Proceedings of the 8th International Conference on Automotive User Interfaces and Interactive Vehicular Applications (AutomotiveUI '16), October 24–26, 2016, Ann Arbor, MI, USA

Towards a User-Centric In-Vehicle Navigational System

Olivia Wiles
University of Cambridge
Cambridge, UK
olivia.wiles07@gmail.com

Marwa Mahmoud
University of Cambridge
Cambridge, UK
marwa.mahmoud@cl.cam.ac.uk

Peter Robinson
University of Cambridge
Cambridge, UK
peter.robinson@cl.cam.ac.uk

Eduardo Dias
Jaguar Land Rover
Coventry, UK
edias@jaguarlandrover.com

Lee Skrypchuk
Jaguar Land Rover
Coventry, UK
lskrypch@jaguarlandrover.com

ABSTRACT

Current navigational systems rarely consider generic road landmarks in their navigation instructions, which can lead to mistakes, frustration, and distraction. However, automatic detection of road landmarks is difficult, as current approaches to object detection focus either on out-of-context objects which have special characteristics or on very specific domains. This work presents a future direction for a user-friendly navigational system based on state-of-the-art computer vision techniques that use deep learning for object detection. We propose an automatic hierarchical approach for detecting and classifying a set of static and dynamic road landmarks that would be useful in automatic navigational systems. We further demonstrate a set of optimisations that improve performance and accuracy of the basic system. We evaluate our approach on a natural, 'in-the-wild' dataset to determine how well it handles natural automotive input. Finally, we demonstrate a use-case for our system that extracts information about a vehicle's location and intention.

ACM Classification Keywords

H.5.2. Information Interfaces and Presentation (e.g. HCI): User Interfaces; I.4.9. Image Processing and Computer Vision: Applications.

Author Keywords

Deep learning; in-vehicle navigation; object detection.

INTRODUCTION

Early work in psychology and human-computer interaction demonstrated the importance of landmarks in navigation [12, 5]. Landmarks were found to be important for three reasons [5]. First, they are consistent with how humans navigate since we use landmarks as part of our internal representation of an area [28]. Second, landmarks are preferred by drivers in navigation. Landmarks are more popular than distances, number of streets or street names [5, 4]. Finally, landmarks make navigational systems more usable [5]. As a result of these factors, providing directions with generic landmarks improves a users' ability to navigate a given route, leading to fewer errors, less distraction, and safer driving [5]. For example, a navigational system that gives directions such as 'turn left at the church' has better usability than one that gives distance based directions such as 'turn left in about 150 metres'. The driver has more confidence in the decision and is less likely to make mistakes.

Incorporating landmarks into driving directions has been studied using databases holding information about the surrounding environment. Despite the increased amount of information about features of interest in our vicinity (e.g. from Google Maps), systems that rely on this information are not generalisable. The information may easily become outdated, is often only available for a small set of urban areas, and may be proprietary [13]. The system proposed in this paper detects general landmarks in the driver's view, meaning it is generalisable and does not rely on information that may easily become outdated.

In this paper, we describe an automatic, hierarchical approach for detecting and classifying a set of twelve static and dynamic road landmarks that would be useful for an in-vehicle navigation system (an overview of our approach is illustrated in figure 1). We evaluate our approach on a dataset collected 'in-the-wild' on different road types and in varying weather conditions. There are three major contributions:

1. Presenting a dataset of videos collected 'in-the-wild' on different road types using an in-vehicle camera. From these videos, we extracted and labelled a set of twelve static and dynamic salient road landmarks.

2. Proposing an automatic two-stage methodology augmented with filtering to detect and classify salient road objects based on the approach of R-CNN [15, 16]. We further compare methods of filtering the boxes to improve performance.

3. Demonstrating both quantitatively and qualitatively via a prototype that our approach is successful and useful in a navigational environment.

Automotive'UI 16, October 24 - 26, 2016, Ann Arbor, MI, USA

ACM 978-1-4503-4533-0/16/10 $15.00
DOI: http://dx.doi.org/10.1145/3003715.3005457

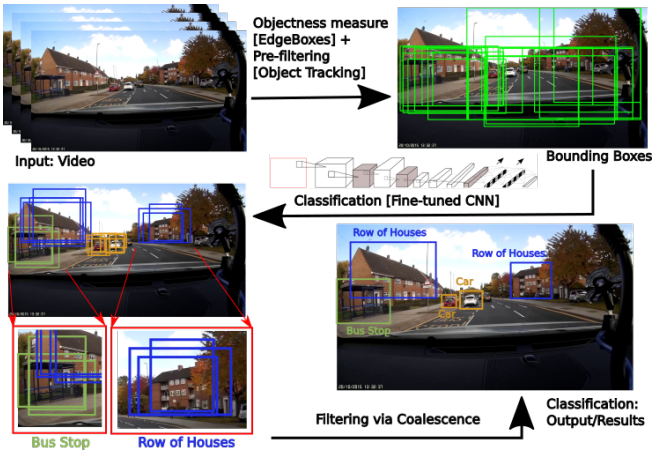

Figure 1. The pipeline of our proposed system. First, the video is divided into frames, which are used to find candidate bounding boxes around objects. These proposals are then filtered to select a subset of high-quality object proposals. These object proposals are classified using a convolution neural network. Finally, filtering is performed to remove overlapping bounding boxes of the same class, and the top candidates are returned.

In this paper, we first give an overview of related work. We then describe the dataset and associated preprocessing steps, followed by our methodology. Finally, we discuss our experimental evaluation, conclusions and future directions.

RELATED WORK

In-vehicle driver assistance has become standard in modern cars. These advanced driver assistance systems (ADAS) have a variety of uses ranging from pedestrian detection and lane departure warning to traffic sign detection and parking assistance. However, there has been little improvement for in-vehicle navigational systems from those that use distance directions based on GPS (e.g. in a SatNav). GPS systems are problematic for pedestrian navigation as users find the navigational task more difficult, are slower, and are more prone to errors [18]. One would expect similar problems for drivers but with more severe consequences.

Work centred on detecting landmarks has tended to focus on extracting specific landmarks based on prior knowledge of the person's location [10, 27, 25, 20]. These systems hold a database of images or important locations related to the user's environment (known apriori or based on GPS) against which images taken from the vehicle are matched. For example, the GIS system [20] uses an annotated map with landmark information (e.g. local businesses or town halls). The user is then presented with an updated map based on their location. This approach relies on a fast data connection speed, as downloading information based on the user's location requires having high bandwidth (1 Gb/km for Google) [30]. This is unrealistic in rural environments and requires apriori knowledge which may not be available or easily applicable.

Other approaches make the map more advanced using 3D imagery (e.g. from Google Earth). The directions are then overlaid on the map using the vanishing points of the image [9]. For example, the 3D ArtMap from Bosch adds 3D models of landmarks [2]. However, 3D ArtMap can only pick up major landmarks (such as Buckingham Palace) [23]. These systems do not detect generic landmarks that are useful in more general situations and must be detected on the fly, as the ability to label and store all occurrences of generic landmarks, such as post boxes, is clearly intractable. Moreover, the ability to render complex 3D imagery is computationally expensive.

	Our Dataset
Average image resolution	562x1000
Average object classes per image	1.41
Average object instances per image	2.16
Average object scale	0.0075

Table 1. The statistics of our dataset.

Other work in vision-based road understanding has focussed on specific road features or scene understanding. For example the systems described in the surveys by Mogelmose et al [22] and Hillel et al. [17] focus on detecting lanes or features along the road such as signs. Other systems focus on detecting street furniture such as *poles* or *trees* using laser scanning data [7, 21]. None of these systems attempt to solve the problem we are considering: namely detecting generic objects for use in a real-time navigational system using computer vision techniques.

Moreover, while the given systems may report impressive results, they tend to focus on one setting. As a result, the given technique may not work as well in other environments. For example, the urban setting is more challenging due to the clutter and occlusion in the scene and scenes with varying weather conditions are also more challenging [3].

DATABASE

We test our pipeline on our dataset, collected 'in-the-wild' with the challenges that such a natural dataset entails. Our dataset includes data from 47 drivers, with approximately 2 hour videos of driving per driver. The dataset used in this paper is a subset of this larger dataset. It includes the driving of 10 drivers and has statistics as given in table 1.

Figure 2. Two sample images from our dataset, demonstrating its particular difficulties. These images show the distortion and glare caused by the windshield, the effects of weather, and that many of the objects to be detected are extremely small.

Of the datasets for either driving or road-scene understanding, such as Kitti [14], Time Motorway [8] and Street Scenes [1], most focus on the detection of pedestrians, cars, or other scene features (e.g. trees, the sky, roads, sidewalks), whereas we are looking to detect landmarks that occur in these images. Moreover, many of these datasets (e.g. Street Scenes) do not include variations in lighting and weather conditions or the distortional effects of the windshield (see figure 2). Thus, our dataset has two important characteristics. First, our dataset

Figure 3. An example of a difficult image and its corresponding annotation. It is not clear where one row of houses ends or another begins. Also, while there are clearly three visible cars in the image, what about the other ones that are almost completely obscured and the ones that are very small. Finally, there is a church at the end of the street that, again, is mostly obscured.

is realistic in terms of the types of images and variations in these images. Second, our dataset includes a set of classes, previously understudied in navigational videos, which would be useful for detecting generic landmarks in the wild.

Landmark selection

The classes chosen were based on those static and dynamic salient landmarks that appeared on the route and are also either mentioned or fit the criteria discussed by Burnett [5, 6]. The criteria Burnett proposes are permanence, proximity to the road, and visibility from far off. We also included dynamic objects such as cars and buses, as these are useful for directions such as 'follow the car ahead' or if the object is unique (e.g. a striking colour) such that it may itself become a landmark. Certain objects are far more common than others, which is to be expected in such 'in-the-wild' datasets. The final set of classes and number of instances per class are: *church* : 876, *bridge* : 487, *bus* : 1453, *traffic light* : 3687, *zebra light* : 1746, *truck* : 2688, *pedestrian crossing* : 577, *car* : 19800, *corner shops* : 1815, *gas station* : 122, *bus stop* : 987, and *row of houses* : 9083.

Annotation was a non-trivial task due to the ambiguity of where a given object starts/ends and how to classify mostly occluded objects. For example, the class *row-of-houses* was particularly challenging, as seen in figure 3. To cope with this, we followed three rules for consistency of the labels: we only labelled objects with less than 50% occlusion, the boxes had to be at least 50% filled by the given object, and finally the objects themselves had to be larger than 40x40 pixels, as objects smaller than this would be too far away and difficult to detect for both the driver and our system.

METHODOLOGY

This section describes our two-stage approach for detecting and classifying landmarks in an image. It first discusses the objectness step for extracting high quality object proposals. It then explains the fine-tuned convolutional neural network (CNN) used for classification. Finally, it discusses two additional filtering steps we considered to improve detection when using only a handful of object proposals.

Objectness

The first step in our methodology as defined in figure 1 is the use of objectness to refine the number of proposals given to the classification stage. 'Objectness' assesses how likely a bounding box is to contain an object using a generic test (e.g. hand-derived characteristics of objects or more complex approaches). Object proposals are widely used in order to make classification problems tractable. As opposed to sliding window approaches that may require over $1,000,000$ proposals to obtain good results, the object proposal approaches may have up to 96% recall with only 1000 proposals [29]. As our system is envisioned as being used in real-time contexts, the improvement in performance obtained from using only a small set of high-quality bounding box proposals is vital.

In the automotive world, objects we wish to detect may be occluded, obscured, distorted, in miniature, etc., making our dataset challenging. Moreover, the images themselves may be noisy, with clutter or the effects of weather (e.g. rain or snow). As a result, we chose an object proposal method that was robust to these challenges. To extract the object proposals, we use EdgeBoxes [31], which measures objectness using edges. A bounding box is likely to contain an object if the edges within the box are self-contained and do not cross the box boundary. We also considered using BING [11], which assumes that most objects, when normalised and resized to an 8x8 window share many similar characteristics. However, as our experiments demonstrate, EdgeBoxes outperforms BING significantly, validating our choice.

Classification

Given the object proposals of the previous step, we implemented a fine-tuned convolutional neural network (CNN) to classify the object proposals as described in R-CNN [15]. Fine-tuning is necessary when the data provided is not sufficient to train a full CNN from scratch. Instead, we can exploit the weights learned for another network trained on similar data to initialise our network. We make the implicit assumption that those weights learned within the 'lower' layers are general and can be shared with our task. This method has been shown to give impressive results, even when using a net learned on one type of dataset for a completely new domain [24, 15]. In order to build this network, we used Caffe [19].

Filtering

A given object will fire many object proposals and correctly classified bounding boxes. Though this is a positive indicator that our system is working correctly, it is problematic when using an object detection system in a wider context for two reasons. First, having too many object proposals makes the system run slowly, as the CNN must classify each proposal and then rank the results. Given a powerful enough GPU, many object proposals may be acceptable, but it is unlikely that an in-vehicle navigational system would have such resources, so this remains an important constraint. Second, it is hard to ascertain the number of objects in the scene. This is important for object tracking while giving directions. For example, two overlapping bounding boxes classified as 'car' may indicate one or two cars. Our approach is twofold: to use pre- and post-processing steps to minimise the number of object proposals and to filter the classified results.

Pre-Filtering

In order to track objects from one frame to the next, we use feature descriptors to estimate the homography, the matrix

that describes the transformation between a point in a given image and where it moves to in the second image. We assume a perspective transform, which relies on the camera having pin-hole perspective, there being no non-linear distortion and the scene having planar perspective. While these conditions do not hold exactly (i.e. there are non-affine transformations due to the warping of the window screen), we hypothesised that the change from one image to the next would be minimal enough to make this a good approximation.

As a result, for every pair of subsequent images, we compute feature descriptors using ORB [26], match them between images and then estimate the homography between the images based on these correspondences. We use the `RANSAC` algorithm to determine a set of consistent inliers when computing the homography, as the correspondences will be noisy since the scene is not static. For example, cars may be moving in many directions within the image.

Then for each pair of boxes in image I_{i-1}, we compute, based on this homography, where the box corners would end up in image I_i to give a list of predicted box locations. Given this predicted list of boxes and the detected list of boxes for the image I_i, we update the probability of a given box b_i in the detected list as follows. We first find the box b'_j in the predicted list that has the highest overlap with this box. Given a box b and an estimate b' then the overlap score of the two is IoU $= \frac{b \cap b'}{b \cup b'}$. We then average the probabilities of b_i and b_j being objects to determine the updated probability of box b_i.

Post-Filtering

Given the set of classified boxes with probabilities computed using the neural network, we perform post processing to conflate these proposals. This step is necessary, as many overlapping boxes are actually of the same object, as shown in figure 1, yet we only want to extract salient, distinct objects.

The first step is to take the top ten candidate boxes for a given class (as we only want the most salient objects of each class) and the second step is to merge these candidates. In order to conflate only candidate boxes that are likely to be the same object, we merge those bounding boxes that have a high overlap and are classified as of the same class with a similar probability. Each box starts out in its own set, and we keep merging sets if there exists a pair of boxes, one from each set, such that the two boxes have a high overlap score, are of the same class, and the absolute value of the difference between their probability from the neural network is below a given threshold. Two boxes have a high overlap score if the IoU between the bounding boxes is greater than a given threshold. For each set, we construct a new box as follows. We average the location of the x/y coordinates for all boxes within the set to determine the new bounding box coordinates and set the probability of this box equal to that of the box in the set that has maximum probability.

OBJECT TRACKING

Given the initial system we have described in the previous sections, we also created a prototype for how one could track vehicles through frames and determine their position and intention (average velocity) in a light-weight manner. This prototype demonstrates the utility of the underlying landmark detection system.

In order to track dynamic objects, we use a graph-based approach. For each landmark class, we create a separate graph with a maximum depth of 20 frames. The nodes are the objects in a given frame and an edge connects two nodes in consecutive frames if the IoU between the bounding boxes is greater than a given threshold (set to 0.5 in this case). Given a graph, we extract objects based on the assumption that paths through a graph correspond to moving objects, so the longer the path, the more likely the path corresponds to a true object. Given a graph, for each node at the top most level of the graph (corresponding to the current frame), we determine the length of the longest path ending at this node as well as the average movement from frame to frame. To compute the intention, we average the change in the coordinates of the centre of the bounding boxes over the course of the corresponding path. This gives a vector describing the corresponding object's average velocity or its intention.

EXPERIMENTAL EVALUATION

In order to evaluate our approach, we first divided the dataset by two modes: route and road type. There are ten routes (one for each participant) and four road types. We first analyse the objectness step in isolation, then the entire system in isolation, and finally the pre- and post-filtering techniques. To ensure generalisability of our system, we use leave-one-out cross-validation. For each test/train set, the CNN was re-trained, and then used to detect and classify objects in the test set. We then discuss some of the practical considerations of our system. Finally, we present qualitative results from our system and from the prototype that determines a vehicle's intention.

Objectness evaluation results

First, we compared the result of the objectness approaches: EdgeBoxes and BING [11]. A bounding box was considered as correct using the approach of ImageNet. Two boxes are considered correct if their IoU score is > 0.5

As can be seen in table 2, EdgeBoxes outperforms BING on all landmarks *except* for *zebra light*. As a result, we use only EdgeBoxes when determining the bounding box proposals for the classification stage.

System Performance criteria

In order to evaluate the results of our system, we calculate the AP (average precision) measure for each class, as done in ImageNet. We also compute an overall AP score for all classes as opposed to *averaging* the results of each class, as done in ImageNet. Our aim is to strongly penalise false positives when detecting landmarks, which is vital in this context else the driver may become irritated and distracted.

Initial Classification Results

Since the number of object proposals affects the results of the CNN in terms of computational cost and accuracy, we first investigated this tradeoff. We chose to use 1400 candidate bounding boxes since it provides a good tradeoff between these two factors, as can be seen in figure 4.

	Bridge	Bus	Bus Stop	Car	Corner Shops	Gas St	Ped. Crossing	Row of Houses	Traffic Lt	Truck	Zebra Lt
EdgeBoxes	0.72	0.94	0.45	0.94	0.77	0.86	0.15	0.93	0.71	0.91	0.00
BING	0.28	0.36	0.11	0.18	0.16	0.00	0.04	0.61	0.48	0.26	0.05

Table 2. Percentage of objects found by class using 1000 candidate boxes and IoU of 0.5.

Figure 4. Demonstrates the tradeoff between AP score and computational cost for the entire pipeline (objectness and classification).

We compared the CNN classification to using a support vector machine (SVM). We trained the SVM on feature descriptors extracted from 10,000 randomly chosen training images using one-versus-one multi-class classification. When using the CNN and SVM to classify test images (the pre-cropped images of landmarks), the CNN clearly outperforms the SVM. The CNN yields an average of 92% accuracy over the route test set and an 88% percent accuracy over all road types versus 67% and 57% for SVM respectively. Both of these approaches perform significantly better than a naive majority vote baseline classifier that returns the most common class, which would have a 45.7% accuracy.

These differences are compounded when considering the overall AP scores of the SVM versus CNN. To compute the overall AP score, we used an IoU threshold of 0.5 on a subset of randomly chosen frames for each train/test set for both the CNN and SVM. For each train/test set, we computed the overall AP score and AP score per class with and without filtering using take-one-out cross validation on both routes and road types. The SVM has an overall AP score of 0.064 over all routes and 0.03 over all road types whereas the CNN has scores 0.32 and 0.28. See figure 5 for a break down by class. The error bars indicate one standard deviation.

The nature of the dataset makes our landmark classification task difficult, as the ground-truth placement was ambiguous (see figure 3). However, as can be seen by our results, the CNN approach is generalisable to unseen route conditions and road types. As shown in figure 5, our system can detect and classify landmarks such as *bus*, *car*, *row of houses*, etc.

Classification Results with Filtering

We then considered the improvements of pre- and post-filtering. In all of these experiments, we only considered

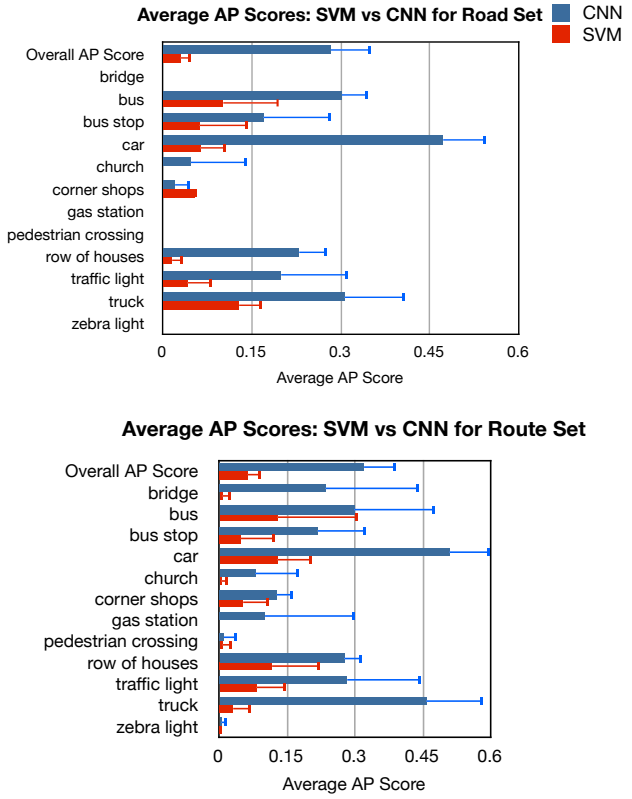

Figure 5. A comparison of the results using the SVM and CNN approaches. Clearly the CNN outperforms the SVM on both the route and road set and over all landmarks.

the CNN, as the initial results demonstrate that the CNN consistently outperforms the SVM approach.

Pre-Filtering

We compared the result of the object tracking approaches to two baselines: non maximal suppression and simply returning the top object proposals found by EdgeBoxes. Here we consider the results of participants 1 and 2 using the CNN model trained on the other participants. The AP score as a function of number of boxes returned is given in figure 6. Using object tracking improves performance when using only a very small number of boxes.

While using a GPU improves performance substantially, for some number of boxes the performance will degrade. Though the object tracking approach adds some overhead (< 0.1 seconds), this is negligible compared to the overhead of having to classify an additional batch of images. For example, one can use 50 object proposals with object tracking and obtain similar performance to using 150 boxes without filtering. This corresponds to a 2x speedup on a GPU with batch size 100.

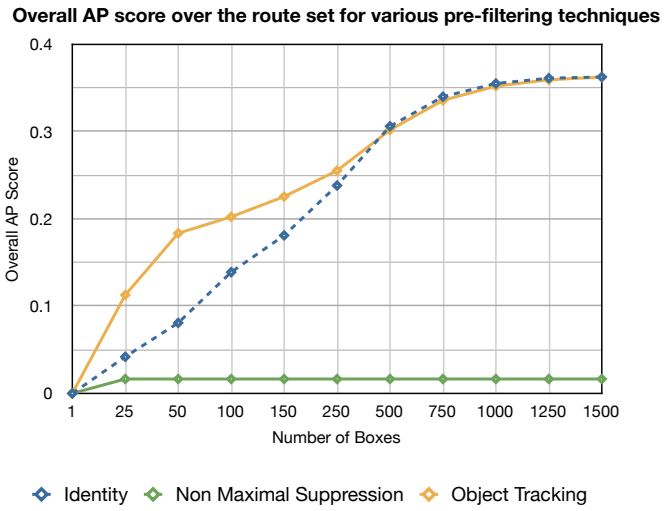

Figure 6. The AP score as a function of the number of object proposals returned for each image.

Post-Filtering

For post-filtering, we compared coalescing boxes to two baselines: non maximal suppression and performing no post-filtering. Graph 7 gives the overall AP result for these approaches for participants 1-4. (The error bars indicate one standard deviation.)

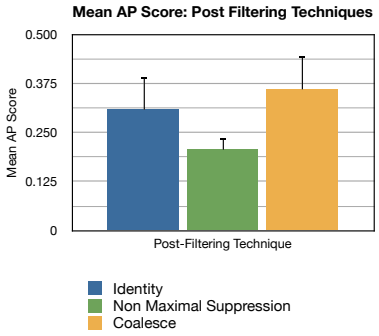

Figure 7. The AP score for the given post-filtering techniques.

Coalescing boxes performs consistently well across all dynamic and static landmarks and improves the overall AP score by approximately 0.05. This implies that this approach generalises well and is good in filtering the large number of boxes.

Practical Considerations

While the previous sections have considered the performance of our system, there are some practical considerations that we consider in this section. First, this system requires a large amount of hand-annotated data that is similar to the area of interest. A system trained in the British countryside will not necessarily generalise to an urban environment in China. Second, not all landmarks are universal. While bus stops may be distinctive and similar in one environment, they may not be so in another. As a result, one would have to consider having separate landmarks for separate environments or only using a very small set of landmarks (e.g. car and row of houses) that are seen in all environments. Irrespective of these considerations, the previous results demonstrate that this approach is promising, as we can achieve good performance with limited data in a challenging environment.

Sample Output

Finally, we showcase the results of our approach on two sample participants, participants 1 and 5. Figure 8 gives the landmark detection output for participants 1 and 5 without filtering using 1400 object proposals. Figure 9 gives the results using 50 object proposals, pre-filtering via object tracking, and post-filtering by coalescence on the same frame sequence. These sequences demonstrate how using more proposals gives the ability to find more difficult objects in the scene (such as the largely occluded *rows of houses*). However, this is at the expense of more false positives (e.g. the non-existent *corner shop*) and the same landmark (e.g. the *car* or *rows of houses*) being detected multiple times.

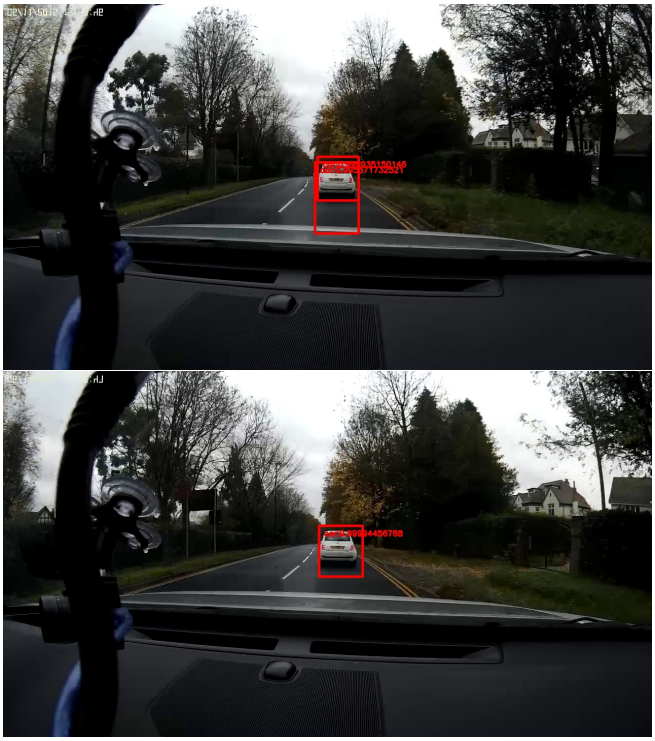

Figure 8. A sequence of frames with the objects extracted using 50 object proposals, pre-filtering based on object tracking, and post-filtering based on coalescing boxes. The red boxes indicate the *car* class.

Object Tracking

Finally, we showcase our object tracker for determining the intention of a vehicle, its velocity vector, in figure 10. These cropped frames overlay the output of our prototype on a frame taken from participant 5. This prototype is able to track a vehicle and determine its intention. It demonstrates how one could use our system to give more complex directions, such as 'follow the car in front' or 'follow the white truck around the roundabout'. It further demonstrates that our proposed system can be used to build a useful, powerful, and lightweight tool.

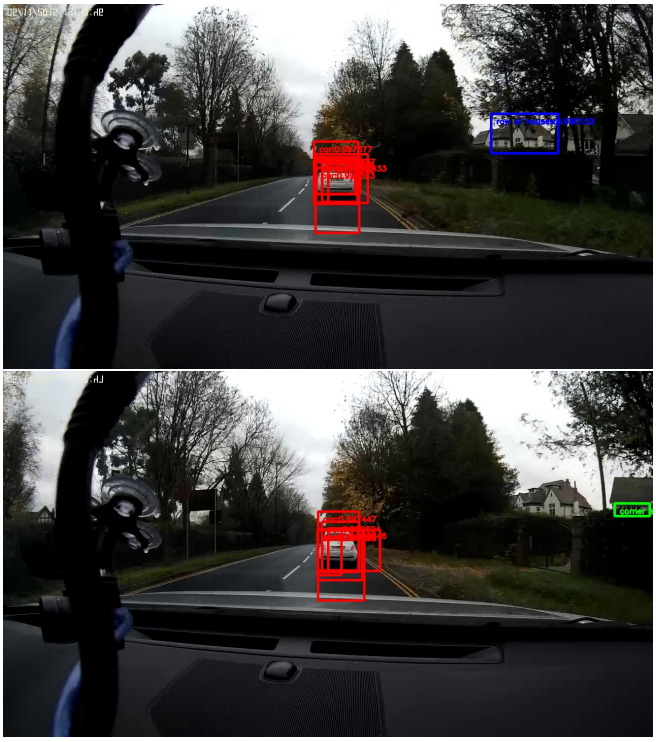

Figure 9. A sequence of frames with the objects extracted using 1400 **object proposals and no filtering. The red boxes indicate the *car* landmark class, the dark blue *row of houses*, and the green *corner shops*.**

CONCLUSIONS AND FUTURE WORK

We have presented a methodology for detecting landmarks in driving vehicles using state-of-the-art vision techniques. Our work has demonstrated good results for detecting landmarks but also highlighted the associated challenges. We have further described our 'in-the-wild' dataset and choice of twelve salient landmarks useful for in-vehicle navigational systems. Finally, we have demonstrated via a prototype how a simple lightweight method for determining a vehicle's intention could be built on top of our system. Our vision-based approach would be useful in advanced driver assistance systems (ADAS) as it is a cheap alternative to the expensive and computationally heavy approaches used by current systems (e.g. LIDAR for object detection and Google MAPs [20]). However, there are many challenges still to be overcome before such a system could be integrated in a commercial setting.

Future work would focus on improving classification accuracy with additional training samples, improving performance by implementing the system natively (e.g. in C++ as opposed to python), developing a prototype for drivers, and considering methods of improving directions. More descriptive directions would incorporate characteristics of the landmark (e.g. the 'red-car') and the context (e.g. that there is only one car in the driver's view) to improve the specificity of the direction and minimise driver confusion. We would also consider how to incorporate directions based on another vehicle's motion, as discussed in section 6.8. The prototype would use the landmarks detected to generate directions for drivers.

Figure 10. An image with the intention of the detected vehicles indicated. The cropped images highlight the detected objects, with the objects' velocity vectors overlain. These results are from using only 50 **classified boxes per frame with pre-filtering based on object tracking and post-filtering based on coalescing boxes.**

Future directions also include evaluating our prototype in a user study to determine which landmarks are useful, when directions should be given and user satisfaction when given different types of directions. For example, one would expect a black car by itself in a busy street to be more salient than a black car in a busy street. However, it is unclear whether a direction involving another moving vehicle (e.g. 'turn where that red car just turned') is more salient than one involving a stationary object (e.g. 'turn at the traffic lights'). We hope that our work encourages future research into context-aware navigational systems.

ACKNOWLEDGMENT

The work presented in this paper was funded and supported by Jaguar Land Rover, Coventry, UK.

REFERENCES

1. Stanley Michael Bileschi. 2006. *StreetScenes: Towards scene understanding in still images*. Ph.D. Dissertation. Citeseer.
2. Bosch. 2016. *Car Multimedia*. `http://www.bosch-presse.de/presseforum`
3. Norbert Buch, Sergio A Velastin, and James Orwell. 2011. A review of computer vision techniques for the analysis of urban traffic. *IEEE Trans. on Intelligent Transportation Systems* 12, 3 (2011), 920–939.
4. Gary Burnett. 1998. *Turn right at the King's Head: Drivers' requirements for route guidance information.* Ph.D. Dissertation. Loughborough University, UK. `http://dspace.mit.edu/handle/1721.1/14225`.
5. Gary Burnett. 2000. Turn right at the Traffic Lights: The requirement for landmarks in vehicle navigation systems. *Journal of Navigation* 53, 03 (2000), 499–510.
6. Gary Burnett, Darren Smith, and Andrew May. 2001. Supporting the navigation task: Characteristics of 'good' landmarks. *Contemporary ergonomics* 1 (2001), 441–446.
7. C Cabo, C Ordoñez, Silverio García-Cortés, and J Martínez. 2014. An algorithm for automatic detection of pole-like street furniture objects from Mobile Laser Scanner point clouds. *ISPRS Journal of Photogrammetry and Remote Sensing* 87 (2014), 47–56.

8. Claudio Caraffi, Tomas Vojir, Jura Trefny, Jan Sochman, and Jiri Matas. 2012. A System for Real-time Detection and Tracking of Vehicles from a Single Car-mounted Camera. In *ITS Conference*. 975–982.
9. Hui Chao, Sameera Poduri, Saumitra Mohan Das, Ayman Fawzy Naguib, and Faraz Mohammad Mirzaei. 2015. Sensor calibration and position estimation based on vanishing point determination. (2015). US Patent 9,135,705.
10. David M Chen, Georges Baatz, Kevin Köser, Sam S Tsai, Ramakrishna Vedantham, Timo Pylvä, Kimmo Roimela, Xin Chen, Jeff Bach, Marc Pollefeys, and others. 2011. City-scale landmark identification on mobile devices. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*. IEEE, 737–744.
11. Ming-Ming Cheng, Ziming Zhang, Wen-Yan Lin, and Philip Torr. 2014. BING: Binarized Normed Gradients for Objectness Estimation at 300fps. In *2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 3286–3293.
12. Ann K Deakin. 1996. Landmarks as navigational aids on street maps. *Cartography and Geographic Information Systems* 23, 1 (1996), 21–36.
13. Matt Duckham, Stephan Winter, and Michelle Robinson. 2010. Including landmarks in routing instructions. *Journal of Location Based Services* 4, 1 (2010), 28–52.
14. Andreas Geiger, Philip Lenz, and Raquel Urtasun. 2012. Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.
15. Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 580–587.
16. Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. 2016. Region-based convolutional networks for accurate object detection and segmentation. *IEEE Trans. on Pattern Analysis and Machine Intelligence* 38, 1 (2016), 142–158.
17. Aharon Bar Hillel, Ronen Lerner, Dan Levi, and Guy Raz. 2014. Recent progress in road and lane detection: a survey. *Machine Vision and Applications* 25, 3 (2014), 727–745.
18. Toru Ishikawa, Hiromichi Fujiwara, Osamu Imai, and Atsuyuki Okabe. 2008. Wayfinding with a GPS-based mobile navigation system: A comparison with maps and direct experience. *Journal of Environmental Psychology* 28, 1 (2008), 74–82.
19. Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. 2014. Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the ACM International Conference on Multimedia*. ACM, 675–678.
20. Michael T Jones, Brian McClendon, Amin P Charaniya, and Michael Ashbridge. 2011. Entity display priority in a distributed geographic information system. (April 26 2011). US Patent 7,933,897.
21. Matti Lehtomäki, Anttoni Jaakkola, Juha Hyyppä, Antero Kukko, and Harri Kaartinen. 2010. Detection of vertical pole-like objects in a road environment using vehicle-based laser scanning data. *Remote Sensing* 2, 3 (2010), 641–664.
22. Andreas Mogelmose, Mohan M Trivedi, and Thomas B Moeslund. 2012. Vision-based traffic sign detection and analysis for intelligent driver assistance systems: Perspectives and survey. *IEEE Trans. on Intelligent Transportation Systems* 13, 4 (2012), 1484–1497.
23. James Morris. 2012. Bosch Navigation 1.5 - Maps and Navigation. (2012). `http://www.trustedreviews.com/bosch-navigation-1-5-review-maps-and-navigation-page-2`
24. Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems*. 91–99.
25. Duncan P Robertson and Roberto Cipolla. 2004. An Image-Based System for Urban Navigation.. In *BMVC*. Citeseer, 1–10.
26. Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. 2011. ORB: an efficient alternative to SIFT or SURF. In *IEEE International Conference on Computer Vision (ICCV)*. IEEE, 2564–2571.
27. Grant Schindler, Panchapagesan Krishnamurthy, Roberto Lublinerman, Yanxi Liu, and Frank Dellaert. 2008. Detecting and matching repeated patterns for automatic geo-tagging in urban environments. In *IEEE Conf. on Computer Vision and Pattern Recognition*. IEEE, 1–7.
28. Alexander W Siegel and Sheldon H White. 1975. The development of spatial representations of large-scale environments. *Advances in child development and behavior* 10 (1975), 9–55.
29. Koen EA Van de Sande, Jasper RR Uijlings, Theo Gevers, and Arnold WM Smeulders. 2011. Segmentation as selective search for object recognition. In *Computer Vision (ICCV), 2011 IEEE International Conference on*. IEEE, 1879–1886.
30. Junko Yoshida. 2016. Mobileye, Nvidia (and Others) Spar over Cars | EE Times. (2016). `http://www.eetimes.com/document.asp?doc_id=1328655`
31. C Lawrence Zitnick and Piotr Dollár. 2014. Edge boxes: Locating object proposals from edges. In *Computer Vision–ECCV 2014*. Springer, 391–405.