Camera-based Obstacle Classification for Automated Reach Trucks Using Deep Learning

Marian Himstedt, Institute of Computer Engineering, University of Lübeck, himstedt@iti.uni-luebeck.de, Germany
Erik Maehle, Institute of Computer Engineering, University of Lübeck, maehle@iti.uni-luebeck.de, Germany

Abstract

This paper focuses on the classification of obstacles that are widely present in warehouse environments using an RGBD camera. Our approach applies depth segmentation to detect obstacles which are classified using a Convolutional Neural Network and a Support Vector Machine. Our system is evaluated on real-world data captured from an automated reach truck in a warehouse environment.

1 Introduction

The detection of obstacles is a fundamental prerequisite for a mobile robot in order to safely navigate in its environment. Static obstacles are incorporated when planning a path from the current position to a given goal. Dynamic obstacles are taken into account by the local motion planning system in order to avoid collisions. While avoiding static obstacles such as walls, trees, racks is quite straightforward, incorporating dynamic obstacles in motion planning is more challenging. This is particularly the case if prior knowledge about an obstacle is not available which poses significant challenges when predicting its future positions and velocities. Many robotic systems are able to detect obstacles blocking or crossing a planned path. Depending on the size and available free space, the robot is forced to stop and wait or bypass the latter. The missing context knowledge about the kind of obstacle, however, limits the potential avoiding maneuvers and requires the robot to reduce its speed significantly. If, however, knowledge about the type of obstacle is available, a robot is able to carry out more intelligent and specifically tailored navigation. This paper introduces an obstacle classification system allowing to incorporate environment-specific knowledge. We therefore analyze our target environment for those objects being commonly expected as obstacles. In the case of warehouses we found the following object classes to be most relevant: *humans*, *palleted goods* and *forklifts*. The forklift class covers an extensive number of wheeled warehouse vehicles, ranging from lifting carts to large forklift trucks. The class *palleted goods* describes the majority of goods being stored and moved on pallets in warehouses (e.g. pallet cartons, stillages). We observed that these are the main classes of objects occurring as obstacles on driving paths in the target environment and being worth to be threatened individually. Of course the number of object classes that might appear in warehouses is likely larger than three. However, these are still detected as obstacles but not necessarily classified. Reducing the number of object classes helps minimizing confusions and usually increases the classification certainty particularly in the presence of visually similar objects.

Figure 1: An automated reach truck in a warehouse serving as test environment.

Even though we emphasize the application of our system for robot motion planning, the particular implementation and evaluation of the latter is not in the scope of this paper. The goal is to present and evaluate our approach for detecting and classifying obstacles in one specific type of environment while motivating its application for robotic navigation tasks. Given RGB images of captured obstacles, the goal is to extract features from the image serving as input for a multi-class classifier. A pretrained convolutional neural network (convnet) is used for feature extraction. The output of a higher layer of the convnet is used to obtain finegrained representations of individual objects. The high-dimensional feature vectors being generated for detected obstacles are passed to a multi-class support vector machine (SVM). The idea of combining convnet features and SVMs for object recognition is inspired by the work of Yosinski et al. [8] reporting surprising classification results. Razavian et al. investigate an extensive study about the portability of pre-learned convnet features for setting up visual recognition systems [6]. The convnet features have also drawn particular interest by the robotics community. Suenderhauf et al. re-

cently presented a large-scale place recognition system based on a-priori learned convnet landmarks [7].
Key contributions:

- Segmentation of depth images to detect obstacles
- Use of ConvNets for feature extraction
- Training data obtained solely from public sources
- System evaluation in a warehouse environment

The paper is structured as follows: Our system is described in Section 2. We present our experimental results in Section 3 before concluding the paper.

2 System overview

This section provides an overview of our obstacle classification system and revises the methodological background of our work. The pipeline of our approach is illustrated by Figure 2 and is divided into the following steps:

1. The input depth image is segmented to detect and separate objects
2. For each detected object a region of interest in the RGB image is determined
3. The sub image defined by the region of interest is passed to a ConvNet and searched for features in the RGB space
4. ConvNet output is evaluated by N SVMs with N being the number of expected classes (in our case three)

The steps mentioned above are described in depth in the following.

2.1 Obstacle Detection

The input RGBD data of a 3D camera is searched for obstacles. Thanks to the range data, the detection of occupied space in close proximity of the vehicle is simplified. Our goal is to identify objects being close to the vehicle which might potentially block the path to a goal. For this purpose a depth segmentation is applied to the depth image. This enables to separate back- and foreground as well as the ground plane of the captured scene given a maximum object range of r_m. The ground plane is estimated within the system calibration during a prior teach-in procedure. We therefore fit planes inside the point cloud computed based on the depth image and the calibration parameters. The RANSAC-based implementation for plane estimation of the C++ library PCL [5] is used for this step. The ground plane computed within the teach-in phase is kept fixed. We continuously check the validity of the ground plane to avoid miscalibrations due to sensor relocations. The ground plane is further used to transform the input point cloud relative to the vehicle coordinate system. Having removed the background and ground plane, the remaining measurements are considered for further object detection. First, the remaining point cloud is sub-sampled for performance reasons and subsequently used to build a voxel grid map. A kd-tree based on 2D voxel data in the xy-coordinate system (planar parallel to the ground) is built and serves as input for further processing. We apply an agglomerative clustering approach based on a L^2 norm to identify groups of points referring to an object. From each data point we search within a surrounding radius of $r_c = 0.5m$ for neighboring points. Once all data points are visited at least once, we stop clustering. Note that this approach also bases on existing implementations of PCL [5]. For each cluster we estimate the boundaries (min_x, max_x), (min_y, max_y), (min_z, max_z). These points are used to estimate 3D bounding boxes around objects and provide a prior for the classification since object dimensions are expected to be known a-priori. At this step we make all detected objects available for other modules on the robot irrespective of whether they can be classified at a later stage.

2.2 Regions of interest

In order to estimate the class of an object, we need to bridge from the depth data to the RGB image. From the preceding step we acquire a number of object priors with associated 3D bounding boxes. Given the camera calibration parameters we project the 3D boundaries of the object back in the 2D coordinate frame of the RGB image. The estimated 2D vertices of the object are used to define windows inside the RGB image. Since our system only classifies a limited number of objects, we filter the detections based on their physical dimensions. We consider the object's height (max_z of object) and the length of the largest object width being visible. Note that depending on the camera's pose w.r.t. to the object, there are 1-2 object faces visible. The object dimensions considered in our system are shown in Table 1.

Class	w_{min}	w_{max}	h_{min}	h_{max}
Forklift	1.0	4.0	1.0	3.0
Pallet	0.75	1.25	0.12	0.9
Human	0.2	0.7	1.4	2.1

Table 1: Allowed dimensions of objects to be considered. Values are in $metres$. Dimensions for pallets are based on euro-pallets. Meaning: e.g. w_{min} determines minimum width, h_{max} the maximum height.

As a result of this step we obtain a set of ROIs whose boundaries define regions of interests (ROIs). The latter serve as input for the subsequent layers.

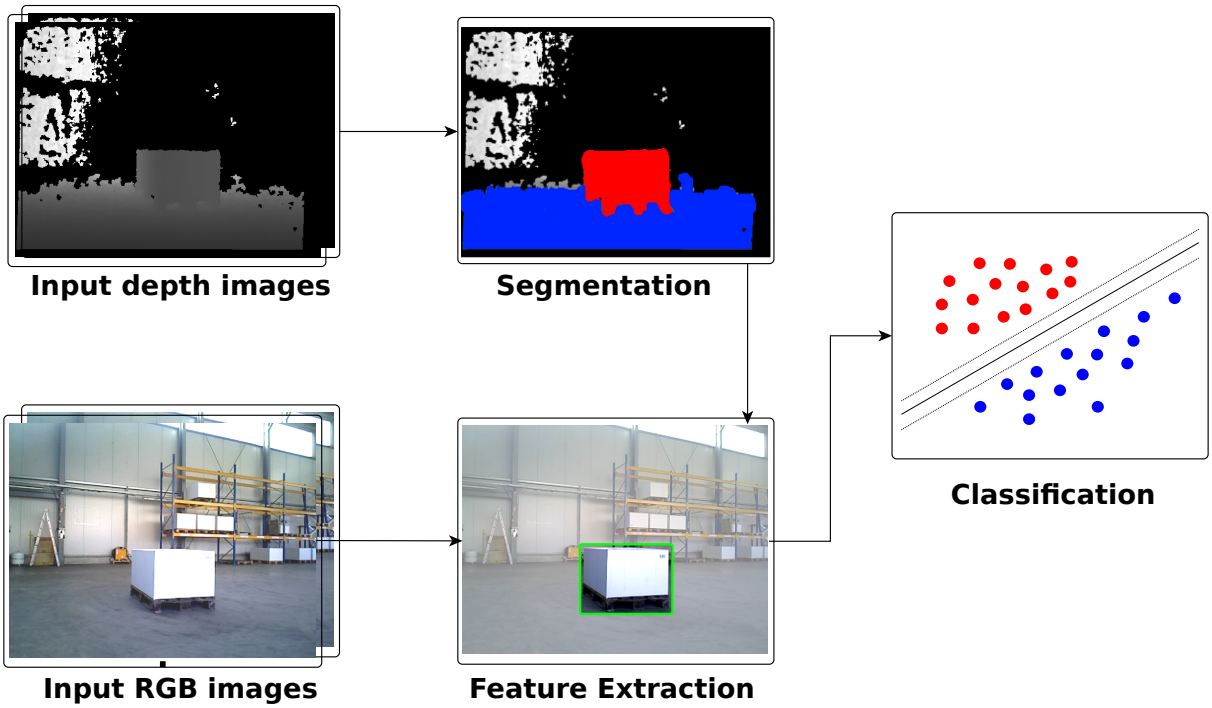

Figure 2: System overview. The input depth images are segmented. The ground plane itself and objects sticking out are determined. Regions of interest are generated around detected objects. ConvNet features are extracted from the RGB images inside the ROIs. The features are passed to a set of linear SVMs for classification.

2.3 Feature Extraction

This layer extracts features from the RGB image data. This process is restricted to the areas defined by the ROIs of the prior detection step. For this purpose we use a Convolutional Neural Network (convnet). The principle of convnets can be summarized as follows. A set of convolutional filters are repeatedly applied to the 2D image data. The filters' outputs are collected into non-overlapping grids. The next layer subsamples the input data by applying pooling methods such as taking the maximum or average of the grid. The combination of convolving and subsampling the input data is repeatedly carried out at successive network layers. This method allows to learn features at different scales and spatial positions in the image. The complex fully-connected layers of neural networks are typically found at the end of a convnet. The outputs of different convnet layers can be combined for the final output. The convnets differ significantly from other feature extraction methods used in computer vision since they learn features and their distributions at different levels (e.g. parts, objects, local characteristics) given the training data. Depending on the depth, the layers respond to different scales of an object. The further a layer is located from the input layer the more local will be the response and the smaller the affected area of a firing neuron. As a feature extractor we make use of the pre-trained convnet *CaffeNet* [4] which consists of 7 layers with our system utilizing the fc7-layer. Since the latter is located at the end of the network we obtain a 4096-dimensional feature vector capturing local image characteristics.

2.4 Obstacle Classification

The content of each detected object is classified based on the convnet features extracted within its ROI and the object's dimensions. We train a multi-class Support Vector Machine (SVM) following a one-versus-one schema, hence we train one binary SVM for each class i:

$$\mathbf{b_i} \cdot (\mathbf{w_i^T} \cdot \mathbf{x_i} + \mathbf{w_{i,0}}) \geq 1 \tag{1}$$

We use a linear kernel which can be defined as the following cost function:

$$\Psi(\mathbf{w_i}) = \frac{1}{2}\mathbf{w_i^T}\mathbf{w_i} \tag{2}$$

We observed that using a linear kernel provides promising classification results while keeping the computational costs at a minimum. This is necessary since a more complex system has to evaluate a large number of classifiers for each obstacle being detected at a high frequency.

3 Experiments

In this project we exemplarily trained an SVM using convnet features for the following object classes: forklift trucks (Forklift), humans (Human), palleted goods (Pallet). These classes are expected to be most common in

warehouse environments. In our first experiments we evaluated the contribution of training an additional class explicitly accounting for walls, large racks and clutter being expected in warehouses. However, we observed that adding this class rather introduced unintended classification uncertainty since it covers widely spread clusters in feature space due to their large variance in visual appearance. By not explicitly considering the background we are able to mitigate deteriorations of classification. Thanks to our prior segmentation and object filtering step (see Section 2.2) we already get rid of the clutter.

3.1 Datasets

Our recognition system is trained based on publicly available image data for the mentioned object classes. Specifically we use the Image-net database [3] for annotated images of forklifts and pallets. The training data for the class humans is obtained from the INRIA person dataset [2]. Since the number of training samples obtained from these sources is limited, we added further training images from the internet. This process was automated using the Microsoft Bing API [1]. All training images obtained in this way are manually inspected and partly cropped. In total we obtain a dataset consisting of 360 samples for each class which we divided into 240 training and 120 testing samples. Note that this data is solely originated from publicly available image sources. Our system was trained and tested given this data. An additional dataset captured in a typical warehouse environment was recorded in order to evaluate the generalization ability of our system. This validation dataset was captured by manually steering a reach truck equipped with an RGBD camera (see 1). We applied depth segmentation and feature extraction to the depth and RGB images as explained in Sections 2.1-2.3. The extracted ROIs for obstacle priors are passed to our classifier (see Section 2.4).

3.2 Results

Table 2 shows the confusion matrix obtained for the test dataset. It is obvious that the classes can be well distinguished from each other. The results obtained for the *Forklift* class are slightly worse than those for the other classes. This is probably due to the fact that this class captured a large variety of different wheeled vehicles typical for warehouses ranging from small automated lifting carts to large forklift trucks. The other classes rather vary in pose variance than actual visual appearance.

Forklift	Pallet	Human	...	Acc
108	2	10	Forklift	0.900
2	117	1	Pallet	0.975
2	0	118	Human	0.983

Table 2: Confusion matrix obtained for the testing dataset. *Acc* denotes the overall classification accuracy for the given class.

Table 3 shows the confusion matrix obtained for the validation dataset. The classification results are outstanding particularly if one considers that image data recorded with a Asus Xtion camera inside the author's test environment differs from the image data typically found on the internet. The latter is mostly recorded with cameras having larger sensors and hence providing images of higher quality.

Forklift	Pallet	Human	...	Acc
93	2	5	Forklift	0.930
2	98	0	Pallet	0.980
3	0	97	Human	0.970

Table 3: Confusion matrix obtained for validation dataset. *Acc* denotes the overall classification accuracy for the given class.

We observed that the convnet features obtained from the fc-7 layer provide a substantial benefit for distinguishing different classes. Experiments with adjacent layers showed comparable results whereas those extracted at lower layers performed worse. Our system relies on linear kernels for the SVMs which did not show any disadvantages in our experiments. Samples of our training/testing as well as validation datasets are illustrated by Figures 3 and 4 respectively.

3.3 System requirements

The experiments were carried out on a Dell E6320 laptop equipped with an Intel Core i7 (dual core CPU) and 8GB of RAM. Table 4 summarizes the mean run time of the entire system and the individual components.

Step	**time [ms]**
Obstacle detection	47.26
Feature extraction	124.82
Obstacle classification	15.61
total	**187.69**

Table 4: Run time for system components.

3.4 Discussion

We demonstrated that the presented approach can be integrated into robotic systems allowing obstacle detection and classification at a frame rate of more than 5 Hz. A large number of applications using Convnets require GPUs for performance reasons. Our experimental platform, an automated reach truck, is equipped with an electric engine and would, in fact, have sufficient power to serve a GPU. However, we dispensed using this architecture for generalization reasons since it is not always available on mobile robots. The detection does not have to run at full frame rate for our application. Once an obstacle is detected and classified, a tracker can be initialized

Figure 3: Examples of training dataset. The data contains images of various kinds of forklifts, palleted goods and humans. It is solely originated from publicly available sources.

to follow its movements. This is why it is not necessary to apply the ConvNet feature extraction to each detection ROI. We will exploit this in our future work.

4 Conclusions

This paper introduced an approach to obstacle classification using methods of deep learning. We motivated the benefit of incorporating environment-specific knowledge. We experimentally evaluated our approach on the example of warehouses by specifically learning common obstacle classes expected for this type of environment. Features were extracted from labelled images given a pretrained convolutional neural network and subsequently classified using Support Vector Machines. Our system was trained using publicly available image data. We tested our approach in a warehouse environment providing data the system has never seen before. This emphasizes our system's ability to generalize. We expect this to be highly beneficial for automated forklifts in warehouse environments providing a significant contribution towards intelligent robotic navigation.

5 Acknowledgement

This work has received funding from the Federal Ministry for Economic Affairs and Energy (BMWi) in the framework of the research project FTF out-of-the-box under grant agreement no. 01MA13005E.

References

[1] Microsoft bing search api. https://datamarket.azure.com/dataset/bing/search/. [Online; accessed 8-April-2016].

[2] N. Dalal. INRIA person dataset. http://pascal.inrialpes.fr/data/human/, 2005. [Online; accessed 14-September-2015].

[3] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

[4] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014.

[5] Radu Bogdan Rusu and Steve Cousins. 3D is here: Point Cloud Library (PCL). In *IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China, May 9-13 2011.

[6] Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. Cnn features off-the-shelf: An astounding baseline for recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2014.

[7] Niko Suenderhauf, Sareh Shirazi, Adam Jacobson, Feras Dayoub, Edward Pepperell, Ben Upcroft, and Michael Milford. Place recognition with convnet landmarks: Viewpoint-robust, condition-robust, training-free. In *Proceedings of Robotics: Science and Systems*, Rome, Italy, July 2015.

[8] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In Z. Ghahramani, M. Welling, C. Cortes, N.D. Lawrence, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27 (NIPS '14)*, pages 3320–3328. Curran Associates, Inc., 2014.

Figure 4: Results obtained on the validation dataset captured in a warehouse environment. Our system is able to detect and recognize objects of the classes: *forklifts*, *humans* and *palleted goods* under varying poses and illumination conditions.