

# Tracking Persons and Vehicles in Outdoor Image Sequences Using Temporal Spatio-Velocity Transform<sup>\*</sup>

Koichi Sato and J. K. Aggarwal

Department of Electrical and Computer Engineering  
The University of Texas at Austin  
Austin, TX 78712, USA  
{ koh, aggarwaljk }@mail.utexas.edu

## Abstract

*This paper presents a methodology for tracking moving persons and vehicles in outdoor image sequences. A person or a vehicle in the image sequence is treated as one or more continuous objects in the spatio-temporal image cube. Slant cylinders are used to model persons and the vehicles. Our system first detects the motion pixels using a motion subtraction technique and then extracts pixel velocities using a temporal spatio-velocity (TSV) transform. Extracted pixels in the TSV image sequence are fitted to slant cylinders in the spatio-temporal cube. Our system performs well for image sequences containing noise such as tree movement, shadow movement and brightness change. It tracks correctly, smaller and low-speed objects such as humans.*

## 1. Introduction

Tracking persons and objects is an important problem that has received significant attention over the last several decades. Our group has studied various facets of tracking as well as the representation of moving objects. In [1], Jain et al. extracted images of moving objects in dynamic scenes using a differencing operation. In [2], Yalamanchili et al. segmented moving objects using a region-growing process based on the motion differencing operation. More recently, in [3], Cai et al. tracked persons using multiple cameras and a combination of several techniques, including motion analysis of 3D geometry in multiple perspectives, motion detection, segmentation and Bayesian pattern recognition. In [4], Ali et al. segmented and tracked the human body and recognized human activities using the skeletonized human body shape.

Other groups who have worked on human tracking include that of Pentland/Adelson at MIT [5-7], and Davis

et al. at University of Maryland [8,9] as well as several others.

In [5], Oliver et al. segmented humans by subtracting an eigenbackground, which was generated using Principal Component Analysis (PCA) from several static background images. They tracked humans based on their position and velocity, estimated using a Kalman filter and the Probability Density Function (PDF) in each color component of the human blob. In [6], Niyogi segmented humans using Hough transform line detection on a spatio-temporal sliced image of background-subtracted image sequences. They assumed that the subject crosses the field of view at a constant speed; that is, the human trajectories appear as straight lines on the spatio-temporal sliced image. This method is useful for extracting a human trajectory; however, it does not apply to image sequences that contain persons or vehicles moving randomly (stopping and starting), in which case the human trajectories do not appear as straight lines.

In [8,9], Haritaoglu et al. tracked multiple people using silhouette and texture of the human images. They tracked humans by matching the segmented contour of the humans as well as by matching the head position matching. Then, they identified each individual in the group of people using temporal texture templates.

In [10], our previous system segments and tracks persons and recognizes two-person interactions in outdoor side-view image sequences. The system segments the human image using a background subtraction technique followed by an object extraction technique and a temporal spatio-velocity (TSV) transform. Human tracking uses several features such as a temporal texture template, blob size and so on. The TSV transform is a technique to extract velocities of moving pixels from a sequence of binary images. The input sequence is composed of a one-dimensional binary image from which the TSV transform generates a two-dimensional grayscale image sequence that consists of a horizontal axis and a velocity axis.

<sup>\*</sup>This work was supported in part by the Army Research Office under contracts DAAD19-00-1-0044, DAAG55-98-1-0230 and DAAD19-99-1-0012 (Johns Hopkins University subcontract agreement 8905-48168).

Our current system uses the TSV transform to extract two-dimensional velocity vectors from two-dimensional image sequences. Thus, the TSV image, which is the output of the TSV transform, is a 4-dimensional image sequence that consists of horizontal and vertical position axes and horizontal and vertical velocity axes. The system tracks objects by fitting the TSV-xels into spatio-temporal slant cylinder (STSC) models that represent moving objects in the spatio-temporal image cube.

This paper is organized as follows: in section 2, the TSV transform technique is described. In section 3, features in the given image sequences are briefly mentioned. In section 4, methodologies, including motion subtraction and model fitting techniques, are presented. Results and conclusion are described in section 5 and section 6, respectively.

## 2. Temporal Spatio-Velocity Transform

The temporal spatio-velocity transform extracts pixel velocities from sequences of binary images. By grouping similar velocities, the random motion of objects such as tree movement can be eliminated. The TSV extracts the velocities along each axis. When the two-dimensional binary image (based on the horizontal and vertical axes) is used, TSV transform extracts the two-dimensional pixel velocities along the horizontal and vertical axes; therefore, the output TSV image sequence is a four-dimensional image sequence. In general, persons and vehicles move randomly in a two-dimensional plane. Thus, it is necessary to analyze two-dimensional position and two-dimensional velocity. A two-dimensional binary image sequences  $S^*(x,y,n)$  is TSV transformed into four-dimensional TSV image sequences  $V_n(x,y,v_x,v_y)$  as follows.

$$V_n(x,y,v_x,v_y) = \text{TSV}\{S^*(x,y,n)\} \quad (1)$$

where  $n$  is the frame number and  $\text{TSV}\{\cdot\}$  represents the TSV transform operator.

The TSV transform consists of two steps. First, it takes the windowing operation over the binary image sequence, and then it takes the Hough Transform in terms of a line over the windowed image sequence.

The windowed image sequence  $L_n(x,y,n)$  is computed using windowing filter  $F_{n_p}(n)$ ,

$$L_n(x,y,n) = S^*(x,y,n) \cdot F_{n_p}(n) \quad (2)$$

where  $S^*(x,y,n)$  is the binary image in  $n$ th frame,  $n_p$  is the current frame.

An exponential window is used to simplify computation,

$$F_{n_p}(n) = \begin{cases} (1 - e^{-\lambda})e^{\lambda(n-n_p)} & n \leq n_p \text{ (past or current)} \\ 0 & n > n_p \text{ (not defined for future)} \end{cases} \quad (3)$$

The windowed image sequence is Hough transformed in terms of Line A  $(p_x, p_y, v_x, v_y)$  and TSV image  $V_{n_p}(p_x, p_y, v_x, v_y)$  is computed:

Line A:

$$\begin{pmatrix} x \\ y \end{pmatrix} = (n - n_p) \begin{pmatrix} v_x \\ v_y \end{pmatrix} + \begin{pmatrix} p_x \\ p_y \end{pmatrix} \quad (4)$$

$$\begin{aligned} V_{n_p}(p_x, p_y, v_x, v_y) &= \text{Hough}\left\{L_{n_p}(x, y, n)\right\}_{\text{Line A}} \\ &= \sum_n L_{n_p}(v_x(n - n_p) + p_x, v_y(n - n_p) + p_y, n) \end{aligned} \quad (5)$$

where  $n_p$  is the current frame,  $(x,y)$  is the image coordinate,  $(v_x, v_y)$  is the slope of the line (that is, the velocity), and  $(p_x, p_y)$  is the position at  $n=n_p$ , (the position at current frame).

Since an exponential window is used, (5) can be simplified into:

$$V_{n_p}(p_x, p_y, v_x, v_y) = e^{-\lambda} V_{n_p-1}(p - v_x, p - v_y, v_x, v_y) + (1 - e^{-\lambda}) S^*(x, y, n_p) \quad (6)$$

This simple form speeds up computation. Figure 1 shows the iterative operations that generate the TSV image sequence.

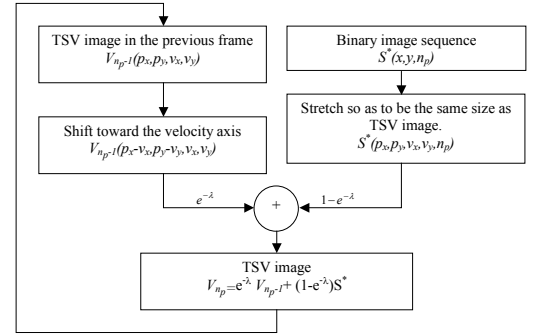


Figure 1. Generating a TSV image sequence

### Time Constant

The TSV image extracts the pixel velocities; however, if the pixels have large acceleration, TSV may fail to extract the velocity. The time constant  $\lambda$  in the TSV transform determines the acceleration range of pixels. When  $\lambda$  is large, it accepts a smaller acceleration range. When  $\lambda$  is small, it accepts the larger acceleration range. In the given dataset, the actual acceleration and velocity do not affect the image equally over the entire image. In other words, two people moving at the same speed may move at different speeds in the image depending on the location along the vertical axis. Therefore, we use different time constant values for different vertical locations in the image. In this system, we set higher time constant value where object movements are large and a lower time constant value where object movements are small. Figure 2 shows the time constant values used for the Dataset2-Camera1. In this picture, the intensity of each image represents the time constant value. (High intensity means a high time constant value). Obviously,

the object movement is larger when the objects come closer to the camera, thus the image shows the lower intensities in the closer part.

For example, there is a tree that always appears in front in the images in dataset2-camera1. The tree blocks our view of any moving object in the data. Thus we set a high time constant value so that the acceleration range is small behind the tree. If the presence of the tree were not available, we would use a constant value in a horizontal location.



**Figure 2. Time Constant Field for Camera2 image**

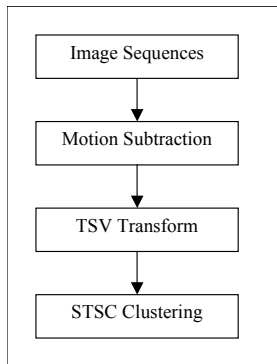
### 3. Description of the image sequences

The images sequences used in this work have the following features:

1. The camera is fixed and distant from the objects.
2. The ground is almost geographically plane, with few ups and downs.
3. Some objects may occlude others, such as trees.
4. Vehicles move and stop.
5. Persons walk randomly in the plane fields.
6. Brightness changes at times.
7. Shadows appear and disappear.
8. Persons walk and ride on bicycles.

### 4. Methodology

The techniques for obtaining the STSC model, are described below, along with the processing order.



**Figure 3. Block diagram of this system**

### Motion Subtraction

Since there are many factors that may contribute to changes in an outdoor image background, such as tree movement or changing shadows and brightness from cloud movement, a simple background subtraction technique may generate significant noise. Thus, the motion subtraction technique is used in our system for segmenting the motion of objects.

$$S(x, y, n) = \begin{cases} 1 & |I(x, y, n) - I(x, y, n - T)| > Th \\ 0 & \text{otherwise} \end{cases} \quad (6)$$

where  $S(x, y, n)$  is a motion detected image in  $n$ th frame,  $I(x, y, n)$  is an image in the  $n$ th frame,  $T$  is a time constant and  $Th$  is a fixed threshold.

This operation simply extracts the moving parts, the object motion and the tree movement. To reduce such noise as a tree movement, our system applies an AND operation for a  $1 \times 3$  block surrounding each pixel.

$$S^*(x, y, n) = S(x, y-1, n) \& S(x, y, n) \& S(x, y+1, n) \quad (7)$$

where  $S^*(x, y, n)$  is motion-extracted image of the noise reduced version and '&' represents the binary 'AND' operation.

### Temporal Spatio-velocity Transform

As detailed in section 2, TSV transforms generate a four-dimensional TSV image sequence.

$$V_n(x, y, v_x, v_y) = \text{TSV} \{S^*(x, y, n)\} \quad (1)$$

### Binarizing the TSV image

To group pixels with similar velocity, we binarized the temporal spatio-velocity image by a specific threshold  $T_v$ .

$$V_{n_p}^*(p_x, p_y, v_x, v_y) = \begin{cases} 1 & \text{if } V_{n_p}(p_x, p_y, v_x, v_y) \geq T_v \\ 0 & \text{otherwise} \end{cases} \quad (8)$$

where  $V_{n_p}^*(p_x, p_y, v_x, v_y)$  is binary TSV image.

### The Slant Cylinder Model

The cylinder model is used to represent objects in the spatio-temporal image cube. Objects in the image sequence form a slant cylinder in the spatio-temporal image cube. The cylinder used in this paper is represented as:

$$C(x, y, n) = \frac{(x - a_x n^2 - v_x n - p_x)^2}{R_x^2} + \frac{(y - a_y n^2 - v_y n - p_y)^2}{R_y^2} \quad (9)$$

where  $C(x, y, n)$  is the cylinder model, the coordinate  $(a_x n^2 + v_x n + p_x, a_y n^2 + v_y n + p_y)$  represents the centroid of this object, and  $(R_x, R_y)$  represents the horizontal and vertical radius of the ellipse fitted to the object.

The objective of this paper is to form the best-fit cylinders that cover the scattered TSV-xels. Hierarchical clustering is used for fitting in every frame. One method for avoiding occlusion errors is to use long data segments in order to track the object before and after the

occlusion. By using slant cylinder models and TSV-xels and ‘best-fitting’ to the cylinder model before and after the occlusion, this system avoids occlusion errors.

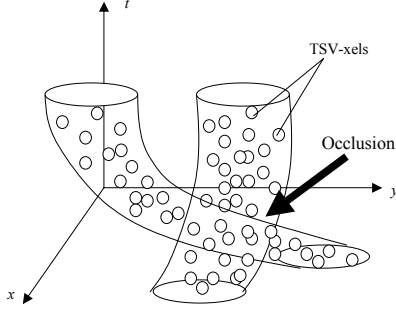


Figure 4. Cylinder models and TSV-xels

### Parameter Computation

The parameters  $a_x, a_y, v_x, v_y, p_x, p_y, R_x, R_y$  are computed by a statistical approach using the coordinates of TSV-xels belonging to the cylinder.

$$\begin{aligned}
 a_x &= \frac{\sigma_t^2 \tau_{t^2} - \tau_{t^2} \tau_{t,x}}{\sigma_t^2 \sigma_t^2 - \tau_{t^2}^2}, a_y = \frac{\sigma_t^2 \tau_{t^2} - \tau_{t^2} \tau_{t,y}}{\sigma_t^2 \sigma_t^2 - \tau_{t^2}^2} \\
 v_x &= \frac{\tau_{t^2} \tau_{t^2,x} + \sigma_t^2 \tau_{t,x}}{\sigma_t^2 \sigma_t^2 - \tau_{t^2}^2}, v_y = \frac{\tau_{t^2} \tau_{t^2,y} + \sigma_t^2 \tau_{t,y}}{\sigma_t^2 \sigma_t^2 - \tau_{t^2}^2} \\
 p_x &= \bar{x} - a_x \bar{t}^2 - v_x \bar{t}, p_y = \bar{y} - a_y \bar{t}^2 - v_y \bar{t} \\
 R_x^2 &= a_x^2 \sigma_t^2 + v_x^2 \sigma_t^2 - \sigma_x^2 \\
 &\quad + 2a_x v_x \tau_{t^2} - 2v_x \tau_{t,x} - 2a_x \tau_{t^2,x} \\
 R_y^2 &= a_y^2 \sigma_t^2 + v_y^2 \sigma_t^2 - \sigma_y^2 \\
 &\quad + 2a_y v_y \tau_{t^2} - 2v_y \tau_{t,y} - 2a_y \tau_{t^2,y}
 \end{aligned} \quad (10)$$

where  $\sigma_k^2$  represents the variance of the variable  $k$ ,  $\tau_{k,l}$  represents the covariance of the variables  $k$  and  $l$ , and  $\bar{k}$  represents the mean value of the variable  $k$ .

Also, the cylinder density  $r$  is defined as:

$$r = \frac{N}{\pi R_x R_y h} \quad (11)$$

where  $h$  represents the height of the cylinder, that is, the time range of the data, and  $N$  represents the number of TSV-xels in the cylinder. The denominator is the cylinder volume.

### Hierarchical Clustering

A hierarchical clustering technique is used to build the spatio-temporal slant cylinder model. Figure 5 shows an overview of the clustering steps.

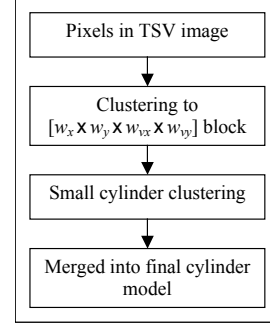


Figure 5. Clustering STSC model procedures

First, the pixels in the binary TSV image are clustered into  $[w_x \times w_y \times w_{yx} \times w_{yy}]$  size blocks, which are equally distributed throughout the entire TSV image field, where  $w_x, w_y, w_{yx}, w_{yy}$  are fixed parameters. This speeds up the computation rather than computing each pixel in the later stage. Second, each block is merged into a small cylinder model that might be a part of the object. Finally, each small cylinder model is merged into the final model. Figure 6 shows the detailed procedure for best fitting to the cylinder models.

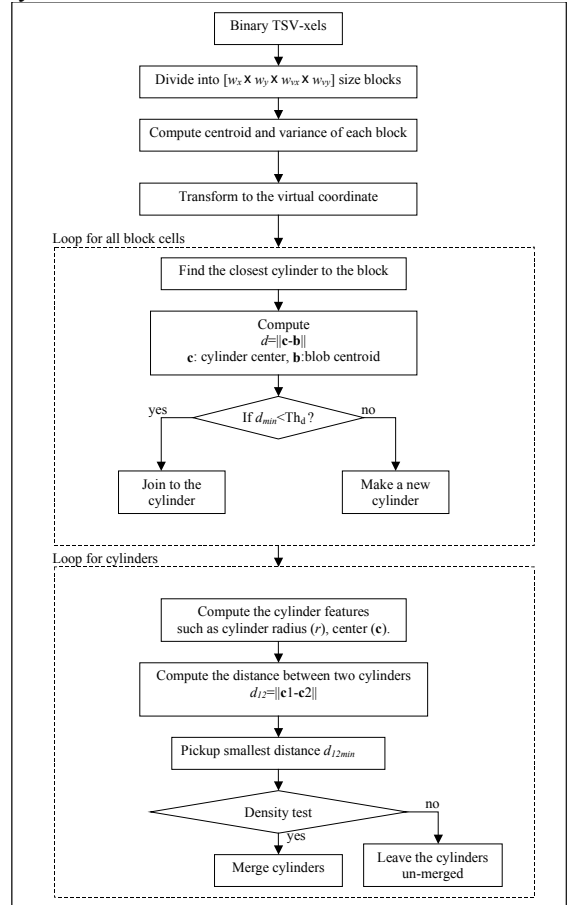


Figure 6. Cylinder model fitting procedure

### Block Segmentation

The four-dimensional  $[w_x \times w_y \times w_{vx} \times w_{vy}]$  TSV image is segmented into a block, and then the parameters of each block (such as centroid coordinate and variance) are computed.

### Coordinate Transformation

The actual position and image location are not linearly related, that is, object size at a far location is smaller than at a close location in the image. Thus, the segmented blocks were transformed into virtual positions that have linear relationships with the actual coordinates.

$$\begin{pmatrix} X \\ Y \end{pmatrix} = \begin{pmatrix} K_1 \frac{x}{y} \\ K_2 \frac{1}{y} \end{pmatrix} \quad (13)$$

where  $(X,Y)$  is the virtual position,  $K_1, K_2$  are constant values and  $(x,y)$  is the coordinate of the image.  $K_1, K_2$  is obtained from the camera parameters such as camera height, focal distance and so on. In this case, they are obtained from the training image sample.

### Cylinder Fitting

Each block is clustered to a small cylinder model under the following rules:

1. Compute the distances between the block and all cylinders.
2. If the minimum distance is less than a fixed threshold  $Th_d$ , join the block to the cylinder, otherwise, make a new cylinder.

The distance between cylinders to the block is computed:

$$d = \frac{(X_B - X_C)^2}{R_1^2} + \frac{(Y_B - Y_C)^2}{R_2^2} \quad (14)$$

where  $d$  is the distance,  $(X_B, Y_B)$  is the block centroid coordinate,  $(X_C, Y_C)$  is a cylinder's coordinate.

### Cylinder Merging

Cylinders are merged under the following rule:

1. Compute the distances between all cylinders.
2. Focus on a cylinder pair whose distance is minimum.
3. Compute the density of a combined cylinder of the two cylinders.
4. If true for the density test, merge the two cylinders, otherwise, leave them un-merged.

The density test is:

$$\frac{r_c}{(r_1 + r_2)/2} > Tr \quad (15)$$

where  $r_c$  is the density of a combined cylinder,  $r_1$  and  $r_2$  is the density of the two cylinders.

### Transform to image coordinate

To display the trajectories of the tracking objects, the following transformation is used:

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} \frac{K_2 X}{K_1 Y} \\ K_2 \frac{1}{Y} \end{pmatrix} \quad (16)$$

All the notation is the same as equation (13).

## 5. Results

We used the dataset 2 (testing camera 1) image sequences with the following specifications.

**Table 1. Specifications**

Size	320x240
Color depth	R(8bit) G(8bit) B(8bit)
Frame rate	29.97 [frame/sec]
Dataset	2 (Testing Camera 1)
OS	MS Windows 98
CPU	Pentium 4, 1GHz
Input Format	MPEG compressed AVI File
Output Format	Uncompressed AVI File
SDK	MS Video For Windows SDK

### Processing Speed

Table 2 gives the processing time for the dataset.

**Table 2. Processing Time**

With Image Output	462 [sec]
Without Image Output	431 [sec]

‘With image output’ means the processing time when we use a program that displays 4 processing images and outputs 1 result images. ‘Without image output’ means the processing time when we use the program that outputs only trajectory coordinates, that is, skips the display procedure. Thus the latter procedure is slightly faster than the first one.

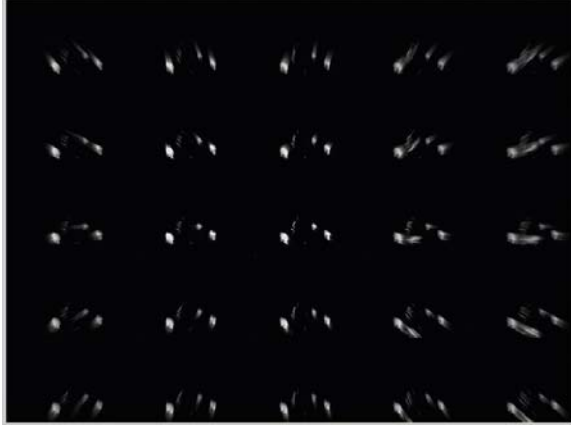
### Tracking Result

Table 3 shows the result of the dataset 2 (Camera1, Testing).

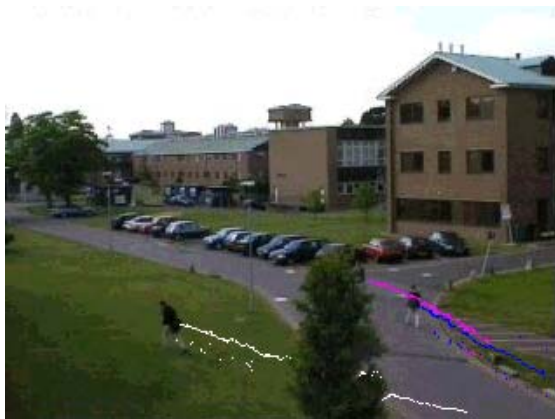
**Table 3. Result of Dataset 2 (Camera1, Testing)**

	Number	Correct	Incorrect
Human	6	6	0
Bicycle	2	1	1
Vehicle	2	0	2

The images in this dataset have a big tree that occludes all other objects, making it difficult to identify the objects around the tree. Our system performs well for smaller and low-speed objects such as a human blob. However, our system was unable to identify a vehicle that stopped near the tree.



**Figure 7. One frame of the TSV image sequence**



**Figure 8. One frame of the result sequence**

The color line after each person represents the trajectory of each person.

## 6. Conclusion

In this paper, we have presented a system that can track humans and vehicles in outdoor image sequences. The system uses temporal spatio-velocity transform technique on image sequences containing noise such as tree movement, shadow movement and brightness change. This system performs well for smaller and low-speed objects such as humans.

## 7. References

- [1] R. Jain, W. N. Martin and J. K. Aggarwal, "Segmentation through the Detection of Changes Due to Motion", *Computer graphics and image Processing II*, pp. 13-33, 1979.
- [2] S. Yalamanchili, W. N. Martin and J. K. Aggarwal, "Extraction of Moving Object Descriptions via Differencing", *Computer graphics and image processing 18*, pp.188-201, 1982
- [3] Q. Cai and J. K. Aggarwal, "Tracking Human Motion in Structured Environments Using a Distributed-Camera System", *IEEE Transactions on pattern analysis and*

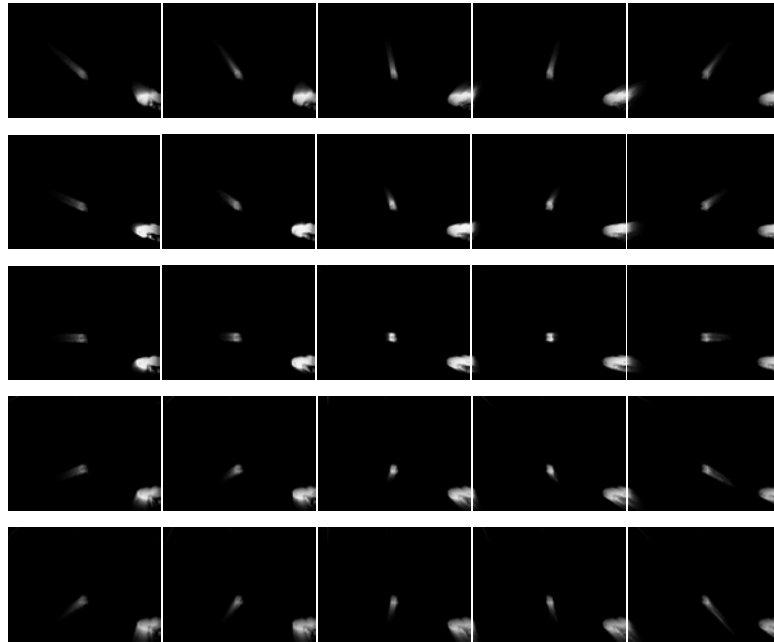
*machine intelligence*, vol. 21, No.12, pp. 1241-1247, November 1999.

- [4] A. Ali and J. K. Aggarwal, "Segmentation and Recognition of Continuous Human Activity", *2001 IEEE Workshop on Events in Video*, July 2001, Vancouver, Canada.
- [5] N. Oliver, B. Rosario and A. Pentland, "A Bayesian computer vision system for modeling human interactions", *Proceedings of International Conference on Vision Systems '99*, Gran Canaria, Spain, pp. 255-272, January 1999.
- [6] S. Niyogi and E. Adelson, "Analyzing gait spatio-temporal surface", *Proceedings of the 1994 IEEE Workshop*, pp.64-69, 1994.
- [7] A. Pentland and A. Liu. "Towards augmented control systems", *IEEE Intelligent Vehicles '95*, Detroit, MI, pp.350-355, September 1995.
- [8] I. Haritaoglu, D. Harwood and L. Davis, "W4: Who, When, Where, What: A real time system for detecting and tracking people", *Third International Conference on Automatic Face and Gesture, Nara*, pp 222-227, April 1998.
- [9] I. Haritaoglu and L. Davis, "Hydra: Multiple people detection and tracking using silhouettes", *IEEE Workshop on Visual Surveillance*, pp.6-13, 1999.
- [10] K. Sato and J. K. Aggarwal, "Tracking and Recognizing Two-person Interaction in Outdoor Image Sequences", *2001 IEEE Workshop on Multi-Object Tracking*, pp.87-94, Vancouver, CA, July, 2001.
- [11] R. Gonzalez and R. Woods, *Digital Image Processing*, Addison Wesley, 1999.
- [12] R. O. Duda, P. E. Hart and D. G. Stork, "Pattern Classification", Second Edition, A Wiley-Interscience Publication, Second Edition, 2001.

## 8. Sample Images



**Figure 9. Original Image**



**Figure 10. TSV image of Figure 7**

All pictures are slices of a TSV image sequence. The center image is a TSV slice image at  $(v_x, v_y)=(0,0)$ , the left-top image is a TSV slice image at  $(v_x, v_y)=(-2,-2)$  [pixel/frame]. All images are arranged in this way. The smaller blob in each image is a human blob and the bigger blob is a vehicle blob. (This picture is compatible to the original image.) We can see that the human blob has the highest intensity at the center picture, and the vehicle blob has the highest intensity at the  $(v_x, v_y)=(-1,-2)$  picture.