UChile Peppers 2018 Team Description Paper

Javier Ruiz-del-Solar, Cristopher Gómez, Enmanuel Almonte, José Astorga, Lukas Pavez, Nicolás Marticorena and Rodrigo Salas.

[1] Department of Electrical Engineering – Advanced Mining Technology Center (AMTC)
Universidad de Chile
http://robotica-uchile.amtc.cl/pepper-index.html

Abstract. The UChile Peppers robotics team participates in the RoboCup @Home Social Standard Platform League (SSPL) since its creation in 2017. The team is established under the Department of Electrical Engineering and the Advanced Mining Technology Center (AMTC) of the Universidad de Chile. The team's focus is the advancement of HRI research through the Pepper robot, and the development of applications that will benefit diverse populations. Our main technological and scientific contribution will be the application of the deep learning paradigm in the Pepper robot. The team collaborates with both the UChile Homebreakers and the UChileRT, from the @Home OPL and the soccer SPL, respectively, to support each other's research and developments that will benefit the RoboCup community at large.

Keywords: Deep Learning, Social Robots, Social Standard Platform League.

1 Introduction

The UChile Peppers team has been a part of the @Home SSPL since its inception, and is focused on improving the Pepper robot's service capabilities. The team works to both improve the functionality of the Pepper system through software advancements, and to develop new applications for the robot. The UChile Peppers team is committed to Open Source development and contributing to the RoboCup community. To the extent possible, all advancements are made available through GitHub; and the team looks forward to connecting and sharing with the rest of the teams at the RoboCup 2018 in Montreal.

2 Background

UChile's team members have participated in RoboCup activities in many ways: The UChile Robotics Team (UChileRT) has been involved in RoboCup competitions since 2003 in different leagues: Four-legged 2003-2007, Standard Platform League (SPL) in 2008-2017, Humanoid League in 2007-2009, @Home in 2007-2015, @Home SSPL in 2017, and @Home OPL in 2017; Dr. Javier Ruiz-del-Solar was the organizing chair of the Four-Legged competition in 2007, TC member of the Four-Legged league in 2007, TC member of the @Home league in 2009, Exec Member of the @Home league between 2009 and 2015, and co-chair for the RoboCup 2010 Sympo-

sium. Among the main scientific achievements of the group to be highlighted, are five important RoboCup awards: RoboCup 2004, 2015 and 2017 Best Paper Award, and RoboCup @Home Innovation Award in 2007 and in 2008. In addition, UChile's team members have published a total of 42 papers in RoboCup Symposia between 2003 and 2017, 19 of them corresponding to oral presentations.

In the SPL league where Nao robots are used, UChileRT reached the fourth place in the three RoboCup World Competitions (2014 in Brazil, 2015 in China and 2016 in Germany). In addition, given our work in domestic robotics during the past 10 years, and our participation in the RoboCup@Home with our own-designed Bender robot, in which we use standard development tools and middleware such as ROS, we are in a privilege position to bring together our experience of software development from the Nao platform, and our experience in domestics robotics using our own developments and standard tools (e.g. ROS) into Pepper.

In summary, we feel that we are very prepared to carry out innovative research work Pepper robot, because we have a vast experience working with Naoqi at different levels, and we have already developed software for a service robot in the @Home league using ROS, which is the framework that we want to use in Pepper. In addition, it is worth to stress that we already participated in the SSPL in 2017, obtaining the 5th position between seven teams.

3 Current research

3.1 Application of the Deep Learning Paradigm in the Pepper Robot

Deep learning has allowed a paradigm shift in pattern recognition, from using hand-crafted features together with statistical classifiers, to using general-purpose learning procedures to learn data-driven representations, features, and classifiers together. The use of the deep learning paradigm has facilitated addressing several computer vision problems in a more successful way than with traditional approaches. In fact, in several computer vision benchmarks, such as the ones addressing image classification, object detection and recognition, semantic segmentation, and action recognition, just to name a few, most of the competitive methods are now based on the use of deep learning techniques. In addition, most of the recent presentations at the flagship conferences in this area (e.g. CVPR, ECCV, ICCV) use deep learning methods or hybrid approaches that incorporate deep learning. Deep learning has already attracted the attention of the robot vision community [1]. However, given that new methods and algorithms are usually developed within the computer vision community and then transferred to the robot vision community, the question is whether or not new deep learning solutions to computer vision and recognition problems can be directly transferred to robot vision applications. We believe that this transfer is not straightforward considering the multiple requirements of current deep learning solutions in terms of memory and computational resources, which in many cases include the use of GPUs. We want to address this important challenge with the Pepper robot, by developing vision applications for the Pepper that can work without any external hardware. In order

to achieve this, we will base our work in the use of novel compression and quantization implementations of existing deep networks. In [2] we analyzed the general problem of using CNNs in robots with limited computational capabilities, and we proposed general design guidelines for their use. In addition, two different CNN based robot detectors that are able to run in real-time, in NAO robots while playing soccer, were proposed. Each detector is able to process a robot object-proposal in ~1ms, with an average number of 1.5 proposals per frame obtained by the upper camera of the NAO. The obtained detection rate was ~97%. This work will be extended and applied in Pepper robots.

3.2 Neural Networks Models

Different approaches have been proposed for the compression and quantization of CNNs. Among them, methods that compute the required convolutions using FFT [6], methods that use sparse representation of the convolutions such as [7] and [8], methods that compress the parameters of the network, and binary approximations of the filters [4]. This last option has shown very promising results. In [4], two binary-based network architectures are proposed: Binary-Weight-Networks and XNOR-Networks. In Binary-Weight-Networks, the filters are approximated with binary values in closed form, resulting in a 32x memory saving. In XNOR-Networks, both the filters and the input of convolutional layers are binary, but non-binary non-linearities like ReLU can still be used. This results in 58x faster convolutional operations on a CPU, by using mostly XNOR and bit-counting operations. The classification accuracy with a Binary-Weight-Network version of AlexNet is only 2.9% less than the full-precision AlexNet (in top-1 measure); while XNOR-Networks have a larger, 12.4%, drop in accuracy. An alternative to compression and quantization is to use networks with a low number of parameters in a non-standard CNN structure, such as the case of SqueezeNet [3]. Vanilla SqueezeNet achieves AlexNet accuracy using 50 times fewer parameters. This allows for more efficient distributed training and feasible deployment in low-memory systems such as FPGA and embedded systems such as robots. In this work, we select XNOR-Net and SqueezeNet for implementing object detectors to be used in the Pepper robots.

3.3 Design and Training Guidelines

In [2] we proposed general design guidelines for CNNs to achieve real-time operation and still maintain acceptable performances. These guidelines consist on an initialization step, which sets a starting point in the design process by selecting an existing state-of-the-art base network, and by including the nature of the problem to be solved for selecting the objects proposal method and size, and an iterative design step, in which the base network is modified to achieve an optimal operating point under a Pareto optimization criterion that takes into account inference time and the classification performance.

Initialization

- Object Proposals Method Selection: A fast method for obtaining the object proposals must be selected. This selection will depend on the nature of the problem being solved, and on the available information sources (e.g., depth data obtained by a range sensor). In problems with no additional information sources, color-based proposals are a good alternative (e.g., in [5]).
- Base Network Selection: As base network a fast and/or lightweight neural model, as the ones described in sub-section 3.2 must be selected. As a general principle, networks already applied in similar problems are preferred.
- Image/Proposal Size Selection: The image/proposal size must be set accordingly to the problem's nature and complexity. Large image sizes can produce small or no increases in classification performance, while increasing the inference times. The image size must be small, but still large enough to capture the problem's complexity. For example, in face detection, an image/window size of 20x20 pixels is enough in most state-of-the-art detection systems.

Sequential Iteration

A Pareto optimization criterion is needed to select among different network's configurations with different classification performances and inference times. The design of this criterion must reflect the importance of the real-time needs of the solution, and consider a threshold, i.e. a maximum allowed value, in the inference time from which solutions are feasible. By using this criterion, the design process iterates for finding the Pareto's optimal number of layers and filters:
- Number of layers: Same as in the image size case, the needed number of layers depends on the problem complexity. For some classification problems with a high number of classes, a large number of layers is needed, while for two-class classification, high performances can be obtaining with a small number of layers (e.g. as small as 3). One should explore the trade-off produced with the number of layers, but this selection must also consider the number of filters in each layer. In the early stages of the optimization, the removal of layers can largely reduce the inference time without hindering the network's accuracy.
- Number of filters: The number of filters in each convolutional layer is the last parameter to be set, since it involves a high number of correlated parameters. The variations in the number of filters must be done iteratively with slight changes in each step, along the different layers, to evaluate small variations in the Pareto criterion.

The proposed guidelines are general, and adaptations must be done when applying them to specific deep models and problems.

4 Approach implemented on the Pepper

In order to develop software for the Pepper robot we use the ROS (Robot Operating System) interface provided by Aldebaran. The interface enables the reading of the sensors and the control of the actuators through the standards of ROS. Also, provides a wrapper for the Naoqi library that has some interesting functionalities, like Speech Recognition and Sound Localization. With this, we can use all the software already developed for ROS compatible machines, especially those that were developed for

our Bender robot of the OPL. The diagram of the Fig. 1 shows the relationships between the ROS software and the Pepper robot. Another plus coming from using ROS is the option to use the ROS-Python Skill interface developed in our laboratory. This interface makes the high-level behaviors be easy to program. Additionally the ROS-Python Skill interface grants the option to share behaviors between our Bender robot of OPL and our Pepper robot of SSPL.

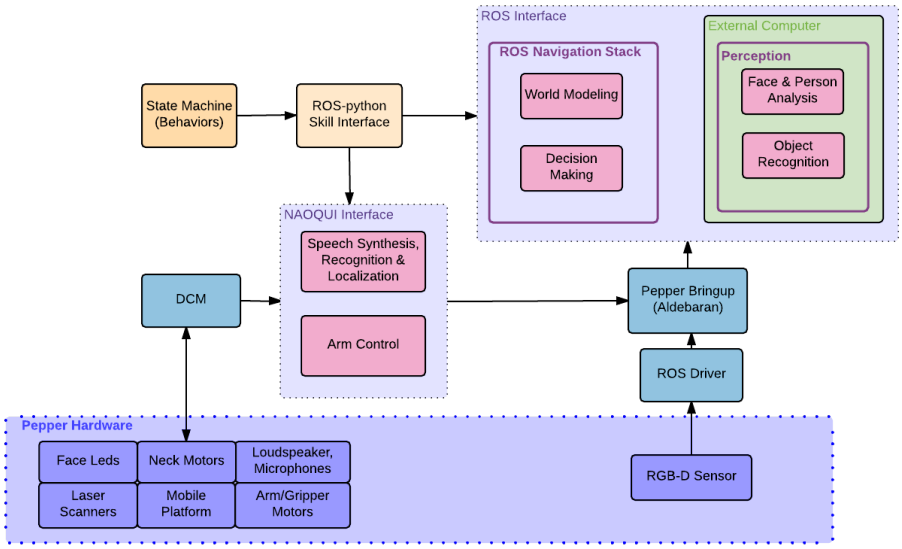

Fig. 1. Diagram ROS/NAOQUI Interface

The ROS framework has been successfully installed on the Pepper's onboard computer. The challenge to install ROS in Pepper could be very time consuming for new teams. The Operating System running on the onboard computer is an old version of Gentoo without a good package manager. Therefore, the best option is to install ROS and his dependencies from source. A list of the dependencies libraries can be found in our github page.

The main system running on the Pepper's onboard computer is the ROS Navigation Stack. The Face & Person Analysis and Object Recognition systems run on an external computer connected by wireless network with Pepper (see Fig. 1.). Given that the mentioned subsystems are very expensive in terms of computational power, just one system can run on the onboard computer. Navigation has priority above the others, because it needs a fast and reliable communication with the sensors and actuators of the robot.

5 Re-usability of the System for other Research Groups

Our goal is that all our developments will be open source. That means that the following components will be open source and will be made available:

- Vision library based on DNNs for Pepper.
- Facial features recognition system based on DNNs library.
- Image segmentation system based on DNNs library.

6 Applications in the real world

Pepper is a social robot designed to charismatically relate to people, as it has many skills with the potential for great social impact. The UChile Peppers team is disseminating the associated technologies, contributing to the development of robotics itself and motivating the interest of children and young people, through talks at different schools and universities, continuing and supplementing the efforts started with the Bender robot of the UChile Homebreakers team. Taking advantage of its social skills and their impact on children, Pepper can help teach different things in schools, not just robotics. Moreover, Pepper will be very relevant in the development of students at university level as a research subject in different fields, especially in those dedicated to human-robot interaction, due to its vanguard technology.
Two initiatives under development are the following:

- A project carried out by one of the team members (E. Almonte) for making Pepper a sight guide for people with impaired vision. A first version of this system will be ready at the end of this year.
- The use of Pepper as a motivation tool for children. Our Pepper robot has already participated in two science fairs (see pictures in Fig. 2), and monthly visit of schoolchildren to our laboratory are already scheduled for this and next years.

Fig. 2. Pepper participating in science fairs.

7 Conclusions and future work

In this TDP, the plans and goals of the new UChile Peppers robotic team have been described. As a new team, they count on the support provided by the Universidad de Chile's other robotics teams, namely UChile Homebreakers and UChileRT.

To this end, various lines of research are being pursued, with the common goal of implementing powerful solutions in computationally limited platforms. This is important research because by their very nature, mobile robots such as Pepper are computationally limited.

The research efforts of the UChile Peppers team are focused around the application of the Deep Learning Paradigm through the use of Deep Neural Networks (DNNs), where the team is investigating novel implementations of compression and quantizations of DNNs. Alongside this research, the team is developing algorithms for Semantic Segmentation of indoor environments and for Object Recognition, using DNNs.

8 Acknowledgments

This work was funded by FONDECYT Project 1161500.

References

1. RSS 2016. Workshop: Deep Learning For Autonomous Robots.
2. Cruz, N., Lobos-Tsunekawa, K., Ruiz-del-Solar, J., (2017). Using Convolutional Neural Networks in Robots with Limited Computational Resources: Detecting NAO Robots while Playing Soccer. RoboCup Symposium 2017 (in press; best paper award).
3. Iandola, F.N., Moskewicz, M.W., Ashraf, K., Han, S., Dally, W.J., Keutzer, K.: Squeezenet: Alexnet-level accuracy with 50x fewer parameters and<1mb modelsize. CoRR (2016), http://arxiv.org/abs/1602.07360
4. Rastegari, M., Ordonez, V., Redmon, J., Farhadi, A.: XNOR-Net: ImageNet Classification Using Binary Convolutional Neural Networks, ECCV, 2016
5. Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., Darrell, T.: Caffe: Convolutional architecture for fast feature embedding. arXiv preprint arXiv:1408.5093 (2014)
6. Jaderberg, M., Vedaldi, A., Zisserman, A.: Speeding up Convolutional Neural Networks with Low Rank Expansions. arXiv:1405.3866 [cs.CV]
7. Liu, B., Wang, M., Foroosh, H., Tappen, M., Penksy, M.: Sparse Convolutional Neural Networks, 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, 2015, pp. 806-814.
8. Han, S., Mao, H., Dally, W.J.: Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding (ICLR'16, best paper award)
9. Davis E. King. Dlib-ml: A machine learning toolkit. Journal of Machine Learning Research, 10:1755–1758, 2009.
10. Brandon Amos, Bartosz Ludwiczuk, and Mahadev Satyanarayanan. Openface: A General-Purpose Face Recognition Library with Mobile Applications. Technical report, CMU-CS-16-118, CMU School of Computer Science, 2016.
11. Gil Levi and Tal Hassner. Age and Gender Classification Using Convolutional Neuralnetworks. In IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) workshops, June 2015.
12. Luca Mella and Daniele Bellavista. Recognizing Emotional States in Faces. https://github.com/luca-m/emotime

Robot's Description

Pepper's hardware description is standard and can be found on the official documentation provided by Aldebaran.

Regarding software, the Aldebaran supported ROS packages provides the core features to work with the robot (drivers and URDF model of Pepper). Furthermore, we have developed an improved version of the simulated robot for Gazebo, on top of the baseline provided by Aldebaran. A major problem with the ROS approach is the limited computational resources that Pepper has. Considering this, an important amount of effort is being dedicated to optimize the originals ROS packages.

- Navigation: ROS Navigation stack will be used for mapping, localization and planning. However, due to the limited field of view of Pepper short-range lasers, the robot's cameras will be used. A Visual SLAM implementation is in process.
- Face detection/recognition: For detection we use the **dlib** library [9]. For recognition we use the Openface research [10] based on the framework Torch.
- Facial features recognition: The following facial features have been developed: age and gender recognition [11] using CNN with the framework Caffe, and emotion recognition with the emotime library [12].
- Speech recognition and generation: We use the already incorporated speech recognition and generation system in the robot; Nuance.
- Arms control and two-hand coordination: The ROS MoveIt! Package is used and has been successfully tested in the simulated version of Pepper.

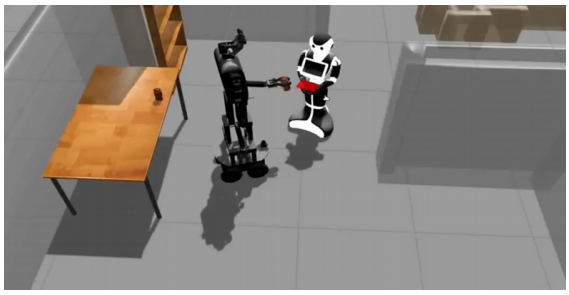

Fig. 1. Simulation of Bender and Pepper robots.

The implementation of high level behaviors is achieved with hierarchical state machines, programed with the **smach** library of Python. An emphasis is applied to the modularity and reutilization of code.

Additionally, for the purpose of making the programming of high level behaviors more straightforward, a middle layer between the states machines and the ROS control interface exist based on the **robot_skill** interface designed by Tech United Eindhoven @HOME team.