A lightweight perception module for planning purposes

Muhayy Ud Din[1], Jan Rosell[1], Sohail Bukhari[2], Mansoor Ahmad[2], and Wajahat M Qazi[2]
[1]Institute of Industrial and Control Engineering (IOC)
Universitat Politècnica de Catalunya (UPC) – Barcelona Tech, Spain.
[2]Department of Computer Science
COMSATS University Islamabad – Lahore Campus, Pakistan.

***Abstract*— Sensing is an essential component for robots to perform the manipulation tasks in real environments. This study proposes a lightweight deep-learning-based sensing modules which allows the robots to automatically model the workspace for manipulation planning. This sensing module is developed as a part of our ongoing manipulation planning framework. It will be used to enhance the sensing accuracy and make it capable of planning the manipulation tasks in real environments. The retrained model is further trained over commonly used objects to enhance the prediction accuracy.**

I. Introduction

Robotic planning is one of the key areas with the focus on bringing the robots off the lab and making them capable of performing complex manipulation tasks. It consists of various subdomains such as motion planning, grasp planning, task planning and manipulation planning. Early works in these domains were focused on the formulation of the underlying theories along with the necessary conditions to fulfill the task requirements. Motion planning focused on developing the methods (such as sampling based planning approaches) to determine collision-free paths for the robot, and grasp planning on the search of stable grasps satisfying the force closure constraint. Similarly, task planning approaches focused on establishing underlying frameworks (such as heuristic search approaches) to evaluate the potential sequence of actions to perform a desired task satisfying a given set of constraints.

The current focus of planning approaches is to make robots capable of performing the tasks in real scenarios, which requires robust sensing strategies to efficiently perceive the environment and construct the robot workspace for planning. Various computer vision techniques are available for object recognition and localization [1] [2] [3]. The choice of sensing approaches largely depends on the tools that are used for planning. For instance, point-cloud-based object detection is used in *GraspIt!* (`https://graspit-simulator.github.io/`) for online grasp planning and in *moveIt!* (`https://moveit.ros.org/`) for motion planning. Object localization with fiducial markers like AR tags is done in approaches using *The Kautham Project* (a tool for task and motion planning, `https://sir.upc.edu/projects/kautham/`), which is a simple and robust method although it requires an explicit labeling of the tags over each object. Recently, deep-learning-based approaches gained much attention among the robotic community, mainly in perception using vision.

This study enhances the sensing module of *The Kautham Project* planning framework with a lightweight deep-learning-based approach. It allows Kautham to efficiently recognize and localize the objects for constructing the robot workspace for planning purposes at different levels.

The rest of the paper is structured as follow. Sec. II briefly describes the motivation behind this work, Sec. III explains the proposed approach and Sec. IV discusses how will it be used to enhance the planning framework. Finally, Sec. V concludes the study and presents the future works.

II. Motivation

The main motivation behind this work is to develop a lightweight and efficient sensing module for The kautham Project which will be used to address the following problems.

- *Sensing for motion planning:* Motion planning requires precise modeling of the workspace in order to determine the collision-free path from a start to a goal state. The proposed sensing module will recognize and localize the objects in the workspace in order to set the planning scene.
- *Sensing for grasp planning:* Grasp synthesis requires the 3D model of the object in order to determine the contact points on the objects for the force closure grasp. Planning approaches using Kautham employ GraspIt! for grasp planning, which includes a point-cloud-based sensing module. However, in the clutter scenes, it may not be able to determine the exact models and poses of the objects. The proposed module will recognize the objects and retrieve their exact 3D models from a database for the grasp computation.
- *Coping with uncertainties in task planning:* In order to handle uncertainties, task planning approaches require to incorporate sensing actions to close the loop at action level. The outcome of the sensing actions affect the further execution of the task. The availability of a simple and fast recognition module will allow the correct execution of the sequence of actions.
- *Spatial reasoning for knowledge-based task planning:* Knowledge-based task planning approaches reason over the execution of the actions. For instance, if an object is in the shelf it should be picked with the side grasp.

This work was partially supported by the Spanish Government through the project DPI2016-80077-R.

A sensing action is required to determine the spatial relation between the objects, which will be used to apply the reasoning procedures over the way of manipulation the objects.

These above stated problems have previously been addressed within The Kautham Project planning framework using AR tag-based sensing, which determines the poses of the objects using AR tags labeled over all the objects. This has been enough since the focus was on the planning strategies and the work done on our robotics lab; the details of the approaches tackling the four problems can be found in [4] [5] [6] and [7]. The current proposal will allow to work in different scenarios with every-day objects, extending all the planning framework off the lab.

III. Proposed Framework

A. Assumptions

The following assumptions are considered:

- The approach is designed for table-top manipulation problems, where objects are placed over a flat horizontal surface.
- The 3D models of all the potential objects in the environment are known.

B. Sensing

The sensing module is responsible for object recognition, localization and 3D model retrieval from the model database.

Object recognition is performed using *ImageAI* (`https://github.com/OlafenwaMoses/ImageAI`, [8]), a python-based open-source library for computer vision. It provides a wide variety of algorithms for object detection, image prediction, and video object tracking using built-in image prediction and training algorithms, trained using *ImageNet-1000* dataset (`http://image-net.org`). Furthermore, it also supports object detection, object tracking and video detection using *RetinaNet* (`https://github.com/fizyr/tf-retinanet`), *TinyYOLOv3* and *YOLOv3* (`https://pjreddie.com/darknet/yolo/`), trained on *COCO* dataset (`http://cocodataset.org`). For novel objects, it allows to train custom models for object detection and recognition.

This work uses *ResNet-50* (`https://www.kaggle.com/keras/resnet50`) for object recognition, it is a 50 layer deep convolutional neural network. Due to the feature of identity shortcut connection, its performance and accuracy is high as compared to other detection models such as *Yolo*. In order to enhance the prediction accuracy and to incorporate custom objects which are not supported by the pre-trained model, the *ResNet-50* model is further trained using the transfer learning feature of the *ImageAI* library. The data is prepared by annotating the images using *LabelImg* (a python-based tool for graphical image annotation, `https://pypi.org/project/labelImg/`). The annotations are saved in XML files as a *PASCAL VOC* format (`https://pjreddie.com/media/files/VOC2012_doc.pdf`). The generated XML files along with the images are used by the *ResNet-50* for transfer learning.

The poses of the detected objects are computed with the method described in [9]. Once the objects are recognized, their 3D models are loaded from the database and passed to the Kautham module along with their poses, as detailed next.

C. Workspace generation

The Kautham Project [10] is a C++ based open-source software for teaching and research in robot motion planning. This planning tool, whose main core of planners is provided by the Open Motion Planning library (OMPL, [11]), allows to cope with problems with one or several robots (generally considered as kinematic trees with a mobile base), and provides several advanced features like the facility to define coupling between degrees of freedom (through the definition controls that can actuate more than a single degree of freedom), the use of planners with dynamic simulation (using the ODE dynamic engine) and the integration with task planers through a flexible ROS (`https://www.ros.org/`) interface that is provided besides the console and GUI interfaces. Particularly, some relevant ROS services offered are: *OpenProblem*, *SetPlanner*, *SetQuery*, *GetPath*, *SetRobotsConfig*, *SetObstaclePos*, *AttachObstacle2RobotLink*, *SetRobControls*.

Motion planning problems are modeled in Kautham using an XML file with information on the models of the robot(s) and obstacles, the controls and the planner, as shown in Fig 1. For the robots a URDF file is given that contains the kinematic model of the robot (the joints and links), as well as the geometric models for visualization and collision-checking, and dynamic parameters such as damping and masses. Besides the URDF file, the translation limits (in case of mobile base) and the home position with respect to a fixed world reference frame are set. Obstacles can be defined using URDF or just by giving the file containing its geometry in .wrl, .stl, or .dae formats, and its location with respect to the fixed world reference frame. Internally obstacles are defined as robot data structures with no actuated degrees of freedom for the general case of fixed obstacles. For controls, an XML file is given where they are defined and related to the joints of the URDF robot model. If no control file is given, one control per joint is assumed. Finally, regarding the planner, the type of planer is given together with its parameters and the query to be solved.

In order to accurately match the environment with its model in Kautham, and be able to plan motions for the robot that are actually collision-free when executed in the real robot two alternatives are possible. In any case, it is assumed that the set of all the possible objects in the scene is known, and that the files describing their geometry is available. Then we can:

1) Use the object recognition and localization method to identify the objects in the scene and their poses and, for each of them, write the corresponding XML tag in the kautham problem file with the information of the model file and the home location.
2) Load a problem file with all the objects that can be potentially in the scene, and locate them at a given

```
<Problem name="Tiago fixed in kitchen at counter A">
  <Robot robot="robots/tiago/tiago_with_gripper.urdf" scale="1.0">
    <Limits name="X" min="-3.0" max="3.0"/>
    <Limits name="Y" min="-3.0" max="3.0"/>
    <Limits name="Z" min="0.0985" max="0.0986"/>
  </Robot>
  <!-- Red can  -->
  <Obstacle obstacle="obstacles/kitchen/can.urdf" scale="1">
    <Home TH="0" WZ="0.0" WY="0.0" WX="0.0" Z="0.91" Y="0" X="1.1"/>
  </Obstacle>
  <!--  Yellow can  -->
  <Obstacle obstacle="obstacles/kitchen/cangoal.urdf" scale="1">
    <Home TH="0" WZ="0.0" WY="0.0" WX="0.0" Z="0.90" Y="0.7" X="1.3"/>
  </Obstacle>
  <Obstacle obstacle="obstacles/kitchen/glass.urdf" scale="1">
    <Home TH="0" WZ="0.0" WY="0.0" WX="0.0" Z="0.90" Y="-0.3" X="1.2"/>
  </Obstacle>
  <Obstacle obstacle="obstacles/kitchen/glass.urdf" scale="1">
    <Home TH="0" WZ="0.0" WY="0.0" WX="0.0" Z="0.90" Y="0.4" X="1.3"/>
  </Obstacle>
  <Obstacle obstacle="obstacles/kitchen/kitchenfloor.iv" scale="1">
    <Home TH="0" WZ="0.0" WY="0.0" WX="0.0" Z="-0.001" Y="0" X="0"/>
  </Obstacle>
  <Obstacle obstacle="obstacles/kitchen/kitchen.urdf" scale="1">
    <Home TH="0" WZ="0.0" WY="0.0" WX="0.0" Z="0.0" Y="0" X="0"/>
  </Obstacle>
  <!--  Controls to move only the torso and arm keeping the base and head fixed  -->
  <Controls robot="controls/tiago_simple_only_arm_gripper_counterA.cntr"/>
  <Planner>
    <Parameters>
      <Name>omplRRTConnect</Name>
      <Parameter name="Range">0.05</Parameter>
      <Parameter name="Max Planning Time">3</Parameter>
      <Parameter name="Speed Factor">5</Parameter>
    </Parameters>
    <Queries>
      <Query>
        <Init>0.459 0.023 0.628 0.688 0.820 0.083 0.974 0.553 1.000</Init>
        <Goal>0.846 0.944 0.861 0.786 0.594 0.226 0.850 0.429 1.000</Goal>
      </Query>
    </Queries>
  </Planner>
</Problem>
```

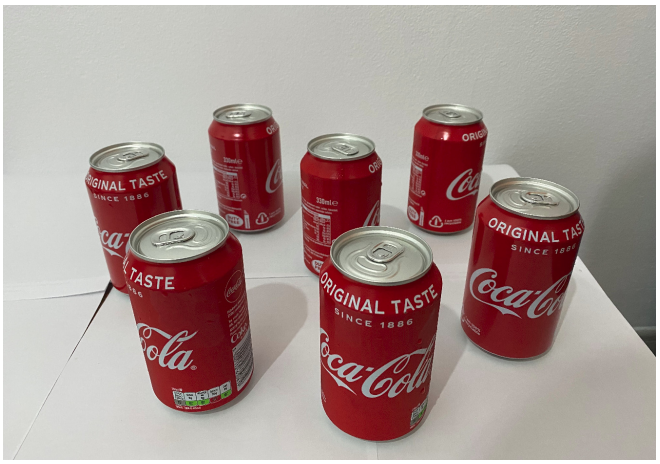
Fig. 1: An example of a problem file.

dummy location outside the workspace. Then use the object recognition and localization method to identify the objects in the scene and their poses and, for each of them, call the *SetObstaclePos* service to place them on the scene.

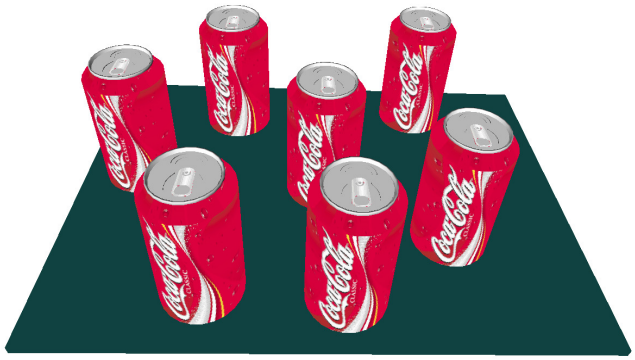
IV. Discussion

This study developed a lightweight and efficient sensing module for The Kautham Project. This module will serve as the sensing component for the knowledge-oriented task and motion planning framework. The choice of the sensing technique and the software tools to use is largely dependent on the problem being addressed. The reason not to choose point-cloud-based approaches is the sensing complexity that may be encountered due to clutter environments such as those shown in [4] [12]. It is really computationally intensive and challenging to detect the objects in clutter environments. Particularly if the the objects are placed close to each other or one behind the other. We are addressing the problems in manipulation planning domains mainly with the integration of task and motion levels of planning using knowledge-based reasoning. The knowledge is mainly used to define the way of manipulation for the objects. In this scope The Kautham Project provides an easy and flexible way of integrating different planning modules such as task planning and knowledge-based reasoning.

Previously, sensing in Kautham was performed using AR-tag based object localization and for recognition a map is generated which retrieve the object name corresponding to the AR-tag id. This approach required to place the explicit



(1)



(2)

Fig. 2: An example scene: (1) the camera image; (2) the scene in Kautham.

tag over each object, every new object must have a tag and also must be incorporated into the mapping function. This solution may not be feasible outside the lab environment. The current sensing module that is presented in this paper is a step towards more realistic and robust sensing approach for our planning framework.

An example is shown in Fig. 2 where a camera image of the real world is presented in Fig.2-1. The sensing module detects the objects and computes their poses, and also retrieves the 3D models from the model data base and writes a Kautham problem file. The obtained Kautham scene is depicted in Fig. 2-2.

The presented sensing module is a part of our on-going work for the planning framework. Below are some use cases of the sensing module within the planning framework.

The task modeling is usually done using the Planning Domain Definition Language (PDDL, [13]). Planning tasks specified in PDDL are separated into two files, a domain file for predicates and actions, and a problem file for objects, initial state and goal specification. The sensing module will

be used to automate the task modeling process regarding the PDDL problem file (the objects involved in the planning, the initial symbolic state, and the conditions that the goal has to satisfy).

Real world task planning should have the ability to handle the uncertainties at symbolic level. This module will be used within the contingent planning [6] [14] to perform the sensing actions. The contingent planning generates the conditional plan by considering the uncertainty in the initial state and in the action effect. The proposed sensing module will be used to observe the certain aspects of the current state of the world during the plan execution. The plan will be branched depending on the results of the observation. For instance, if a cup exist at the serving table then fill the cup, otherwise locate the cup, bring it to the serving table, and then fill it.

Reasoning play an important role in task and motion planning [7]. Two types of reasoning will be used in the planning framework that are: 1) *knowledge-based reasoning* that will be used to define the way of manipulation, such as a cup can be grasped from its handle; 2) *spatial reasoning* that will be used for skill-based manipulation planning, such as a drawer should be opened before locating an object in the drawer to pick. This reasoning engine will use the proposed sensing module to perceive the environment.

V. Conclusion

This study proposed a lightweight sensing module for manipulation planning. It provides an easy and flexible way to perceive the environment. This module will use to automatically create the robot workspace, model the task planning scene using PDDL. Furthermore it will be used to observer the environment during task execution to handle the uncertainty at task level and for reasoning process. This is a step in our continues development process where we will replace our AR-Tag based sensing approach with this approach.

References

[1] K. Pauwels, L. Rubio, and E. Ros, "Real-time pose detection and tracking of hundreds of objects," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 26, no. 12, pp. 2200–2214, 2015.

[2] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, "Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes," 2018.

[3] J. Josifovski, M. Kerzel, C. Pregizer, L. Posniak, and S. Wermter, "Object detection and pose estimation based on convolutional neural networks trained with synthetic data," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 6269–6276.

[4] Muhayyuddin, M. Moll, L. Kavraki, and J. Rosell, "Randomized physics-based motion planning for grasping in cluttered and uncertain environments," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 712–719, 2018.

[5] J. Rosell, R. Suárez, N. García, and M. Ud Din, "Planning grasping motions for humanoid robots," *International Journal of Humanoid Robotics*, vol. 16, no. 06, p. 1950041, 2019.

[6] A. Akbari, M. Diab, and J. Rosell, "Contingent task and motion planning under uncertainty for human–robot interactions," *Applied Sciences*, vol. 10, no. 5, p. 1665, 2020.

[7] M. Diab, A. Akbari, M. Ud Din, and J. Rosell, "PMK - A knowledge processing framework for autonomous robotics perception and manipulation," *Sensors*, vol. 19, no. 5, p. 1166, 2019.

[8] Moses and J. Olafenwa, "ImageAI, an open source python library built to empower developers to build applications and systems with self-contained computer vision capabilities," mar 2018–. [Online]. Available: https://github.com/OlafenwaMoses/ImageAI

[9] J. Liu and S. He, "6d object pose estimation based on 2d bounding box," *ArXiv*, vol. abs/1901.09366, 2019.

[10] J. Rosell, A. Pérez, A. Aliakbar, Muhayyuddin, L. Palomo, and N. García, "The kautham project: A teaching and research tool for robot motion planning," in *Proc. of the IEEE Int. Conf. on Emerging Technologies and Factory Automation*, 2014.

[11] I. Sucan, M. Moll, L. E. Kavraki *et al.*, "The open motion planning library," *Robotics & Automation Magazine, IEEE*, vol. 19, no. 4, pp. 72–82, 2012.

[12] W. C. Agboh and M. R. Dogar, "Real-time online re-planning for grasping under clutter and uncertainty," in *2018 IEEE-RAS 18th International Conference on Humanoid Robots (Humanoids)*, 2018, pp. 1–8.

[13] M. Ghallab, A. Howe, C. Knoblock, D. Mcdermott, A. Ram, M. Veloso, D. Weld, and D. Wilkins, "PDDL—The Planning Domain Definition Language," 1998.

[14] J. Hoffmann and R. Brafman, "Contingent planning via heuristic forward search with implicit belief states," in *Proc. ICAPS*, vol. 2005, 2005.