car detection, character recognition, traffic scenes, vision sensor

Włodzimierz Kasprzak *, Marcin Jankowski

The implementation of a vision sensor for traffic surveillance

1 Road traffic control within OMNI

The EC project OMNI ("Open Model For Network-wide Heterogeneous Intersection-based Transport Management") [1] provides an open-architecture model for road traffic management, by defining: standard interfacing for new applications or devices, and the integration of technology (surveillance applications, advanced sensors, other urban traffic strategies, etc) from different vendors.

First of all a surveillance application allows for an automatic incident detection (Fig. 1). Next, the advanced traffic control system means the integration of two complementary systems for traffic light control: one for the real-time local control of signals at the junction level, the other for the congestion monitoring and control at the network level. Other important application is devoted for customized information to users via WWW - real time traffic information may be presented to the user through the INTERNET (traffic status on a set of given itineraries, on line trip-planning, alternative routing and information customized to a profiled user). Finally, the fleet management means the use of GPS/GSM and the road sensors to locate the vehicles and eventually to reschedule their activities.

The OMNI model provides classes and interfaces that describe various traffic situations appearing on road crossings in an urban network (Fig. 2).One of the devices on road crossings, specified in the OMNI model, is the video sensor [2], [3] (Fig. 3).

In this paper we describe the implementation of a modular video sensor software - the general design was initiated in our earlier paper [9]. We have implemented following vision modules within OMNI [4]: (1) camera auto-calibration, (2) traffic flow and (3) queue length detection, (4) car plate recognition. An additional communication module provides an interface to the OMNI-MOUN data base. A programming class implements each vision module and it can be transformed into an independent process or object, if a distributed system is required for it, by using the libraries of DCOM or CORBA.

2 Communication with OMNI

The OMNI interface module constitutes an object of a class called `LLFieldVideoSensor`. The information exchange between OMNI and our video sensor is implemented by a set of DCOM interfaces, which are defined by OMNI: `ILLFieldVideoSensor` - to control the image anal-

* Institute of Control and Computation Eng.,Warsaw University of Technology, ul. Nowowiejska 15/19, 00-665 Warszawa. E-mail: W.Kasprzak@ia.pw.edu.pl , M.Jankowski@elka.pw.edu.pl

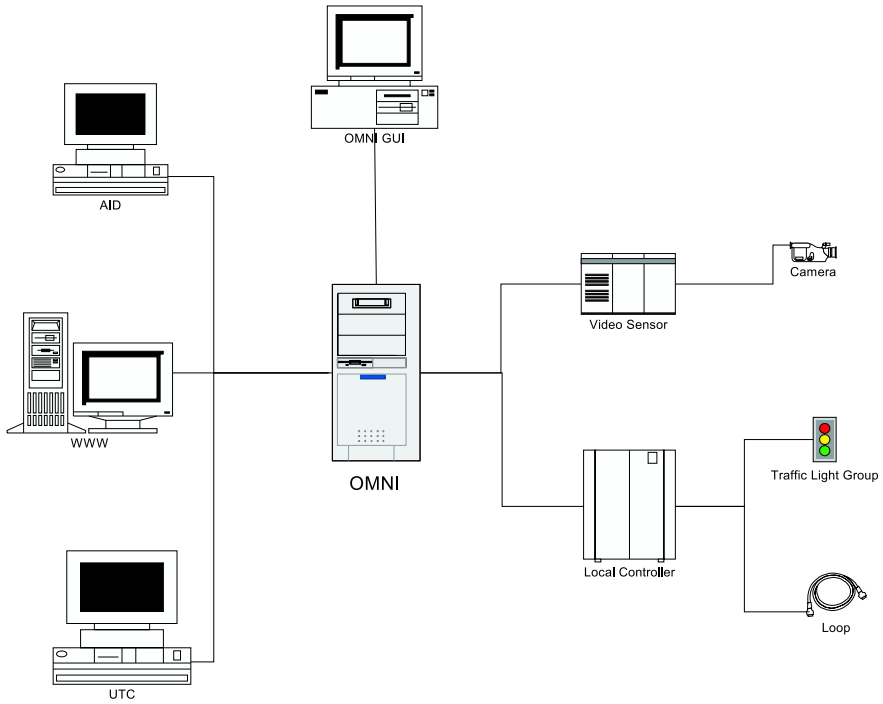

Figure 1. Example of an OMNI-based traffic information system.

ysis process, `ILLLaneSensor` - to inform about the lane's queue length and occupancy ratio, `ILLZoneSensor` - to inform about the traffic flow (Fig. 4).

The selection of appropriate interface objects appears by referring the observed road (selecting the interface HLArc). Then the image segment along which we measure the queue length corresponds to an object of type HLLane, that is linked with the selected HLArc, whereas the border line, which we use to detect the passing vehicles, corresponds to an object of LLZone, linked to HLArc.

Let us shortly describe the initialization procedure of our sensor program, which is a DCOM-client of the OMNI-MOUN application. At first, the client should get the pointer to the server object interface `MOUNManager`. Next we select the sensor (`LLFieldVideoSensor`), to be

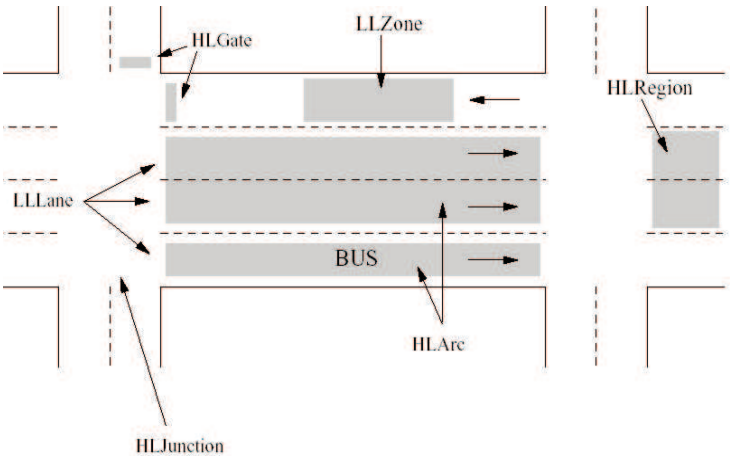

Figure 2. The road concepts provided in the OMNI model [1].

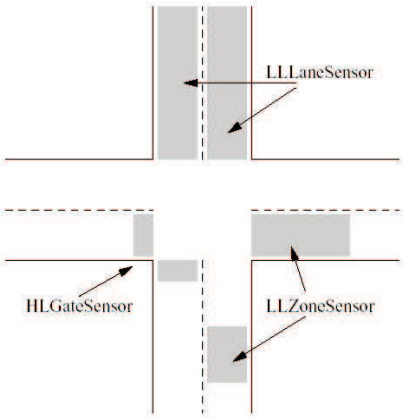

Figure 3. Road crossings controlled by sensors in the OMNI model [1].

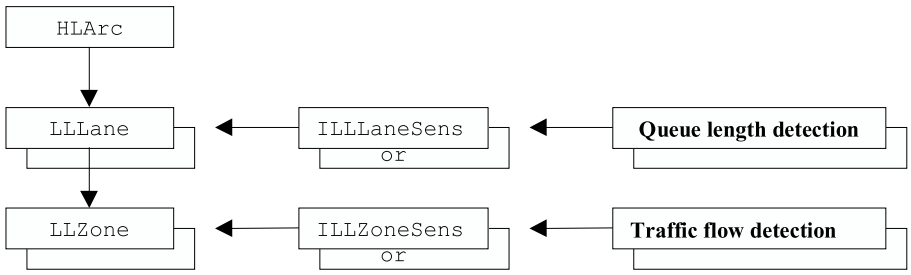

Figure 4. The objects responsible for communication of the vision sensor with OMNI.

controlled by the client program, by calling the method `IMOUNManager::getIdsOfType()`. In order to select the observed road one should get the object of class `HLArc`, whose one of sensor's is our selected `LLFieldVideoSensor`, by calling `IHLArcContainer::getSensorsOnArc()`. For each road lane, along which the queue length detection should be performed, we select an appropriate road lane object of type `LLLane` and get an pointer to its interface `ILLLaneSensor` by calling `ILLLaneContainer::getSensorsOnLane()`. For each zone, where we want to detect traffic flow, we select an appropriate object of class `LLZone` and get the pointer to its interface `ILLZoneSensor` by calling `ILLZoneContainer::getSensorsOnZone()`.

In order ro transmit the measurements the client calls the method `Measure`, which is declared by both interfaces `ILLZoneSensor` and `ILLLaneSensor`.

3 The auto-calibration module

A pin-hole camera model is assumed, which requires the on-line detection (or a priori knowledge) of 6 geometric parameters - the position (X_c, Y_c, Z_c) and the orientation of the camera (α, β, γ), and 4 intrinsic parameters - the focal length of the projective transformation (F), the pixel size (sP) and the localization of the image origin point (X_0, Y_0)). The intrinsic parameters (except focal length, which may be changed) are usually calibrated before the analysis system starts its operation, whereas the 6 geometric parameters should be modified on-line, in accordance with instantaneous, real position and orientation of the camera. Our auto-calibration procedure for traffic scene analysis assumes for simplicity, that the height (i.e. $Y_c = H$) over the road plane is known and that the the Z_c value remains constant (let fix it to $Z_c = 0$). In this way the camera direction angles (α, β, γ) and the "side" position X_c relative to the general road system (for example fixed with the central road axis) have to be on-line re-computed.

We have implemented a semi-automatic *camera calibration* procedure, a simplification of [5]. The user should first measure manually in the environment and secondly he should set the following parameters, required by the semi-automatic calibration procedure (Fig. 5): to select image lines, defining the Vanishing Point (VP), to point the line segment, according to which the calibration of the camera's focal length shall be performed, to measure the height over the road of camera's origin center. After the transformation parameters from the camera to road coordinates are known the complete camera transformation matrix can be computed.

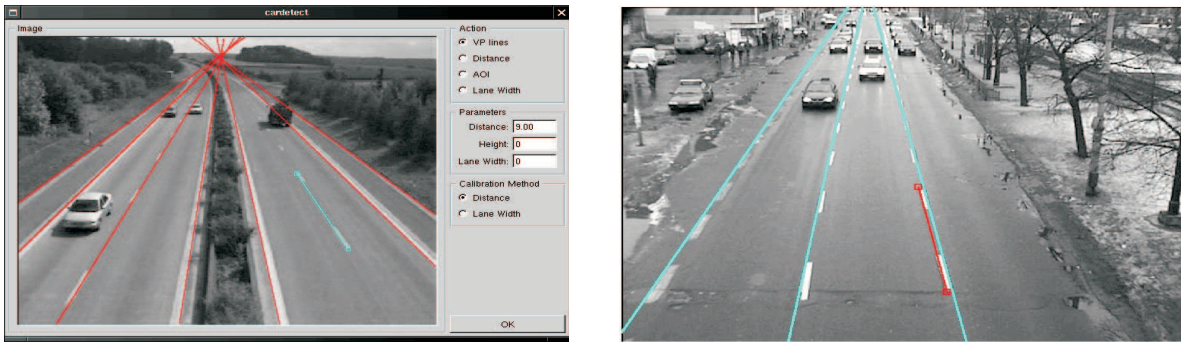

Figure 5. Camera calibration: (left) the dialogue window, used for camera calibration; (right) example of camera calibration (blue lines were automatically detected - red line is marked by the user to specify the scale factor).

4 Traffic flow and queue length detection

In a selective way, for each lane, the following 'measurement' is performed:

- *traffic flow*, i.e. the number of vehicles in a lane per minute, which are passing the zone (Fig. 6);
- *queue length* (occupancy ratio) detection - if vehicles are queuing (Fig. 7).

Figure 6. The horizontal lines represent two scan lines in the image, where the number of passing cars is counted. The vertical lines represent lanes, along which the queue lengths are detected.

The user is positioning a horizontal line segment in the image, where the vehicle counting should take place (Fig. 6). The intensity distribution along this line is examined from image to image. A normal distribution corresponds to the road area - a situation with no vehicle. An intensity increase to more brightness or an intensity drop toward more darkness is recognized as the appearance of a vehicle. The count of vehicles is increased by one. As long as the intensity distribution does not change to the usual "empty road"-one, we still assume the single vehicle continues.

Queue length detection (or in other words - line occupancy detection) means, that for each predefined lane of the road the current queue length and the occupied parts in predefined area are detected. On the basis of an already calibrated camera and a user-given (vertically or

Figure 7. Example of queue length detection: the car detected in the lane on the left side causes two black and two light regions along the line of search.

Figure 8. The queue lengths are detected and expressed numerically in terms of average car lengths, for each road lane under control.

mostly diagonally elongated) set of image lines (usually - one line per one road lane) the queue length (or dually the vehicle occupancy ratio) is detected (Fig. 7). Each image line is virtually projected back onto the road plane and its intensity distribution is examined. The road's normal intensity (exactly a small interval of intensity values) is first detected (assuming no vehicle along the line) for each distribution cell. In the working phase each cell with intensities below the minimum road intensity or above the maximum road intensity is assumed to contain a vehicle. The real length corresponding to such "occupied" cells is summarized, giving the queue length for the corresponding road lane (Fig. 8). The occupancy ratio can be computed from the queue length data, due to a simple division of the queue length by the total length of examined road.

The tests of the traffic flow and queue length detection modules have been provided in an off-line mode, while running simulations for various image sequences acquired in the road environment of Warsaw city [4] (Fig. 9). The processing speed of system modules was tested on a processor with 345.5 MFLOPS (Tab. 1).

Module	Analysis time (for 1000 frames)	MFlops required (for 25 frames/sec)
Queue length	1.77 sec	15.3 MFlops
Car counting	0.37 sec	3.2 MFlops
Licence plate	14.6 sec	126 MFlops

Table 1. Average processing times for three vision modules.

5 Licence plate recognition

The licence plate detection performs three main tasks: (1) the detection of a rectangular image area, where the licence plate is expected to be located, (2) the detection of each individual character in the licence plate area ([4], [6]); (3) applying a conventional OCR (optical character recognition) package or our character recognition procedure, which can learn character patterns from current user-defined image data ([7], [8]).

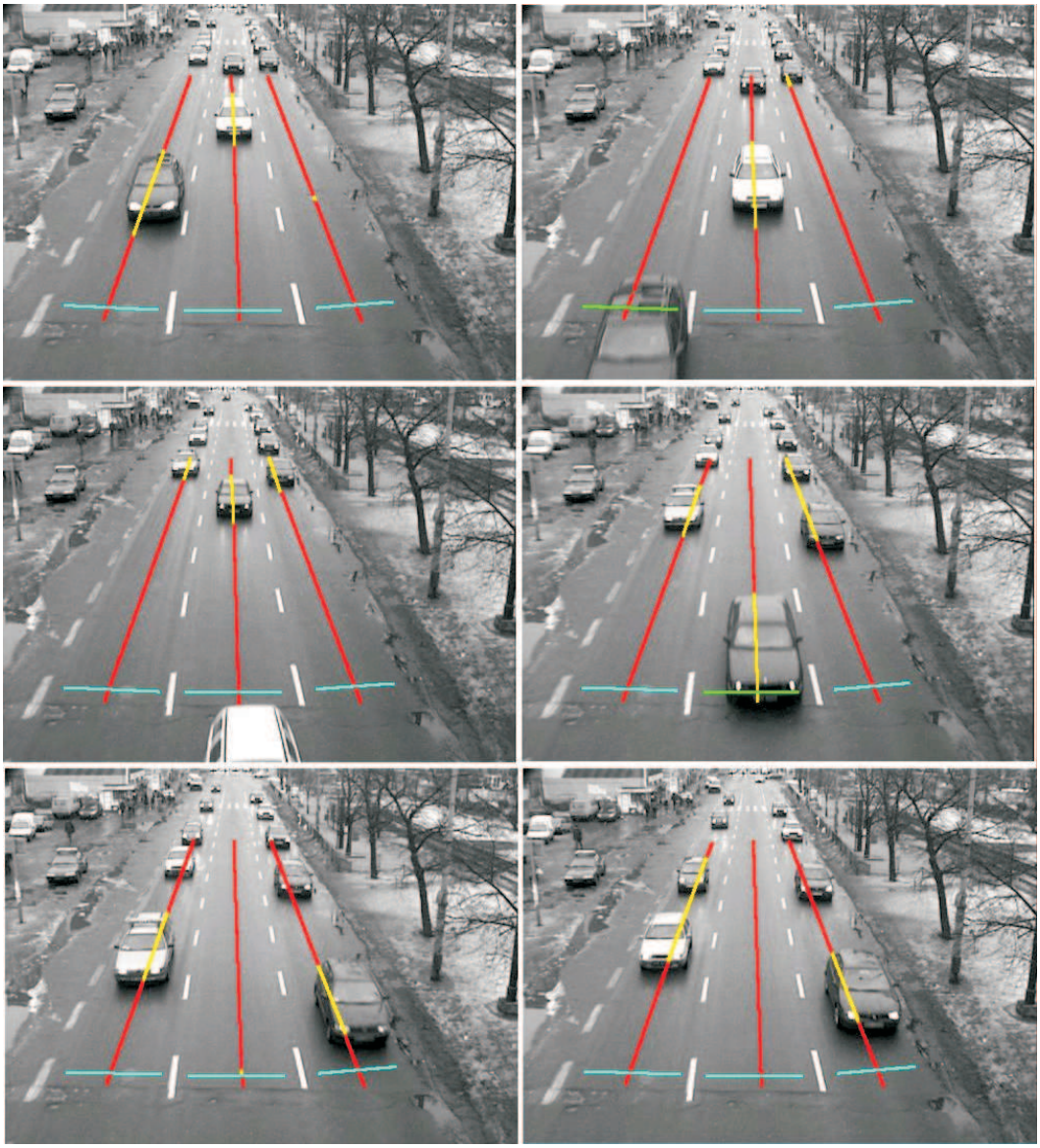

Figure 9. Some results of car counting (change from a blue horizontal line to a green one) and queue length estimation (red line is changing to a yellow one).

The first task can be performed by searching for intensity gradients along scan lines. Several adjacent lines showing similar high-gradient behavior allows to generate a hypothesis of a licence plate. In our simplified approach we operate directly on scan lines (Fig. 10). The detection procedure counts the number of 0-1 "connected" peak pairs, which appear directly one after the other along the line. If this number is over some threshold a segment hypothesis is stored (Fig. 11). Some number of next or previous image lines should also show similar behavior, then the segment hypotheses are verified and a licence plate is assumed in these segment's positions.

For every licence plate hypothesis we try to decompose its entire region into individual characters as follows :

1. Local mean removal. The local mean grey-level intensity is removed.

Figure 10. The idea of licence plate detection: the analysis of intensity distributions along image lines.

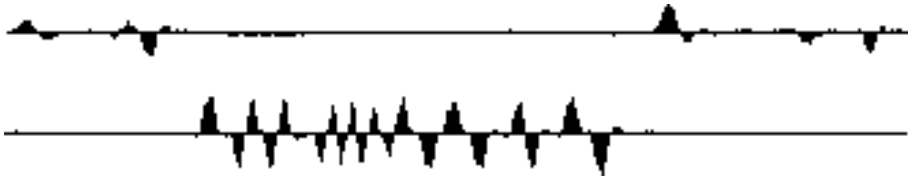

Figure 11. The distribution of first derivatives of the image intensity function along 2 image lines.

2. Binary image computation. The intensity image is converted into a binary image by comparing pixel intensities with a threshold.

3. Connected component detection. The significant pixels are grouped into 4-way connected components - the width, height, area and centric position of each connected component are recorded (Fig. 12).

4. Size filtering. All candidate-connected components greater, or smaller than a pre-defined size, corresponding to the approximate character dimension, are rejected.

5. Detection of character areas. Two histograms are computed for the remaining components - the first one along the X axis, and the second one - along the Y axis. In the X-histogram a uniformly distributed pattern of low-valued breaks between characters is detected, whereas in the Y-histogram the top and bottom borders of the character set are detected (Fig. 13).

6. Finally, the detected character areas allow to generate from the binary image a sequence of rectangular windows, corresponding to single characters (Fig. 14).

The "Plate detection" window

On the right-hand side of the window the analyzed image appears (Fig. 15). Additionally, there are visible parameters and buttons "Detect" and "Calibrate". On the analyzed image a red or blue rectangle is overlaid, which allows the calibration of the automatic detection procedure. The parameters can be set "by hand" or automatically (in the latter case the user should select some image region, which contains only the character region of the license plate, and push the button "Calibrate").

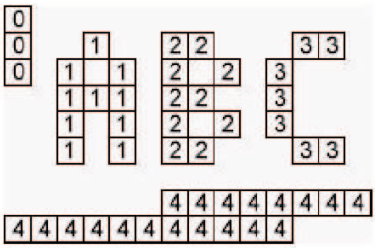

Figure 12. Detection of connected components in the binary image - one label identifies one chain.

Figure 13. Detecting the areas of individual characters by exploring two histograms - along X and Y axes.

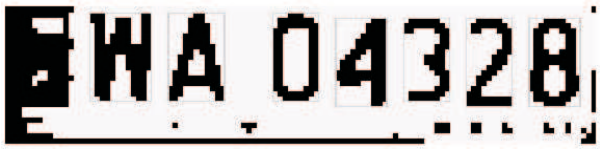

Figure 14. The detected image windows containing single characters.

The "OCR" window

Our OCR module recognizes car plates, set in accordance with polish standard. The program requires some tuning and adjustment of parameters performed by an operator, especially in the character template learning stage. The result of analysis is the set of characters, that were recognized on the car's plate, for example: "WA 04328". Figure 16 presents the window after selection of the menu item "Ocr". It contains: a magnification of the detected license plate region, a magnification of the selected character and several buttons and menu items. A blue rectangle shows the boundary of a selected character. The user can point to some character, which will be shown on the bottom, and the recognized character will be shown in an editor window. Then the user can change the recognized character and by pushing the button "Add" he can store it as a "new character".

There exists many methods of character recognition. Two simple methods are (Fig. 17: (1) counting several distances from the bottom and top lines to the character, (2)counting the number of intersections between the character and several horizontal and vertical lines. In our system we implemented a template-matching approach, where the candidate character is size-normalized to a reference size before a template matching against a set of stored templates is performed (Fig. 18).

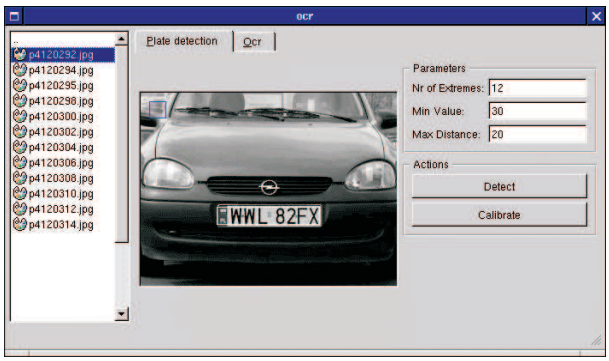

Figure 15. Dialogue window for licence plate detection.

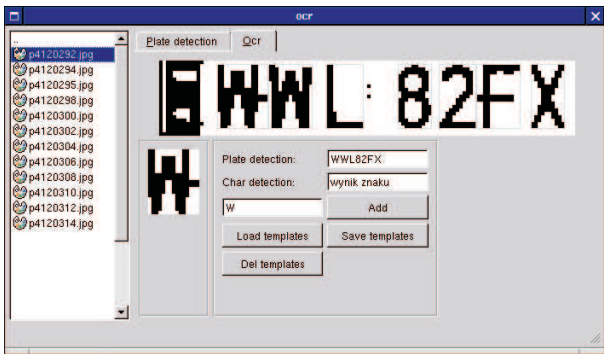

Figure 16. Dialogue window for character recognition.

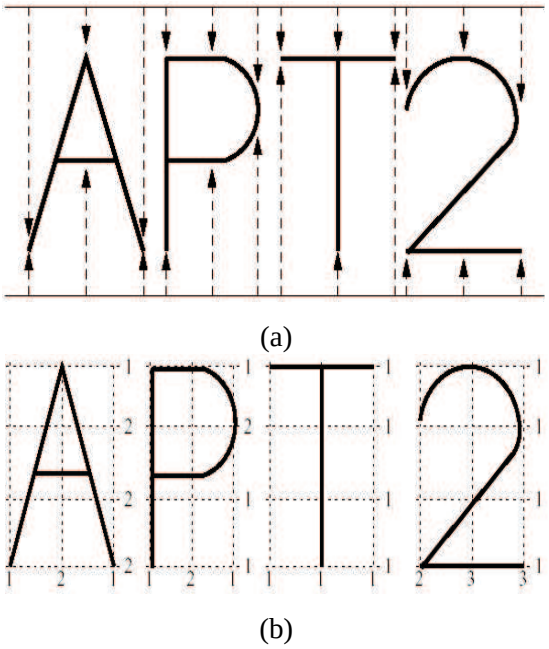

Figure 17. Character recognition by (a) distance measurement or by (b) counting the intersections with horizontal and vertical lines.

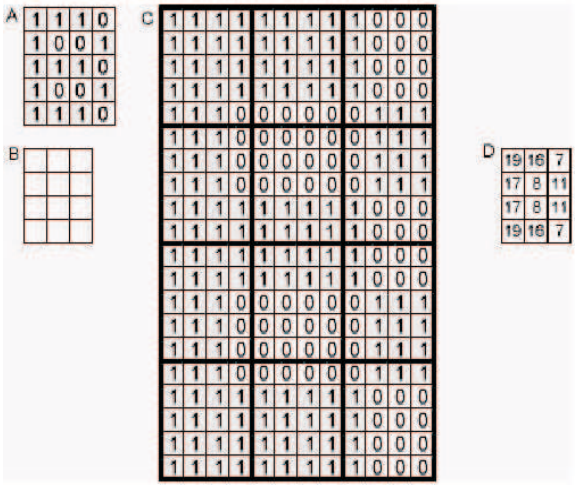

Figure 18. The window normalization procedure: (A) some detected character window, (B) the reference size required, (C) transforming the original window into an intermediate one with size determined by the product of the original size and the reference size, (D) shrinking the intermediate window to the reference size with numbers of "ones" counted in sub-windows of the intermediate window.

6 Summary

The implementation of a vision-based traffic scene sensor, working within OMNI was reported. The following vision modules were implemented and tested: (1) camera auto-calibration, (2) traffic flow and (3) queue length detection, (4) car plate recognition. The communication module - the OMNI-MOUN interface module was also implemented.

Future image measurement tasks should include vehicle classification and vehicle tracking. The first task is to detect and to classify a particular vehicle object into small car, lorry, bus, van, etc. [9]. Different approaches are possible: an iconic classification approach, assuming the front view availability [9], line-density and -orientation detection [10] or a model-based 3-D approach, assuming both height and length of a vehicle are detectable in the image [11]. The most complex image analysis includes the interpretation of object tracking data: the individual and average speed detection of vehicles in each lane over a given period of time [5]; the detection of events, like crossing a continuous road line, stopping in a non-allowed area.

Hence, we expect that the fully developed sensor, for each predefined lane, inside of some zone of some arc, will deliver following measurement data: TRAFFIC_FLOW - the number of vehicles per minute, which are passing the zone; QUEUE_LENGTH - if vehicles are queuing and not moving; VEHICLE_IDENT - for a car passing a predefined zone with a sufficiently large front (or back) part, visible in the image, its licence plate will be detected, stored and recognized; VEHICLE_CLASS - the vehicle will be classified into: person car, truck, bus and lorry; VEHICLE_TRACK - the individual speed of a vehicle passing a predefined zone will be estimated.

Acknowledgment

This work was supported by project "Open Model for Network-Wide-Heterogeneous Intersection-Based Transport Management (OMNI)", EC-IST 1999-11250. (http://www.omniproject.net).

References

[1] WWW page of the OMNI project: *Open Model for Network-Wide-Heterogeneous Intersection-Based Transport Management*, IST 1999-11250. WWW page: http://www.omniproject.net

[2] R. Blissett. *Eyes on the road*. Roke Manor Research, IP Magazine, May/June 1992, U.K.. WWW page: www.roke.co.uk

[3] K. Takahashi et al.. Traffic flow measuring system by image processing. MVA'96, *IAPR Workshop on Machine Vision Applications*, Tokyo, Japan, 1996, 245-248.

[4] M. Jankowski. *Implementation of a road traffic sensor*. Warsaw University of Technology, M.Sc. thesis, ICEE WUT, Warsaw, Feb. 2003.

[5] W. Kasprzak, H. Niemann. Adaptive Road Recognition and Egostate Tracking in the Presence of Obstacles. *International Journal of Computer Vision*, Kluwer Academic Publ., Boston - Dordrecht - London, vol 28(1998), No. 1, pp. 6-27.

[6] J. Balas-Cruz, J. Barroso, A. Rafael, E.L. Dagless. Real-time number plate reading. *4th IFAC Workshop on Algorithms and Architectures for Real-time Control*, Vilamoura, Portugal, April 1997.

[7] J. Barroso, A. Rafael, E.L. Dagless, J. Balas-Cruz. Number plate reading using computer vision. *IEEE International Symposium on Industrial Electronics*, Guimaraes, Portugal, July 1997.

[8] Y.G. Won, Y-K. Park. Property of greyscale hit-or-miss transform and its applications, *Machine Graphics & Vision*, vol. 9(2000), 539-547, ICS PAS Warsaw, Poland.

[9] W. Kasprzak. An Iconic classification scheme for video-based traffic sensor tasks", in: W.Skarbek (ed.): *Computer Analysis of Images and Patterns 2001*, Springer-Vg. LNCS 2124, Berlin, 2001, pp. 725-732.

[10] T.N. Tan, G.D. Sullivan, K.D. Baker. Fast Vehicle Localisation and Recognition Without Line Extraction and Matching. BMVC94. *Proceedings of the 5th British Machine Vision Conference*, Sheffield, U.K., BMVA Press, 1994, 85-94.

[11] W. Kasprzak. *Adaptive Erkennung von bewegten Fahrbahnobjekten in monokularen Bildfolgen mit Eigenbewegung*. Infix Publ., Sankt Augustin, Germany, 1997.