Computer Vision Task for Manipulation & Grasping

This document discusses the steps followed for achieving the detection of object by using computer vision for grasping task.
The task involves:

1. RGBD Camera interfacing with ROS based computer
2. Detection and localization of object with respect to camera link
3. Publish the TF information for moveIt for grasp planning

Following are the important factors to be considered for experiment setup:

1. Object must have some detectable features (design/patterns, written letters etc.)
2. For better depth perception, distance as recommended by Astra camera must be more than 0.4m
3. Good lighting conditions
4. Manipulator base with reference to object must be stationary at time of detection and localization of object.

1. RGBD Camera interfacing with ROS based computer

Camera used: Astra RGBD camera /USB Interface
System: Ubuntu 14.04/ROS indigo
Install required packages using following commands:

sudo apt-get install ros-indigo-astra-camera
sudo apt-get install ros-indigo-astra-launch

Follow the following link to configure the camera:

http://wiki.ros.org/astra_camera

2. Detection and localization of object with respect to camera link

- Already available ROS package find_object_2d is used to achieve the task:
  sudo apt-get install ros-indigo-find-object-2d
- Run the astra camera for publishing the image and depth information:
  roslaunch astra_launch astra.launch
- In another terminal run the find_object_3d:
  roslaunch find_object_2d find_object_3d.launch

A graphical user interface will be opened in which we can select multiple options for descriptors and detector. From edit menu user can save an object from the live stream and save it along features for detection in real time. According to our observation for best results, Fast and BRISK should be used while saving the features and GFTT and BRISK must be used for real time detection.

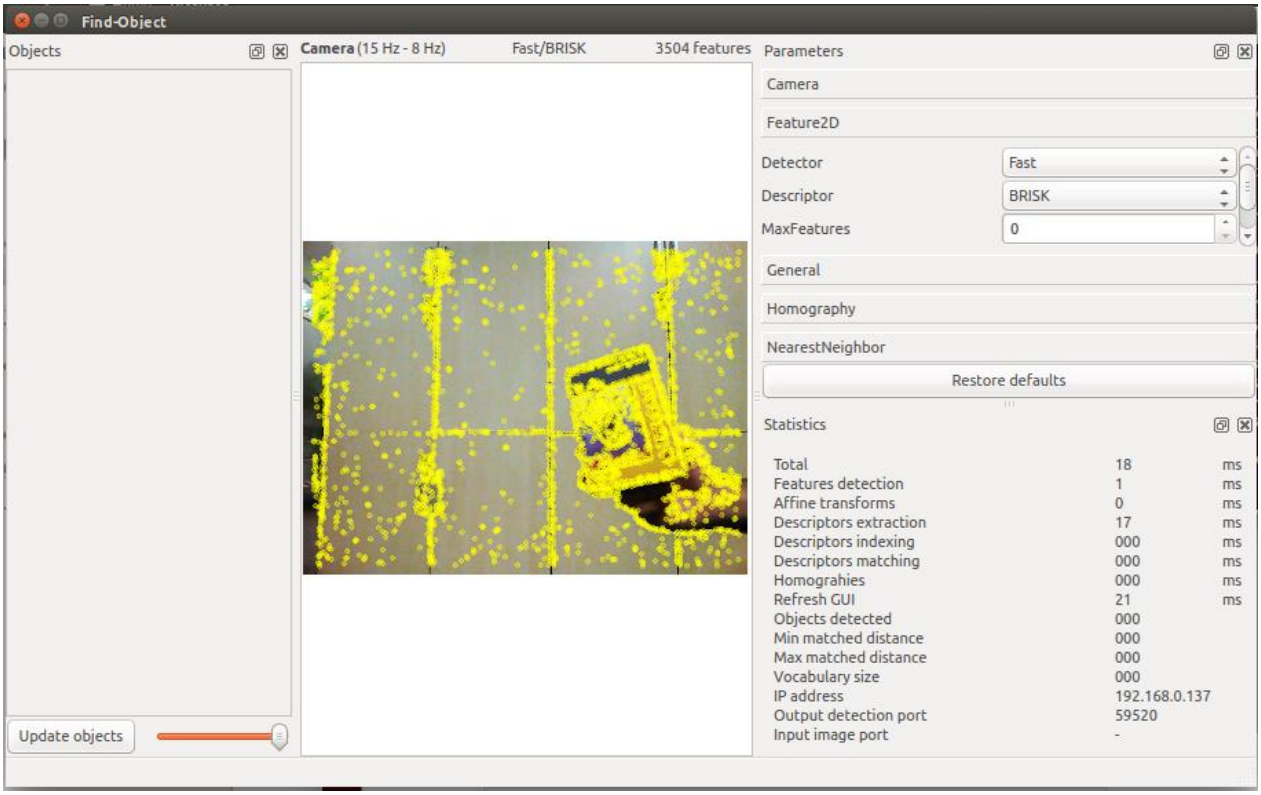

Figure 1: Using Fast as detector and BRISK as Descriptor

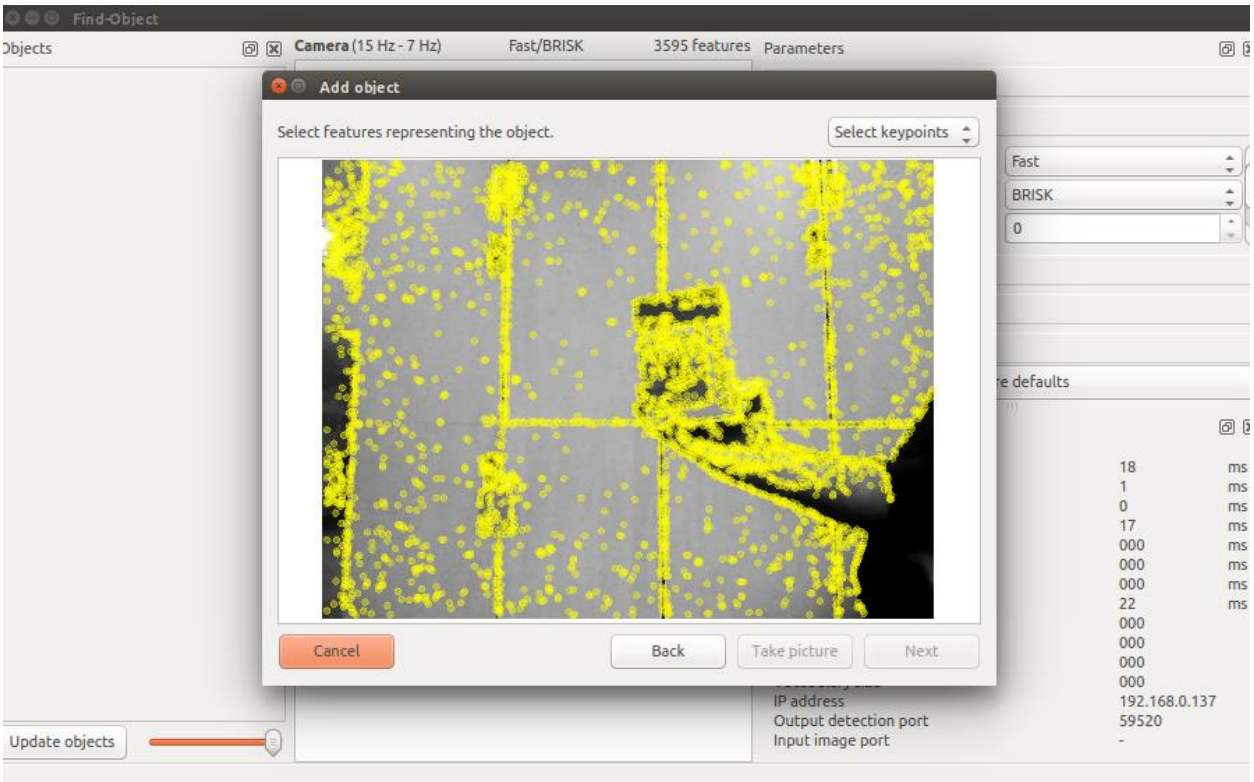

Figure 2: Saving the object with its keypoints features

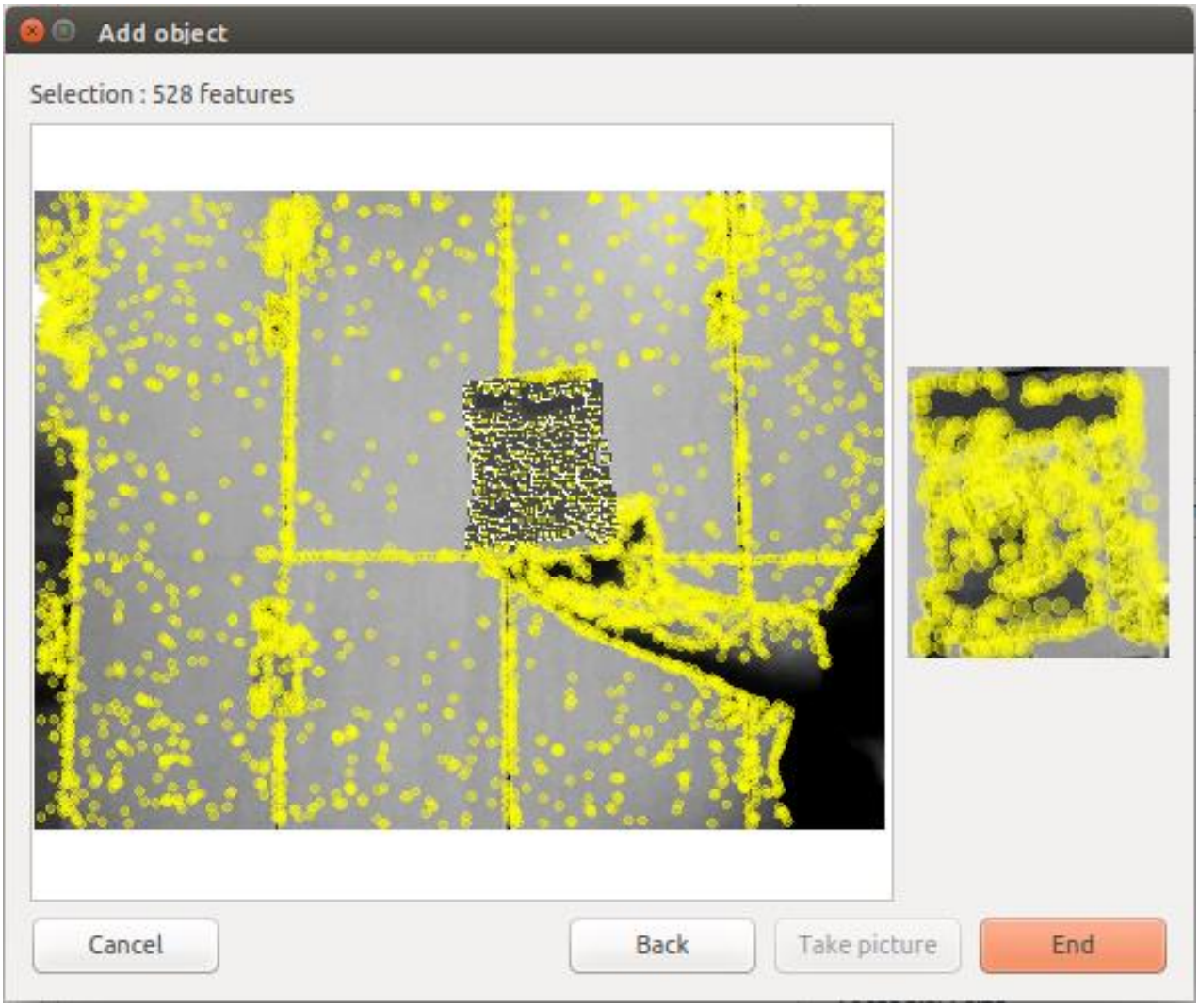

Figure 3: Selected object

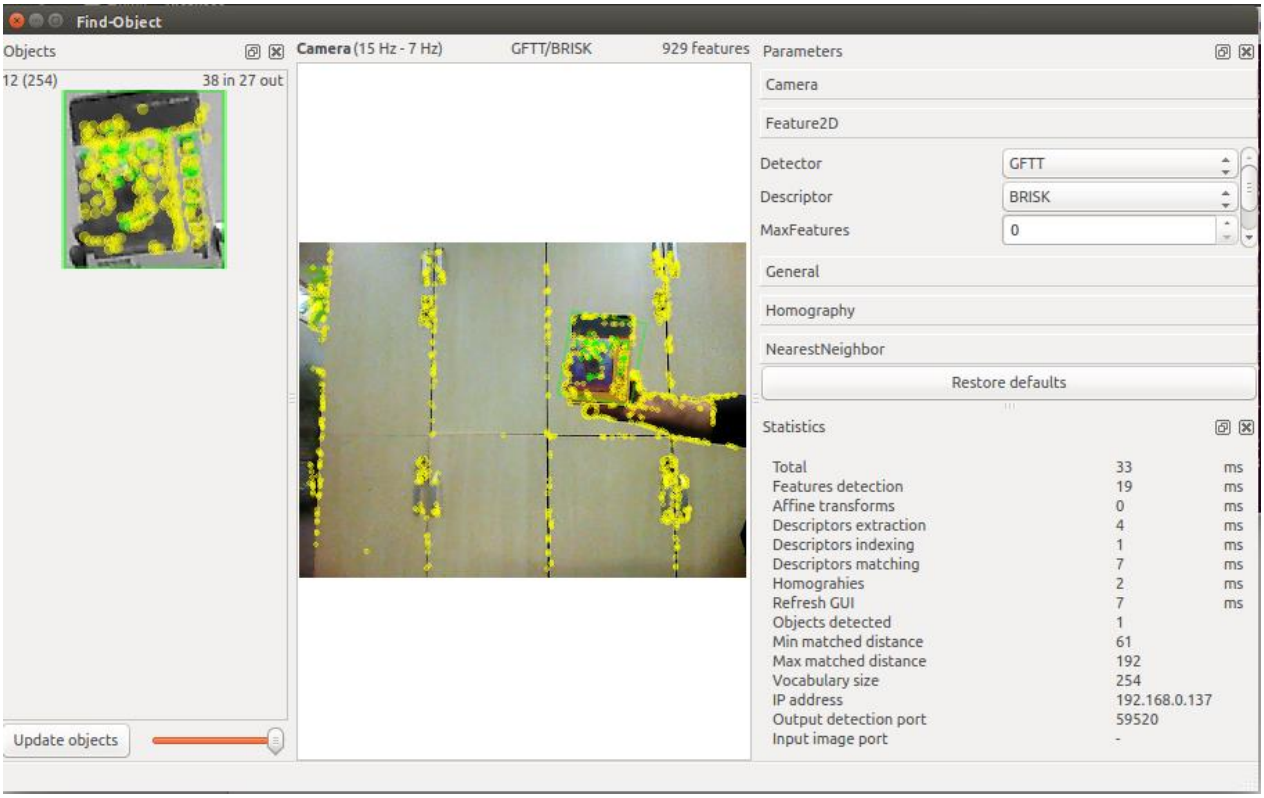

Figure 4: The object is detected in realtime at run time use GFTT as detector for the best performance

3. Publish the TF information for moveIt for grasp planning

Once the detection is done the tf information is published over the rostopic. The frames can be visualized using RVIZ by running following command:

- *rosrun rviz rviz*
- Select camera_link as fixed frame
- Add TF from add menu to visualize the detected object with reference to camera_link frame

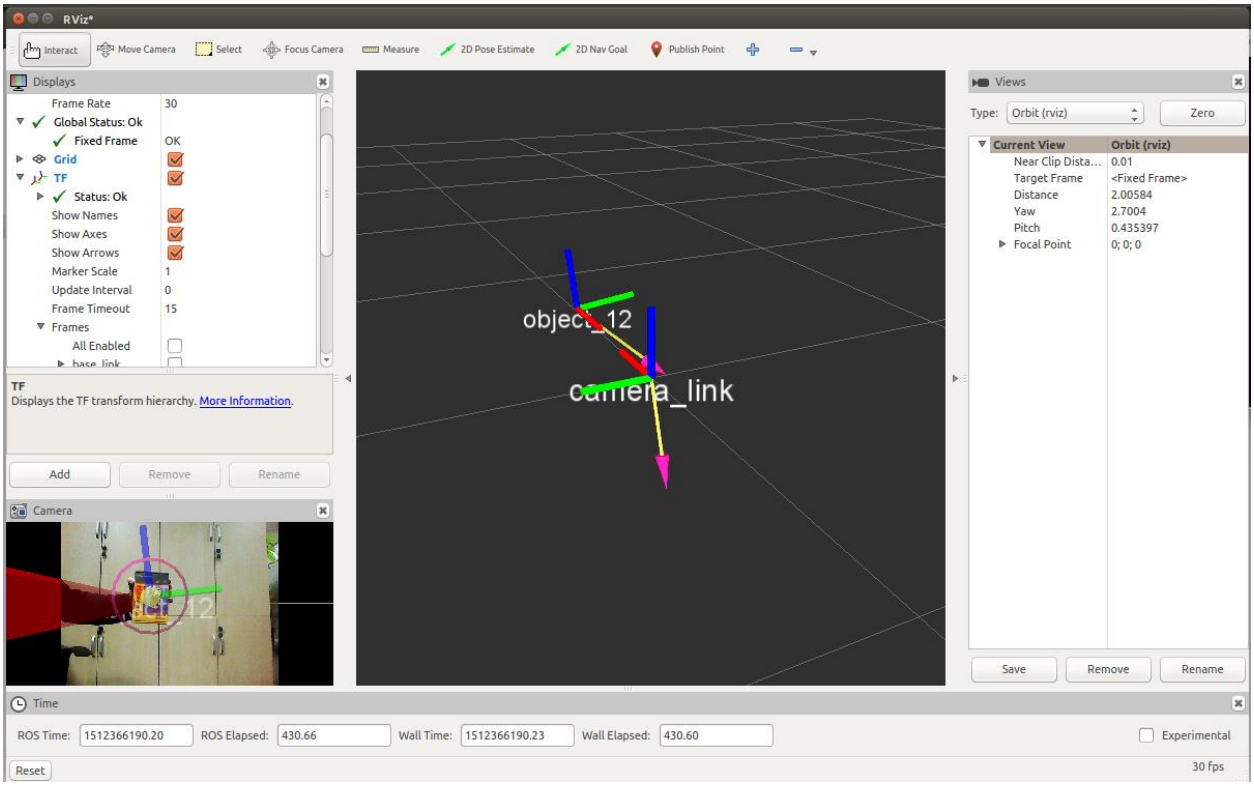

Figure 5: Visualization of object in rviz

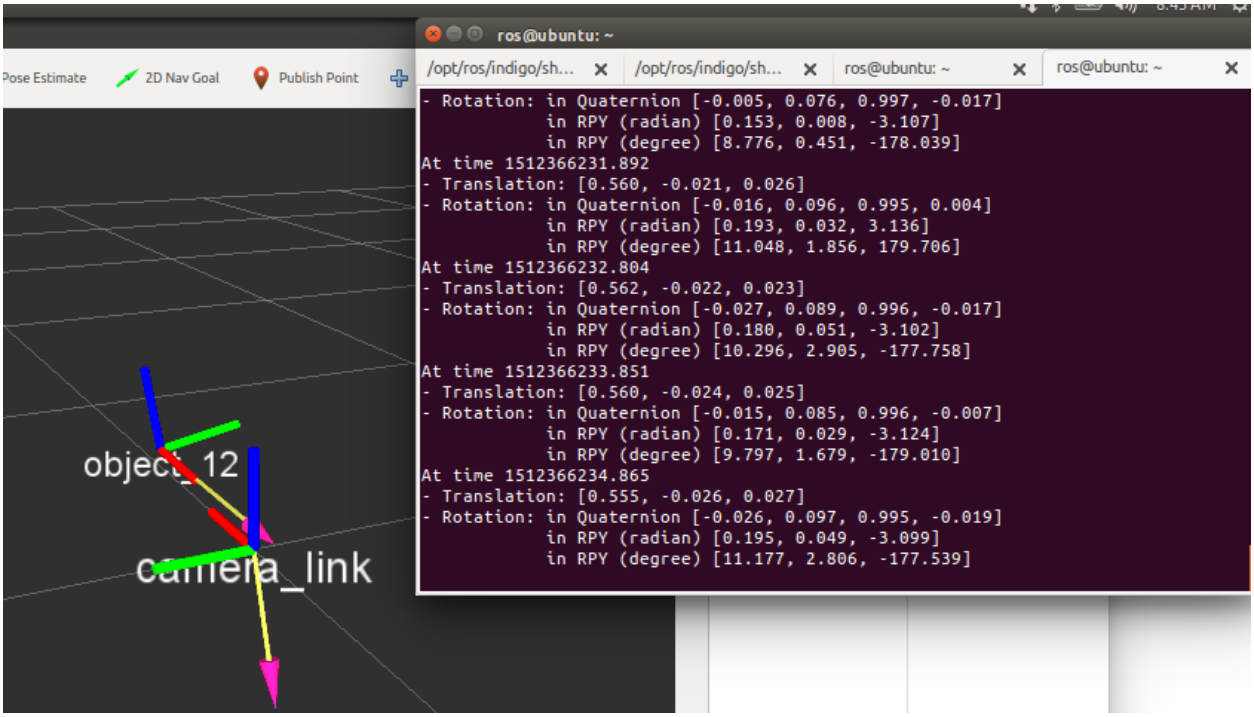

Figure 6: The TF publishing Translation and Rotation information in various formats, which can be subscribe by moveit for manipulation. Use following command to check tf transforms

rosrun tf tf_echo /camera_link object_12

Conclusion:

We have verified that object detection is fairly robust and also object is localized very well. We have used measuring tape to verify the values being published by find_object_3d ROS node.