An Intelligent Framework for Spatio-Temporal Vehicle Tracking

Shu-Ching Chen[1*], Mei-Ling Shyu[2], Chengcui Zhang[1]

[1]Distributed Multimedia Information System Laboratory
School of Computer Science, Florida International University, Miami, FL 33199
[2]Department of Electrical and Computer Engineering, University of Miami,
Coral Gables, FL 33124

Abstract--**Recently, the Intelligent Transportation Systems (ITS) have been identified as the new paradigm to address the growing mobility problems, and to alleviate congestion and augment the quality of vehicular flow. This paper presents an intelligent and automatic framework for spatio-temporal vehicle tracking in video sequences in order to provide useful information to the traffic control center. The spatio-temporal relationships of the objects are captured automatically via a proposed unsupervised image/video segmentation method and multimedia input strings. A traffic video sequence is used to demonstrate the effectiveness of the proposed framework to identify and track the vehicle objects in the video sequence.**

Index Terms--**Vehicle tracking, spatio-temporal relationships, ITS, ATIS, ATMS.**

I. Introduction

In recent years, Intelligent Transportation Systems (ITS), which integrate advances in telecommunications, information systems, automation, and electronics to enhance the efficiency of existing road networks, have been identified as the new paradigm to address the growing mobility problems, and to alleviate congestion and augment the quality of vehicular flow. ITS are currently addressing several problems that require real-time solutions. Examples are network-wide optimal information provision to users under Advanced Traveler Information Systems (ATIS), and the use of the advanced sensor systems to manage traffic during incidents under Advanced Traffic Management Systems (ATMS) [9, 10, 11, 12]. In order to provide real-time information to drivers under ATIS and ATMS, it is very important for the traffic control center to collect and analyze the most current traffic information. For this purpose, we propose an intelligent framework for spatio-temporal vehicle tracking. Possible applications of this framework are intersection traffic monitoring, incident detection, vehicle queue length determination at signaled street intersections, and intelligent traffic signal control.

The proposed intelligent framework is to analyze the traffic video sequences to identify the vehicle objects and the spatio-temporal relationships of those vehicles. Our emphasis in this paper is on traffic video indexing for spatio-temporal relationships of vehicle objects for vehicle tracking. Multimedia input strings that adopt the notations from regular expressions [8] are proposed to represent the spatio-temporal relations of the vehicle objects.

In order to identify and track the temporal or relative spatial positions of vehicle objects in video sequences, it is necessary to have object-based representation of video data. For this purpose, more and more attention has been devoted to segmenting video frames into regions such that each region, or a group of regions, corresponds to an object that is meaningful to human viewers [4, 5, 6]. While most of the previous works are based on some low level global features such as color histogram and texture features, our video segmentation method focuses on obtaining object level segmentation and obtaining objects in each frame and their traces across the frames. The multimedia input strings are used to capture the temporal and spatial relations of vehicle objects thereafter. The video segmentation method mentioned here is unsupervised. Another advantage is that it uses the segmentation result of the previous video frame to speed up the segmentation process of the current video frame. In this paper, for the specific application domain of the given example query, some techniques of object tracking and post processing will be discussed to facilitate the query process.

In this paper, the proposed intelligent framework is used to process a real-life traffic video sequence. The obtained results clearly demonstrate that our proposed framework can identify vehicle objects and track the spatio-temporal relationships of vehicle objects.

The organization of this paper is as follows. In next section, the multimedia input strings are introduced. Section 3 gives the details of the unsupervised video segmentation method and the related vehicle object tracking technique we used. How the proposed framework is used to estimate the traffic flow is discussed in Section 4. Along with the discussion, an example real-life video sequence is used. Conclusions are presented in Section 5.

II. Multimedia Input Strings

Multimedia input strings are proposed to represent the spatio-temporal relations of vehicle objects in video sequences. Each symbol of a multimedia input string contains one or more objects that are enclosed by parentheses and are in one image or video frame.

* This research was supported in part by NSF CDA-9711582.

Figure 1 shows two example video frames. Assume there are three objects that are the *ground*, *car*, and *bus* and are represented by **G**, **C**, and **B**, respectively. Each vehicle object is surrounded by a minimal bounding rectangle. The multimedia input string for Figure 1 is:

$$\underbrace{(G_1 \,\&\, C_{13} \,\&\, B_{19})}_{X_1}\underbrace{(G_1 \,\&\, C_{13} \,\&\, C_4 \,\&\, B_{25})}_{X_2}$$

Input symbols X_1 and X_2 represent frame 1 and frame 10. From frame 1 to frame 10, one *car* object (the upper left one) moves to the left edge of the intersection area, and the *bus* object moves to the bottom right while another *car* object moves into the main area of that intersection. Here, the subscripted numbers in the multimedia input strings indicate the relative spatial relations of the vehicle objects (as shown in Table 1). Table 1 shows part of the twenty-seven relative spatial relations. The complete twenty-seven relative spatial relations are defined in [2]. For example, G_1 indicates that the object *ground* is chosen as the target object, C_{13} means that the vehicle object *car* is above and to the left of the *ground*, B_{19} represents the *bus* is to the right of the *ground*, and so on. B_{25} in the second input symbol means the relative position of the bus changes from right to bottom right, and C_4 in the second input symbol says that a new *car* object appears above the *ground*.

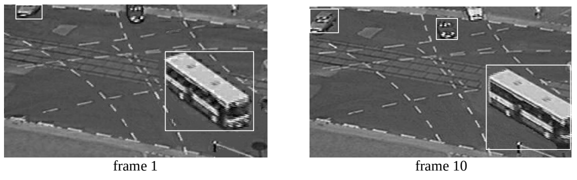


Figure 1: Two example video frames with three objects: *ground, car and bus*. The vehicle object *bus* moves to the bottom right, and one vehicle object *car* moves into the intersection area in frame 10.

Table 1: Part of the three-dimensional relative positions for objects: The first and the third columns indicate the relative position numbers while the second and the fourth columns are the relative coordinates. (x_t, y_t, z_t) and (x_s, y_s, z_s) represent the X-, Y- and Z-coordinates of the target and any vehicle object, respectively. The "≈" symbol means the difference between two coordinates is within a threshold value.

Num.	Relative Coordinates	Num.	Relative Coordinates
1	$x_s \approx x_t, y_s \approx y_t, z_s \approx z_t$	4	$x_s \approx x_t, y_s < y_t, z_s \approx z_t$
13	$x_s < x_t, y_s < y_t, z_s \approx z_t$	19	$x_s > x_t, y_s \approx y_t, z_s \approx z_t$
25	$x_s > x_t, y_s > y_t, z_s \approx z_t$		

III. Video Segmentation and Object Tracking

The spatio-temporal relations of the vehicle objects in the video sequence must be captured in an efficient way, which are indexed by multimedia input strings. As shown in Figure 1, there are vehicle objects such as *car*, *bus* in each video frame with their relative positions roughly represented by the minimal bounding boxes together with their centroids. This kind of information can be obtained by using our unsupervised video segmentation method. Furthermore, by applying the proposed object tracking technique, we can find out the trace tubes for vehicle objects, which enable our framework to provide useful and accurate traffic information for ATIS and ATMS.

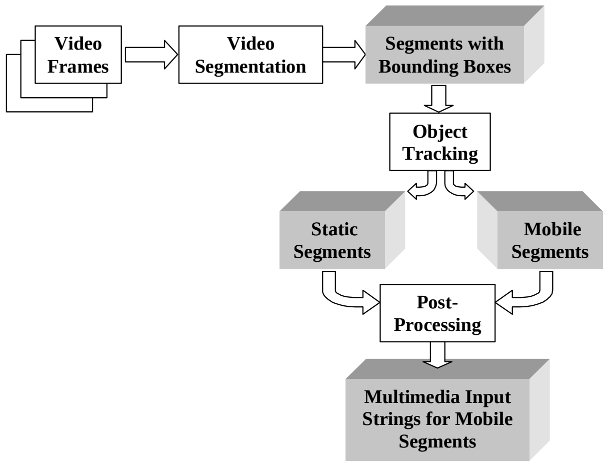


Figure 2: The outline of the proposed framework.

In this paper, a real-life traffic video sequence has been used to demonstrate the power and potential of the proposed framework. Our goal is to try to identify the moving vehicles and track their trace tubes. Therefore, the proposed framework can be applied to traffic applications such as "Find the traffic flow of the shown intersection area in the given traffic video sequence." The outline of this framework is described in Figure 2.

As shown in Figure 2, first, do segmentation on each traffic video frame and obtain all the extracted segments as well as their bounding boxes. Secondly, apply some object tracking technique to group the segments into '*static segments*' and '*mobile segments*' respectively. Then, using this information and multimedia input strings to index the spatial relations of the vehicle objects (the *mobile segments* such as the cars and buses) and their bounding boxes. It needs to be noted that the proposed segmentation method can only identify vehicle objects but it cannot tell whether these vehicle objects are *cars* or *buses*. Therefore, pre-knowledge (sizes, lengths, etc.) of different vehicle classes should be provided. In the following subsections, the proposed framework will be described step by step.

A. *Unsupervised Video Segmentation Method (SPCPE)*

The SPCPE (Simultaneous Partition and Class Parameter Estimation) algorithm is an unsupervised video segmentation method to partition video frames. A given class description determines a partition. Similarly, a given partition gives rise to a class description, so the partition and the class parameter have to be estimated simultaneously. In practice, the class descriptions and their parameters are not readily available. An additional difficulty arises when images have to be partitioned automatically without the intervention of the user. Thus, we do not know a priori which pixels belong to which class. In the SPCPE algorithm, the partition and the class parameters are treated as random variables. The method for partitioning a video frame starts with an arbitrary partition and employs an iterative algorithm to estimate the partition and the class parameters jointly [3, 14]. Since the successive frames in a video do not differ much, the partitions of adjacent frames do not differ significantly. Each frame is partitioned by using the partition of the previous frame as an initial condition so the number of iterations in processing can be greatly reduced. A randomly generated initial partition is used for the first frame since no previous frame is available.

The mathematical description of a class specifies the pixel values as functions of the spatial coordinates of the pixel. The parameters of each class can be computed directly by using a least square technique. Suppose we have two classes. Let the partition variable be $c = \{c_1, c_2\}$ and the classes be parameterized by $\theta = \{\theta_1, \theta_2\}$. Also, suppose all the pixel values y_{ij} (in the image data Y) belonging to class k $(k=1,2)$ are put into a vector $\boldsymbol{Y_k}$. Each row of the matrix Φ is given by $(1, i, j, ij)$ and a_k is the vector of parameters $(a_{k0}, \ldots, a_{k3})^T$.

$$y_{ij} = a_{k0} + a_{k1}i + a_{k2}j + a_{k3}ij, \ \forall(i,j) \ y_{ij} \in \boldsymbol{c_k}$$

$$\boldsymbol{Y_k} = \Phi\, a_k$$

$$\hat{a}_k = (\Phi^T\Phi)^{-1}\Phi^T\,\boldsymbol{Y_k}$$

We estimate the best partition as that which maximizes the a posteriori probability (MAP) of the partition variable given the image data Y. Now, the MAP estimates of $c = \{c_1, c_2\}$ and $\theta = \{\theta_1, \theta_2\}$ are given by

$$(\hat{c},\hat{\theta}) = Arg\max_{(c,\theta)} P(c,\theta \mid Y)$$

$$= Arg\max_{(c,\theta)} P(Y \mid c,\theta)\,P(c,\theta)$$

Let $J(c,\theta)$ be the functional to be minimized. With appropriate assumptions, this joint estimation can be simplified to the following form:

$$(\hat{c},\hat{\theta}) = Arg\min_{(c,\theta)} J(c_1,c_2,\theta_1,\theta_2)$$

$$J(c_1,c_2,\theta_1,\theta_2) = \sum_{y_{ij}\in c_1} -\ln p_1(y_{ij};\theta_1) + \sum_{y_{ij}\in c_2} -\ln p_2(y_{ij};\theta_2)$$

It may appear as though the simultaneous minimization of J on c and θ is hard, but it is not. We will show how the minimization of J can be carried out alternately on c and θ in an iterative manner. Let $\hat{\theta}(c)$ represent the least squares estimates of the class parameters for a given partition $\boldsymbol{c}$. The final expression for $J\left(c,\hat{\theta}(c)\right)$ can be derived easily and is given by

$$J\left(c,\hat{\theta}(c)\right) = Arg\min_{(c_1,c_2)}\left\{\frac{N_1}{2}\ln\hat{\rho}_1 + \frac{N_2}{2}\ln\hat{\rho}_2\right\}$$

where $\hat{\rho}_1$ and $\hat{\rho}_2$ are the estimated model error variances of the two classes.

The algorithm starts with an arbitrary partition of the data and computes the corresponding class parameters. With these class parameters and the data, a new partition is estimated. Both the partition and the class parameters are iteratively refined until there is no further change in them.

After the segmentation, the minimal bounding rectangle (MBR) concept in R-tree [7] is adopted so that each vehicle object is bounded by a rectangle (as shown in Figure 1). Moreover, the centroid point of each vehicle object is mapped to a point object for spatial reasoning.

B. *Object Tracking*

After video segmentation, the segments (objects) with their bounding boxes and centroids are extracted from each frame. In order to answer the traffic flow query mentioned earlier, we must first have the ability to track the vehicle objects (segments) within the successive video frames [1]. Note that the partition of the previous frame was used as the initial condition in partitioning the current frame. Intuitively, we connect two segments that are spatially the closest in the adjacent frames. Euclidean distance is used here to measure the distance between their centroids.

Definition 1: A bounding box B (of dimension 2), will be defined by the two endpoints S and T of its major diagonal [13]:

B=(S, T), where S=$[s_1, s_2]$ and T=$[t_1,t_2]$ and $s_i \le t_i$ for i=1,2. Due to Definition 1, the area of B: $Area_B=(s_2-s_1)\times(t_2-t_1)$.

Definition 2: The centroid ctd_B of a bounding box B corresponding to an object O is defined as following:

$ctd_O=[ctd_{O1}, ctd_{O2}]$, where

$ctd_{O1}= \left(\sum_{i=1}^{No} O_{xi}\right)/No$; $ctd_{O2}= \left(\sum_{i=1}^{No} O_{yi}\right)/No$; where

No is the number of pixels belonging to object O within bounding box B; and O_{xi} represents the x-coordination of the ith pixel in object O while O_{yi} represents the y-coordination of the ith pixel.

Let ctd_M and ctd_N be the centroids of segments M and N that exist in consecutive frames respectively, and δ be a threshold. The Euclidean distance between them should not exceed the threshold δ if M and N represent the same object in consecutive frames:

$$DIST(ctd_M - ctd_N) = \sqrt{(ctd_{M1}-ctd_{N1})^2 + (ctd_{M2}-ctd_{N2})^2} \le \delta$$

Besides using Euclidean distance, we also apply some size restriction to the process of object tracking. If two segments in successive frames represent the same object, the difference between their sizes should not be large. Moreover, in order to answer the example query, only those objects such as the moving vehicles in each video frame are important for this purpose. Figure 3 gives the segmentation result for frame 3, all of the gray segments in Figure 3(b) are supposed to be the vehicle objects we want to extract from frame 3. But the problem is: there are lots of segments not corresponding to moving vehicles. For example, the traffic lines across the intersection area are identified as many separated small segments distributed throughout the area. Also, some small patches on the ground are also identified as dotted segments. For this kind of small segments, we can use the pre-knowledge such as the usual size scale of vehicles to ignore them, which means the segmentation result do not generate bounding boxes for them so that they will not be considered in object tracking process. Figure 3(c) shows the result after filtering small segments in frame 3.

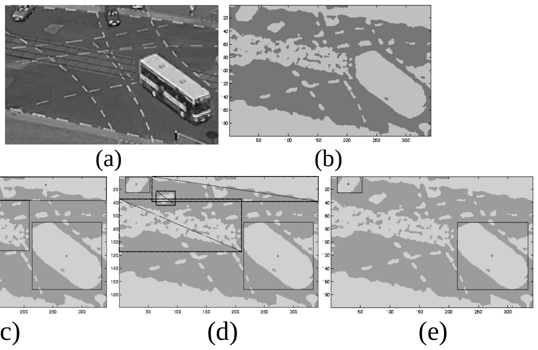


Figure 3: (a): the original video frame 3; (b): the segmentation result for (a); (c): the segments and bounding boxes after filtering small objects; (d): the segments with bold border and diagonals are identified as 'static segments'; (e): the segments and bounding boxes after filtering '*static segments*'.

Another problem is, as shown in Figure 3(c), there still exist some objects that are not corresponding to moving vehicles such as the tiled ground, roadsides and some patches on the ground. Since the location of the camera monitoring this intersection area is fixed above the ground, the centroids of those static segments (tiled ground, patches, etc.) should remain static throughout the video sequence. That is why we can distinguish this kind of '*static segments*' from those '*mobile segments*' (moving vehicles) in this application domain. Before discussing the steps and details of object tracking technique used here, another definition should be mentioned.

Definition 3: The trace tube T of a segment/object S with bounding box B is defined as:

$T_S=\{(ctd_{S1}^{i}, ctd_{S2}^{i}), (ctd_{S1}^{i+1}, ctd_{S2}^{i+1}), \ldots, (ctd_{S1}^{k}, ctd_{S2}^{k})\}$ where i is the first frame where S appears and k is the last frame where S exists.

Steps of object tracking:

Step 1: For each segment S in each frame,

```
    if ($Area_S < \delta_{Area}$) then
        skip this segment and go to next segment;
        repeat step1;
    else
        generate bounding box $B_s$ and centroid $ctd_s$ for segment $S$.
end if;
```

Step 2: After step1, only those segments/objects with reasonable size could be assigned bounding boxes and centroids as can be seen from Figure 3(c). Start from the first frame:

For each segment S with bounding box B_s and centroid ctd_s in current frame i,if there exists a segment S_r in next frame which satisfies:

```
    ($DIST(ctd_S, ctd_{Sr}) \le \delta$ ) and ($|Area_{Bs}-Area_{Bsr}| \le \delta_A$)
    then
        mark $S$ and $S_r$ as related segments;
        add ($ctd_{Sr1}^{i}, ctd_{Sr2}^{i}$) to the trace tube $T_S$;
```

then go to next frame i+1 and repeat the same process to find out all the trace tubes as far as possible for all the segments.

Step 3: For each trace tube T_S corresponding to segment S,

```
    $Max\_ctd_{S1} = \max\{ (ctd_{S1}^{j}) \mid (ctd_{S1}^{j}, ctd_{S2}^{j}) \in T_S \}$;
    $Min\_ctd_{S1} = \min\{ (ctd_{S1}^{j}) \mid (ctd_{S1}^{j}, ctd_{S2}^{j}) \in T_S \}$;
    $Max\_ctd_{S2} = \max\{ (ctd_{S2}^{j}) \mid (ctd_{S1}^{j}, ctd_{S2}^{j}) \in T_S \}$;
    $Min\_ctd_{S2} = \min\{ (ctd_{S2}^{j}) \mid (ctd_{S1}^{j}, ctd_{S2}^{j}) \in T_S \}$;
    if (($Max\_ctd_{S1}- Min\_ctd_{S1}) \le \delta_1$) and
      (($Max\_ctd_{S2}- Min\_ctd_{S2}) \le \delta_2$)
        segment S is marked as 'static segment';
    else
        segment S is marked as 'mobile segment';
    end if;
```

Once finish step 3, the groups of '*static segments*' and '*mobile segments*' can be identified as shown in Figure 3(d), where the segments with bold border and diagonal in its bounding boxes are identified as '*static segments*'.

C. *Post Processing*

Based upon the object tracking results, the post processing on the segmentation results becomes relatively simple. In this step, two main tasks will be done:

- Eliminating the bounding boxes and centroids of those 'static segments' in each video frame.
- For the remained 'mobile segments', generate their corresponding multimedia input strings for each frame.

Figure 3(e) shows the segmentation result for frame 3 after object tracking and post processing. Now the result is clean and is able to focus on the moving vehicle objects. We put the details of multimedia input strings for this video sequence and test results in Section 4.

IV. SPATIO-TEMPORAL VEHICLE TRACKING EXAMPLES

A traffic video sequence is used to give an example of spatio-temporal vehicle tracking using the proposed framework. The original video sequence is downloaded from the website of KOGS/IAKS Universitat Karlsruhe [15]. The example traffic video consists of 50 frames with each frame of size 512 rows×512 columns. It is a grayscale video that shows the traffic flow of a road intersection within a time duration. As Figure 4 shows, since our focus is mainly on the intersection area of the video frames, the center part (intersection area) of each video frame is selected for segmentation purpose. A small portion of the traffic video is used to illustrate how the proposed framework can be applied to traffic applications such as “Estimate the traffic flow of this road intersection from 8:00 AM to 8:15 AM.” This requires the ability to obtain the information of the number of vehicles passing through the intersection in a given time duration as well as the types of the vehicles (such as “car”, “bus”, etc.).

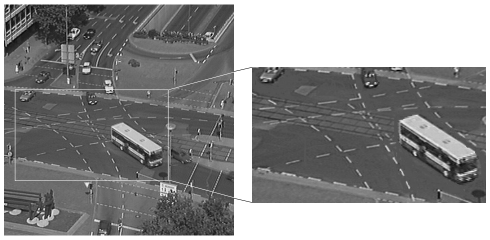
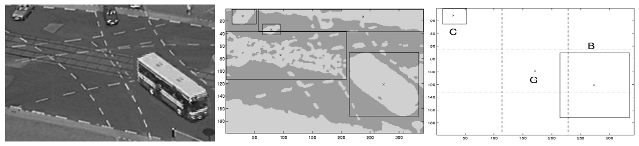


Figure 4: Traffic video frame 3 and the intersection area.

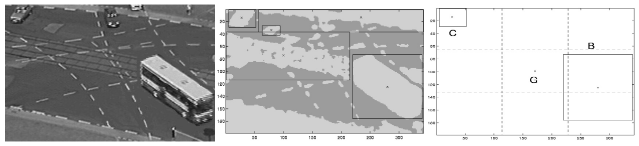
The proposed video segmentation method is applied to these video frames by considering two classes. The first frame is partitioned with two classes. After obtaining the partition of the first frame, we compute the partitions of the subsequent frames using the previous partitions as the initial partition of the subsequent video frames since there is no significant difference between consecutive frames. From the segmentation results, a few frames – 3, 5, 8, 10 and 11 – are shown in Figure 5 along with the original frames adjacent to them. In Figure 5, video frames in left column are the original frames. Center column shows the segments extracted from the video frames, and right column shows the bounding boxes of the vehicle objects in the video frames. As can be seen from Figure 5, a single class captures all the vehicles, the traffic lines, the roadsides and the tiled part of the ground as well as some patches. Most part of the ground is captured by another class. Some of the vehicles have been combined with other objects into a single segment since they are too close together. For example, in frames 3, 5 and 8, the dark gray car on the top center of the frame overlapped with the roadsides. Notice that in frames 10 and 11, the dark gray car moved into the main area of the intersection and separated from the roadsides so that it can be identified as an individual object. Also, the gray car on the top left and the big bus on the bottom right are successfully identified in all of these frames.

Since only the vehicles are important for this application, we use the right column in Figure 5 to simplify the segments for each frame. Notice that the bounding boxes for those segments corresponding to roadsides, tiled ground and patches are eliminated, and only the bounding boxes of those vehicle segments are left. The way to do that is by applying the object tracking and post processing technique mentioned above to track the related segments in consecutive video frames. For example, in this case, since the positions and sizes of those non-vehicle objects (roadsides, patches and tiled ground) keep the same throughout the video sequence, it is easy for us to track them and find out the fact that these objects are ‘*static segments*’ compared to those ‘*mobile segments*’ such as moving vehicles, so that these segments can be eliminated in video indexing.

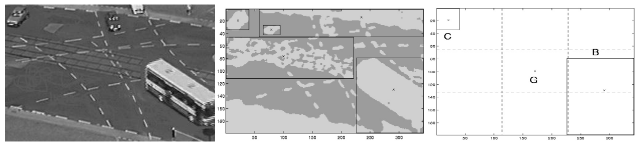
For the simplified segmentation results (right column of Figure 5), we use symbolic representations (multimedia input strings) to represent the spatial relationships of the objects in each frame. As shown in the right column of Figure 5, the ground (**G**) is selected as the target object and the segments are denoted by **C** for cars or **B** for buses. Each



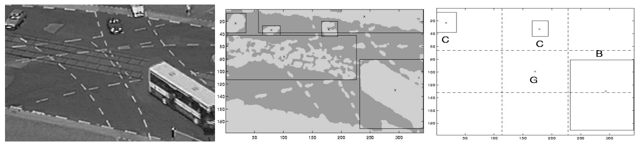
frame 3 *multimedia input string*: $\boldsymbol{G_1 \& C_{13} \& B_{19}}$



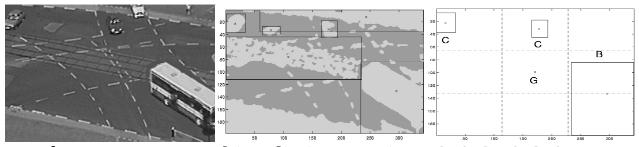
frame 5 *multimedia input string*: $\boldsymbol{G_1 \& C_{13} \& B_{19}}$



frame 8 *multimedia input string*: $\boldsymbol{G_1 \& C_{13} \& B_{19}}$



frame 10 *multimedia input string*: $\boldsymbol{G_1 \& C_{13} \& C_4 \& B_{19}}$



frame 11 *multimedia input string*: $\boldsymbol{G_1 \& C_{13} \& C_4 \& B_{25}}$

Figure 5: Segmentation results as well as the multimedia input strings for frames 3, 5, 8, 10 and 11. Left column gives the original video frames, center column shows the segments extracted from the video frames, and right column shows the bounding boxes of the vehicle objects.

segment is indexed in a multimedia input string based on the spatial relation of its centroid. As mentioned earlier, the dark gray car (on the top center) is put into a single segment only when it is separated from the roadside. As shown in the multimedia input strings in frames 3, 5 and 8, there are two vehicles passing through the intersection. We use subscript numbers to indicate the relative spatial relations of the vehicle objects with respect to the target object from the viewer’s perspective. $\boldsymbol{G_1}$ indicates that the ground (**G**) is the target object; $\boldsymbol{C_{13}}$ means that the gray car is above and to the left of **G,** and $\boldsymbol{B_{19}}$ means that the bus is to the right of **G**. In frame 10, the dark gray car is identified as an individual segment and indexed by $\boldsymbol{C_4}$. In frame 11, the centroid of bus moved from the subregion to the right of **G** to the subregion below and to the right of **G** so that its index symbol changes from $\boldsymbol{B_{19}}$ to $\boldsymbol{B_{25}}$. As described above, it can be seen that the multimedia input strings can model not only the number of objects, but also the relative spatial relations. In this case, in order to estimate the intersection traffic flow, we need a 'judge line' in the frame to determine the traffic flow of specified direction, which can be given by users. For example, the 'judge line' could be the line before the vehicles going into or out of the intersection area. By using the information of centroid's position of each object, we can roughly determine the traffic flow of specified direction in the intersection area. Moreover, since the types of vehicles are also important for estimating the traffic flow, we can use the size of the bounding box to roughly determine that vehicle's type (such as 'car' and 'bus').

V. Conclusions and Future Work

In this paper, an intelligent framework for spatio-temporal vehicle tracking is presented. This proposed intelligent framework can help traffic control center to collect and analyze the most current traffic information for ATIS and ATMS by incorporating unsupervised image/video segmentation and object tracking techniques. The spatio-temporal relationships of the vehicle objects are captured via the image/video segmentation method automatically and modeled by the multimedia input strings. Furthermore, object tracking and post processing techniques have been explored to track the desired vehicle objects while eliminating other objects that are not important for this application. A real-life traffic video sequence is used to illustrate the effectiveness of the proposed framework.

However, there are some possible future work can be pursued with the proposed framework. One of the most possible ways to improve this framework is to adopt the idea of user information feedback. For example, users can help to determine which objects they are interested in and which are not. That would be very flexible. Another direction is to improve the unsupervised video segmentation method so that it can capture more accurate information of the spatial relations of the semantic objects. Furthermore, applying this framework to other applications such as incident detection, vehicle queue length determination at signaled street intersections, and intelligent traffic signal control will be studied.

References:

[1] S-C. Chen, M-L. Shyu, C. C. Zhang, and R. L. Kashyap, "Object Tracking and Augmented Transition Network for Video Indexing and Modeling," *12th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2000)*, pp. 428-435, November 13-15, 2000, Vancouver, British Columbia, Canada.

[2] S-C. Chen and R. L. Kashyap, "A Spatio-Temporal Semantic Model for Multimedia Presentations and Multimedia Database Systems," accepted for publication in *IEEE Transactions on Knowledge and Data Engineering*, 1999.

[3] S-C. Chen, S. Sista, M-L. Shyu, and R. L. Kashyap, "An Indexing and Searching Structure for Multimedia Database Systems," *IS&T/SPIE conference on Storage and Retrieval for Media Databases 2000*, pp. 262-270, January 23-28, 2000, San Jose, CA, U.S.A.

[4] J.D. Courtney, "Automatic Video Indexing via Object Motion Analysis," *Pattern Recognition*, vol. 30, no. 4, pp. 607-625, 1997.

[5] L.X. Fan and K.K. Sung, "Model-Based Varying Pose Face Detection and Facial Feature Registration in Video Images," *Proc. 8th ACM International Conference on Multimedia*, pp. 295-302, 2000.

[6] A.M. Ferman, B. Gunsel, and A.M. Tekalp, "Object Based Indexing of MPEG-4 Compressed Video," in *Proc. SPIE: VCIP*, pp. 953-963, vol. 3024, San Jose, USA, February 1997.

[7] A. Guttman, "R-tree: A Dynamic Index Structure for Spatial Search," in *Proc. ACM SIGMOD*, pp. 47-57, June 1984.

[8] S.C. Kleene, "Representation of Events in Nerve Nets and Finite Automata, Automata Studies," Princeton University Press, Princeton, N.J., pp. 3-41, 1956.

[9] H. S. Mahmassani, T. Hu, S. Peeta, and A. Ziliaskopoulos, "Development and testing of dynamic traffic assignment and simulation procedures for ATIS/ATMS Applications," Technical Report DTFH61-90-C-00074-FG, Center for Transportation Research, The University of Texas at Austin, 1994.

[10] S. Peeta, "System Optimal Dynamic Traffic Assignment in Congested Networks with Advanced Information Systems," Ph.D. Dissertation, The University of Texas at Austin, 1994.

[11] S. Peeta, and H. S. Mahmassani, "System Optimal and User Equilibrium Time-dependent Traffic Assignment in Congested Networks," *Annals of Operations Research*, pp. 81-113, 1995.

[12] S. Peeta, and H. S. Mahmassani, "Multiple User Classes Real-time Traffic Assignment for Online Operations: A Rolling Horizon Solution Framework," *Transportation Research*, Vol. 3, No. 2, pp. 83-98, 1995.

[13] N. Roussopoulos, C. Faloutsos, and T. Sellis, "Nearest Neighbor Queries," *Proc. ACM SIGMOD Intl. Conf. on Management of Data*, pp. 71-79, 1995.

[14] S. Sista and R. L. Kashyap, "Unsupervised video segmentation and object tracking," in *IEEE Int'l Conf. on Image Processing*, 1999.

[15] http://i21www.ira.uka.de/cgi-bin/download?bad.