Summary

Shaoqing Yu, syu702

In this paper, a cloud-based object recognition engine, named CORE, for robotics is introduced by Beksi et al. [1]. With a distributed, modular and scalable architecture, this platform, which lives in ROS ecosystem, can solve the conflict between the increasing data amount as well as complexity and the limited on-board resources of robotics, by keeping the necessary object recognition resources on-board while distributing computational data to the cloud.

The authors firstly describe and analyse the challenges resulting from the increasing demand on vast computational data and resources in improving the performance of robotic object recognition, and then point out that a cloud-based system like CORE, which enable to access computing and storage facilities through network, is the solution. Some other specialized recognition systems currently used academically or industrially are also compared here to confirm the advantages and wide acceptance of the cloud-based approach.

In next section, the authors explain the architecture and design choices about CORE in detail. The CORE system consists of 3 parts, which aim to large-scale object training, feature-descriptors-and-classifiers-based object recognition and intelligent data transferring, respectively. To have better classifier models for object classification, CORE can be configured to pull data from a dataset named RGB-D with the large-enough amount of labelled training data. After that, the classifier models, which are updated when a new object is discovered by robotics, can be utilized to deal with the object classification queries by robotics pushing RGB-D point data over the network. Specific to the on demand machine learning process, CORE is designed with different machine learning classifiers and object feature descriptors to handle different cases. The users can even use their own support vector machine or dictionary learning as object classifier cooperating with the embedded descriptors in PLC. To deal with the large amount of data cased by high frame rates to be transferred, CORE will perform culling by measuring the scene entropy of sequential point cloud frame to reduce the data on transmission.

The communication protocols of CORE are also explained and demonstrated over examples by the authors. The communication can be roughly divided into 2 types: robot to cloud and robot to robot. The robot to cloud communication is achieved by UNIX sockets and rosbridge. There are 4 kinds of messages involved, namely, Initialize Connection, Query, Prediction and Close Connection. The local communication between robots is done by publishing and subscribing to ROS topics.

The authors also share the results of 2 specific use cases, one is CORE connected by many robots carrying on object recognition tasks, and the other handles the situation where cloud services are not available, to evaluate the performance of CORE and draw a conclusion that CORE is an quite ideal framework for performing object recognition with groups of RGB-D sensor equipped robots. The development of additional machine learning components and deployment of CORE by software containers, such as Docker, are mentioned at last as the future plan for CORE.

Overall, this is a good paper for readers to learn the architectural design and detailed implementation of CORE. However, the technologies associated with machine learning are over-emphasised, while the information about robotics, such as ROS, is very few. To gain a comprehensive under-

standing of robotics area, it is better to read some papers containing more general background information of robotics.

References

[1] W. Beksi, J. Spruth and N. Papanikolopoulos, "CORE: A Cloud-based Object Recognition Engine for robotics", *in Proceedings of the 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015, pp. 4512-4517.