Individual Lab Report 10

Songjie Zhong

16-682 MRSD Project 2

Team B: Space Robot

(Brian Boyle, Nathaniel Chapman, Songjie Zhong ,

Ardya Dipta Nandaviri)

April 4th, 2014

Individual Progress

During this week, I have two responsibilities. One is to figure out how to create a CV_Bridge ROS package to have a ROS node that can get the ROS images (see figure 1) and then convert them to OpenCV images. The reason for doing this one is that we can’t do image comparison directly on the ROS images from our wireless camera through ROS. All the OpenCV realted operations or commands require images to be OpenCV images.

The other one is to create a TF ROS package. This package has two ROS nodes. The ROS node named matrix_changing can get the information from the april_tags ROS node. As we mentioned in previews demo, our april_tags ROS node could get the ID number of the April tag, distance, translation vector and Euler angles. The matrix_changing ROS node is to get the translation vector and Euler angles, then convert these values to translation and quaternion vectors. After this, the other ROS node named tf_frame is to use parent frame’s information, translation and quaternion vector to get the child frame (see figure 2).

Challenges/Issues

The most difficulty part for me was to figure out how to understand the online resources for the related ROS packages that I needed. Both Dipta and me found that some packages online were either not so correct or so specific for their own situations. So we need to modify and create new packages by ourselves. It took me a long time to understand the whole process of how to create the ROS package, create the ROS node, and match this ROS node to the whole ROS environment.

For the TF ROS package, I found myself difficult in understanding the quaternion vector The original transform ROS package was using translation and quaternion vectors to get the transform format message (see figure 2). Then I asked Nate for help, and we finally found that there was a function online to change the Euler angles to quaternion vector.

For the CV_Bridge ROS package, Dipta and I were working together for the pipeline. Our wireless camera brought lots of inconvenience to us since the online instruction of how to use this camera was few and not quite right. It took him long time to figure out our wireless camera to get the ROS images in ROS. He needed to create the new ROS package for our wireless camera and until the morning before the demo he successfully got the results. Since the limitation of time, we didn't assemble our codes together for the demo this time.

Cross-reference/Teamwork

Brian was working on the iterations of the magnetic foot system. He adjusted the design in order to have a better integration to our chassis and make it more stable for the attachment and detachment. Nate was still working on the new path planner that has a better GUI. He also helped Brian together to design and manufacture the new version of our magnetic foot system.

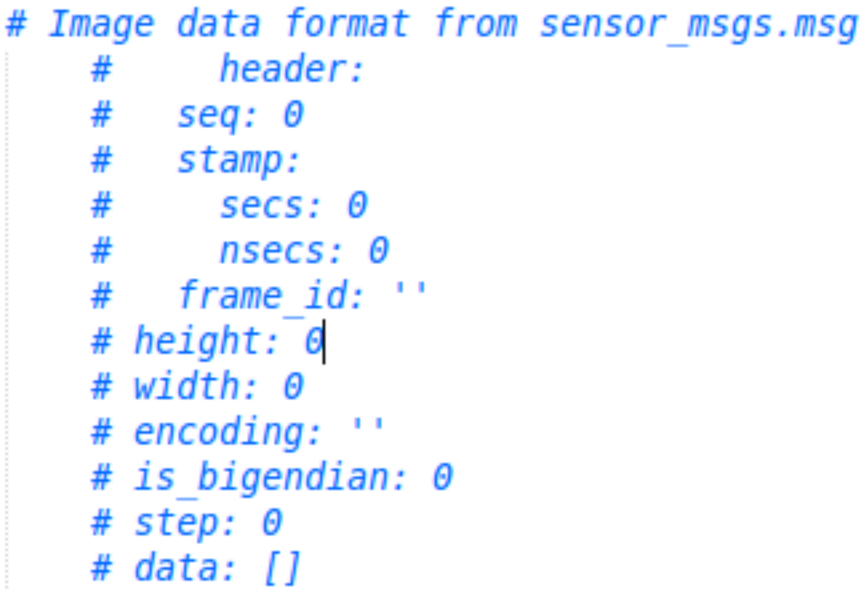
Dipta helped Nate figure out to add IMU data into ROS. He also was working together with me for the CV_Bridge for ROS. CV_bridge was a ROS package that we modified by using online resources. It had two ROS nodes. One is to convert ROS images to OpenCV images and the other one was to convert OpenCV images to ROS images. His responsibility was to integrate our wireless camera to ROS that could convert ROS images to OpenCV images. I got the results from his side to do the image comparison based on these OpenCV images.

Plans/Future Work

I will work together with Nate to figure out all the frames in our whole system. He was doing inverse kinematic calculations of our robot last week. I was working on our wireless camera's frame. I figured out how to use wireless camera to detect April tags to get the translation and Euler angles. TF ROS package helped to use parent frame to get the child frame by using translation and quaternions vectors. We will integrate our codes together to get all the frames' information for real time.

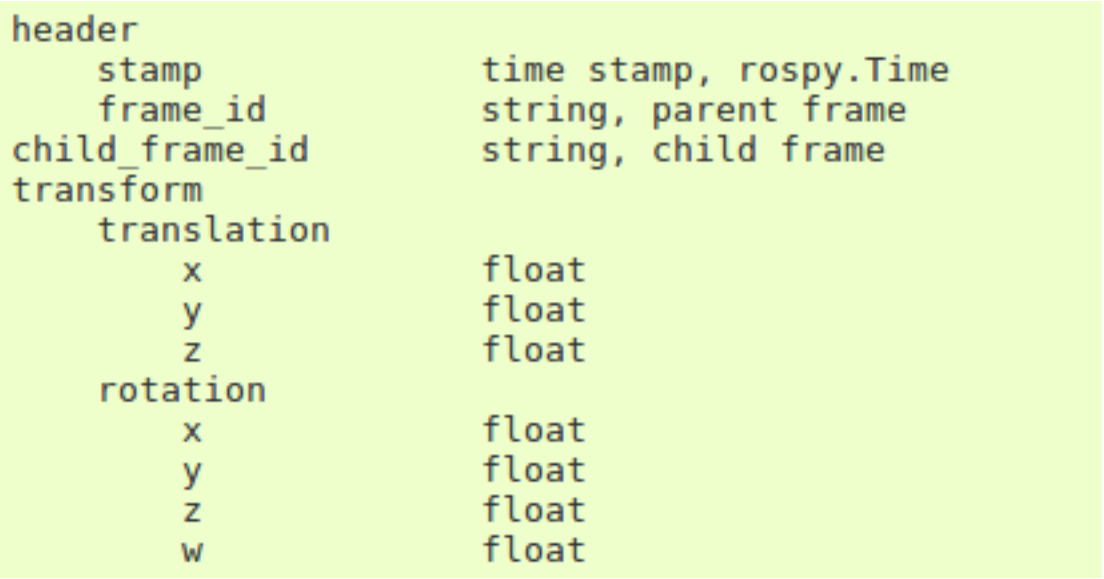
On the other side, I will keep continue working on the image comparison part together with Dipta. We will first assemble our codes, ROS nodes and have a test for the whole process for this pipeline. We will test this by all ROS nodes from very beginning to catch the camera input views by using our wireless camera, change ROS images to OpenCV images, do image comparison by working on these OpenCV images, show bonding boxes for the results of our Inspection for the test surface.

Figures



```
# Image data format from sensor_msgs.msg
    #     header:
    #   seq: 0
    #   stamp:
    #     secs: 0
    #     nsecs: 0
    #   frame_id: ''
    # height: 0
    # width: 0
    # encoding: ''
    # is_bigendian: 0
    # step: 0
    # data: []
```

Figure1: the ROS images format



```
header
    stamp               time stamp, rospy.Time
    frame_id            string, parent frame
child_frame_id          string, child frame
transform
    translation
        x               float
        y               float
        z               float
    rotation
        x               float
        y               float
        z               float
        w               float
```

Figure2: the frame transform format