2012 IEEE International Conference on Robotics and Automation
RiverCentre, Saint Paul, Minnesota, USA
May 14-18, 2012

A Probabilistic Framework for Car Detection in Images using Context and Scale

David Held, Jesse Levinson and Sebastian Thrun

***Abstract*— Detecting cars in real-world images is an important task for autonomous driving, yet it remains unsolved. The system described in this paper takes advantage of context and scale to build a monocular single-frame image-based car detector that significantly outperforms the baseline. The system uses a probabilistic model to combine multiple forms of evidence for both context and scale to locate cars in a real-world image. We also use scale filtering to speed up our algorithm by a factor of 3.3 compared to the baseline. By using a calibrated camera and localization on a road map, we are able to obtain context and scale information from a single image without the use of a 3D laser. The system outperforms the baseline by an absolute 9.4% in overall average precision and 11.7% in average precision for cars smaller than 50 pixels in height, for which context and scale cues are especially important.**

I. INTRODUCTION

Autonomous driving is an important task that could potentially save over a million lives each year [1]. However, fully autonomous driving, especially using affordable sensors, remains an unsolved problem. In order to safely drive on highways and urban streets, it is important for an autonomous system to be aware of the surrounding environment. To avoid collisions with other vehicles, it is crucial to have a system that can accurately detect nearby cars. Additionally, for driving at high speeds, detecting cars from a distance is also important. In this paper, we present an image-based car detection system using context and scale that significantly outperforms the baseline detector on this task.

A number of recent robotics efforts have combined 3D depth information with 2D appearance cues for indoor object detection. For example, affordable 3D range sensors have been used for object detection at close range in indoor environments [2]–[4] but these sensors fail at longer range and in outdoor environments. Stereo data has been used to assist object detection [5], but such data is noisy and most useful at shorter ranges. Multiple frames can also be combined to estimate depth using a structure-from-motion approach, as in [6] and [7]. Time-of-flight cameras have also been used [8], which provide low resolution images with a depth map at short ranges.

For long ranges in outdoor environments, the Velodyne multi-beam laser has been used successfully for high performance in both segmentation [9] and track classification [10]. However, this laser costs about $80,000 and is thus prohibitively expensive for commercial applications such as affordable autonomous driving.

In the vision community, some researchers have attempted to improve object detection by inferring scale from a single image, most notably in [11]. However, it is unclear from this

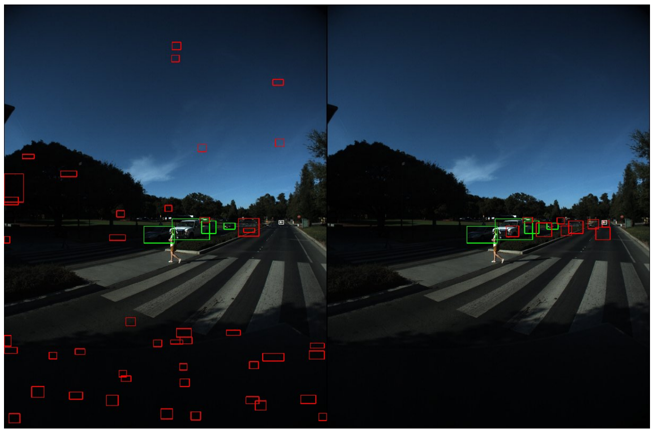
Fig. 1. Left: Detections returned by Felzenszwalb's state-of-the-art car detector. Note the large number of detections at inappropriate scales and contexts. Right: Detections returned by our algorithm, combining appearance, scale, and context. Note that the detections are all much more reasonable. False positives are shown in red; true positives are shown in green. This figure was produced by lowering the threshold for illustration purposes. At a higher threshold, only the correct detections are returned by our method.

work how many of the errors are a result of an incorrect scale estimation. For instance, [11] presents an example in which the windows of a building cause the model to estimate an incorrect location for the horizon line, leading to poor detection performance. Although the performance is impressive given their difficult task, in robotics applications we can take advantage of external knowledge without having to infer the scene geometry from visual cues alone.

Over the past decade, there has been a profusion of computer vision research using context for object detection [12]–[15]. As with scale, in robotic applications we can use maps combined with localization to obtain context information directly, without having to infer it from the image.

In this paper, we present a probabilistic model for combining appearance, scale, and context scores for object detection in images obtained from a robotic framework. Because of the flexibility of our model, we can combine multiple pieces of evidence for both scale and context without having to infer a single understanding of the scene. We are able to obtain scale and context information using only our localization system and a road map, without having to use any sensors to measure the scale or context directly. We also filter our images by scale prior to feature extraction, leading to a speed up over the baseline of a factor of 3.3. The resulting system significantly outperforms the baseline, especially on distant cars for which context and scale cues are particularly important relative to appearance cues alone.

II. DETECTION SCORES

A. Appearance Model

The appearance model we use is Felzenszwalb's deformable parts-based model [16], trained on the PASCAL 2007 training set. We use the cascade-detection speed-up described in [17]. This model comes pre-trained with a baseline car detector [18]. Briefly, the model uses a latent SVM to learn a set of car templates (Figure 2) that represent 6 different car orientations. Each orientation consists of a "root" template that roughly corresponds to the outline of a car, and 6 "part" templates that contain more detailed representations of different sections of a car in an image. Each of these templates is convolved with the gradient of the test image at multiple scales, and the locations with high responses for the root and part templates, in which the part templates are reasonably positioned with respect to the root, are declared to be car detections.

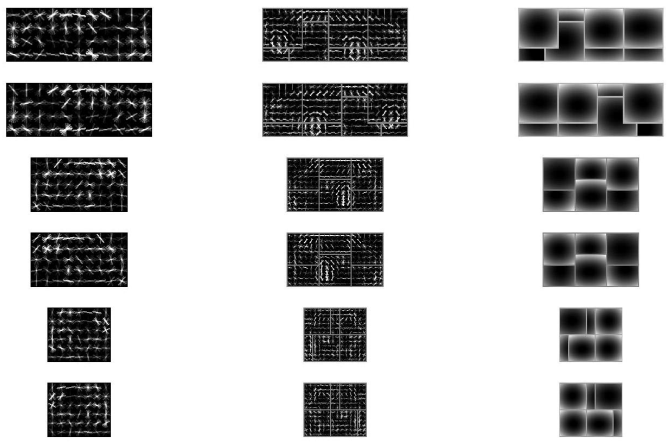
Fig. 2. Car models used by the Felzenszwalb appearance model (best viewed on a computer at high resolution). Left column: Car root templates. Middle column: Car part templates. Right column: Expected position of each of the parts, relative to the root. Each row represents a car at a different orientation.

The first step of our algorithm is to run Felzenszwalb's appearance model on the scale-filtered image, described in section III-A. The result is a set of bounding boxes, each with a score which indicates how much the appearance of the bounding box matches with that of a car. Note that because of shadows, occlusions, or car types that do not match to one of the templates, some cars will get a relatively low score from our appearance model. Additionally, some objects that are not cars will get a relatively high score due to random gradients in the image that happen to align with one of the car templates. To remove these false positives without creating new false negatives, our algorithm uses the appearance score as just one of a set of scores to determine the final classification for each bounding box.

B. Scale Scores

As shown in Figure 1, using just the appearance model alone results in many false detections at the wrong scale. For example, the appearance model will return detections that appear close to the camera but are extremely small, and detections that are far from the camera that are extremely large. Both of these are unrealistic given the natural range of car sizes in the real world.

To help remove some of these false positives, we compute two scores for each bounding box based on how scale-appropriate the size of the box is given its location on the image. The two scores take into account two different sources of variance. First, cars come in different shapes and sizes, and the variance in real-world car height causes a similar variance in the size of cars in the image. Second, errors in elevation, pitch, calibration, and discretization can compound to lead to an error in the estimated height of an object in the image. Because of these two different sources of variance, we compute two scale scores, each of which assumes a different source of variance.

To compute the first scale score, we use the camera's known position and orientation relative to the ground to estimate the real-world height (in meters) of an object contained within a given bounding box in the image. The implementation details of this height estimation using our robotic system are explained in section IV-A. Next, we calculate the probability of a real car having this estimated height using a probability distribution over car heights. In this case, we assume that cars have a mean height of 1.6 m with a standard deviation of 0.4 m. The computed probability is rescaled from 0 to 1 and the resulting value is used as a scale score. This score predicts the scale-appropriateness of a given bounding box, while taking into account the real-world variance in car heights.

We also add a second score to take into account the variance caused by errors in our height estimation. To calculate this score, we first compute the expected height h, in pixels, of a car that is 1.6 m tall and located at the location of the bounding box in the image. We then estimate that, because of errors in our prediction, the actual size (in pixels) of a bounding box from a car of this height might be modeled by a Gaussian distribution, with a mean of h and a standard deviation of 20 pixels. Using this distribution, we compute the probability that a car projected into the image will be the size of our bounding box, given our estimated distance to this bounding box. We scale the result to range from 0 to 1. Using these two scores, we can prune away false positives that appear at unrealistic scales for cars in images.

C. Context Scores

Figure 1 also demonstrates that the appearance model often detects cars located at unrealistic positions in the image, such as in the sky or inside a tree. Because our autonomous vehicle already requires a road map for navigation, we can use this road map, and our known position and orientation on this map, to automatically estimate the position of the road in an image. The implementation details of this computation for our robotic system are described in section IV-B.

Using the estimated position of the road in an image, we would like to give a low weight to detections located at unlikely positions. Note that, because we are giving a higher weight to cars cars located on the road, we are less likely to detect cars parked on the grass or in a nearby parking lot. For our application, this is acceptable or even desired, because we wish to use this car detector to locate other cars on or near the road, in order to avoid accidents while driving.

As with scale, we compute two context scores, to take into account two sources of variance. First, a car might not always be driving in the middle of the road, but rather it might be driving in the shoulder of the road. Second, errors in calibration might cause our system to incorrectly estimate which parts of the image contain the road. Thus we add two context scores, each of which will take into account a different source of variance.

The first context score takes into account the variance from cars pulled over to the side of the road, or otherwise not driving in the center of the road. To compute this context score, we first estimate the location of the bounding box in global coordinates. To do this, we assume that the bottom center pixel of the bounding box is a point on the ground. We then use our localization system and our calibrated camera to estimate the global coordinates of the object contained within this bounding box. Using this estimated real-world position, we then estimate the distance in meters to the nearest road using a quadtree road network system. The implementation details are further described in section IV-B. Using this distance, we compute the first context score as

$$\text{score}_m = \frac{1}{d_m + 1}$$

where d_m is the estimated distance in meters to the nearest road. When the bounding box is located on the road, we have $d_m = 0$, leading to a score of 1. For cars located to the side of the road, the score decreases slowly to 0. This score can be visualized in Figure 3.

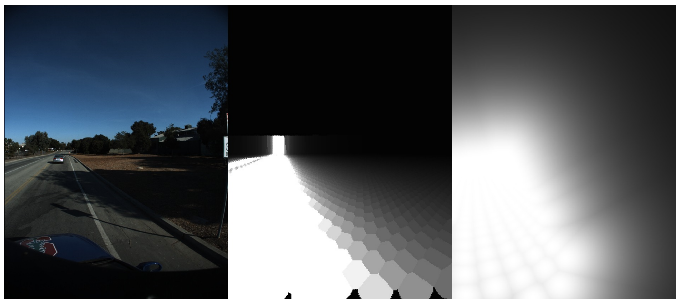

Fig. 3. Left: car image. Middle: Context score, based on the estimated distance to road in meters. Right: Context score, based on the estimated distance to road in pixels. Brighter parts indicate a higher context score; darker parts indicate a lower context score.

On the other hand, errors in localization might cause us to incorrectly estimate the position of the road in the image. To account for these types of errors, we add another context score in which we estimate the distance in pixels, d_p, from the bounding box to the nearest road pixel. This context score is now computed as

$$\text{score}_p = \frac{1}{d_p + 1}$$

This score can be visualized in Figure 3. Note that both of the context scores are computed automatically using our localized position on a road map and our calibrated camera.

III. ALGORITHM

A. Scale Filtering

In order to speed up the algorithm, we first filter the image based on scale. Felzenszwalb's car detector [16] iteratively searches for cars at different scales in the image. Because we can estimate the scale of objects in the image using the position of our calibrated camera relative to the ground plane, we can limit our search to regions of the image that are appropriate for each scale. Thus at each scale, we black out the portions of the image that are not relevant for that scale, as shown in Figure 4. Specifically, we find these regions using our pixel-based scale score, removing all portions of the image that would receive a scale score of less than 0.1, on a scale from 0 to 1. The result of this blackout is that many of the features computed are 0, and these regions of the image are pruned away at the beginning of the cascade detection step of [16], allowing the detector to spend more time searching for cars in regions of the image that are more scale-appropriate for each bounding box size. This gives a small boost in performance, as shown in section VII, and it gives a large increase in speed, decreasing the runtime of the algorithm by a factor of 3.3.

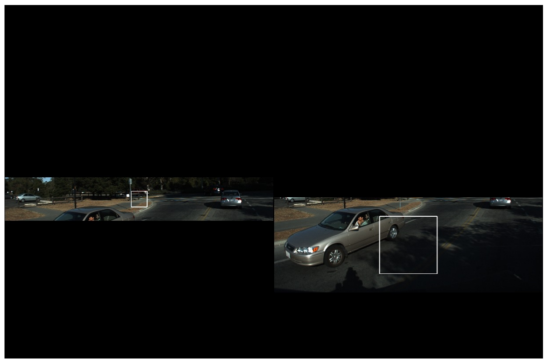

Fig. 4. Left: We black out the image except near the horizon, to search for smaller cars. Right: We black out the image except lower pixels, to search for larger cars. The white boxes indicate the scale of car that is being searched for in the image. By limiting our search space at each scale, we get better performance and a large increase in the speed of our algorithm.

After applying the scale filtering, we run the appearance model to get bounding boxes over the image, and for each bounding box we compute scale and context scores. Thus, for each bounding box, we obtain an appearance score, two scale scores, and two context scores.

B. Probabilistic Framework

Using the appearance, scale, and context scores described in section II, we compute a new prediction for each bounding box using the dual form of L2-regularized logistic regression.

This regression is trained using our training set, described below. Our training set contains 282,283 negative examples and only 2,180 positive examples. Because of this large imbalance, we weight the C parameter differently across the different classes. For the negative class, we use $C_0 = 0.007664 * C$ and for the positive class we use $C_1 = 0.992336 * C$, thereby weighting the loss differently for positive and negative training examples to counter the imbalance in the class sizes. We choose a C value of 10,000 from cross-validation by holding out 20% of our training set for parameter tuning. After choosing C, the final model is learned on all of the training data.

Using the resulting logistic regression model, we predict a new confidence value for each bounding box. Note that, because we are using a probabilistic framework, we are not limited to compute a single maximum-likelihood estimate of our distance to the road or the real-world height of each bounding box object. Rather, we can compute two scores for each of scale and context and one score for appearance, and we can use all of these scores to predict our final classification. Thus, our model is extremely flexible and can be used with any number of scores that may not be conditionally independent.

C. *Aggressive non-maximum suppression*

After applying the logistic regression to rescore all of our detections, we often end up with a large number of overlapping high-scoring detections that all correspond to the same car in the real world. To handle this, we apply an aggressive non-maximum suppression algorithm, discarding all bounding boxes that overlap with a higher-scoring bounding box by at least 20%. For cars that occlude each other by more than 20%, this will result in missed detections. However, because cars tend to stay separated in the real world in order to avoid collisions, this assumption is reasonable in many cases. This non-maximum suppression results in an overall gain on our validation performance as well as our test performance, so we include this as the final step of our algorithm.

IV. SYSTEM

Our research vehicle is a 2006 Volkswagen Passat wagon. A Point Grey Ladybug-3 panoramic RGB camera with six 1600x1200 cameras and a 15 Hz frame rate is used to capture video; for simplicity, the results in this paper use only a single forward-facing camera out of the six. An Applanix POS-LV 420 GPS/IMU system with Omnistar satellite-based Virtual Base Station service generates pose and inertial updates for the vehicle at 200 Hz. This localization, combined with the hand-measured calibration between the Ladybug camera and the vehicle (which must be performed once when the camera is fixed to the car), give us a registration between the camera images and the digital road map, represented in global coordinates. Note that, although our Applanix localization system is rather expensive ($150,000), Section VIII shows that our method is robust to localization errors typical of consumer-grade GPS systems, so a much cheaper and less accurate localization system could also have been used.

A. *Estimating Scale*

Given the registration between the camera images and the global coordinate system, we can project any point from the real world onto the camera image. However, the reverse transformation is ambiguous; a single point on the image corresponds to a line of points in the real world. In order to estimate the distance to objects in the image, we compute a set of "image reference points" as follows: first, we imagine a grid placed on the ground plane surrounding the ego vehicle. In our case, we use a 100 m by 100 m grid in which points are spaced 0.5 m apart. For this paper, we have assumed that the ground is planar, although maps with elevation data could easily be used if necessary.

We then project each grid point onto the image to obtain a set of "image reference points," as shown in Figure 5. Because we know the grid point from which each image point was projected, these image reference points define a mapping from pixels to global coordinates. Thus for any pixel in the image, we can estimate its distance to the camera by simply returning the distance of its nearest image reference point in the 2d image plane. This process is shown in Figure 5.

To determine which points in the image correspond to ground and which correspond to sky, we note that points on the ground all lie near to some image reference point, which was projected from a grid on the ground plane surrounding the ego vehicle to the ground plane in the image. On the other hand, pixels in the sky lie farther away from any image reference point (see Figure 5). Thus we use a simple cutoff and assume that any point that is more than 100 pixels away from its nearest image reference point must be in the sky.

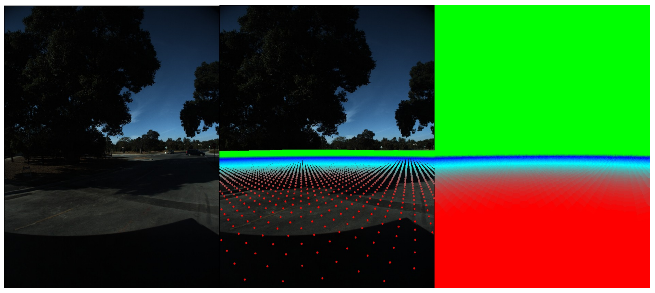
Fig. 5. Left: Normal image. Middle: Image with reference points. Right: Estimated distance to all image pixels. Red points are estimated to be closer to the camera, and green points are estimated to be farther from the camera.

B. *Estimating Context*

Using our GPS/IMU system, we are able to localize ourselves on a pre-recorded road map. These maps are necessary for the autonomous system to navigate from a given starting point to a desired destination. We have entered the road network into a quadtree data structure for fast queries. By localizing our position and orientation on the map, and by using the scale calibration described in section IV-A, we are able to automatically estimate which pixels in the image lie on the road and which do not, as follows: For a given pixel, we use the image reference point system described in section IV-A to estimate the position of this pixel relative to the car, assuming that this pixel lies on the ground plane. Then given our position in global coordinates, we can use this relative position to estimate the global position of this pixel. Next we simply look up the global position of this point in our quadtree road network to estimate the distance from this pixel to the nearest road. This information is used to compute the context score, described in section II-C and visualized in Figure 3.

V. BASELINE

The baseline of our algorithm is the appearance model from Felzenszwalb's deformable parts-based model [16], trained on the PASCAL 2007 training set, as described in section II-A. The appearance model we use contains two minor variations on top of the original code that is publicly available online. First, the original model combines 8 neighboring pixels into a single super-pixel, in order to compress the image to the size of the template. We found that 8 pixels was too large to detect the smallest cars in our test set, so we lower this super-pixel size to 4 neighboring pixels. We change this parameter for both our context/scale detector and for the baseline detector, allowing us to detect smaller cars in both cases.

Second, the non-maximum suppression step that is used in the original code removes bounding boxes for which the ratio of the intersection with a higher-scoring bounding box to the area of the higher-scoring bounding box alone is greater than 0.5. We find that this tends to favor larger bounding boxes at the expense of filtering out smaller bounding boxes, which is not preferred. Thus we modify this slightly to only filter bounding boxes for which the ratios of intersection to the area of both the lower and higher scoring bounding boxes are both greater than 0.5.

Both the change in the bin-size and the change in the non-maximum suppression algorithm are only minor adjustments to the original code from [18]. We use the original algorithm with these two adjustments as the baseline to compare performance with our algorithm.

As described in section III-C, the last step in our algorithm is to apply an aggressive non-maximum suppression. To isolate the effect of using scale and context on our performance, we compare the performance of our algorithm to a second baseline, in which we apply this aggressive non-maximum suppression to Felzenszwalb's original deformable parts-based model [16].

VI. DATASET

A. Training Sets

For the appearance model, we used the pre-trained car models provided with [18] that were trained on the car images from the PASCAL 2007 training set [19]. This training set consists of 625 labeled cars from 376 separate images. Note that these images were downloaded from Flickr, each taken with a different camera, and they include many different types of scenes. Some of the types of images that were used in training are shown in Figure 6. On the other hand, the images used in our test set are all real-world outdoor images recorded while driving with the Ladybug camera mounted on top of a car. The system presented in this paper did not require any retraining of the appearance model, despite the large differences in the types of images used in the training and test sets. However, retraining the appearance model using the same type of images found in the test set could lead to additional performance gains.

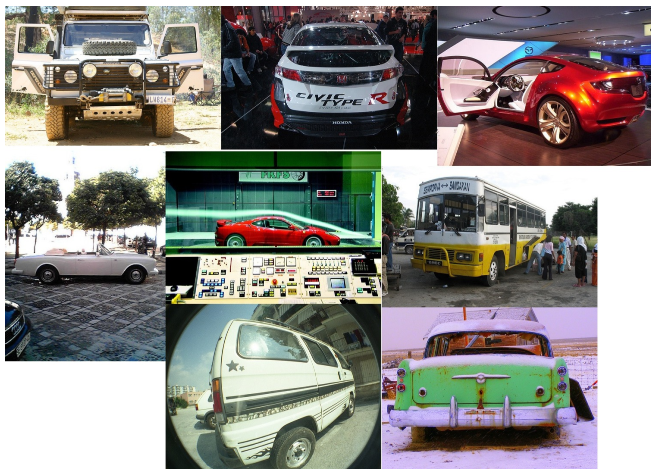

Fig. 6. A selection of images from PASCAL 2007 used to train the appearance model in [18], which is used as part of our algorithm. Notice the difference in the type of images used in the training set compared to the type of images found in our test set (e.g. Figure 1)

For training the parameters of our logistic regression, we use a second training set recorded while driving around Stanford campus, taken in a different part of campus from what was used for the test set. In this training set, we have 2597 labeled cars from 1501 separate images.

B. Test set

In our test set, we have 1932 labeled cars from 1120 separate images, recorded while driving around Stanford campus. Each image is 1232 x 1616 pixels, which is much larger than the standard image used in the PASCAL competition (typically around 500 x 350 pixels). Because our images are so large, both our algorithm and the baseline algorithm [18] run significantly slower than the same algorithms run on the smaller PASCAL images. Because the size of our images affects both the baseline detector as well as our improved context/scale detector, we can use the runtime of the baseline detector on these larger images as the basis for comparison.

This real-world data set is challenging for many reasons. There are many occlusions resulting from nearby cars overlapping each other in the image. There are also many shadows from trees or other objects that add spurious gradients to the image. Labeled cars range in size from 16 to 405 pixels and have a wide range of models, colors, and orientations. These are all challenges that a real car will encounter while trying to drive autonomously, and all of these issues must be dealt with in some way before vision-based autonomous driving is possible.

VII. RESULTS

By pre-filtering each image by scale, we are able to reduce the runtime of our algorithm from 43 seconds per frame down to 13 seconds, or a speedup of 3.3 times over the baseline on the same set of images. The time for computing the context and scale scores, and using these scores in the logistic regression framework, has a negligible effect on our total computation time. Although our unoptimized implementation currently runs at approximately 1% of real-time, we believe that with intelligent optimizations and a GPU implementation, real-time performance could be achieved.

In looking at the weights learned by our logistic regression classifier, the distance to the road is given the most weight and is thus deemed to be very important in correctly identifying an object as a car. The appearance and height scores are given similar weights and are thus considered to have roughly equal importance.

The performance of our algorithm compared to the baseline can be seen in Figure 7. As explained in section V, we have two algorithms that we both consider to be our baselines - Felzenszwalb's original appearance model, and Felzenszwalb's model with aggressive non-maximum suppression. Figure 7 shows that our algorithm achieves an average precision of 52.9%, whereas the better of the two baselines achieves an average precision of only 43.5%. Note that, because cars make up a small percentage of each image, both the baseline and our algorithm are performing far better than random chance. The total area occupied by all cars in our test set makes up about 1.1% of the total area of all test images, so a random classifier would have a precision of about 1.1%.

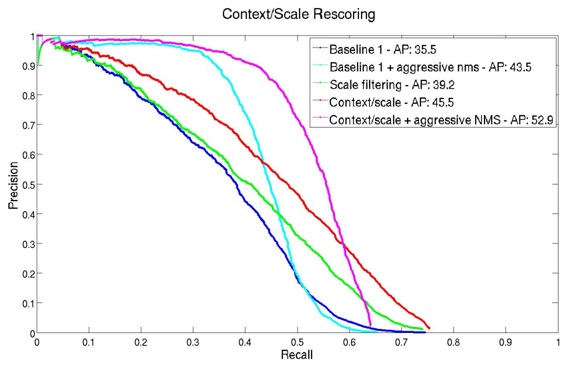

Fig. 7. Comparison of our algorithm (magenta) to the baseline (cyan). Dark blue: Baseline 1, from Felzenszwalb [18]. Cyan: Baseline 1 using aggressive non-maximum suppression. Green: Scale filtering. Red: Context/scale rescoring. Magenta: Context/scale rescoring with aggressive non-maximum suppression

The majority of the benefit of using context and scale comes when searching for cars that are far away. Because these cars have a small size in the image, using context and scale is especially important, since the appearance of the car in the image gives less useful information. Figure 8 shows the benefit of using our context and scale model when looking for cars less than 50 pixels in height. Although finding such small cars is difficult for any detector, using context and scale leads to an improvement of 11.7% in average precision. Finding distant cars is extremely important, especially for highway driving, in which one must be aware of a distant but quickly approaching car when deciding whether to change lanes. Being aware of distant cars is also important when we are driving quickly in order to slow down in time to avoid hitting a car that has stopped in front of us because of heavy traffic.

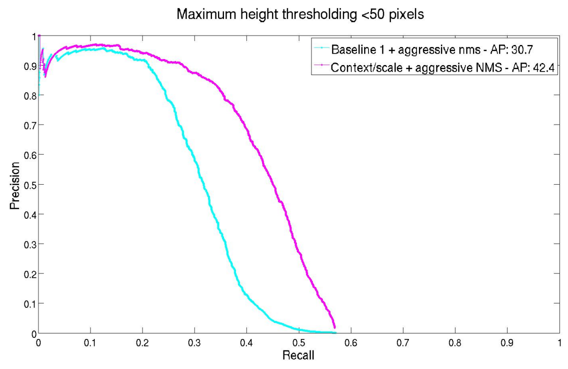

Fig. 8. Comparison of our method (magenta) with the baseline (cyan) on small (i.e. distant) cars

Naturally, detecting nearby cars is also important. Thus, to fully evaluate the performance of our algorithm, we recompute the average precision while ignoring cars below a certain pixel size threshold. Further, we ignore cars that are cut off by the edge of the image, determined by whether the bounding box is within 10 pixels from the edge of the image. In a real system, we would be using all 6 cameras from our Ladybug-3 panoramic camera system. Thus a car that is cut off by the edge of one image will most likely appear in a neighboring image, because of overlap between adjacent cameras. As shown in Figure 9, our system achieving 81.6% average precision for cars greater than 50 pixels in height.

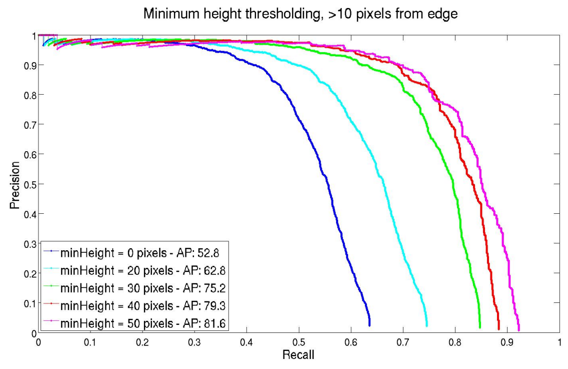

Fig. 9. Performance of our method when ignoring cars below a given number of pixels in height.

VIII. SENSITIVITY TO LOCALIZATION ACCURACY

Because our method relies on localization to improve object detection, it is informative to quantify the effect of errors in localization on detection performance. To test this, we artificially added Gaussian noise e to our localization estimates, with e $\sim$ N(0, σ), for $\sigma \in \{0, 0.5, 1, 1.5, 2, 2.5\}$ meters. We resample the noise at every time step. The results are shown in Figure 10. By introducing a Gaussian noise of e $\sim$ N(0, 2.5) meters, our performance degrades from 52.9% average precision to 52.0%. Thus our method is not very sensitive to localization accuracy, and even a cheap GPS system can be used to obtain nearly the same gains in performance.

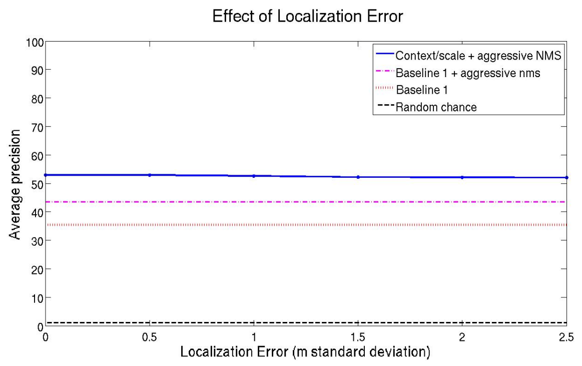

Fig. 10. Effect of GPS error on our performance. Our method (blue line, top) exhibits a very slight decrease in performance with increasing GPS noise. The other methods do not use localization information to assist in detection so they are unaffected by localization errors.

IX. CONCLUSIONS

By using scale and context cues obtained from our localization system and a road map, we are able to improve on the state-of-the-art in car detection systems without using 3D range data. Our vision-based system is able to detect distant cars with a higher degree of accuracy than previous methods, leading the way towards a camera-based autonomous driving system. Our probabilistic model combines multiple sources of evidence for scale, context, and appearance to make a more informed prediction about car locations. Because of the flexibility of this design, additional sources of information can easily be added to the system to further improve the detector.

Although the reliable detection of vehicles using only a monocular camera remains an elusive goal, our work shows that the state of the art can be significantly improved by considering context and scale. Given that all autonomous cars must have a localization system and some form of a road map, our insight provides an encouraging improvement to vision-based object detection for any similar system.

X. ACKNOWLEDGMENTS

We would like to thank David Jackson, Alex Teichman, Michael Sokolsky, Vaughan Pratt, David Stavens, and Soeren Kammel for their helpful comments and advice.

REFERENCES

[1] United Nations General Assembly. *Global road safety crisis*, 2003.

[2] Stephen Gould, Paul Baumstarck, Morgan Quigley, Andrew Y. Ng, and Daphne Koller. Integrating visual and range data for robotic object detection. In *ECCV Workshop on Multi-camera and Multi-modal Sensor Fusion Algorithms and Applications (M2SFA2)*, 2008.

[3] Kevin Lai, Liefeng Bo, Xiaofeng Ren, and Dieter Fox. Sparse distance learning for object recognition combining rgb and depth information. In *ICRA*, pages 4007–4013, 2011.

[4] Morgan Quigley, Siddharth Batra, Stephen Gould, Ellen Klingbeil, Quoc Le, Ashley Wellman, and Andrew Y. Ng. High-accuracy 3d sensing for mobile manipulation: Improving object detection and door opening. In *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, pages 2816 –2822, may 2009.

[5] S. Helmer and D. Lowe. Using stereo for object recognition. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 3121 –3127, may 2010.

[6] Bastian Leibe, Nico Cornelis, Kurt Cornelis, and Luc Van Gool. Dynamic 3d scene analysis from a moving vehicle. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 0:1–8, 2007.

[7] Gabriel J. Brostow, Jamie Shotton, Julien Fauqueur, and Roberto Cipolla. Segmentation and recognition using structure from motion point clouds. In *Proceedings of the 10th European Conference on Computer Vision: Part I*, ECCV '08, pages 44–57, Berlin, Heidelberg, 2008. Springer-Verlag.

[8] Amit Bleiweiss and Michael Werman. Fusing time-of-flight depth and color for real-time segmentation and tracking. In Andreas Kolb and Reinhard Koch, editors, *Dynamic 3D Imaging*, volume 5742 of *Lecture Notes in Computer Science*, pages 58–69. Springer Berlin / Heidelberg, 2009.

[9] J.R. Schoenberg, A. Nathan, and M. Campbell. Segmentation of dense range information in complex urban scenes. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 2033 –2038, oct. 2010.

[10] Alex Teichman, Jesse Levinson, and Sebastian Thrun. Towards 3d object recognition via classification of arbitrary object tracks. In *ICRA*, pages 4034–4041, 2011.

[11] D. Hoiem, A.A. Efros, and M. Hebert. Putting objects in perspective. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 2, pages 2137 – 2144, 2006.

[12] S.K. Divvala, D. Hoiem, J.H. Hays, A.A. Efros, and M. Hebert. An empirical study of context in object detection. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 0:1271–1278, 2009.

[13] Geremy Heitz and Daphne Koller. Learning spatial context: Using stuff to find things. In David Forsyth, Philip Torr, and Andrew Zisserman, editors, *Computer Vision ECCV 2008*, volume 5302 of *Lecture Notes in Computer Science*, pages 30–43. Springer Berlin / Heidelberg.

[14] Antonio Torralba, Kevin P. Murphy, William T. Freeman, and Mark A. Rubin. Context-based vision system for place and object recognition. *Computer Vision, IEEE International Conference on*, 1:273, 2003.

[15] Bangpeng Yao and Li Fei-Fei. Modeling mutual context of object and human pose in human-object interaction activities. In *The Twenty-Third IEEE Conference on Computer Vision and Pattern Recognition*, San Francisco, CA, June 2010.

[16] Pedro F. Felzenszwalb, Ross B. Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32:1627–1645, 2010.

[17] P.F. Felzenszwalb, R.B. Girshick, and D. McAllester. Cascade object detection with deformable part models. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2241 –2248, june 2010.

[18] P. F. Felzenszwalb, R. B. Girshick, and D. McAllester. Discriminatively trained deformable part models, release 4. http://people.cs.uchicago.edu/ pff/latent-release4/.

[19] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results. http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html.