Sample Return Robot Challenge 2014
Design Proposal

Team: RPIRR (Rensselaer Polytechnic Institute Rock Raiders)

Overview of design

This document contains a brief overview of our design goals as well as relevant descriptions of materials and preliminary design choices. As this is our first year participating, we will be trying to keep our design straight forward. "Rockie" will be a four-wheeled mobile robot with 1-DOF mandibles. Objects will all be stored in the hull, separated by cloth or plastic dividers. An on board computer running ROS will control the autonomous sensing, roving, and object acquisition.

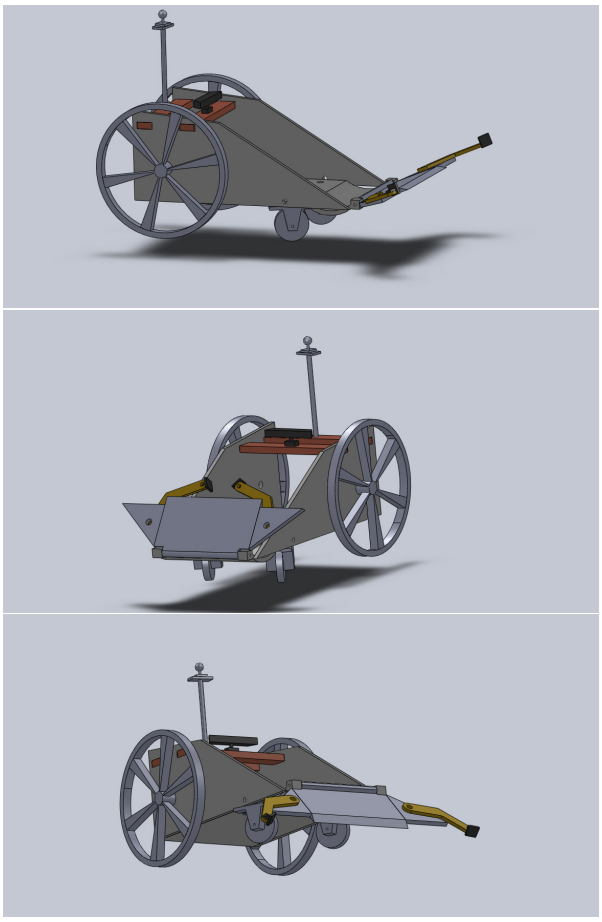


Fig. 1: A sketchup of Rockie.

Hardware and electronics

The only communication we anticipate implementing is a wireless pause switch. For this, we will use an XBee PRO at 2.4 GHz, since it has excellent range and is easy to use.

Rockie will be equipped with at least two cameras:

- A high-definition color camera for long range perception (mounted on the pole in Figure 1)
- A Kinect for point cloud data and SLAM refinement at medium range

All cameras will be able to contribute to object and sample identification. Navigation, including localization and obstacle avoidance, will involve the very popular SLAM approach. In particular, we will utilize the rgbdslam ROS package.

The following is a list of electronic components we will use in addition to the cameras:

- Laptop running ROS
- 2x geared 12V DC motor with H bridges
- Arduino as motor controller
- Deep-cycle lead-acid battery and voltage regulator
- High torque servo or small DC motor for mandibles

We are currently deciding between methods of how to store acquired objects without contamination. Our approach will likely involve scooping the object onto the front of the hull onto sheets of plastic. Each sheet can be drawn up or folded over to make contain it and make room for the next object.

Safety features

Rockie will have a master power switch, mechanical e-stop, and wireless pause switch in accordance with rules R18 and R19. Rockie will also have a safety light to display its state. Currently, we do not plan to provide power to the home beacon. Rockie will contain no hazardous materials as per rule R17.

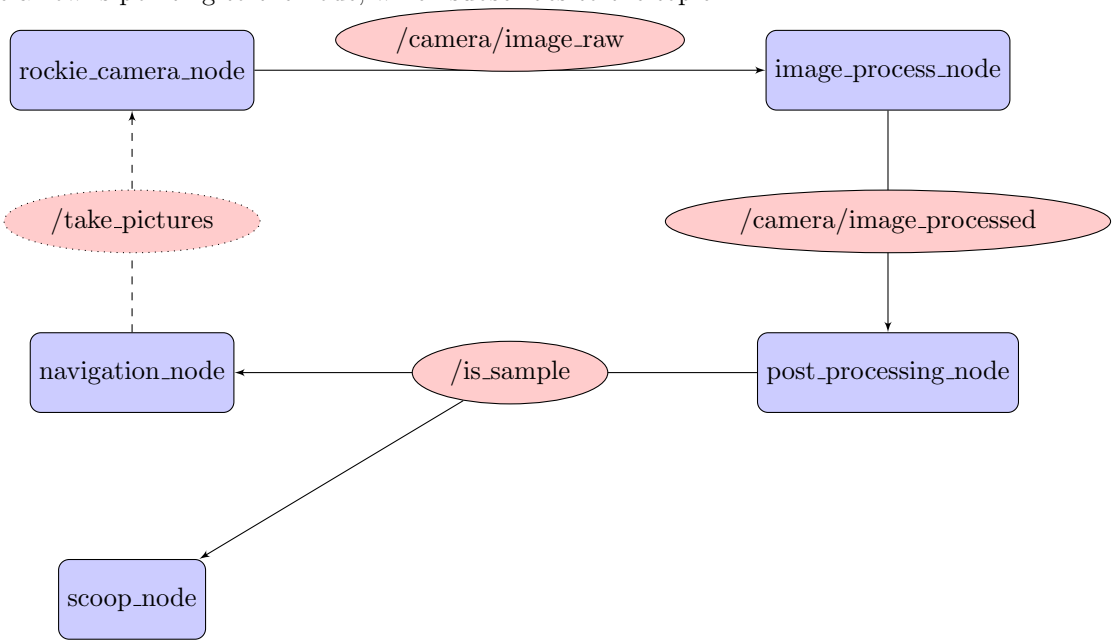
Software

We will use ROS as our software platform since it offers many useful libraries, in particular SLAM, SIFT, and PCL to name a few.

Object identification using OpenCV and ROS

The open-source vision package OpenCV has been around for a long time and there are many available algorithms that we could take advantage to identify the "samples" that we want to pick up. Since our software platform is Robotics Operating System (ROS), we need to set up the interface between ROS and OpenCV. The output from the tracking system provides important information for the SLAM and navigation.

The following is a high-level flow chart of how the nodes communicate using topics. The blue rectangles are the nodes while the red ellipses are the topics. The arrow is pointing to the node, which subscribes to the topic.



We are using the Logitech $C920$ camera. Once the camera takes a picture, the frame will be published as a topic "cameraimage_raw". Since the "image_process_node" subscribes to this topic, once the raw_image is published, it will be processed inside "image_process_node" and a new topic "cameraimage_processed" is published for post processing. Here it is worth to mention that the OpenCV feature detection algorithms are used in the "image_process_node". After the post processing, the robot is confident whether the object is an sample and decides whether or not to pick it up.

The following are some initial object matching results using combination of ROS and OpenCV package: Thick green square means the algorithm is confident to find the cached object in the active frame, while the lines shows the matching between the cached image and the picture from the live frame. Right now this object identification testing is limited to feature and geometry. Colour detection will be added to the package.

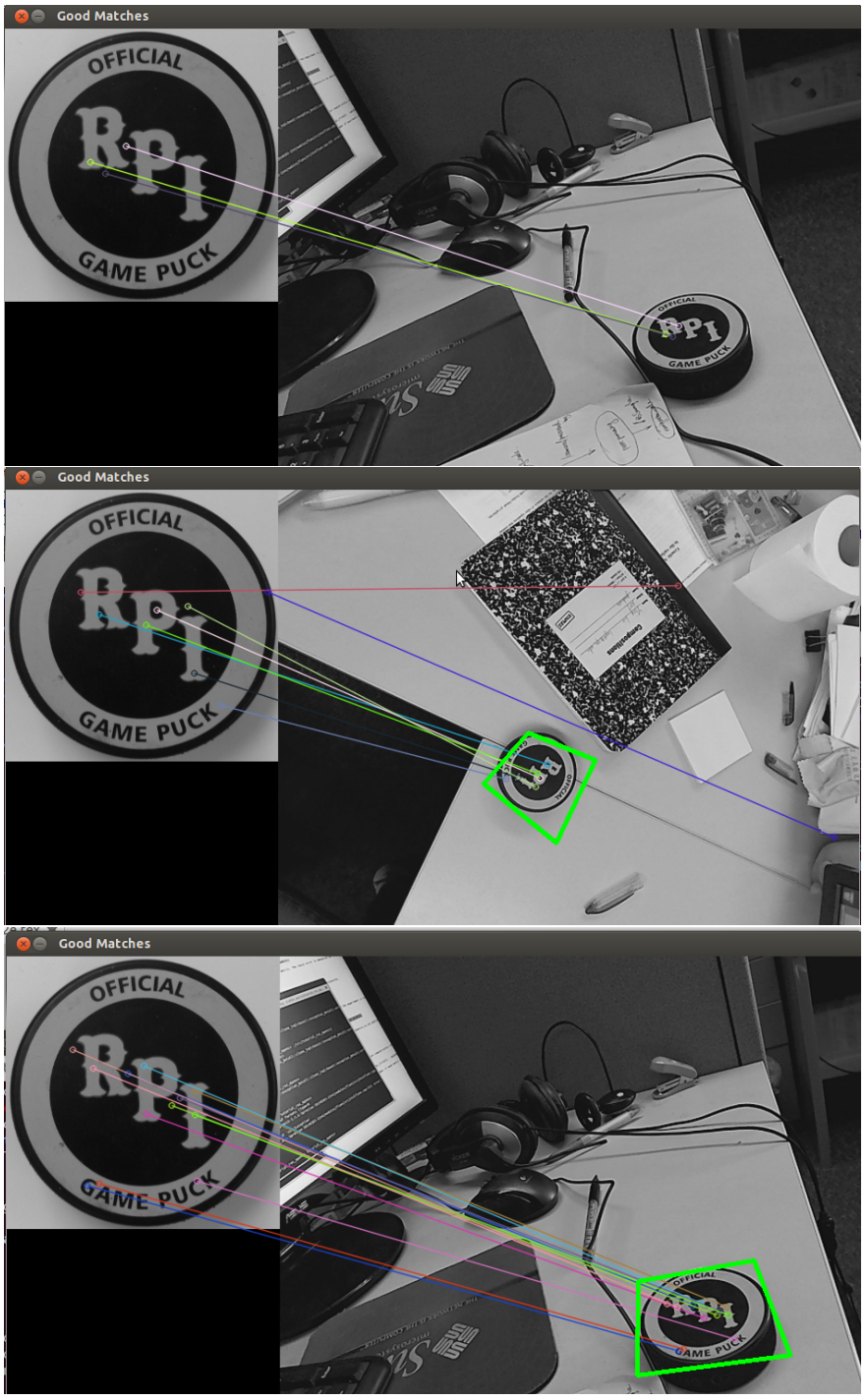


Fig. 2: The left picture is a puck image cached in database, the right one is taken from the web camera.