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 scoop and rake. 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.

Detailed Description of Approach

Upon startup, Rockie will scan surrounding area in attempt to triangulate position with respect to given features. If unsuccessful, Rockie will traverse the course in a manner to expand its map, until a tranform between its map and given map is formed. Upon success, it will travel to precached object, continuing to generate a more detailed map and noting possible samples. Upon arriving at precached approximate location, it will do an expanding search until locating the sample. Using stereo vision, it will use depth location to navigate within range of the object. Then it will lower its scoop, with the rake extended. Upon verifying object within bounding box of the rake and scoop, the rake will retract, pulling the sample onto the scoop. Once verified the sample is on the scoop, the scoop will be raised, dropping the object into the robot, onto a cloth sheet. Verifying the object is within the robot, a motor will raise the sheet, pulling the object further into the robot and creating a barrier between it and further objects. Keeping track of time, the robot will further explore the area, furthering it boundaries, searching for possible samples. Rockie will explore possible candidates and correct as described above. Once time becomes an issue, it will return to the platform using its map and a homing beacon of a large easily identified image.

Hardware and electronics

The only communication we will implement 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 stereo camera mounted near the front of the robot with 2 DOF, pan and tilt, for SLAM and object recognition.
- A webcam for object and capture verification

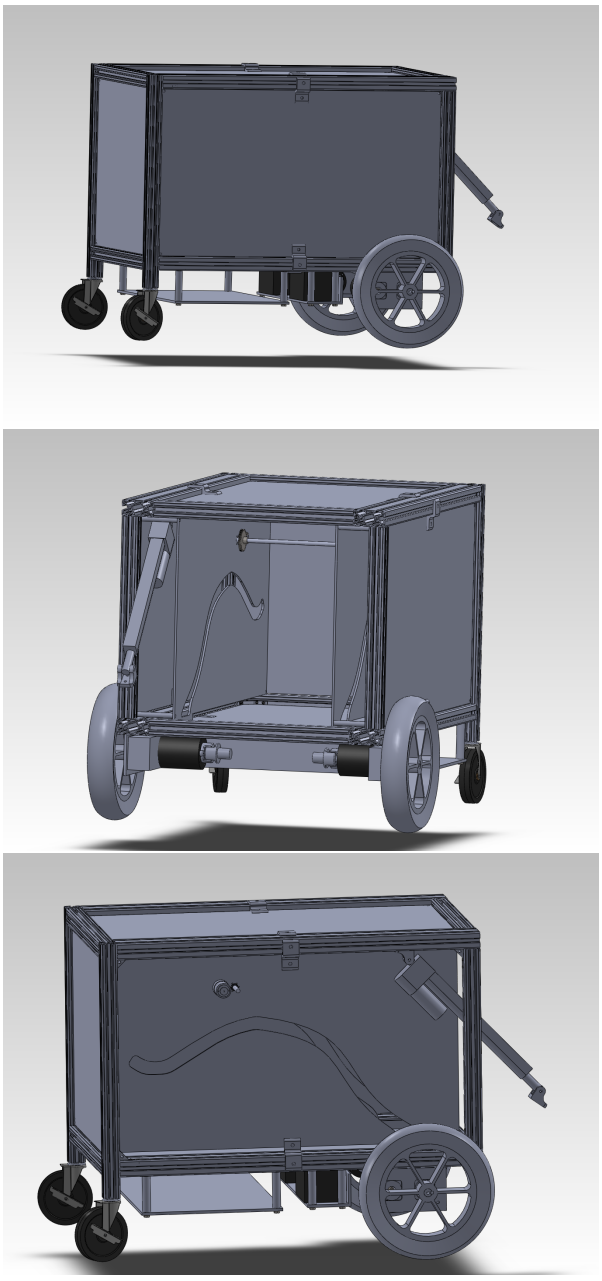


Fig. 1: A sketchup of Rockie.

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.

In addition to cameras, Rockie will have 2 Hokuyo LIDAR (URG-04LX-UG01) for navigation.

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

- Laptop running ROS
- 2x geared 12V DC motors for driving
- Sabertooth Dual 60A 6-30V Regenerative motor driver for drive motors
- 2x 12V linear actuators with L298N H-bridges
- geared 12V DC motor for folder system with L298N H-bridges
- Arduino as motor controller
- Deep-cycle lead-acid battery and voltage regulator

Safety features

Rockie will have a master power switch, mechanical e-stop (see circuit drawing Figure 2), and wireless pause switch in accordance with rules R18 and R19. The pause switch will be a simple remote switch that sends a message over xbee to stop all moving parts, with the exception of a built in cooling fan on the sabertooth motor driver. Rockie will also have a safety light to display its state. 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.

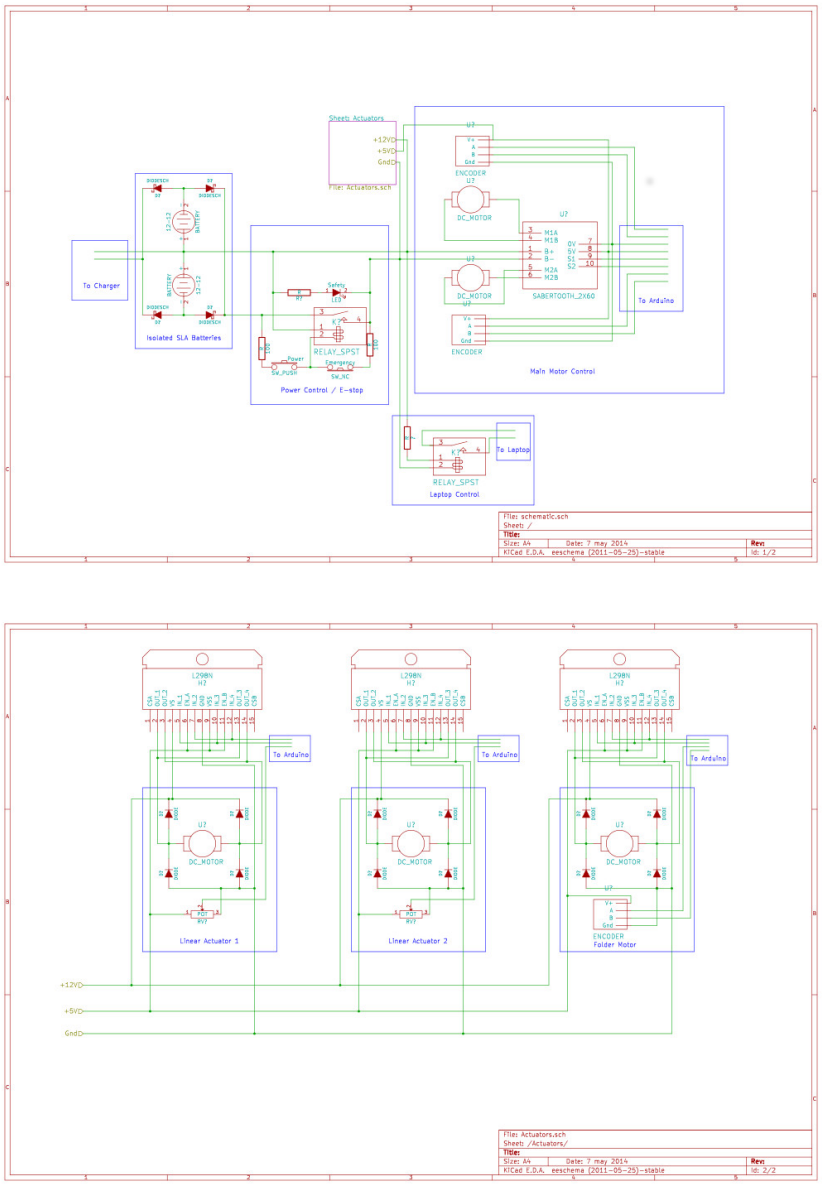
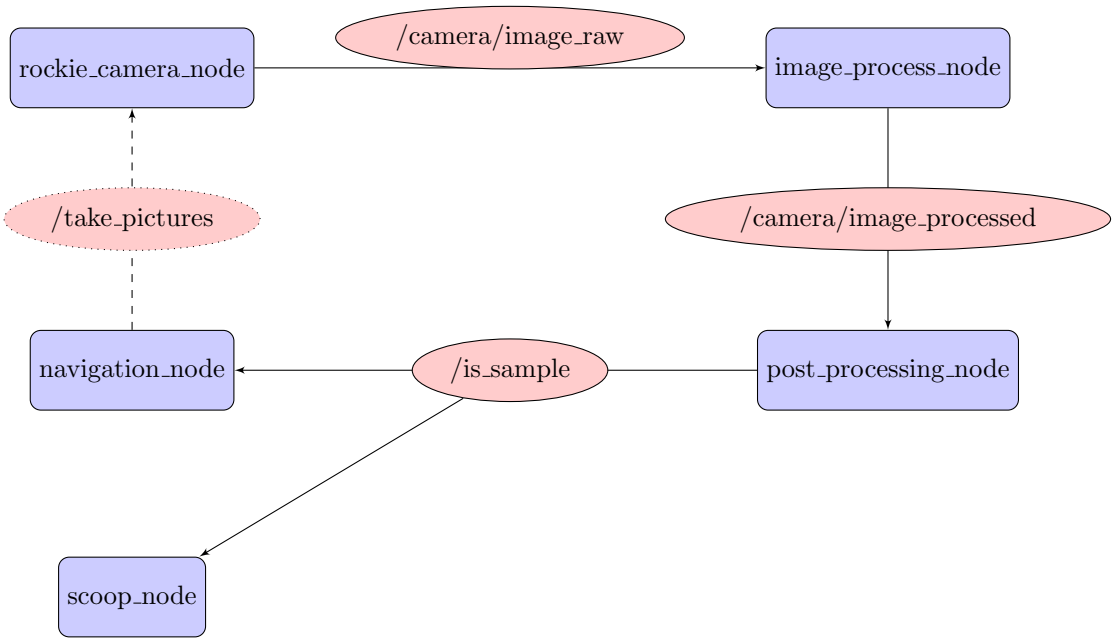


Fig. 2: Circuit diagram of Rockie



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.

Navigation

Most navigation will be done using existing ROS packages. SLAM will be done using RGBD-SLAM package. The map will be flattened to 2D with obstacles generated for navigation purposes. Rockie navigates with the ROS navigation stack. Odometry will be gathered with the visual odometry package viso2. Motor control is done on the arduino with PD velocity controller using encorder values of the motor.

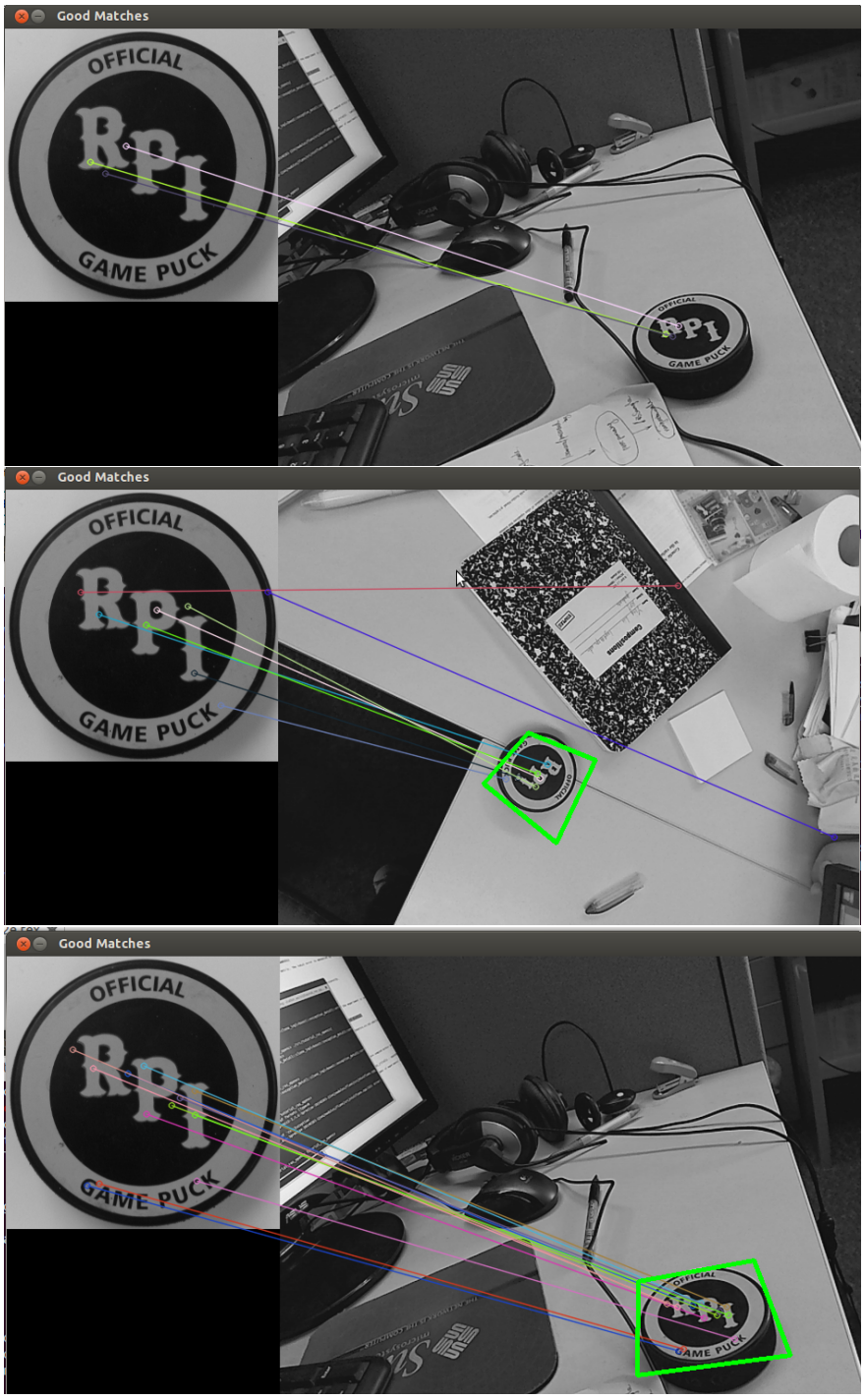


Fig. 3: The left picture is a puck image cached in database, the right one is taken from the web camera.

Future Work

There is a great amount of work we would still like to do before the competition.

Hardware

In hardware, we need to finish the rake and attach it to the scoop. A first level rake has been designed and implemented, but due to miscommunication leading to delayed parts, the is unattachable. In addition, there are some improvements we would like to the initial rake for more secure object capture. Related, we would like to make the scoop a little more rigid and add some basic features for more robust object capture.

The next task in hardware is mounting the sensors. The stereo camera system will be self built, but has yet to be finished. Additionally the extra cameras and lidar system mounts have yet to be built. Also the final electronic mounts need to be completed.

The folder system, while designed, needs to be built.

Electronics

While most circuits are designed, some still need to be wired. In addition, the laptop power needs to be integrated with the E-stop as stated in the design.

The pause controller needs to be built and the related circuits.

Software

The code for exploring and expanding the map has yet to be implemented. In addition the code for approaching the object and capturing it has yet to be written.

The executative code that runs the system is a system state machine. This code has been started, but needs to be finished. As part of this, the different code parts need to be integrated together, as many of them are still separate.

Further, while we have moderate object recognition software, we hope to improve it to be more robust, precise, and accurate.