Response letter: Learning Long-range Perception using Self-Supervision from Short-Range Sensors and Odometry

Mirko Nava*, Jérôme Guzzi, R. Omar Chavez-Garcia,
Luca M. Gambardella and Alessandro Giusti

December 2018

We thank the editor and reviewers for the insightful comments. We respond to all these comments below and outline the consequent improvements in the paper.

We also remark that the approach described in this paper has been demonstrated to the public during CORL 2018 in Zurich with real robots, and that an improved version of the same demo will be shown at AAAI 2019 (as described in the accepted abstract [Nava et al., 2019, to appear]). In case this paper is accepted, it the AAAI demo will be a nice way to disseminate it.

1 Editor comments

> The authors propose a method of cross-modal sensor training. Overall, the paper has significant strengths and the problem is interesting.
>
> Reviewer 1 has serious issues with the evaluation.
> Reviewer 3 has some issues with the independence assumptions.
> Reviewer 6 has some issues with the training regime and use of the "unknown" data. I find merit in all of these issues. Also the claims of "self-supervision" should be clarified.
>
> All issues must be addressed in a potential revised version of this Paper.

We believe that this revision addresses all reviewer comments, as detailed in the responses below; the revision also presents a completely new

*All authors are with the Dalle Molle Institute for Artificial Intelligence (IDSIA), USI-SUPSI, Lugano, Switzerland. mirko@idsia.ch

simulation experiment as suggested by Reviewer 1, which is complementary to the previous real-robot experiments. This allows us to further test the limits of the current approach, and, in our opinion, significantly strengthens the paper contribution and makes our results easily replicable.

2 Reviewers comments

2.1 Reviewer 1

> This paper presents an innovative approach to generating supervisory signals for sensor perception tasks. The central idea is thus: we learn to interpret the outputs of a long-range sensor by learning a predictive model that estimates the outputs of short-range sensors.
> This stems from the hypothesis that short-range sensors will measure roughly the same 'events' as long-range sensors, albeit with a time lag. The authors demonstrate this concept using a camera feed to learn to predict the future observations of a set of proximity sensors.
>
> The paper is easy to follow, and the central concept is well-emphasized. However, I have several remarks to make.
>
> - It is commendable that the authors have taken the efforts to implement their approach on a real robot platform. However, they fall short on experimental analysis of the several properties of their algorithm. Although the authors claim that the approach is general and can work for a variety of target poses, the experiments are performed only for poses that are aligned with the longitudinal axes of the robot, i.e., for poses that lie on a straight line in front of the robot. The results are hence, in my opinion, not watertight.

Predicting the outputs of the short range sensors for target poses that are not on a straight line in front of the robot is indeed a very interesting task to test; in the revision, as per your suggestion, we describe a simulated experiment which reports additional, complementary results wrt. the previous real-robot experiments. In particular, the new experiments compare to the previous ones as follows.

Dimension	Real-robot experiments	Simulated Experiments
Environment	10 different real-world scenarios	10 different procedurally-generated simulated worlds
Target poses	30, aligned with the robot's longitudinal axis	$17 \times 17 = 289$, on a grid centered on the robot
Orientation of target pose is relevant	Yes	No
Short-range sensors	Five (proximity sensors)	One (floor color sensor)
Long-range sensors	One non-calibrated camera	Three non-calibrated cameras mounted at random unknown angles
Image appearance	Challenging, real-world	Simple
Controller for data acquisition	Ad-hoc	Random-walk

> - The nature of experiments carried out by the authors seems to be handicapped by the fact that they had to acquire datasets out in the real-world. From a scientific perspective, however, I strongly believe the authors could use a simulation environment to perform an extensive analysis of the approach, see where it breaks, and subsequently harden the algorithm. A thorough analysis in a simulator would greatly strengthen the paper, and be a valuable reference to the self-supervised learning community.

Using real-world data indeed limited our ability to explore some facets of our approach; the new simulation experiments allowed us to test the approach on unexplored aspects such as: heavy unbalance about the amount of data available for different poses; having a large majority of the poses unobserved for each training instance (which is only possible thanks to our masked loss approach); prediction of poses for which the long-range-sensors never observe any data; prediction of poses that are visited in the past instead than in the future. We will publish the entire simulation setup (in addition to the real-robot setup) in the paper's github repository[1], so that our experiments can be effortlessly replicated and new experiments can be designed and run easily.

[1] `https://github.com/idsia-robotics/learning-long-range-perception`

> - The current approach falls into a band of approaches that attempt to utilize data from auxiliary sensors to learn task-oriented percepts. Other approaches that lie within the band, but were not dealt with in the paper are "End-to-end learning for self-driving cars" by Bojarski et al. and "A Machine Learning Approach to Visual Perception of Forest Trails for Mobile Robots" by Giusti et al. The underlying principle in the proposed approach is similar at the core, and I suspect would suffer from the same drawbacks that these methods to: out-of-distribution data would result in cascading failures. This is another area where experiments in a simulation environment could make a strong case for the proposed algorithm.

We know both cited papers well (our group authored the second). We can see the link between our present work and these approaches exploring end-to-end control; however, we believe the similarity is not particularly strong, especially in comparison to the other literature we cite. In particular, we are not dealing with end-to-end control nor with imitation learning. Yet, it is true that out-of-distribution data will cause the approach to fail, as with most other ML applications; however, one important feature of our approach (and self-supervised approaches in general) is that they can adapt to shifting environments by continuously acquiring training data and the respective ground truth; we remark this aspect in the revised version of the paper.

> - The lack of a baseline renders the results difficult to comprehend. The only approach that seems to be evaluated is from a neural network. Akin to [3], [9], a few 'weak classifiers' (such as the dataset mean for example) could be used as baselines. That would clarify if at all a powerful model such as the CNN they use is needed at all.

In the paper we consider only binary classification problems and we evaluate the quality of our predictions via the Area Under the ROC Curve metric; one of the advantages of such metric, which is widely used in the ML literature for binary classification problems, is that the baseline classifier which returns the mean over the dataset always has a score of 0.5. Hence, the lower bound in all our AUC plots is 0.5, and our heat-maps map 0.5 to black; anything higher than 0.5 means that our classifier is better than the dataset-mean baseline. We remark this explicitly in the revised text. It is true that a comparison with simpler image classification methods would be interesting, but our main contribution is not the CNN itself or the ability to interpret visually difficult images (as also exemplified by the new simulated experiments, for which we purposefully use simple scenarios). On the contrary, we use the CNN without any specific optimization, as a simple "black box" tool that gets the job done.

2.2 Reviewer 3

> The paper proposed a deep learning approach for self-supervised learning of predicting future short-range sensor outputs (e.g., infra-red proximity sensors) from current long-range sensor readings (e.g., front-facing camera). Training data is collected with an ad-hoc controller, from which the labels for a set of target poses (15) for each pose are populated with the information from the closest pose within the dataset. A cnn is then trained to fit the dataset. Various results are presented to validate the proposed method. The paper is interesting, and as pointed out in the paper, has many use cases. However, there are some problems worth considering. As shown in Fig.6, the dataset contains a very unbalanced collection of positive and negative samples, which can largely affect the performance of the training of deep networks; while the authors do not seem to make efforts to make up for this.

It is true that, depending on the task, training datasets could be very unbalanced at least for some labels: in particular, the datasets on which we operate in the real-world experiments are heavily unbalanced. A common solution could be oversampling the minority class (a "data-level" method, following Buda et al. [2017]); this is not feasible in our approach because we have a multi-label problem with multiple independent binary labels: oversampling the minority class of one label might over-sample the majority class of another. However, we can easily apply "classifier-level" methods such as:

post-scaling at test time, dividing the network output for each label for each class by its estimated prior probability (computed from the training set),

cost-sensitive loss at training time, weighting the loss from miss-classifying minority labels by a factor equal to the inverse of their frequency.

We discuss the issue in the paper but, to keep the implementation focused on our main contribution, we don't include these techniques in the experiments; we observed that they do not improve the performance of our classifiers as measured by the AUC (which is unaffected by class imbalance); moreover, we see value in allowing the prior probabilities for each class to be reflected in the network outputs.

> Also, the design of the output of the network might not be optimal, as the 15 outputs are actually highly correlated (that the label for the near pose has much information of the label of a farther point), while the proposed method treat them as Independent.

It is true that our method does not explicitly model the correlations between different labels; that could be done with a variety of methods [Read et al., 2011, Zhang and Zhou, 2014, Ruder, 2017]. However, note that since we train a single model to predict all labels simultaneously, we benefit from

parameter sharing between the different tasks. Then, it is expected that the multi-label model will perform better on a given label than the single-label model trained only on that label; we attempted an experiment to verify this but observed similar performance; probably a more complex task is needed to highlight this property.

> And as a side note, there exist very advanced deep learning approaches that are capable of predicting the entire depth image from the RGB image, from which much more sophisticated planning algorithms can be applied, compared to which, the planning approach that can be applied on top of the proposed approach might seem quite limited.

We agree that the main contribution of our paper is not specific to monocular obstacle detection, which is probably better solved with specialized techniques. The new experiment with simulated data should better highlight that our main contribution is methodological.

> As a conclusion, the proposed method is simple and could have many potential use cases (with possibly limited task complexity). There are points that could be improved, but the various results presented validate the proposed method.

2.3 Reviewer 6

> This paper proposes a method for annotating a dataset of long-range sensory readings by combing future short-range readings with odometry and using these annotations to train a small robot to perform collision avoidance.
>
> The paper, in general, is well-written with the exception of several minor typos. The authors clearly define the problem and provide example extensions of their solution directly in the beginning of the paper to provide more motivation. The key contribution is in how the author prepare and pre-process their dataset before training. Specifically, this involves the method for generating labels to train their supervised learning model on.
>
> The main concern I have with the learning method is relating to how "unknown" data is handled:
>
> - In Sec. 3B the authors define labels to "unknown" if the distance between $p(t')$ and p_j are outside of some threshold δ.
> - However, Sec. 3E states that these "unknown" labels are handled in training by using a binary mask to either mask or accept the error signal. In other words, if the model is fed an "unknown" sample, their method will multiply the error with a binary mask of zeros (no matter what the

> error is), which forces these errors to *not* be back-propagated.
> - The authors motivate this behavior by claiming that the errors should not be back-propagated since the labels are unknown, which I agree with. However, if the errors are not back-propagated, this means that the network never learns anything from seeing these examples in the first place. If this is true, the authors are completely ignoring the majority of their dataset and incorrectly interpreting how their model is being trained.

We understand the issue the reviewer raises, which is due to a misunderstanding of our "masked loss" approach; we improved the description in the revised version. In particular, note that if a given instance has at least one known label, then there *will* be an error signal during back-propagation; this error signal will depend only on the subset of labels that are known for that given instance. The new description explains the mask, which is computed separately for each instance, as follows:

> Each value in the mask is equal to 1 if the corresponding label is known, and equal to 0 if the label is unknown. During the forward propagation step, this mask is multiplied element-wise with the difference between the prediction and the ground truth for each output, i.e. the error signal. This nullifies the error where the mask is 0 (i.e. for the subset of labels which are unknown for the given instance), and lets it propagate where the mask is 1.

The approach is used among others by Eigen et al. [2014]. The fact the the approach works properly should also be apparent in the new simulated experiment, where for each given instance most labels are unknown (and some labels are even unknown for all instances!).

> Additionally, I was concerned with the author's general claim that their method was performing "self-supervised" learning. I would disagree with this claim and argue that their method only presents a way of automatically labeling their dataset, which in turn is used in an entirely *supervised* pipeline. Even claiming that the labels are automatically generated is a stretch here, since it relies on the fact that accurate odometry information is available. A good example of self-supervised learning would be a monocular depth estimation model which learns by predicting the left camera from the right camera and enforcing depth to be learned inherently by the model (eg. Godard et al 2018). The model is not being supervised to predict depth, but outputs depth as a side product of training. In this paper, the authors directly supervise to predict the labels in the data, there is no self-supervision in the learning process, only to generate labels in the pre-processing phase before any learning occurs.

Indeed a very recent line of research in the deep learning field uses "self-supervised learning" with the meaning intended by the reviewer (the paper

by Godard et al. [2018] is an example, and a nice high-level overview of the research line is available as a slide deck [Zisserman, 2018].

In our paper, we use "self-supervised" in the sense that most frequently appears in the robotics literature. In particular, the term refers to the absence of an external supervisory signal (no human labeling of data) when you consider the whole robot+controller as an entity (the robot's autonomous interactions with the environment generated supervisory data). Citing van Hecke et al. [2016, 2017]:

> Self-supervised learning is a reliable learning mechanism in which a robot uses an original, trusted sensor cue for training to recognize an additional, complementary sensor cue.

(in our case, the "trusted sensor" cue is the short-range sensor reading from a future (or past) time-stamp). Many other works in the robotics literature use the "self-supervised" term with this meaning: Ridge et al. [2015], Sofman et al. [2006], Brooks and Iagnemma [2012], and several high-impact papers from Sebastian Thrun's group in 2005-2012 [Dahlkamp et al., 2006, Stavens and Thrun, 2006, Lieb et al., 2005, Lookingbill et al., 2006].

Therefore, we think that the term "self-supervised" in our title is justified and best characterizes our work within the robotics community; however, at the beginning of the revised related work section we explicitly address the distinction with the line of work mentioned by the reviewer.

We explicitly state in the title and introduction that we rely on an odometry source; odometry is not required to be accurate, as shown by our experiments with a real robot whose odometry is very roughly estimated (not even from encoders, but from the motors' measured current draw).

> The paper addresses a key challenge in robotics: learning robotic control from large scale datasets without the need for manually annotating said datasets. Additionally, the authors evaluate their algorithm on a real-world system. However, I would recommend the paper to undergo the revisions to clarify the points above before acceptance.

References

Christopher A Brooks and Karl Iagnemma. Self-supervised terrain classification for planetary surface exploration rovers. *Journal of Field Robotics*, 29(3):445–468, 2012.

Mateusz Buda, Atsuto Maki, and Maciej A. Mazurowski. A systematic study of the class imbalance problem in convolutional neural networks. *CoRR*, abs/1710.05381, 2017. URL `http://arxiv.org/abs/1710.05381`.

Hendrik Dahlkamp, Adrian Kaehler, David Stavens, Sebastian Thrun, and Gary R. Bradski. Self-supervised monocular road detection in desert terrain. In *Robotics: Science and Systems*, 2006.

David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. In *Advances in neural information processing systems*, pages 2366–2374, 2014.

Clément Godard, Oisin Mac Aodha, and Gabriel J. Brostow. Digging into self-supervised monocular depth estimation. *CoRR*, abs/1806.01260, 2018. URL `http://arxiv.org/abs/1806.01260`.

David Lieb, Andrew Lookingbill, and Sebastian Thrun. Adaptive road following using self-supervised learning and reverse optical flow. In *Robotics: Science and Systems*, 2005.

Andrew Lookingbill, John Rogers, David Lieb, J. Curry, and Sebastian Thrun. Reverse optical flow for self-supervised adaptive autonomous robot navigation. *International Journal of Computer Vision*, 74:287–302, 2006.

Mirko Nava, Jerome Guzzi, Ricardo Omar Chavez-Garcia, Luca Maria Gambardella, and Alessandro Giusti. Demo: Learning to perceive long-range obstacles using self-supervision from short-range sensors. In *AAAI-19, Demo session*, 2019, to appear.

Jesse Read, Bernhard Pfahringer, Geoff Holmes, and Eibe Frank. Classifier chains for multi-label classification. *Machine learning*, 85(3):333, 2011.

Barry Ridge, Aleš Leonardis, Aleš Ude, Miha Deniša, and Danijel Skočaj. Self-supervised online learning of basic object push affordances. *International Journal of Advanced Robotic Systems*, 12(3):24, 2015. doi: 10.5772/59654.

Sebastian Ruder. An overview of multi-task learning in deep neural networks. *CoRR*, abs/1706.05098, 2017. URL `http://arxiv.org/abs/1706.05098`.

Boris Sofman, Ellie Lin Ratliff, J. Andrew (Drew) Bagnell, Nicolas Vandapel, and Anthony (Tony) Stentz. Improving robot navigation through self-supervised online learning. In *Proceedings of Robotics: Science and Systems*, August 2006.

David Stavens and Sebastian Thrun. A self-supervised terrain roughness estimator for off-road autonomous driving. In *Proceedings of the Twenty-Second Conference on Uncertainty in Artificial Intelligence*, pages 469–476. AUAI Press, 2006.

Kevin van Hecke, Guido de Croon, Laurens van der Maaten, Daniel Hennes, and Dario Izzo. Persistent self-supervised learning principle: from stereo to monocular vision for obstacle avoidance. *CoRR*, abs/1603.08047, 2016. URL `http://arxiv.org/abs/1603.08047`.

Kevin van Hecke, Guido C.H.E. de Croon, Daniel Hennes, Timothy P. Setterfield, Alvar Saenz-Otero, and Dario Izzo. Self-supervised learning as an enabling technology for future space exploration robots: Iss experiments on monocular distance learning. *Acta Astronautica*, 140:1 – 9, 2017.

Min-Ling Zhang and Zhi-Hua Zhou. A review on multi-label learning algorithms. *IEEE transactions on knowledge and data engineering*, 26(8): 1819–1837, 2014.

Andrew Zisserman. Self-supervised learning. `https://project.inria.fr/paiss/files/2018/07/zisserman-self-supervised.pdf`, 2018.

Learning Long-range Perception using Self-Supervision from Short-Range Sensors and Odometry

Mirko Nava, Jérôme Guzzi, R. Omar Chavez-Garcia, Luca M. Gambardella and Alessandro Giusti

***Abstract*— We introduce a general self-supervised approach to predict the future outputs of a short-range sensor (such as a proximity sensor) given the current outputs of a long-range sensor (such as a camera); we assume that the former is directly related to some piece of information to be perceived (such as the presence of an obstacle in a given position), whereas the latter is information-rich but hard to interpret directly. We instantiate and implement the approach on a small mobile robot to detect obstacles at various distances using the video stream of the robot's forward-pointing camera, by training a convolutional neural network on automatically-acquired datasets. We quantitatively evaluate the quality of the predictions on unseen scenarios, qualitatively evaluate robustness to different operating conditions, and demonstrate usage as the sole input of an obstacle-avoidance controller. We additionally instantiate the approach on a different simulated scenario with complementary characteristics, to exemplify the generality of our contribution..**

Videos, Datasets, and Code

Videos, datasets, and code to reproduce our results are available at: `https://github.com/idsia-robotics/learning-long-range-perception/`.

I. Introduction

We consider a mobile robot capable of odometry and equipped with at least two sensors: a long-range one, such as a camera or laser scanner; and a short-range sensor such as a proximity sensor or a contact sensor (bumper). We then consider a specific perception task, such as detecting obstacles while roaming the environment. Regardless on the specific choice of the task and sensors, it is often the case that the long-range sensors produce a large amount of data, whose interpretation for the task at hand is complex; conversely, the short-range sensor readings directly solve the task, but with limited range. For example, detecting obstacles in the video stream of a forward-pointing camera is difficult but potentially allows us to detect them while they are still far; solving the same task with a proximity sensor or bumper is straightforward as the sensor directly reports the presence of an obstacle, but only works at very close range.

In this paper we propose a novel technique for solving a perception task by learning to interpret the long-range sensor data; in particular, we adopt a self-supervised learning approach in which future outputs from the short-range sensor are used as a supervisory signal. We develop the complete pipeline for an obstacle-detection task using camera frames as the long-range sensor and proximity sensor readings as the short-range sensor (see Figure 1). In this context, the camera frame acquired at time t (input) is associated to proximity sensor readings obtained at a different time $t' \neq t$ (labels); for example, if the robot's odometry detects it has advanced straight for 10 cm between t and t', the proximity sensor outputs at t' correspond to the presence of obstacles 10 cm in front of the pose of the robot at t. These outputs at time t' can be associated to the camera frame acquired at time t as a label expressing the presence of an obstacle 10 cm ahead. The same reasoning can be applied to other distances, so that we define a multi-label classification problem with a single camera frame as input, and multiple binary labels expressing the presence of obstacles at different distances.

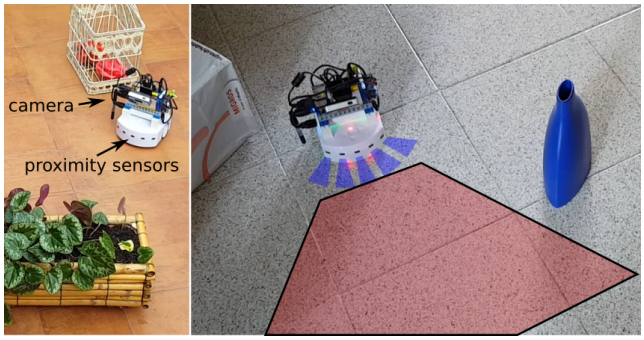

Fig. 1: The Mighty Thymio robot in two environments; five proximity sensors can easily detect obstacles at very close range (blue areas), whereas the camera has a much longer range (red area) but its outputs are hard to interpret.

The approach is *self-supervised* because it does not require any explicit effort for dataset acquisition or labeling: the robot acquires labeled datasets unattended and can gather additional labeled data during its normal operation. Long-range sensors do not need to be calibrated: in fact, they could even be mounted at random, unknown poses on the robot. Exploiting a combination of long-range sensors is handled naturally by just using all of them as inputs to the learning model.

Potential instances of the approach include: a vacuuming robot that learns to detect dirty areas, by using a camera as the long-range sensor and an optical detector of dust in the vacuum intake as the short-range sensor; an outdoor rover learning to see challenging terrain by relating camera and/or LIDAR readings to attitude and wheel slippage sensors; a quadrotor learning to detect windows at a distance, using camera/LIDAR as long range sensors and a vision-based

All authors are with the Dalle Molle Institute for Artificial Intelligence (IDSIA), USI-SUPSI, Lugano, Switzerland. This work is supported by the Swiss National Science Foundation (SNSF) through the NCCR Robotics.

door/window detector which works only at close range as the short-range sensor. Note that in this case, the short-range sensor is not a physical one but is the output of an algorithm that operates on camera data but is unable to produce long-range results.

The main contribution of this paper is a novel, general approach for self-supervised learning of long-range perception (Section III). In Section IV we implement this model on the Mighty Thymio robot [1], [2] for obstacle detection using a forward-looking camera as the long-range sensor and five forward-looking proximity sensors as the short-range sensor. In Section V we report extensive experimental results on this task, and quantitatively evaluate the quality of predictions as a function of distance. To test the generality of the approach, we finally instantiate the it on a different task and report results obtained in simulation.

II. Related Work

In robotics, self-supervised learning consists in the automated acquisition of training data [3], [4], [5], [6], [7], [8], usually by exploiting multiple sensors during the robot's operation; the technique has been used for ground robot navigation, most often for interpreting data from forward-looking cameras to detect obstacles or traversable regions. The term *self-supervised*, in this context, refers to the absence of an external supervisory signal (i.e., no human labeling of data), as the robots autonomous interactions with the environment generate supervisory information. The same term has been also used in the last few years to denote a related but much broader line of research [9] applied to various tasks within the field of deep learning, which aims to use the data itself as a supervisory signal (sometimes but not always [10] using data from different sensors).

In this paper we consider the meaning of the term specific to robotics, where self-supervision indicates that a robot autonomously acquires ground truth labels. Then, one can categorize the different approaches by the strategy adopted to compute such labels. Hadsell et al. [3] derive the supervisory signal from a point cloud obtained from a stereo vision sensor; this point cloud is processed with heuristics in order to generate a segmentation of the observed area to 5 classes (super-ground, ground, footline, obstacle, super-obstacle), which is then used as ground truth for learning the image appearance of each class. Zhou et al. [11] propose a similar approach using a lidar point cloud registered to image pixels; from the former, "terrain" and "obstacle" classes are extracted and used as labels for learning to classify image patches. Dahlkamp et al. [12] uses data from a line laser scanning the area just in front of the car: information about local height differences are used to segment such area – which maps to a trapezium in the camera view – into terrain and obstacle classes; the segmentation related to image pixels to learn a visual model of the road. Maier et al. [13] use a 2d laser sensor mounted on a humanoid robot as the source of labels, which are used to learn to detect obstacles in the corresponding images.

A common theme in these works is that labels are acquired simultaneously to the data they are associated to. Our approach crucially differs in that we derive supervisory labels from short-range sensor data acquired at a *different time* than the long-range data to be classified, when the robot is at a different pose.

In this regard, similar approaches have been used in literature for terrain classification [14], [15]: in these approaches, accelerometer data is collected along with the front-facing camera's feed. Training examples are generated by matching the two streams in such a way that the image collected at a given time, which contains the visual representation of a terrain patch in front of the robot, is associated to the accelerometer readings collected when the robot was traversing that specific terrain patch, from which the label is derived. Note that this implies that the mapping between the image and the future robot poses is known, i.e., that the long-range sensor is calibrated. Our approach does not rely on the knowledge of such mapping; instead, we expect the Machine Learning model, which is fed the raw long-range sensor data without any specific geometric interpretation, to automatically devise it; this also allows us to simultaneously train for multiple labels at different relative poses.

Gandhi et al. [16] train a model to determine whether the image acquired by the front camera of a drone depicts a nearby obstacle or not. The former class is assigned to all images acquired near the time in which of a drone crash is detected; remaining images are associated to the latter class. This approach can be seen as a specific instance of the one we propose, with the camera as a long-range sensor, a crash detector as a short-range sensor, and a single label corresponding to a generic "nearby" pose. Van Hecke et al. [17] adopt a similar approach to estimate average depth using a monocular image, by using the stereo vision depths from the past as trusted ground truth.

One of the main advantages of self-supervised learning approaches is that they can feature on-line learning, i.e. automatically adapting models with training data acquired on the spot, or even learning models from scratch. For example, Lieb et al. [18] rely on the assumption that the robot is initially placed on a road: then, the trapezoidal region just in front of the robot can be safely assumed to be a good representation of the road's visual appearance; a learned model then classifies similar patches in the rest of the image as traversable. Most mentioned works motivate self-supervision as an effective way to automatically adapt to new environments, counteract changes in illumination and environment conditions; this solves an important drawback with machine learning applications to robotics. Our approach shares these potential advantages, which are intrinsic to all self-supervised approaches; however, note that our experimental validation does not investigate the advantages of online learning, and instead carefully avoids to use training data acquired in the same environment as evaluation data, in order to produce conservative performance metrics.

Self-supervised techniques are also frequently adopted for grasping tasks [19], [20]. For example, Pinto and Gupta [21]

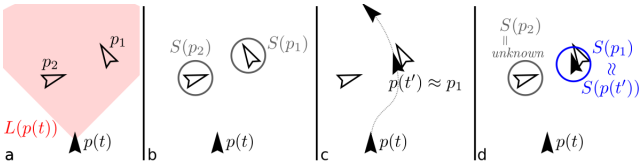

Fig. 2: (a) A mobile robot at pose $p(t)$ has a long-range sensor L (red) and (b) a short-range sensor S. Our objective is to predict the value of S at n target poses $p_1, p_2, \dots p_n$ from the value of $L(p(t))$. (c, d) For a given instance, we generate ground truth for a subset of labels by searching the robot's future trajectory for poses close to the target poses.

predict the probability of successfully grasping an object for different orientations of the end-effector using a camera image of the object to be grasped as input. A force sensor attached to the end-effector is used to determine if the grasp was successful and thus generate binary labels for each attempt. An automated approach can generate a large dataset of 50K attempts from 700 robot hours with limited human effort; note that the dataset we collect in this work has a similar cardinality but was acquired with a much reduced expense of robot time, since in our setting samples can be generated at a much higher rate.

III. Model

A. Problem Definition and Notation

We consider a mobile robot with pose $p(t)$ at time t; the robot is equipped with one long range sensor L and one or more short-range sensors $S_i, i = 1, \dots, m$. For simplicity, we limit the analysis to wheeled mobile robots for which $p(t) \in SE(2)$. We model all sensors as functions L, S_i that map the robot's pose to sensor readings.

We assume that short-range sensors return binary values $S_i(p) \in \{0,1\}$ that provide very local but unambiguous information for the robot (e.g., bumpers). Instead, long-range sensors provide a wider but maybe not directly interpretable information (e.g., a camera).

We define a set $\{p_1, \dots, p_n\}$ of predefined *target poses* relative to the current pose $p(t)$ (see Figure 2): our objective is to predict the readings $S_i(p_j)$ of short-range sensors at the target poses, given the current reading $L(p(t))$ of the long-range sensor.

B. Learning-based solution

We cast the problem as a supervised learning task. We gather a large dataset of training instances and use it to model the relation between L and S. Every sample consists of a tuple $(L(p), S_1(p_1), S_2(p_1), \dots, S_m(p_1), S_1(p_2), \dots, S_m(p_n))$.

a) Data collection: The dataset is collected in a self-supervised manner as the robot roams in the environment while sensing with all its sensors and recording odometry information; for each time t, we record $(p(t), L(p(t)), S_1(p(t)), \dots S_m(p(t)))$.

b) Self-supervised label generation: After the data is collected, we consider each pose in the dataset as a training sample. Let $p(t)$ be such pose. In order to generate ground truth labels, we consider each of the target poses $\{p_1, \dots, p_n\}$ in turn. For each given target pose p_j, we look for a time t' such that $p(t')$ is closest to p_j. In this step, we may limit the search to $t' \in [t - \Delta t, t + \Delta t]$, e.g., to limit the impact of odometry drift. If the distance between $p(t')$ and p_j is within a tolerance δ, the recorded values of $S_i(p(t')), i = 1 \dots n$ are used as the labels for target pose p_j. Otherwise, the labels associated to target pose p_j are set to *unknown*. Therefore, it is possible that for a given instance some or even all labels are unknown. While in the former case the instance can still be used for learning, in the latter case it must be discarded.

The amount of training instances for which a given label is known depends on the corresponding target pose and on the trajectory that the robot followed during data acquisition. Section IV-B illustrates a robot's behavior designed to efficiently generate a large dataset for a specific set of target poses.

c) Learning: The machine learning problem is an instance of multi-label binary classification with incomplete training labels that predicts the value of m sensors at n poses (i.e., $n \times m$ labels) given one reading from L. The specific model to solve this problem depends on the type of the data generated by L; in Section IV-E, we consider a setting in which L outputs images, therefore we adopt a Convolutional Neural Network.

IV. Experimental Setup

A. Test Platform

The robot platform adopted for the experiments is a Mighty Thymio [1], a differential drive robot equipped with 9 infra-red proximity sensors with a range of approximately 5 cm to 10 cm, depending on the color and size of the object. 5 of these sensors point towards the front of the robot at angles of $-40°$, $-20°$, $0°$, $20°$, $40°$ with respect to the robot's longitudinal axis; we use these five sensors as the short-range sensors $S_1, \dots, S_5$, and treat their output as a binary value (1: obstacle in range, 0: no obstacle in range). The robot is also equipped with a forward-looking 720p webcam with an horizontal field of view of $68°$, used as the long-range sensor.

We define a set of 31 target poses $\{p_0, \dots, p_{30}\}$ which lie in front of the robot, aligned with its longitudinal axis, evenly spaced at a distance of 0 cm to 30 cm. Note that since target pose p_0 coincides with the current robot pose $p(t)$, labels for p_0 are present in every training instance.

B. Data Acquisition Controller

We implemented an ad-hoc controller for efficient unattended collection of datasets, consisting of the readings from the five short-range sensors, the robot odometry and the camera feed. The controller behavior is illustrated in Fig. 4: the robot moves forward (a) until an obstacle is detected (b) by any of the proximity sensors; at this point, it stops and defines 5 directions which are offset from the current

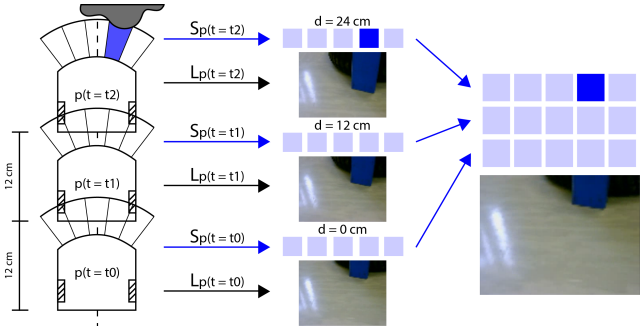

Fig. 3: Simplified illustration describing how a training instance is built. The camera image from the current pose $p(t)$ (bottom) is associated to the sensor readings (blue squares) from future poses that are close to the target poses aligned with the robot's axis.

direction by $-30°$, $-15°$, $0°$, $15°$ and $30°$ respectively (*c*, the figure shows three for clarity). For each of these directions in turn, the robot: rotates in place to align with this direction, moves back by a fixed distance of 30 cm (*d, f, h*), then moves forward by the same distance, returning to the starting position (*e, g, i*). After the process is completed, the robot rotates away from the obstacle towards a random direction, then starts moving forward again (*j*) and continues the exploration of the environment.

Note that the controller is built in such a way to efficiently populate labels for the target poses (i.e., it proceeds straight when possible); moreover, the controller strives to observe each obstacle from many points of view and distances, in order to mitigate the label unbalance in the data.

However, it is important to note that the general approach we propose is not dependent on any special controller. For any given choice of the target poses, a random-walk trajectory would eventually (albeit inefficiently) collect instances for all labels.

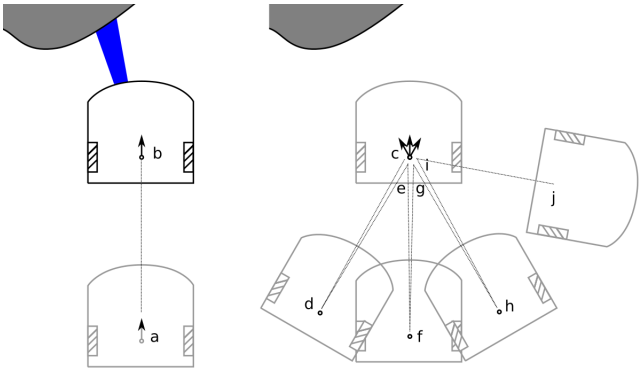

Fig. 4: Example trajectory generated by the data acquisition controller.

C. Datasets

We acquired datasets from 10 different scenarios (see Fig. 5), some indoor and some outdoor, each featuring a different floor type (tiled, wooden, cardboard, linoleum) and a different set of obstacles. For each scenario, we left the robot unattended, acquiring data for about 10 minutes using the controller described above.

The collected data amounts to 90 minutes of recording, which is then processed in order to generate labeled instances as described in section III, resulting in a total of 50K training examples extracted at about 10 Hz. Figure 6 reports the total number of known labels as a function of the distance of the corresponding target pose. Note that the total of known labels for a distance of 0 cm amounts to 250K, i.e., 50K for each of the 5 sensors. We observe that the classification problem is unbalanced in favor of negative labels; a potential countermeasure, which is not necessary in our case, is to implement a cost-sensitive loss [22].

All quantitative experiments reported below split training and testing data by grouping on scenarios, i.e., ensure that the model is always evaluated on scenarios different from those used for training. This allows us to test the model's generalization ability.

D. Data Preprocessing and Augmentation

Camera frames are resized using bilinear interpolation to 80×64 pixel RGB images, then normalized by subtracting the mean and dividing by the standard deviation. The robot's pose is expressed with three degrees of freedom $\langle x, y, \theta \rangle$ since the robot operates in 2D.

Data augmentation has been adopted to synthetically increase the size of the datasets: with probability 0.5, the image is flipped horizontally, and the corresponding target labels are modified by swapping the outputs of the left and right sensors, and the outputs of the center-left and center-right sensors; with probability 1/3, a Gaussian noise with $\mu = 0$ and $\sigma = 0.02$ is added to the image; also, with probability 1/3 the image is converted to grayscale; lastly, a smooth grayscale gradient with a random direction is overlayed on the image so to simulate a soft shadow.

E. Network Architecture and Training with Masked Loss

We use a convolutional neural network, with input shape 64 x 80 x 3 and output shape 1 x 155. Namely, the outputs consist of one binary label for each of the five sensors, for each distance in the set $\{0\,\text{cm}, 1\,\text{cm}, \ldots, 30\,\text{cm}\}$. The architecture is a simple LeNet-like architecture [23], [24] with interleaved convolutional and max-pooling layers, followed by two fully connected layers. The architecture is detailed in Fig. 7. The model is trained for a total of 15 epochs with 1000 steps per epoch, using gradient descent on mini-batches composed by 64 instances; we use the ADAM [25] optimizer with a learning rate $\eta = 0.0002$; the loss function is the mean squared error computed only on the available labels.

Because our dataset has incomplete labels (meaning that for a given instance only a subset of the labels might be known), we adopt a masking approach to prevent the loss function to be influenced by the outputs corresponding to the labels that are missing; in turn, this ensures that

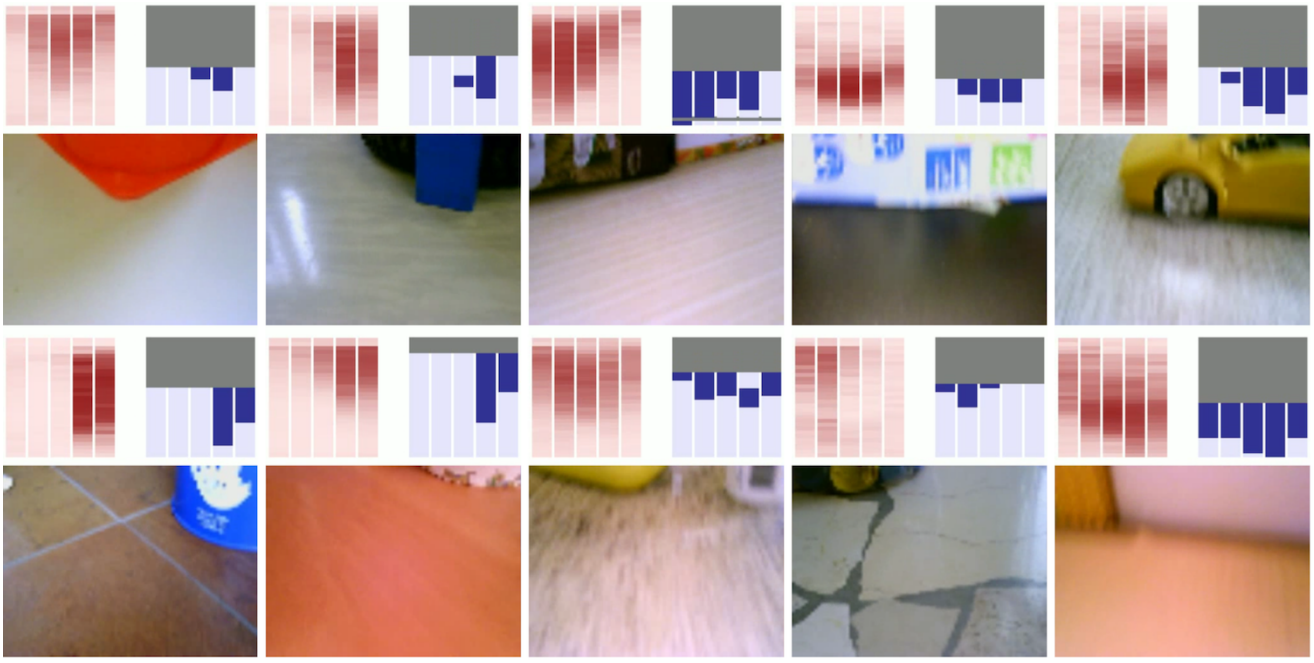

Fig. 5: 10 instances from the acquired dataset, each coming from a different scenario (top row: scenarios 1 to 5; bottom row: 6 to 10). For each instance, we show the camera image (bottom) and the 31×5 ground truth labels as a blue matrix (top right): one row per distance, one column per sensor; dark = obstacle detected; light = no obstacle detected; distances masked by gray rectangles correspond to unknown labels (due to the robot never reaching the corresponding pose). The red heatmap at the top left shows the predictions of a model trained on the other 9 scenarios.

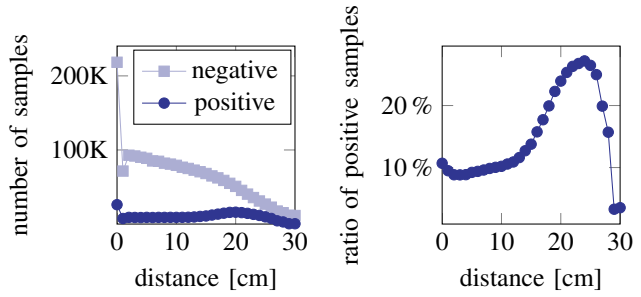

Fig. 6: Left: number of known positive (obstacle) and known negative (no obstacle) labels as a function of the distance of the corresponding target pose. Right: percentage of positive labels as a function of distance.

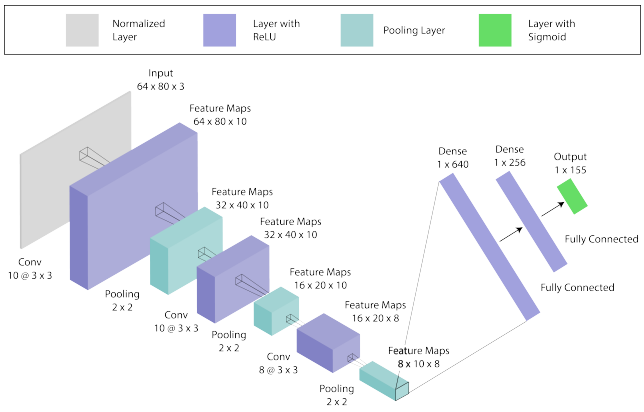

Fig. 7: Convnet architecture

the corresponding errors will not be backpropagated, which would compromise the learning process.

To achieve this, for each instance we build a binary mask containing one value per label, as implemented by Eigen et al. [26]; each value in the mask is equal to 1 if the corresponding label is known, and equal to 0 if the label is unknown. During the forward propagation step, this mask is multiplied element-wise with the difference between the prediction and the ground truth for each output, i.e., the error signal. This nullifies the error where the mask is 0 (i.e., for the subset of labels which are unknown for the given instance), and lets it propagate where the mask is 1.

F. Performance Metrics

We evaluate the performance of the model by computing the area under the receiver operating characteristic curve (AUC) for every output label (corresponding to a distance-sensor pair). This metric is evaluated over 100 rounds of bootstrapping [27] to robustly estimate a mean value and a confidence interval. Because of the heavy class unbalance in the dataset (see Figure 6), accuracy is not a meaningful metric in this context; instead, AUC is not affected by class unbalance and does not depend on a choice of threshold. In particular, an AUC value of 0.5 provides a clear baseline: in fact, it corresponds to the performance of a baseline classifier

that always returns the most frequent class; conversely, an AUC value of 1.0 corresponds to an ideal classifier. We use these fixed bounds in all our figures.

V. Experimental Results

We report two sets of experiments. In the first set (Section V-A) we quantify the prediction quality using the datasets described above, acquired on the Mighty Thymio. In the second set (Section V-C) we aim to test the robustness of the approach: to this end, we consider a model trained on these datasets and report qualitative results for testing on video streams acquired in different settings; we finally refer the interested reader to videos showing the system in use as the sole input of an obstacle avoidance controller.

A. Quantitative results on Mighty Thymio datasets

We consider a model trained on scenarios 1 to 8, and we report the results on testing data from scenarios 9 and 10.

Figure 8 reports the AUC values obtained for each sensor and distance. Figure 9 reports the same data as a function of distance, separately for the central and lateral sensors. Note that "distance" here does not refer to the distance between the obstacle and the front of the robot; instead, it refers to the distance of the corresponding target pose as defined in Section III, which corresponds to the distance that the robot would have to travel straight ahead before the proximity sensor is able to perceive the obstacle.

We observe that overall prediction quality decreases with distance. This is expected for two reasons: 1) the training dataset contains fewer examples at longer distances, and those examples exhibit more extreme class unbalance (Figure 6); 2) obstacles at a long range might be harder to see, especially considering the limited input resolution of the network. We also observe that:

- AUC values at very short distances (0 cm, 1 cm, 2 cm) are slightly but consistently lower than the AUC observed between 4 cm and 8 cm. This is caused by the fact that when obstacles are very close to the robot, they cover almost the whole camera field of view, and there might be no floor visible at the bottom of the image; then, it is harder for the model to interpret the resulting image.
- AUC values dramatically drop to 0.5 (i.e., no predictive power) for distance values larger than 28 cm. This is expected, since this value corresponds to the distance of obstacles when they appear at the top edge of the image; an obstacle that is placed farther than that will not be visible in the camera frame.
- For distances lower than 10 cm the central sensor is significantly easier to predict than lateral sensors. This is explained by the fact that objects that are detected by lateral sensors are at the edge of the camera field of view when they are close, but not when they are far away.

Figure 9:Right reports the AUC values obtained for each sensor, separately for each testing environment. These values have been obtained by a leave-one-scenario-out cross validation scheme. We observe that the predictive power of the model is heavily dependent on the specific scenario. In particular, scenarios 9 and 10, which were used as a testing set for the experiments above, are in fact harder than the average.

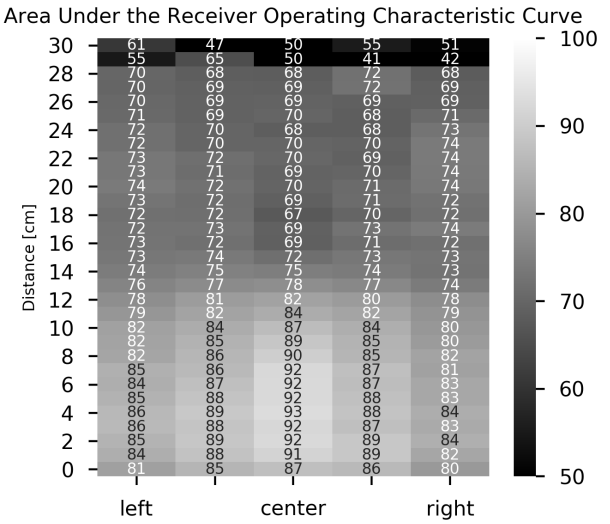

Fig. 8: AUC value obtained for each sensor (column) and distance (row).

B. Robustness tests and control

Figure 10 reports qualitative results concerning the performance of the model, trained on the whole dataset described above, when used for inference in two setups which do not match the training data. We run the model on the video stream from a TurtleBot 2 [28] robot, acquired by a laptop webcam mounted about 60 cm over the ground (compare with the Mighty Thymio camera, which is 12 cm from the ground), and oriented with a similar pitch as the Mighty Thymio camera. Because the robot has no proximity sensors, we don't have ground truth information; still, we qualitatively observe that obstacles are detected reliably. The same figure reports the results we obtain when feeding the model with data coming from a webcam mounted on the belt of an user during walking (height 95 cm, variable pitch). Videos are available as supplementary material, and also include a brief experiment showing the effects of extreme camera pitch angles. Supplementary videos show the system used as the sole input of an obstacle avoidance controller, both on the Mighty Thymio robot (with disabled proximity sensors) and on the TurtleBot 2 robot; the robots react to obstacles appropriately.

C. Simulation experiments on generic target poses

In order to highlight the generality of the approach, we run an additional experiment using a Pioneer 3-AT platform simulated in Gazebo (see Figure 11), equipped with 3 RGB cameras looking at random angles (long-range sensor) and a single short-range sensor observing the floor color just below the robot (which returns binary data: bright or dark).

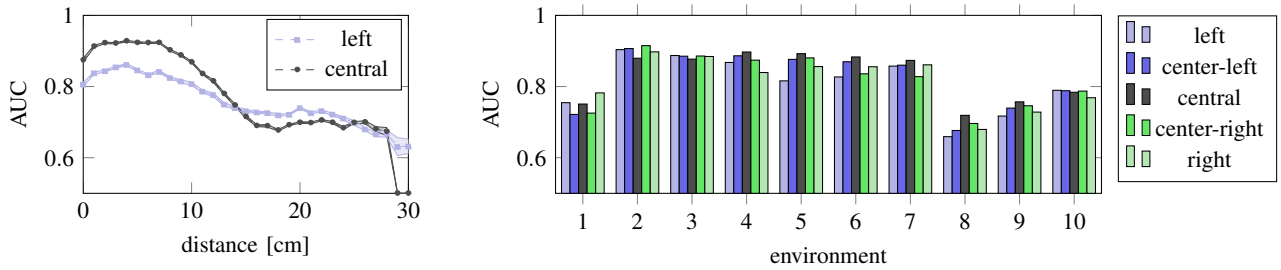

Fig. 9: Average AUC over 100 bootstrap rounds. Left: AUC for the center (black) and left (cyan) sensors as a function of distance. Shaded areas report 95% confidence intervals on the mean value (dashed line) over all environments. Right: AUC for each sensor for each environment, averaged over all distances between 0 and 30 cm.

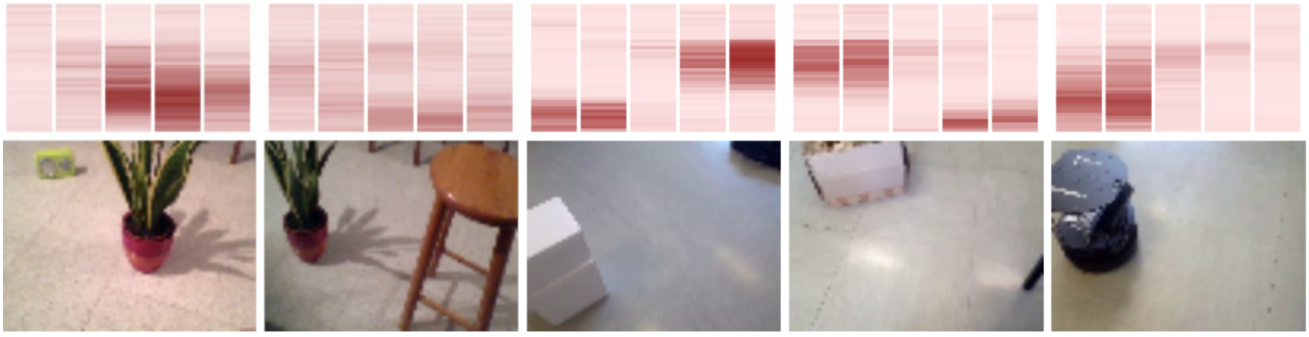

Fig. 10: Robustness tests: input (bottom images) and outputs (top heatmap) of the model trained on datasets acquired by the Mighty Thymio robot. Leftmost two images are acquired by a TurtleBot 2; the remaining three images are acquired by a belt-mounted camera.

Data is collected while the robot moves at a constant linear speed of $0.5\,\mathrm{m\,s^{-1}}$ and every 3 seconds changes its angular speed to a randomly chosen value between $-15\,^\circ\mathrm{s^{-1}}$ and $+15\,^\circ\mathrm{s^{-1}}$. We use 10 large maps with size $50\,\mathrm{m} \times 50\,\mathrm{m}$ each, featuring a planar floor textured in a random procedurally-generated black and white image obtained by thresholding low-frequency Perlin noise. On these maps, we run the controller for a total of 70 simulated minutes, respawning the robot to the center of the area should it get too close to the edge. This results in 84000 training examples collected at 20 Hz; examples for 5 maps are used for self-supervised training, the remaining for evaluation.

We consider a set of $17 \times 17 = 289$ target poses $\{p_1, p_2, \ldots, p_{289}\}$ in a square grid with a step of 0.5 m; because the short-range sensor is not affected by the robot's orientation, we disregard the orientation of the poses and depict them as small circles; the grid covers an area of $8\,\mathrm{m} \times 8\,\mathrm{m}$ and is horizontally centered on $p(t)$; it extends to 5 m in front and 3 m behind $p(t)$. The task is to predict the color of the floor (dark or bright) that the robot would measure at $p_1, p_2, \ldots, p_{289}$ given the three camera images acquired at $p(t)$.

The results on the right of Figure 11 shows that the approach learns to predict the output of short-range sensors for generic target poses, including those not on the robot's longitudinal axis, as long as the pose is visited often (i.e., its label is known in a sufficient number of training instances). Interestingly, the approach learns to predict even some target poses that are *not directly observed* by any of the three cameras; for example, the poses directly under the robot and up to two meters behind it. Note that this can not be due to the short-range sensor or its history, because the predictions are a function of a single input: the long-range sensor readings at the current timestep. Instead, the model has learned to exploit the fact that bright and dark areas in the floor are smooth and vary with low spatial frequency: this makes it possible to extrapolate the floor color on poses behind the robot, as long as the true labels for these poses are observed frequently in the training set.

VI. Conclusions

We presented a self-supervised approach that learns how to predict the future or past outputs of an informative short-range sensor by interpreting the current outputs of a long range sensor, which might be high-dimensional and hard to interpret. We implemented the approach on the Mighty Thymio robot, for the specific task of predicting the future outputs of the robot's proximity sensors (i.e., the presence of obstacles at different distances from the robot) from the video stream of the robot's forward-pointing camera. We quantitatively verified that the approach is effective and generalizes well to unseen scenarios; we qualitatively evaluated robustness to different operating conditions and usage as input to an obstacle-avoidance controller. Finally,

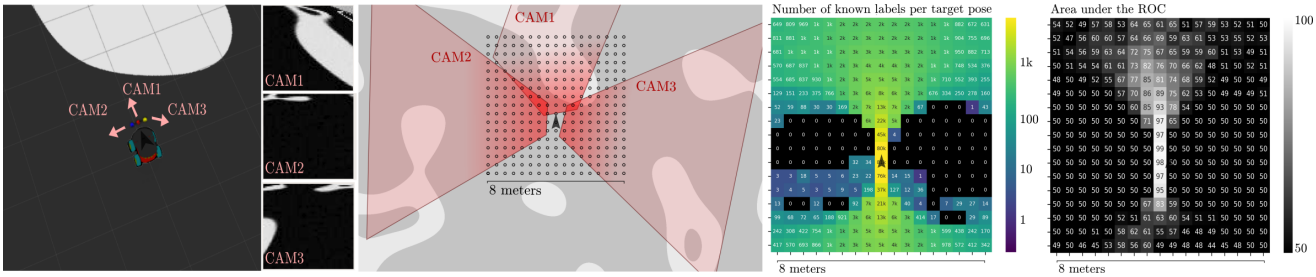

Fig. 11: From left to right: the simulated Pioneer 3-AT platform on one of the 10 random maps; images acquired by the three cameras; top-down view with the grid of target poses and the exact area of floor seen by each camera depicted in red (note that CAM1 is tilted laterally, so its imaged area is not a trapezoid); log-scale heatmap of the number of known labels per target pose in the training set; heatmap of the AUC on the testing set for each target pose.

we successfully instantiated the approach on a different, complementary task in simulation.

References

[1] J. Guzzi, A. Giusti, G. A. Di Caro, and L. M. Gambardella, "Mighty thymio for higher-level robotics education," in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence (The Eighth Symposium on Educational Advances in Artificial Intelligence, EAAI-18)*, 2018.

[2] S. Toniolo, J. Guzzi, A. Giusti, and L. M. Gambardella, "Learning an image-based obstacle detector with automatic acquisition of training data," in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence (AAAI-18 Demo Track)*, 2018.

[3] R. Hadsell, P. Sermanet, J. Ben, A. Erkan, M. Scoffier, K. Kavukcuoglu, U. Muller, and Y. LeCun, "Learning Long-Range Vision for Autonomous Off-Road Driving," *Journal of Field Robotics*, vol. 26, no. 2, pp. 120–144, 2009.

[4] K. van Hecke, G. de Croon, L. van der Maaten, D. Hennes, and D. Izzo, "Persistent self-supervised learning principle: from stereo to monocular vision for obstacle avoidance," *CoRR*, vol. abs/1603.08047, 2016. [Online]. Available: http://arxiv.org/abs/1603.08047

[5] B. Ridge, A. Leonardis, A. Ude, M. Denia, and D. Skoaj, "Self-supervised online learning of basic object push affordances," *International Journal of Advanced Robotic Systems*, vol. 12, no. 3, p. 24, 2015.

[6] B. Sofman, E. L. Ratliff, J. A. D. Bagnell, N. Vandapel, and A. T. Stentz, "Improving robot navigation through self-supervised online learning," in *Proceedings of Robotics: Science and Systems*, August 2006.

[7] A. Lookingbill, J. Rogers, D. Lieb, J. Curry, and S. Thrun, "Reverse optical flow for self-supervised adaptive autonomous robot navigation," *International Journal of Computer Vision*, vol. 74, pp. 287–302, 2006.

[8] C. A. Brooks and K. Iagnemma, "Self-supervised terrain classification for planetary surface exploration rovers," *Journal of Field Robotics*, vol. 29, no. 3, pp. 445–468, 2012.

[9] J. Ren, "Awesome self-supervised learning," https://github.com/jason718/awesome-self-supervised-learning#robotics, 2018.

[10] C. Godard, O. Mac Aodha, and G. J. Brostow, "Digging into self-supervised monocular depth estimation," *CoRR*, vol. abs/1806.01260, 2018. [Online]. Available: http://arxiv.org/abs/1806.01260

[11] S. Zhou, J. Xi, M. W. McDaniel, T. Nishihata, P. Salesses, and K. Iagnemma, "Self-Supervised Learning to Visually Detect Terrain Surfaces for Autonomous Robots Operating in Forested Terrain," *Journal of Field Robotics*, vol. 29, no. 2, pp. 277–297, 2012.

[12] H. Dahlkamp, A. Kaehler, D. Stavens, S. Thrun, and G. Bradski, "Self-supervised monocular road detection in desert terrain," in *Proceedings of Robotics: Science and Systems (RSS)*, 8 2006.

[13] D. Maier, M. Bennewitz, and C. Stachniss, "Self-supervised Obstacle Detection for Humanoid Navigation Using Monocular Vision and Sparse Laser Data," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2011, pp. 1263–1269.

[14] C. A. Brooks and K. Iagnemma, "Self-supervised terrain classification for planetary surface exploration rovers," *Journal of Field Robotics*, vol. 29, no. 3, pp. 445–468, 2012.

[15] M. A. Bekhti, Y. Kobayashi, and K. Matsumura, "Terrain Traversability Analysis Using Multi-Sensor Data Correlation by a Mobile Robot," in *Proceedings of the IEEE/SICE International Symposium on System Integration (SII)*, 2014, pp. 615–620.

[16] D. Gandhi, L. Pinto, and A. Gupta, "Learning to Fly by Crashing," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 3948–3955.

[17] K. van Hecke, G. C. de Croon, D. Hennes, T. P. Setterfield, A. Saenz-Otero, and D. Izzo, "Self-supervised learning as an enabling technology for future space exploration robots: Iss experiments on monocular distance learning," *Acta Astronautica*, vol. 140, pp. 1 – 9, 2017.

[18] A. Lookingbill, J. Rogers, D. Lieb, J. Curry, and S. Thrun, "Reverse optical flow for self-supervised adaptive autonomous robot navigation," *International Journal of Computer Vision*, vol. 74, no. 3, pp. 287–302, 9 2007.

[19] T. Mar, V. Tikhanoff, G. Metta, and L. Natale, "Self-supervised learning of grasp dependent tool affordances on the icub humanoid robot," in *Robotics and Automation (ICRA), 2015 IEEE International Conference on*. IEEE, 2015, pp. 3200–3206.

[20] S. Levine, P. Pastor, A. Krizhevsky, J. Ibarz, and D. Quillen, "Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection," *The International Journal of Robotics Research*, vol. 37, no. 4-5, pp. 421–436, 2018.

[21] L. Pinto and A. Gupta, "Supersizing Self-supervision: Learning to Grasp from 50K Tries and 700 Robot Hours," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 3406–3413.

[22] M. Buda, A. Maki, and M. A. Mazurowski, "A systematic study of the class imbalance problem in convolutional neural networks," *CoRR*, vol. abs/1710.05381, 2017. [Online]. Available: http://arxiv.org/abs/1710.05381

[23] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[24] F. J. Huang, Y.-L. Boureau, Y. LeCun, *et al.*, "Unsupervised learning of invariant feature hierarchies with applications to object recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2007, pp. 1–8.

[25] D. P. Kingma and J. Ba, "Adam: a method for stochastic optimization," in *International Conference on Learning Representations (ICLR)*, 2015.

[26] D. Eigen, C. Puhrsch, and R. Fergus, "Depth map prediction from a single image using a multi-scale deep network," in *Advances in neural information processing systems*, 2014, pp. 2366–2374.

[27] B. Efron, "Bootstrap methods: Another look at the jackknife," *Ann. Statist.*, vol. 7, no. 1, pp. 1–26, 01 1979. [Online]. Available: https://doi.org/10.1214/aos/1176344552

[28] "Turtlebot 2," https://www.turtlebot.com/turtlebot2/, accessed: 2018-09-10.