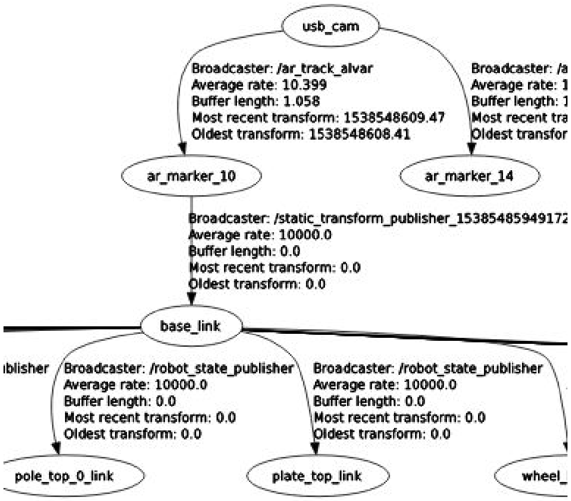
Here’s how we’ve gotten (most) turtlebots working with AR tracking (note: the ar marker numbers for the ar tags on the turtlebots are written on the backside of the tags):

1. Create `lab6` workspace
2. Run `git clone https://github.com/ucb-ee106/ar_track_alvar.git` inside the `src` directory to get the package
3. Download `ARTag_Resources.zip` folder from under Lab 4 on bCourses, extract the folder and copy the files inside it to the new `ar_track_alvar/launch` directory (there should be no extra folders inside this directory, only launch files)
4. Modify `webcam_track.launch` to have only the Logitech parameters uncommented, make sure the `camera_info_url` param has `value=″file://$(find ar_track_alvar)/launch/usb_cam.yml″`
5. Change the `marker_size` to be `17.7` in order to account for using the larger AR Tags
6. Run `catkin_make` in the new workspace to build the `ar_track_alvar` package
7. Add the line export `ROS_MASTER_URI=http://[TurtleID].local:11311` to your `~/.bashrc` file, making sure to replace `[TurtleID]` with the color of your turtlebot (ie `red, blue, black, yellow, green`)
8. Make sure the turtlebot is turned on (switch on the base, a green light should light up while on)
9. In terminal, run `ssh turtlebot@[TurtleID].local` (password is `EE106A18`) again replacing `[TurtleID]` with the proper name. (if you are not able to log in, go into your `~/.bashrc` and replace the line `export ROS_HOSTNAME=$(hostname --short).local` with `export ROS_HOSTNAME=192.168.1.[COMPUTER_NUMBER]` where `[COMPUTER_NUMBER]` is your computer’s number, value 1 through 10. Make sure to `source ~/.bashrc` again before continuing)
    a. If errors are thrown about `std::runtime_error` upon logging in, run `export LC_ALL=″C″` while still logged in. Note that you’ll have to re-run this command any time you log into the turtlebot
10. In the same terminal, while still logged in, run `roslaunch turtlebot_bringup minimal.launch --screen` and do not kill this window. The turtlebot should beep if this worked
11. In a new terminal (without logging in to the turtlebot) run `roslaunch ar_track_alvar webcam_track.launch` and make sure that you have sourced your lab 6 workspace
12. In another new terminal (again without logging in to the turtlebot) run the command `rosrun tf static_transform_publisher [args]` where you will replace `[args]` with the appropriate arguments to this function (you’ll have to look it up! There’s a hint in the lab manual)
13. In yet another terminal on your local machine, run `roslaunch turtlebot_rviz_launchers view_navigation.launch`
    a. You may have to run more or different launch files to get full functionality working for checkoffs, this document is meant for getting the basic ar tracking with turtlebots running

14. In the Rviz window that opens, change Global Options > Fixed Frame to `base_link`, then modify the view type to `ThirdPersonFollower (rviz)`. This option should be in a panel at the top right corner of the screen
15. Add a camera in the same way you did in lab 4 and set the image topic to be `/usb_cam/image_raw`
    a. If you do not get any images, make sure the camera has a clear view of the ar tag on top of your turtlebot, otherwise the static transform can't be published and this will block the image from being displayed (you might have to restart rviz/webcam_track.launch with an unobstructed view)
16. Add an Axis and set the frame to be `ar_marker_[NUM]` where `[NUM]` is the ar marker number for the marker on top of your turtlebot (this number should be written on the backside of the tag itself; you might have to look for it a bit). At this point, you should have a working camera display in Rviz with axes displayed on the AR tag on your turtlebot. There should also be a robot model and axis display in rviz itself. You should be able to move the robot around and see that the ar tag is tracked properly. Be careful not to obstruct the camera's view of the ar tag, otherwise images won't be displayed properly.
    a. To make sure the static_transform_publisher is working properly and the camera can see the ar tag, run `rosrun rqt_tf_tree rqt_tf_tree` on your local machine. You should see something like this in the middle of the tree:



    b. The `usb_cam` frame should be the parent of the `ar_marker_[NUM]` frame on your turtlebot (`ar_maker_10` in this case) and this frame should be the parent of `base_link`. If this is not the case, either the camera is obstructed or the `static_transform_publisher` command is incorrect