The Thirty-Second AAAI Conference
on Artificial Intelligence (AAAI-18)

Learning an Image-Based Obstacle Detector with Automatic Acquisition of Training Data

Stefano Toniolo, Jérôme Guzzi, Luca Maria Gambardella, Alessandro Giusti
Dalle Molle Institute for Artificial Intelligence, USI-SUPSI, Lugano (Switzerland)
Email: stefano.toniolo@idsia.ch

Abstract

We detect and localize obstacles in front of a mobile robot by means of a deep neural network that maps images acquired from a forward-looking camera to the outputs of five proximity sensors. The robot autonomously acquires training data in multiple environments; once trained, the network can detect obstacles and their position also in unseen scenarios, and can be used on different robots, not equipped with proximity sensors. **We demonstrate both the training and deployment phases on a small modified Thymio robot**.

Overview

We propose a learning-based approach (Muller et al. 2006) for perceiving obstacles in the images acquired by a forward-facing camera mounted on a small robot. Other than detecting whether an obstacle is present, the system also estimates its position.

Instead of attempting to reconstruct the 3D structure of the environment in front of the robot, we follow a conceptually simpler and computationally lighter approach which considers each frame independently and does not rely on a sophisticated computer vision pipeline. As humans, when we observe a single picture, we can instinctively infer where an obstacle is present and which areas are free; this is because we have a prior expectation (learned from experience) on the appearance of free space and obstacles, not because we performed a multi-view 3D reconstruction of the scene.

In order to achieve a similar goal, our approach works by acquiring training datasets on a robot that is equipped with both a camera and a number of proximity sensors that can detect obstacles in the same area imaged by the camera, and thus produce a ground truth. For dataset generation, the robot autonomously explores a scenario unattended, driven by an ad-hoc controller that maximizes the amount of useful information acquired. The quality of the resulting classifiers and their generalization ability to different environments is quantitatively measured using a cross-validation scheme on multiple datasets acquired in disjoint scenarios.

Our demonstration uses a modified Thymio[1] robot, which is equipped with 5 forward-facing infrared proximity sensors and one forward-looking webcam (Fig. 2 Left); our approach however is general and would directly apply when using different robot platforms or sensing modalities.

At a given time t, the camera acquires an image $I(t)$ and each of the five proximity sensors either detects an obstacle or not; we represent the sensor outputs by means of a vector $y(t)$ containing Boolean values (1 if the sensor detected an obstacle, 0 otherwise). In our case, $y(t) = [y_{\text{LL}}, y_{\text{L}}, y_{\text{C}}, y_{\text{R}}, y_{\text{RR}}]$, representing the output of the leftmost, center-left, center, center-right, and rightmost proximity sensors, respectively.

Our goal is to gather training data, in the form of a set of pairs $(I(t), y(t))$ and then build a classifier that given $I(t)$ estimates $\hat{y}(t)$; $\hat{y}(t)$ will be a vector of real-valued numbers in the range $[0, 1]$, representing the probability that the corresponding element of $y(t)$ is 1.

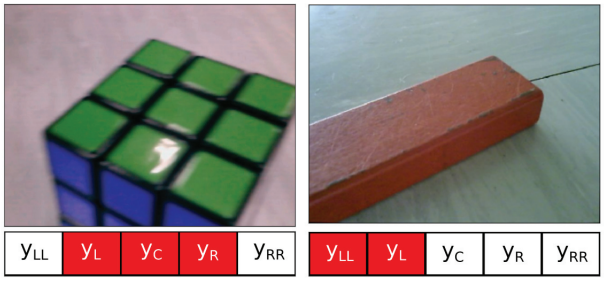

Figure 1: Images paired with corresponding proximity sensor data (groundtruth)

Automated acquisition of training datasets

Data gathering is automated by a smart controller which drives our robot inside an environment following three simple rules for obtaining a varied dataset. 1) The robot drives forward if no obstacles are detected by the sensors. 2) When an obstacle is detected by a sensor, the robot starts rotating in place towards the obstacle, so that the obstacle is seen at different positions in the camera field of view. 3) When the detected obstacle exits from the sensors' range, the robot continues rotating for a random angle between 0 and 180 degrees, which ensures a non-repetitive coverage of an unknown environment.

[1] https://www.thymio.org

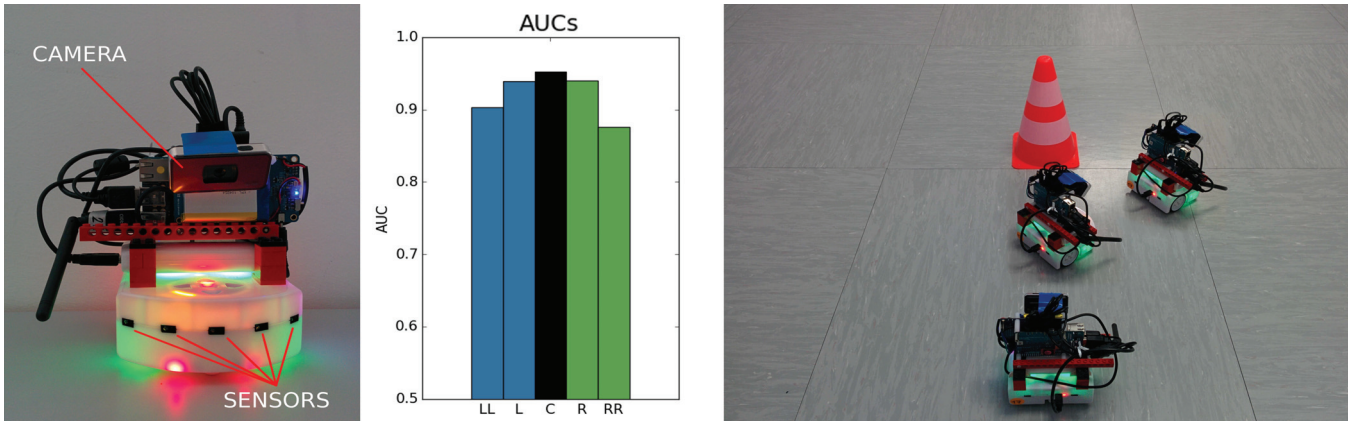

Figure 2: Left: custom Thymio used in this experiments, Center: performance of our neural network: AUC of binary predictions for each sensor (black - center, light blue - left, light green - right), Right: example of the robot's behavior

Experimental validation

Datasets, cross-validation and data augmentation

The dataset was built by autonomously exploring 8 different scenarios, each with a different floor, set of obstacles and lighting. The resulting dataset contains 53560 images sampled from 106 minutes of recording.

We implement a leave-one-scenario-out cross validation scheme: in each fold, 7 scenarios are used for training and the remaining one for testing.

During training, every image is resized to $60x60$ and augmented by: applying a random gain perturbation ($\pm 10\%$) on red, green and blue channels separately (to simulate different white balance); overlaying a random semitransparent gradient (to simulate smoothly-uneven lighting); with a 50% probability, mirroring around the vertical axis, which implies reversing the order of sensors in the ground truth.

Classifier

Our approach is based on a convolutional neural network with: a 60×60 RGB input; 3 pairs of 3×3 convolutional and 2×2 maxpooling layers, with 10 maps and ReLU activation; a 128-neuron densely-connected layer with ReLU activation; a 5-neuron densely-connected output layer with sigmoid activation, where each neuron corresponds to one sensor; note that this is not the same as a 5-class classifier, since we don't expect that one and only one output is active at a time (thus we don't implement a softmax layer); instead, the architecture corresponds to 5 binary classifiers which share everything except the last layer. Given an input frame, we obtain 5 values corresponding to the probability that an obstacle is in front of each sensor.

The net is built with the Keras framework (Franois Chollet 2017) using Tensorflow as backend.

Performance

The net is solving 5 binary classification problems in parallel: Fig 2 (center) reports the Area Under the ROC Curve computed for each sensor. We note that the central sensor and its neighbors are predicted more accurately than the peripheral sensors (LL and RR); this is expected as the area covered by such sensors is at the periphery of the camera field of view, and obstacles seen by the sensors may not be visible in the camera image.

Controller implementation

We implemented a reactive controller that processes the camera stream in realtime and uses the network outputs for control; in particular, the five outputs are converted to linear and angular speeds by means of simple rules similar to those used to generate obstacle-avoidance behaviors from proximity sensors: the resulting trajectories are visible in attached videos and Fig. 2 right.

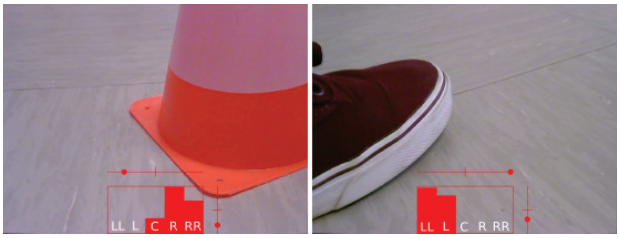

Figure 3: Camera view of the robot; outputs of the deployed net (red bars); resulting steering (horizontal slider) and linear speed (vertical slider).

References

Franois Chollet, o. 2017. Keras. https://github.com/fchollet/keras.

Muller, U.; Ben, J.; Cosatto, E.; Flepp, B.; and Cun, Y. L. 2006. Off-road obstacle avoidance through end-to-end learning. In *Advances in neural information processing systems*, 739–746.