Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnLine 52-8 (2019) 194–199

Intelligent Autonomous Robot Supporting Small Pets in Domestic Environment

Artur Chrzanowski [*] **Patryk Detko** [*] **Tomasz P. Stefański** [*]

[*] *Gdansk University of Technology, 11/12 Gabriela Narutowicza,*
80-233 Gdansk Poland (e-mail: arturchrzan@gmail.com,
detko.patryk@gmail.com, tomasz.stefanski@pg.edu.pl).

Abstract
In this contribution, we present preliminary results of the student project aimed at the development of an intelligent autonomous robot supporting small pets in a domestic environment. The main task of this robot is to protect a freely moving small pets against accidental stepping on them by home residents. For this purpose, we have developed the mobile robot which follows a pet and makes an alarm signal when a human is approaching. A pet is recognized in images with the use of a convolutional neural network. Walls and obstacles are detected with the use of ultrasonic sensors. A control system of the robot is implemented with the use of the Jetson TX2 platform. Preliminary tests of the robot demonstrate not only usefulness of our solution but also further directions for its development.

Keywords: Robots, Robot programming, Robot vision, Neural networks, Intelligent control.

1. INTRODUCTION

Although the topic of a human support by robots in a domestic environment has been investigated for many years (Fischinger et al., 2016; Antonello et al., 2017; Yuan and Li, 2017), the problem of animal support in these conditions is rather marginally treated. Therefore, we decided to investigate prospects for the development of a robot which protects a freely moving small pets in a domestic environment against accidental stepping on them by home residents.

First and foremost, we select adequate apparatus for this specific task. The robot is developed based on a small four-wheeled vehicle equipped with control electronics and sensors. In the developed robot, four ultrasonic sensors are employed to avoid collisions with environment elements as well as to detect when somebody is approaching a pet. Taking advantage of the GoogLeNet convolutional neural network (CNN) (Szegedy et al., 2015) running on Jetson TX2 platform (NVIDIA, 2017), vehicle control and detection of a pet is realized. The position of a pet and its relative distance to the robot are estimated from CNN output based on bounding boxes. In our tests, a guinea pig is employed as a small pet supported by the developed robot.

2. MECHANICAL DESIGN

The developed robot supporting small pets is presented in Figure 1. It is designed based on a small four-wheeled vehicle equipped with control units and sensors. Its body includes two plexiglass plates which provide mounting space for the following devices:

– bottom plate

- power supply (10 Li-ion batteries)
- four ultrasonic sensors
- Arduino Mega 2560 module (Atmel, 2014)
- motor drivers
- DC motors

– upper plate:

- Jetson TX2 board with camera
- LCD screen
- LED ring.

LED ring and LCD screen provide a feedback about actual state of the robot. The Jetson platform sends commands through a Bluetooth link to the Arduino module which controls steering, sensors and power supplies.

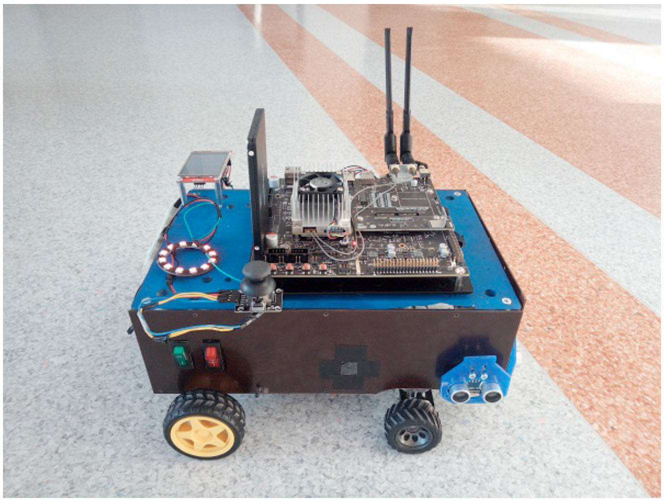

Figure 1. View of the developed robot.

10.1016/j.ifacol.2019.08.070

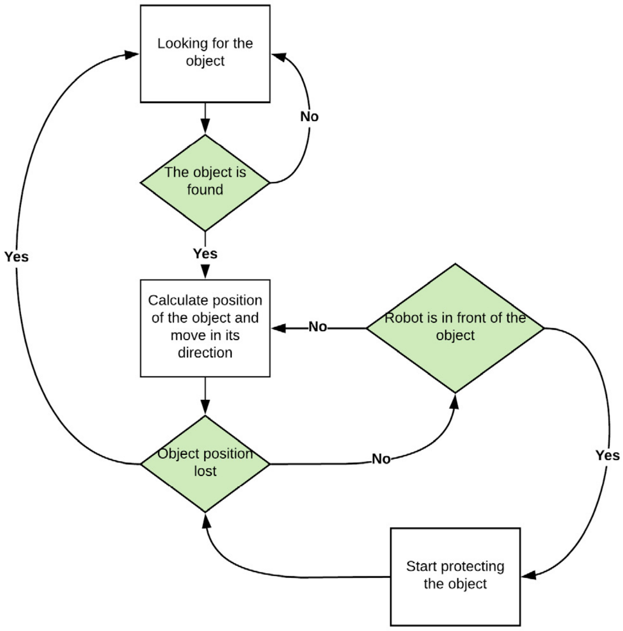
Looking for the object
No
The object is found
Yes
Yes
Calculate position of the object and move in its direction
No
Robot is in front of the object
Object position lost
No
Yes
Start protecting the object

Figure 2. Flowchart of the robot operation.

3. ROBOT OPERATION

Tasks of the robot can be divided into several groups that are presented in the following subsections. The flowchart of the robot operation is presented in Figure 2.

3.1 Searching for the pet

When the pet is out of sight for a few seconds, the robot starts to turn around until the recognized pet is found. When the pet is detected, the robot changes this operation mode into the next one.

3.2 Following the pet

Detection of the pet makes that the robot moves in its direction. The pet can change its position dynamically, thus, the robot can also change the direction of movement in response to the pet moves. For each processed frame of an image, which includes the detected pet, the relative distance between the pet and the robot is estimated. Then, it is used to determine if the pet is at a safe distance from the robot. When distance is less than defined threshold, the robot starts its next operation mode.

3.3 Alerting people

Without real-time control of the distance to the pet, the vehicle could collide with it. When the distance is sufficiently small, the robot is stopped. Then, the robot generates an alarm when someone approaches it. For this purpose, ultrasonic sensors are used. The robot not only draws an attention of people nearby the pet but also makes an alarm when ultrasonic sensors detect them. If the animal changes its distance to the robot, then the robot starts to search for the pet again.

Thanks to the ultrasonic sensors installed on the robot, it is able to detect obstacles in front and around it, refer to Figure 3. When any obstacle is detected, then the robot stops in front of it. The sensors used in the developed robot are not able to measure the distance to obstacles that are made of soft materials. Furthermore, some obstacles can reflect the wave in the direction that does not reach the receiver in the sensor.

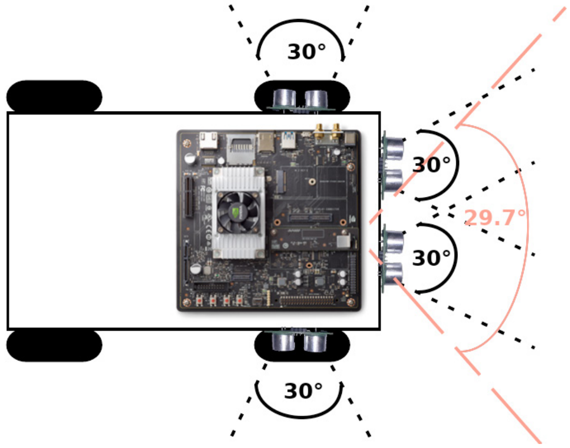

Figure 3. Perception of an environment by the robot (red lines - camera, black lines - ultrasonic sensors).

4. CONVOLUTIONAL NEURAL NETWORK

CNN is the most important element that enables an autonomous operation of the robot. In order to respond immediately to changes in an environment observed by the robot, the Jetson TX2 is employed as the processing and control unit ensuring high performance computing power with sufficient energy efficiency.

4.1 Selecting type of neural network

The Jetson TX2 board provides CUDA cores, which allow for the use of NVIDIA solutions related to the artificial intelligence and deep neural networks. Therefore, we use the project template provided by the manufacturer (NVIDIA, 2018) in the form of a training guide, which is based on the NVIDIA TensorRT platform. As a part of this solution, it is possible to use the following types of neural networks:

- ImageNet for image recognition
- DetectNet for object detection
- SegNet for image segmentation.

For our purposes, the DetectNet network is the most adequate solution. It allows one to load CNN model which, as a result of its operation, generates bounding boxes for detected objects.

4.2 Neural network training

The NVIDIA Deep Learning GPU Training System (DIGITS) is used to train CNN. As a part of this application, one can train neural networks based on such frameworks as:

- Tensorflow
- Caffe
- Torch.

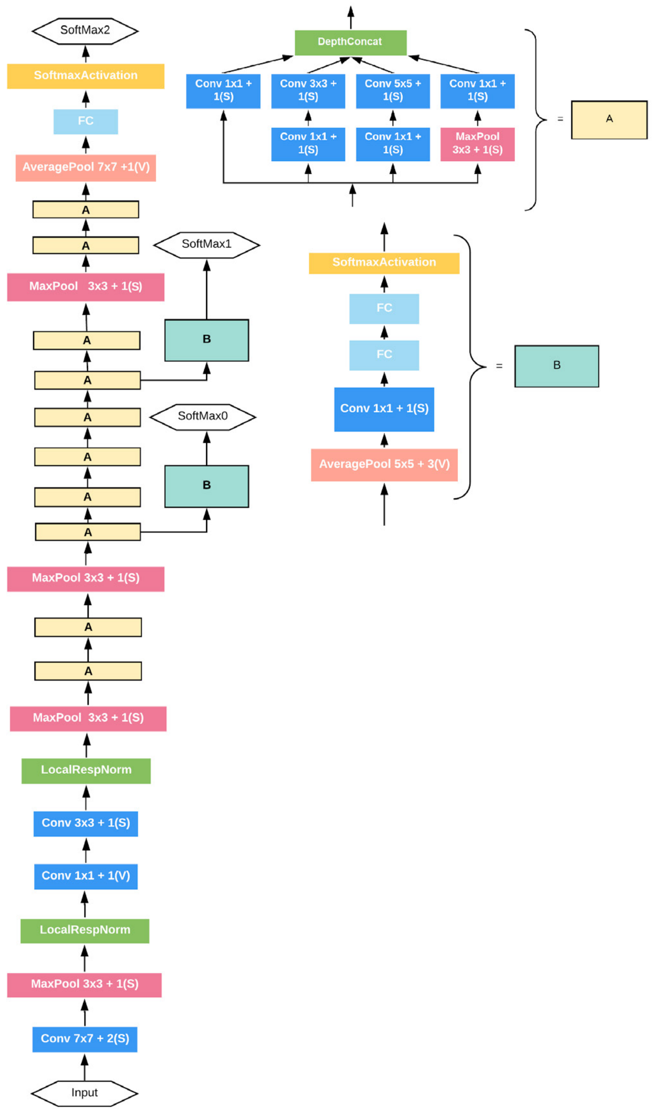

Figure 4. GoogLeNet architecture (Szegedy et al., 2015). Block A is the Inception module and Block B is secondary output layer. Description of color boxes: dark blue - convolution layers, dark red - MaxPool layers, light blue - fully connected layers, light red - AveragePool layers, yellow - output layers with Softmax activation function, green - additional normalization and depth concatenation layers.

Before creating any model, it is necessary to prepare a data set. For this purpose, a collection of 822 pictures of guinea pigs has been prepared using the ImageNet database (Stanford Vision Lab, 2019). The collected pictures are very diverse. They contain both pictures of several guinea pigs and individuals. The pictures have been taken in different environments, from different sides and also in different lighting conditions and zooms. With such a diversity of the data set, the network is learnt the appearance of the animal independent of the environment. For each picture in the prepared set, the coordinates of bounding boxes of guinea pigs are written down. Such a set of pictures and labels is used to create input data for DIGITS. Before the CNN learning process could be started, the data set is divided into a training set (574 pictures) and a cross-validation set (248 pictures).

Model training is executed on a host PC due to the higher computing power comparing with the Jetson board. For the reported research, the training is executed on the NVIDIA GeForce GTX970 graphics card, equipped with 4 GB of memory. Before start of the training, one needs to configure several training parameters (values in brackets are parameters we have taken to train our network):

- Number of training epochs (400)
- Solver type (Adam)
- Learning rate (2e-05)
- Policy of learning rate change and its gamma rate (Exponential decay, gamma = 0.99)
- Batch size (2)
- Batch accumulation (5).

Our trained CNN model is based on a non-standard network architecture in DIGITS, i.e., it is based on the architecture of the GoogLeNet network (Szegedy et al., 2015) and the Caffe framework (refer to Figure 4). The selected CNN architecture (also known as Inception v1) is characterized by several properties that make it so efficient. The main feature of this type of network is the Inception module, which consists of four parallel convolutional layers, including one with applied pooling. The applied convolution operations in this module uses a 5x5 and 3x3 kernels (filters). It requires a large number of computing operations, which would result in a long learning time. For this reason, before computations of this type, a convolution layer with a 1x1 kernel is used. This allows reducing the dimensionality of the input data of the layer, and thus speed up the computations. Another improvement that is applied to this architecture is the AveragePooling layer, which task is to average feature maps from sizes 7x7 to 1x1. Then, the result of this layer is passed to the standard fully connected layer. The last operation is to use the Softmax activation function on data coming from the fully connected layer. GoogLeNet has three output layers. Two of them are used on intermediate values that have not yet passed through all layers of the network. They are active only during the training process and their loss is added to the total loss with the weight equal to 0.3. It is aimed to help solve the gradient vanishing problem and ensure regularization (Nusrat and Jang, 2018), i.e., improve the application of the patterns learned by the model to objects that have not yet been seen.

In order to speed up the process of training new objects, we employ the weights of the pre-trained GoogLeNet network. After configuring all necessary parameters, the training process is started. The course and current effectiveness of the network being trained is visualized on the graph in the DIGITS application. In Figure 5, one can observe changes in individual parameters that determine the effectiveness of the network training. The meaning of each parameter is as follows (Barker and Prasanna, 2016):

- `loss_bbox` is the mean absolute difference between true and predicted corners of bounding box
- `loss_coverage` is defined as the sum of squares of differences between the true and predicted object coverage

- precision is the ratio of correctly detected objects to correctly detected objects plus detected objects which should not be detected
- recall is the ratio of correctly detected objects to correctly detected objects plus not detected objects which should be detected
- mAP is defined as the product of precision and recall parameters. It measures how sensitive the network is and how it avoids false detections.

After a few hours of training, one can observe the first increases in parameters: mAP, recall and precision, which means that the network learned to recognize guinea pigs. At the same time, it can be noted that other parameters decrease and oscillate around stable values. Hence, it can be concluded that differences between bounding boxes of detected objects and actual ones decrease.

Completed successfully neural network training allows one to download a trained model and run it in the DetectNet script in order to start detection of objects by the robot. Before running the code, it is possible to adjust the threshold value from which the object is treated as a guinea pig. Figure 6 presents exemplary case of the detection of the guinea pig by CNN. As seen, the pet is correctly detected but its size is overestimated due to human hands.

5. IMPLEMENTATION IN HARDWARE

Despite the high computing power and the wide range of applications that the Jetson board provides, we decided to use an additional electronic module that controls motors and sensors. The Arduino Mega board is used for this purpose, which provides not only a large number of input and output ports (both digital and analog) but it is also energy efficient. The presented solution enables division of the robot hardware into two main modules responsible for the object detection and data processing as well as the control over the vehicle.

5.1 Object detection and data processing module

The main task of this module is to analyse and detect the pet and estimate its position. Then, it should determine in which direction the robot should move. The Jetson board carries out this analysis alone. The direction of the robot movement is determined by bounding boxes, which are the result returned by CNN. For this purpose, the centroid coordinates of the obtained bounding box are computed. Using these coordinates, an image zone is determined in which the object is located, and thus how much the robot should turn, refer to Figure 7. Dividing an image into a larger number of zones increases the robot sensitivity to changes of the pet position. However, due to fluctuations of sizes of bounding boxes, this could cause that the robot rotates continuously in order to keep up with small changes in the position of the pet. The applied division of the image into five zones ensures a sufficient sensitivity in our case.

Rectangles surrounding detected objects are used to determine the distance to the animal. For this purpose, the area of each rectangle is computed and the distance is estimated based on its value. When a regularly shaped object is detected (i.e., object rotation does not change bounding box dimensions for the same distance), then an area of bounding box allows to estimate the distance unambiguously. Unfortunately, animals are rarely regular objects and this is not the case of guinea pig. Therefore, the rectangle surrounding detected object is assumed to be the bounding box of the animal facing the robot. Under this assumption, even when the animal stands sideways, the robot stops at a greater distance, but still at the safe distance.

The task of this module is also to send appropriate commands when the object disappears from the camera view. If the detection does not take place within a defined number of frames recorded by the camera, the vehicle is stopped. This is aimed to prevent the robot from following in a direction in which the pet has been seen for the last time. If the detection does not take place for some time after the robot stopping, the robot switches to the search mode.

5.2 Control over the vehicle module

The Arduino Mega 2560 module acquires and sets digital and analog signals for the following purposes:

- sending a command to perform the measurement and taking results from the ultrasonic sensors
- determining the direction of rotation of the rear and front motors
- determining the required power of motors
- reading the steering angle of wheels using a linear potentiometer
- writing information on the LCD screen
- displaying the current state of the robot using different colors on the LED ring.

The Arduino module is an intermediary in executing the control commands sent by Jetson. Any information received from Jetson is acknowledged. If the command releted to the wheel positioning is sent, an acknowledge of the end of the wheel positioning is sent back to Jetson. Then, the algorithm execution can be continued on Jetson. In addition to constantly listening to commands from Jetson, Arduino also deals with measuring distances from obstacles and displaying information on the LCD screen and LED ring.

6. ROBOT TESTS

The robot tests consist of a number of trials during which the robot is placed at the center of a large room or a corridor. The robot task is to find the pet whose initial location changes in every attempt. Then, the robot approaches the pet and stops in a safe distance to it. If someone is approaching a pet, then ultrasonic sensors should detect it and make an alarm sound. However, we do not test this part of the robot operation to avoid stressful situations for a guinea pig.

The tests show that the robot finds a guinea pig in most cases and reaches it, refer to Figure 8. However, lighting of the room or covering the pet by another object may affect the detection probability. If the exposure is too low, the image captured by the camera may be blurred, which makes impossible to detect the animal. Furthermore, when

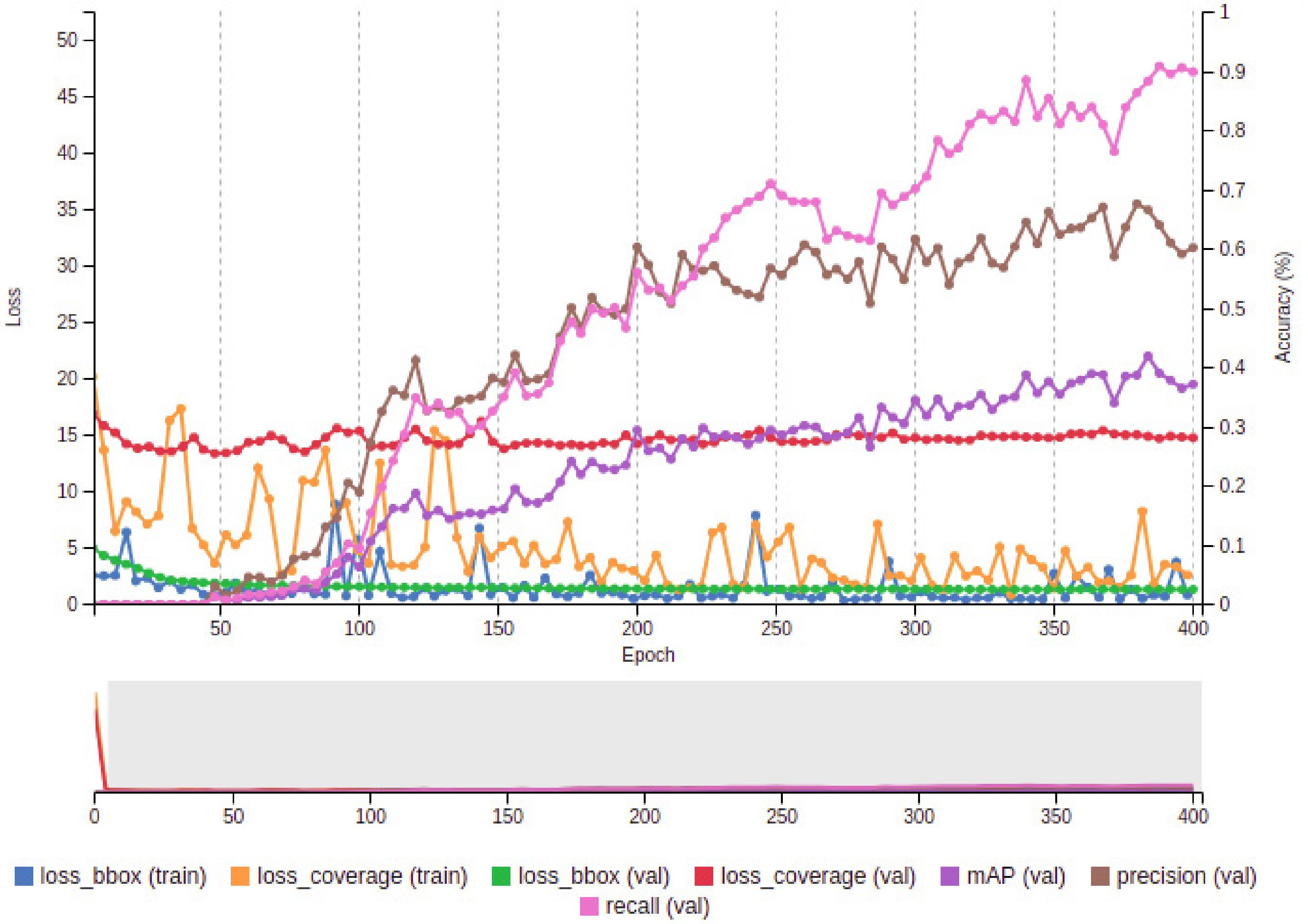

Figure 5. The course of training of the neural network detecting guinea pigs that is employed in the developed robot.

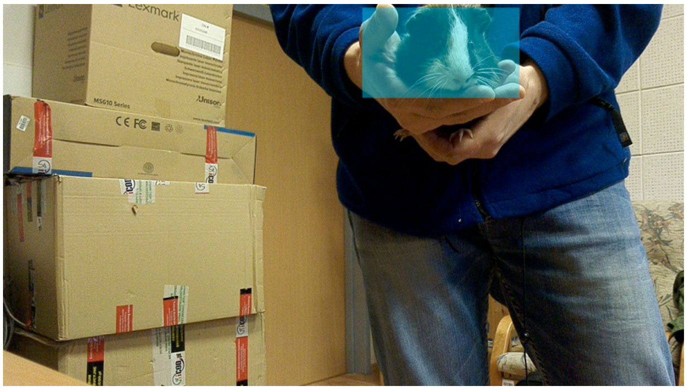

Figure 6. The detection of the guinea pig by CNN.

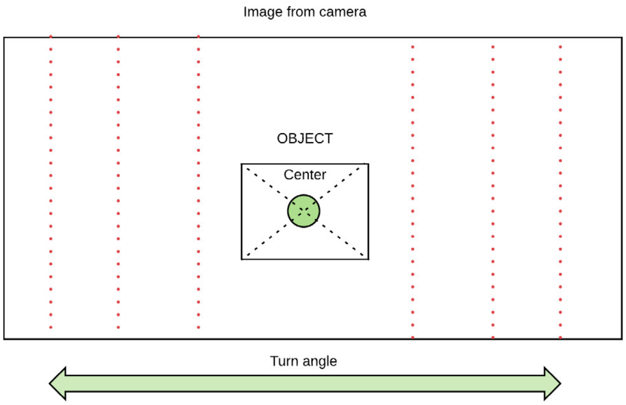

Figure 7. The camera image divided into zones.

the robot or the pet is illuminated by a strong-light source, detection may not be possible, refer to Figure 9. The robot may not be able to detect the pet in the case of a partial covering of the animal, but it depends on the degree of its covering. Due to the swivel front axle of the wheels, the robot is able to dynamically change its direction of movement, and thus smoothly react to changes of the animal position. However, if the direction changes are to fast then the robot moves along a curved path instead of a straight path. When the pet detection is lost, the robot starts rotations and the searching for a pet, refer to Figure 10.

The developed robot stops when the size of the detected object in an image is larger than the threshold. However, the distance estimation from the bounding box area can be inaccurate. It is due to the lack of the second camera, which might be used for depth estimation (Kowalczuk and Merta, 2016).

Another problem is related to the fact that animals can behave in unpredictable way. For instance, the pet can hide when it is left alone. In such a case, the robot is not able to protect the pet against accidental stepping on it.

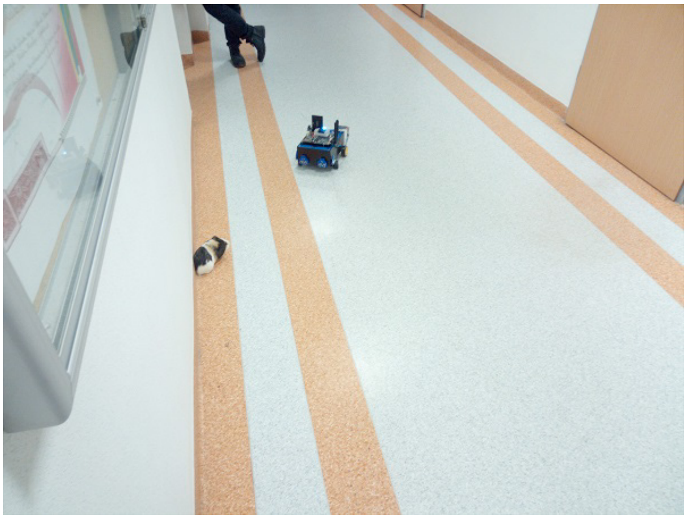

Figure 8. The robot detects the pet and moves towards it.

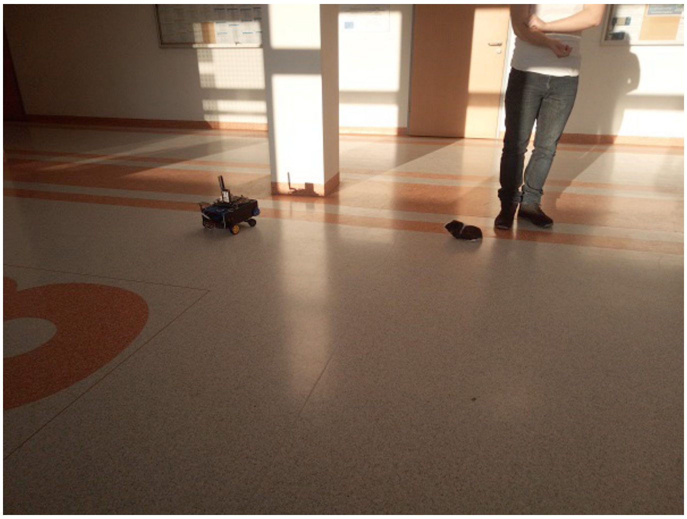

Figure 9. The robot cannot detect the pet due to a strong-light illumination.

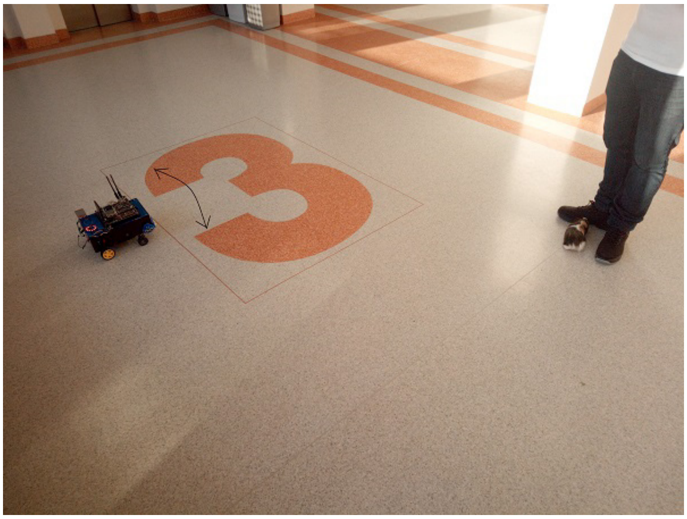

Figure 10. The robot does not detect the pet and starts turning around.

7. CONCLUSION

Our tests prove that the robot can be used to support pets in a domestic environment but its operation still requires improvements. The robot correctly detects a pet. However, the error of the distance estimation is large in our solution based on a single camera. Hence, for further development of the robot supporting a pet, better method of the distance estimation should be used, e.g., based on an additional camera. The developed robot is able to perform its tasks only in the case of a typically behaving pet.

The use of the neural network in the field-programmable gate array (FPGA) is the next stage of the robot development. It should allow us to reduce the size and the power consumption of the robot.

REFERENCES

Antonello, M., Carraro, M., Pierobon, M., and Menegatti, E. (2017). Fast and robust detection of fallen people from a mobile robot. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 4159–4166.

Atmel (2014). 8-bit atmel microcontroller with 16/32/64kb in-system programmable flash [datasheet]. In *Atmel ATmega640/V-1280/V-1281/V-2560/V-2561/V*. URL `http://ww1.microchip.com`.

Barker, J. and Prasanna, S. (2016). Deep learning for object detection with digits. URL `https://devblogs.nvidia.com`.

Fischinger, D., Einramhof, P., Papoutsakis, K., Wohlkinger, W., Mayer, P., Panek, P., Hofmann, S., Koertner, T., Weiss, A., Argyros, A., and Vincze, M. (2016). Hobbit, a care robot supporting independent living at home: First prototype and lessons learned. *Robotics and Autonomous Systems*, 75, 60 – 78.

Kowalczuk, Z. and Merta, T. (2016). Three-dimensional mapping for data collected using variable stereo baseline. In *2016 21st International Conference on Methods and Models in Automation and Robotics (MMAR)*, 1082–1087.

Nusrat, I. and Jang, S.B. (2018). A comparison of regularization techniques in deep neural networks. *Symmetry*, 10, 648.

NVIDIA (2017). User guide. In *Jetson TX2 Developer Kit*. URL `https://www.nvidia.com`.

NVIDIA (2018). In *Deploying Deep Learning*. URL `https://github.com/dusty-nv/jetson-inference`.

Stanford Vision Lab, Stanford University, P.U. (2019). In *ImageNet*. URL `http://image-net.org/index`.

Szegedy, C., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2015). Going deeper with convolutions. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1–9.

Yuan, W. and Li, Z. (2017). Development of a human-friendly robot for socially aware human-robot interaction. In *2017 2nd International Conference on Advanced Robotics and Mechatronics (ICARM)*, 76–81.