Car Make and Model Recognition using 3D Curve Alignment

Edward Hsiao, Sudipta N. Sinha, Krishnan Ramnath,
Simon Baker, Larry Zitnick, and Richard Szeliski

February 2014
Technical Report
MSR-TR-2014-9

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. We also build an appearance model of taillights which is used as an additional cue. Our approach is able to verify the exact make and model of a car over a wide range of viewpoints and background clutter.

Microsoft Research
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052
http://www.research.microsoft.com

1 Introduction

[**Note:** *This technical report describes research done during Edward Hsiao's summer internship at Microsoft Research in 2011 and was authored during the fall of 2011. A conference version of this work, presented at WACV 2014 (Ramnath et al. 2014), contains newer pose estimation techniques and updated experimental results, but not some of the appearance-based matching described in this report. The two appendices in this report are new and provide additional details on the pose estimation component and additional experimental results relating to (Ramnath et al. 2014).*]

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 in this paper is therefore: is car recognition an easily solvable example of subordinate category recognition (Rosch *et al.* 1976, Farrell *et al.* 2011) 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 (Petrovic and Cootes 2004, Dlagnekov and Belongie 2005, Santos and Correia 2009, Pearce and Pears 2011). In parallel, work on general object category detection and recognition is starting to simultaneously recognize the rough pose of the object (Thomas *et al.* 2006, Torralba *et al.* 2007, Savarese and Fei-Fei 2007, Sun *et al.* 2009, Su *et al.* 2009, Ozuysal *et al.* 2009, Li *et al.* 2011, Glasner *et al.* 2011). 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 long term goal is to develop a fully automated recognition system, in this paper we focus on the verification stage, i.e., determining the exact model given a rough initial pose estimate for the car.

In the following, we describe our processing pipeline and matching algorithms for solving this

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.

problem. We begin with a review of previous work in this field. We follow this with an overview of the problem and justify our choice of view-based 3D space curves as the basis of our system. We then describe how we build 3D visual hulls to serve as our initial 3D model (Section 4) and how we refine our 3D space curves using three-view matching (Section 5). 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 6). In Section 7, we show how the location and appearance of taillights can be used to improve our recognition rates. Section 8 describes how we combine our chamfer and appearance model scores to perform recognition using logistic regression. Finally, we present our experimental results on six different car models and close with a discussion of our results and 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 (Szeliski 2010). Hoiem and Savarese (Hoiem and Savarese 2011) survey this topic as well as a the large number of papers recently 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 (Huttenlocher and Ullman 1990, Lowe 1991, Ullman and Basri 1991). 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 (Thomas *et al.* 2006, Savarese and Fei-Fei 2007, Sun *et al.* 2009, Su *et al.* 2009).

Work in detecting cars, often from fixed viewpoints such as side views, has been evolving in parallel (Schneiderman and Kanade 2004). Detecting cars (along with people, bicycles, and other objects) has become a central component of the PASCAL VOC (Visual Object Category) challenge (Everingham *et al.* 2010).

More recently, researchers have started to combine object detection with pose estimation (Li *et al.* 2009b, Ozuysal *et al.* 2009, Liebelt and Schmid 2010, Gu and Ren 2010, Li *et al.* 2011, Glasner *et al.* 2011). For example, Ozuysal *et al.* (Ozuysal *et al.* 2009) use an initial bounding box of a car to select a view-specific classifier to refine the hypothesis, while Glasner *et al.* (Glasner *et al.* 2011) vote for potential pose parameters and then refine these using view-specific SVMs. Li *et al.* (Li *et al.* 2011) 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.* (Shotton *et al.* 2005) 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 (Liu *et al.* 2010).) Lu *et al.* (Lu *et al.* 2010) extract edgels and link them to form contours, which are then grouped into part bundles and matched using shape context similarity (Belongie *et al.* 2002). Most recently, Payet and Todorovic (Payet and Todorovic 2011) 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. (Dlagnekov and Belongie 2005, Santos and Correia 2009, Pearce and Pears 2011). A notable exception is the recent paper of Jang and Turk (Jang and

Turk 2011), 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 provide better results than competing approaches.

3 Problem formulation

We build our car recognition models from a set of V images taken at regular intervals around each car. Figure 1 shows three of the 36 training images used to build one of our models. The test images we use are taken in natural settings with complex cluttered backgrounds, as seen in Figure 7.

For the image sequences we used, feature-based approaches failed to find enough repeatable interest points or regions to build a recognition system (Section 9). This is because most car surfaces are characterized by textureless regions with large amounts of reflections, which are often extracted as features.

Curve extraction and matching using 2D alignment and chamfer matching (Shotton *et al.* 2005, Liu *et al.* 2010) also run into problems, even when using non-rigid alignment techniques such as Active Shape Models (ASMs) (Cootes *et al.* 1995, Li *et al.* 2009a). The large variation in curve position due to 3D viewpoint changes make it difficult to simultaneously obtain good registrations between the model and test images while still discerning the subtle shape differences that differentiate one car model from another (Section 9).

3.1 3D Curve Model

Since active shape models will in general not reflect the true 3D transformation of rigid car features, a better approach is to directly build a 3D model. This requires edges on the car to be represented by 3D space curves. We extract these curves from natural training images, rather than CAD models, since these better match the curves that are extracted from test images taken "in the wild". In the following, we discuss our approach to generate a 3D space curve model as well as our method to match them to new input images.

We represent the 3D curves in each view of the 3D car model by a set of N_{M} 3D points P_i. The goal of alignment (Section 6) is to recover the 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 the N_{T} 2D edge points in the test image, $\{p_k\}$. The optimal transformation M^* is the one that minimizes

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

where $d(p, q)$ is one of the 2D distance metrics discussed in Section 6. Here, the operator $\mathcal{P}$ projects 3D points onto a 2D plane, and the minimum distance over the test image points $\{p_k\}$ can be efficiently computed using a distance transform (Huttenlocher *et al.* 1993).[1]

Instead of constructing a single global 3D model, we construct a view-based model consisting of V separate 3D models. For a new test image, we choose the 3D points from the closest training image and align these to the test image using a rigid 3D perspective image transformation.[2] This has the advantage that subtle view-dependent features can be modeled, and the visibility of the curves is handled naturally.

The most common method for obtaining 3D points from 2D points is to first obtain correspondences in 2D using discriminative features such as SIFT and to then triangulate these points in 3D. For curves, however, computing point-to-point correspondences from adjacent images is a challenging problem. For this reason, we first build a visual hull model of each car (Section 4), 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 5).

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

[2] Our view-based approach is similar in spirit to the linear combination of approach of Ullman and Basri (Ullman and Basri 1991). However, since cars are often shot in close proximity with large perspective effects, we use a full 3D model and alignment framework.

4 Visual Hull for 3D Curves

Our training data consists of V images of each car model taken at regular angular intervals against a clean backgrounds. Such images are often available on manufacturers' Web site, and in a production system, would be captured with carefully calibrated cameras. Because in our case we did not have access to the intrinsic camera parameters, we automatically detected the ellipses corresponding to the wheels in the subset of images where these could be reliably detected and used 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 space curves by backprojecting the 2D image curves onto an automatically generated visual hull of the car (Laurentini 1994). First, we turn each input image into a binary silhouette using thresholding followed by morphological operations to clean up the outline and remove small holes due to highlights. Next, we build a voxel-based 3D model by intersecting all of the silhouettes in 3D, and project this model into each image to obtain an associated depth map. For a point p in image v, the 3D point P is obtained by backprojecting the point onto the visual hull using

$$P = dR_v^{-1}K_v^{-1}\tilde{p} - t_v, \tag{2}$$

where the camera matrix for view v is $M_v = K_v[R_v|t_v]$, $\tilde{p} = (x, y, 1)$ is the homogeneous representation of the 2D point, and the depth of the visual hull at point p is d.

The training images we currently use were shot against a white background, which makes the silhouette extraction for visual hull computation relatively easy. For sequences shot against textured backgrounds, we could use a 3D structure from motion system such as Bundler (Snavely *et al.* 2006) to build a 3D model of the background and then segment the car from this background (Glasner *et al.* 2011).

(a) (b) (c)

Figure 2: Three-view consistent edges: (a) original unfiltered edges; (b) 3D positions of edges in neighboring views; (c) the three-view consistent edges.

5 3D Curve Model Refinement

Many of the edges seen on cars arise from lighting effects such as reflections and specularities. These spurious edges introduce errors into the 3D chamfer matching score and need to be removed for robust recognition. The key insight to removing these edges is to notice that although spurious edges from neighboring views may be consistent in 2D, their locations in 3D are not consistent with the visual hull model, as shown in Figure 2a. We refine our 3D model by finding 3-view consistent edges using the 3D positions of the curves.

We begin with the 3D curve model generated in Section 4. To remove the spurious edges for view c, we choose a 3D point P_c and consider the neighboring left view l and right view r. For simplicity, let's first consider two-view consistent edges with the left view l.

We project the 3D point P_c^j from the center image into the left view using the camera matrix of the left view M_l, i.e., $p_{c,l}^j = \mathcal{P} M_l P_c^j$. If a projected point $p_{c,l}^j$ is not within some distance threshold τ_d to any edge point p_l^k in view l, the point is removed. The same is done for the right view. This is illustrated in Figure 3. For those points P_c^j that are retained, we refine their 3D positions by considering the nearest neighbor points $p_l^{k^*}$ and $p_r^{k^*}$. Given the original 2D point p_c^j, their corresponding neighbor points, $p_l^{k^*}$ and $p_r^{k^*}$, and the camera matrices from the neighboring views, we use the Direct Linear Transform (Szeliski 2010) to triangulate a more accurate 3D position for P_c^j.[3] We apply the above refinement technique for every point in every view of the model. An example of the resulting model can be seen in Figure 2b.

[3]The DLT provides sufficient accuracy since the baselines are nearby and uniformly spaced.

Figure 3: Projection of 3D points of center view in neighboring views.

6 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 initialize the pose using a technique that reliably determines the car orientation from a test image. While a variety of such techniques have been developed (Ozuysal *et al.* 2009, Li *et al.* 2011, Glasner *et al.* 2011), we perform a rough manual alignment between each model and each new test image to systematically analyze the benefits of using 3D curve alignment for car make and model recognition. In our evaluation section, we report on the sensitivity of our recognition results with respect to initial pose accuracy and show that our approach does not require very precise initial poses.

Given the 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 (Szeliski 2010, §6.2.2). Figure 4 shows an example of the initial manual alignment followed by the automatic alignment obtained

Figure 4: 3D chamfer matching, showing the projected 3D edges overlaid on a test image after manual initialization and after refinement.

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, as we now describe.

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

Euclidean vs. 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.

Oriented chamfer matching 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 (Shotton *et al.* 2005),

$$D_\theta = \frac{1}{N_{\mathrm{M}}} \sum_{i=1}^{N_{\mathrm{M}}} |\theta(p_k) - \theta(q_i)|, \tag{3}$$

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.

7 3D Appearance Model of Taillights

The appearance of a car's taillights are consistent across a make and model and are often very distinctive. The ability to extract these regions and to reproject them using a 3D model can constrain the viewpoint of the car as well as assist in identifying its make and model.

We model the appearance of the taillights with a Gaussian Mixture Model (GMM) on the a and b channels in L*a*b color space across all training images. We assume that the taillights of cars are orange to red in color, which is true for most cars. We do not use the L channel as we want the system to be robust to lighting conditions.

From manually labeled data, we learn a GMM, $P(X = FG)$ with two components and choose the dominant component as the taillight foreground model. The smaller component corresponds to the white and black portions of the taillights, which are not useful when learning a generic taillight detector. We model the background using a GMM $P(X = BG)$ with three components. A pixel in the image is classified as taillight if $P(X = FG) > P(X = BG)$.

To use the taillights for verification, we first identify them in the training images and position them on the 3D curve model (Figure 5a). We use the generic taillight detector described previously to identify potential taillight regions in the training images. For regions that are large enough, we backproject the boundary onto the visual hull to obtain the region in 3D. We then compute an appearance model for each taillight region separately to make the model more distinctive.

(a) (b) (c) (d) (e)

Figure 5: Taillight model: (a) the 3D location of the lights marked in blue; (b) projection onto a test image; (c) another test image; (d) likelihood ratio; (e) classification.

Given an image and the aligned 3D model, we verify the appearance of the taillight regions by projecting the boundary of the taillights into the test image. This projection defines the predicted taillight region to be verified. From this projection, we verify if this region is similar to our model by learning a GMM using the pixels inside the region and then comparing it to the model GMM of that region. Since the appearance model is a probability distribution (Figure 5d), we use the KL divergence to compare how similar the model and image distributions are. We use the Unscented Transform (Goldberger *et al.* 1993) to approximate the KL divergence of two GMMs.

We compute both the KL divergence from model to image, $D_{t1} = D_{KL(M||I)}$ as well as the KL divergence from image to model, $D_{t2} = D_{KL(I||M)}$.[4] We then use these values as features for classification. The reason we use both KL divergences is that the two color distributions can be significantly different.

Since there are usually multiple taillight regions, we combine the KL divergence scores of the different regions by weighting the KL divergence of each region based on its area.

8 Verification

A model correctly aligned to an image will have a low chamfer distance as well as satisfy the appearance of the taillights in the image. The features we use for classification are the average chamfer distance D_c, the average orientation distance D_θ, and the two KL divergence metrics for the taillight appearance, D_{t1} and D_{t2}. The average chamfer distance is computed by choosing

[4] We use D_{t1} and D_{t2} as the shorthand for the two "taillight" distances since they are more compact to write.

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. For images where the taillight are not visible, we use only the chamfer distance and orientation distance.

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

$$P(Y = 1|D, \beta) = \frac{1}{1 + e^{-D_\beta}}, \quad \text{with} \tag{4}$$

$$D_\beta = \beta_0 + \beta_1 D_c + \beta_2 D_\theta + \beta_3 D_{t1} + \beta_4 D_{t2}. \tag{5}$$

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

$$\beta^* = \underset{\beta}{\operatorname{argmax}} \sum_t \ln P(Y_t|D_t, \beta) - \frac{\lambda}{2}||\beta||^2, \tag{6}$$

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

9 Evaluation

We evaluated our car make and model verification system using 150 non-training test images, 20 for each of our 6 models and 30 other random cars, which we found using Internet search engines. We specifically avoided test images taken against uniform backgrounds, since we wanted to test our algorithm operating "in the wild" with cars shot in front of varied textured backgrounds, potentially with lots of body reflections.

For each image, we manually aligned the 3D model using an interactive viewer and then refined this initial pose using the 3D chamfer matching of Section 6. Some sample alignments for all 6 car models can be seen in Figure 7. 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 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.

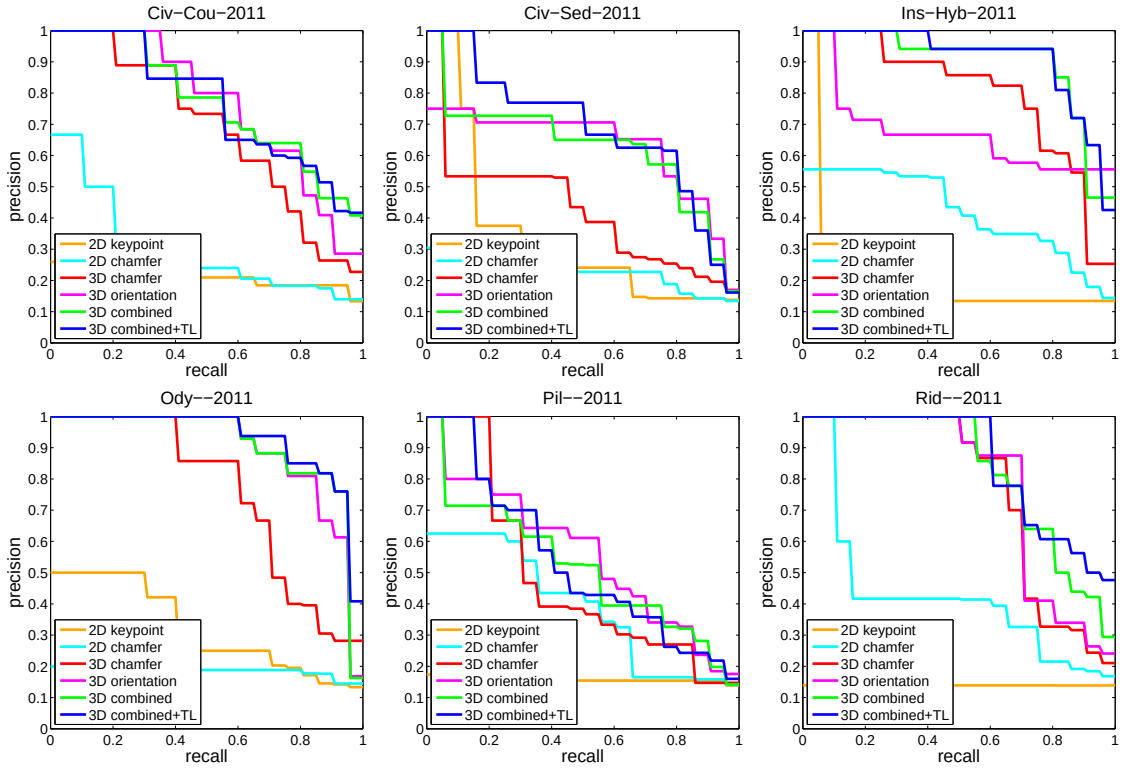

Figure 6: Our recognition results for 6 car models on 150 test images. We compare the baselines of 2D keypoint and chamfer matching versus the different alignment measures using our 3D model: chamfer distance, orientation distance, both, and both + taillight appearance. There are 20 positives images per car and 30 none-of-the-above images.

We also used our manual alignments to test two baseline methods. The first is a classic feature-based method, where keypoint features and descriptors 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 which satisfy the ratio test. We use the Laplace interest point detector and the DAISY descriptors for matching using the parameters described in (Lowe 2004, Brown *et al.* 2011). Due to absence of texture and presence of specular reflections, the interest points locations are not repeatable and the matching usually fails to identify salient features that would allow us to distinguish between car models. The results for keypoint-based recognition turned out to be barely above chance.

Model	2D key.	2D chamf.	3D chamf.	3D orient.	3D comb.	3D+TL
CivC	0.217	0.293	0.684	0.764	0.769	0.767
CivS	0.339	0.241	0.423	0.629	0.624	0.684
InsH	0.198	0.419	0.796	0.671	0.898	0.909
Ody	0.318	0.185	0.752	0.892	0.908	0.928
Pil	0.158	0.390	0.480	0.555	0.513	0.531
Rid	0.139	0.422	0.763	0.781	0.823	0.851
Average	0.228	0.325	0.650	0.715	0.756	0.778

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

We also tried affine matching between the 2D model and 2D test image 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 the different alignment techniques by generating Precision/Recall curves as shown in Figure 6. For each of our six models, we compute probabilities as described in the previous section across all 150 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 six models are 0.228 and 0.325 for the 2D keypoint and chamfer baselines, and 0.650, 0.715, 0.756, and 0.778 for the chamfer distance, orientation distance, chamfer+orientation and chamfer+orientation+taillights measure with 3D alignment. As you can see, we perform significantly better than the 2D baselines and get a performance boost from each of our improvements.

Figure 7: Some sample alignments of 3D curves with some positive examples (green) and negative examples (red) for each of the 6 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.

Table 2 shows the confusion table generated by picking the most likely model for each of our 150 test images. To reject the "other" class, we use LOOCV to choose a threshold based on percentage recall of each model car. We report the classification performance at an operating point of 90% recall, where a car is classified as "other" only if it is rejected by all models. As you can see, the Civic Sedan is most easily confused with other models, while the Insight and Ridgeline are both highly distinctive. Even though we only report results on six car categories, you can see that the task of reliably discriminating among these six categories is already quite challenging.

Looking at these results, we see that the chamfer distance by itself performs reasonably, but

	CivC	CivS	InsH	Ody	Pil	Rid	other
CivC	**0.85**	0.05	-	-	-	-	0.10
CivS	0.10	**0.70**	-	0.10	-	-	0.10
InsH	-	-	**0.90**	-	-	-	0.10
Ody	-	-	-	**0.85**	0.05	0.05	0.05
Pil	-	-	-	-	**0.85**	0.05	0.10
Rid	-	-	-	-	0.05	**0.90**	0.05
other	0.03	0.13	-	-	0.30	0.07	**0.47**

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 90% recall for each car model to reject the "other" category.

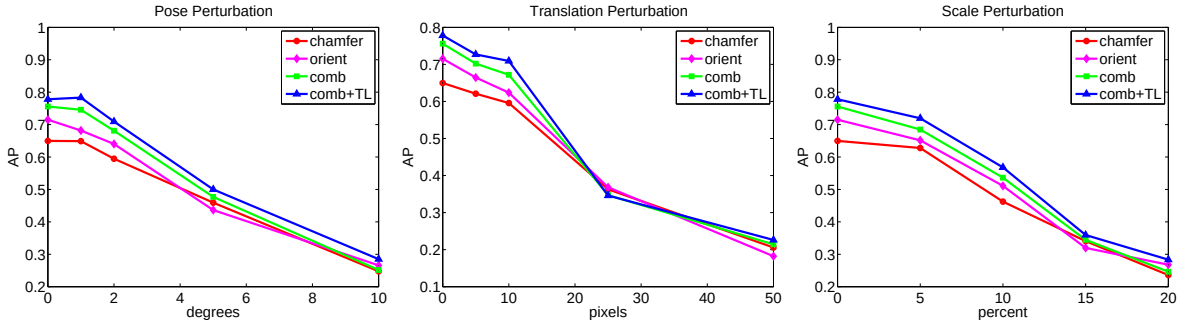

Figure 8: We analyze the sensitivity of our approach to the initial pose by adding random perturbations to the manual alignment. We perturb the pose in rotation, translation and scale.

since there are many edges in the interior of the car and clutter edges on the background, the chamfer distance alone is not sufficient for verification. The orientation distance on the other hand is a better metric, since it measures how well the points locally match in overall shape. Combining the chamfer distance and orientation distances generally improves the results, as does the addition of the taillight appearance model.

To analyze the sensitivity of our approach to the alignment obtained from a car pose estimation system, we randomly perturbed our manual alignment in rotation, translation and scale. Given

that the manual alignment is already fairly rough as shown in Figure 4, our method can handle an additional 2 to 3 degrees in rotation, 10 pixels in translation and 5% of scale change, as shown in Figure 8.

10 Conclusion

In this paper, we have developed a view-based 3D model for verifying the make and model of a car from a single image taken from an arbitrary viewpoint. Our approach combines the benefits of using features from nearby views with the ability of a 3D model to address views significantly different from the model images. We construct our 3D curve models by backprojecting edge points onto automatically generated visual hulls reconstructed from silhouettes, which we then refine by matching curves in their two neighboring training images.

Our experiments demonstrate that the estimated depths of the curves are accurate enough for good alignment to novel images. Although the chamfer distance and orientation distance provide reasonable verification performance, an appearance model for the taillights of the car can increase the robustness of the system. Our results demonstrate that our model can verify the make and model of a car under arbitrary viewpoints.

The biggest planned extension to our work is to incorporate an automatic detection and high-accuracy pose estimation stage (Ozuysal *et al.* 2009, Li *et al.* 2011, Glasner *et al.* 2011). Another direction involves learning discriminative features to add to the raw chamfer matching and taillight appearance model scores. We could also use curve-to-curve instead of point-based matching metrics, which would allow us to weight curves that are more discriminative (i.e., increase differentiation between models) more highly. Finally, we would like to use additional information such as logos and text on the car to boost recognition.

References

Belongie, S., Malik, J., and Puzicha, J. (2002). Shape matching and object recognition using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *24(4)*, 509–522.

Brown, M., Hua, G., and Winder, S. (2011). Discriminative learning of local image descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *33(1)*, 43–57.

Cootes, T., Cooper, D., Taylor, C., and Graham, J. (1995). Active shape models—their training and application. *Computer Vision and Image Understanding*, *61(1)*, 38–59.

Dlagnekov, L. and Belongie, S. (2005). Recognizing cars. *UCSD, La Jolla, CA, TR. CS2005-0833*, .

Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., and Zisserman, A. (2010). The PASCAL visual object classes (VOC) challenge. *International Journal of Computer Vision*, *88(2)*, 147–168.

Farrell, R., Oza, O., Zhang, N., Morariu, V., Darrell, T., and Davis, L. (2011). Birdlets: Subordinate categorization using volumetric primitives and pose-normalized appearance. In *13th International Conference on Computer Vision (ICCV 2011)*, Barcelona, Spain.

Glasner, D., Galun, M., Alpert, S., Basri, R., and Shakhnarovich, G. (2011). Viewpoint-aware object detection and pose estimation. In *13th International Conference on Computer Vision (ICCV 2011)*, Barcelona, Spain.

Goldberger, J., Gordon, S., and Greenspan, H. (1993). An efficient image similarity measure based on approximations of kl-divergence between two gaussian mixtures. In *ICCV*.

Gu, C. and Ren, X. (2010). Discriminative mixture-of-templates for viewpoint classification. In Daniilidis, K., Maragos, P., and Paragios, N., editors, *Computer Vision – ECCV 2010*, pages 408–421, Heidelberg, Springer.

Hoiem, D. and Savarese, S. (2011). *Representations and Techniques for 3D Object Recognition and Scene Interpretation*. Morgan & Claypool.

Huttenlocher, D. P. and Ullman, S. (1990). Recognizing solid objects by alignment with an image. *International Journal of Computer Vision*, *5(2)*, 195–212.

Huttenlocher, D. P., Klanderman, G., and Rucklidge, W. (1993). Comparing images using the Hausdorff distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *15(9)*, 850–863.

Jang, D. M. and Turk, M. (2011). Car-rec: A real time car recognition system. In *IEEE Workshop on Applications of Computer Vision (WACV 2011)*, Kona, Hawaii, IEEE Computer Society.

Laurentini, A. (1994). The visual hull concept for silhouette-based image understanding. *PAMI*, , 150–162.

Li, C., Gatenby, C., Wang, L., and Gore, J. (2009a). A robust parametric method for bias field estimation and segmentation of mr images. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009)*, Miami Beach, FL.

Li, H., Shen, T., and Huang, X. (2009b). Global optimization for alignment of generalized shapes. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009)*, Miami Beach, FL.

Li, Y., Gue, L., and Kanade, T. (2011). Robustly aligning a shape model and its application to car alignment of unknown pose. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *33(9)*, 1860–1876.

Liebelt, J. and Schmid, C. (2010). Multi-view object class detection with a 3d geometric model. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2010)*, San Francisco, CA.

Liu, M.-Y., Tuzel, O., Veeraraghavan, A., and Chellappa, R. (2010). Fast directional chamfer matching. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2010)*, San Francisco, CA.

Lowe, D. G. (1991). Fitting parameterized three-dimensional models to images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *13(5)*, 441–450.

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

Lu, C. *et al.*. (2010). Contour based object detection using part bundles. *Computer Vision and Image Understanding*, *114(7)*, 827–834.

Ozuysal, M., Lepetit, V., and Fua, P. (2009). Pose estimation for category specific multiview object localization. In *CVPR*, IEEE.

Payet, N. and Todorovic, S. (2011). From contours to 3d object detection and pose estimation. In *13th International Conference on Computer Vision (ICCV 2011)*, Barcelona, Spain.

Pearce, G. and Pears, N. (2011). Automatic make and model recognition from frontal images of cars. In *8th IEEE International Conference on Advanced Video and Signal-Based Surveillance (AVSS)*, Klagenfurt, IEEE Computer Society.

Petrovic, V. and Cootes, T. (2004). Analysis of features for rigid structure vehicle type recognition. In *BMVC*.

Ramnath, K., Sinha, S., Szeliski, R., and Hsiao, E. (2014). Car make and model recognition using 3d curve alignment. In *IEEE Winter Conference on Applications of Computer Vision (WACV 2014)*, Steamboat Springs, CO, IEEE Computer Society.

Rosch, E. *et al.*. (1976). Basic objects in natural categories. *Cognitive Psychology*, *8(3)*, 382–439.

Santos, D. and Correia, P. L. (2009). Car recognition based on back lights and rear view features. In *10th Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS '09)*, London, IEEE Computer Society.

Savarese, S. and Fei-Fei, L. (2007). 3D generic object categorization, localization and pose estimation. In *Eleventh International Conference on Computer Vision (ICCV 2007)*, Rio de Janeiro, Brazil.

Schneiderman, H. and Kanade, T. (2004). Object detection using the statistics of parts. *International Journal of Computer Vision*, *56(3)*, 151–177.

Shotton, J., Blake, A., and Cipolla, R. (2005). Contour-based learning for object detection. In *Tenth International Conference on Computer Vision (ICCV 2005)*, pages 503–510, Beijing, China.

Snavely, N., Seitz, S. M., and Szeliski, R. (2006). Photo tourism: Exploring photo collections in 3D. *ACM Transactions on Graphics (Proc. SIGGRAPH 2006)*, *25(3)*, 835–846.

Su, H., Sun, M., Fei-Fei, L., and Savarese, S. (2009). Learning a dense multi-view representation for detection, viewpoint classification and synthesis of object categories. In *Twelfth International Conference on Computer Vision (ICCV 2009)*, Kyoto, Japan.

Sun, M., Su, H., Savarese, S., and Fei-Fei, L. (2009). A multi-view probabilistic model for 3D object classes. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009)*, Miami Beach, FL.

Szeliski, R. (2010). *Computer Vision: Algorithms and Applications*. Springer, New York.

Thomas, A., Ferrari, V., Leibe, B., Tuytelaars, T., Schiele, B., and Van Gool, L. (2006). Towards multi-view object class detection. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'2006)*, pages 1589–1596, New York City, NY.

Torralba, A., Murphy, K., and Freeman, W. (2007). Sharing visual features for multiclass and multiview object detection. *PAMI*, .

Ullman, S. and Basri, R. (1991). Recognition by linear combination of models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *13(10)*, 992–1006.

A Visualizations of car models

The visualizations for the 8 different car models used in (Ramnath *et al.* 2014), including the original input images, the shaded renderings of the visual hulls, and the refined 3D curve models, are shown in Figure 9.

B Detailed description of ellipse-based pose estimation

This appendix provides more details and derivations for the ellipse-based pose estimation described in Section 6.2 of (Ramnath *et al.* 2014).

Following the notation in (Szeliski 2010), we see that if we have two finite vanishing points and a known camera optic center of (c_x, c_y), we can write the vanishing points $\hat{\mathbf{x}}_i$ corresponding to the X and Y object axes as

$$\hat{\mathbf{x}}_i = \begin{bmatrix} x_i - c_x \\ y_i - c_y \\ f \end{bmatrix} \sim \mathbf{R}\,\mathbf{p}_i = \mathbf{r}_i, \tag{7}$$

where $\mathbf{p}_i$ corresponds to one of the cardinal directions $(1, 0, 0)$ or $(0, 1, 0)$, and $\mathbf{r}_i$ is the ith column of the rotation matrix $\mathbf{R}$.

From the orthogonality between columns of the rotation matrix, we have

$$\mathbf{r}_i \cdot \mathbf{r}_j \sim (x_i - c_x)(x_j - c_y) + (y_i - c_y)(y_j - c_y) + f^2 = 0 \tag{8}$$

from which we can obtain an estimate for f^2. Once the focal length f has been determined, an SVD of the first two rows and their cross-product can be used to recover an orthonormal $\mathbf{R}$ matrix.

Non-finite vanishing points. If one or more of the two vanishing points lie close to infinity, the method above will not produce reliable estimates.

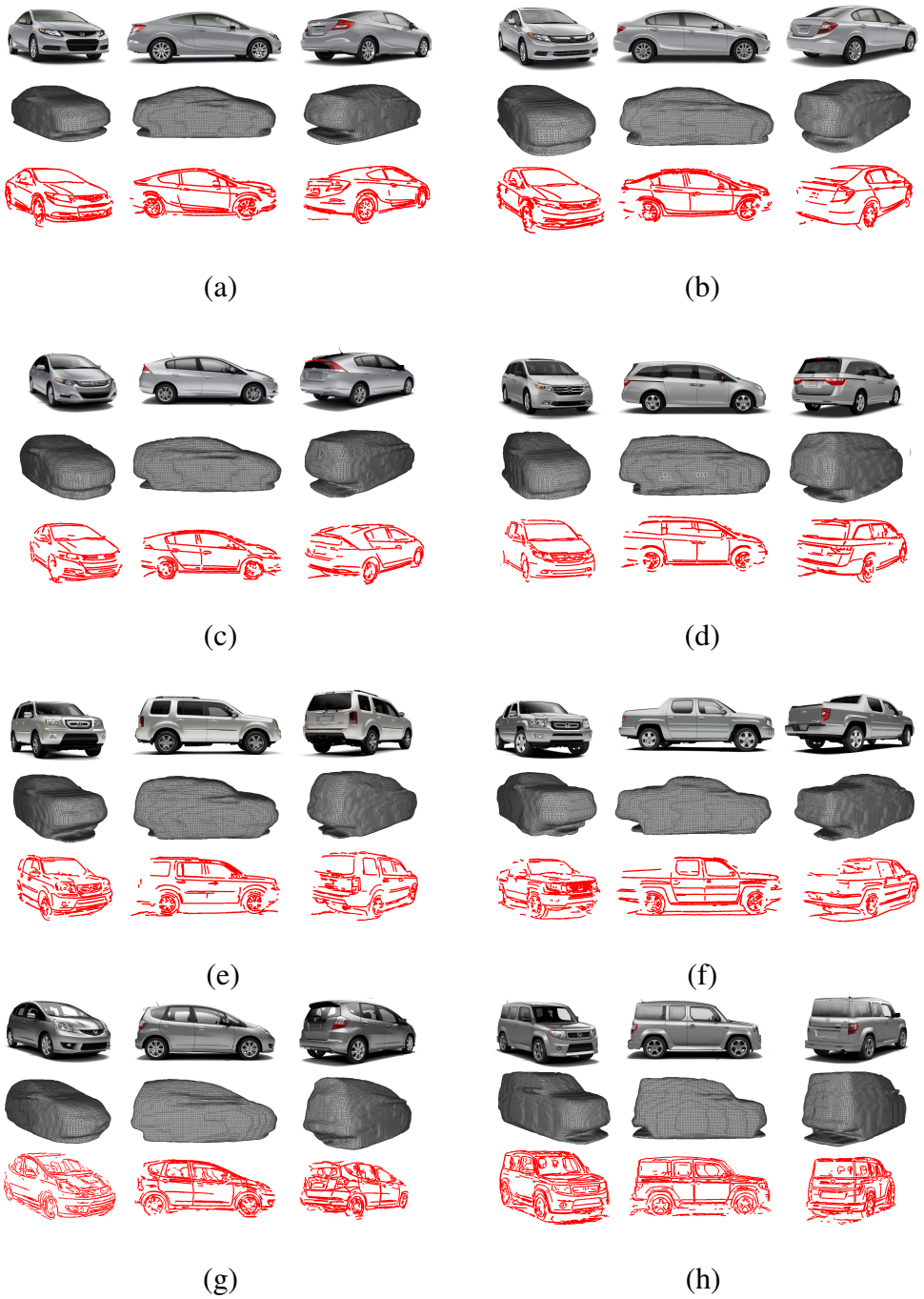

Figure 9: **Car Models in our datasets.:** (a) Honda Civic Coupe; (b) Honda Civic Sedan; (c) Honda Insight; (d) Odyssey 2011; (e) Pilot 2011; (f) Ridgeline 2011; (g) Honda Fit 2011; (h) Element SC 2010. The top row shows three of the input image, the middle row three corresponding shaded views of the visual hull, and the bottom row three views of the 3D curve models.

(a) (b) (c)

Figure 10: (a) An infinitessimal ellipse at the origin with the axes $\mathbf{a}_0$ and $\mathbf{a}_1$ pointing at the vanishing points; (b) car image with detected ellipse fragments and best fitting bitangents and frames; (c) the model points for both set of wheel rims tops and bottoms projected onto the captured image. (The vertical lines indicate which part of the model corresponds to the front wheels.)

Instead, we can use the *shape* of the two ellipses to provide us with a rotation estimate and focal length.[5]

Given the bitangents to the two ellipses and their vanishing points, we can compute a rectifying homography that maps the vanishing points to the points at infinity along the x and y axes.[6] Once we have the rectified ellipses (which transform into ellipses under homographies by simply pre- and post-multiplying the 3×3 conic matrices), we can find their vertical tangents and project these back into the original image, as shown in Figure 10b.

We now have a complete quadrilateral frame around each wheel, which we can use to estimate the rotation and focal length that map this frame into a square.

The easiest way to do this is to observe that if we have a small ellipse sitting at the optic center (Figure 10a), the axes corresponding to its intersection with the lines to the two vanishing points are projections of the orthogonal x-y axes of a circle and hence encode the rotation matrix.

[5] We assume that the ellipses correspond to circles lying in a common plane on the car side. Since the front wheels on a car may be turned, we estimate the rotation and focal length from each wheel in turn and pass both of these hypotheses on for verification.

[6] There are actually complete families of homographies that map these two points. For determining the "frames" around each wheel, it does not matter which one we choose.

Using the usual rules of perspective projections, we can show that the vectors corresponding to these two axes are

$$\mathbf{a}_i = s\tilde{\mathbf{r}}_i, \tag{9}$$

where $\tilde{\mathbf{r}}_i$ are the 2D vectors corresponding to the first two columns in the ith row of $\mathbf{R}$ and $s = f\epsilon/Z$ is an unknown scale factor equal to the product of the focal length f and the circle radius ϵ divided by the scene depth Z.

Inverting this equation, we get

$$\mathbf{r}_i = s^{-1}[a_{i0} \quad a_{i1} \quad \pm\sqrt{s^2 - \|\mathbf{a}_i\|}]. \tag{10}$$

As before, we can use the orthogonality between the rows of $\mathbf{R}$, i.e., $\mathbf{r}_i \cdot \mathbf{r}_j = 0$, to obtain two quadratic equations in the unknown s^2 corresponding to the two possible combinations of $\pm$. Of these, only one will lead to positive discriminants (quantities within the square root). To estimate the rotation matrix, we now have to try both potential signs of s as well as potential $\pm$ signs. If one of the vanishing points is sufficiently finite, we can use its direction from the origin to determine the sign of the z component for that rotation matrix row. If not, as before, we may have to return both potential solutions.

Unfortunately, most of our images will not contain infinitessimal circles at the optic center. (This assumption was just made to cancel higher order terms in the solution.) However, given the image of a circle and its bounding frame, we can estimate the homography that maps the frame into the unit square. This same homography can then be used to find the (warped) point that corresponds to the optic center, and a small set of orthogonal axes can be drawn at that point to numerically obtain the desired $\mathbf{a}_i$ axes.

We have tested this algorithm on a number of images, both with finite and infinite vanishing points, and found that it usually produces good quality estimates. However, since in our application we are dealing with model-based recognition, we can get even higher quality estimates using the 3D coordinates corresponding to the wheel rim tops and bottoms.

Wheel pair-based pose estimation. The reason why this produces higher quality results is that the baseline between the two wheels is always larger than the frame surrounding each wheel, so

our estimate of the foreshortening, on which the calibration is based, will be more accurate.

As before, we compute a homography that rectifies this frame (the quadrilateral bounded by the four sets of long lines in Figure 10b) into a rectangle of a known aspect ratio, determined from the 3D co-planar points that specify the car model wheel rim tops and bottoms. We can now proceed as before to compute the infinitessimal circle at the optic center to recover the focal length and rotation matrix.

Once we have recovered the focal length and rotation matrix, we can use simple least squares (a variant on the classic *direct linear transform*) to estimate the translational component of the camera pose.[7] Figure 10c shows the final results of aligning a car model to an input image, based on the bitangents shown in Figure 10b. In this figure, the horizontal lines join corresponding wheel tops or bottoms and the vertical line joins the top/bottom of each front wheel. As you can see, not only do the reconstructed points on the visible side of the car match well with the detected wheel rim locations, but the extrapolated locations of the wheels on the non-visible side (which can be inferred from the lower portion of the right front wheel) look fairly reasonable.

[7] In brief, form the rational linear equations in the unknown camera location for each of the projected four corners and then cross-multiply to obtain a set of four linear equations in the three positional unknowns.