be graphically shown.

The technique developed in this work allowed to generate an optical flow considering important geometric properties allowing to identify similar categories of moving objects and same characteristics. These geometric properties are intrinsically associated with the curvature of the object's surface in three-dimensional space, called Gaussian curvature, in this case in a 2D image.

The modified optical flow, considering these properties, generated a dense optical flow, allowing the generation of a band, describing a track on the 2D plane. This allowed tracking the movement of the considered object. In the same **Figure 8**, it is possible to observe that at each time interval in which the object was monitored, the dispositions of the vectors for the left and right sides, as shown in **Figures 3–7** were responsible for drawing the track associated with the displaced and that allowed tracking the object as it moves.

Figure 9 shows the vehicle that, when moving, generated the optical flow. In **Figures 10** and **11**, the variations of the optical flow between two-time intervals, Δt_i and Δt_n, $(i < n)$ are shown. In this way, the algorithm allowed tracking the progressive movement of the object (movement adopted as progressive, in this work) and, as this happens, it is possible to predict in which direction it is moving, that is, to the left or to the right (or keeping straight) line.

5.1 Optical flow algorithm in terms of the curvature

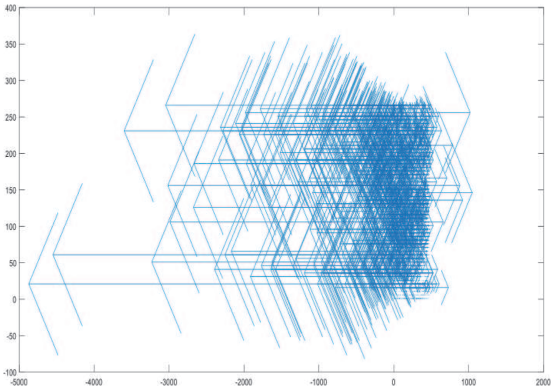

Figure 9.
Variation of the optical flow of the moving object.

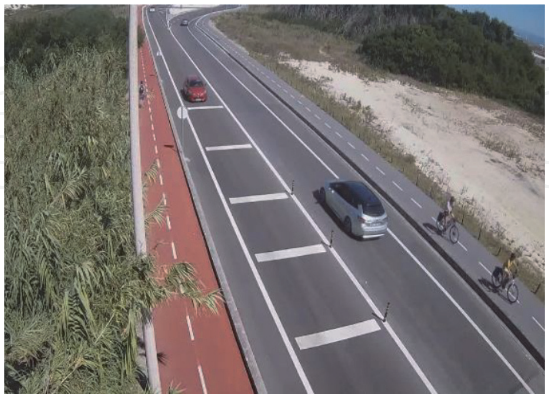

Figure 10.
Vehicle movement.

In the following items, the implementations of the Lucas and Kanade algorithms without or with a Gaussian filter, Horn and Schunck, and Farneback will be shown, using as input data the same sequence of video images used in the algorithm developed in this work. For each, the performance and accuracy obtained will be verified.

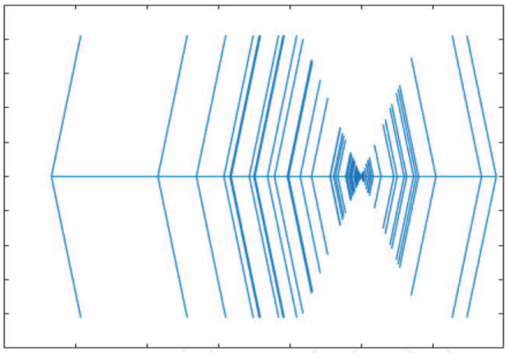

Figure 11.
Object moving to the left side.

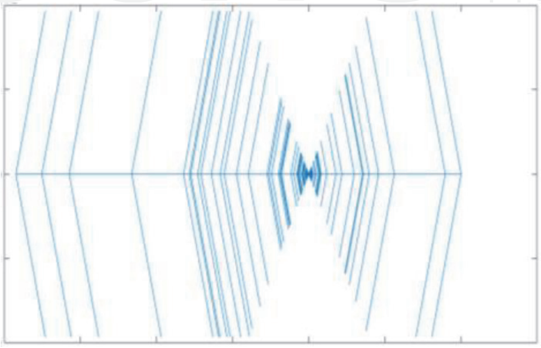

Figure 12.
Object moving to the right side.

5.2 Lucas and Kanade algorithm without Gaussian filter

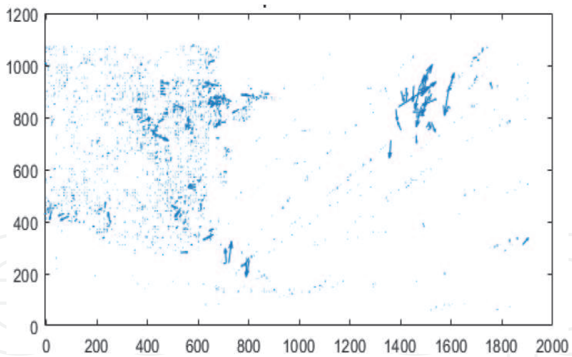

Figure 13.
Variation of the optical flow of the moving object.

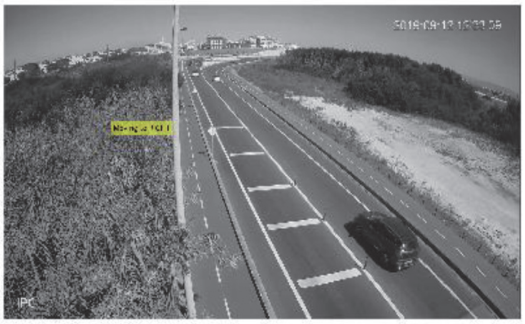

Figure 14.
Vehicle movement.

Figure 15.
Object moving to the left side.

Figure 16.
Object moving to the right side.

5.3 Lucas and Kanade Algorithm with Gaussian filter

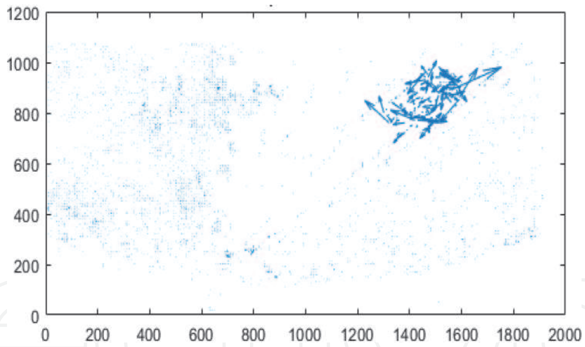

Figure 17.
Variation of the optical flow of the moving object.

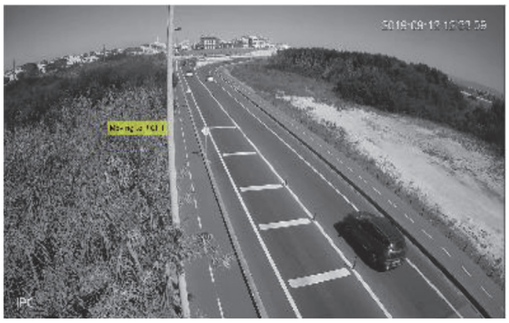

Figure 18.
Vehicle movement.

Figure 19.
Object moving to the left side.

Figure 20.
Object moving to the right side.

5.4 Algoritmo de Horn and Schunck

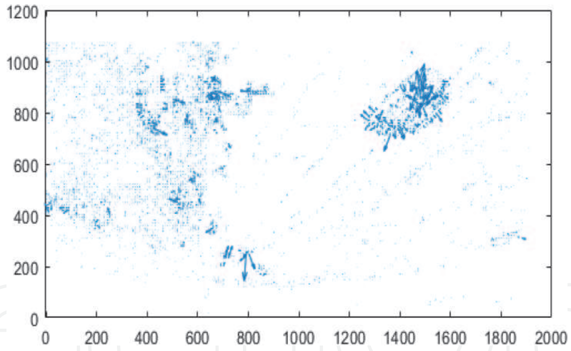

Figure 21.
Variation of the optical flow of the moving object.

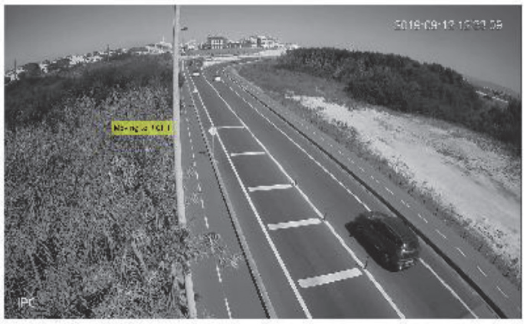

Figure 22.
Vehicle movement.

Figure 23.
Object moving to the right side.

Figure 24.
Object moving to the left side.

5.5 Algoritmo de Farneback

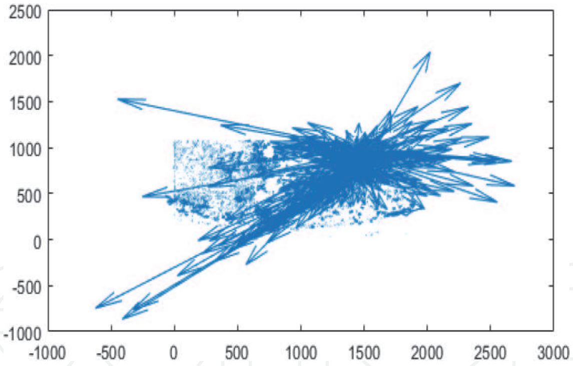

Figure 25.
Variation of the optical flow of the moving object.

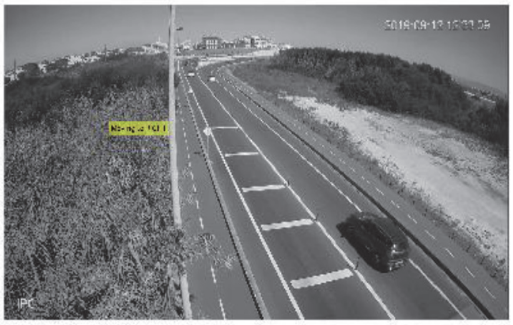

Figure 26.
Vehicle movement.

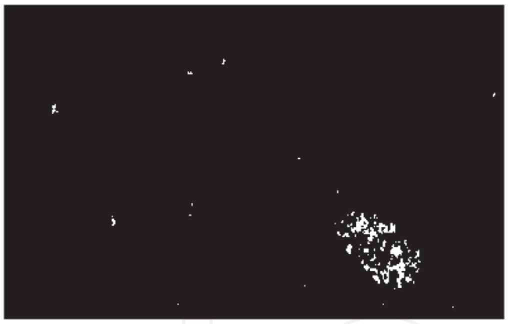

Figure 27.
Object moving to the left side.

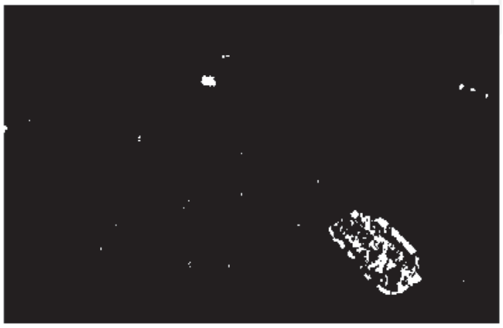

Figure 28.
Object moving to the left side.

For each of the 5 algorithms, 1 frame is shown containing 4 figures, with 2 upper and 2 lower. In each frame, the figure at the top left shows the variation of the vector field between two frames. The right frame, on the other hand, corresponds to the variation of the object's movement in real-time. The lower ones, except for the proposed algorithm, correspond to the number of points on the right or the left, and with this, the movement will occur to the side that has the greatest number of points. In the case of the proposed algorithm, the process will take place through the analysis of vector density. So, to the side where there is greater vector density, this is the side to which the movement will be occurring (see **Figures 12–20**).

Comparing the results presented by the algorithms, it is observed that in the developed model, it was possible to see a dense vector trail of the object, with a slight tendency of displacement to the left, as it continues its movement. In the other models, this was not possible, and it is necessary to resort to a score of points, in the lower table. This process is also possible in the proposed model, but not necessary, which means a reduction in computational cost (see **Figures 21–28**).

Comparing the results, it is observed that the Farneback algorithm also presents high vector density. But the proposed model, as previously said, presents a well-defined vector trail which suggests the non-use of the point count in the lower frame, which does not occur for the Farneback algorithm, indicating higher computational cost, which can affect the accuracy of this algorithm when compared to the proposed algorithm.

Comparing the Horn and Schunck algorithm, a low vector density is observed when compared to the proposed algorithm, which indicates lower accuracy when compared to the proposed algorithm.

Although the two techniques of Lucas and Kanade, are faster applications, indicating low computational cost when compared to the proposed algorithm, the factor of low vector density results in low precision in relation to the proposed method.

6. Final considerations

The proposed method presented good results, showing to be accurate and reasonable speed. This allows this application to be used in critical problems, i.e., to real-world problems. However, it presented limitations that could be verified when compared to the model with Lucas and Kanade, with a Gaussian filter, which is faster and presents good accuracy.

The proposed Method reached only approximately 50% execution speed in relation to the Lucas and Kanade Method, which motivates further improvements to the Method. The technique presented can be applied to other fields of research as in cardiology due to presenting great precision when submitted to small region, which is important because it can be applied with the objective of predicting infarctions and as a current contribution, for the state of the art is to characterize the optical flow in terms of Gaussian curvature, that makes it possible to highlight fields of research such as computational vision and differential geometry.

Acknowledgements

The authors of this work would like to thank the Institute of Electronics and Informatics Engineering of Aveiro, the Telecommunications Institute of Aveiro, and the University of Aveiro for the financial, technical-administrative, and structural support provided that allowed the accomplishment of this work.

Author details

Ronaldo Ferreira[1]*, Joaquim José de Castro Ferreira[1] and
António José Ribeiro Neves[2]

1 University of Aveiro/Telecommunications Institute, Aveiro, Portugal

2 University of Aveiro/Institute of Electronics and Informatics Engineering of Aveiro, Portugal

*Address all correspondence to: ronaldoferreira@ua.pt; an@ua.pt

References

[1] Gonzalez RC, Woods RE. Digital Image Processing. 2002

[2] Abbass MY, Kwon KC, Kim N, Abdelwahab SA, El-Samie FEA, Khalaf AA. A survey on online learning for visual tracking. The Visual Computer. 2020:1-22

[3] Khalid M, Penard L, Memin E. Application of optical flow for river velocimetry. International Geoscience and Remote Sensing Symposium. 2017: 6265-6246

[4] Kastrinaki V, Zervakis M. A survey of video processing techniques for traffic applications. Image and Vision Computing. 2003;**21**(4):359-381

[5] Almodfer R, Xiong S, Fang Z, Kong X, Zheng S. Quantitative analysis of lane-based pedestrian-vehicle conflict at a non-signalized marked crosswalk. Transportation Research Part F: Traffic Psychology and Behaviour. 2016;**42**: 468-468

[6] Tian B, Yao Q, Gu Y, Wang K, Li Y. Video processing techniques for traffic flow monitoring: A survey. In: ITSC. IEEE; 2011

[7] Laurense VA, Goh JY, Gerdes JC. Path-tracking for autonomous vehicles at the limit of friction. In: ACC. IEEE; 2017. p. 56665591

[8] Yilmaz A, Javed O, Shah M. Object tracking: A survey. ACM Computing Surveys. 2006;**38**(2006):13

[9] Veenman C, Reinders M, Ebacker E. Resolving motion matching for densely moving points. IEEE Transactions on Pattern Analysis and Machine Intelligence. 2001;**23**(1):54-72

[10] Goodfellow I, Bengio Y, Courville A, Bengio Y. Deep Learning. Vol. 1. Massachusetts, USA: MIT Press; 2016

[11] Santos Junior JMD. Analisando a viabilidade de deep learning para reconhecimento de ações em datasets pequenos. 2018

[12] Kelleher JD. Deep Learning. MIT Press; 2019

[13] Xiong Q, Zhang J, Wang P, Liu D, Gao RX. Transferable two-stream convolutional neural network for human action recognition. Journal of Manufacturing Systems. 2020;**56**: 605-614

[14] Khan MA, Sharif M, Akram T, Raza M, Saba T, Rehman A. Hand-crafted and deep convolutional neural network features fusion and selection strategy: An application to intelligent human action recognition. Applied Soft Computing. 2020;**87**(73):74986

[15] Abdelbaky A, Aly S. Human action recognition using three orthogonal with unsupervised deep convolutional neural network. Multimedia Tools and Applications. 2021;**80**(13):20019-20065

[16] Rani SS, Naidu GA, Shree VU. Kinematic joint descriptor and depth motion descriptor with convolutional neural networks for human action recognition. Materials Today: Proceedings. 2021;**37**:3164-3173

[17] Farnebäck G. Two-frame motion estimation based on polynomial expansion. In: Proceedings of the Scandinavian Conference on Image Analysis (SCIA). 2003. pp. 363-370

[18] Wang Z, Xia C, Lee J. Group behavior tracking of Daphnia magna based on motion estimation and appearance models. Ecological Informatics. 2021;**61**:7278

[19] Lin W, Hasenstab K, Cunha GM, Schwartzman A. Comparison of handcrafted features and convolutional

neural networks for liver MR image adequacy assessment. Scientific Reports. 2020;**10**(1):1-11

[20] Xu Y, Zhou X, Chen S, Li F. Deep learning for multiple object tracking: A survey. IET Computer Vision. 2019; **13**(4):355-368

[21] Pal SK, Pramanik A, Maiti J, Mitra P. Deep learning in multi-object detection and tracking: State of the art. Applied Intelligence. 2021:1-30

[22] Jiao L, Zhang F, Liu F, Yang S, Li L, Feng Z, et al. A survey of deep learning-based object detection. IEEE Access. 2019;**7**:51837-51868

[23] Pal SK, Bhoumik D, Chakraborty DB. Granulated deep learning and z-numbers in motion detection and object recognition. Neural Computing Applied. 2020;**32**(21):16533-16555

[24] Chung D, Tahboub K, Delp EJ. A two stream siamese convolutional neural network for person re-identification. In: Proceedings of the IEEE International Conference on Computer Vision. 2017. pp. 1983-1671

[25] Choi H, Park S. A survey of machine learning-based system performance optimization techniques. Applied Sciences. 2021;**11**(7):3235

[26] Abdulkareem NM, Abdulazeez AM. Machine learning classification based on Radom Forest algorithm: A review. International Journal of Science and Business. 2021;**5**(2):51-142

[27] Iwendi C, Jo O. COVID-19 patient health prediction using boosted random Forest algorithm. Frontiers in Public Health. 2020;**8**:9

[28] Dolejš M. Generating a spatial coverage plan for the emergency medical service on a regional scale: Empirical versus random forest modelling approach. Journal of Transport Geography. 2020:10 Available from: https://link.springer.com/book/10.687/978-981-15-0637-6

[29] Reis I, Baron D, Shahaf S. Probabilistic random forest: A machine learning algorithm for Noisy data sets. The Astronomical Journal. 2018;**157**(1): 16. DOI: 10.38/1538-3881/aaf69

[30] Thomas B, Thronson H, Buonomo A, Barbier L. Determining research priorities for astronomy using machine learning. Research Notes of the AAS. 2022;**6**(1):11

[31] Yoo S, Kim S, Kim S, Kang BB. AI-HydRa: Advanced hybrid approach using random forest and deep learning for malware classification. Information Sciences. 2021;**546**:420-655

[32] Liu C, Gu Z, Wang J. A hybrid intrusion detection system based on scalable K-means+ random Forest and deep learning. IEEE Access. 2021;**9**: 75729-75740

[33] Paschos G. Perceptually uniform color spaces for color texture analysis: An empirical evaluation. IEEE Transactions on Image Processing. 2001;**10**:932-937

[34] Estrada FJ, Jepson AD. Benchmarking image segmentation algorithms. International Journal of Computer Vision. 2009;**56**(2):167-181

[35] Jaiswal JK, Samikannu R. Application of random forest algorithm on feature subset selection and classification and regression. In: 2017 World Congress on Computing and Communication Technologies (WCCCT). IEEE; 2017. pp. 65-68

[36] Menezes R, Evsukoff A, González MC, editors. Complex Networks. Springer; 2013

[37] Jeong C, Yang HS, Moon K. A novel approach for detecting the horizon

using a convolutional neural network and multi-scale edge detection. Multidimensional Systems and Signal Processing. 2019;**30**(3):1187-1654

[38] Liu YJ, Tong SC, Wang W. Adaptive fuzzy output tracking control for a class of uncertain nonlinear systems. Fuzzy Sets and Systems. 2009;**160**(19): 2727-2754

[39] Beckmann M, Ebecken NF, De Lima BSP. A KNN undersampling approach for data balancing. Journal of Intelligent Learning Systems and Applications. 2015;**7**(04):72

[40] Yoriyaz H. Monte Carlo method: Principles and applications in medical physics. Revista Brasileira de Física Médica. 2009;**3**(1):141-149

[41] Wang X. Intelligent multi-camera video surveillance: A review. Pattern Recognition Letters. 2013;**34**(1):3-19

[42] Wu J, Rehg JM. CENTRIST: A visual descriptor for scene characterization. IEEE Transactions on Pattern Analysis and Machine Intelligence. 2011;**33**(8): 1559-1501

[43] Siegelman N, Frost R. Statistical learning as an individual ability: Theoretical perspectives and empirical evidence. Journal of Memory and Language. 2015;**81**(73):74-65

[44] Kim IS, Choi HS, Yi KM, Choi JY, Kong SG. Intelligent visual surveillance —A survey. International Journal of Control, Automation, and Systems. 2010;**8**(5):926-939

[45] Chan KL. Detection of swimmer using dense optical flow motion map and intensity information. Machine Vision and Applications. 2013;**24**(1): 75-69

[46] Szpak ZL, Tapamo JR. Maritime surveillance: Tracking ships inside a dynamic background using a fast level-set. Expert System with Applications. 2011;**38**(6):6669-6680

[47] Fefilatyev S, Goldgof D, Shceve M, et al. Detection and tracking of ships in open sea with rapidly moving buoy-mounted camera system. Ocean-Engineering. 2012;**54**(1):1-12

[48] Frost D, Tapamo J-R. Detection and tracking of moving objects in a maritime environment with level-set with shape priors. EURASIP Journal on Image and Video Processing. 2013;**1**(42):1-16

[49] Collins RT, Lipton AJ, Kanade T, et al. A System for Video Surveillance and Monitoring. Technical Report. Pittsburg: Carnegie Mellon University; 2000

[50] Viola P, Jones MJ. Robust real-time face detection. International Journal of Computer Vision. 2004;**57**(2):63-154

[51] Rodrigues-Canosa GR, Thomas S, Cerro J, et al. Real-time method to detect and track moving objects (DATMO) from unmanned aerial vehicles (UAVs) using a single camera. Remote Sensing. 2012;**4**(4):770-341

[52] Frakes D, Zwart C, Singhose W. Extracting moving data from video optical flow with Fhysically-based constraints. International Journal of Control, Automation and Systems. 2013; **11**(1):55-57

[53] Sun K. Robust detection and tracking of long-range target in a compound framework. Journal of Multimedia. 2013;**8**(2):98 73, 74

[54] Kravchenko P, Oleshchenko E. Mechanisms of functional properties formation of traffic safety systems. Transportation Research Procedia. 2017; **20**:367-372

[55] Lucas BD, Kanade., T. An iterative image registration technique with an application to stereo vision. In:

International Joint Conference on Artificial Intelligence. 1981

[56] Horn BAND, Schunk B. Determining optical flow. Artificial Intelligence. 1981;**17**:156

[57] Dalal N, Triggs B. Histograms of oriented gradients for human detection. In: 2005 EEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05). Vol. 1. IEEE; 2005. pp. 886-893

[58] Rosten E, Drummond T. Fusing points and lines for high performance tracking. In: 10th IEEE International Conference on Computer Vision. Vol. 2. Beijing, China; 2005. pp. 1508-1515

[59] Shi J, Tomasi C. Good features to track. In: 9th IEEE Conference on Computer Vision and Pattern Recognition. Seattle WA, USA; 1674. pp. 593-600

[60] Cucchiara R, Prati A, Vezzani R. Advanced video surveillance with pan tilt zoom cameras. In: Proceedings of the 6th IEEE International Workshop on Visual Surveillance. Graz, Austria; 2006

[61] Li J, Wang Y, Wang Y. Visual tracking and learning using speeded up robust features. Pattern Recognition Letters. 2012;**33**(16):2094-2269

[62] Fernandez-Caballero A, Castillo JC, Martinez-Cantos J, et al. Optical flow or image subtraction in human detection from infrared camera on Mobile robot. Robotics and Autonomous Systems. 2010;**66**(12):503-511

[63] Frakes D, Zwart C, Singhose W. Extracting moving data from video optical flow with physically-based constraints. International Journal of Control, Automation and Systems. 2013; **11**(1):55-57

[64] Revathi R, Hemalatha M. Certain approach of object tracking using optical flow techniques. International Journal of Computer Applications. 2012;**53**(8):50-57

[65] Cremers D, Schnorr C. Statistical shape knowledge in variational motion segmentation. Israel Network Capital Journal. 2003;**21**:77-86

[66] Li L, Leung MK. Integrating Intensity and Texture Differences for Robust Change. 2002

[67] Smolka B, Venetsanopoulos AN. Noise reduction and edge detection in color images. In: Color Image Processing. CRC Press. 2018. pp. 95-122

[68] Breiman L. Consistency for a Simple Model of Random Forests. 2004

[69] Biau G, Devroye L, Lugosi G. Consistency of random forests and other averaging classifiers. Journal of Machine Learning Research. 2008;**9**(9)

[70] Meinshausen N, Ridgeway G. Quantile regression forests. Journal of Machin Learning Research. 2006;**7**(6)

[71] Ishwaran H, Kogalur UB. Consistency of random survival forests. Statistics & Improbability Letters. 2010; **80**(13–14):746-744

[72] Biau G. Analysis of a random forests model. The Journal of Machine Learning Research. 2012;**13**(1):743-775

[73] Genuer R. Variance reduction in purely random forests. Journal of Nonparametric Statistics. 2012;**24**(3): 565-562

[74] Wager S. Asymptotic theory for random forests. arXiv preprint arXiv: 1405.0352. 2014

[75] Hooker G, Mentch L. Bootstrap bias corrections for ensemble methods. arXiv preprint arXiv:1506.00553. 2015

[76] Scornet E, Biau G, Vert JP. Consistency of random forests. The

Annals of Statistics. 2015;**65**(4): 1716-1741

[77] Murphy KP. Machine Learning: A Probabilistic Perspective. MIT Press; 2012

[78] Yoriyaz H. Monte carlo method: Principles and applications in medical physics. Revista Brasileira de Física Médica. 2009;**3**(1):141-149

[79] Gong Y, Tang W, Zhou L, Yu L, Qiu G. A discrete scheme for computing Image's weighted Gaussian curvature. IEEE International Conference on Image Processing (ICIP). 2021;**2021**:1919-1923. DOI: 10.1109/ICIP42928.2021.9506611

[80] Tran T. Semantic Segmentation Using Deep Neural Networks for MAVs. 2022