lab12-Ar_tracker

Prajwal

2022-11-20

aruco marker

- Will use aruco marker
- use drone camera to scan the AR marker
- will use standard ar marker package to detect the ar marker
- src > bebop_real_nav_packages > get_pointcloud > launch > ar_tracking.launch

```
<?xml version="1.0"?>
<launch>
    <!-- <include file="$(find ar_tracking)/launch/usb_cam.launch" /> -->
    <include file="$(find ar_track_alvar)/launch/pr2_indiv_no_kinect.launch" >
        <arg name="cam_image_topic" value="" />
        <arg name="cam_info_topic" value=" "/>
        <arg name="output_frame" value="" />
        <arg name="marker_size" default="15.5" />
    </include>

</launch>
```

land on the Ar marker ?

- if you launch the lab10 , and see tf frames
- command : rosrun rqt_tf_tree rqt_tf_tree .
- check figure.1 , you will not see camera_optical frame for now .
- we can see the transformation from base_footprint to Bebop . using command or through python script
- command : rosrun tf tf_echo parent_frame child_frame
  - parent frame : base_footprint
  - child frame : Bebop
- Above command shows the Pose of child_frame with respect to Parent_frame
- through script : check online , link : see ros_wiki_tf
- **Big picture** & **lab-12**
  - create a static transformation from Bebop (Bebop frame is approximately in the center of Bebop) to the Bebop Camera . fig1
  -
  - Once Ar marker get detected , A new frame will get added to the tf tree
  - Echo the transformation from world/odom frame to ar_marker_5 frame (This means we have the position of the ar marker with respect to world frame)
  - This position will act as a goal for your drone
  - check how to send goal command from python script instead of rviz
  - once you reach the specified x,y and orientation , land .

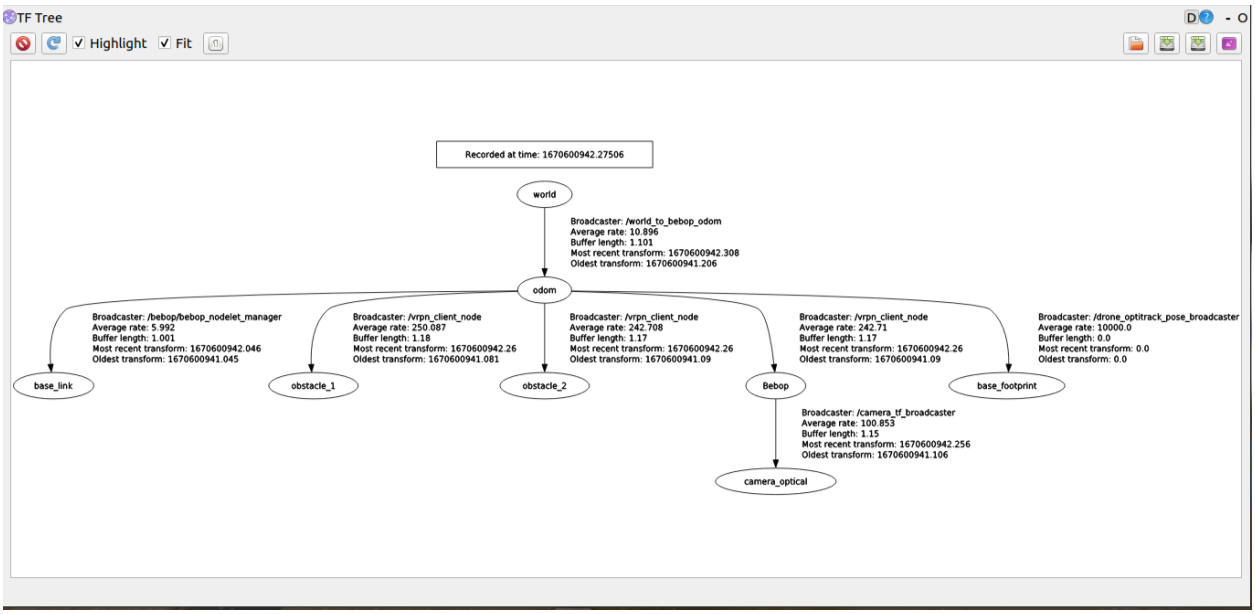


Figure 1: tf_tree_ar

- you might need to control the camera to detect the ar marker
- command:
- ```
  prajwal20@prajwal20:~$ rostopic pub /bebop/camera_control geometry_msgs/Twist "linear:
    x: 0.0
    y: 0.0
    z: 0.0
  angular:
    x: 0.0
    y: 0.0
    z: 0.0"
  ```

lab 12 Instructions

AR with Drone Camera

Goal :

- Detect the Ar marker pose & pass it to Navigation Stack ,
- Few things to remember :
  - That Drone Camera Can detect multiple multiple Ar tags
  - Once you detect the desired Ar marker ,you could make the pose of the marker static so that goal position being send to navigation stack doesn't change very frequently .

Recommended method:

Part-1

- you can do the following setup in lab10
  ```

- copy ar_track_alvar package and ar_tracking.launch file
- In /bebop_driver/bebop_with_vel_controller.launch the default value of argument camera_info_url should be “file://« full path to bebop_driver package in your workspace »/data/bebop1_camera_calib.yaml” />
- do not remove the word : file:// , append the full path after this word .
- for example my camera_info_url is like this : “file:///home/prajwal/catkin_ws/src/bebop_driver/data/bebop1_camera_calib.yaml”
- build your workspace
- launch your main launch file and then rqt_tf_tree (command : rosrun rqt_tf_tree rqt_tf_tree) to visualize the tf_tree , you should see something like fig1
- include the ar_tracking.launch file in your main launch file
- in ar_tracking.launch file fill in the values of cam_image_topic,cam_info_topic & output_frame (for more information about this args check ar_track_alvar ros wiki)
- ~~In your main launch file create a static transform from Bebop to camera_optical only having 0.1 m displacement in x direction and -pi/2 roll and -pi/2 yaw axis rotation.~~ See correction , delete this transformation from your main launch file .
- launch your main launch file configure it according to the figure 2(add topics to visualize) , also check the tf_tree (note: in figure 2 and fig 3 , tf tree is slightly incorrect . odom should be connected to world)

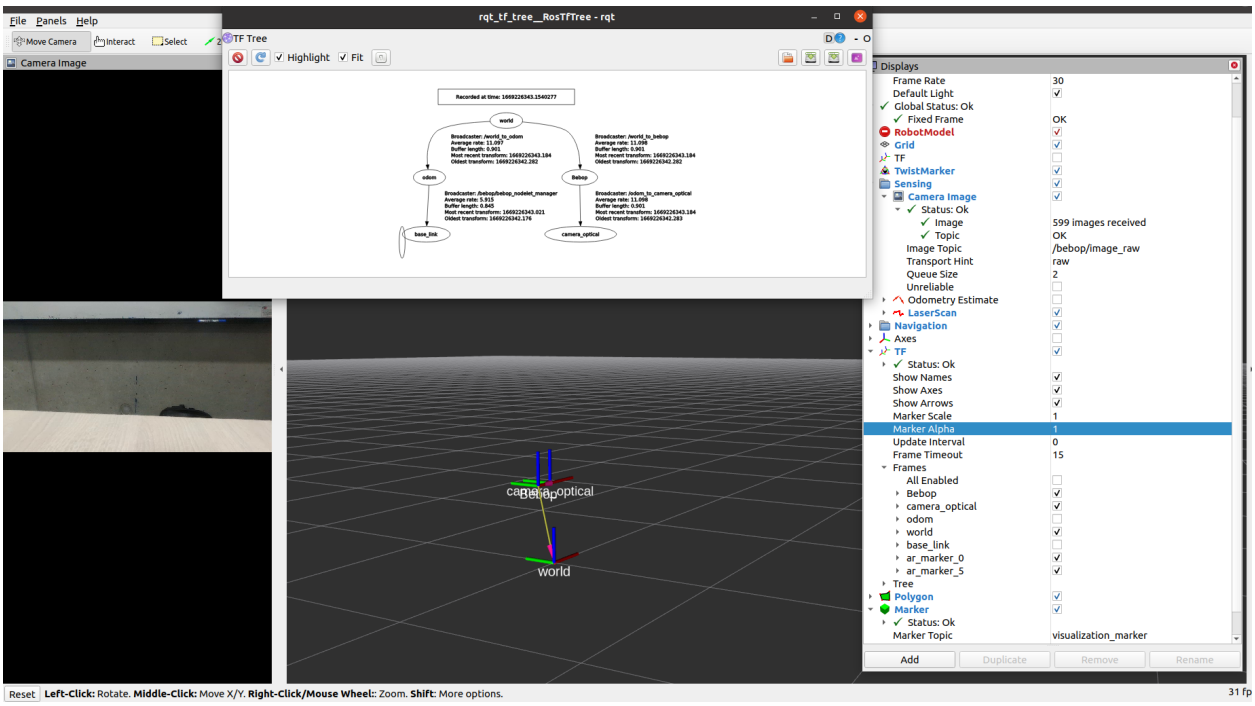


Figure 2: rviz_without_warker

-
- if your camera detects the AR tag , you could see the new link appeared in tf_tree as well as in rviz , see following figure

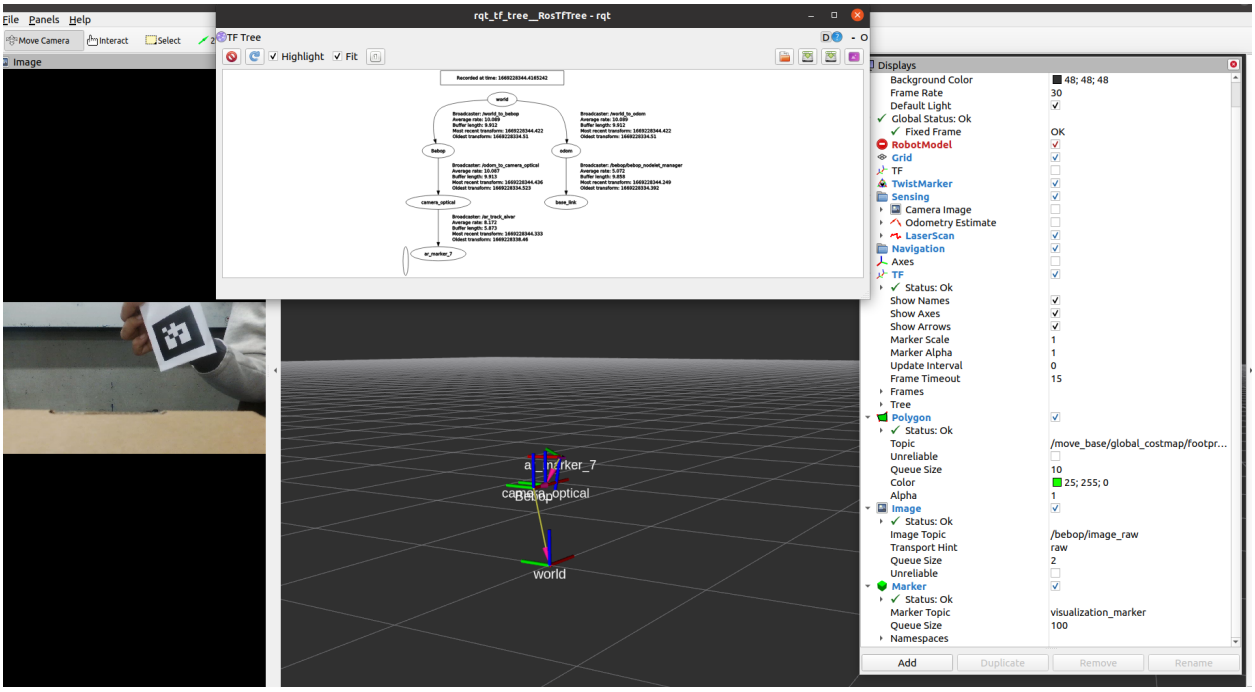


Figure 3: rviz_with_warker

-
-

Part2

- Write A script to receive the ar_track_message and create a publisher to pass the pose of the desired-marker with respect to odom frame to rviz
- the idea is to filter out the marker you want to go, from the list of the markers that are visible and getting detected by the camera .
- tips:
- check the ros topic /ar_pose_marker for more information
- check the link ar_track_alvar_message

Part3 (optional)

- Write a script which subscribe the message from part-2 as goal and pass this information to navigation stack

Note

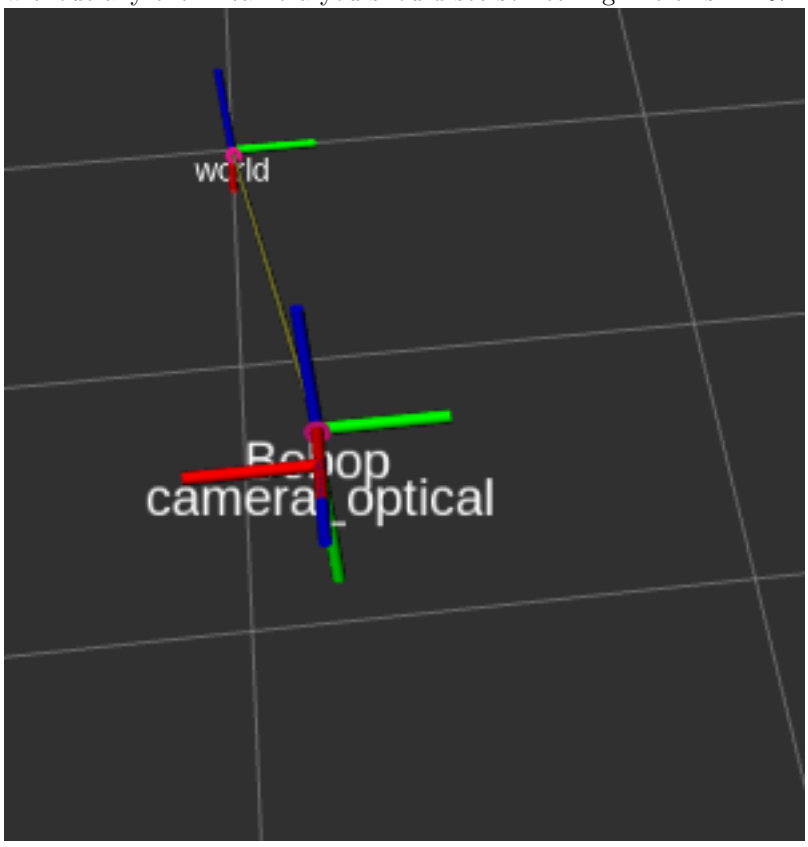
Note : Above mention method is a suggested method , You are free to explore other methods to achieve the same . Lab-12 will be graded based on how well you are able to detect the ar tag (Part-1) and if are able to extract the information from AR tag (part-2)

For grading

- Submit the your package which includes the launch files and scrips for lab-12

- Explain briefly your method
- Demonstrate the method

Mandatory Correction for Part1

- To do :
- ~~In your main launch file create a static transform from Bebop to camera_optical only having 0.1 m displacement in x direction and -pi/2 roll and -pi/2 yaw axis rotation.~~ delete this transformation from launch file
- Idea is to create a dynamic transformation from bebop to optical_camera Instead of static transformation to get the almost accurate ar_marker position with respect to the bebop/world/odom
- copy the bebop_camera_transformation.py (provided) in your main launch file
- complete the TODO's in bebop_camera_transformation.py (provided)
- without any tilt in camera you should see something like this in RViz
- 

important points to consider :

- 1. check the marker_size parameter in ar_tracking.launch, it should match the size of the ar_marker that you want to detect . during competition, we might change the size of the marker. so remember this point!

- 2 . remember to always check the units of msg. for example, you pass angular displacement in degrees to /bebop/camera_control, not in radians and tf.transformations require angular displacement in radians
- for more information bout bebop camera , you can serach online and vist the link : bebop_driver

Regarding deltaX competition

- There will be still an error in getting orientation of ar_marker , this is mainly because of opti track system (we will remove the orientation constraint from the the Delta-X competition .)
  - This means you are free to reach/land on ar_marker in any orientation .