Single Camera Path Detection for Outdoor Navigation

Ondrej Jariabka*[†]
Marek Šuppa*[‡]
Ondrej Rudolf[§]
Supervised by: Pavel Petrovič[¶]

Faculty of Mathematics, Physics and Informatics, Comenius University in Bratislava

Abstract

The ability to find a path leading to an objective is one of the fundamental characteristics of autonomous mobile robots. In the recent years, a class of approaches which makes use of a combination of Computer Vision and Machine Learning has gained traction in the robotics community. In this work we examine the problem of Path Detection in the context of the Robotour competition, by analyzing solutions which were previously proposed in this context and use the aforementioned combination. Furthermore, we propose new models based on recent advances in using Convolutional Neural Networks for related Computer Vision tasks, as well as structural changes to the processing pipeline, which help to improve its performance. Finally, we evaluate all of the described methods on a dataset that is released as part of this work.

Keywords: path detection, computer vision, machine learning, outdoor navigation Convolutional Neural Networks

1 Introduction

Visual path recognition or road recognition is on of the fundamental properties of autonomous robots and vehicles. There have been many attempts to efficiently solve problem of recognizing and following the road, some as early as in 1980s. Many of these algorithms focus on certain features of the road or a path, or make use of expensive sensor such as radars and lidars ([5, 9]).

In the recent years, Artificial Neural Networks and Convolution Neural Networks [2] are more and more widely used in the robotics community. This trend is based on recent advances and increased popularity of these methods, which mostly stems from their superior performances on many Computer Vision tasks. Current approaches to object detection and image segmentation make essential use of these Machine Learning methods, as many of these networks provide state of the art solutions to some of the most fundamental problems [3].

In our work we examine the problem of path detection with specific focus on its application in context of the Robotour competition, and analyze currently proposed solutions. We propose improved models, which build upon previously used solutions, as well as a set of new models based on Convolution Neural Network. Additionally, we present a dataset used to evaluate our methods, along with methods that can be used to augment the data in it, thus considerably increasing its size. Finally, we present and interpret our results.

2 Related Work

Our work mainly builds upon the thesis of Michal Moravčík [6], which discuses previously used solution in the autonomous robot "Smely Zajko" [1]. The purpose of this robot is to compete in the Robotour competition, where it is tasked with delivering payload from point A to point B in previously unseen or unknown outdoor environment of a city park.

Since this work is a master thesis it also focuses on wide range of parts of the robot, such as hardware architecture or navigation based on Open Street Maps, but also introduces a new module that proceses input from camera mounted on the "Smely Zajko" robot. The input from this camera is first preprocessed and then sent to an Artificial Neural Network, which should detect which parts of the image are driveable segments of a path the robot can drive on. The author of this work tests multiple Artificial Neural Network architectures based on Multi-Layered perceptrons (MLPs), as well as various preprocessing methods. Images extracted from the camera are first converted to the LAB color space and then segmented into smaller pathes of size 5×5, which then serve as input to the network. Various handcrafted features are also discussed, such as histograms, medians of color channels, which are concatenated with image patch into one feature vector.

Another well established publication that tries to solve a very similar problem presents a solution for detecting road

*Both authors contributed equally to this work
[†]o.jariabka@gmail.com
[‡]marek@suppa.sk
[§]rower@naivenauron.com
[¶]ppetrovic@acm.org

[1]This name can be loosely translated in English as "Brave Rabbit"

is based on Neural Network with combination of Computer Vision algorithm used for feature detection [8]. In this work, the solution is based on extracting numerous features from images, which are then used in two separate Neural Networks, one of which is used to detected the horizon and the other to detect parts of the scene which can be used for navigation (such as the road for instance). The solution proposed in [6], which segments image into smaller squares and evaluate each of the separate squares, is also based on this paper.

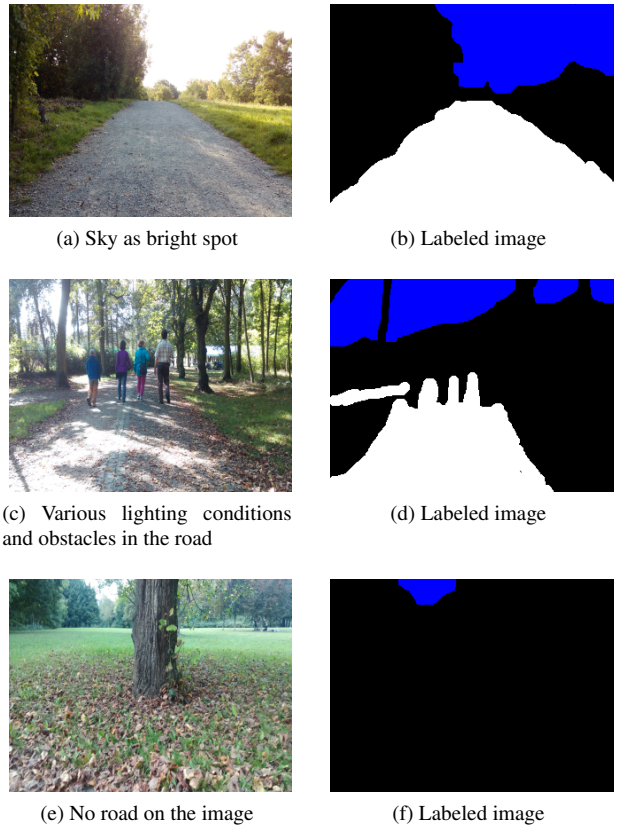
An approach similar to that of ours can be found in [1], where the authors consider the task of predicting whether a single pixel can be considered the street or not. This is a very similar problem to the one we address in this work, although it is solved in a different scenario (namely on the street in a self-driving car rather than in a city park in a mobile robot). This scenario requires the authors to ensure that their processing is fast (a prediction on a single image cannot take longer than 20ms) and also provides a standardized evaluation method in the form of the KITTI dataset.

There have also been attempts to solve the problem of outdoor navigation and road detection on mobile platforms or small devices, in order to help people with disabilities that also uses the aforementioned combination [10]. In this paper the authors proposed a solution based on an Artificial Neural Network and learning objective that should determine direction of the road seen by the camera.

3 Dataset

One of the biggest problem we identified in the related work was the lack of a standardized dataset for the task at hand. This means that the effectiveness of the proposed methods is difficult to assess, as their performance is usually reported on different datasets. This led us to create a dataset that can be used to evaluate different methods for off-road path detection. In this section we describe smely-zajko dataset which is released as part of this work.

This dataset consist out of 534 labeled images. These images were taken during previous years of Robotour competition, and also include images from near by parks and other off-road scenarios (Figure 1). Each image has its own corresponding mask which contains two classes - path, a road or any other navigable surface depicted in the image (white) and the sky (blue) (Figure 1d). Images were manually labeled by multiple human experts. They were taken with a low cost analog camera in various lighting and weather conditions, such as, sunny, cloudy, rainy, etc. The camera was mounted on a mobile robot, which was supposed to navigate in the proposed off-road settings by following the road, in order to achieve previously defined goal or specified location. For this reason some of the images contain large amount of blur, since robot was moving on uneven surfaces. Due to the long period of time during which images were were acquired (from 2011 to 2016), some images differ in resolution. Also, many of the included images were taken from different heights since the architecture of the mobile robot they originate from has changed over time. This closely resembles usual setup of home made or low cost devices that might be used to navigate in such a conditions.



(a) Sky as bright spot

(b) Labeled image

(c) Various lighting conditions and obstacles in the road

(d) Labeled image

(e) No road on the image

(f) Labeled image

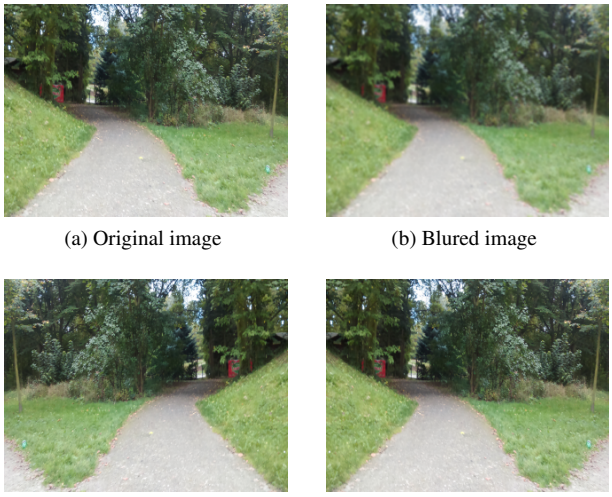
Figure 1: Sample images from smely-zajko dataset

Since most of the images were taken in parks, lighting conditions change drastically from image to image, mostly because of shadows casted by trees, bushes and other objects (Figure 1c). Images also contain people, animals and other objects commonly occurring near a road that have similar color, such as, benches or trash cans. Significant part of the dataset was collected in autumn and therefore the majority of images also contains leafs which cover parts of the road. The dataset also contains images whitch show only small parts of a road or no road at all (Figure 1e). In order to make it easily consumable and directly useful for other machine learning research, the dataset is split to train set and test set with standard 80:20 ratio.

3.1 Data Augmentation

Since the presented dataset is rather small, we needed to use heavy data augmentation for more complex models, such as Convolutional Neural Networks (CNN). The first form of data augmentation consists of simply flipping the

image on the horizontal axis together with its coresponding mask. This form of augmentation should somewhat help the model to achieve location invariant property. The next form is based on the fact, that some of the training images are blurred. It is not unusual for the robot to move on uneven surfaces, and so many of the images contain blur caused by the movement of the robot on these surfaces. Thus, we artificially blur images with Gaussian filter to simulate this behavior. The strength of the filter was determined by detecting how much blur the images already contain, by convolving the grayscale version of the image with a Laplacian filter, and computing the variance on the smoothed image as proposed in [7]. This should help the model to deal with heavy blur caused by movement of the robot in real life setting.



(a) Original image (b) Blured image

(c) Flipped image (d) Fancy PCA

Figure 2: An example of data augmentation.

The last form of augmentation that was applied to these images was the "fancy PCA" method proposed in [4], that does the Principal Components Analysis on color channels throughout the dataset. This method slightly changes the color pixel values of each image. In essence, it creates new images whilst still preserving labels of the image (Figure 2). All of these methods combined tripled our dataset in size, which helped our models to converge and allowed them to beter generalize to new data.

4 Experiments setup and tested approaches

The current body of academic literature that deals with the considered problem lacks an evaluation of standard baseline on a standardized dataset. As part of our work we therefore provide an evaluation of standard baselines on the smely-zajko dataset. Our models were trained on 80% of data points included in the dataset. We first segmented every input image into smaller pieces of the same size. Every model was evaluated with different sizes of this cut out window. These results can be found in a Section 6. The smaller cutouts where then fed to our models for training.

Every cutout was fed to a classification model. The model then classified each patch as either class 1 for a road or class 0, if no road was present in the cutout. This then also creates a mask of the input image as shown in Figure 3. These masks only consist of two values so they no longer represent a probability distribution over possible road positions. They also depict hard edges based on the size of a extracted patch.

Layer	Type	Shape
0	Input	$5 \times 5 \times 3$
1	Sigmoid Nonlinearity	
2	Dense Layer	8×1
3	Sigmoid Nonlinearity	
4	Output	

Table 1: Topology of the MLP-based network for pixel-wise classification

If we use a stride smaller than a size of a cutout window to segment the images, then multiple patches can be added together based on some merging scheme, such as for instance the mean of overlapping pixel values for regression or majority vote for classification. Results of these experiments for standard baselines and proposed new models can be found in Section 6.

5 Learning models

In this section we describe the used models along with the standard baselines, which were also evaluated. Linear regression was used as standard baseline for the regression approach. For classification approach, we used Logistic Regression, as well as Support Vector Machine (SVM) with both linear and Gaussian kernel without regularization. The Gaussian's kernel γ parameter was chosen by grid search and the optimal value was found to be 0.7. For both approaches we evaluated simple neural networks, also know as Multi Layered Perceptrons (MLP). Given the size of our dataset and complexity of this task, the general architecture of our models is rather simple (Table 1). Since the considered models ought to perform a classification task, at training time they are asked to minimize their binary crossentropy:

$$crossentropy(t,o) = -(t \cdot \log(o) + (1-t) \cdot \log(1-o))$$

Where o denotes the output of the network and t denotes the target from the ground truth. These models were trained using the RMSProp optimizer with the learning rate set to 0.01.

Layer	Type	Shape
0	Input	$20 \times 20 \times 3$
1	Convolution	10 filters, each 5×5
2	ReLU Nonlinearity	
3	Convolution	10 filters, each 5×5
4	ReLU Nonlinearity	
5	Pooling	2×2
6	Output	
7	SoftMax Nonlinearity	

Table 2: Topology of the CNN for pixel-wise classification

Finally, we also introduce a Convolutional Neural Network with architecture proposed in Table 2. The core of the network consists of two layers of 10 trainable convolutional filters of size 5×5, followed by the Rectified Linear Unite (ReLU) nonlinearity. The following pooling layer of size 2×2 then downsamples the data going through the network and prepares it for the final classification.

One may note that this architecture is quite similar to the one described in [1], which is not surprising, given the similarity of the considered tasks.

Layer	Type	Shape
0	Input	$20 \times 20 \times 3$
1	Convolution	10 filters, each 5×5
2	ReLU Nonlinearity	
3	Convolution	10 filters, each 5×5
4	ReLU Nonlinearity	
5	Pooling	2×2
6	Convolution	10 filters, each 5×5
7	ReLU Nonlinearity	
8	Convolution	10 filters, each 5×5
9	ReLU Nonlinearity	
10	Pooling	2×2
11	Dense Layer	50
12	ReLU Nonlinearity	
13	Output	
14	Sigmoid Nonlinearity	

Table 3: Topology of the CNN for pixel-wise classification

Dropout layers were also added between convolution layers with dropout probability of 0.5 to prevent overfitting. The architecture of this network was based on our experiments with other layouts. The network is rather shallow, which is due to the previously mentioned reason of complexity of the task and the size of the dataset. The size of the output layer of these networks networks was set to two neurons.

Encouraged by the results of this network we also experimented with a bigger architecture described in Table 3. While this model has many more trainable parameters and therefore may not be well suited for deployment on a robot, where the processing speed is crucial, we investigated this model to see whether a bigger amount of parameter can help improve the classification results. To this end a new fully connected (also denoted as "Dense") layer was added to the architecture.

(a) Original image (b) Prediction

Figure 3: Result of logistic regression with window 10×10.

The CNN models were trained using the Adam optimizer with the learning rate set to 0.01, along with weight decay of 0.00005 per epoch. Similar to the MLPs discussed above, they also optimized the binary cross entropy.

6 Results

As we can see in Table 4, the best results were achieved by the Logistic Regression model with window size of 20×20. Although the model that was trained with window 40×40 window achieved better training accuracy, we can see that test accuracy was lower than previous models and training accuracy spiked quite high. This is most probably caused by the 40×40 model overfitting the training data. The difference between the first three models is marginal for simple models such as Logistic Regression (LR). This led us to believe that such models may be a good and robust baseline.

window size	Train accuracy	Test accuracy
5×5	0.8706	0.8518
10×10	0.8801	0.8623
20×20	0.8929	**0.8670**
40×40	**0.9521**	0.8375

Table 4: Classification results of Logistic Regression models.

The data presented in Table 6 suggest, that classification of these patches is quite simple task which leads to severe overfitting of a more complex model such as an SVM. The only sound results were achieved by choosing a large-enough cutout window of size 40×40, and even despite this model still overfitted by quite a large margin.

As we can see in Table 5, the CNN model achieved the best results by almost more than 6% (Figure 5). The fact that CNN did not overfit as much as SVM can be attributed

model	type	train acc	test acc
MLPs	8 neurons	0.8215	0.8190
MLPs \w LAB	12 neurons	0.8297	0.8243
LR		0.8929	0.8670
CNN	1 conv block	0.9133	0.9125
CNN	2 conv blocks	**0.9413**	**0.9223**

Table 5: Evaluation of considered models for pixel-wise path classification. Note that only the best models of each type are presented.

to the way these models are predicting probability of each class and then the highest probability is picked as final prediction. This is bit more complex then just separating the data in order to perform binary classification. To the best of our knowledge, this newly proposed CNN model outperformed the previous state-of-the-art results on the introduced dataset at the time of this writing.

Since the first layers of CNN models consist of trainable filters, it is quite interesting to visualize them. By doing that, one may gain an intuition regarding what does the model focus on in order to discriminate the classes considered by the task at hand. As we can see in Figure 4 many filters correspond to gradient transition between brown and green or gray pixels, which is not supprising considering that many images in the training dataset contain transitions from brown graass or leaves to path. Also the third filter in the first row of Figure 4 has high activation on regions of images that consinst mainly out of green pixels, which can be interpreted as parts of images that have no navigatable surface. A fair amount of filters contains blue pixels, which can be coused by the cases where the path in the training images stretches all the way to the horizont. Moreover, given various lighting condition during which the images where taken, in many of them the road has blue tint which can also be seen in Figure 5. The big amount of blue pixels in the visualized filters is therefore not very surprising.

7 Conclusions

In our work we discuss the problem of off-road navigation using image input. Based on analysis of relevant work we propose a set of standardized baselines for a classification approach, which can be used to assess the difficulty of the task at hand. Moreover, in order to standardize the comparison of methods used for this task we introduce a new dataset called `smely-zajko`, which can be utilized for further research in the are a of autonomous off-road navigation. To the best of our knowledge, this is the only publicly available dataset for off-road path detection at the time of this writing. Finally, we proposed a new model based on Convolutional Neural Networks which outperformed all of the other models considered so far, and to the best of our knowledge outperformed the previous state-of-the-art on this task.

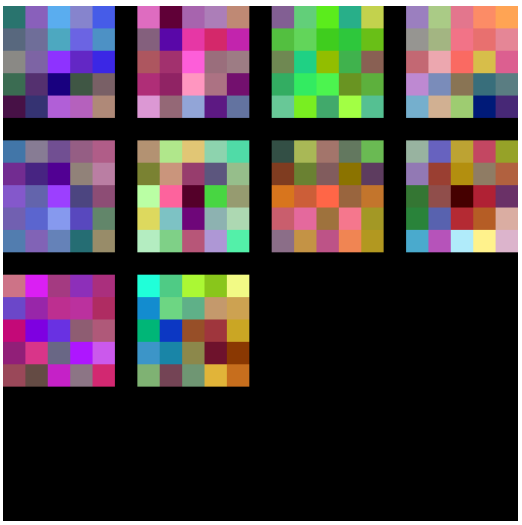


Figure 4: Visualized first layer weights of CNN model.

Kernel	Train accuracy	Test accuracy
Linear	0.9688	**0.8191**
RBF	**1.0**	0.5841

Table 6: Classification results of SVM models with window of size 40×40.

Given the exhaustive analysis that was done as part of this work, we believe that as stated, this task may soon be solved and the studied models may be used as building block for an end to end navigational solution. In that case the task of navigation as a whole will be considered, and trained in an end-to-end manner. Given their well established generalization properties, we believe that the CNN-based models may provide a good set of building blocks in this regard.

The code used to reproduce the experiments conducted as part of this work, as well as the newly introduced dataset can be found at `https://github.com/NaiveNeuron/smely-zajko-dataset`.

References

[1] Sebastian Bittel, Vitali Kaiser, Marvin Teichmann, and Martin Thoma. Pixel-wise segmentation

Figure 5: Visualized predictions of categorical models

of street with neural networks. *arXiv preprint arXiv:1511.00513*, 2015.

[2] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*, 2015.

[4] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[5] Jesse Levinson, Jake Askeland, Jan Becker, Jennifer Dolson, David Held, Soeren Kammel, J Zico Kolter, Dirk Langer, Oliver Pink, Vaughan Pratt, et al. Towards fully autonomous driving: Systems and algorithms. In *Intelligent Vehicles Symposium (IV), 2011 IEEE*, pages 163–168. IEEE, 2011.

[6] Michal Moravčík. Autonomous mobile robot for robotour contest. Master's thesis, Faculty of Mathematics, Physics and Informatics, Comenius University in Bratislava, 2013.

[7] José Luis Pech-Pacheco, Gabriel Cristóbal, Jesús Chamorro-Martinez, and Joaquín Fernández-Valdivia. Diatom autofocusing in brightfield microscopy: a comparative study. In *Pattern Recognition, 2000. Proceedings. 15th International Conference on*, volume 3, pages 314–317. IEEE, 2000.

[8] Patrick Y Shinzato, Valdir Grassi, Fernando S Osorio, and Denis F Wolf. Fast visual road recognition and horizon detection using multiple artificial neural networks. In *Intelligent Vehicles Symposium (IV), 2012 IEEE*, pages 1090–1095. IEEE, 2012.

[9] Chris Urmson, Joshua Anhalt, Drew Bagnell, Christopher Baker, Robert Bittner, MN Clark, John Dolan, Dave Duggins, Tugrul Galatali, Chris Geyer, et al. Autonomous driving in urban environments: Boss and the urban challenge. *Journal of Field Robotics*, 25(8):425–466, 2008.

[10] Zejia Zhengj and Juyang Weng. Mobile device based outdoor navigation with on-line learning neural network: A comparison with convolutional neural network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 11–18, 2016.