2018 IEEE Intelligent Vehicles Symposium (IV)
Changshu, Suzhou, China, June 26-30, 2018

Learning to Drive: Using Visual Odometry to Bootstrap Deep Learning for Off-Road Path Prediction

Christopher J. Holder, Toby P. Breckon

Abstract— **Autonomous driving is a field currently gaining a lot of attention, and recently 'end to end' approaches, whereby a machine learning algorithm learns to drive by emulating human drivers, have demonstrated significant potential. However, recent work has focused on the on-road environment, rather than the much more challenging off-road environment. In this work we propose a new approach to this problem, whereby instead of learning to predict immediate driver control inputs, we train a deep convolutional neural network (CNN) to predict the future path that a vehicle will take through an off-road environment visually, addressing several limitations inherent in existing methods. We combine a novel approach to automatic training data creation, making use of stereoscopic visual odometry, with a state of the art CNN architecture to map a predicted route directly onto image pixels, and demonstrate the effectiveness of our approach using our own off-road data set.**

I. Introduction

A huge body of research has been conducted in the field autonomous driving, from both academia and the automotive industry, with much notable work in the areas of scene understanding [1] and road detection [2]. However, the more challenging problem of off-road autonomous driving has received relatively little attention, with only a limited body of work covering off-road scene understanding [3] and path detection [4]. In the off-road environment, path detection can be much more difficult than on-road, due to uneven terrain, hidden obstacles and an overall lack of structure. However there are many real-world applications for such technology, including in agriculture [5], military [6], and planetary exploration [7].

Convolutional Neural Networks (CNN) have demonstrated unprecedented results at a multitude of image classification tasks [8], revolutionising the field of computer vision in recent years. Loosely based on the biological brain, CNNs offer a 'black box' approach to machine learning, where the designer is aware of input and output data, but not necessarily of how that data is processed intermediately. Whilst this can create some interesting ethical and legal challenges, especially when dealing with autonomous vehicles where human lives may be at stake, it also means that CNN are likely to excel at tasks that humans can perform intuitively without relying on a structured set of rules, for example planning a safe route through an off-road environment.

Research supported by the Engineering and Physical Sciences Research Council, UK and Jaguar Land Rover.

Both authors are with the Department of Computer Science, Durham University, UK (phone: +44 191 334 1736; e-mail: c.j.holder@durham.ac.uk).

This idea underpins the concept of end-to-end autonomous driving, first proposed by Pomerleau in 1989 [9] with the Autonomous Land Vehicle in a Neural Network (ALVINN), which uses a neural network comprising a single fully-connected layer, taking a grayscale image and laser rangefinder data as input, trained to predict the steering wheel inputs made by a human driver. In 2004, the DARPA Autonomous Vehicle (DAVE) project [10] trained a more complex, six-layered network to drive a radio-control car in off-road environments, using data collected over several hours of human driving. More recent advances in deep-learning have led to the approach proposed in [11], which uses a network of 5 convolutional layers and 3 fully-connected layers, trained with 72 hours of human driving data, to successfully follow lanes on public roads.

In all three approaches, a neural network is fed an image from a vehicle mounted camera and trained to predict the steering input a human driver would make at the time the image was captured. However, we have identified three major limitations with this method: a) only immediate driving inputs are considered, with no thought as to how the vehicle path might change over time; b) driving inputs are learned for the characteristics of a specific vehicle, and so to apply the technique to a new vehicle with different steering characteristics, for example a tracked vehicle, would require the system be retrained on data from that vehicle; c) steering inputs do not necessarily relate consistently to the movement of the vehicle, especially in off-road environments where effective traction may be severely limited.

In this paper, we address these limitations by proposing a visual end-to-end autonomous driving approach, whereby a CNN is trained to map the future vehicle path directly to pixels in an image from a forward-facing camera. Training data is created automatically by using stereoscopic visual odometry [12] to track the motion of a human-driven vehicle through a sequence of images and then map this motion into the image space of the initial frame such that pixels that the vehicle traverses are labelled as 'path'. This addresses the identified limitations of existing end-to-end autonomous driving approaches [9,10,11], whereby only immediate driver input is predicted, by predicting a path that takes account of future changes in direction and does not rely on a direct link to driver inputs. Furthermore, the output of this process could be combined with semantic scene understanding, such as the approach described in [3], to semantically label path pixels

978-1-5386-4451-5/18/$31.00 ©2018 IEEE

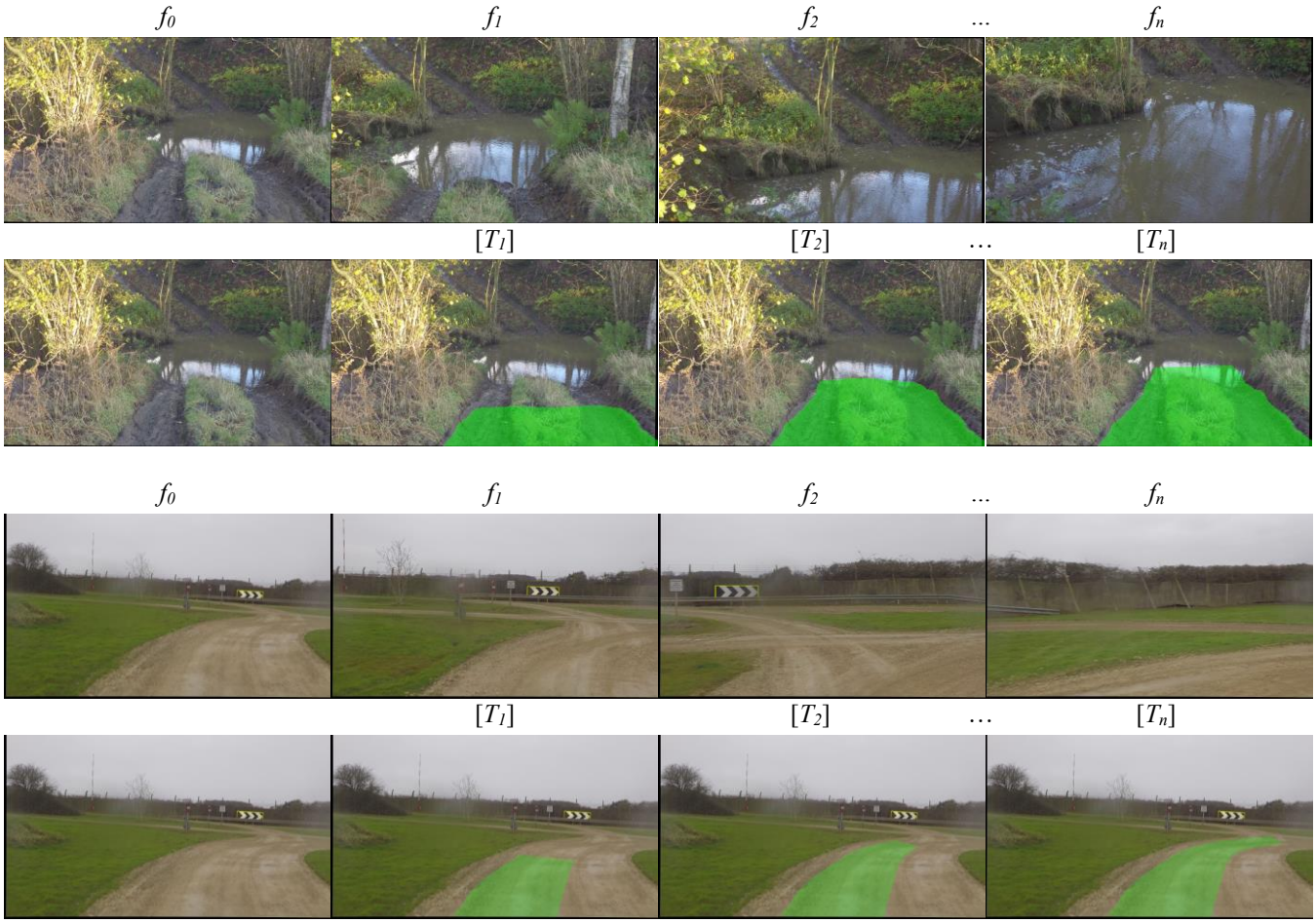

Fig. 1 example sequences from our data set: starting from frame f_0, transformation matrices $[T_1]$ to $[T_n]$ are computed for camera position in n subsequent frames, from which vehicle footprint can be calculated and translated back into image space so that path pixels can be labelled. Top row contains original frames f_0 to f_n, while bottom row shows aggregate computed footprint at each frame overlaid onto f_0.

and create awareness of upcoming terrain characteristics that can be used to setup vehicle parameters, such as suspension stiffness and gear ratio, for optimum traction and passenger comfort.

Subsequently, we use this automatically labelled data to train three state-of-the-art CNN architectures, each originally designed to perform a different segmentation or classification task. We also create a test dataset in the same manner, which we use to carry out a quantitative analysis of the performance of our approach using the three architectures.

II. Approach

The problem we are solving is the prediction of the path that a human driver would take through an off-road environment, made from a single image of that environment taken by a forward-facing vehicle-mounted camera. Our approach involves the automatic creation of training-data, whereby labelling of image pixels is automated using visual odometry to track vehicle motion under the control of a human driver, then using this data to train a CNN to map future vehicle path to image pixels, and by extension its real-world location.

A. Automated Dataset Creation

Our training data comprises individual colour images captured by a forward-facing vehicle-mounted camera and the corresponding automatically labelled binary ground truth images. Images were captured by stereoscopic camera mounted on the roof of an off-road vehicle at a rate of 10 frames per second. The 3D transformation matrix between each consecutive pair of stereo frames is computed using the

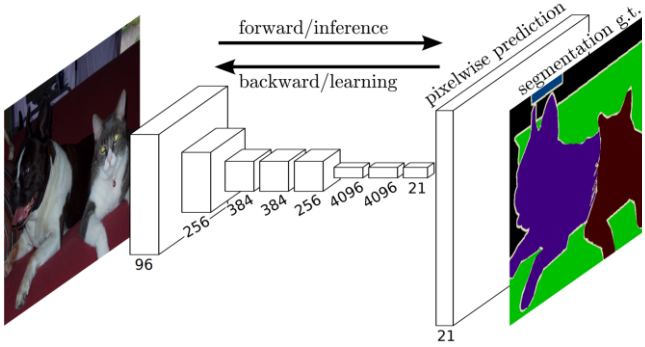

Fig. 2 An example of an FCN for performing a semantic segmentation task [16]

visual odometry approach of Geiger et al [12] such that the transformation between any pair of frames within a sequence can be obtained. As the stereo depth data is only required for the creation of this ground-truth data for training, a deployed version of this system would only require a single monocular camera.

To select the frames that will form our dataset, we begin at the start of a video sequence and look for the first frame containing movement, f_0, for which we create a label image L of matching dimensions with every pixel labelled as 'not path'. Subsequent transformation matrices [T_1], [T_2] … [T_n] are used to calculate the location and orientation of the camera, and by extension the vehicle footprint, in respective frames, f_1, f_2, … f_n, relative to the camera position in f_0. We then reproject this vehicle footprint back into camera space at f_0, labelling any pixel $L_{x,y}$ contained within as 'path'. This process, illustrated in Fig. 1, continues until the magnitude of the global transformation vector between the camera positions in f_0 and f_n is greater than distance threshold D, at which point the process is started again using the frame midway between f_0 and f_n as the new starting point. Empirically we found $D = 20m$ to be an appropriate value, as larger distances tended to cause noticeable propagation of transformation errors during the stereo visual odometry process.

We collected two sets of data, each using a different stereo camera (firstly a pair of GoPro Hero 4 [13] cameras placed on a 3D printed mount with a stereo baseline of 400mm, then a Carnegie Robotics MultiSense S21 [14], with a baseline of 210mm), vehicle and location to ensure variability within the data. In addition mirrored versions of every image were added to ensure an equal frequency of left and right turns. In total, our dataset comprises ~1000 RGB images of dimensions 512×288 along with corresponding binary ground truth images of the same dimensions. We use a 90/10 split to divide our data into training and test sets. The challenges associated with capturing off-road data, along with the lack of publicly available datasets, are significant areas for future consideration.

B. *Network Architectures*

We use our data to train three CNN architectures: Segnet [15], Fully Convolutional Network (FCN) [16], and U-Net [17], each of which represents a slightly different approach to the concept of an encoder-decoder architecture for pixelwise labelling and segmentation.

The design of SegNet, shown in Fig. 3, was motivated by segmentation and classification of road scenes for autonomous driving. It is a symmetrical architecture comprising an encoder based on the VGG16 [18] network, with fully connected layers removed, followed by a mirror-

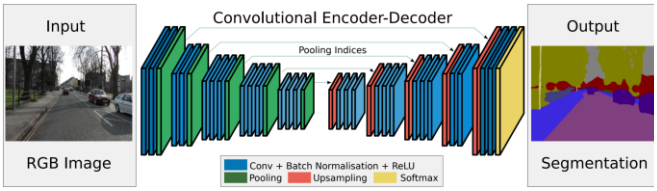

Fig. 3 SegNet CNN consisting of symmetrical encoder/decoder architecture [15]

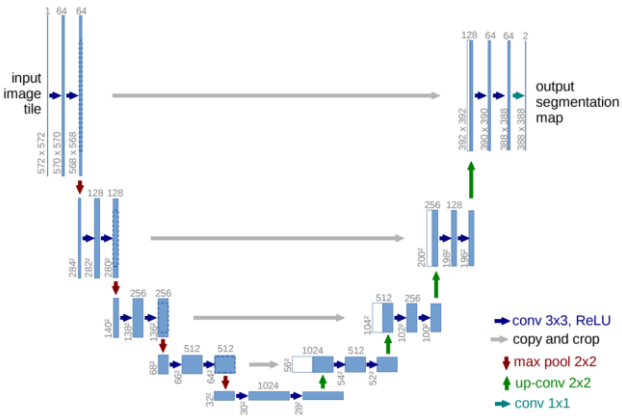

Fig. 4 U-Net CNN architecture [17]

image decoder. The encoder consists of 13 convolution layers, each using a *3 × 3* kernel followed by batch normalization and rectified linear units, interspersed with 5 max pooling layers, each downsampling its input by a factor of 2. The decoder replaces the max pooling layers with upsampling layers that use the corresponding max-pooling indices from the encoding phase to restore some of the spatial information that would otherwise be lost to pooling. The final layer of the network is a softmax classifier of dimensions $c \times w \times h$ where c is the total number of classes (in this case 2: {'path', 'not path'}), and w and h are the dimensions of the original input image.

FCN, shown in Fig. 2, was designed primarily for object segmentation and classification, and while versions have been implemented based on various network architectures, we use a version based on VGG16 for better comparison with SegNet. The fully connected layers originally used in VGG16 are replaced by convolution layers with large receptive fields, and dropout is used to reduce the risk of overfitting. A *1 × 1* convolution is used to predict class likelihoods at each scale during the decoder phase, which are then concatenated with the corresponding max-pooling output before upscaling. Output is again a softmax classifier of dimensions $c \times w \times h$.

U-Net, shown in Fig. 4, was initially motivated by segmentation of medical imagery, and is a more compact architecture than SegNet or FCN, comprising eighteen *3 × 3* convolutions and four each of pooling and upsampling operations. The output of each upsampling operation is concatenated with the input of the corresponding pooling operation in order to retain spatial information that would otherwise be lost through downsampling. Again, Output is a softmax classifier of dimensions $c \times w \times h$.

All three networks are implemented in Caffe [19], and in the case of SegNet and FCN we use models made publicly available by the original authors [15, 16].

The encoder section of each network is pretrained on the ImageNet [20] object recognition dataset, while the decoder is initialized with random weights - ImageNet presents a classification problem that outputs a single feature vector per image and so does not make use of a decoder. Our training data is then passed through each network in batches of 6 images, with the softmax classification error computed at every pixel to give a training loss value for the entire batch,

Fig. 5 Example output path confidence maps, a) before and b) after post processing

which is propagated backwards through the network using stochastic gradient descent with momentum (we set momentum=0.9 for all three network architectures). For SegNet we use a step function to decrease learning rate over training epochs, while the learning rate is fixed for U-Net and FCN. The initial learning rate is selected empirically to enable parameter fine-tuning without overfitting: for both SegNet and U-Net the value selected for initial loss was 1×10^{-3}, while for FCN it was 1×10^{-10}. A weight decay of 5×10^{-4} is also used in all three cases. Training continues until training loss is minimized such that no further gains are observed: this happened after 60,000 iterations for SegNet, 30,000 iterations for U-Net, and 100,000 iterations for FCN, most likely due to the lower learning rate.

C. Post Processing

CNN output is an array of dimensions $2\times w\times h$, where one channel contains a map of confidence values C_0 *{0→1}* that express the likelihood that a given pixel belongs to the class 'path', and the other channel contains a map C_1 *{0→1}* expressing the confidence that each pixel belongs to the class 'not path', such that C_0 and C_1 are the inverse of each other and $C_{0,x,y}+C_{1,x,y}=1$. As C_1 only contains information that can easily be inferred from C_0, it can be discarded.

An additional post-processing step is applied to C_0 to create the final path confidence map for evaluation against ground truth data. Firstly, we use stereo disparity data to compute the distance from the camera to each pixel location in the image, and any pixel further than the distance threshold $D = 20m$ is set to 0, as these pixels will have been ignored during the ground truth creation process. We then convolve the image with a Gaussian kernel of $\sigma = 6$ to smooth out any high-frequency noise.

Next, we set the confidence values of all pixels that are disconnected from the main path segment to 0. For the purposes of this step, we use a very low path confidence threshold δ and set all pixels where $C_0 < \delta$ to 0. Empirically, we found a value of $\delta = 0.025$ to give the best results. If the image contains multiple disconnected path segments, we determine which to consider the actual path by finding the pixel where $C_{0,x,y} > \delta$ with the smallest Euclidean distance from $C_{0,w/2,h}$ (the central pixel of the bottom row of the image), and performing a flood fill operation starting at that point that treats pixels with a value of 0 as component boundaries. Any pixel that is outside of the component filled by this operation is set to 0.

Some examples of output confidence maps before and after post-processing are shown in Fig. 5.

D. Evaluation Methodology

We evaluate the performance of the three trained networks, both with and without the post processing steps detailed above, using the 10% of our dataset that was set aside for testing.

In all cases we threshold the output path confidence map such that any pixel that satisfies the condition $C_{0,x,y} > 0.5$ is labelled 'path', and any that does not is labelled 'not path'. We compare this thresholded image to the ground truth data to compute the following four metrics across the entire test dataset: accuracy expresses the proportion of total image pixels that are classified correctly as either 'path' or 'not path'; precision expresses the proportion of those pixels that are labelled 'path' in the output image that are also labelled as such in the ground truth data; recall expresses the proportion of the pixels that are labelled 'path' in the ground truth data that are correctly classified as such in the output image; intersection over union (IoU) expresses the number of pixels that are correctly classified as 'path' as a proportion of the total number of pixels that are either labelled as such in the output image or labelled as such in the ground truth data. In all four cases, values are expressed in the range *{0→1}* with a higher value demonstrating superior performance.

III. End to End Learning

In addition to evaluating our visual path prediction approach, we also reimplement and evaluate the end-to-end-learning approach of Bojarski et al [11] in an off-road environment. Although no direct quantitative comparison can be made between the two approaches, we briefly outline our implementation and results.

A. Implementation

The end-to-end-learning network architecture consists of five convolution layers, three of which use a *5 × 5* kernel, the final two using a *3 × 3* kernel, followed by three fully-connected layers. Input is a single colour image of dimensions *256 × 136* and output is a single floating-point value *{-1→1}* that represents normalized steering wheel angle. The original authors used a significantly smaller input size of *200 × 66*, however we found that too much useful information was lost when performing such a drastic downsampling of our off-road dataset. Fig. 6 shows the network layout as implemented by the original authors.

We train this network for 100,000 iterations using ~20,000 images taken from our off-road data, each with an associated ground-truth normalised steering wheel angle, using a batch size of 6 images. Loss is computed as the mean squared error between predicted and ground-truth steering wheel angle across each batch, and backpropagated by stochastic gradient descent with momentum of 0.9. The initial learning rate is set to 2×10^{-4} and a step function is used to decrease this as training progresses.

B. Evaluation

The original authors of the end-to-end-learning approach we are implementing [11] did not conduct a quantitative evaluation of their approach, instead relying on the subjective

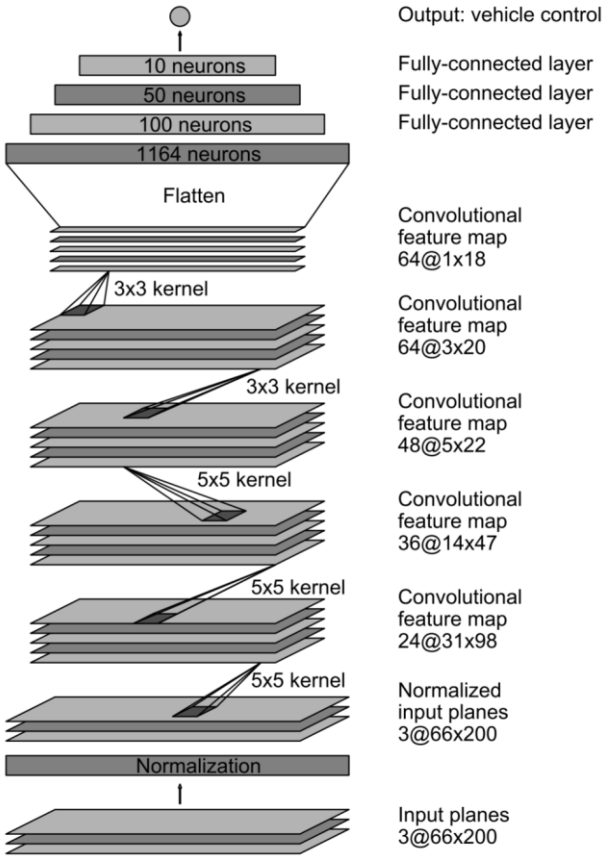

Fig. 6 The CNN architecture proposed by Bojarski et al [11] for end to end learning

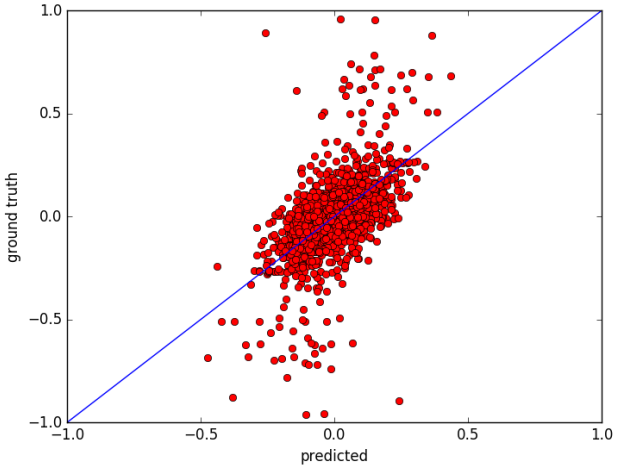

Fig. 7 Steering wheel angle values output by the end-to-end CNN for our test data set plotted against ground truth values.

measurement of the number of times a human passenger felt it necessary to override the driving decisions of the system. We evaluate performance by measuring the mean error between predicted and ground truth steering wheel angle. Over our test dataset, mean error between predicted and ground truth normalized steering wheel angle was 0.097. Unnormalised, this translates to an average error of 52.6° at the steering wheel, which, given the steering ratio of the vehicle, translates to approximately 3.5° of steering direction error at the road wheels.

Fig. 7 plots predicted steering angle against ground truth for our test data set, demonstrating a clear bias towards straight-ahead within the data which appears to be amplified in the predictions – despite ground truth samples existing right up to the full steering extents, no prediction demonstrated a magnitude of greater than 0.5. This suggests that the network optimised to a local minimum whereby it could minimize error by constraining its predictions close to the mean. This could potentially be addressed by modifying the training data so that a greater proportion of samples demonstrate sharp turns, or by modifying the training loss function so that loss is weighted proportionally to the distance a given sample lies from the mean. However, this does demonstrate an additional limitation of this approach that is especially significant in an off-road environment and that we aim to address with our own approach.

IV. Results

Our results are shown in Table 1 with illustrative examples shown in Fig. 8, based on an evaluation over our test dataset.

In terms of accuracy, the performance was similar across all three network types - SegNet and U-Net both demonstrated an accuracy of 0.95, while FCN did slightly worse with 0.94 – and post-processing had no clear advantageous effect. While these figures may appear highly impressive, accuracy alone is of limited utility in this case as it takes account of every pixel across the entire image when certain image

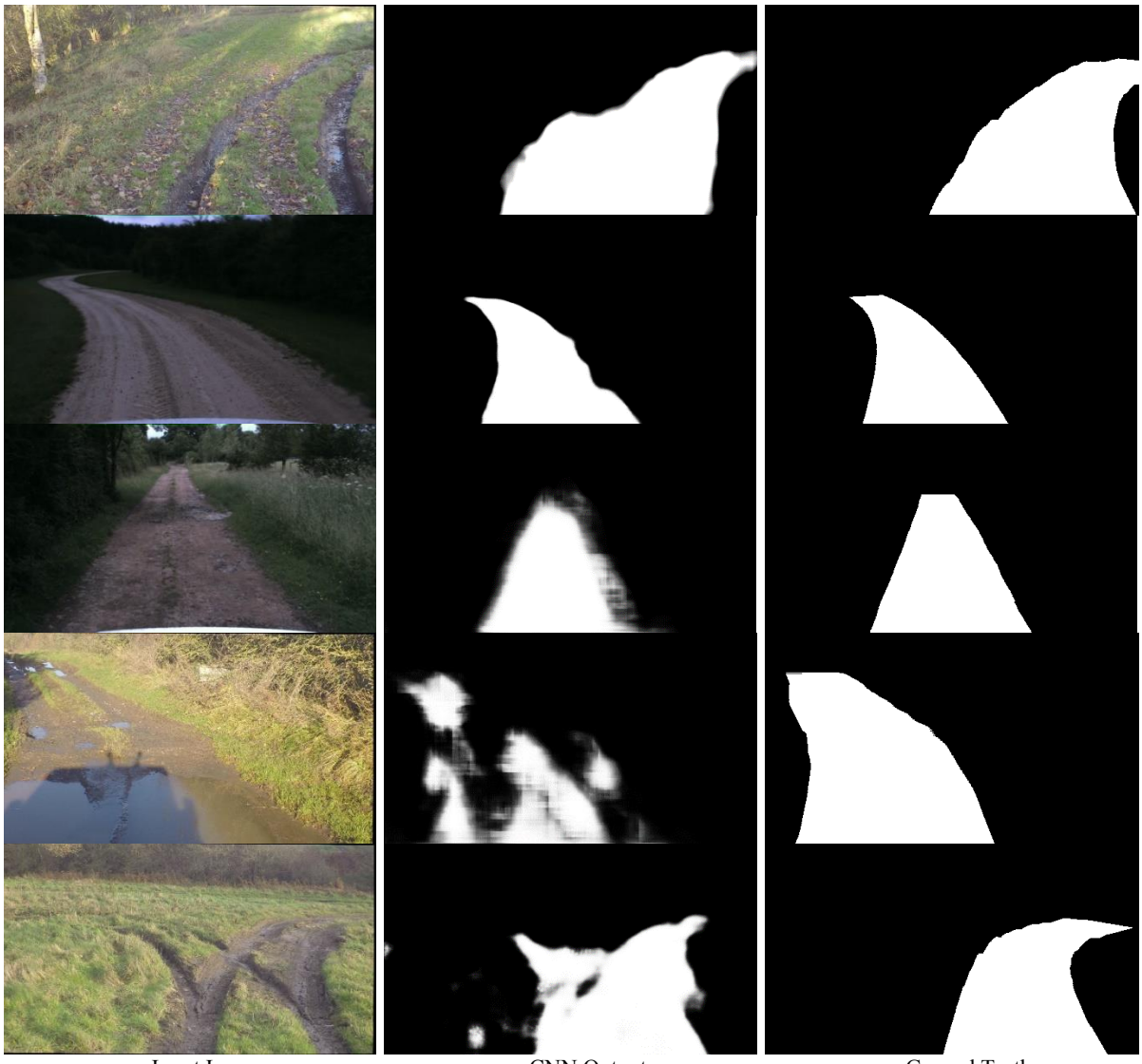

Fig. 8. Five samples from our test data set. Top three rows: good results obtained respectively from FCN, Segnet and u-net; Penultimate row: an example of a poor result obtained from u-net, in this case caused by a combination of shadow and water on the ground; bottom row: an example where the track forks in front the vehicle creating two valid paths, although our ground truth only includes the path that the vehicle originally took.

regions will consistently contain only one label across the entire dataset. To demonstrate this, we compared every output path confidence map to every non-corresponding ground truth image (i.e. those derived from different input images) and found the mean accuracy in this case to be 0.84. This implies that there could be a risk that a CNN optimize to a local minimum whereby it outputs a single average path that demonstrates high accuracy across its entire training dataset, however our results show that this did not happen.

Recall again demonstrated very similar performance from SegNet and U-Net, while FCN performs slightly worse, however in this case the post-processing steps degraded performance slightly: from 0.86 to 0.85 in the case of SegNet and U-Net and from 0.84 to 0.82 in the case of FCN. The opposite is true of precision, which increased slightly with post processing – from 0.84 to 0.85 in the case of FCN, from 0.86 to 0.88 in the case of SegNet and to from 0.86 to 0.89 in the case of U-Net. This is because the post- processing step will have removed or lowered the confidence value of more path pixels than it added.

Regarding intersection over union, SegNet performed best without post-processing (0.76), however U-Net output would appear to benefit the most from post-processing, with its IoU improving from 0.75 to 0.77. Again, FCN was the worst performer (0.72), and neither the results from it nor SegNet showed any improvement with post-processing. We believe IoU to be the most useful metric for measuring performance at this task as it takes account of both false positives and false negatives while ignoring true negatives, which make up a significant proportion of the data and are part of the reason accuracy is so high: in the experiment described above where path confidence maps were compared against non-corresponding ground truth images, the mean IoU was only 0.41.

	Accuracy	Recall	Precision	IoU
SegNet	**0.95**	**0.86**	0.87	0.76
SegNet post processed	**0.95**	0.85	0.88	0.76
FCN	0.94	0.84	0.84	0.72
FCN post processed	0.94	0.82	0.85	0.72
U-Net	**0.95**	**0.86**	0.86	0.75
U-Net post processed	**0.95**	0.85	**0.89**	**0.77**

Table 1 Results from each CNN architecture, both with and without post processing

V. CONCLUSIONS

In this work we have proposed an approach to off-road path prediction that combines a novel method for automatically creating labelled training data with state of the art convolutional neural network architectures designed for pixelwise labelling and segmentation tasks [15,16,17].

We created our own off-road dataset which we used to train networks based on the SegNet, Fully Convolutional Network, and U-Net architectures. These networks were then evaluated using our testing dataset, and all three demonstrated good performance. Overall, the best result was obtained from U-Net output, which considering its advantages in terms of speed and memory usage would make it ideally suited for deployment on an autonomous vehicle.

Our approach addresses several limitations with existing end-to-end driving methods [9,10,11]. Firstly, our approach predicts the vehicle path up to a set distance, while existing approaches only consider immediate control inputs. Secondly, our approach is not tied to any specific vehicle or steering apparatus, whereas existing approaches are only applicable to the vehicle used to generate training data, as any change to steering ratio, turning circle, or steering method, for example on a tracked vehicle, would require different control inputs. By removing the direct link between CNN output and vehicle controls, we also believe that our approach is better suited to off-road environments where traction may be limited, and vehicle movement might not necessarily correlate with steering direction. Finally, by implementing one of these approaches ourselves, we demonstrated an additional limitation in its proclivity for predicting low-magnitude steering inputs.

VI. FUTURE WORK

One aspect we did not explore with this work is temporal consistency – the data used is derived from video sequences, and so an additional constraint that filters predicted path across consecutive frames would likely improve performance. Another potential approach for achieving this could be the modification of CNN architecture such that some information from previous frames is retained to inform future predictions. The size and variability of available data is another area for consideration.

REFERENCES

[1] Ess A, Müller T, Grabner H, Van Gool LJ. Segmentation-Based Urban Traffic Scene Understanding. British Machine Vision Conference. 2009.

[2] Kong H, Audibert JY, Ponce J. General road detection from a single image. IEEE Transactions on Image Processing, vol. 19, no. 8, pp. 2211-2220, 2010.

[3] Holder CJ, Breckon TP, Wei X. From on-road to off: transfer learning within a deep convolutional neural network for segmentation and classification of off-road scenes. European Conference on Computer Vision. 2016.

[4] Thrun S, Montemerlo M, Dahlkamp H, Stavens D, Aron A, Diebel J, Fong P, Gale J, Halpenny M, Hoffmann G, Lau K. Stanley: The robot that won the DARPA Grand Challenge. Journal of field Robotics, vol. 23, no. 9, pp. 661-692, 2006.

[5] Subramanian V, Burks TF, Arroyo AA. Development of machine vision and laser radar based autonomous vehicle guidance systems for citrus grove navigation. Computers and electronics in agriculture, ol. 53, no. 2, pp. 130-143, 2006

[6] Manduchi R, Castano A, Talukder A, Matthies L. Obstacle detection and terrain classification for autonomous off-road navigation. Autonomous robots, vol. 18, no. 1, pp. 81-102, 2005

[7] Gennery DB. Traversability analysis and path planning for a planetary rover. Autonomous Robots, vol. 6, no. 2, pp. 131-146, 1999

[8] Krizhevsky A, Sutskever I, Hinton GE. Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems. 2012.

[9] Pomerleau DA. Alvinn: An autonomous land vehicle in a neural network. Advances in neural information processing systems. 1989.

[10] LeCun Y, Cosatto E, Ben J, Muller U, Flepp B. End-to-End Learning of Vision-Based Obstacle Avoidance for Off-Road Robots. Learning@Snowbird Workshop, April 2004.

[11] Bojarski M, Del Testa D, Dworakowski D, Firner B, Flepp B, Goyal P, Jackel LD, Monfort M, Muller U, Zhang J, Zhang X. End to end learning for self-driving cars. arXiv:1604.07316. 2016.

[12] Geiger A, Ziegler J, Stiller C. Stereoscan: Dense 3d reconstruction in real-time. Intelligent Vehicles Symposium. 2011.

[13] GoPro Inc, GoPro Hero 4 Black, www.gopro.com

[14] Carnegie Robotics LLC, "MultiSense Stereo Compact & Accurate 3D Data Collection," 2014. [Online]. Available: http://files.carnegierobotics.com/products/MultiSense_S21/MultiSense_Stereo_brochure.pdf. [Accessed April 2018]

[15] Badrinarayanan, V., Kendall, A., Cipolla, R. Segnet: A deep convolutional encoder-decoder architecture for scene segmentation. IEEE transactions on pattern analysis and machine intelligence, vol 39, no 12, pp.2481-2495, 2017.

[16] Long J, Shelhamer E, Darrell T. Fully convolutional networks for semantic segmentation. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2015.

[17] Ronneberger O, Fischer P, Brox T. U-net: Convolutional networks for biomedical image segmentation. International Conference on Medical Image Computing and Computer-Assisted Intervention. 2015.

[18] Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition. arXiv:1409.1556. 2014.

[19] Jia Y, Shelhamer E, Donahue J, Karayev S, Long J, Girshick R, Guadarrama S, Darrell T. Caffe: Convolutional architecture for fast feature embedding. 22nd ACM international conference on Multimedia. 2014.

[20] Deng J, Dong W, Socher R, Li LJ, Li K, Fei-Fei L. Imagenet: A large-scale hierarchical image database. IEEE Conference on Computer Vision and Pattern Recognition. 2009.