Incremental Vehicle 3-D Modeling from Video

N. Ghosh and B. Bhanu
Center for Research in Intelligent Systems (CRIS), University of California, Riverside, CA 92521, USA
{nirmalya, bhanu} @ ee.ucr.edu

Abstract

In this paper, we present a new model-based approach for building 3-D models of vehicles from color video provided by a traffic surveillance camera. We incrementally build 3D models using a clustering technique. Geometrical relations based on 3D generic vehicle model map 2D features to 3D. The 3D features are then adaptively clustered over the frame sequence to incrementally generate the 3D model of the vehicle. Results are shown for both simulated and real traffic video. They are evaluated by a new structural performance measure underscoring usefulness of incremental learning.

1. Introduction

Present traffic surveillance systems depend on license plate extraction [1], which is not robust to illumination variations. Static uncalibrated video camera that watches moving vehicles provides different views in a partially redundant manner that has the potential for incremental 3D modeling of vehicles from a frame sequence.

Previous research in this field has focused on vehicle detection and tracking using PCA, neocognitron, and eigen-learning [3] and alignment-based 2D matching and tracking [5, 6, 7, 8]. But 2D-to-3D projection makes view-invariant vehicle recognition a challenging task. Some research in this direction includes unsupervised learning of scale-invariant local features from 2D structures, stereovision setup [4], neural networks, and several other similar methods. Aerial "image"-based 3D modeling approaches for buildings [9] considers nearly top-view and avoids depth-computation (as required for 3D model building). But the rich information in the form of inter-frame view-relations in video-data has not been utilized. In most cases it has been assumed that the *complete* vehicle is visible at different orientations, which is not the case for traffic videos. Hence the real applications need incremental 3D model learning over the frame-sequence in the face of *partial* visibility of the vehicle. The parameters of a generic vehicle model can be incrementally learned for the current vehicle instance [2].

The proposed approach estimates frame-based 3D features of the partially seen vehicle in the present frame, adaptively cluster the *same* features over frame-sequence seen till that time point and incrementally learn the parameters of a 3D generic model (for the particular vehicle instance in view). The estimated 3D model can be used for vehicle-type-based applications like, automated toll-stations and traffic-flow monitoring and surveillance applications like, monitoring activity of a *particular* vehicle. 3D models can handle occlusions and case of multiple objects at different distances from the camera.

Key **contributions** of this work are: 1) novel template based matching to estimate 3D orientation of object from 2D frame and to account foreshortening in projection, 2) incremental 3D model-building using correspondence across the frame sequence, and 3) novel performance index for 3D modeling.

2. Technical Approach

The key assumptions are: (i) vehicle 3D surfaces are planner and edge segments are linear; (ii) vehicle in 3D can rotate only around Z-axes; and (iii) different but constant 3D to 2D projection scales for different 3D directions.

2.1 Generation of Template Library

Perspective projection causes foreshortening of the linear distances and nonlinear mapping of the 3D solid angles to their 2D counterparts. While working with uncalibrated traffic cameras, it is difficult to estimate the projection matrix. In this work, 3D-to-2D nonlinear mapping relations are estimated using a novel idea called "Template Library" and these relations are used to estimate 3D model parameters from 2D features detected in frames. Assumption (iii) above implies (see Fig 1(a)):

$$D_{3D} = S.D_{2D}$$

where S is different for different line orientations. Using the prior knowledge that *most* of the 3D linear edge segments in object centered coordinate (OCC) of vehicles are parallel to one of the coordinate axes in OCC, we just need three such constants along each of the coordinate axes ($[S_x, S_y, S_z]$) for each of the possible azimuths.

Hence a 3D coordinate axes system, with *each 3D axis of unit length,* is rotated around Z-axis for 360 possible azimuths and 360 templates are grabbed. For *each* frame, a template vector (T) is computed (offline) as follows:

$[R, m, n, p, S_x, S_y, S_z]$ *where*
R : azimuth or orientation angle
$[m, n, p]$: 2D angles made by 3D axes in image plane
$[S_x, S_y, S_z]$: 3D - to - 2D scale factors in axes directions

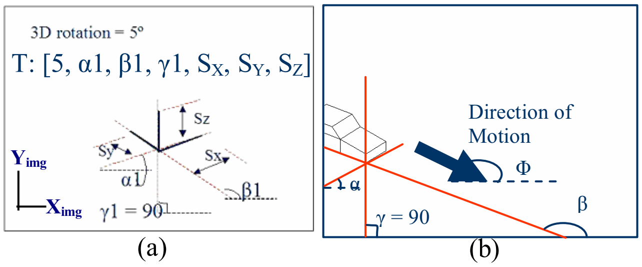
One example frame, with 5° orientation (azimuth) angle is shown in Fig 1(a). Template library is the collection of 360 such vectors (T) for 360 possible azimuths.

2.2 Finding 3D orientation and projection scales

The lower right vertex of frontal plane of the moving vehicle has been selected as the origin of the OCC

The 18th International Conference on Pattern Recognition (ICPR'06)
0-7695-2521-0/06 $20.00 © 2006 IEEE

framework. The 2D angles subtended by the edges at the OCC origin in the image-plane are extracted as shown in Fig 1(b). Orientation assumption constrains one edge-angle to be 90 degrees (the Z axes). Ambiguity between X and Y directions in 2D is solved by the inter-frame motion computation. The angle closest to the motion angle (Φ) is the direction of OCC Y axes. (Note, Φ and β are not always the same due to presence of rotation in vehicular motion.) As in Fig 1(b), we get [α β γ] in [X Y Z] directions. Euclidian match of [α β γ] vector over the corresponding vectors in the template library gives orientation R, and projection scales [u, v, w].



(a) (b)

Fig 1: (a) Example template frame & corresponding template vector, angles computed with respect to Image X axis X_{img} (b) OCC origin, corresponding axes, and motion direction

2.3 3D estimates: vertices and corresponding edges

Initialization is done with the OCC origin (O) as [0 0 0] in 3D. 2D parallelism and projection scales [u, v, w] are used to map the 2D edge-length (in pixels) to 3D units. Then, starting from O [0 0 0] and using 3D edge-lengths and parallelism constraints, the 3D locations of the vertices directly connected to O are estimated. This method is then propagated along different 3D edge-paths to estimate other vertices in turn. For vertices connected by edge segments not parallel to any of the OCC axes, (approx.) geometric relations are used to map image-plane 2D angles to 3D solid angles and then trigonometric relations estimate 3D locations from 2D image-plane locations (Fig. 2).

2.4 3D features

Notably, due to view-volume limits of the camera, not all the vertices and corresponding connecting edge segments are seen completely in every frame. Hence all the vertices are decoupled according to edge segments connected and features are computed for each of the sub-vertices and corresponding (complete or incomplete) edge segments. The 3D view-invariant features are:

- 3D locations of seen sub-vertices, $V = [v_1 \; v_2 \; v_3]$
- Directional parameters of the completely seen edge segment: e.g. for edge segments L_n connecting P and Q
  $L_n = P - Q \; = [(p_1 - q_1) \;\; (p_2 - q_2) \;\; (p_3 - q_3)]$

2.5 Incremental learning using adaptive clustering

Features estimated from a single frame are not very robust due to the approximations used. As more frames are considered, incremental estimates are expected to get more reliable. As in general for traffic video ground truth is not available i.e. the 3D model of the vehicle is not available, unsupervised learning has been resorted. Steps in the incremental unsupervised learning are shown in Fig. 3.

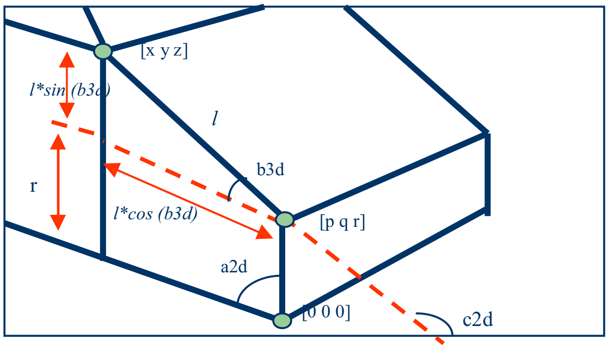


Fig 2: Schematic diagram for estimating 3D vertex locations connected with edges not in direction of any of the OCC axes

```
For each frame:
  1. Extract features for the current frame
  2. For each feature
     a. Cluster (K-means) valid 3D values over seen frames
     b. Fit 3D Gaussian distribution: get mean ($\mu 1$) and standard deviation ($\sigma 1$)
     c. Adaptation: Remove points outside ($\mu 1 \pm 2\sigma 1$)
     d. Unsupervised learning: Fit 3D Gaussian for remaining feature points, get ($\mu$, $\sigma$)
     e. Exponential forgetting: Remaining feature points from (2.c) are added with exponential forgetting
     f. Incrementally learn estimate: normalizing the results from (2.e)
     g. Sub-vertices and edge segment reliability scores: performance measure computation
  3. Incrementally learnt vertices' estimates: weighted sum of corresponding sub-vertices
  4. Vertices' reliability scores: median of the sub-vertices' reliabilities
  5. Model reliability: function of feature reliability scores from (2.g) and 4.
```

Fig 3: Pseudo-code of the incremental learning procedure

Adaptation step is basically outlier rejection for final unsupervised learning by 3D Gaussian distribution fitting and estimating cluster variance. This variance is a measure of learning performance. It is noteworthy that, although we are estimating a constant 3D model of the vehicle in the video-clip, the estimates from different frames are not same due to different noise levels and different estimation-errors due to geometrical projection-approximations in subsections 2.1-2.3. For the incrementally learnt estimate of the model parameters (that are 3D features as well), exponential forgetting has been applied on final cluster, as feature points seen long before are less relevant for present frame estimate.

Incremental estimate of feature F at frame t :

$$F(t) = \frac{\sum_{fr=1}^{t} e^{-L(t-fr)} * k(fr) * F(fr)}{\sum_{fr=1}^{t} e^{-(t-fr)} * k(fr)}$$

where : L = scale factor for controling the effect of forgetting

$$k(fr) = \begin{cases} 0 & \text{if the F(fr) is removed as outlier} \\ 1 & \text{otherwise} \end{cases}$$

and $fr = 0, 1, \ldots.. N$ (total number of frames)

2.6 Reliability Scores: performance measure

Reliability scores are structural accuracy measure of the estimates, with respect to the generic model and the ground-truth. These scores serves dual purpose in this work: (i) finding dynamically adaptive weights for estimates of different 3D model parameters, to incrementally modify the model; and (ii) evaluate the estimated 3D model at any stage for correctness. Reliability has been measured at different abstraction levels, as follows.

2.6.1 Sub-vertex reliability

The factors governing reliability are:

- **Normalized StdDev**: divergence in cluster (2.d, Fig 3)

$$\sigma' = \sigma / \left(1 + \|\mu\|\right)$$

- **SubVdisp (D):** disparity of estimate V' from actual V

$$subVdisp = (V - V') * dispW * (V - V') \Big/ \|V - V'\|$$

dispW changes according to importance of different directions of [X Y Z] in OCC for that vertex.

- **LnCompRatio (C):** edge segment completeness

$$LnCompRatio = \left\|V_1' - V_2'\right\| \Big/ \left\|V_1 - V_2\right\|$$

- **LnAngErr (E):** error between edge segment angle (θ) and ground-truth angle (φ)

$$LnAngErr = abs(\theta - \varphi)/\varphi$$

Reliability of the sub-vertices (subVrlb) are computed as weighted sum of the factors where weights (rlbW) are decided according to their importance at different cases (like complete and not complete):

$$subVrlb = rlbW * \left[C \quad 1/(1+D) \quad (1-E) \quad 1/(1+\sigma')\right]^T$$

2.6.2 Incremental vertex estimate

Vertex incremental estimates are found by weighted sum of the corresponding visible sub-vertices, where weights coming from the sub-vertex reliabilities:

$$V' = \sum_{i:\text{visible subvertices}} subVrlb_i * V'(i) \Big/ \sum_{i:\text{visible subvertices}} subVrlb_i$$

2.6.3 Vertex reliability (Vrlb)

It is the median of the reliability values of the corresponding visible sub-vertices for the present frame.

2.6.4 Edge segment reliability

Edge segment reliability factors are disparity values and reliability values of the terminal sub-vertices, LnCompRatio, LnAngErr, and StdDev (σ) (from 2.d, Fig 3) of the linear segments. These are weighted by rlbW.

$$Erlb = rlbW * \left[C \quad (1-E) \quad 1/(1+\sigma') \quad 1/(1+D_1) \quad rlbVrlb_1 \quad 1/(1+D_2) \quad rlbVrlb_2\right]^T$$

2.6.5 Model reliability

It is the normalized sum of the reliability values of the visible vertices and edge segments.

$$Mrlb = \frac{1}{2}\left(\frac{\sum_{i:\text{all visible vertices}} Vrlb_i}{\sum_{i:\text{all generic vertices}} 1} + \frac{\sum_{i:\text{all visible edge segments}} Erlb_i}{\sum_{i:\text{all generic edge segments}} 1}\right)$$

3. Results and discussion

3.1 Traffic video data

- **Simulated data:** An 8-vertex-8-surface block-based vehicle has been developed and its motion has been simulated with both translation and orientation changes over the frames.

- **Real Traffic video:** Real traffic video data has been collected by an uncalibrated camera in a right-angle street-curve so that the vehicles go slow giving enough frames and also multiple different views for modeling. Feature correspondence is manually done in this study to evaluate the effect of incrementally building the 3D model.

3.2 Results for simulated data

After checking with the number of exponential scale factors (L) in (0.5, 0), we used L = 0.5. Incremental results for frames 25, 35, 60 and 100 are shown in Fig 4. The numbers of the edge segments and vertices are shown. Edge segments are color-coded according to reliability values; from red to violate is increase of reliability.

Availability of ground-truth makes the evaluation of the proposed framework easier. The model reliability value over the complete video sequence is shown in Fig 8(a). As expected for incremental learning, the reliability value increases gradually as more frames are seen with minor deviations due to some newly seen vertex affecting other estimations. Note, some of the vertices are never seen over the entire video sequence.

3.3 Results for real traffic data

For real traffic data, vehicle view changes at a faster rate (i.e., less correlation between frames and estimates may fluctuate due to noise as well). Hence we have used L = 0.7 in exponential forgetting. For the real traffic video data, incremental frame-based results are shown in Fig 5, 6 and 7, with the results superimposed on the actual frames as well. The same color-coding scheme is used.

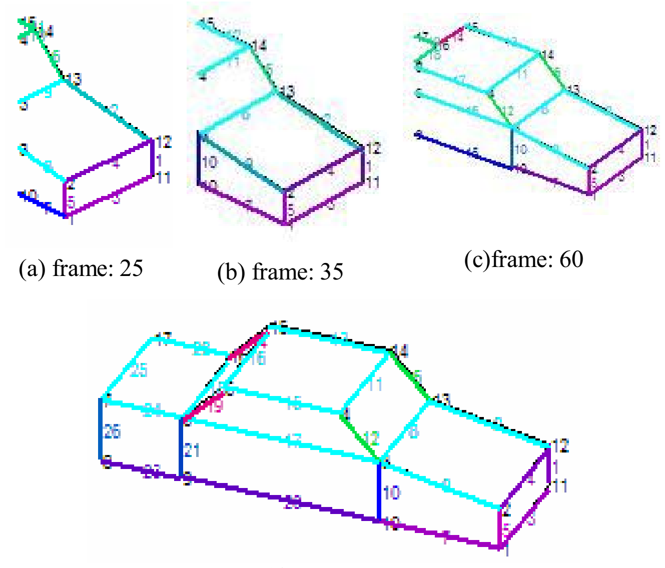


Fig 4: Incremental models at different stages of the sequence

To evaluate the proposed methodology, we have manually estimated an approximate block-based 3D model of the car in this video and computed reliability measures. The reliability value of the estimated 3D model over the video clip is shown in Fig 8(b). Due to inherent noise, coarse ground-truth model, and small number of frames, 3D model estimated from real traffic data is less reliable compared to the simulated data.

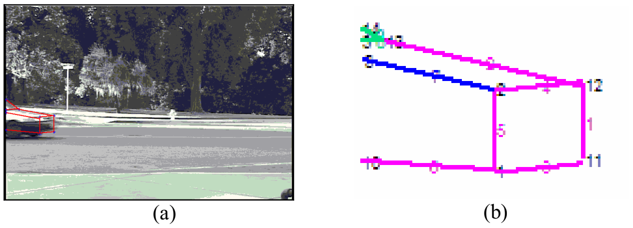


Fig 5: Estimated 3D model after Frame 6 (traffic video)

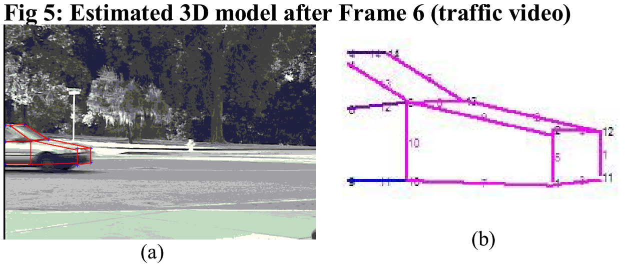


Fig 6: Estimated 3D model after Frame 10 (traffic video)

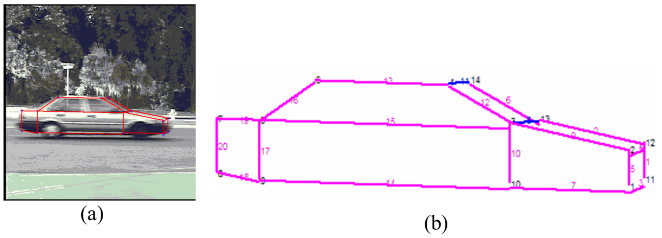


Fig 7: Estimated 3D model after Frame 22 (traffic video)

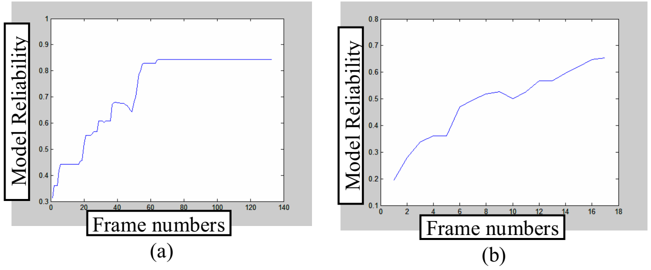


Fig 8: Reliability of the estimated 3D model (a) Simulated video (b) traffic video

Contrary to the expectation, the model-reliability is not monotonically increasing. This is due to the appearance of new (may be noisy) edge segments and vertices that affects previous estimates and hence the reliability values.

4. Conclusions

This paper describes a learning based incremental 3D modeling approach for vehicle modeling from video frame-sequences in an uncalibrated environment. The performance for real traffic video data can be further improved if we acquire more number of frames per vehicle (possibly at higher frame-rate) and possibly from a view-angle where top-surface of the vehicle is also visible, as in the simulated case. As future works we will consider extension to a variety of vehicles, multiple vehicle cases, occlusion/robustness, computational complexity and learning of crucial parameters (like L in exponential forgetting). Bayesian incremental learning is one option.

References

1. JW Hsieh et al, "Morphology-based license plate detec -tion from complex scenes," *ICPR*'02 (3), 176-179.
2. X. Limin, "Vehicle shape recovery and recognition using generic models," *4rth Wrld Cong. Intel. Ctrl. & Aut.*, June 2002, 1055-1059.
3. J. Ferryman et al, "Learning enhanced 3D models for vehicle tracking," *Brit. Mach. Vis. Cnf.* '98, 873-882.
4. M. Kimachi et al, "A vehicle recognition method robust against vehicles' overlapping based on stereo vision," *Proc IEEE of the Intelligent Transp. Sys.*, 5-8 Oct 1999: pp 865-869.
5. G. Foresti et al, "Vehicle recognition and tracking from road image sequences," *IEEE Trans. Vehic. Tech.*, 48(1), 1999, pp 301-318.
6. Y. Guo et al, "Vehicle fingerprinting for reacquisition & tracking in videos," *CVPR*'05, (2), 761-768
7. Y Shan et al, "Unsupervised learning of discriminative edge measures for vehicle matching between non-overlapping cameras," *CVPR*'05, (1), 894-901
8. D. Kollar et al, "Model-based object tracking in monocular image sequences of road traffic scenes," *IJCV*'93, 10(3), 257-281.
9. S. Noronha et al, "Detection and description of buildings from multiple aerial images," *PAMI* 2001, 23(5) 501-518