Car Make and Model Recognition using 3D Curve Alignment

Krishnan Ramnath, Sudipta N. Sinha, Richard Szeliski
Microsoft Research
{kramnath,sudipsin,szeliski}@microsoft.com

Edward Hsiao
Carnegie Mellon University
ehsiao@cs.cmu.edu

Abstract

We present a new approach for recognizing the make and model of a car from a single image. While most previous methods are restricted to fixed or limited viewpoints, our system is able to verify a car's make and model from an arbitrary view. Our model consists of 3D space curves obtained by backprojecting image curves onto silhouette-based visual hulls and then refining them using three-view curve matching. These 3D curves are then matched to 2D image curves using a 3D view-based alignment technique. We present two different methods for estimating the pose of a car, which we then use to initialize the 3D curve matching. Our approach is able to verify the exact make and model of a car over a wide range of viewpoints in cluttered scenes.

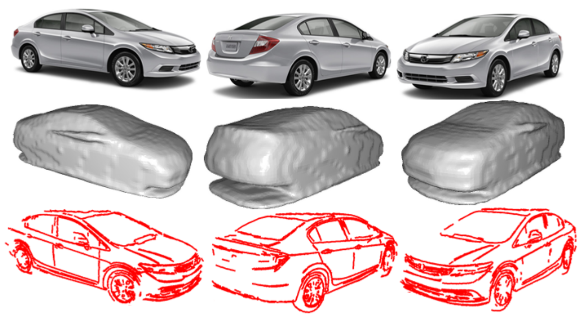
Figure 1. Steps in constructing our 3D car model for a 2011 Honda Civic Sedan: (top) three of the images used to generate the visual hull; (middle) the visual hull; (bottom) 3D space curves projected onto the visual hull.

1. Introduction

Recognizing the exact make, model, and year of a car from an arbitrary viewpoint is something that car aficionados do with relative ease. To date, however, no computer vision system can mimic this ability over a wide range of viewpoints. Why is this so?

At first glance, car recognition should be readily solvable, since it is an example of *instance recognition*, where we can build or obtain high-quality 3D models or large numbers of reference images. However, cars are more challenging than many other man-made objects, since they have large untextured regions and their appearance is often dominated by highlight lines and environmental reflections. The question we address is therefore: is car recognition an easily solvable example of fine-grained subordinate category recognition [6, 7] or an extremely challenging vision problem due to the complexity and variability in appearance?

To date, most of the work in recognizing the exact make and model of a car involves only a single viewpoint or limited viewpoint change [24, 4, 23]. In parallel, work on general object category detection and recognition is starting to simultaneously recognize the rough pose of the object [30, 28, 21, 17, 8]. Our paper aims to unify these two approaches, by developing a system than can handle an arbitrary viewpoint while also determining the exact make and model of a vehicle. While our main focus is on the verification stage, i.e., determining the exact model given a rough initial pose estimate for the car, we also present two different pose estimation (automatic initialization) methods, one that is an extension of a previous technique [31] and one we develop in this paper.

In the following, we describe our processing pipeline and matching algorithms for solving this problem. We begin with a review of related work in this field. We follow this with an overview of the problem and justify our choice of view-based 3D curve models as the basic representation used in our system. We then describe the steps for automatic reconstruction of these 3D curve models from training images (Sections 3.2 and 3.3). We then show how our models can be rigidly aligned to novel test images using oriented 3D chamfer matching to perform simultaneous pose and make/model recognition (Section 4). Section 5 describes how we combine our chamfer matching scores to perform classification using logistic regression. Section 6 describes two different techniques for estimating the initial pose of the car, which enables the system to be fully automatic, albeit at a reduced performance rate. Finally, we present our experimental results on eight different car models and close with a discussion of our results and ideas for future work.

2. Previous work

3D object recognition is a fundamental problem in computer vision, with a long history dating back to the late 1960s. Hoiem and Savarese [9] present a nice survey on this topic and a comprehensive list of the large number of papers published in this area.

Some of the early techniques were based on the concept of 3D alignment between 3D curve models and 2D image edges [12]. However, these techniques fell out of favor, since they required strong curvilinear features and were replaced by view-based techniques which used interest point or region detectors and descriptors [30, 28].

Work in detecting cars, often from fixed viewpoints such as side views, has been evolving in parallel [25] and has become, along with detecting people, bicycles, and other objects, a central component of the PASCAL VOC (Visual Object Category) challenge [5].

More recently, researchers have started to combine object detection with pose estimation. For example, Ozuysal *et al.* [21] use an initial bounding box of a car to select a view-specific classifier to refine the hypothesis, while Glasner *et al.* [8] vote for potential pose parameters and then refine these using view-specific SVMs. Li *et al.* [17] learn the appearance of features located at key points (wheels, corners) of a car and use an elastic shape model to detect both the location and pose of cars in street images.

While most of these techniques are based on traditional feature detectors and descriptors, a parallel line of work uses contour-based recognition and pose estimation. Shotton *et al.* [26] use a star-shaped model of object contours and introduce oriented chamfer matching, where the orientations of the edges are taken into account when measuring their correspondence distance. (More recent work on oriented chamfer matching includes [18].) Lu *et al.* [20] extract edgels and link them to form contours, which are then grouped into part bundles and matched using shape context similarity [1]. Leotta and Mundy [15] build a 3D deformable vehicle model and show good edge-based fitting results on images. Most recently, Payet and Todorovic [22] use view-based shape templates and a bag of boundaries representation placed on a deformable 2D grid and show good results on boundary detection and pose estimation.

To date, most of the work on recognizing specific car models has been limited to fixed viewpoints, e.g., front or rear views of cars [4, 23]. A notable exception is the recent paper of Jang and Turk [13], which uses SURF features and a quantized bag of words approach to recognize toy model cars shot against a uniform background. Our work extends the domain of applicability to cluttered real-world scenes with strong reflections and demonstrates that curve-based recognition can be more accurate than existing methods.

3. 3D Curve Models

In our technique, each car model is represented by a set of non-parametric 3D curves. We reconstruct these curves from natural images rather than from CAD models, since image-based modeling approaches generate 3D curves that are more likely to match 2D curves detected in test images taken "in the wild". In this section, we present our approach for reconstructing these 3D curve models and matching these models to new test images.

Instead of using a single global 3D model, we use a view-based representation consisting of separate view-dependent partial 3D curve models, where each model is associated with the viewpoint corresponding to each of the training images. In order to align the 3D curve model to a new test image, we find the closest training image and then align the corresponding partial 3D curve model to the edges in the test image by estimating a rigid 3D perspective image transformation. This has the advantage that subtle view-dependent features can be modeled, and the visibility of the curves is handled naturally.

Curve matching could be performed using pure 2D alignment and chamfer matching [26, 18]. However, even with non-rigid alignment techniques such as Active Shape Models (ASMs) [3, 16], the large variation in the curve rendered from an arbitrary camera viewpoint make it difficult to simultaneously recover a good registration between the 3D model and the test image while still discerning the subtle shape differences that differentiate one car model from another (Section 7). Since Active Shape Models will in general not reflect the true 3D transformation of a rigid 3D car model, a better approach is to directly use a 3D model.

3.1. Formulation

We represent the 3D curves in each view of the 3D car model by a set of N_{M} 3D point samples P_i. The goal of alignment (Section 4) is to recover the unknown camera pose or viewing transformation $M = K[R|t]$ of the 3D model that minimizes the sum of reprojection errors between the N_{M} projected 3D model points and their corresponding nearest 2D edge points or *edgels*, amongst the N_{T} edgels in the test image denoted as $\{p_k\}$. The optimal transformation M^* is the one that minimizes

$$D_c = \frac{1}{N_{\mathrm{M}}} \sum_{i=1}^{N_{\mathrm{M}}} \min_k d(p_k, \mathcal{P} M P_i), \tag{1}$$

where $d(p, q)$ is one of the 2D distance metrics discussed in Section 4. The operator $\mathcal{P}$ projects 3D points into the test image, and the minimum distance over the edgels $\{p_k\}$ can be efficiently computed using a distance transform [11].[1]

[1]This is often called *chamfer matching* since the distance transform resembles the chamfers used as guides in manufacturing processes.

The most common method for obtaining 3D shape from calibrated images is to first recover pixel correspondences across images captured from multiple viewpoints and then perform multi-view triangulation. However, for curves, directly computing reliable pixel-to-pixel correspondence from adjacent images is a challenging problem. For this reason, we first build a visual hull model (Section 3.2), and then use this geometry as the initial basis for our 3D curve locations, which we then further refine using robust three-view stereo matching (Section 3.3).

3.2. Visual Hull Reconstruction

Our input data consists of multiple turntable images of each car model taken at regular angular intervals against a clean background (Figure 1). Such images are often available on manufacturers' web sites, and in a production system, would be captured with carefully calibrated cameras. Since in our case we did not have access to the intrinsic camera parameters, we automatically detect the ellipses corresponding to the wheels (Section 6.2) in the subset of images where these could be reliably detected and use the vanishing points defined by the ellipse bitangents to estimate both the focal length and optic center of each camera in a sequence.

For each calibrated sequence, we obtain our initial 3D curves by backprojecting the 2D image curves onto an automatically generated visual hull reconstruction of the car [14]. We extract binary silhouettes from each input image using thresholding followed by appropriate morphological operations. Next, we build a volumetric 3D model by intersecting the backprojected silhouette cones in 3D, and finally project this model into each image to obtain an associated depth map. Since all cameras are calibrated, any 2D point p in an image with a valid depth estimate can be mapped to a 3D point P on the visual hull. Using this technique, we map a set of 2D points constituting a 2D curve in the image to obtain an initial estimate of the corresponding 3D curve. These curves are assumed to be visible from viewpoints near the original camera viewpoint.

As an alternative to our approach, in order to handle car images with textured backgrounds, one could use structure from motion to reconstruct the full scene and then automatically segment the car using the method proposed in [8].

3.3. Model Refinement

Edges seen on cars can arise from specular reflections in the scene. The resulting spurious curves in the 3D curve model can corrupt the 3D chamfer matching score and need to be removed for robustness. We filter these spurious edges based on the observation that given multiple viewpoints, the 3D position of points on these curves inferred from the 3D model are not consistent across different views (Figure 2b).

To detect spurious edges for a reference view, we consider its left and right neighboring views in the input sequence. Each 3D point on a curve is projected into the neighboring views to test whether the projected point lies within τ_d pixels of an edge point on the corresponding epipolar line in the respective image. Points that have valid matches in both neighboring views are retained.

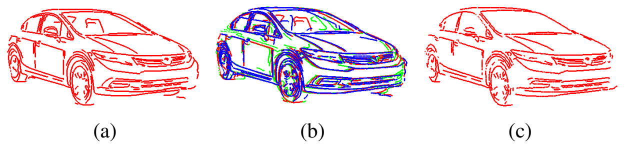
Figure 2. Curve Refinement: (a) original curves; (b) curves from neighboring views reprojected into current view; (c) the subset of curves with depths consistent in three views.

For each retained point, given the two points matched on corresponding epipolar lines in the neighboring views, we use the Direct Linear Transform [29], *i.e.* linear triangulation to re-estimate the position of the 3D point. This refinement is performed for all 3D points on all 3D curves in every viewpoint of the model . An example of the refined model is shown in Figure 2c.

4. 3D Chamfer Matching

Once we have built our 3D view-based curve models, we can use these to recognize the car make and model of a new test image. For each model, we estimate the transformation $M = K[R|t]$ that minimizes the sum of reprojection errors D_c given in equation (1) between the N_M projected 3D points of the model, $\mathcal{P}MP_i$, and the N_T 2D points in the image, $\{p_k\}$.

To avoid an expensive search over all possible model poses and positions, we would like to efficiently recover the initial pose of the car using a pose estimation approach. A number of such techniques have been developed [21, 31]. In Section 6 we describe a novel technique for automatic initialization of the model pose and in Section 7.1 we show evaluation results using this technique in comparison with hand initialized poses.

Given an initial pose estimate, we refine it using chamfer matching by minimizing (1) using the Levenberg-Marquardt non-linear least squares algorithm. To update the parameters controlling the camera projection matrix M, we compute the Jacobian J for our camera parameters. We represent the camera rotation by the axis-angle representation $\omega = \theta\hat{n} = (\omega_x, \omega_y, \omega_z)$ and the camera position by the camera center $c = (c_x, c_y, c_z)$. We also allow the focal length f to vary and assume that the principal point (o_x, o_y) is at the center of each test image. The camera parameter vector is thus specified by $\gamma = (\omega_x, \omega_y, \omega_z, c_x, c_y, c_z, f)$. Details for how the Levenberg-Marquardt algorithm can be used to perform this alignment can be found in [29, §6.2.2]. Figure 3 shows an example of the initial manual align-

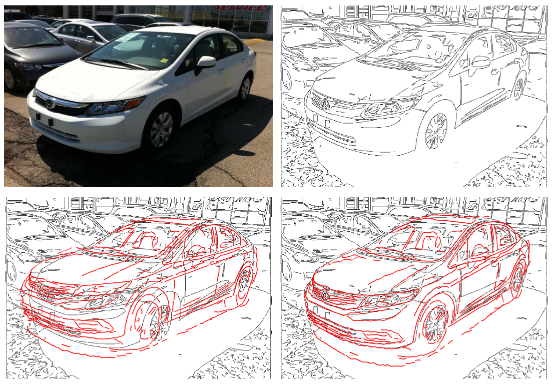

Figure 3. 3D chamfer matching. The top row shows the car image and the edges detected on the image. The bottom row shows the projected 3D edges overlaid on a test image after manual initialization and after refinement.

ment followed by the automatic alignment obtained with 3D chamfer matching.

One detail that was left unspecified in the chamfer matching formula (1) was the form of the distance function $d(p, q)$. The most common choice for this function is the squared Euclidean distance $d(p, q) = \|p - q\|^2$, but other, more robust or discriminative functions are possible, which we describe below.

Perpendicular distance. Instead of minimizing the Euclidean distance $d(p, q) = \|p - q\|^2$, which fixes the association between model and test points, we can instead use a *perpendicular distance* $d_\perp = n \cdot (p - q)$ and $n = (p - q)/\|p - q\|$, where n remains fixed during the Jacobian computation. This allows points to "slide" along curves in the direction perpendicular to the current error. In our experiments, we have found that this formula results in much faster convergence.

Robust matching. To make our alignment process more robust to missing model points, we use a Huber function in our distance measurements. The Huber function is a quadratic function for $\|p - q\| < 10$ pixels and a linear penalty for larger deviations.

Orientation distance. Since most edge pixels belong to long smooth contours, they have an associated orientation. For two shapes to align properly, we not only want close alignment of model to image edges, but also the orientation of the edges to be the same. For example, we do not want a vertical model edge to align well with a region with many horizontal edges even though the distance to the nearest image edge is very small. To penalize such deviations, we add the orientation metric introduced in [26],

$$D_\theta = \frac{1}{N_\mathrm{M}} \sum_{i=1}^{N_\mathrm{M}} |\theta(p_k) - \theta(q_i)|, \tag{2}$$

where $\theta(p_k)$ is the orientation of the closest edge point found in the original chamfer match (1), $\theta(q_i)$ is the orientation of the projected model point $q_i = \mathcal{P}MP_i$, computed from its neighboring projected points, and $|\theta_1 - \theta_2|$ measures the angular difference modulo π.

5. Verification

A model correctly aligned to an image will have a low chamfer distance. We use the average chamfer distance D_c, and the average orientation distance D_θ as features in a logistic regression classifier to predict the correct model for each test image. The average chamfer distance is computed by choosing the nearest image point for each projected model point and summing the robust distance functions, divided by the number of model points N_M to make the scores invariant to the number of model edges. The orientation distance is the average difference in normal orientation of the corresponding points.

We normalize all of the features to have zero mean and unit variance for classification. The logistic regression outputs a probability that the aligned image is of the specific make and model,

$$P(Y = 1|D, \beta) = \frac{1}{1 + e^{-D_\beta}}, \quad \text{with} \tag{3}$$

$$D_\beta = \beta_0 + \beta_1 D_c + \beta_2 D_\theta. \tag{4}$$

To estimate the best β parameters for each car model, we use leave-one-out cross validation (LOOCV) and find

$$\beta^* = \operatorname*{argmax}_\beta \sum_t \ln P(Y_t|D_t, \beta) - \frac{\lambda}{2}||\beta||^2, \tag{5}$$

where $Y_t = 1$ for positive training examples and $Y_t = 0$ for negative examples.

6. Detection and Pose Estimation

In this section we describe two methods for automatically detecting the pose and location of the cars in the test images. The first is an extension of the work in [31]. The second is a novel technique based on finding the ellipses corresponding to the wheels of the cars and then using the ellipse bitangents, along with the 3D model, to accurately estimate the car pose. These pose estimation algorithms are used to automatically initialize the chamfer matching algorithm described in Section 4.

6.1. Aspect layout detection and pose estimation

We build upon the work in [31] for initializing the pose for chamfer matching. The authors use the aspect layout of the object to detect the object in the image as well as provide a coarse estimate of the viewpoint of the object. However, this estimate alone is not enough to initialize the chamfer matching as we need the full 3D pose of the model.

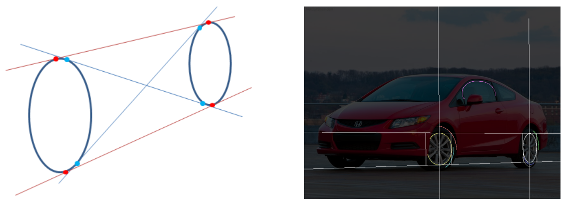
(a) (b)

Figure 4. 3D pose estimation of the car using two detected wheels: (a) bitangents for a pair of ellipses; (b) an example image showing detected ellipses on the wheel rims, the corresponding bitangents and four points of bitangency used to estimate the full 3D pose.

To estimate the full 3D pose, we start with the initial estimates of the viewpoint and the detection region of the car. We then perform a local search around the initial viewpoint estimates in the pose space (optimizing over the focal length, camera center, 3D rotation and translation) to align the model with the car in the test image. We build the scoring function for this as follows. We extract the model edges corresponding to a particular viewpoint, as well as the test image edges that fall inside the bounding box spanned by the model edges. We blur these edge images, perform dense descriptor matching [2] on them and compute the match distance. The dense descriptor matching provides an overlap distance between the model and the test edges while allowing for some slack. We found this metric to be a good predictor of the rough initial alignment. We report the full 3D pose corresponding to the lowest match distance as the initial pose estimate.

6.2. Ellipse-based pose estimation

While developing a general-purpose feature-based detection and pose estimation technique has some advantages, an alternative approach involves using category-specific geometric features. Cars and trucks contain a variety of structural features such as headlights and taillights, grills, and logos, which could all potentially be used for detection and pose estimation. They also have bilateral symmetry [27], which for frontal or near-frontal views, can provide one degree of freedom – the rotation of the car around its vertical axis.

Amongst all these potential features, the wheels on cars and trucks may be the most prominent and easy to detect, at least for views where they are not too foreshortened (Figure 4b). The advantage of using wheels is that they provide highly accurate pose estimates because the full 3D pose is encoded in the elliptical shape of the projected wheel (hubcab) rims as well as their relative positions.

Consider the schematic illustration of a pair of wheels shown in Figure 4a. After detecting the ellipses corresponding to the two wheels (Figure 4b), we can find the *bitangents* that pass through the tops and bottoms of the two wheel rims and hence form a pair of parallel “horizontal” lines in 3D. Connecting the bitangency points with an orthogonal pair of lines provides us with the vertical vanishing point.[2]

It is well known that an orthogonal pair of *finite* vanishing points can be used to recover the orientation (3D rotation) of an object or coordinate frame, assuming a known image optical center [29]. Unfortunately, we often see cars under configurations, e.g., true side views or any view at car level, where one or both vanishing points are infinite (the bitangent lines are parallel). Fortunately, the shape of the ellipses can come to our rescue, since the foreshortening (eccentricity) can be used to recover the rotation. In brief, the horizontal and vertical vanishing points along with the ellipse shape can be used to compute a homography that maps any ellipse into a unit square. The derivative (Jacobian) of this homography at the image origin provides two axes that encode the rotation matrix. Finally, the focal length can be determined from the rotation matrix and the original vanishing points. We provide more detailed mathematical derivation in our supplementary material [10].

In our application, we can use a stronger cue to recover a more accurate pose estimate. Since we have built a 3D model for each of the cars, we use four known 3D points corresponding to the wheel rim tops and bottoms (for each car model) to compute a full 3D pose estimate. The details here are even simpler than the model-free pose estimation (essentially using a direct linear transform) and are also described in the supplementary materials [10].

7. Evaluation

For our evaluation, we collected 190 test images from the web, 20 images for each of eight Honda models and 30 other random cars. The eight models span various car categories such as sedan, SUV, hatchback, mini-van and truck, and are shown in Figure 9 (Appendix A) of our supplementary [10]. We avoided images taken against uniform backgrounds, since we wanted to test our system operating “in the wild”, with cars shot in front of varied textured backgrounds from different viewpoints and with lots of body reflections.

In order to evaluate the 3D chamfer matching and verification system described in Sections 4 and 5, we first present results where the initial poses for matching are hand generated. For each test image, we manually aligned the 3D model using an interactive viewer. In Section 7.1 we present comparative results with our automatic pose initialization algorithms described in Section 6.

Some sample results from our chamfer matching algorithm are shown in Figure 5. Alignments to positive examples are shown in green and alignments to negative examples are shown in red. Notice that, as expected, the alignment for the positive examples is usually much better than

[2] As shown in Figure 4a, we discard the blue bitagents since the ellipse centers lie on different sides of these lines.

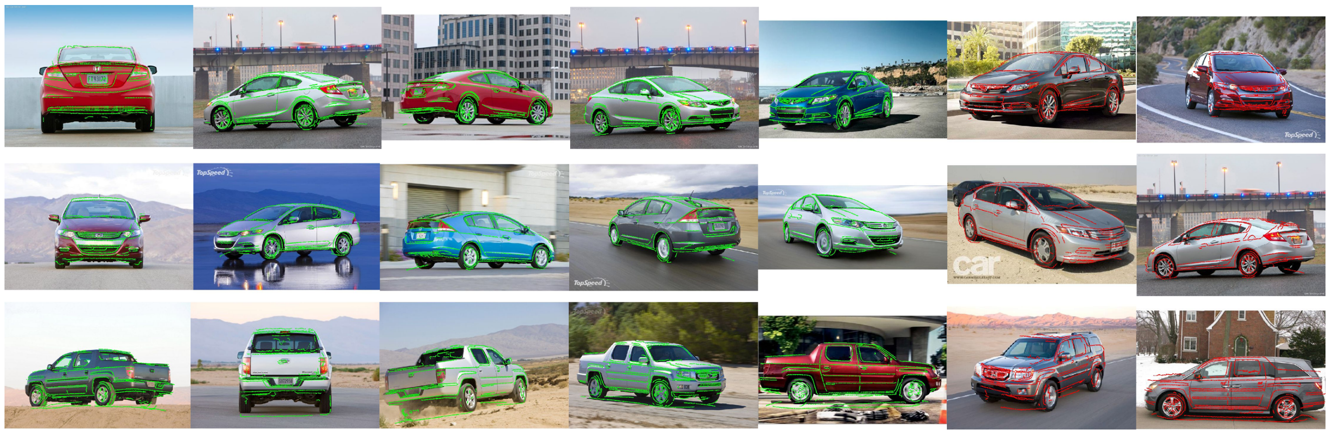
Figure 5. Some sample alignment results of 3D curves with positive examples (green) and negative examples (red) for a few of the car models (row-major order). Note that even though the negative car examples look similar to positive examples in many cases, the fits are much better for the positive examples.

the alignment for the negative examples, even though in some cases the cars look quite similar, e.g., the Civic Sedan, the Civic Coupe, and the Insight.

In addition to evaluating our chamfer matching algorithm, we also used our manual initializations to test two baseline methods. The first is a classic feature-based method, where keypoint descriptors [19, 2] are matched between a test image and a model image. For each test image, we choose the closest model viewpoint and score the image as the number of inlier matches that satisfy the ratio test. Due to absence of texture and presence of specular reflections, the interest points locations are not repeatable, leading to poor matching performance. The second baseline we evaluated is affine matching between the 2D model and 2D test image based curves. For a test image, we consider the least squares affine transformation between the 2D curves of the closest model viewpoint and the 2D projection of the manual alignment. The transformation is further refined using affine chamfer matching and the score for a test image is the average chamfer distance. While affine matching is often able to align to the coarse shape of the car, our results show that it is unable to discriminate the finer details needed for car recognition.

We evaluate the performance of these different matching techniques by generating Precision/Recall curves as shown in Figure 6. For each of our eight models, we compute probabilities as described in the previous section across all 190 test images. We then generate the P/R curves by ranking the images based on these probabilities and counting the true positive and false positive rates.

Table 1 summarizes these P/R curves using mean area under the Precision/Recall curve (AUC) numbers. The AUC values averaged across all eight models are 0.189 and 0.256 for the 2D keypoint and chamfer baselines, and 0.507, 0.767, 0.799, for the chamfer distance, orientation distance and chamfer+orientation measure with 3D alignment. These results indicate that the 3D chamfer distance performs better than both baselines. However, spurious edges in the interior of the car or in the background can corrupt the chamfer metric, leading to poor accuracy for some models. The orientation distance on the other hand is a better metric as it measures local agreement in shape of the matching curves. Combining the chamfer and orientation distances generally improves the accuracy.

Model	2D key	2D chamf	3D chamf	3D orient	3D comb
CivC	0.175	0.173	0.624	0.862	0.844
CivS	0.257	0.383	0.655	0.636	0.668
InsH	0.155	0.232	0.521	0.838	0.869
Ody	0.281	0.180	0.563	0.919	0.912
Pil	0.124	0.434	0.437	0.559	0.709
Rid	0.114	0.406	0.681	0.801	0.873
Ele	0.240	0.117	0.279	0.896	0.867
Fit	0.168	0.119	0.292	0.624	0.647
Avg	0.189	0.256	0.507	0.767	0.799

Table 1. The mean area under the precision-recall curve (AUC) for the two baselines 2D keypoint and chamfer matching, and the three different cost functions for 3D alignment, namely chamfer, orientation and chamfer+orientation.

Table 2 shows the confusion table generated by selecting the most likely model for each of our 190 test images. To reject the "other" class, we use leave-one-out validation to choose a threshold based on percentage recall for each model. We report the classification accuracy at an operating point of 95% recall, where a car is classified as "other" only if it is rejected by all the models. The Civic Coupe and Sedan are often confused with each other whereas the Insight, Element and Fit are accurately recognized more often. The Pilot is sometimes confused with the Ridgeline as they look similar from the front. The cars in the "other" category also get frequently confused with the eight car categories.

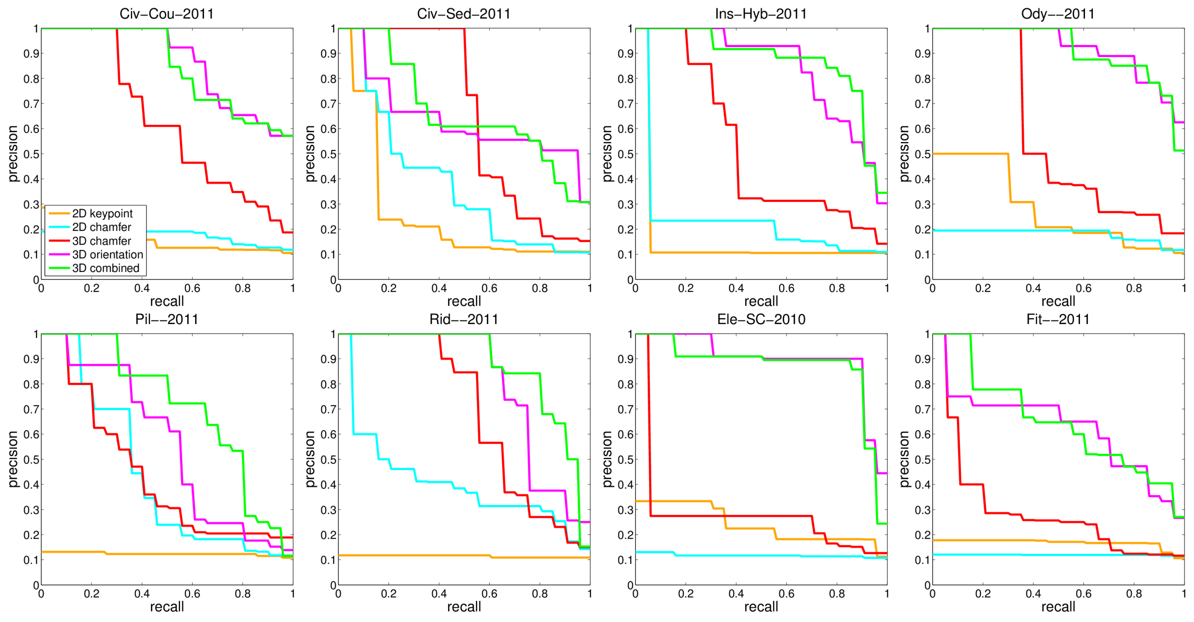

Figure 6. Our recognition results for eight car models on 190 test images. We compare the baselines of 2D keypoint and chamfer matching versus the different alignment measures using our 3D model: chamfer distance, orientation distance and both combined. There are 20 positives images per car and 30 none-of-the-above images.

	CivC	CivS	InsH	Ody	Pil	Rid	Ele	Fit	other
CivC	**0.85**	0.10	-	-	-	-	-	-	0.05
CivS	0.05	**0.75**	0.05	-	-	-	-	0.10	0.05
InsH	-	-	**0.90**	-	-	-	-	0.05	0.05
Ody	-	-	-	**0.90**	-	-	-	0.05	0.05
Pil	-	-	-	-	**0.85**	0.10	-	-	0.05
Rid	-	-	-	-	0.10	**0.85**	-	-	0.05
Ele	-	-	-	-	0.05	-	**0.95**	-	-
Fit	-	-	-	-	-	-	-	**0.95**	0.05
other	0.03	0.15	0.03	-	0.30	0.03	0.03	0.03	**0.40**

Table 2. Confusion matrix. Each row shows the results for 20 test images of one car model and 30 test images of category other, and each column shows which cars it was classified as using our full system. We use a threshold set at 95% recall for each car model to reject the “other” category.

7.1. Automatic pose initialization

In this section, we present results from automatically initializing the pose estimates using the algorithms described in Section 6. We use the estimates from the layout model-based algorithm as well as the bitangent-based algorithm for initialization. The chamfer matching algorithm runs the matching with each of these initial estimates and retains the one with the lowest match distance. In our experiments, we found that the chamfer matching scheme preferred the bitangent-based estimates more often than the layout model-based estimates, which is reasonable, as the layout-based estimates are obtained using a greedy approach that is more prone to be stuck in a local minima. However, the layout model-based estimates were useful when the cars wheels were not visible.

	CivC	CivS	Ins	Ody	Pil	Rid	Ele	Fit	Avg
auto	0.67	0.41	0.74	0.51	0.46	0.79	0.66	0.55	0.6
hand	0.84	0.67	0.87	0.91	0.71	0.87	0.87	0.65	0.8

Table 3. The mean area under the precision-recall curve (AUC) for automatic initialization (auto) versus manual initialization (hand) using the combined chamfer and orientation distance metric.

	CivC	CivS	InsH	Ody	Pil	Rid	Ele	Fit	other
CivC	**0.55**	0.10	0.20	-	0.05	-	0.05	0.05	-
CivS	0.20	**0.45**	-	0.05	0.05	-	0.15	0.10	-
InsH	-	-	**0.90**	-	-	-	0.10	-	-
Ody	0.05	0.10	0.10	**0.30**	0.25	0.05	0.15	-	-
Pil	-	0.15	0.05	0.05	**0.60**	0.10	0.05	-	-
Rid	-	0.05	-	0.05	-	**0.80**	0.05	0.05	-
Ele	0.05	-	-	0.10	0.05	-	**0.80**	-	-
Fit	-	0.05	-	0.10	-	-	0.10	**0.75**	-
other	-	0.23	0.03	0.23	0.24	0.07	0.17	0.03	**-**

Table 4. Confusion matrix for classification using automatic initialization. Each row shows the results for 20 test images of a car model and 30 images of the *other* category, and each column shows the predictions from our method. We use a threshold set at 95% recall for each car model to reject the *other* category.

In Table 3 we summarize the mean area under the Precision/Recall curve (AUC) numbers for chamfer matching using the chamfer+orientation distance metric with the automatically initialized poses versus the hand initialized ones (which is same as the last column in Table 1.) In Table 4 we also present the confusion matrix for the automatically initialized matching, similar to the one in Table 2. The automatic pose estimates perform reasonably well but they are not as good as the hand initialized ones.

8. Conclusion

In this paper, we have developed an approach for verifying the make and model of a car from a single image taken from an arbitrary viewpoint using view-based 3D curve models. Our experiments show that these models can be accurately aligned to test images after which chamfer distance and orientation distance can be used to recognize the car make and model under wide range of viewpoints. We also presented two automatic methods for recovering the full 3D pose of the car in the test image which is used to initialize the chamfer matching based alignment method.

In the future, we plan to incorporate appearance-based features into our view-based 3D models and extend our approach to automatically learn semantic parts of cars that can be highly discriminative for make and model recognition, such as car logos, grill, headlights and taillights. (Some preliminary results on incorporating taillights can be found in [10].) We will explore extensions that combine appearance and geometric features with the goal of improving accuracy in detection, pose-estimation and make and model recognition of a car from a single image.

References

[1] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *TPAMI*, 24(4):509–522, April 2002. 2

[2] M. Brown, G. Hua, and S. Winder. Discriminative learning of local image descriptors. *TPAMI*, 33(1):43–57, January 2011. 5, 6

[3] T. Cootes, D. Cooper, C. Taylor, and J. Graham. Active shape models—their training and application. *CVIU*, 61(1):38–59, January 1995. 2

[4] L. Dlagnekov and S. Belongie. Recognizing cars. *UCSD, La Jolla, CA, TR. CS2005-0833*, 2005. 1, 2

[5] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL visual object classes (VOC) challenge. *IJCV*, 88(2):147–168, July 2010. 2

[6] R. Farrell, O. Oza, N. Zhang, V. Morariu, T. Darrell, and L. Davis. Birdlets: Subordinate categorization using volumetric primitives and pose-normalized appearance. In *ICCV*, 2011. 1

[7] J. Krause, M. Stark, J. Deng, L. Fei-Fei 3D Object Representations for Fine-Grained Categorization . In *4th IEEE Workshop on 3D Representation and Recognition (ICCV 2013)*, December 2013. 1

[8] D. Glasner, M. Galun, S. Alpert, R. Basri, and G. Shakhnarovich. Viewpoint-aware object detection and pose estimation. In *ICCV*, 2011. 1, 2, 3

[9] D. Hoiem and S. Savarese. Representations and Techniques for 3D Object Recognition and Scene Interpretation. Morgan & Claypool, 2011. 2

[10] E. Hsiao, S. Sinha, K. Ramnath, S. Baker, L. Zitnick, and R. Szeliski. Car make and model recognition using 3d curve alignment. Technical Report MSR-TR-2014-9, Microsoft Research, February 2014. 5, 8

[11] D. P. Huttenlocher, G. Klanderman, and W. Rucklidge. Comparing images using the Hausdorff distance. *TPAMI*, 15(9):850–863, September 1993. 2

[12] D. P. Huttenlocher and S. Ullman. Recognizing solid objects by alignment with an image. *IJCV*, 5(2):195–212, 1990. 2

[13] D. M. Jang and M. Turk. Car-rec: A real time car recognition system. In *WACV*. IEEE Computer Society, January 2011. 2

[14] A. Laurentini. The visual hull concept for silhouette-based image understanding. *PAMI*, pages 150–162, 1994. 3

[15] M. Leotta and J. Mundy. Predicting High Resolution Image Edges with a Generic, Adaptive, 3-D Vehicle Model. In *CVPR*, 2009. 2

[16] C. Li, C. Gatenby, L. Wang, and J. Gore. A robust parametric method for bias field estimation and segmentation of mr images. In *CVPR*, 2009. 2

[17] Y. Li, L. Gue, and T. Kanade. Robustly aligning a shape model and its application to car alignment of unknown pose. *TPAMI*, 33(9):1860–1876, September 2011. 1, 2

[18] M.-Y. Liu, O. Tuzel, A. Veeraraghavan, and R. Chellappa. Fast directional chamfer matching. In CVPR, 2010. 2

[19] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, November 2004. 6

[20] C. Lu et al. Contour based object detection using part bundles. *CVIU*, 114(7):827–834, July 2010. 2

[21] M. Ozuysal, V. Lepetit, and P. Fua. Pose estimation for category specific multiview object localization. In *CVPR*, 2009. 1, 2, 3

[22] N. Payet and S. Todorovic. From contours to 3d object detection and pose estimation. In *ICCV*, November 2011. 2

[23] G. Pearce and N. Pears. Automatic make and model recognition from frontal images of cars. In *AVSS*, September 2011. 1, 2

[24] V. Petrovic and T. Cootes. Analysis of features for rigid structure vehicle type recognition. In *BMVC*, 2004. 1

[25] H. Schneiderman and T. Kanade. Object detection using the statistics of parts. *IJCV*, 56(3):151–177, February 2004. 2

[26] J. Shotton, A. Blake, and R. Cipolla. Contour-based learning for object detection. In *ICCV*, volume 1, pages 503–510, October 2005. 2, 4

[27] S. N. Sinha, K. Ramnath, and R. Szeliski. Detecting and Reconstructing 3D Mirror Symmetric Objects. In *ECCV*, 2012. 5

[28] H. Su, M. Sun, L. Fei-Fei, and S. Savarese. Learning a dense multi-view representation for detection, viewpoint classification and synthesis of object categories. In *ICCV*, 2009. 1, 2

[29] R. Szeliski. *Computer Vision: Algorithms and Applications*. Springer, New York, 2010. 3, 5

[30] A. Thomas, V. Ferrari, B. Leibe, T. Tuytelaars, B. Schiele, and L. Van Gool. Towards multi-view object class detection. In *CVPR*, volume 2, pages 1589–1596, June 2006. 1, 2

[31] Y. Xiang and S. Savarese. Estimating the aspect layout of object categories. In *CVPR*, 2012. 1, 3, 4