Here is a sample VRPN code used to communicate with Optitrack through ROS:

Now with VRPN client we needed to write a launch file:

****

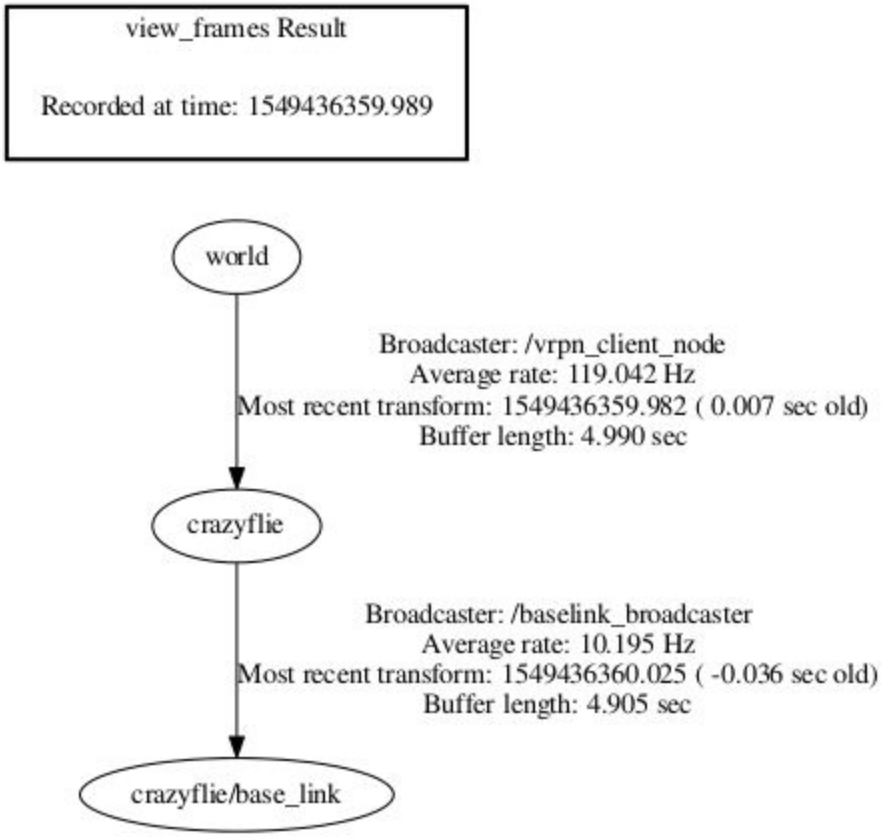
```
<launch>

  <arg name="server" default="192.168.1.2"/>

  <node pkg="vrpn_client_ros" type="vrpn_client_node" name="vrpn_client_node"
output="screen">
        <rosparam subst_value="true">
        server: $(arg server)
        port: 3883

        update_frequency: 100.0
        frame_id: world

        # Use the VRPN server's time, or the client's ROS time.
        use_server_time: true
        broadcast_tf: true

        # Must either specify refresh frequency > 0.0, or a list of trackers to create
        refresh_tracker_frequency: 1.0
        #trackers:
        #- cf

        </rosparam>
  </node>

</launch>
```

****

Make sure the configuration of your nodes is as follows: (You may check it using Rostopics)
Rostopic list
Two topics : crazyflie node and world frame
Which can be added to rviz for demonstration :

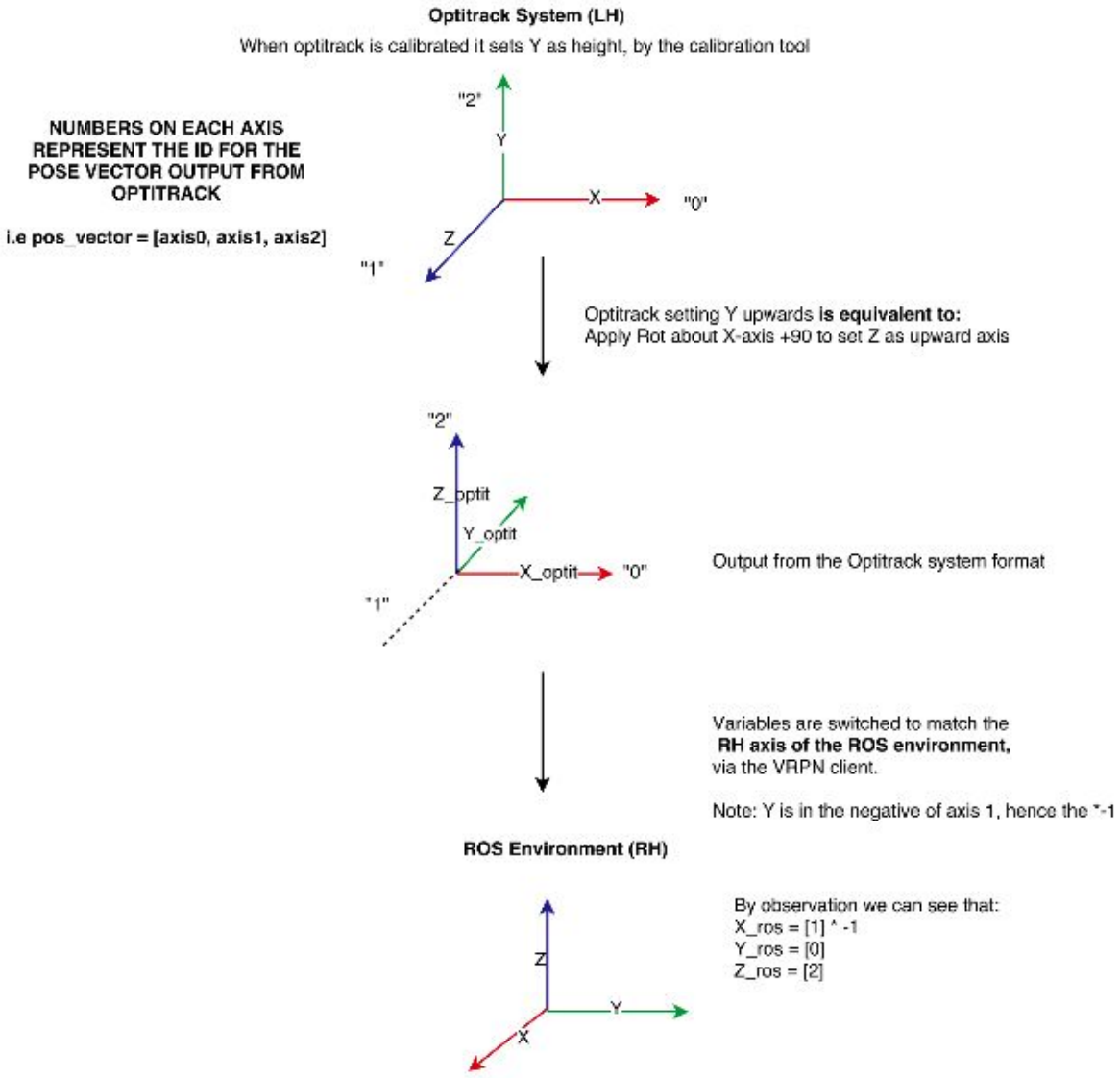
Run rviz rviz -f frame_name

In order to check whether it's working or not we can check tf frames and we got below picture where cf stands for our crazyflie drone.
Sample launch file :



Another important consideration here is to adjust the coordinate systems. Coming from Optitrack to ROS there are some changes to consider. Optitrack coordinate system has y UP

whereas in ROS it's Z up. So not only coordinates but also the position and acceleration values needs to get swapped. Do the changes in VRPN .cpp file. (Here's the link to our gitlab account:). Here is a picture I found online to better show the changes applied:



In previous posts I talked about how the bright, shiny surface of our floor interfered with data collection of Optitrack and I also mentioned how to put each of the cameras in Grayscale mode to better see the area with high error rate. Try to move your origin away from those shiny area

(by refining the ground plane in Optitrack Calibration). After calibrating the cameras and adjusting the ambient light put at least 4 markers on the crazyfile as it would make it easier to be detected (Also glue them to make it easier for later on). Markers should be spheres otherwise sometimes there are not detectable by the cameras.

Finally It's important to activate the kalman filtering on the crazyflie so that it would use the position estimation for the purpose of set point flying or hovering. In order to modify the crazyflie firmware to include the kalman filter I guess the easiest way would be to use the Virtual machine provided by bitcraze. Here are the links to how to set up the virtual machine and how to activate the kalman filter and flash/update the firmware.

- Installing vm: https://www.bitcraze.io/getting-started-with-the-crazyflie-2-0/#inst-virtualmachine

- How to activate the Kalman Filter:
  https://forum.bitcraze.io/viewtopic.php?t=3068

After that you may run the hver_vrpn.launch file updating the VRPN section of it (as we are using vrpn_ros_client) and you should get sth like this::
https://youtu.be/reBzRPjIRBA