Udacity Robotics Nanodegree

Project 1: Search and Sample Return with autonomous Rover Robot.

Author / Student: Henrique Carneiro

June/2017

The “NASA Rover Autonomous Rock Picker” simulator is the first project in Udacity’s Robotics Nanodegree course.

The project aims to exercise skills obtained throughout the first block of lessons, focusing in essential elements of robotics: perception, decision making and actuation. Basic idea is: enable a robot-rover to navigate autonomously on a pre-defined landscape, so it can map the area, locate and pick-up golden rocks (“samples”).

Udacity provided students with a basic structure of virtual environment and “skeleton” of functions, programmed in Python language, so that the student would be able to implement new

1 Perception

The perception aspect consists of images captured by the rovers' "camera".

These images would need to be processed by several filtering functions, in order to deal with colors, dimensions & scale, rotation and axis-projection, as well as a very important mapping: determine 3 categories of area that the rover sees: Terrain, Obstacle and Rocks

. Following images exemplify the steps involved:

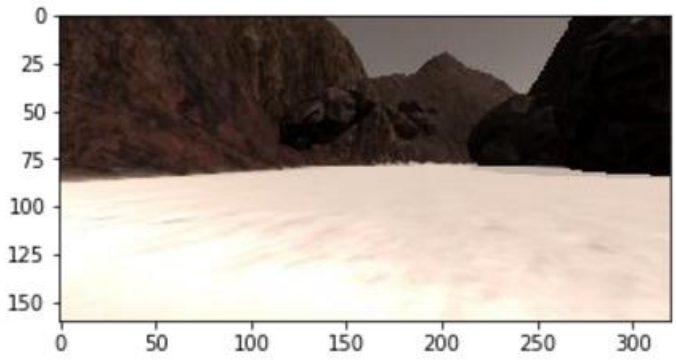

Figure 1 - Random image of terrain

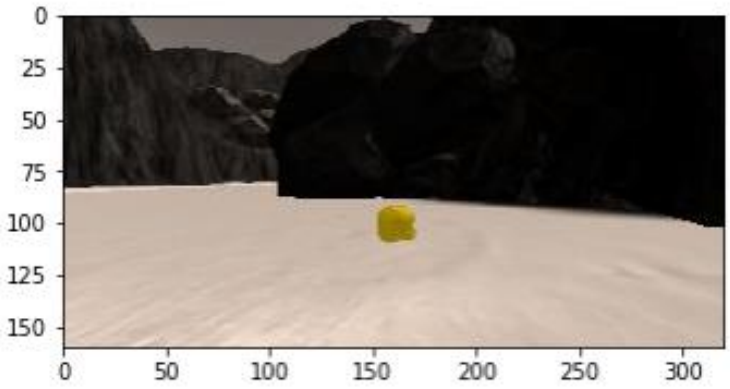

Figure 2 - Example of golden rock ("sample")

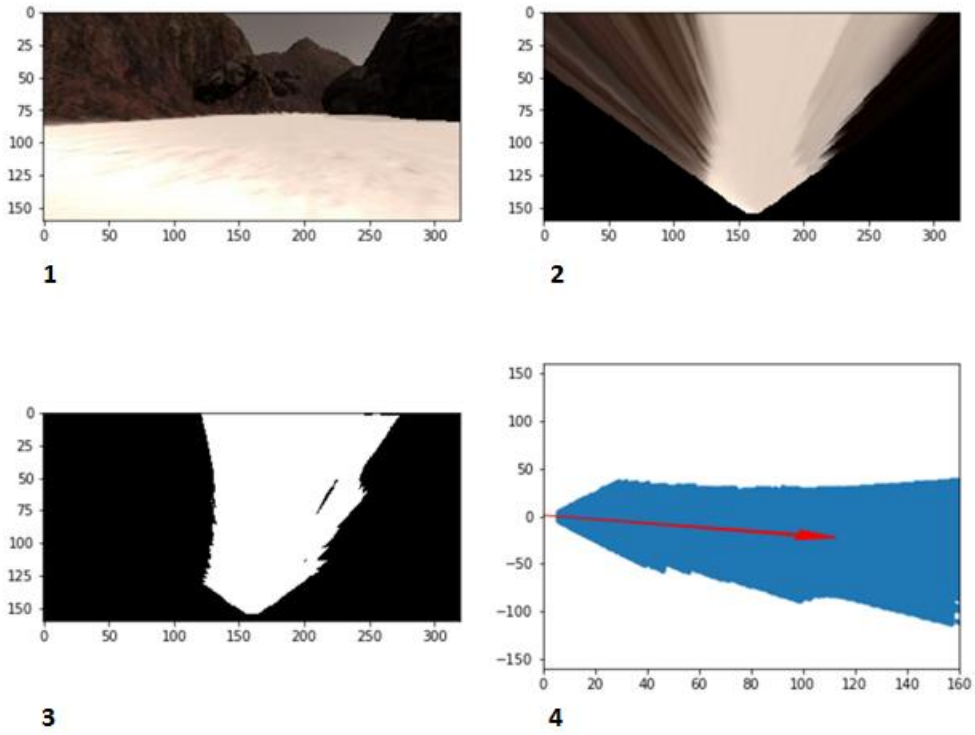

Figure 3 - Treatment of image

Clockwise in Figure 3 - Treatment of image:

1. Original image
2. Perspective transform
3. Color threshold: this is used to determine 3 categories of area that the rover sees: Terrain, Obstacle and Rocks
4. Change of coordinates: modify coordinate system of image to fit the rover's orientation. The red arrow represents the mean angle to steer the rover and provide a possible direction of movement. This angle is calculated considering the possible points (blue area) where the rover could move to.

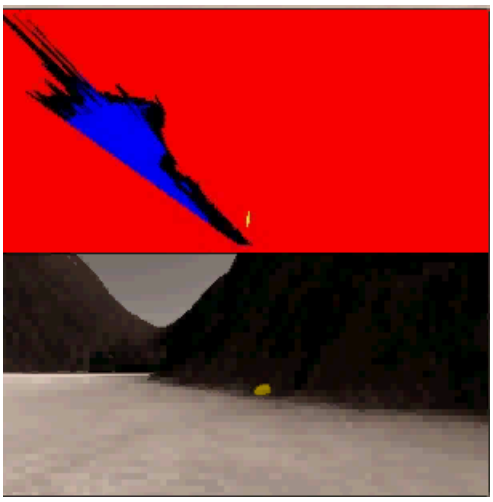

Figure 4 - Original X treated image

It is possible to see in Figure 4 - Original X treated image the original image of a golden rock and the treated version, showing obstacle (red), terrain (blue) and rock (yellowish).

2 Decision-Making and Actuation

Rover must be able to "decide" what to do in face of the inputs it receives via imagery and telemetry sensors (velocity, angular movement, etc.).

After deciding what to do, Rover will "actuate" according to it. This is the step will work as the "feed-back" to enable a closed loop of decision-and-actuation.

In short, the decision-actuation process involves a loop of steps:

1. Obtain image
2. Treat image (as in 1Perception)
    a. As result of image treatment, rover receives an updated set of coordinates and direction where it could move towards to.
3. Based on the possible "place and direction to go" and on its current state (moving, stopped of stuck), Rover will decide about:
    a. Keep moving
        i. On same direction (forward, backward)
        ii. On other direction / angle
        iii. Spin on its current place
    b. Stop moving
    c. Spin (to try to get out of a stuck situation)

    d. Pick up rock sample
4. Rover will obtain a new “state” of itself based on:
    a. its current positon, speed
    b. its status (“going forward”, “stopped” or even “stuck”)
    c. it sensorial data and commands to read/set values for controls such as “steering angle”, “throttle value”, “brake value” and “speed value”
5. Update statistics about:
    a. % Mapped Surface
    b. # Rocks found

3 Conclusion and further steps

My implementation of new functions and to the Rover fits the goals of this project, which were:

- to map at least 40% of the environment at 60% fidelity
- to locate at least one of the rock samples
- to pick up rock sample

Files “perception.py”, “decision.py”, “drive_rover.py”, “supporting_functions.py” and “environment.yml” can be found in my Github repository:

https://github.com/HenriqueCarneiro/RoboND-Rover-Project/tree/master/code

Although meeting the minimal requirements of challenge, this implementation could benefit of further analysis and development, such as:

- Vary the angle of steering of wheels
- New way of deciding calibration of colors of images
- Make rover return to its original point after all rock sample have been collected
- Create parameters to make rover stop operating (example: after collecting 80% of available samples, or after mapping 90% of the area)