2018 IEEE International Conference on Robotics and Automation (ICRA)
May 21-25, 2018, Brisbane, Australia

DeepVP: Deep Learning for Vanishing Point Detection on 1 Million Street View Images

Chin-Kai Chang Jiaping Zhao Laurent Itti

Abstract— **We propose a novel approach to detect vanishing points in images using a convolutional neural network (CNN) trained on a newly collected Google street-view image dataset. By utilizing the camera parameters and road direction data from Google street view, we collected a total of 1,053,425 images with inferred ground-truth vanishing points, along 23 world-wide routes totaling 125,165 kilometers. We then formulate vanishing point detection as a CNN classification problem using an output layer with 225 discrete possible vanishing point locations. Experimental results show that our deep vanishing point system outperforms the state-of-the-art algorithmic vanishing point detector. We achieved 99% accuracy in recovering the horizon line and 92% in locating the vanishing point within a ± 5-degree range.**

I. Introduction

The capability of an autonomous driving vehicle to detect road boundary is crucial to determining the heading direction and staying on the road (Fig. 1). Typical navigation tasks relie on proximity sensors such as Laser Range Finders (LRF) [1], [2] to capture the surrounding geometry information. Current LRFs such as from Velodyne [3] are able to measure depth in 360 degrees. They are the primary sensors that have allowed Google's Car [4] to travel autonomously over 1 million miles. However, these sensors have challenges to recognizing lane markings on the road without physical geometry differences.

Road recognition methods can be categorized into two types. The first is the color-based approach, which recovers road boundaries by recognizing specific features such as color histograms [5], [6], color contrast between road and flanking areas [7], and road region candidates from color-based segmentation [8]–[10]

The second type of road recognition technique finds the road's vanishing point first to determine heading. Vanishing Point (VP) is the point where parallel road boundaries converge. By identifying the vanishing point location in the image, the system can further recover both road boundaries. Traditional vanishing point detection usually relies on hand-crafted edge sensitive features such as Gabor filters [11]–[13], Gaussian filters [14], or Hough transform lines [15]–[19]. In addition, typical filter-based VP detectors tend to be slow because they use a voting scheme which may have up to $\mathcal{O}(n^2)$ time complexity. As a remedy, [20] use sky segmentation to first estimate the horizon line, to reduce the voting area to the non-sky area. Moreover, [12], [21] use only 4 gabor orientation channels $\{0°, 45°, 90°, 135°\}$ and selective voting to speed up the voting process. Although traditional VP detection methods are able to estimate the road with certain accuracy, hand-crafted features are usually sensitive to spurious edges such as shadow boundaries.

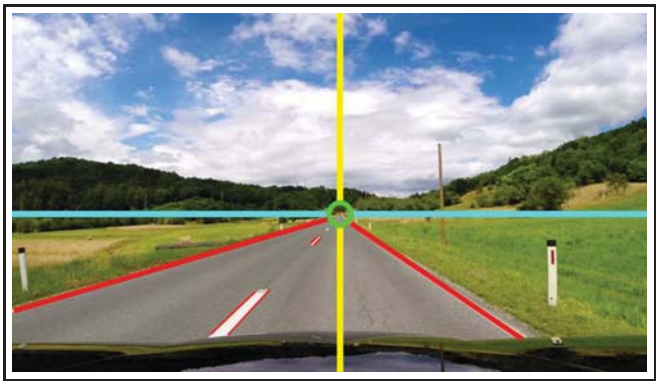

Fig. 1. Autonomous car navigation in a highway environment. The system needs to estimates the road heading as well as road boundary

Recently, deep learning [26] has dramatically improved the state-of-the-art results in many machine learning domains. Here we briefly review applications of deep architecture models with a focus on convolutional neural networks (CNNs) in the domains of computer vision, speech recognition, and natural language processing. Convolutional neural networks are composed of multiple processing layers, trained end-to-end to automatically learn representations of data with multi-level abstractions. In computer vision, CNNs have achieved great success in image classification [27]–[29], object detection [30], [31], image segmentation [32], [33], activity recognition from videos [34], [35] and many others. Two sensational advances are: in 2012, Krizhevsky et al. [27] trained a 7-layer convolutional neural network on the large-scale ImageNet [36] dataset to do image classification, and achieved a top-5 test error rate of 15.3%, almost half the error rate of the second best algorithm at the time. In 2014, Karpathy et al. [34] collected a sports-1M video dataset with 487 classes of sports, and used CNN to learn spatio-temporal information from video clips to classify videos into different sports categories, again beating the state of the art by a large margin.

In speech recognition, deep architectures have boosted the results significantly [37], [38]. Similar to training CNNs

C. Chang and J. Zhao are with Department of Computer Science, University of Southern California, Hedco Neuroscience Building - Room 10, 3641 Watt Way, Los Angeles, California, 90089-2520, USA. (chinkaic,jiapingz)@usc.edu

L. Itti is with the Faculty of Computer Science, Psychology, and Neuroscience, Univesity of Southern California, Hedco Neuroscience Building - Room 30A, 3641 Watt Way, Los Angeles, California, 90089-2520, USA. itti@pollux.usc.edu

978-1-5386-3081-5/18/$31.00 ©2018 IEEE

TABLE I

OVERVIEW OF MOST AVAILABLE VANISHING POINT DATASET.

Year	Dataset	Ref	Domain	Scene Class	Scene Coverage	Total Images
2003	York Urban Lane Segment	[22]	urban	2	small	102
2009	Kong'09	[11]	outdoor	3	medium	1,003
2011	Eurasian Cities	[23]	urban	1	small	103
2012	PKU Campus	[17]	urban	1	small	200
2012	Chang'12	[12]	urban	2	small	25,076
2014	Le'14	[24]	urban	1	small	1,600
2015	TVPD	[25]	urban	2	small	102
2018	DeepVP-1M	-	outdoor	23	large	1,053,425

in vision, researchers [38] used large voice datasets to train their CNN. Deep architectures have also been used to solve various natural language processing tasks. Kim [39] proposed to use a CNN to classify sentences, and Zhang and LeCun [40] trained CNNs on various large-scale datasets to understand text from scratch.

The success of deep neural models (e.g., CNNs) is partially attributed to the availability of big datasets. In vision, the ImageNet database [36] has 14,197,122 images, sports-1M [34] has 1M sports videos; in speech, researchers used hours of transcripted voice datasets to train DNNs/CNNs as well; in language, the training data is even larger as we have freely available online articles and documents. A sufficiently large amount of training data with sufficient diversity is necessary to effectively train complex deep architectures, since these have very large numbers of degrees of freedom.

In light of the limitation of traditional detectors and of the recent successes of deep neural networks, we propose to leverage the power of deep learning models to automatically learn a VP detector. Our novel contribution hence starts by presenting a completely new approach to estimate vanishing point adapted from deep learning. We train a deep model end-to-end, which takes an image as input, and outputs its vanishing point location. Compared to traditional algorithmic methods, which go through several sequential steps to predict the vanishing point location, our deep vanishing point algorithm is a fast, feed-forward neural network evaluation that directly returns the VP. However, training deep network models requires large-scale labeled datasets. To the best of our knowledge, available labeled vanishing point datasets are too small to properly train a deep network (Table I). This motivates us to collect our own dataset with 1 million Google street-view images with labeled vanishing point locations. The dataset includes worldwide range of road appearance in various lighting conditions. The dataset is the only one that contains different cameras angles from each scene location. We also present a method to auto-collect and label VP images by utilizing Google Street View API, which enables future dataset expansion. We make the dataset and source code available publicly [1]. We trained the convolutional neural network on this largest VP dataset available to date, and, as detailed below we achieved, by far, the best results ever reported in vanishing point detection method.

[1]http://ilab.usc.edu/kai/deepvp

In the following section II, we describe our new VP dataset. We present the network architecture in section III. We report the testing results in section IV, and discuss the findings in section V.

II. VANISHING POINT DATASET

Fig. 2. DeepVP Dataset collected from worldwide routes

A. New dataset for deep learning training

Current available image datasets to assist vanishing point algorithm development and testing are shown in Table I. Most are collected and hand labeled in urban area. The most common one [11] contains 1,003 images from web image search and desert road images.

Deep learning approaches typically require a much larger amount of training examples. Existing VP datasets fall short in the total number of images; even the largest existing dataset [12] only contains 25,076 images from four continuous video routes. Moreover, most of them are only collected in a small area, where the scenes are too similar and lack appearance variations. In addition, the collected VP images are usually recorded from a front facing camera mounted on a moving vehicle. This causes most of the VP location to be near the center of the image. Such unbalanced training labels may cause strong center bias and may decrease accuracy at other image locations.

Here we collected a freely-distributed dataset, the DeepVP-1M dataset, designed for large scale machine learning purposes. It contains 1,053,425 images with resolution 300x300 from 23 routes across 21 countries. The total length

of the routes is 125,165 kilometers, which ensures wide coverage of road appearance.

B. Data collection method

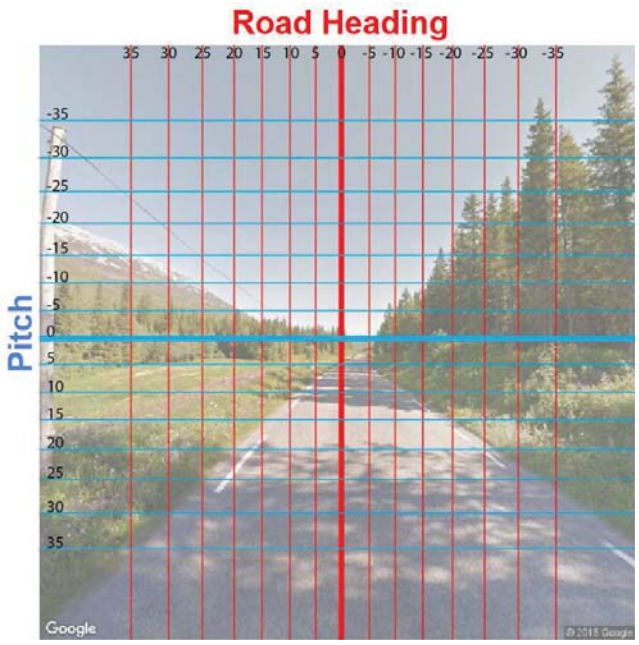

Fig. 3. Discretized VP labels in a total of 15x15=225 labels

We propose a novel way to collect labeled vanishing point images from Google Street View. Street View was chosen because 1) it covers a wide range of road appearances, 2) it has a panorama capability which can generate multiple VP views from individual locations, 3) it provides camera parameters such as pitch information to the estimate horizon line, and 4) when using the image capture vehicle's navigation direction, we can align VP center to the road heading direction based on the navigation route. Furthermore, Google Street View provides an API to fetch an image given GPS coordinates (longitude and latitude) and camera parameters (pitch, heading).

$$img = StreetViewAPI(lon, lat, pitch, heading) \quad (1)$$

C. Image Collection from a single GPS location

To predict VP coordinates in an image, we collect at every location a set of images that covers different VPs from top left to bottom right. We pan and tilt the camera view with step 5°from -35°to 35°in the panorama scene, resulting in a total of $15 \times 15 = 225$ images from a single GPS location (shown in figure 3). Image data collected in this way guarantees that the vanishing points are evenly distributed across the full image, instead of center-biased.

D. Align road heading with image heading

While camera pitch, which is compensated for road inclination, can be directly translated into the VP's Y-axis coordinate (assuming that the road's up or down curvature is small as is the case in most paved roads), estimating the VP's X-coordinate requires us to find angle difference θ between camera heading and road direction heading, as illustrated in Fig. 4. When we request a google navigation direction from point A to B, google map returns a series of polylines to represent each road segment. Along each polyline, we can further break down to a group of waypoints that indicate each GPS location in the same road segment. Given two waypoint coordinates $P1_{(lon,lat)}$ and $P2_{(lon,lat)}$, we then compute latitude and longitude difference (Ψ ,Φ) to approximate the angle of the road with respect to camera heading. This angle θ is the offset of the road heading.

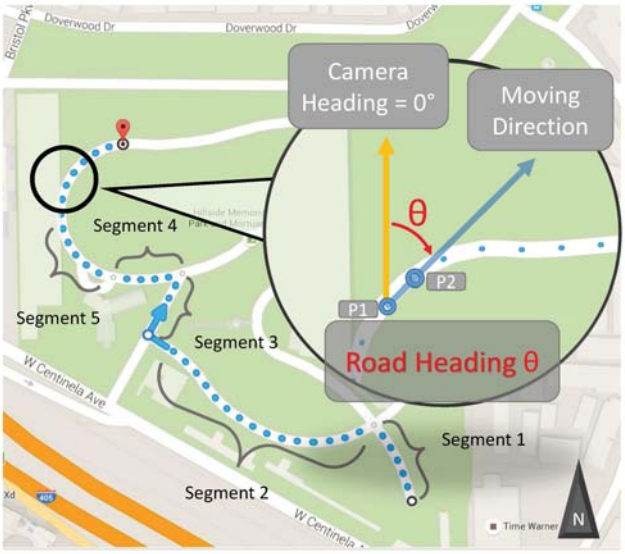

Fig. 4. Compute angle difference between camera heading and road heading

$$\Phi = \log\left(\frac{\tan\left(\frac{P2_{lat}}{2} + \frac{\pi}{4}\right)}{\tan\left(\frac{P1_{lat}}{2} + \frac{\pi}{4}\right)}\right) \quad (2)$$

$$\Psi = P2_{lon} - P1_{lon} \quad (3)$$

$$\theta = atan2(\Psi, \Phi) \quad (4)$$

Because the VP is likely to not be well defined at intersections where the Google vehicle took a sharp turn, we here only use waypoints well within the route segments, as the segment boundaries often coincide with such turns.

E. Mapping camera angle to pixel coordinate

To overcome camera distortion and non-linearity, we manually create a correspondence map from camera pitch angle to image Y-axis, using a third-order polynomial, whose parameters were derived from a small subset of manually-anotated data.

$$y = a\theta^3 + b\theta^2 + c\theta + d \quad (5)$$

Similarly, we can apply the same principle to find the road heading to x-axis mapping.

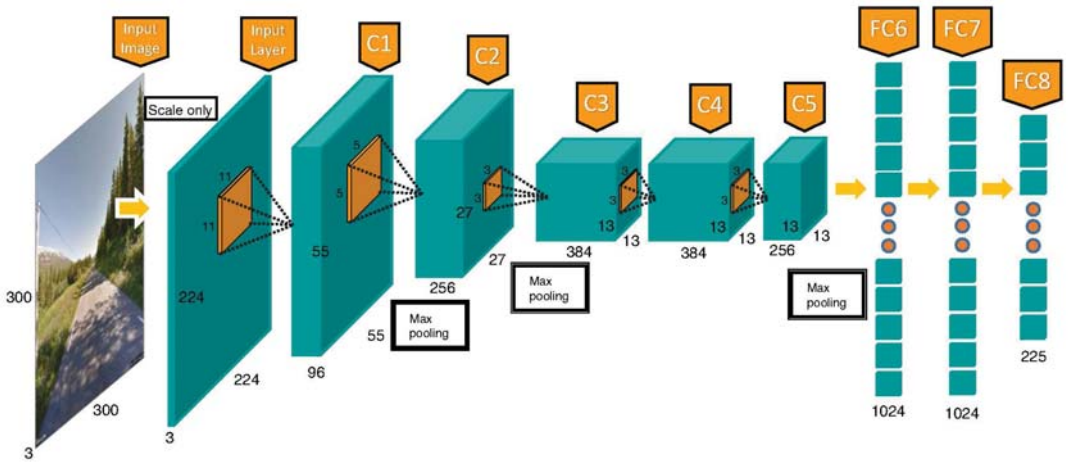

Fig. 5. The CNN architecture used for vanishing point detection. We closely follow the architecture of AlexNet [27], only changing the filters of the last two fully connected layers to 1024. The architecture in our notation form is: C96-P-C256-P-C384-C384-C256-P-F1024-F1024-O. Refer to text (section III from description.

III. Network architecture and its adaptation to vanishing point detection

In our scenario, we formulate vanishing point detection as a CNN classification problem. First we discretize numerical vanishing point coordinates into discrete labels. In a way, vanishing point detection is transformed into a classification problem: given a image, predicting the location of the vanishing point is equivalent to predicting the discrete label of its vanishing point. The predicted label is remapped to numerical coordinates in the image, obtaining the vanishing point location.

A. Network architecture

There are several prevalent CNN architectures, including AlexNet, VGG, Inception and ResNet. Here, without loss of generality, we use AlexNet [27]. AlexNet has achieved astonishing performance in object and scene classification [27], [41]. It is a linear chain feed-forward architecture with images as input and class labels as output.

We use an AlexNet implementation in [42], which consists of the input layer, 5 convolutional layers (Conv), 2 fully connected layers (fc) and the top label layer. Concretely, the first two layers are divided into 4 sub-layers: convolution, local response normalization (LRN), rectified linear units(ReLUs) and max-pooling. Layers 3 and 4 are composed of convolution and ReLUs. Layer 5 consists of convolution, followed by ReLUs and max-pooling. There are two fully connected layers, $fc6$ and $fc7$, stacked on top of $pool5$, and each of them is followed by ReLU non-linearity. The last label layer is a fully connected layer with the number of nodes equal to the number of discrete classes.

To be concise, we use abbreviations Ck, $ReLU$, Fk, P, D, O to represent a convolutional layer with k filters, an ReLU non-linearity layer, a fully connected layer with k filters, a pooling layer, a dropout layer, and an output layer. Since ReLU non-linearity layer is followed by every convolutional/fully-connected layer, we further omit it for simpler and cleaner architecture representation, in this way, the AlexNet is represented as: $C96-P-C256-P-C384-C384-C256-P-F4096-F4096-O$. In the following text, we use our notation for architecture representations.

B. vanishing point detection

We closely follow the original AlexNet architecture, only changing the number of filters on the last two fully connected layers to 1024. To be specific, we use this architecture for vanishing point detection: $C96-P-C256-P-C384-C384-C256-P-F1024-F1024-O$. By reducing the number of filters, the capacity of the CNN is reduced accordingly, which may result in a decrease in prediction accuracy. However, in our case, we only have 225 discrete vanishing point labels, and experimental results show that it is sufficient to maintain the prediction accuracy, but with much fewer parameters to learn. In total, we have 50M parameters, compared with 220M in the original AlexNet.

IV. Experiments

A. Dataset setup

We collected Google Street View images from 23 different routes. Along each route, we use images collected along the first 3/4 of the route as training data, and the remaining 1/4 route as test data. Under this partition, images in test are never seen during training.

We end up with 790,069 training images and 263,356 test images. Each image is associated with 1 (out of 225) discretized vanishing point label.

B. CNN setup

We train AlexNet to predict the discretized vanishing point after initializing the parameters with random Gaussian weights. Dropout layers following two fully connected layers use dropout rate of 0.5. We do not peform data augmentation

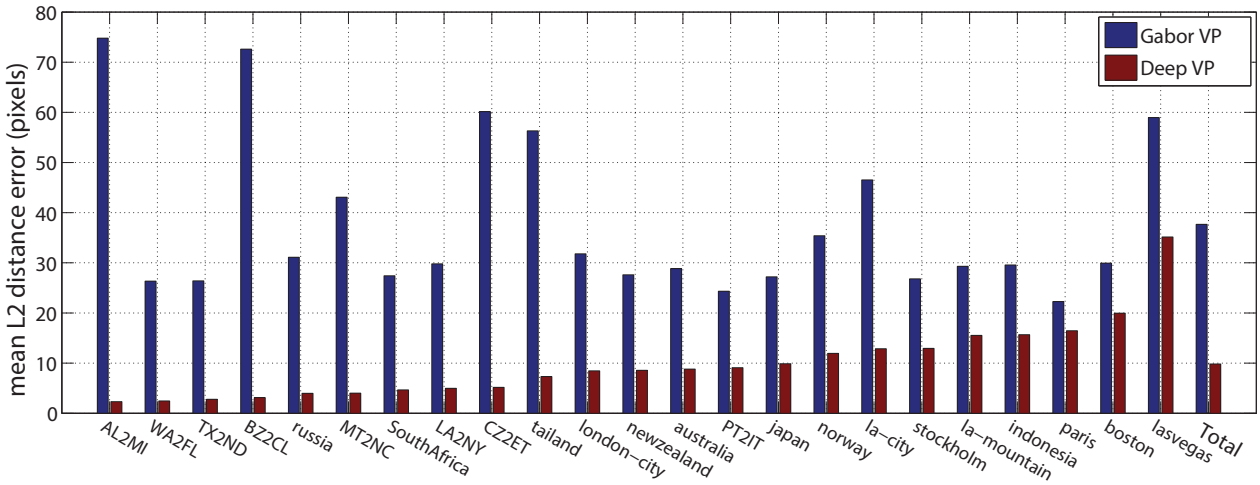

Fig. 6. Route-wise vanishing point detection accuracy of two algorithms. As seen the DeepVP algorithm consistently wins the Gabor VP method over all routes, with significantly accuracy improvements. This is the clear demonstration of the superiority of DeepVP over the traditional arithmetic algorithms.

during training since our dataset is already sufficiently large to allow us to achieve the best known results.

To optimize settings, we use the typical softmax loss function as the objective, and run stochastic gradient descent (SGD) to minimize the loss objective, with batch size set to 128. We set the starting learning rate to 0.01. The network is trained for 20 epochs, approximately 129K iterations. The learning rate is reduced by a factor of 10 after each 10,000 iteration. We train AlexNet using the publicly available MatConvNet [42] toolkit on an Intel 6-Core i7-5930K 3.5GHZ computer with a Nvidia Tesla K40 GPU.

C. Quantitative Comparison

We evaluate out new Deep VP method prediction on a total of 263,356 test images, and compare it to Chang's method [12] implemented in C++, as a representative Gabor VP method which has already been shown to perform better than [11] and [5].

Table II shows that Deep VP has a clearly improved overall accuracy (92.09%) compared to Gabor VP (47.99%). Deep VP is extremely accurate (99.25%) in estimating the horizon line (X-axis accuracy in Table II. We believe this is because the X-axis ground truth labels are directly mapped from camera pitch, which has minimal uncertainty. In comparison, the Y-axis accuracy (92.38%), which estimates the road heading, requires an extra angle offset estimation (θ), which may decrease precision.

In computation time comparison, DeepVP is 28 times faster than Gabor VP in terms of conventional CPU speed. This is due to DeepVP not requiring a complex voting scheme like Gabor VP. Furthermore, DeepVP can efficiently utilize a GPU to compute its result, further increasing its speed by $\sim 63\times$.

D. Independent Routes Comparison

In figure 6, we compute the Euclidean distance between the labeled ground truth and a method's prediction result for each route. The results show that the DeepVP method is more accurate than Gabor VP in all routes. Total L2 distance error from DeepVP is 9.83 pixels compared to Gabor VP's 37.67 pixels.

TABLE II
COMPARISON OF RESULTS WITH HAND-CRAFTED FEATURE APPROACH

methods	Deep VP	Gabor VP [12]
X-axis Accuracy($\pm 5°$)	99.25%	59.24%
Y-axis Accuracy($\pm 5°$)	92.38%	72.78%
Overall Accuracy($\pm 5°$)	**92.09%**	47.99%
CPU time(s)	0.56	15.73
GPU time(s)	0.0089	-

E. VP Coordinate Comparison

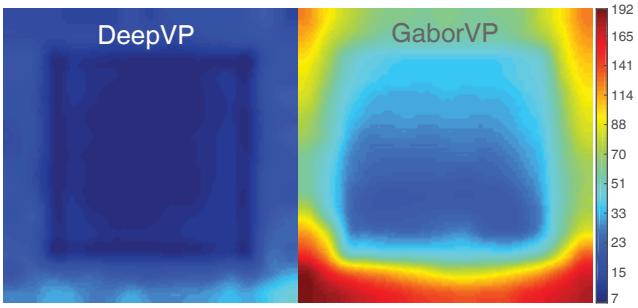

Fig. 7. The location-dependent accuracy of vanishing point estimation by two algorithms. As seen, the Gabor VP algorithm is biased, since it achieves better prediction accuracy when the ground-truth VPs are near the image center, while the prediction accuracy of DeepVP is relatively location-insensitive.

Fig. 7 evaluates the L2 distance error on each vanishing point coordinate. It shows that ground truth locations affect the traditional Gabor VP method significantly. Gabor VP has better accuracy when the vanishing point is near the middle region slightly below the image center, indicated in blue in Fig. 7. As we examine each prediction result, we find that

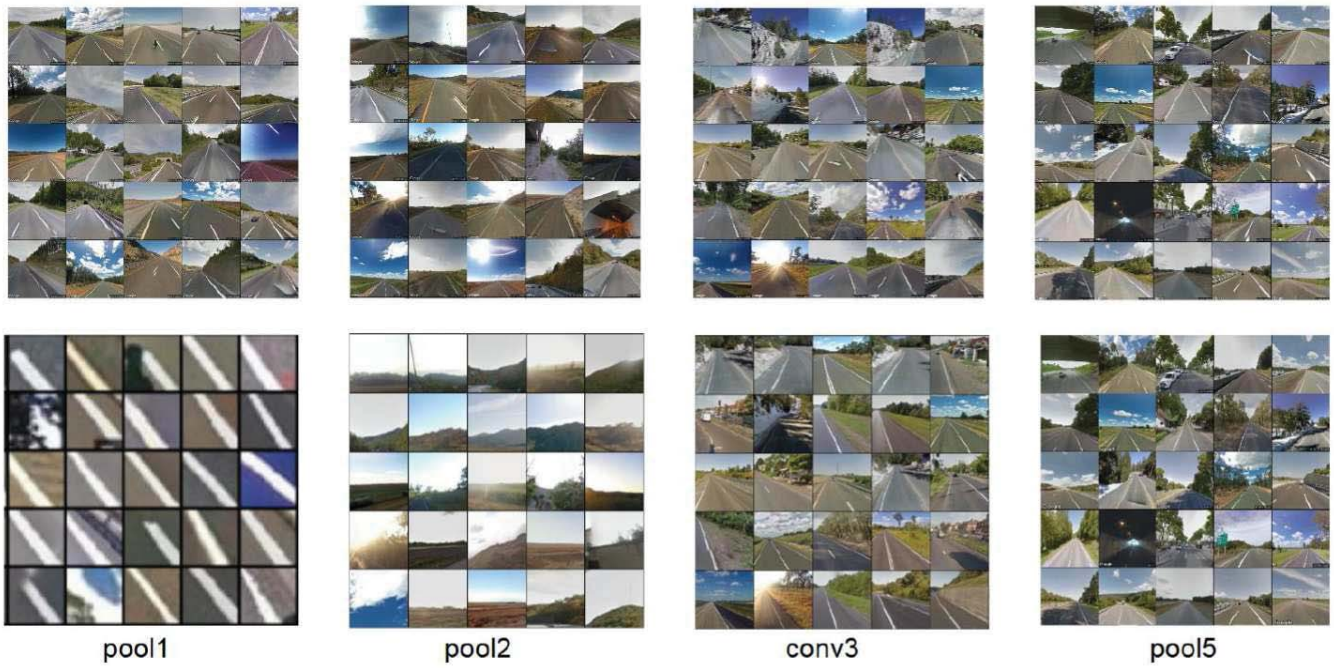

Fig. 9. Visualization of receptive fields of filters at different layers. **Top:** top 25 images with the strongest activations for filters at a few layer (pool1, pool2, conv3, and pool5); **Bottom:** 25 image patches from the corresponding images on the 1st row, that excite the filter the most.

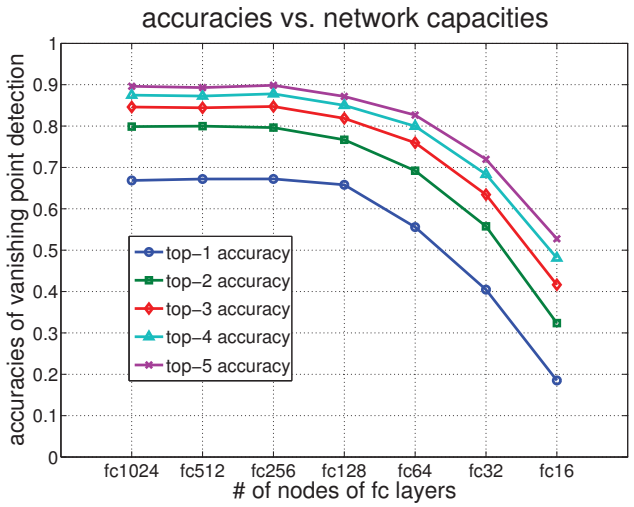

Fig. 8. Effect of network capacity on VP detection accuracy. We show empirically that when the number of neurons in the last two fully connected layers is as large as the number of the output layer labels, the CNN achieves as good a performance as a larger capacity network.

if the ground truth location is too high in the image, the Gabor VP method is affected by the shadows and markings on the road. If the ground truth location is too low, there is not enough road region in the image for Gabor VP to operate reliably. This increases the prediction error in Fig. 7, indicated in red. On the other hand, DeepVP is more stable in all regions. That is because our dataset has training labels equally distributed among all locations. The system not only relies on edge-sensitive features but more balanced dataset.

All the results were obtained under the architecture $C96-P-C256-P-C384-C384-C256-P-F1024-F1024-O$, which has 1024 neurons in each of the last two fully connected layers. Here we further investigate how network capacity affects VP prediction accuracy. We shrink the number of neurons in the last two fully connected layers by a factor of 2 gradually, train the network using the same training data, and plot the prediction accuracies in Fig. 8. As long as the number of neurons approaches or exceeds the number of output labels, 225 in our case, the prediction accuracies plateau.

F. Visualization

We use all test images as input, compute their activation responses for each filter on each layer, and show the top 25 images (Fig. 9, top) with the strongest activations for each filter and the corresponding image patch (Fig. 9, bottom) within the image which excites the filter most. Fig. 9 shows (1) receptive field size increases as the layer goes deeper; (2) meaningful sub-structures of scenes are learned automatically, e.g., Gabor-like edges in $pool1$ and horizon lines in $pool2$.

V. Conclusions

In this paper, we present an innovative vanishing point estimation method based on deep convolutional neural networks. Our results show that DeepVP outperforms the state-of-the-art algorithmic vanishing point method. DeepVP is capable of predicting vanishing point location in a wide range of environments with highly efficient performance. Also note that we collected the largest vanishing point dataset to date, with over 1 million images, to provide more comprehensive training, testing, and development for future vanishing point

Fig. 10. Examplar images from different sites: as seen our dataset collection procedure makes the vanishing points evenly distribute across the full image. This partially explains that our deepVP system is more location-insensitive, and less center-biased.

algorithms. Using the same a novel way to automatically collect the dataset and ground truth labels, more domain-specific datasets can also be added for different navigation application.

VI. Acknowledgments

This work was supported by the National Science Foundation (grant numbers CCF-1317433 and CNS-1545089), the Army Research Office (W911NF-12-1-0433), and the Office of Naval Research (N00014-13-1-0563). The authors affirm that the views expressed herein are solely their own, and do not represent the views of the United States government or any agency thereof.

References

[1] Z. Hu, H. Wang, L. Zhang, and H. Yu, "Laser based sensor boundary recognition of mobile robot," in *International Conference on Networking, Sensing and Control*, Okayama, Japan, March 2009.

[2] K. Wurm, R. Kummerle, C. Stachniss, and W. Burgard, "Improving robot navigation in structured outdoor environmets by identifying vegetation from laser data," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, oct. 2009, pp. 1217 –1222.

[3] C. Glennie and D. D. Lichti, "Static calibration and analysis of the velodyne hdl-64e s2 for high accuracy mobile scanning," *Remote Sensing*, vol. 2, no. 6, pp. 1610–1624, 2010.

[4] S. Thrun, "Google's driverless car," 2011, "Talk was viewed at http://www.ted.com/talks/sebastian_thrun_google_s_driverless_car.html on September 1, 2012".

[5] C. Rasmussen, Y. Lu, and M. Kocamaz, "Appearance contrast for fast, robust trail-following," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, October 2009, pp. 3505 – 3512.

[6] T. Kuhnl, F. Kummert, and J. Fritsch, "Monocular road segmentation using slow feature analysis," in *IEEE Intelligent Vehicles Symposium (IV)*, june 2011, pp. 800 – 806.

[7] P. Santana, N. Alves, L. Correia, and J. Barata, "Swarm-based visual saliency for trail detection," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, oct. 2010, pp. 759 –765.

[8] Y. He, H. Wang, and B. Zhang, "Color-based road detection in urban traffic scenes," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 5, no. 4, pp. 309–318, 2004.

[9] K.-Y. Chiu and S.-F. Lin, "Lane detection using color-based segmentation," in *Intelligent Vehicles Symposium, 2005. Proceedings. IEEE*. IEEE, 2005, pp. 706–711.

[10] P. Felzenszwalb and D. Huttenlocher, "Efficient graph-based image segmentation," *International Journal of Computer Vision*, vol. 59, pp. 167–181, 2004.

[11] H. Kong, J.-Y. Audibert, and J. Ponce, "General road detection from a single image," *IEEE Transactions on Image Processing*, vol. 19, no. 8, pp. 2211 – 2220, August 2010.

[12] C.-K. Chang, C. Siagian, and L. Itti, "Mobile robot monocular vision navigation based on road region and boundary estimation," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct 2012, bb, pp. 1043–1050.

[13] W. Yang, X. Luo, B. Fang, D. Zhang, and Y. Y. Tang, "Fast and accurate vanishing point detection in complex scenes," in *Intelligent Transportation Systems (ITSC), 2014 IEEE 17th International Conference on*. IEEE, 2014, pp. 93–98.

[14] H. Kong, H. C. Akakin, and S. E. Sarma, "A generalized laplacian of gaussian filter for blob detection and its applications," *Cybernetics, IEEE Transactions on*, vol. 43, no. 6, pp. 1719–1733, 2013.

[15] Y. Wang, E. K. Teoh, and D. Shen, "Lane detection and tracking using b-snake," *Image and Vision computing*, vol. 22, no. 4, pp. 269–280, 2004.

[16] C. Siagian, C. Chang, and L. Itti, "Mobile robot navigation system in outdoor pedestrian environment using vision-based road recognition," in *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, May 2013, bb, both first authors contributed equally.

[17] B. Li, K. Peng, X. Ying, and H. Zha, "Vanishing point detection using cascaded 1d hough transform from single images," *Pattern Recogn. Lett.*, vol. 33, no. 1, pp. 1–8, Jan. 2012. [Online]. Available: http://dx.doi.org/10.1016/j.patrec.2011.09.027

[18] J.-C. Bazin and M. Pollefeys, "3-line ransac for orthogonal vanishing point detection," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*. IEEE, 2012, pp. 4282–4287.

[19] B. Yu and A. K. Jain, "Lane boundary detection using a multiresolution hough transform," in *Image Processing, 1997. Proceedings., International Conference on*, vol. 2. IEEE, 1997, pp. 748–751.

[20] C. Ziyu and H. Zhen, "Simple road detection based on vanishing point," *Journal of Electronic Imaging*, vol. 23, no. 3, p. 033015, 2014.

[21] P. Moghadam, J. A. Starzyk, and W. S. Wijesoma, "Fast vanishing-point detection in unstructured environments," *Image Processing, IEEE Transactions on*, vol. 21, no. 1, pp. 425–430, 2012.

[22] P. Denis, J. H. Elder, and F. J. Estrada, "Efficient edge-based methods for estimating manhattan frames in urban imagery," *Computer Vision – ECCV 2008: 10th European Conference on Computer Vision, Marseille, France, October 12-18, 2008, Proceedings, Part II*, pp. 197–210, 2008.

[23] O. Barinova, V. Lempitsky, E. Tretiak, and P. Kohli, "Geometric image parsing in man-made environments," in *Proceedings of the 11th European Conference on Computer Vision: Part II*, ser. ECCV'10. Berlin, Heidelberg: Springer-Verlag, 2010, pp. 57–70. [Online]. Available: http://dl.acm.org/citation.cfm?id=1888028.1888034

[24] M. C. Le, S. L. Phung, and A. Bouzerdoum, "Lane detection in unstructured environments for autonomous navigation systems," in *Computer Vision–ACCV 2014*. Springer, 2014, pp. 414–429.

[25] V. Angladon, S. Gasparini, and V. Charvillat, "The toulouse vanishing points dataset," in *Proceedings of the 6th ACM Multimedia Systems Conference (MMSys '15)*, Portland, OR, United States, 2015.

[26] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[27] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[28] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[29] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1–9.

[30] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580–587.

[31] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, "Overfeat: Integrated recognition, localization and detection using convolutional networks," *arXiv preprint arXiv:1312.6229*, 2013.

[32] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Semantic image segmentation with deep convolutional nets and fully connected crfs," *arXiv preprint arXiv:1412.7062*, 2014.

[33] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3431–3440.

[34] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2014, pp. 1725–1732.

[35] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," in *Advances in Neural Information Processing Systems*, 2014, pp. 568–576.

[36] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE, 2009, pp. 248–255.

[37] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *Signal Processing Magazine, IEEE*, vol. 29, no. 6, pp. 82–97, 2012.

[38] O. Abdel-Hamid, A.-r. Mohamed, H. Jiang, L. Deng, G. Penn, and D. Yu, "Convolutional neural networks for speech recognition," *Audio, Speech, and Language Processing, IEEE/ACM Transactions on*, vol. 22, no. 10, pp. 1533–1545, 2014.

[39] Y. Kim, "Convolutional neural networks for sentence classification," *arXiv preprint arXiv:1408.5882*, 2014.

[40] X. Zhang and Y. LeCun, "Text understanding from scratch," *arXiv preprint arXiv:1502.01710*, 2015.

[41] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva, "Learning deep features for scene recognition using places database," in *Advances in neural information processing systems*, 2014, pp. 487–495.

[42] A. Vedaldi and K. Lenc, "Matconvnet – convolutional neural networks for matlab," *Proceeding of the ACM Int. Conf. on Multimedia*, 2015.