Car Detection in Consecutive Road Images by Stochastic Model

Masahiro Tanaka[†], Ryo Hamamura[‡] and Andrzej Bargiela[*]
[†] Department of Information Science and Systems Engineering, Konan University
Kobe 658-8501, Japan
m_tanaka@konan-u.ac.jp
[‡] Graduate School of Natural Science, Konan University
Kobe 658-8501, Japan
mn324008@center.konan-u.ac.jp
[*] Department of Computing, The Nottingham Trent University
Nottingham NG1 4BU, UK
andre@doc.ntu.ac.uk

Abstract

This paper discusses the image processing part in the hierarchical processing of traffic images of real-time. Since the camera is subject to pan and tilt, we first calibrate the camera position via image itself, and then extract the car imformation using stochastic model of the background. The moving objects like cars are treated as some occluding object. The processed result show a good performance.

1 Introduction

Processing of traffic image has been attracting attention of researchers of image processing and ITS [4, 5]. Cucchiara and Piccardi [2] developed a method of vehicle detection under day and night illumination. Shimai et al. [6] applied the robust statistical method to adaptive background estimation problem.

We have been developing a distributed system of processing road images and dispatching extracted, aggregated information of the traffic condition [7]. Our target is the traffic of Nottingham. The Nottingham Traffic Control Centre has been providing small images of 44 places in Nottingham captured by surveillance camera on the Web at every 5 minutes or 10 minutes, and now it is dispatching images of large size (768 × 576) for every second at 5 sites in Nottingham via Web that is made accesible to specified persons. The images on the Web are taken by the image processing system in Konan University, and the system is designed to dispatch aggregated visual information via Internet. We proposed a hierarchical structure of the processing steps [7] as Table 1, and this paper is devoted to the identification of camera position and the car detection.

Table 1: Hierarchical Structure of the Image Processing System

Layer	Processing
5	draw a congestion map and post on Web site
4	congestion estimation
3	flow estimation and signal state estimation
2	car detection
1	identification of camera position
0	input images

The camera is subject to be panned, tilted and zoomed for surveillance, and this is uncontrollable for us. Thus our problem first is to know the change of the camera position. We will develop the identifying method of pan and tilt in this paper. The car detection algorithm can be used after the calibration of camera position.

We considered an ad hoc method to find the car-existing areas in the images in SSS'03 [8], where two or three consecutive images were used to analyse. However, the processing time was too large to be used in the real-time processing system.

Here in this paper, we focus our attention on the nature of the background. The background colour is not constant, but it usually changes slowly. Thus we formulate the background image as stochastic processes, and make a judgment of car existence by using a likelihood ratio test. The points to be used for stochastic processing is defined to the points which lie along a lane on a road with constant interval.

2 Description of Incoming Images

Nottingham City Transport has been providing traffic images to the Web site from 7:00-19:00 on Mon-Sat, where the images are provided from the cameras over the streets of 44 places in Nottingham[1]. The images

[1] http://www.itsnottingham.info/site.htm

are small enough (324×240 pixels) to avoid exposure of personal details.

The server is providing larger images of size 768×576 via Internet with secure access. Since it is practically impossible to upload and rewrite such large size of images with a short interval (e.g. every second), with cooperation of Traffic Control Centre in Nottingham, we have developed a system using many different file names. In our system, the images are posted to 10 different URLs sequentially. The image dispatching system is now designed to provide images from 5 different sites in the city: hence the image files are given as 01a∼01j, 02a∼02j, ..., 05a∼05j. After writing the final file, the system comes back to 01a and overwrites a new image there. The captured image is posted to the URL as soon as it is captured. So, the interval of the capturing and the posting is basically the same, hence the images are overwritten by the new ones with same URLs with the interval of 50 seconds or so.

3 Calibration of Camera Direction

Due to the nature of this imaging system, we need to calibrate the camera direction, i.e. pan and tilt.

3.1 Model Generation

First we make a model using selected images. As is often done in pattern recognition, we use a subspace derived from the principal component analysis. Eigenface is a famous method of this [9].

However, in our case, it is also necessary to make the computational load very light so that we can search a certain area in a real-time processing setting. Thus we will use the aggregated information (sum of the values of pixels for each direction of the axis) of recutangular images.

We will make the model in the following order.

1. Collect the trimmed images of the same area manually from many images.

2. Figure out the principal eigenvectors corresponding to the several largest eigenvalues of the covariance matrix of the trimmed sample images.

and search the sub-image in the test image that is most similar to the feature space defined above. The detail of the technique will be described below.

The trimmed images are expressed as

$$P_m, \; m = 1, \ldots, C \tag{1}$$

$$P_m = \{p_m(i,j) | 1 \le i \le N_1, 1 \le j \le N_2\} \tag{2}$$

Also, the averages of the grey scale to the hotizontal and vertical directions are expressed as $\boldsymbol{\alpha}_m$ and $\boldsymbol{\beta}_m$, respectively. Let $\boldsymbol{\alpha}_m$ and $\boldsymbol{\beta}_m$ be column vectors, and each element is given by

$$\alpha_m(i) = \frac{1}{N_2} \sum_{j=1}^{N_2} p_m(i,j), \quad i = 1, \ldots, N_1$$

$$\beta_m(j) = \frac{1}{N_1} \sum_{i=1}^{N_1} p_m(i,j), \quad j = 1, \ldots, N_2$$

Since the following processing is the same for $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, we will describe only the case of $\boldsymbol{\alpha}$.

Let the covariance matrix of $\{\alpha_m\}$ as

$$M = \frac{1}{C} \sum_{m=1}^{C} (\boldsymbol{\alpha}_m - \bar{\boldsymbol{\alpha}})(\boldsymbol{\alpha}_m - \bar{\boldsymbol{\alpha}})' \tag{3}$$

Note that $\bar{\boldsymbol{\alpha}}$ is the mean

$$\bar{\boldsymbol{\alpha}} = \frac{1}{C} \sum_{m=1}^{C} \boldsymbol{\alpha}_m \tag{4}$$

Next we sort the eigenvalues in the descending order

$$\lambda_1 \ge \lambda_2 \ge \cdots \ge \lambda_C$$

and the corresponding eigenvectors for the eigenvalue λ_k be $\boldsymbol{v}_k$, where each engenvector is normalized to the length 1.

By choosing an appropriate integer r, we have the feature vector space that is a set spanned by linear combination of eigenvectors

$$\boldsymbol{F} = \left\{ [\boldsymbol{v}_1 \; \boldsymbol{v}_2 \; \cdots \boldsymbol{v}_r] \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_r \end{bmatrix} \right\} \tag{5}$$

where $a_i (i = 1, \ldots, r)$ are scalars.

Let us call the orthonormal vectors $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_r$ as the feature vectors. We also define the matrix

$$V = [\boldsymbol{v}_1 \; \cdots \; \boldsymbol{v}_r]$$

$$\boldsymbol{\theta} = [a_1 \; a_2 \; \cdots \; a_r]'$$

for the future reference.

3.2 Matching

We will use the 1-D vectors $\boldsymbol{\alpha}(s,t)$ of the rectangular image X whose (top, left) corner is $(top, left) = (s,t)$.

The projection of $\boldsymbol{\alpha}(s,t)$ onto $\boldsymbol{F}$ is $V\theta$. The optimal projection is given by solving

$$\frac{\partial}{\partial \boldsymbol{\theta}} \|\boldsymbol{\alpha}(s,t) - \bar{\boldsymbol{\alpha}} - V\boldsymbol{\theta}\|^2 = 0 \tag{6}$$

which yields the optimal solution

$$\hat{\boldsymbol{\theta}} = (V'V)^{-1}V'(\boldsymbol{\alpha}(s,t) - \bar{\boldsymbol{\alpha}}) \tag{7}$$

Next, the square distance between $\boldsymbol{\alpha}$ and its projection $V\hat{\theta}$ is given by

$$\begin{aligned} d_\alpha(s,t) &= (\boldsymbol{\alpha}(s,t) - \bar{\boldsymbol{\alpha}})'(I - V(V'V)^{-1}V')' \\ &\times (I - V(V'V)^{-1}V')(\boldsymbol{\alpha}(s,t) - \bar{\boldsymbol{\alpha}}) \end{aligned} \tag{8}$$

where $(I - V(V'V)^{-1}V')'(I - V(V'V)^{-1}V')$ is computable before we get the image $\boldsymbol{\alpha}$. The procedure above is also done for $\boldsymbol{\beta}$ and we can compute $d_\beta(s,t)$.

Note that, by using $d_\alpha(s,t)$ and $d_\beta(s,t)$ we must search (s,t) that gives the minima of both criteria.

Fig. 1 shows the matching result, where the sample image is attached to the area that matched best. Fig. 2 shows the criterion by changing x of the top, where y was fixed to 60. Next we got the criteria by changing y of the left corner where x was the value obtained above as shown in Fig. 3.

We can see that the best value was obtained with a sharp curve near the optimal point. In this way, we don't search the whole space, but we have got a very good result (as we can see in the figure). There is a small disagreement in the right part of the figure, but this is inevitable due to the distortion of the image.

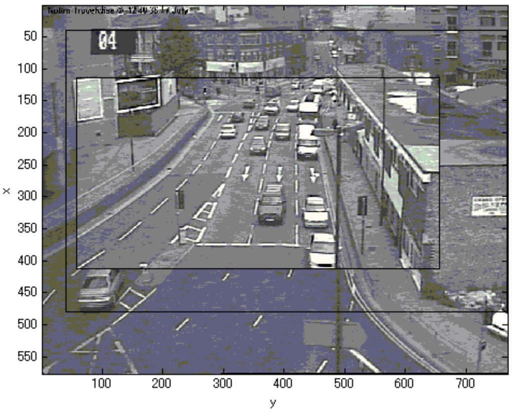


Fig. 1: Result of matching

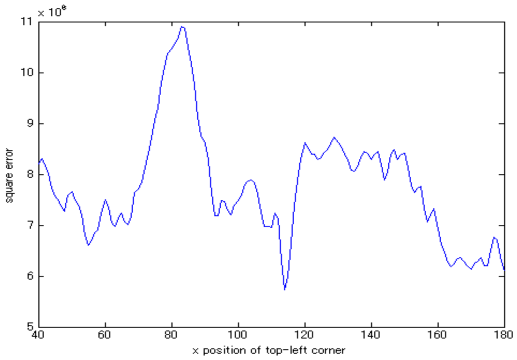


Fig. 2: Errors by moving x

Also, d_x is very insensible to the direction of t with same value s. This is natural, as this vector is averaged to other direction. Hence we can compute almost correctly the optimal coordinate (s,t) by computing separately. The following is such an algorithm.

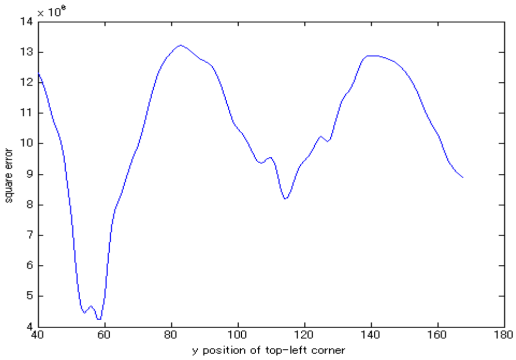


Fig. 3: Errors by moving y

1. Set t an appropriate value and compute.

$$s^* = \arg\max_s d_\alpha(s,t)$$

2. By using the obtained s^* we compute

$$t^* = \arg\max_t d_\beta(s^*,t)$$

4 Car Detection

4.1 State Space Model

The observed images are 2-dimensional where moving cars can be found on it through the observation mechanism.

Now we consider an image at time t which we express as $I(t)$, and the image after calibration of the position by the method in the previous section is expressed as $Y(t)$ which consists of the three colours $Y^c(t)$ where $c \in \{r,g,b\}$.

Next we consider the roads where cars run. The cars run along the roads, and it is better to consider the coordinate along the lane of the roads. For the simplicity of the problem, we consider one lane.

Let $P = (i,j)$ be the position of the image which we consider the farthest of the observable points and $Q = (i,j)$ is the end position of the lane in the image. The lane of the road can be observed by the perspective transformation of the 1-dimensional world coordinate k. Thus we may have a transformation

$$T : k \to (i,j), \quad k = 1,2,\ldots,N \qquad (9)$$

where $T(0) = P$ and $T(N) = Q$. The coordinate on the observation image can be obtained by the perspective transformation for $k = 1,2,\ldots$ where the real length corresponding to the unit value of k is to be decided as $1m$ or so based on physical consideration. Thus we have a sequence of the tuples (i_k, j_k) for $k = 1,2,\ldots,N$.

Here we introduce $\boldsymbol{x}_k(t)$ which is the ground colour at position k at time t and is not necessarily known. The

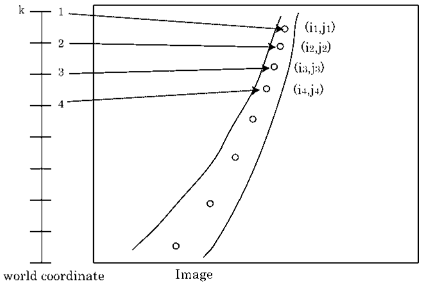


Fig. 4: World coordinate and the image coordinate

colour of the position k can be expressed as a vector $\boldsymbol{x}_k(t)$ where

$$\boldsymbol{x}_k(t) = [x_k^r(t)\ x_k^g(t)\ x_k^b(t)]$$

The relation between the observation on the image and the ground image at the world coordinate is given by

$$\boldsymbol{y}_{T(k)+\gamma(t)}(t) = (1 - b(t))\boldsymbol{x}_k(t) + b(t)\boldsymbol{v}_k(t) \tag{10}$$

where $\gamma(t)$ is a random number around 0 caused by the incomplete calibration of the camera, $b(t)$ is an unknown binary number and $\boldsymbol{v}_k(t)$ is the colour of the occluding object of the place including cars. If there is no car, the value of $\boldsymbol{v}_k(t)$ is nearly zero.

The uncertainty of the position expressed by γ is hard to deal with. So, we will use an expression

$$\boldsymbol{y}_{T(k)}(t) = (1 - b(t))\boldsymbol{x}_k(t) + b(t)\boldsymbol{v}_k(t) + \boldsymbol{r}_k(t) \tag{11}$$

instead of equation (10), where $\boldsymbol{r}_k(t)$ is an observation noise. We can assume that $\boldsymbol{v}_k(t)$ and $\boldsymbol{r}(t)$ are mutually uncorrelated and independent from $\boldsymbol{x}(\tau)$ for any τ.

The ground colour can change from time to time, which is expressed as

$$\boldsymbol{x}_k(t+1) = \boldsymbol{x}_k(t) + \boldsymbol{w}_k(t) \tag{12}$$

where $\boldsymbol{w}_k(t)$ is a vector that usually consists of small values, and the covariance matrix is Q.

The Kalman filter algorithm (e.g.[1]) is given as follows, where the positional index is omitted for the simplicity of the notation.

Initial Condition

$$\hat{\boldsymbol{x}}(0) = \boldsymbol{x}_0 \tag{13}$$

$$\hat{P}(0) = P_0 \tag{14}$$

Recursive Form

$$\bar{\boldsymbol{x}}(t) = \hat{\boldsymbol{x}}(t-1) \tag{15}$$

$$\bar{P}(t) = \hat{P}(t-1) + Q \tag{16}$$

$$K(t) = (1 - b(t))\bar{P}(t)\left[(1 - b(t))\bar{P}(t) + b(t)^2 S + R\right]^{-1} \tag{17}$$

$$\hat{\boldsymbol{x}}(t) = \bar{\boldsymbol{x}}(t) + K(t)\left[\boldsymbol{y}(t) - (1 - b(t))\bar{\boldsymbol{x}}(t)\right] \tag{18}$$

$$\hat{P}(t) = \bar{P}(t) - (1 - b(t))K(t)\bar{P}(t) \tag{19}$$

Note that the relation

$$(1 - b(t))^2 = 1 - b(t) \tag{20}$$

holds for binary $b(t)$ and was used in the above equations.

4.2 Decision of $b(t)$

We will estimate the value of $b(t)$ by the maximum likelihood method. The criterion is given by $p(\boldsymbol{y}(t)|Y^{t-1}, b(t))$, and this value is easily seen to be Gaussian as

$$\begin{aligned} &p(\boldsymbol{y}(t)|Y^{t-1}, b(t) = 0) \\ &\propto \frac{1}{|\bar{P}(t) + R|^{1/2}} \\ &\times e^{\left(-\frac{1}{2}(\boldsymbol{y}(t) - \bar{\boldsymbol{x}}(t))^\top (\bar{P}(t)+R)^{-1}(\boldsymbol{y}(t) - \bar{\boldsymbol{x}}(t)\right)} \end{aligned} \tag{21}$$

and

$$\begin{aligned} &p(\boldsymbol{y}(t)|Y^{t-1}, b(t) = 1) \\ &\propto \frac{1}{|S + R|^{1/2}} e^{\left(-\frac{1}{2}\boldsymbol{y}^\top(t)(S+R)^{-1}\boldsymbol{y}(t)\right)} \end{aligned} \tag{22}$$

where the omitted constant value in (21) and (22) is the same.

The decision is made by

$$\hat{b}(t) = \begin{cases} 0 & \text{if } (21) \geq (22) \\ 1 & \text{if } (21) < (22) \end{cases}$$

where $b(t) = 0$ means the normal state and 1 means there is something, possibily a car.

4.3 Timing in Implementation

The necessary time to download an image from the site via Internet depends on the network environment. In our experiment, it approximately needed 2-3 seconds to download an image and save as a file in a local PC in LAN environment at the university.

This means that, if we get the images 01a$\sim$01j with this order sequentially, the writing speed in the server is faster. Thus, a case may happen where the image of 01c is an image of one-cycle later than the one in 01b and hence there is a discontinuity, for example. If this happens, it becomes very complicated to consider the processing method. Thus we developed a method to wait downloading 01a for the time-stamp is updated. Then the overwriting problem is not likely to happen because downloading images at 01a through 01j takes 20-30 seconds which is far less than the overwriting interval.

The first parameters and the initial values are set by using the observed images of the first iteration.

Since there is a possibility that some cars are included in the images, we use the mean values and the covariances together with the robust estimation. Our policy is to reject the outliers because we consider the outliers due to some cars or their effect.

4.4 Setting Parameters

There are several parameters in the filter. The parameters can be determined by using some identification algorithm, but we will not use such algorithms here because those parameters don't seem unique due to the complicated structure of the model. We set the parameters based on the physical consideration of the model.

4.5 Iteration Processing

When the event appers (i.e. $b(t) = 1$) many times in one cycle, the state estimate is not updated sufficiently, hence the reliability of the values seems to have decreased. There is also a possibility that the state has caught up some wrong values of the data. By considering these phenomena, we make the values of the covariance matrix large, i.e.

$$\hat{P}(t) = d \cdot \bar{P}(t) \tag{23}$$

where we have set the value of $d = 10$. Note that, as time proceeds after this value magnified, it converges to the original stable values as normal state continues.

Figure 5 denotes the estimated value of the background as well as the observed data. The line graph with solid line denotes the observation data and the dashed line denotes the estimated values by the above algorithm.

By appropriately selecting the parameter values of R, Q and S, the estimator works well. Figures 6-9 show some processed results. Fig. 9 shows the final result of this module, where 1 denotes $\hat{b}(t) = 1$ and 0 for otherwise.

5 Conclusions

In this paper, we formulated the car detection problem as the occluding object detection using statistical test between the background image and the object.

Acknowledgment

The authors express their sincere gratitude to Nottingham Traffic Control Centre for developing the image dispatching system and giving us a permission to use the images.

References

[1] B. D. O. Anderson and J. B. Moore, Optimal Filtering, Prentice-Hall, 1979.

[2] R. Cucchiara and M. Piccardi: Vehicle detection under day and night illumination, Proc. of 3rd International ICSC Symposium on Intelligent Industrial Automation, 618-623, 1999.

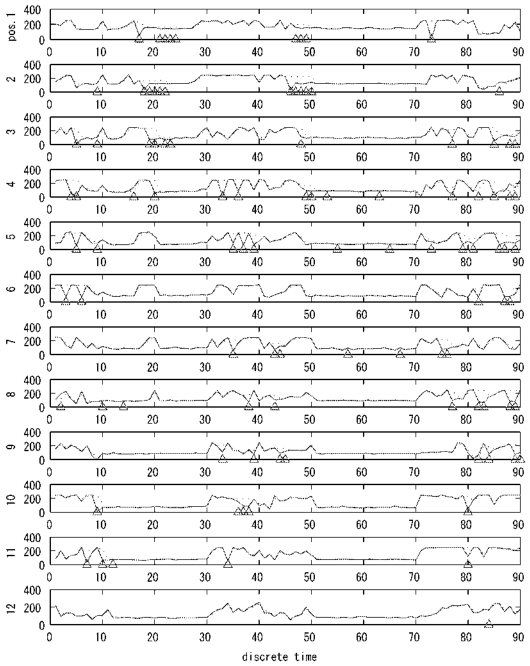


Fig. 5: Processed result (morning)

[3] D. Koller et al.:Towards robust automatic traffic scene analysis in real time, D. Koller, Proc. Int'l Conf. Pattern Recognition, 126-131, 1984.

[4] C. S. Regazzoni, G. Fabri and G. Vernazza (eds): Advanced Video-Based Surveillance Systems, Kluwer Academic Publishers, Boston, 1999.

[5] P. Remagnino et al.: Video-based surveillance systems –Computer vision and distributed processing–, Kluwer Academic Publishers, Boston, 2002.

[6] H. Shimai et al.: Adaptive background estimation by robust statistics, The IEICE Transactions on Information and Systems, Pt.2 (Japanese edition), J86-D-II (6), 796-806, 2003.

[7] M. Tanaka, A. Bargiela, J. Coggan and S. Adachi: Development of traffic image analysing system for mobile terminals using Internet, The Fifth Asia-Pacific Industrial Engineering Management Systems Conference, Dec. 2004, to appear.

[8] M. Tanaka, R. Hamamura and A. Bargiela: Information extraction from traffic images, The 35th ISCIE International Symposium on Stochastic Systems Theory and Its Applications, 41–46, 2004.

[9] M.Turk and A.Pentland: Eigenfaces for recognition, Journal of Cognitive Neuroscience, 3, 71-86, 1991.

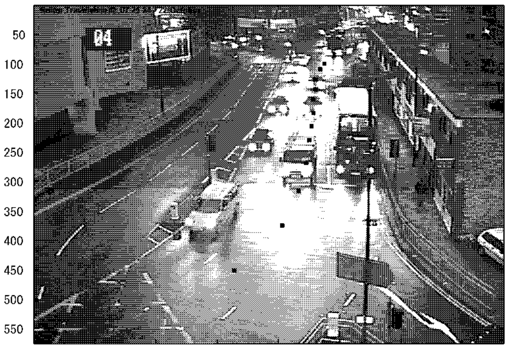


Fig. 6: Observing points (morning)

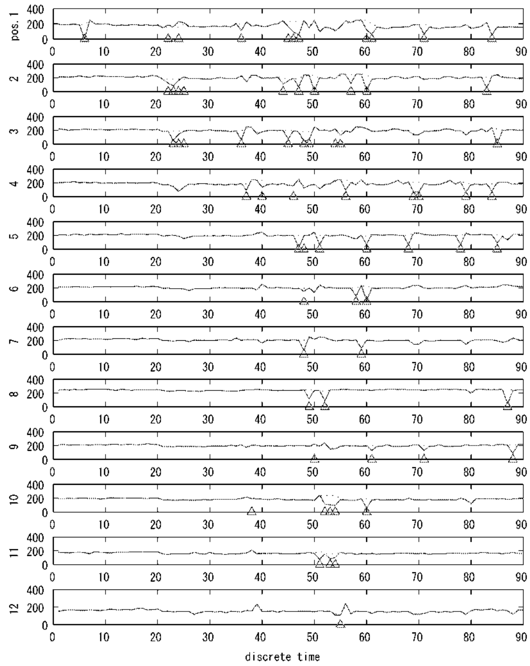


Fig. 7: Processed information (daytime)

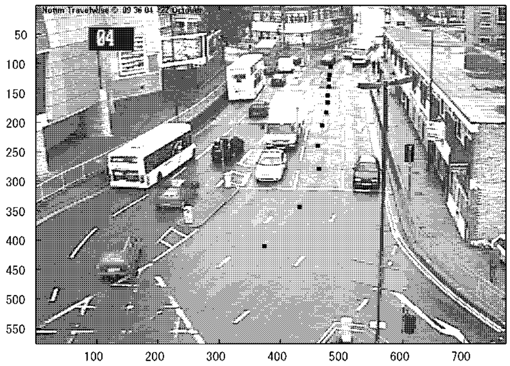
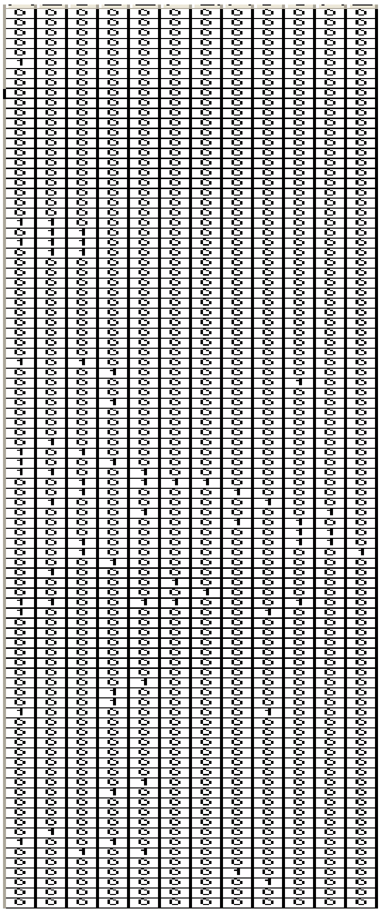


Fig. 8: Observing points (daytime)



0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0	0	0	0
0	1	1	0	0	0	0	0	0	0	0	0
1	1	1	0	0	0	0	0	0	0	0	0
0	1	1	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	1	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0	0	0	0
1	0	0	1	0	0	0	0	0	0	0	0
1	1	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	1	1	0	0	0	0	0
0	0	1	0	0	0	0	1	0	0	0	0
0	1	0	0	0	0	0	0	1	0	0	0
0	0	0	0	1	0	0	0	0	0	1	0
0	0	0	0	0	0	0	1	0	1	0	0
0	0	0	0	0	0	0	0	0	1	1	0
0	0	1	0	0	0	0	0	0	1	1	0
0	0	1	0	0	0	0	0	0	0	0	1
0	0	0	1	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	0	0	0	0	0	0
0	0	0	0	0	0	1	0	0	0	0	0
1	1	0	0	1	1	0	0	0	1	0	0
1	0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	0	0
1	0	0	1	0	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0

Fig. 9: Detected Objects