International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Volume XXXIX-B3, 2012
XXII ISPRS Congress, 25 August – 01 September 2012, Melbourne, Australia

REAL TIME SPEED ESTIMATION FROM MONOCULAR VIDEO

M. S. Temiz [a, *], S. Kulur [b*], S. Dogan [a]

[a] Ondokuz Mayis University, Dept. of Geomatic Engineering, Samsun, TURKEY – (mstemiz, sedatdo)@omu.edu.tr
[b] Istanbul Technical University, Dept. of Geomatic Engineering, Istanbul, TURKEY - kulur@itu.edu.tr

Commission III, WG III/V

KEY WORDS: Video images, speed estimation, monocular video, object tracking, optical flow, traffic surveillance.

ABSTRACT:

In this paper, detailed studies have been performed for developing a real time system to be used for surveillance of the traffic flow by using monocular video cameras to find speeds of the vehicles for secure travelling are presented. We assume that the studied road segment is planar and straight, the camera is tilted downward a bridge and the length of one line segment in the image is known. In order to estimate the speed of a moving vehicle from a video camera, rectification of video images is performed to eliminate the perspective effects and then the interest region namely the ROI is determined for tracking the vehicles. Velocity vectors of a sufficient number of reference points are identified on the image of the vehicle from each video frame. For this purpose sufficient number of points from the vehicle is selected, and these points must be accurately tracked on at least two successive video frames. In the second step, by using the displacement vectors of the tracked points and passed time, the velocity vectors of those points are computed. Computed velocity vectors are defined in the video image coordinate system and displacement vectors are measured by the means of pixel units. Then the magnitudes of the computed vectors in the image space are transformed to the object space to find the absolute values of these magnitudes. The accuracy of the estimated speed is approximately ± 1-2 km/h. In order to solve the real time speed estimation problem, the authors have written a software system in C++ programming language. This software system has been used for all of the computations and test applications.

1. INTRODUCTION

Using of video sequences are increasing in several applications for automation, for instance tracking moving objects, extracting trajectories, finding traffic intensity or estimating vehicle velocity etc. In this paper we explain the results of our system which we have developed for automatic real-time estimation of the speed of moving vehicles from video camera images. This approach requires only the knowledge of two lengths on the ground plane, no interior or exterior calibration parameters are required if frontal image acquisition plan is assumed. So we assume that the rest of interesting road segment is planar and straight and the camera is fixed on the ground. Our system can determine vehicle speed in real time and with high accuracy by using any kind of digital camera. This procedure involves two main steps to be solved. At the first step, enough number of points from the vehicle is to be selected, and these points should be tracked at least on two successive video frames. At the second step, by using the displacement vectors of the tracked points and passed time, velocity vectors of those points are computed. Due to the nature of the images' perspective effects, the certain geometric properties of the scene such as lengths, angels and area ratios are distorted. These distortions must be corrected. At first, the background image is detected by using background extraction methods and lines on the images are detected automatically with Hough Transformation approach. In order to rectify images of the scene, we use vanishing point geometry and thus solve the scale problem. Vanishing points are automatically detected with those extracted lines by using least squares adjustment. Subsequently a projective transformation is applied to rectify images by using these vanishing points. Actually there is no need to apply projective transformation for all over the image for rectification. Instead of rectifying the whole image, we only rectify the values of the distorted velocity vectors and thus gain time for real time computations.
In order to track moving objects from video images, the points to be tracked which belong to the moving object on the successive video frames should be selected automatically. For this purpose we use corner detection algorithms to automatically select those points.

In the literature, generally two methods are used for tracking the selected points. Maduro et al. (2008), have used Kalman filtering method for tracking points on the subsequent frames to estimate the velocities of the vehicles, and they have reported 2% accuracy. In the similar manner, Li-Qun et al. (1992), Jung and Ho (1999), Melo et al. (2004) and Hu et al. (2008b) have also used the Kalman filtering method for tracking the selected points. The other method used for tracking the selected points is optical flow. Sand and Teller (2004), Sinha et al. (2009), Doğan et al. (2010) and Santoro et al. (2010), have used optical flow method for tracking points in subsequent frames.

For tracking of the selected points we use Lukas – Kanade (LK) Optical Flow approach. In order to test the system, we have monitored a car which moves with a GPS receiver to measure its speed by GPS technique and compared the GPS speed values to the values of our video camera speed estimation system and we obtained the vehicle speed within ± 1 km/h accuracy. In this paper, we explain the determination of the vehicle's speed and we give sample applications selected from our test studies. We also explain the approaches and mathematical models that we used for the solution of the problem. Solutions and the models to be used for speed estimation problem vary according to the applications and their final purposes. When applications related to vehicle speed estimation problems are investigated, two main fields are distinguished: traffic surveillance (Gupte, et al., 2002)

* Corresponding author.

and driver assistance systems or intelligent vehicle systems (Guo, et al., 2005).

Traffic surveillance systems generally involve those applications which require global information on the general traffic situation of the roadways rather than individual vehicles travelling on the roads. For example, estimation of the speed of a traffic flow of a roadway at different times and dates (Dailey, et al., 2000), (Schoepflin and Dailey, 2003) belongs to this group, as well as determination of the traffic density, timing of the traffic lights, signalization works, etc.

2. RECTIFICATION OF FRAMES

Due to the nature of the images' perspective effects, certain geometric properties such as lengths, angels and area ratios are distorted. These distortion effects must be corrected. If the image plane is in the ideal case, then any parallel line in the vertical planes must remain parallel in the image plane. Similarly, the parallel lines on the horizontal plane must also remain parallel in the image plane. If the image plane is far away from the ideal situation, these parallel lines will not be parallel in the image plane. This means that those parallel lines in the object space intersect to each other on the image plane. Intersection points of the parallel lines are known as vanishing points. By using vanishing points and their corresponding vanishing planes at the horizontal and vertical directions, the images can be rectified by using vanishing points geometry (Heuvel, 2000), (Simond and Rives, 2003), (Cipolla, et al., 1999), (Grammatikopoulos, et al., 2002) so that they represent the ideal case. For this purpose, we used two methods. The first one is finding the lines manually and the second one is finding the vanishing lines automatically by using the Hough transformation. After Hough transformation, we compute the intersection points (vanishing points) of the selected lines in the image coordinate system. By using those vanishing points we rectify the image by making the vanishing lines parallel to each other. Figure 1 shows original and rectified frames.

Figure 1. The original (left) and the rectified (right) frame.

When the rectification parameters are found for the first time, they can be used until the camera changes its position. Thus, at the beginning of the speed estimation application, at first the rectification parameters can be found for the first time and these parameters can be used as long as the camera stays stable. For the speed estimation problem, after rectification parameters have been found, it is not necessary to rectify the whole image. Instead, only the selected and tracked point coordinates may be rectified for speed improvement of the real time computational cost. But however, we give the wholly rectified image on the right image of the Figure 1, for visual evaluation of the reader.

3. SPEED ESTIMATION

At the first step, enough number of points from the vehicle should be selected, and these points should be tracked at least on two successive video frames.

3.1 Automatic Selection of Points to be Tracked

In order to track moving objects with video images, points to be tracked which belong to the object on the successive video frames, should be selected automatically. It is well known that good features to be tracked are corner points which have large spatial gradients in two orthogonal directions. Since the corner points cannot be on an edge (except endpoints), aperture problem does not occur. One of the most frequently used definitions of a corner point is given in (Harris and Stephens, 1988). This definition defines a corner point by a matrix which is expressed by second order derivatives. These derivatives are partial derivatives of pixel intensities on an image and are $\partial 2x$, $\partial 2y$ and $\partial x \partial y$. By computing second order derivatives of pixels of an image, a new image can be formed. This new image is called "Hessian image". The name "Hessian" arises from the Hessian matrix that is computed around a point (Doğan, et. al, 2010). The Hessian matrix in 2D space is defined by:

$$\begin{bmatrix} \frac{\partial^2 I}{\partial x^2} & \frac{\partial^2 I}{\partial x \partial y} \\ \frac{\partial^2 I}{\partial y \partial x} & \frac{\partial^2 I}{\partial y^2} \end{bmatrix} \quad (1)$$

Shi and Tomasi (1994), suggest that a reasonable criterion for feature selection is for the minimum eigenvalue of the spatial gradient matrix to be no less than some predefined threshold. This ensures that the matrix is well conditioned and above the noise level of the image so that its inverse does not unreasonably amplify possible noise in a certain critical directions.

When it is desired to extract precise geometric information from the images, the corner points should be found within a sub-pixel accuracy. For this purpose, the all candidate pixels around the corner point can be used. By using the smallest eigenvalues at those points, a parabola can be fitted to represent the spatial location of the corner point. The coordinates of the maximum of the parabola is assumed to be the best location for being a corner. Thus the computed coordinates are obtained in subpixel precision (Doğan, et. al, 2010).

In our system, as soon as the camera begins for image acquisition, points are selected continuously in real time from the frame images. On the first frame, points are selected and on the next frames those points are tracked and instantaneous velocity vectors of those points are computed.

3.2 Tracking of Selected Points

For speed estimation, correspondence of each selected point on the first frame on which the vehicle appears for the first time, must be found on the next (successive) frame. In the ideal case, correspondence of a selected point must be the same point on the next frame. In order to find the corresponding point, there is no prior information other than the point itself. If we assume that each image in the each frame is flowing by the very short time period and thus changing the position during the flow, then a modelling approach which models this flow event can be used. These kinds of flow models are called “optical flow”.

3.2.1 Lukas-Kanade (LK) Optical Flow Method: When only one video camera is used, there is no information other than themselves of the selected points to find their correspondences on the next frame. For this reason, it is not possible to know exactly where the corresponding points are on the next frame. But however, by investigating the nature of the problem, some assumptions may be made about the possible locations where the corresponding points might be located. In order to ensure these assumptions are as close to the physical reality as possible, there must exist a theoretic substratum at which these assumptions are supported. Furthermore, this theoretic substratum must be acceptable under some certain situations. Lukas and Kanade have cleverly given three assumptions for the solution of the correspondence problem in their paper (Lukas and Kanade, 1981). The assumptions of Lukas-Kanade Method are:

1. *Intensity values are unchanged:* This assumption asserts that the intensity values of a selected point p(t) and its neighbours on the frame image I(t), do not change on the next frame I(t + Δt), where Δt is too short time period less than one second. When the time interval Δt that passed between two successive image frames is too short, it can be seen that really the possibility of the occurrence of this assumption is too high. Because, in a very short time period which is measured in milliseconds, the effects such as the lighting conditions of the scene medium etc. that cause the intensity values to be changed must not lead to meaningful change effects since the time is too short.
2. *Location of a point between two successive frames changes by only a few pixels:* The reasoning which the assumption is based on, is similar to the reasoning of the first assumption. Between the frame images I(t) and I(t + Δt), when Δt is getting smaller, then the displacement amount of the point also gets smaller. According to this observation, a point p(t) at (x,y) coordinates of image I(t) will be at the coordinates (x + Δx, y + Δy) on image I(t + Δt) and these new coordinates will be closer to the previous coordinates within a few pixels. Thus the positions of the corresponding points on both images will be very close to each other.
3. *A point behaves together with its neighbours:* The first two assumptions, which are assumed to be valid for a point must also be valid for its neighbours. Furthermore, if that point is moving with a velocity v, then its neighbours must also move with the same velocity v, since the motion duration Δt is too short.

The three assumptions above help develop an effective target tracking algorithm. In order to track the points and to compute their speeds by using the above assumptions, it is necessary to express those assumptions with mathematical formalisms and then velocity equations must be extracted by using these formalisms. For this purpose, the first assumption can be written in mathematical form as follows:

$$I(\mathbf{p}(x,y,t),t) = I(\mathbf{p}(x,y,t),t+\Delta t) \tag{2}$$

where I(**p**,t) is the intensity value of a point p on the image I(t) which was taken at the time instant t. Note that the geometric location of the point is expressed by its position vector $\mathbf{p} \in R^2$ (i.e., in 2D space). Here I(**p**,t) expresses the intensity value of a pixel at the point **p** on the frame image I(t). In similar way, the right side of the equation expresses the intensity value of the corresponding pixel at the point **p** + Δ**p** on the frame image I(t + Δt). Accordingly, Equation (2) says that the intensity value of the point p on the current image frame does not change during the time period Δt that passed. In other words, it expresses that the intensity I(**p**,t) does not change by the time Δt. In the more mathematical sense, change rate of I(**p**,t) iz zero over the time period Δt. This last situation is formally written as follows:

$$\frac{\partial I(\mathbf{p}(x,y,t),t)}{\partial t} = 0 \tag{3}$$

If the derivative given in Equation (3) is computed by using the chain rule of derivative, we obtain:

$$\frac{\partial I(\mathbf{p}(x,y,t),t)}{\partial t} = \frac{\partial I(\mathbf{p},t)}{\partial \mathbf{p}}\frac{\partial \mathbf{p}(t)}{\partial t} + \frac{\partial I}{\partial t} = 0 \tag{4}$$

In Equation (4), the derivative $\partial I / \partial p$ is spatial derivative at point **p** on the image frame I(t). Thus it can be expressed by $\partial I / \partial p = \nabla I$. We can write this expression in explicit form as follows:

$$\nabla I = \frac{\partial I}{\partial x} i + \frac{\partial I}{\partial y} j = I_x i + I_y j \tag{5}$$

The derivative $\partial \mathbf{p}(t) / \partial t$ can also be written in a more explicit form:

$$\frac{\partial \mathbf{p}(t)}{\partial t} = \frac{\partial}{\partial t}(xi + yj) = \frac{\partial x}{\partial t} i + \frac{\partial y}{\partial t} j \tag{6}$$

If Equation (6) is investigated carefully, it can easily be seen that the vector $(\partial x / \partial t)i$ is equal to the velocity of the point **p** in the x-axis direction. In other words, it is the x component namely v_x component of the velocity vector **v**. In similar way, the vector $(\partial y / \partial t)\, j$ is the y component namely v_y component of the velocity vector v. Now we can rewrite the Equation (6) as follows:

$$\frac{\partial \mathbf{p}(t)}{\partial t} = v = v_x i + v_y j \tag{7}$$

If again Equation (4) is investigated, it is seen that the derivative $\partial I / \partial t = I_t$ expresses the change rate of the intensity values at point **p**, between the frame images I(t) and I(t + Δt). Thus, Equation (4) can be rewritten as follows:

$$\nabla I.V + I_t = 0 \tag{8}$$

where:

$$\nabla I = \begin{bmatrix} I_x \\ I_y \end{bmatrix} \text{ and } \mathbf{V} = \begin{bmatrix} V_x \\ V_y \end{bmatrix} \tag{9}$$

Then Equation (8) can be written as:

$$\begin{bmatrix} I_x & I_y \end{bmatrix} \begin{bmatrix} V_x \\ V_y \end{bmatrix} = -I_t \tag{10}$$

The values of I_x, I_y and I_t in Equation (10) can easily be computed from the frame images. The variables v_x and v_y are two unknown components of the velocity vector **v** and these are respectively the components in the directions x and y axes of the image coordinate system. In Equation (10), we have two unknowns to be solved, but we only have one equation. Since only one equation is not enough for unique solution of the unknowns, at the moment it seems not possible to solve these unknowns. In order to solve these two unknowns, we need more independent equations. For this purpose, the third assumption of the LK algorithm is used. That is, point **p** behaves together with its neighbours. So its neighbours must also satisfy the Equation (10). In other words, neighbour points (or pixels) of point **p** must move with the same velocity $\mathbf{v}(v_x,v_y)$. According to these explanations, the same equations as (10) are written for 3 × 3 or 5 × 5 neighbourhood of the point **p**. In this case, we totally have 9 or 25 equations with the same unknowns v_x and v_y. Now the unknowns can be solved with overdetermined set of Equations (10) by using least squares or total least squares estimation method (Doğan, et. al, 2010).

During the real time tracking, some selected points may not be seen on the next frame. This situation may arise because of different reasons. Especially, when the vehicle is entering into or exiting from the FOV of the camera, the possibility of occurrence of this situation is too high. In order to prevent such situations, we have interpreted the algorithm with the image pyramid approach, which uses coarse to fine image scale levels. For details of the image pyramid approaches, we refer the reader to (Bouget, 2000) and (Bradsky and Kaehler, 2008).

3.3 Estimation of Speed

To find the vehicle speed, successive frame images of the camera can be used. In this case, only the instantaneous speed can be found. This instantaneous speed is computed as follows:

$$\mathbf{v} = \frac{\Delta p}{\Delta t} \tag{11}$$

where **v** is instantaneous velocity vector of a point and $\mathbf{v} \in R^2$ (i.e., in 2D space since one camera is used), $\Delta\mathbf{p}$ is displacement vector of that point and $\Delta\mathbf{p} \in R^2$. The displacement vector expresses the spatial displacement of a point during the time interval Δt. Here the time interval Δt is equal to the time which passes between two successive video frames and is equal to the frame replay rate (or frame capture rate) of the camera. In the experiments given in this paper, Δt is 33.3 milliseconds, which is the frame capture time of the camera that we used. Equation (11) gives the instantaneous speed (or velocity) of a point which is marked on the vehicle and selected for tracking. To find the velocity of the vehicle, only one point is not enough. During the selection of the points from the image of the vehicle, local approaches are used. If some errors occur during this selection step, the computed velocity vector will be affected by those errors and so the computed speed will be erroneous. For this reason, to estimate the speed of a vehicle, many more than one point should be selected and all of their instantaneous velocity vectors should be computed. Then by averaging the instantaneous velocity vectors of the whole selected points, the instantaneous velocity vector of the vehicle is found. For the formal expression, let us assume that n points are selected from the vehicle to be tracked and let $\mathbf{v}_i$ (t) (i = 1, ..., n) represent the instantaneous velocity vectors of each of n points at time instance t. Then by using those instantaneous velocity vectors, we can find the instantaneous velocity vector of the vehicle by:

$$\mathbf{v_{iv}} = \frac{1}{n} \sum_{i=1}^{n} v_i \tag{12}$$

where $\mathbf{v_{iv}}$ is the instantaneous velocity vector of the vehicle at time instance t, v_i is the instantaneous velocity vector of i^{th} point on the vehicle and n is the number of the selected and tracked points. Here, it should be noted that, if some of the vi vectors are erroneous, then $\mathbf{v_{iv}}$ will also be erroneous. So, before computing the instantaneous velocity $\mathbf{v_{iv}}$ of the vehicle, the erroneous $\mathbf{v_i}$ vectors must be eliminated. Then the value of n also changes, i.e., number of the points decreases. For the elimination of the erroneous vectors, standard deviation of the n velocity samples can be used for fast evaluation:

$$\|\mathbf{V}\| \le \|\bar{V} - \delta_V\| \tag{13}$$

In order to find absolute values of displacement vectors or velocity vectors in object space, the vectors computed in video image coordinate system should be transformed to the object coordinate system which is in the object space. For this purpose, at least the length of a line joining two points within the field of view of the camera and on the road and aligned along the velocity vectors, must be measured precisely. In this paper, we measured the lengths of two lines along the road by geodetic measurements using a simple measurement tape, within a precision of ±1 millimetre.

4. EXPERIMENTS AND RESULTS

In this paper, we propose a method for real-time estimation of the speed of moving vehicles by using uncalibrated monocular video camera. Since it is not possible to extract 3D geometric information with one camera, in order to solve the speed estimation problem, some geometric constraints are required and the images should be taken under these constraints and the processing procedures should also be performed with those restrictions. For example, we assume that the imaged scene is flat. Perspective distortions on the acquired images must be either very small or of a degree that they can easily be rectified.

We have used a camera with a frame rate of 30 fps and with an effective area of 640 x 480 pixel^2. The pixel size which corresponds to the effective area of the camera is 9 microns. The focal length of the camera is 5.9 mm. We capture images in grey level mode at 30 fps (frames per second), meaning that a frame is captured within 33.3 milliseconds after the previous frame had been obtained. In order to solve the real time speed estimation problem, the authors have written a software system in C++ programming language. This software system has been used for all of the computations and test applications. Our software consists of two steps which contains offline and online operations, for details refer to (Doğan, et. al, 2010). The operations of step I are performed offline at the beginning of the speed estimation problem and it contains rectification procedures. After step I has been completed, the real time procedures begin. We have used OpenCV API functions to perform the capturing images from camera and eliminate undesired background changes operations. The rest of the operations are performed with our own codes written with Visual Studio C++ 2010. The total time of the operations takes about 30 milliseconds for our real time applications with a laptop computer (Intel Core i7 2.6 Ghz CPU, 8 GB RAM). Figure 2 shows a general view of our software.

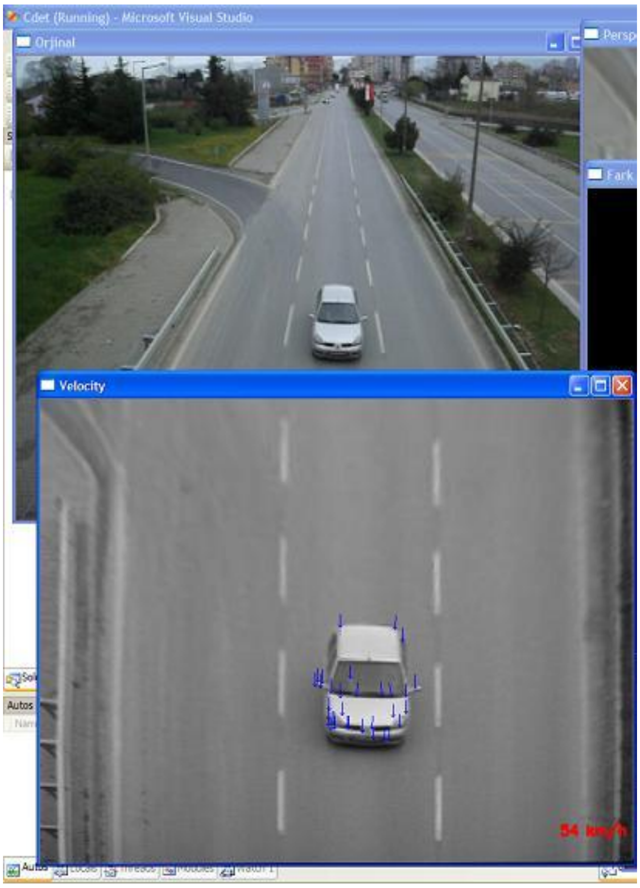

Figure 2. Estimation of speed.

Accuracy of the estimated speed of our system is ±1-2 km/h. We tested the system by comparing the estimated speeds to GPS measured speeds which measures speeds with very high accuracy about 0.1 knot (0.05 m/s) or 0.018 km/h (Keskin and Say, 2006), (Al-Gaadi, 2005).

5. CONCLUSIONS

In this paper, we have explained the real time speed estimation problem and its solution by using monocular video images of the vehicle. The accuracy of the estimated speed had been obtained and is approximately ±1- 2 km/h. The sparse optical flow technique is a very effective technique for the speed estimation of the vehicles.

In our earlier study, we have used our technique for the speed estimation of the vehicles from side view images of the road scene (Doğan, et. al, 2010). In this current paper, we have modified some steps of our earlier system and used a camera tilted downward a bridge and so we have acquired the top view images of the road scene. As seen from the results related to the test experiments, both of our methods give the same accuracy.

REFERENCES

Bouget, J.Y., 200. Pyramidal Implementation of the Lucas Kanade Feature Tracker Description of the Algorithm; Intel Microprocessor Research Labs: Santa Clara, CA , USA.

Bradsky G., and Kaehler A., 2008. *Learning OpenCV Computer Vision with the OpenCV Library*, O'Reilly Media, Inc, 555 p.

Cipolla, R., Drummond, T., Robertson, D., 1999. Camera calibration from vanishing points in images of architectural scenes. In *Proceedings of the 10th British Machine Vision Conference*, Nottingham, England, September 1999.

Dailey, D.J., Cathey, F.W. and Pumrin, S., 2000. An Algorithm to Estimate Mean Traffic Speed Using Uncalibrated Cameras, *IEEE Transactions on Intelligent Transportation Systems*, Vol: 1, No: 2, pp. 98-107.

Doğan, S., Temiz, M.S. and Külür, S., 2010. Real Time Speed Estimation Of Moving Vehicles From Side View Images From An Uncalibrated Video Camera. *Sensors.* 10(5), pp. 4805-4824.

Grammatikopoulos, L.; Karras, G.E.; Petsa, E. Geometric information from single uncalibrated images of roads. *Int. Arch. Photogramm. Remote Sens.*, 34, pp. 21-26.

Gupte, S., Masaud, O., Martin, F.K.R., Papanikolopoulos, N.P., 2002. Detection and classification of vehicles. *IEEE Trans. Intell. Transp. Syst.*,pp. 37-47.

Guo, M., Ammar, M.H., Zegura, E.W., 2005. A vehicle to vehicle live viedo streaming architecture. *Pervasive Mob. Comput.* (1), pp. 404-424.

Harris C. and Stephens M., 1988. A Combined Corner and Edge Detector, Proceedings of the *4th Alvey Vision Conference*, pp.147-151.

Heuvel, A.F., 2000. Line photogrammetry and its application for reconstruction from a single image. *Publikationen der Deutschen Gesellschaft für Photogrammetrie und Fernerkundung*, Vol. 8, pp. 255-263.

Hu, N., Aghajan, H., Gao, T., Camhi, J., Lee, C.H. and Rosario, D., 2008. Smart Node: Intelligent Traffic Interpretation, *World Congress on Intelligent Transport Systems*.

Jung, Y.K. and Ho, Y.S., 1999. Traffic Parameter Extraction Using Video-based Vehicle Tracking, *Proc. IEEE Intelligent Transportation System Conf.'99*, Vol.1, pp. 746-769.

Li-Qun, X., Young, D. and Hogg, D.C., 1992. Building a model of a road junction using moving vehicles in: Hogg, D & Boyle, R.D. (editors) *British Machine Vision Conference*, pp. 443-452. Springer.

Lucas B.D., and Kanade T., 1981. An Iterative Image Registration Technique with an Application to Stereo Vision, Proceedings of the *1981 DARPA Imaging Understanding Workshop*, pp. 121-130, Washington, USA.

Maduro, C, Batista, K, Peixoto, P. and Batista, J, 2008. Estimation Of Vehicle Velocity And Traffic Intensity Using Rectified Images, *15th IEEE International Conference on, Image Processing*, pp. 777 – 780.

Melo J., Naftel A., Bernardino A., and Victor J.S., 2004. Viewpoint Independent Detection of Vehicle Trajectories and Lane Geometry from Uncalibrated Traffic Surveillance Cameras, *Int. Conf. On Image Analysis and Recognition*, 29 Sept.- 1 Oct., Porto, Portugal.

Sand, P. and Teller, S., 2004. Video Matching, *ACM Transactions on Graphics (TOG)*, Vol. 22, No. 3, pp. 592-599.

Santoro, F., Pedro, S., Tan, Z.H. and Moeslund, T.B., 2010. Crowd Analysis by using Optical Flow and Density Based Clustering, *18th European Signal Processing Conference, Aalborg*, Denmark, August 23-27.

Schoepflin, T.N., Dailey, D.J., 2003. Dynamic camera calibration of roadside traffic management cameras for vehicle speed estimation. *IEEE Trans. Intell. Transp. Syst,* Vol. 4, pp. 90-98.

Shi J., and Tomasi C., 1994. Good Features To Track, *9th IEEE Conference on Computer Vision and Pattern Recognition*, pp. 593-600, Seattle, USA.

Simond, N., Rives, P., 2003. Homography from a vanishing point in urban scenes. In *Proceedings of the Intelligent Robots and System*, Las Vegas, NV, USA, October 2003; pp. 1005-1010.

Sinha, S., Frahm, J.M., Pollefeys, M. and Genc, Y., 2009. Feature Tracking and Matching in Video Using Programmable Graphics Hardware, *Machine Vision and Application*, Nov. 2009.