DNN's for precise manipulation of household objects

Personal Information

- **Name:** Mohamed Shawky Zaky.
- **Email:** mohamedshawky911@gmail.com
- **Mobile:** (+20)1271194141
- **LinkedIn Profile:** https://www.linkedin.com/in/mohamed-shawky/
- **Github Profile:** https://github.com/DarkGeekMS

GSoC Information

- **Have you participated in Google Summer of Code in the past?** No.
- **Have you applied to GSoC in the past?** No.
- **Are you applying to any other organizations this year?** No.
- **How many hours will you devote to your GSoC project each week? What are your other summer plans?** My college semester ends in the third week of June. So, in this period I can devote 5-6 hours a day for the project. After that, I will be able to work 8-10 hours a day or even more, if required. I don't have other summer plans than GSoC.

Technical Information

- **Experience with Git:** I'm working with Git/Github through all my personal, college and internship projects. My Github profile: https://github.com/DarkGeekMS
- **Experience with C++ and Python:** Most of the projects I have done are in Python and C++, some projects can be found on my Github profile.

Example Python Projects:

- https://github.com/DarkGeekMS/Tensorflow-GANs-Architectures-Implementation
- https://github.com/DarkGeekMS/artistic-style-transfer-using-texture-synthesis

Example C++ Projects:

- https://github.com/DarkGeekMS/OpenGL-In-The-Middle-Of-Nowhere
- https://github.com/DarkGeekMS/mini-os-sim

- **Experience with Deep Learning:** I have been working with Deep Learning for nearly three years now, had an internship as a Deep Learning Researcher at Valeo and a winter internship as a Machine Learning Engineer at Arête Global. I have a publication at NeurIPS 2019 (https://arxiv.org/abs/1911.10575).

- Experience with Machine Learning libraries: I have worked with PyTorch, Tensorflow, Keras, OpenCV, SKLearn and more throughout my college and internship projects and also Kaggle competitions. Some of which can also be found in my Github profile.

- Experience with RoboComp: I have been working with RoboComp for two months. I went through the installation and tutorials, understood how components work and tried the connection with V-REP. Also, I migrated some components to Python3 to further understand the structure. Furthermore, I started working on my own Python pose estimation component.

- Experience with Pose Estimation: Recently, I read the recent research work on object pose estimation. I have explored open source codes for different architectures. Previously, I have worked on human pose estimation in a college project.

Project Information

- Idea: DNN's for precise manipulation of household objects.

- Mentors: Pablo Bustos, Pilar Bachiller.

- Personal Motivation: I'm extremely interested in Machine Learning (Deep Learning) applications in Computer Vision and Robotics. Recently, I got very interested in 3D Deep Learning and I think 3D Pose Estimation is one great application of it. Also, I'm interested in the RoboComp environment. So, I think it will be a great learning opportunity for me to work on this project.

- Project Problem and Motivation: The ability of a robot to detect, grasp and manipulate objects is one of the key challenges in building an intelligent humanoid robot. For a collaborative robot, Dealing with objects is a key part of its job and its ability to manipulate objects starts with the accuracy of grasping. The control of a robotic arm in a collaborative robot has been an active area of research for a long time and one of the main problems in such area is the grasping ability. Intelligent control of a robotic arm to manipulate objects is based on detecting the object and understanding its pose. Precise 3D poses of the objects are necessary to achieve a successful grasp on the objects. We can use the available data from the surrounding environment to recognize the objects poses. In our environment, the data provided will be RGBD frames. Recent work in Deep Learning has achieved amazing results in the problem of 3d pose estimation using RGBD data. In this project, we will work to integrate a pose estimation component to RoboComp using some state-of-art work in Deep Learning. The output poses will be visualized in V-REP simulator and used to drive the new Kinova Gen3 arm, in order to precisely grasp and manipulate some objects.

- Recent Approaches: Having investigated the recent work in pose estimation using deep neural networks, recent pose estimation models can be divided into three main types based on the used technique. First, there are models based on comparison with single object point cloud, these models use a set of predefined point clouds of a set of objects to check whether the corresponding object exists in the scene or not and its pose. Second, models that use regions of interest to detect all objects, these models use region proposal techniques and then perform pose estimation on the deduced regions of interest. Third, models that detect all objects in a single shot, these models perform pose estimation in a single shot manner, where you have a backbone that extracts features and some other classifier/regressor network at its end to predict the object class and pose.

- Proposed System:

1) Architecture: I propose that we work with an architecture that estimates poses in a single shot, as this type of networks are fast and easy to tune, while maintaining high accuracy. *Segmentation-driven 6D Object Pose Estimation* (https://arxiv.org/abs/1812.02541) is one the recent architectures that performs accurate pose estimation in a single shot manner, by using *Yolov3's Darknet-53* encoder and two streams for segmentation and regression of interest points. The network is simple, yet very efficient. Also, we can consider other architectures in our trials, like *SingleShotPose* (https://arxiv.org/abs/1711.08848), which is another efficient single shot pose estimator, *PoseCNN* (https://arxiv.org/abs/1711.00199), which is a region proposal based model, and *DenseFusion* (https://arxiv.org/abs/1901.04780), which is based on object model comparison.

2) Datasets: The training can be conducted on some open source dataset like: *YCB Video* or *LineMOD*. Also, we can collect data from the simulator and use it to augment our training dataset, in order to further improve generalization and add more objects, if needed. We can use *LabelFusion* (https://github.com/RobotLocomotion/LabelFusion) to label our collected data and create poses. The trained model will be tested on direct data from the V-REP simulator using real Gen3 arm world.

3) Optimization: All of the previously mentioned architectures achieve reasonable performance in time. However, we can further improve the network performance by using some network compression techniques, if needed.

4) Integration to RoboComp: We can start building the component with some pre-trained model and get a working pipeline with V-REP and other required components. After so, we start tuning and even training the model for our needs. I suppose that it will be good if we have two components, one to interface with the V-REP simulator, receive the data, pre-process it, pass it to the other component and display the outputs of the other component back in the simulator. The other one is for pose estimation model inference.

Proposal Schedule

Time Slot	Tasks	Progress Indicators
Before May 4	• **Familiarize** myself more with RoboComp component structure. • **Familiarize** myself with recent work with the real Gen3 arm. • **Dive** deeper into the pose estimation architectures and know how to adapt them.	----------
May 4 – June 1 (Community Bonding):	• **Experiment** with the proposed pose estimation architectures and visualize the results on the simulator data. • **Further discuss** the ideas and goals with my mentor to decide the used architecture, the process we will go through and the future improvements to be done. Thus, with the help of my mentor, I will have a whole vision on what exactly will be done and final deliverables.	• **Visual results** on simulator collected data. • **A full plan** of the best architecture to be used, objects of interest and required improvements.
June 1 – June 22	• **Collect** data from the simulator in order to augment the training data of the chosen model with the required objects. • **Use** LabelFusion tool to label the required data. • **Complete** the training code for the proposed architecture. • **Create** the data processing component to interface with the simulator.	• **A full dataset** with the required objects and corresponding labels. • **Data processing component.** • **Full code** for the decided network.
June 22 – July 20	• **Train** the network on the collected data and do the necessary parameter tuning to reach best possible results. • **Test** the trained network on various setups to confirm generalization. • **Complete** the pose estimation component for the trained model. • **Integrate** the trained model to the pose estimation component.	• **A trained model** for pose estimation. • **Full pose estimation component** working with V-REP simulator.
July 20 – August 3	• **Complete** all required interfaces for pose estimation component with the rest of the real Gen3 arm controller environment.	• **All required interfaces** with other controller components.

	• **Visualize** the output poses and point cloud of the final model in V-REP simulator. • **Test** the grasping performance of the real Gen3 arm controller on the final model.	• **A full review** on the component performance (accuracy + time).
August 3 – August 10	• Code refactoring and documentation.	• **Required components** code with documentation. • **Used network** training code with documentation. • **Project** documentation and final report.

A buffer of two weeks has been kept for any unexpected delay or further experimentation.