© IEEE 2009 — J. Prokaj and G. Medioni. 3-D model based vehicle recognition. In *2009 Workshop on Applications of Computer Vision (WACV)*, pages 1-7, 2009. https://doi.org/10.1109/WACV.2009.5403032

3-D Model Based Vehicle Recognition

Jan Prokaj and Gérard Medioni
Institute for Robotics and Intelligent Systems
University of Southern California
Los Angeles, CA 90089
prokaj@usc.edu, medioni@usc.edu

Abstract

We present a method for recognizing a vehicle's make and model in a video clip taken from an arbitrary viewpoint. This is an improvement over existing methods which require a front view. In addition, we present a Bayesian approach for establishing accurate correspondences in multiple view geometry.

We take a model-based, top-down approach to classify vehicles. First, the vehicle pose is estimated in every frame by calculating its 3-D motion on a plane using a structure from motion algorithm. Then, exemplars from a database of 3-D models are rotated to the same pose as the vehicle in the video, and projected to the image. Features in the model images and the vehicle image are matched, and a model matching score is computed. The model with the best score is identified as the model of the vehicle in the video.

Results on real video sequences are presented.

1. Introduction

The number of surveillance systems around us has increased in recent years and is likely to continue growing. Thanks to advances in hardware and lower manufacturing costs, video cameras in these systems now have very high resolution (1080i is commonly supported by cameras on store shelves). This combination of more video sources and higher resolutions produces a staggering amount of data, which must be then reviewed in some way. However, the amount of data precludes a thorough search for objects of interest by a human operator. Instead, content-based retrieval algorithms need to be used to solve this problem. The question then becomes what can be inferred from the raw video data.

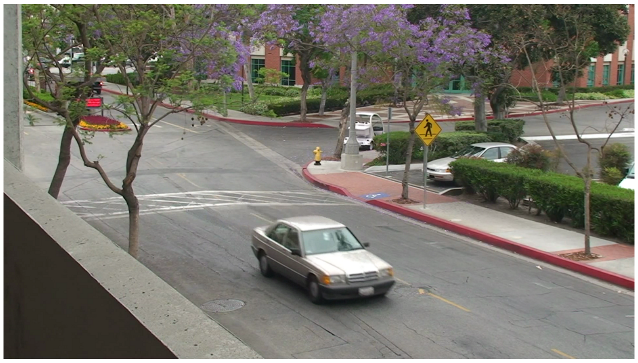

Figure 1. The surveillance scenario considered here.

In this work, our interest is in surveillance systems where the objects of interest are vehicles. Figure 1 illustrates the surveillance scenario considered in this paper. The most common information gathered about vehicles from video is their position over time. There may be some additional information such as the vehicle's appearance, and broad category (sedan, van, truck), but little beyond that. A highly desirable information that is currently not inferred is the vehicle's make and model.

Knowing the make and model of a vehicle is desirable, because, among other things, it allows queries to a retrieval system such as "When was the last time a Ford Focus traveled through this location?". The difficulties in achieving this goal include varying appearance (due to body color and reflections), the varying pose of a vehicle, and the relatively fine (small-scale) features distinguishing different vehicle models from each other.

The main contribution of this paper is a new algorithm that solves exactly this problem: inferring the vehicle's make and model in video clip taken from an arbitrary viewpoint. The key idea is to determine the vehicle's pose, and match the projection of a 3-D model to the vehicle in the video.

Until recently, vehicle type recognition was limited to identifying one of a small set of generic categories, such as a sedan, or a truck [4, 3, 9, 5]. This limitation was removed by Petrovic and Cootes in [15], where the specific vehicle make and model was identified. In that work, a particular region of the car (the front) is used for recognition. This region is normalized to a fixed size, and various features capturing the image structure are calculated from it and form a feature vector. Nearest neighbor classification is used to finally identify the specific vehicle type. Of the features tested, the best performance (over 97% vehicles correctly identified) was achieved using Square Mapped gradients. In the same vein, Negri et al [14] used slightly different features, also based on gradients, and a different, voting-based, classifier, and achieved similar performance.

The weakness of these methods is their reliance on a specific viewpoint, the vehicle's front-view. In addition, these methods work on single images, and do not take advantage of multiple frame data available in a video surveillance context. In this work, we remove these restrictions, and show that vehicle make and model can be identified from a video from an arbitrary viewpoint, and with a great accuracy.

We take a model-based, top-down approach. First, the vehicle pose is estimated in every frame by calculating its 3-D motion using a structure from motion algorithm and assuming the vehicle is moving forward and on a plane. Then, each model in the database of 3-D models is rotated to the same pose, and projected to the image. Features in the model images and the vehicle image are matched, and a model similarity score is computed. The model with the best score is reported as the model of the vehicle in the video.

To establish correspondences for the structure from motion algorithm, we take inspiration from [20, 2] and use a global motion pattern as a prior in calculating sparse optical flow. This increases the tolerance of correspondences to noise from reflections on the car and to ambiguities arising from the aperture problem.

The reconstruction problem in our case is made easier by realizing the motion of the vehicle is planar. There are several approaches [17, 7, 11] to solving the structure from motion problem with this useful constraint. In [17], Rother shows that knowledge of four coplanar points simplifies the problem to a linear system of equations. A plane+parallax formulation of the problem in [7] yields dense reconstruction using direct image measurements without the need for image correspondences. A factorization approach to solve this problem is presented in [11]. Here, we solve this problem by assuming a calibrated camera and a knowledge of the ground plane, and using an incremental reconstruction algorithm. The camera is assumed to be static (or stabilized). The additional knowledge is used to constrain optimization of the reprojection error.

2. Correspondences

Accurate correspondences are critical to the success of algorithms taking advantage of multiple view geometry. Since we are dealing with video data, we would like to use an optical flow based algorithm to get a sparse set of correspondences between adjacent frames of the video. The problem is that in our setting, the correspondences are on the car body, which has a strong specular component. This causes changes in appearance from frame to frame. In addition, we need to deal with ambiguities caused by the aperture problem. To handle all these problems, we formulate the optical flow problem in a Bayesian setting, and use the global motion pattern as a prior. The effect of this prior is to guide the search for correspondences. This idea has been successfully used in tracking vehicles in airborne video and tracking in high density crowd scenes [20, 2].

This global motion pattern is calculated by tracking the vehicle in the video. Here we use a simple tracker based on the Hungarian algorithm [8]. We first subtract the background to find vehicles in every frame (assuming the only large moving objects in the scene are vehicles). The background is modeled as the mode of a sliding window of frames. Then, vehicles are associated from frame to frame using the Hungarian algorithm, where the similarity between two vehicles in different frames is based on how much they overlap. The frame rate is high enough, such that the motion smoothness assumption is satisfied. The final trajectory of a vehicle is then used as a prior in the following optical flow computation.

The optical flow is calculated for a sparse set of stable features, which are computed using the Harris corner detector. For each feature i, the probability of flow $\mathbf{v}_i$ in the frame $\mathbf{I}_j$ is

$$p(\mathbf{v}_i|\mathbf{I}_j) \sim p(\mathbf{I}_j|\mathbf{v}_i)p(\mathbf{v}_i) \tag{1}$$

where $p(\mathbf{I}_j|\mathbf{v}_i)$ denotes the image likelihood and $p(\mathbf{v}_i)$ is the flow prior. The flow is then the maximum a posteriori (MAP) estimate,

$$\mathbf{v}_i^* = \arg\max_{\mathbf{v}} p(\mathbf{v}|\mathbf{I}_j) \tag{2}$$

If the probability of $\mathbf{v}_i^*$ is less than a threshold, this feature point is removed.

Here, $\mathbf{v}_i$ is a discrete variable, whose possible values are determined by a small circular region centered on the predicted location of the feature point in the current frame. The radius of the region is $1.5\|\overline{\mathbf{v}}\|$, where $\overline{\mathbf{v}}$ is the global motion in the current frame computed from the vehicle trajectory. If the position of the vehicle's center in the current frame is $\mathbf{c}_j$, and in the previous frame is $\mathbf{c}_{j-1}$,

$$\overline{\mathbf{v}} = \mathbf{c}_j - \mathbf{c}_{j-1} \tag{3}$$

The prediction location of the feature point, $\mathbf{x}_i$, is simply the previous position, $\mathbf{x}_{i-1}$ plus $\overline{\mathbf{v}}$.

The flow prior regularizes both the flow direction and magnitude:

$$\begin{aligned} p(\mathbf{v}_i) &= p_{direction}(\mathbf{v}_i)p_{magnitude}(\mathbf{v}_i) && (4) \\ p_{direction}(\mathbf{v}_i) &= k\,0.5(\hat{\mathbf{v}}_\mathbf{i} \cdot \hat{\overline{\mathbf{v}}} + 1) && (5) \\ p_{magnitude}(\mathbf{v}_i) &= k\,e^{-(\|\mathbf{v}_i\|-\|\overline{\mathbf{v}}\|)^2/\sigma^2} && (6) \end{aligned}$$

where k denotes a normalization constant so that the probability of all possible flows sums to 1, and σ is set to a fixed value, or a fraction of $\|\overline{\mathbf{v}}\|$, such that the allowable range of flow magnitude has non-zero probability. The prior expresses that the flow should be similar in magnitude and direction to the global motion of the vehicle.

The flexibility of a Bayesian formulation allows one to model the likelihood in many different ways. Here, the likelihood is computed using normalized cross-correlation at multiple scales:

$$p(\mathbf{I}_j|\mathbf{v}_i) = \prod_{s\in S} k_s NCC(\mathbf{I}_{j-1}(\mathbf{x}_{i-1} + \mathbf{v}_i), \mathbf{I}_j(x_i), s) \quad (7)$$

where S denotes a small set of square sized windows, and k is again a normalization constant. Cross-correlation is chosen for its invariance to lighting changes.

In the current formulation, $\mathbf{v}_i^*$ is integer valued. Since we need accurate correspondences for geometry calculations, subpixel accuracy is desired. To accomplish this, a quadratic is fit to $p(\mathbf{v}|\mathbf{I})$ in the immediate neighborhood of the optimum $\mathbf{v}_i^*$.

If the vehicle trajectory is noisy, it can adversely impact the flow estimate through the influence of the prior. This problem is handled by simultaneously estimating $\overline{\mathbf{v}}$ in an EM-like fashion over a small number of iterations. First, (2) is solved using the current estimate of $\overline{\mathbf{v}}$. Then $\overline{\mathbf{v}}$ is computed as

$$\overline{\mathbf{v}} = \mathbf{E}[\mathbf{v}] = \frac{1}{N}\sum_i \mathbf{v}_i \quad (8)$$

This effectively removes any noise in $\overline{\mathbf{v}}$ in as few as 3 iterations.

If the likelihood only considers translating motion, as in our case by using cross-correlation, some accuracy is lost. This can be rectified by realizing that the motion of the correspondences is planar. So to further refine the estimates a homography is estimated between the frames, and the process just described repeated, only this time with one of the frames warped in the cross-correlation (likelihood) computation.

The complexity of this approach is dominated by the likelihood computation. The likelihood needs to be determined for every possible value of flow, which can be a relatively large set of values to consider. In our implementation, we used normalized cross-correlation at three different scales to compute the likelihood, which is inefficient, but more practical schemes can be easily substituted.

3. Structure from Motion

The needed pose of the vehicle is determined by calculating its 3-D motion on a plane and assuming the vehicle is moving forward. In our current implementation the camera is assumed to be static, but a moving camera can be incorporated by stabilizing the video first. Since we are interested in the pose of the moving vehicle, only correspondences on the vehicle are used. These are determined from the background subtraction. A standard pinhole camera model is assumed, and the camera is calibrated off-line, which is reasonable in a surveillance context.

Determining the motion of the vehicle with a static camera is equivalent to assuming the vehicle is static and determining the motion of a virtual camera. We now show that the motion of a virtual camera is inverse of the motion of the vehicle. Since a vehicle is a rigid body, its motion is completely described by a 3-D rotation, $\boldsymbol{R_m}$, and a translation, $\boldsymbol{t_m}$:

$$\boldsymbol{M} = \begin{bmatrix} \boldsymbol{R_m} & \boldsymbol{t_m} \\ \mathbf{0^T} & 1 \end{bmatrix} \quad (9)$$

Let $\boldsymbol{x_i} = \boldsymbol{P}_{\text{static}}\boldsymbol{MX}$ be the projection of an arbitrary point $\boldsymbol{X}$ on the moving vehicle with a static camera $\boldsymbol{P}_{\text{static}}$. Then,

$$\begin{aligned}
\boldsymbol{x_i} &= \boldsymbol{P}_{\text{static}}\boldsymbol{MX} \\
&= \boldsymbol{K}[\boldsymbol{R} \ \ -\boldsymbol{RC}]\boldsymbol{MX} \\
&= \boldsymbol{K}[\boldsymbol{RR_m} \ \ (\boldsymbol{Rt_m} - \boldsymbol{RC})]\boldsymbol{X} \\
&= \boldsymbol{K}[\boldsymbol{RR_m} \ \ -\boldsymbol{R}(\boldsymbol{C} - \boldsymbol{t_m})]\boldsymbol{X} \\
&= \boldsymbol{K}[\boldsymbol{RR_m} \ \ -\boldsymbol{RR}_m(\boldsymbol{R_m^{-1}}(\boldsymbol{C} - \boldsymbol{t_m}))]\boldsymbol{X} \\
&= \boldsymbol{K}[\boldsymbol{R'} \ \ -\boldsymbol{R'C'}]\boldsymbol{X}
\end{aligned}$$

where the motion of a virtual camera from $\boldsymbol{C}$ to its new position, $\boldsymbol{C'}$, is exactly the inverse of the motion of the vehicle.

The key property enforced in our structure from motion algorithm is that the motion of the virtual camera is in a plane parallel to the ground plane. Therefore, we need to know the ground plane. This can be calculated automatically from vanishing points, or determined interactively [19, 6]. Since this only needs to be done once, a semi-automatic method is acceptable. The knowledge of the ground plane is used to define a world coordinate system (WCS) where the x and y axes span the plane, the z axis is perpendicular to the plane, pointing up, and the origin is set by choosing an arbitrary point on the plane. This is illustrated in Figure 2 (a left-handed coordinate system is used). Defining the WCS this way makes it easy to constrain the motion as desired.

With the WCS determined, we can define the virtual camera projection matrix in the first frame, which is the same as that of the static camera. The camera matrices in the following frames will be determined with respect to this camera. Using the notation in [11], let $\mathbf{g_x}, \mathbf{g_y}, \mathbf{g_z}$ be the axes of the WCS in the camera coordinate system (CCS), and let $\mathbf{g_0}$ be the origin in CCS. If (x, y) is the location of the origin in the image, then $\mathbf{g_0} = \lambda(x, y, f)^T$, where f

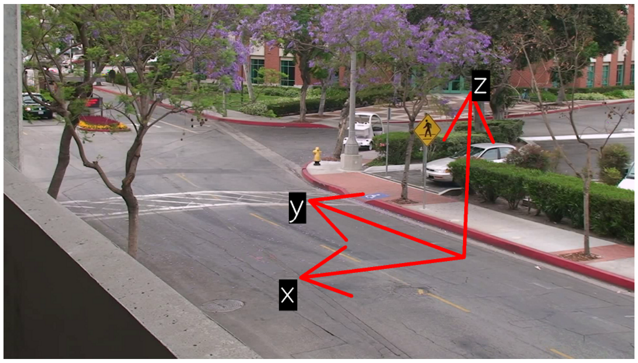

Figure 2. The chosen world coordinate system.

is the known focal length, and λ is the perspective depth, which is set arbitrarily. The camera projection matrix is then

$$\begin{aligned} \boldsymbol{P_0} &= \boldsymbol{K}\left[\begin{array}{cccc} \mathbf{g_x} & \mathbf{g_y} & \mathbf{g_z} & \mathbf{g_0} \end{array}\right] \\ &= \boldsymbol{K}\left[\begin{array}{cc} \boldsymbol{R_0} & \mathbf{g_0} \end{array}\right] \end{aligned}$$

where K is the known camera calibration matrix.

An incremental structure from motion algorithm similar to [16] is used. The video is temporally downsampled to provide a more stable wider baseline in the geometry calculations. In the first two frames, an essential matrix is estimated. RANSAC is used to remove errors in correspondences. The pose of the second camera relative to the first is determined by decomposing the essential matrix and choosing one of four possible solutions [6]. Letting the first camera be $\boldsymbol{P_0}$, and the relative pose be $[\boldsymbol{R_1} \quad \boldsymbol{t_1}]$, the absolute pose of the second camera is

$$[\boldsymbol{R_0}\boldsymbol{R_1} \quad (\boldsymbol{R_0}\boldsymbol{t_1} + \boldsymbol{g_0})] \tag{10}$$

This absolute pose of the second camera is optimized using Levenberg-Marquardt [12] by minimizing the reprojection error such that the motion of the camera is in the ground plane. That is, the rotation is constrained to be about $(0, 0, 1)$, the ground plane normal, and the center of projection is constrained to lie in the same plane as $\boldsymbol{P_0}$, which is parallel to the ground plane. In total, there are four degrees of freedom. Triangulation in two views is done optimally by solving a 6th degree polynomial, which represents the reprojection error parameterized by the choice of epipolar lines [6].

In subsequent frames, the camera pose is estimated by solving the Perspective-n-Point problem. We use the recent algorithm by Lepetit et al [10], which is fast, non-iterative, and gives very accurate results. Additional robustness against noise is gained by using RANSAC. Triangulation of points visible in multiple views is done by forming a linear system and solving with iteratively reweighted least squares.

After every frame is processed, bundle adjustment [18] is used to refine the current solution. The parameters are the camera center (3 unknowns for each camera, except $\boldsymbol{P_0}$), the camera rotation about the ground plane normal (1 unknown for each camera, except $\boldsymbol{P_0}$), and point coordinates (3 unknowns for each point). As before, the cameras are constrained to move in the same plane as $\boldsymbol{P_0}$. Levenberg-Marquardt [12] is used again to do the minimization of the reprojection error.

When the structure from motion algorithm completes, the vehicle's motion direction (and thus pose) in frame i is estimated as $\| - (\boldsymbol{C_i} - \boldsymbol{C_{i-1}})\|$, where $\boldsymbol{C_i}$ is the virtual camera's center of projection in frame i.

The complexity of this step is dominated by bundle adjustment, but overall it is efficient. With the pose of the cameras constrained, the number of degrees of freedom is reduced, so our formulation is actually more efficient than the general case. Performance can be further improved by limiting the bundle adjustment to cameras and structure in the last n frames.

4. Model Classification

The knowledge of the vehicle pose and the pose of the camera allows us to reduce the model classification problem from 3-D to 2-D. This is important, because it makes the problem easier, and the solution more reliable. Measuring similarity in a lower dimensional space is always easier than in a higher-dimensional one. In addition, model classification in 3-D would either require a dense reconstruction of the vehicle in the video, or a fit of sparse reconstruction to a model. Either option adds significant complexity, with no guaranteed gain in performance. Furthermore, the solution for vehicle motion is easily constrained, as we have just shown, whereas the solution for vehicle structure can not be (easily) constrained at all. In other words, there is higher confidence in the motion estimate than in the structure estimate.

The problem is reduced to 2-D by rotating a potential 3-D vehicle model to the same pose as the vehicle in the video, and projecting it to the image. To perform classification, models from a database are projected in turn, and the model with the best matching score is selected. This approach depends on having access to a database of good quality 3-D vehicle models, but this has become less of a problem with the introduction of 3-D model sharing sites, such as Google's 3D Warehouse [1].

We would also like to point out that discretizing the vehicle pose space and skipping the pose estimation will not achieve the same result, unless the video capture is severely restricted. For example, our method will also work with a moving camera after adding a video stabilization module. But there is great benefit even in the static camera case; when the vehicle in the video is making a turn, our method can take advantage of seeing the vehicle from multiple viewpoints.

The previous work in vehicle classification, and object recognition in general, has made good use of histograms of oriented gradients as features. These seem to be very stable, and have good discriminatory characteristics, especially when they are collected over multiple scales. We make use of this type of features here as well. The features are calculated the same way as SIFT [13] features, but the rotation invariance of the descriptor is deliberately disabled. The vehicles in the model image and the video image are already normalized to the same view, so enabling rotation invariance is actually detrimental. In addition, for the same reason, when the features are matched between images (as discussed below), the matches are restricted to be in similar scales. If the scale of a feature in the video image is s, the scale, s', of a matching feature in the model image needs to be $0.75s < s' < 1.5s$.

Our strategy for measuring the similarity between an image of a vehicle from a video and the image of a vehicle from a model consists of two steps. First, we find feature correspondences in the images, and then we compare the relative positioning of the matched features in the model image with the relative positioning of the corresponding features in the video image. Feature correspondences are found by finding the closest descriptor (measured with Euclidean distance) in the model image to each descriptor in the video image. If the distance to the closest descriptor is less than a threshold, the match is valid. In addition, to further decrease the number of spurious matches, we only consider features found at scales larger than some minimum scale (for example, by ignoring features found in the first octave of the Laplacian pyramid). The reason is that features found at small scales miss the "big picture" and their correspondences are not optimal in the global sense. Figure 3 shows an example of valid feature matches between the video image and the model image.

Let $S_f = \{(\mathbf{x}_1^i, \mathbf{x}_2^i) | i \in [1, N]\}$ be the set of valid feature matches in frame f, where $\mathbf{x} = (x, y)$ and the subscript indicates the model/video image, $\mathbf{v}_1^{ij} = \mathbf{x}_1^i - \mathbf{x}_1^j$, and $\mathbf{v}_2^{ij} = \mathbf{x}_2^i - \mathbf{x}_2^j$. The video-model similarity in frame f is

$$\begin{aligned} sim_f(S_f) &= \left(\frac{|S_f|}{M}\right) \cdot \frac{2}{N(N-1)} \sum_{i=1}^{N} \sum_{j=i+1}^{N} Dir_{ij} Mag_{ij} \\ Dir_{ij} &= 0.5(\hat{\mathbf{v}}_1^{ij} \cdot \hat{\mathbf{v}}_2^{ij} + 1) \\ Mag_{ij} &= exp(-| \|\mathbf{v}_1^{ij}\| - \|\mathbf{v}_2^{ij}\| |) \end{aligned}$$

where M is the total number of features in the video image. The total video-model similarity is the average of all frame similarities:

$$sim(S) = \frac{1}{F} \sum_{f=1}^{F} sim_f(S_f) \qquad (11)$$

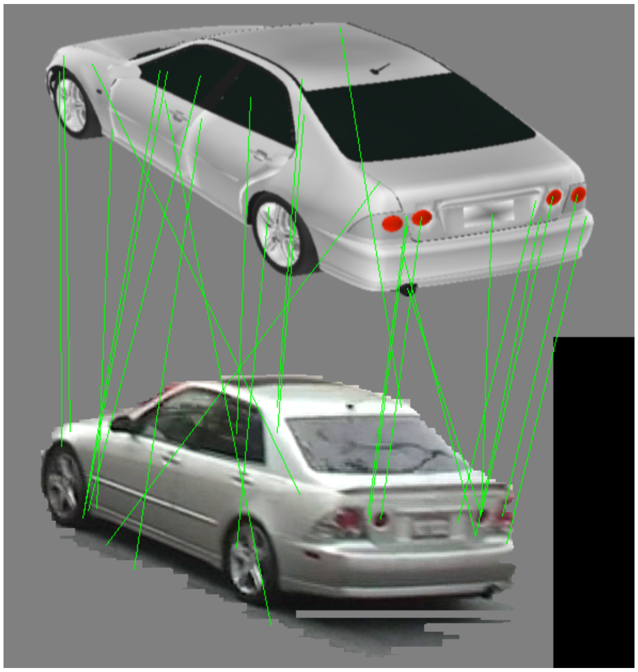

Figure 3. Example feature matches.

where F is the total number of frames in the video. The vehicle in the video is classified with the model having the maximum similarity.

The video-model similarity is efficient to compute. However, the vehicle classification speed decreases with a larger model database, because every model needs to be considered. To achieve a practical runtime performance with large databases, we propose to use a hierarchical organization of the models, where dissimilar models can be quickly identified, and skipped. This is the subject of our future work.

5. Results

Vehicle recognition was evaluated on 20 video clips and a database of 36 models. The length of videos ranged from 4 to 40 frames. The viewpoints of vehicles in the videos can be described as "mostly front" and "mostly back". The video resolution was 1920x1080. 3-D vehicle models were downloaded from [1], and their complexity ranged from 5,000 to 120,000 polygons. A subset of the models used in the experiments is shown in Figure 4. Camera calibration was performed using Jean-Yves Bouguet's camera calibration toolbox. The ground plane was estimated using vanishing points and verified interactively.

Vehicle recognition performance is shown in Table 1. The first row indicates the performance of the algorithm in its standard configuration, where all frames of the video are used to determine the vehicle-model similarity. The next two rows show the performance broken down by the gen-

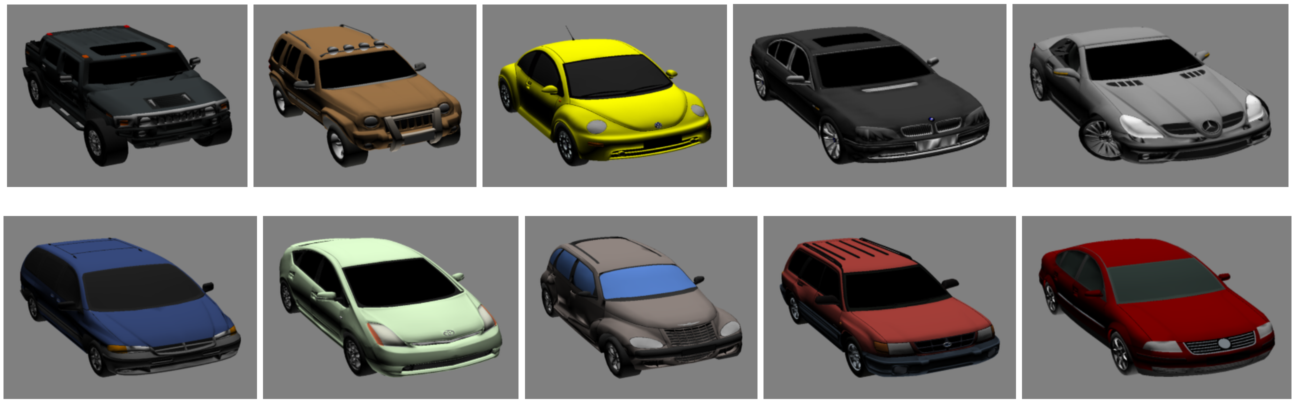

Figure 4. A subset of the models used in the experiments.

	Correct Classification (%)	
	Rank 1	Rank 3
Multiple Frames	50	85
- Front view (8/20)	62.5	87.5
- Back view (12/20)	41.6	91.6
Single Frame	15	40

Table 1. Vehicle recognition performance.

eral vehicle viewpoint. A rank-k classification means a vehicle is classified correctly if it is ranked $\leq k$ in the result set. The result set is ordered using equation (11). Examples of correct and incorrect vehicle recognition are shown in Figures 5 and 6.

The results show that our algorithm is excellent at identifying the most likely makes and models for vehicles in video. The correct make and model is always top ranked. This kind of performance lays very good ground work for a second-stage fingerprinting algorithm, which can use very fine features to determine the precise make and model for a vehicle. The results also show that vehicles with a visible front were classified more accurately than vehicles with a visible back.

In addition to evaluating the algorithm in its standard configuration, we also evaluated it in single-frame mode, where only one frame of the video-clip is used in calculating the vehicle-model similarity. The purpose of this experiment was to see whether classification performance improves with taking advantage of multiple frame data.

Clearly, the multiple-frame performance is much better than single-frame performance. This is not surprising since each frame provides additional matching information to the algorithm. The noise in feature matches is more tolerable when more frames are available.

6. Conclusions

We presented a method for recognizing a vehicle's make and model in video clip taken from an arbitrary viewpoint. The video-model similarity is determined by first matching features similar to histograms of oriented gradients and measuring the relative configuration of the features in video and in the model. The results show excellent performance at identifying the most likely vehicle make and model. In addition, we confirmed that higher classification accuracy is obtained when the front of the vehicle is visible.

We also presented a Bayesian approach for establishing accurate correspondences in multiple view geometry. A global motion pattern was used as a prior in this approach. A structure from motion algorithm was then able to successfully determine the vehicle pose from these correspondences. In determining the vehicle pose, the key is to use the knowledge of the ground plane to impose constraints on the virtual camera motion.

The algorithm's scalability to large model databases is currently being investigated. We propose to use a hierarchical organization of the models, where dissimilar models can be quickly identified, and skipped in subsequent computation.

7. Acknowledgments

This work was supported in part by grant DE-FG52-08NA28775 from the U.S. Department of Energy.

References

[1] Google 3D warehouse. http://sketchup.google.com/3dwarehouse/.

[2] S. Ali and M. Shah. Floor fields for tracking in high density crowd scenes. In *10th European Conference in Computer Vision*, volume 5303 of *LNCS*, pages 1–14. Springer Berlin / Heidelberg, 2008.

[3] M.-P. Dubuisson Jolly, S. Lakshmanan, and A. Jain. Vehicle segmentation and classification using deformable templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(3):293–308, Mar 1996.

[4] J. M. Ferryman, A. D. Worrall, G. D. Sullivan, and K. D. Baker. A generic deformable model for vehicle recognition.

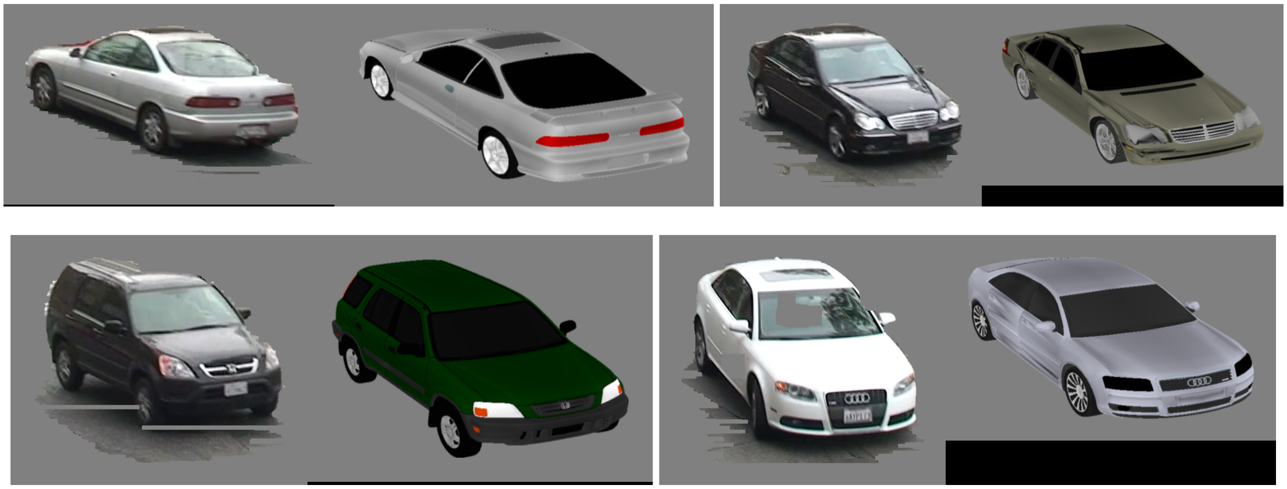

Figure 5. Examples of correct vehicle recognition.

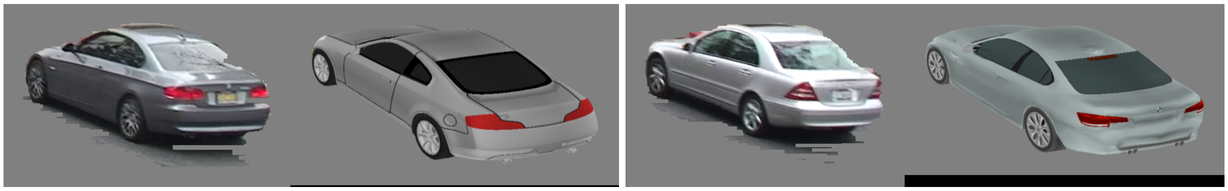

Figure 6. Examples of incorrect vehicle recognition.

In *British conference on Machine vision*, volume 1, pages 127–136. BMVA Press, 1995.

[5] D. Han, M. Leotta, D. Cooper, and J. Mundy. Vehicle class recognition from video-based on 3d curve probes. In *2nd Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, pages 285–292, Oct. 2005.

[6] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2nd edition, 2003.

[7] M. Irani, P. Anandan, and M. Cohen. Direct recovery of planar-parallax from multiple frames. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(11):1528–1534, Nov 2002.

[8] H. Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(1-2):83–97, March 1955.

[9] A. Lai, G. Fung, and N. Yung. Vehicle type classification from visual-based dimension estimation. In *IEEE Intelligent Transportation Systems*, pages 201–206, 2001.

[10] V. Lepetit, F. Moreno-Noguer, and P. Fua. Epnp: An accurate o(n) solution to the pnp problem. *International Journal of Computer Vision*, 81(2):155–166, 2009.

[11] J. Li and R. Chellappa. A factorization method for structure from planar motion. In *IEEE Workshop on Motion and Video Computing, ACV/MOTIONS '05*, volume 2, pages 154–159, Jan. 2005.

[12] M. Lourakis. levmar: Levenberg-marquardt nonlinear least squares algorithms in C/C++. http://www.ics.forth.gr/ lourakis/levmar/, 2009.

[13] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[14] P. Negri, X. Clady, M. Milgram, and R. Poulenard. An oriented-contour point based voting algorithm for vehicle type classification. In *18th International Conference on Pattern Recognition*, volume 1, pages 574–577, 2006.

[15] V. Petrovic and T. F. Cootes. Analysis of features for rigid structure vehicle type recognition. In *British Machine Vision Conference*, pages 587–596, 2004.

[16] M. Pollefeys, L. Van Gool, M. Vergauwen, F. Verbiest, K. Cornelis, J. Tops, and R. Koch. Visual modeling with a hand-held camera. *International Journal of Computer Vision*, 59(3):207–232, 2004.

[17] C. Rother and S. Carlsson. Linear multi view reconstruction and camera recovery. In *IEEE International Conference on Computer Vision*, volume 1, pages 42–50 vol.1, 2001.

[18] B. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon. *Vision Algorithms: Theory and Practice*, volume 1883 of *LNCS*, chapter Bundle Adjustment A Modern Synthesis, pages 153–177. Springer Berlin / Heidelberg, 2000.

[19] A. D. Worrall, G. D. Sullivan, and K. D. Baker. A simple, intuitive camera calibration tool for natural images. In *Conference on British machine vision*, volume 2, pages 781–790. BMVA Press, 1994.

[20] Q. Yu and G. Medioni. Motion pattern interpretation and detection for tracking moving vehicles in airborne videos. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2671–2678, 2009.