Autonomous Wheeled Robot Navigation with Uncalibrated Spherical Images

Lingyan Ran(✉), Yanning Zhang, Tao Yang, and Peng Zhang

School of Computer Science and Engineering,
Northwestern Polytechnical University, Xi'an, Shaanxi, China
Lingyanran@gmail.com

Abstract. This paper focuses on the use of spherical cameras for autonomous robot navigation tasks. Previous works of literature mainly lie in two categories: scene oriented simultaneous localization and mapping and robot oriented heading fields lane detection and trajectory tracking. Those methods face the challenges of either high computation cost or heavy labelling and calibration requirements. In this paper, we propose to formulate the spherical image navigation as an image classification problem, which significantly simplifies the orientation estimation and path prediction procedure and accelerates the navigation process. More specifically, we train an end-to-end convolutional network on our spherical image dataset with novel orientation categories labels. This trained network can give precise predictions on potential path directions with single spherical images. Experimental results on our Spherical-Navi dataset demonstrate that the proposed approach outperforms the comparing methods in realistic applications.

1 Introduction

In the field of autonomous driving, vision-based navigation research has long been a hot topic. On various platforms, like quadrotors, self-driving cars, and robotics, multiple types of sensors and cameras have been facilitated to improve the machine intelligence. In this paper, we propose an alternative approach of using a convolutional neural network (CNN) for addressing the problem of autonomous robot navigation with spherical cameras. Details of our device and framework are shown in Fig. 1.

Accurate position and orientation estimation of a camera is one of the most important tasks for robotic navigation problems. In most scenarios, simultaneously building up the 3D maps of the world while tracking the location and the orientation of the camera is a common approach for a navigation task. In the last two decades, simultaneous localization and mapping (SLAM) method and various derivatives have been dominating this topic. Various systems have been proposed, for example, PTAM [1] etc. Recently, Caruso et al. [2] successfully accomplish a SLAM system for direct use of omnidirectional cameras.

All the SLAM based systems have achieved great performance and offer users a set of candidate solutions for navigation tasks. However, when it comes to

Z. Zhang and K. Huang (Eds.): IVS 2016, CCIS 664, pp. 47–55, 2016.
DOI: 10.1007/978-981-10-3476-3_6

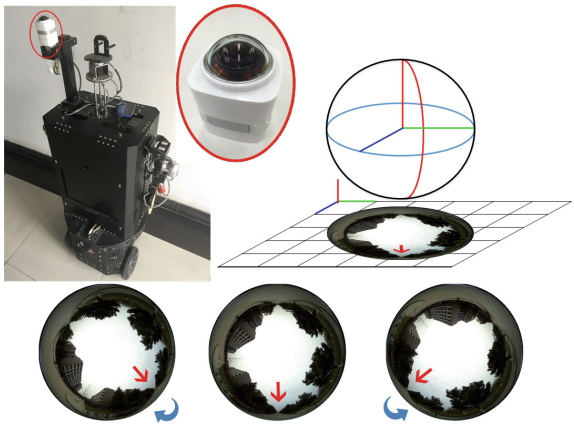


Fig. 1. A spherical driving robot. For spherical images, the central bottom pixels are the front heading of a robot and the red arrows points to the potential path. Our motivation is to generate navigation signals with the severely distorted spherical images. (Color figure online)

moving platforms, like tablet PCs, quadrotors, and moving robotics as in our case, limited computational capabilities pushes the SLAM based navigation task a higher complexity level. Therefore, seeking a low-cost solution is important.

Inspired by the human vision system, another group of methods focus on the problem of road detection and trajectory planning. An intelligent robot could follow the visual paths via local road segmentation [3] and trajectory prediction [4]. For example, Lu et al. [5] build up a hierarchical vision sensor based method for robust road detection in challenging road scenes. Chang et al. [6] present a vision-based navigation and localization system using two biologically-inspired scene understanding models which are studied from human visual capabilities.

In spite of their simplicity, human vision inspired methods highly rely on local features for segmentation tasks and usually lose a whole sense of the environment. What's more, for spherical images, the calibration and wrapping process further complicates those solutions.

As early in the 90s, Pomerleau [7] has treated the road following task as a classification problem. Later on, Hadsell et al. [8] develop a similar system for navigation in unknown environments. Recently, Giusti et al. [9] define the camera orientation estimation as a three-class classification problem (Left, Front, Right) and capture a set of forest trail images with 3 head-mounted cameras, each pointing to one direction. Given one frame image, their trained model can tell a robot whether it need to make a left/right turn or keep straight forward. One drawback here is that the fixed cameras are not flexible for tasks with higher orientation precision demands. For more turning control signals, more cameras are needed. This may not be a good solution for complex applications.

Inspired by [7], in this paper, we propose to make navigation predictions by classifying spherical images into different rotation categories. One characteristic

of spherical images is that it captures a wide view of 360° natural scenes and thus gives much more potentially useful information than those plane images. For generating training images of different orientations, we could just make a rotation of the original front viewed images and do not need to recapture the scene. The contributions of our paper are as following:

- We formulate this spherical camera navigation task as an image classification problem and design an end-to-end CNN for it. It's efficient for reality applications, where we don't need to do pixel-wise labelling of roads or build a complex 3D world with unreasonable high computation and memory costs.
- We build one spherical navigation dataset and raise a novel labelling strategy, which enables us to generate various training images for different complexity and precision applications with a minor change in the labelling process.

2 Navigation Network

Deep convolutional networks have been widely used in many computer vision tasks, e.g. image recognition [10], image segmentation [11], object detection [12] etc. In this paper, we train a convolutional network to estimate accurate robot heading pose orientation in the navigation task using raw spherical images. Details on the problem formulation and the network design are in the following.

2.1 Navigation via Classification Formulation

Accurate position and orientation estimation of a robot is a basic step for building navigation systems. Other than doing high computation cost processes like SLAM or lane segmentation, a novel approach of formulating navigation as an image classification problem is described in details in this subsection.

Figure 1 illustrates a general view of our capturing platform. An upward-looking spherical camera is fixed on top of a wheeled robot, which can capture images with detailed information of its 360° surroundings. Considering when the robot is wandering around within a campus, our problem is to tell the robot to turn left/right or keep straight forward with spherical frames in real-time.

Given a set of N spherical images $X \in \mathbb{R}^D$, we want to get the potential navigation direction $y \in Y$, with a range of $[-k, +k]$, where k equals to $1, 2, 3, \ldots$ for different complexities. Let $y = 0$ denote going straight forward, $y < 0$ for turning left, and $y > 0$ for turning right. For learning the model, the goal is to minimize the global prediction error of: $L = \sum_{n=1}^{N}[1 - \delta(\hat{y}_n, y_n)]$, in which $\hat{y}_n = F(x_n)$ is our prediction for sample (x_n, y_n), $\delta(a, b)$ equals to one if $a = b$ and zero otherwise. The nonlinear warping model $F(x; w, b)$ will be learned next.

2.2 Network Structure

Inspired by the extraordinary works of Alexnet [10] and its extension Giusti et al. [9], we adopt a similar CNN for the spherical image classification problem.

Table 1. The network structure and layer setups of CNN used in our experiments.

	Features	[9]	Model 1	Model 2
0	$3 \times 101 \times 101$	Inputs	Inputs	Inputs
1	$32 \times 98 \times 98$	Conv	Conv	Conv
2	$32 \times 49 \times 49$	Pool	Pool	Pool
3	-	Tanh	P/ReLU	BN+PReLU
4	$32 \times 46 \times 46$	Conv	Conv	Conv
5	$32 \times 23 \times 23$	Pool	Pool	Pool
6	-	Tanh	P/ReLU	BN+PReLU
7	$32 \times 20 \times 20$	Conv	Conv	Conv
8	$32 \times 10 \times 10$	Pool	Pool	Pool
9	-	Tanh	P/ReLU	BN+PReLU
10	$32 \times 7 \times 7$	Conv	Conv	Conv
11	$32 \times 3 \times 3$	Pool	Pool	Pool
12	-	Tanh	P/ReLU	BN+PReLU
13	288–200	FC1	FC1	FC1
14	-	Tanh	P/ReLU	PReLU
15	200–K	FC2	FC2	FC2

A convolutional network conventionally consists of several successive pairs of convolutional layers (Conv), pooling layers (Pool) and fully connected layers (FC). Our network consists of four Conv layers and two FC layers. Each Conv layer is followed by a max-pooling layer to enhance the local contrast.

Table 1 presents the detailed layer-wise network setups. Since Giusti et al. [9] choose to use the scaled hyperbolic tangent activation function (Tanh) and do not give much analysis on the non-linear warping functions, we first put some effort on that. We find out that both Rectified Linear Units (ReLU) [13] and Parametric Rectified Linear Units (PReLU) [14] outperform Tanh units. Here we choose the better one (PReLU in model 1, Table 1) for later experiments.

Optimizing a deep neural network may not be easy because of the gradient vanishing/exploding trouble, and it is highly likely that our model may get stuck in a saddle point and could not well tune the lower level features. This is especially serious for spherical images, as most of them are similar in appearance and the gradient may not be of much different. We here adopt the Batch Normalization (BN) [15] method in model 2, Table 1, which forces the network activations to vary across examples. In this way, we could not only accelerate the training of networks but also achieve a better classification performance.

During training, the networks are optimized with the adagrad strategy, which relieves us from trying learning rates and momentum hyper-parameters back and forth. Further details and analysis on the performance of the network configurations are given in the experiment section.

3 Spherical-Navi Dataset

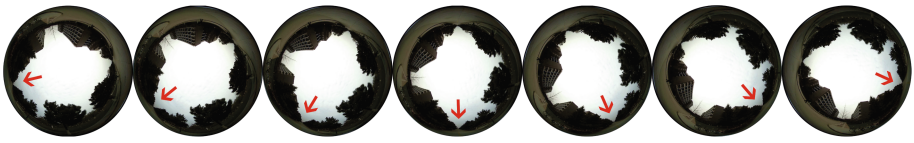
To train a model that could accurately classify images, a dataset with balanced class distribution is required. Since a spherical image has the characteristic that the orientation change of camera viewing only results in a rotation of the spherical images, we could generate simulation images with various robot orientations from one single image. In this way, the drawback of [9] being not flexible is not a serious problem anymore.

Data Capturing. In total, ten video sequences are captured using a spherical camera when the robot is wandering around within a campus. Each video is captured at 60 frames per second with a high resolution of 4608×3456 pixels. To ensure that neural networks actually model the whole dataset well instead of just memorizing the scene, we carefully design the navigation path so that there are fewer overlaps among those video clips.

Data Labelling. For a navigation task with K turning control signals, we actually can generate any orientation posed image by a specific rotation of the original one. For stability, when generating images for different classes, a larger field is given to front views. That's because, in this way, the robot has a higher probability of keeping on going straight other than frequent left/right turn interruptions. Figure 2 gives an illustration of how the image of different classes look like on one same street corner with $K = 7$. Here, we have those random rotations of $(-30°, +30°)$ as front view (i.e. k = 0) and the rest regions are equally divided into six orientation fields.

Data Argumentation. Due to the robot moving jitter, some frames may not contain satisfying high-quality contents. Without loss of generality, in this paper, video sequences are resampled at two frames per second. In total, there are 8000 images for training and 6000 images for testing.

Before feeding into the network, we also apply some pre-processes on the spherical images. Since the spherical camera is fixed upward looking, it captures much unnecessary information to the prediction, such as the central sky pixels in Fig. 2. In the training procedure, those central pixels are masked out. In our experiment, we get about 1% improvement on average when the central pixels are masked out. All images are then normalized in the YUV space with zero-mean and unit variance to reduce the lighting changing effects.



(a) $+70°$ (b) $+50°$ (c) $+30°$ (d) $0°$ (e) $-30°$ (f) $-50°$ (g) $-70°$

Fig. 2. Sample images from 7-class configuration (best viewed with zooming in). The orientation change of a robot corresponds to a rotational difference of a spherical image. (Color figure online)

4 Experimental Results

4.1 Network Setup and Training

Well designed training strategy is essential for good performance. In this paper, three models (as detailed in Table 1) are analyzed on the Spherical-Navi dataset. All of them share the common structure with the filter size equals to four.

For initialization, all the weights of neuron connections are initialized using the strategy in [14] and the biases are set to zero. During training, it would be wise to set a higher learning rate for models with PReLU activations than those with ReLU neurons. In our settings, the learning rate equals to 1e-5, 1e-4 for ReLU and PReLU respectively. And when the training loss first stops decreasing, all learning rates are divided by a scalar of ten. For better generalization, we use a mini-batch size of 10 and make a shuffle of all training images on every epoch.

The proposed algorithm is developed with the Torch7 package [16], which makes many efforts on improving the performance and efficacy of benchmark methods. The training procedure for all models listed in Table 1 can be finished within three days using an Intel Core-i7 3.4 GHz PC, and the CUDA version can be much shorter to less than 20 h with a Nvidia Titan X GPU. When testing, it takes 10ms for the onboard Nvidia Jetson TK1 to classify one image.

4.2 Quantitative Results and Discussion

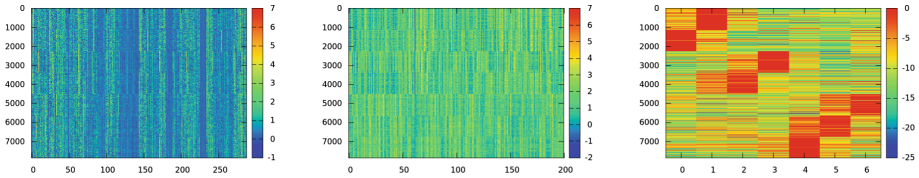
Firstly, for a general view of the accuracy, Table 2 lists out the performance of all those models with the class number K varies. It demonstrates that we could use our approach in practice with a high confidence of larger than 86%.

As demonstrated in Table 2, among all those cases, our model 2 with BN performs best. Further, as the number of directions increase, all those methods' precision drops slightly. That is reasonable because when we have more classes, the chance of overlapping increases and then makes this problem more complicated. Besides, since the videos we captured is not guaranteed to be perfectly straightforward viewing, there might be some miss labelled images. This, in turn, can also affect the final precision.

Then, Fig. 4 gives a detailed classification result when $K = 7$. Since images of different classes are similar to each other in contents and varying with a slight difference of rotational degrees, neighbouring classes are more likely to be misclassified. That's why we have a higher value near the diagonal positions in

Table 2. Average classification results on our SphericalNavi dataset.

	[9]	Our model 1	Our model 2
3 classes	91.14%	92.64%	**94.30%**
5 classes	83.01%	84.82%	**93.07%**
7 classes	73.12%	72.36%	**86.41%**



(a) FC1 Input (b) FC1 Output & FC2 Input (c) FC2 Output

Fig. 3. The feature responses of testing samples from different layers in model 2 (7-class case, best viewed in color). After the mapping of layer FC1(a2b) and FC2(b2c), samples are in a more discriminative subspace.

Fig. 4. The detailed classification accuracies of all the classes in our 7-direction case. The higher the diagonal values, the better classification performance.

the confusion matrix. It should also be noted that, when the robot is walking on a straight road, front-view images and rear-view images may be difficult to distinguish from each other. The same problem applies to left/right case. Consequently, there may be some samples that look like horizontally or vertically flipped. That's why we have a slightly high value in the upright corner.

Lastly, deep learning methods have made extraordinary success in many tasks mainly benefit from its great capability of mapping high dimensional data into discriminative feature spaces. Figure 3 gives an illustration of how features are mapped layer after layer to a compact subspace. Corresponding feature outputs of 8000 sample images from the last convolutional layer (Fig. 3(a)) and the fully connected layers (Fig. 3(b) and (c)) are shown together. As the dimension of features decreases from 288 to 7, those 8000 samples are getting warped into more compact groups and thus much easier for classifiers to make further predictions.

5 Conclusions

In this paper, we formulate the spherical camera based autonomous robot navigation task as an image classification problem. We then solve this problem with a CNN network trained on a set of specifically labeled images. This method

of using raw panoramic information, without any time-consuming calibration and warping or global 3D virtual world building processes, works pretty well on mobile platforms with low computation resources. Benefited from the powerful capability of convolutional networks in solving classification problems, we have achieved impressive performance on our campus navigation experiments.

Acknowledgements. This work is supported by the National Natural Science Foundation of China (No. 61672429, No. 61502364, No. 61272288, No. 61231016), ShenZhen Science and Technology Foundation (JCYJ20160229172932237), Northwestern Polytechnical University (NPU) New AoXiang Star (No. G2015KY0301), Fundamental Research Funds for the Central Universities (No. 3102015AX007), NPU New People and Direction (No. 13GH014604).

References

1. Klein, G., Murray, D.: Parallel tracking and mapping for small AR workspaces. In: 6th IEEE and ACM International Symposium on Mixed and Augmented Reality, pp. 225–234 (2007)
2. Caruso, D., Engel, J., Cremers, D.: Large-scale direct slam for omnidirectional cameras. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (2015)
3. Hillel, A.B., Lerner, R., Levi, D., Raz, G.: Recent progress in road and lane detection: a survey. Mach. Vis. Appl. **25**, 727–745 (2014)
4. Liang, X., Wang, H., Chen, W., Guo, D., Liu, T.: Adaptive image-based trajectory tracking control of wheeled mobile robots with an uncalibrated fixed camera. IEEE Trans. Control Syst. Technol. **23**, 2266–2282 (2015)
5. Lu, K., Li, J., An, X., He, H.: Vision sensor-based road detection for field robot navigation. Sensors **15**, 29594–29617 (2015)
6. Chang, C.K., Siagian, C., Itti, L.: Mobile robot vision navigation & localization using gist and saliency. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 4147–4154 (2010)
7. Pomerleau, D.A.: Efficient training of artificial neural networks for autonomous navigation. Neural Comput. **3**, 88–97 (1991)
8. Hadsell, R., Sermanet, P., Ben, J., Erkan, A., Scoffier, M., Kavukcuoglu, K., Muller, U., LeCun, Y.: Learning long-range vision for autonomous off-road driving. J. Field Robot. **26**, 120–144 (2009)
9. Giusti, A., Guzzi, J., Ciresan, D., He, F.L., Rodriguez, J.P., Fontana, F., Faessler, M., Forster, C., Schmidhuber, J., Di Caro, G., Scaramuzza, D., Gambardella, L.: A machine learning approach to visual perception of forest trails for mobile robots. IEEE Robot. Autom. Lett. **1**, 661–667 (2016)
10. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems, pp. 1097–1105 (2012)
11. Ran, L., Zhang, Y., Hua, G.: CANNET: context aware nonlocal convolutional networks for semantic image segmentation. In: IEEE International Conference on Image Processing (ICIP), pp. 4669–4673 (2015)
12. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Region-based convolutional networks for accurate object detection and segmentation. IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI) **38**, 142–158 (2016)

13. Nair, V., Hinton, G.E.: Rectified linear units improve restricted Boltzmann machines. In: Proceedings of the International Conference on Machine Learning (ICML), pp. 807–814 (2010)
14. He, K., Zhang, X., Ren, S., Sun, J.: Delving deep into rectifiers: surpassing human-level performance on imagenet classification. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1026–1034 (2015)
15. Ioffe, S., Szegedy, C.: Batch normalization: accelerating deep network training by reducing internal covariate shift. In: Proceedings of the International Conference on Machine Learning (ICML), pp. 448–456 (2015)
16. Collobert, R., Kavukcuoglu, K., Farabet, C.: Torch7: a matlab-like environment for machine learning. In: BigLearn, NIPS Workshop. Number EPFL-CONF-192376 (2011)