ROS: Tracking AR tags with USB camera

The objective of this lab is to learn how to use camera in ROS for tracking AR tags.

Setting up your computer

Open terminal window.

Clone your git repository to your home folder:

```
$ git clone URL_for_<yourname>-rtech_repository
```

Use **ls** to confirm that ***<yourname>-rtech*** has been downloaded to your home folder***.***

Mounting a Camera on the Robot

In this lab you are going to:

(1) Install ROS package for USB cameras
(2) Calibrate the camera for distance sensing
(3) Install ROS package for tracking AR tags
(4) Setup launch-files for launching USB camera and AR tracking
(5) Use RViz for displaying camera images
(6) Update R2D2's description so that its lense link acts as the base for the camera

(1) Connect the web camera to your computer and if your computer does not have usb_cam ROS package installed, install it by using apt-get as follows:

```
$ sudo apt-get install ros-kinetic-usb-cam
```

In order to confirm that usb_cam has been properly installed, run:

```
$ roslaunch usb_cam usb_cam-test.launch
```

What do you see? Do you see the image from the recently connected Logitech USB camera? If you would like to change anything in usb_cam-test.launch, first copy the file to an *appropriate* place in your catkin workspace.

(2) Next, you need to calibrate your USB camera. We will use the steps given here: http://wiki.ros.org/camera_calibration/Tutorials/MonocularCalibration

In short:

1) Use rostopic list to see what is the namespace for your camera.
2) Then run the camera calibration node. Make sure to use the correct topic name and namespace.

```
$ rosrun camera_calibration cameracalibrator.py --size 8x6 --square 
0.036 image:=/usb_cam/image_raw camera:=/usb_cam
```

3) Move the 8-by-6 grid in front of the camera until you can press buttons for *calibrate*, *save*, and *commit*.

(3) Now, if your computer does not have ar_track_alvar ROS package installed, install it by using apt-get as follows:

```
$ sudo apt-get install ros-kinetic-ar-track-alvar
```

(4) Create a new launch-file for `my_r2d2` ROS package and name it `ar_tracker.launch`.

Copy the contents of `usb_cam-test.launch` (from `usb_cam`) and `pr2_indiv_no_kinect.launch` (from `ar_track_alvar`) into your `ar_tracker.launch`.

In `ar_tracker.launch` change the default values for `cam_image_topic` and `cam_info_topic`. Let's also set `camera_frame_id` and `output_frame` to `"lense"`.

Now you should be all set for launching the camera with AR tag tracking.

First roslaunch the R2D2 launch-file from the last lab, then, in a separate terminal window, launch the newly created `ar_tracker.launch`.

(5) To visualize images from the camera, use ADD-button to add Image to your RViz configuration. Make sure to set correct image topic name.

Next, use the ADD-button to add a Marker.

If everything is working, you should see images from the camera in a sub-window in RViz and when you place an AR tag in the camera's field of view, you should see marker(s) relative to camera frame, i.e. relative to `lense`.

(6) Next, set up the R2D2 description so that the the `lense` link is the camera and in its default state the camera is looking straight ahead. For that, create a new empty link called `eye` that is jointed with the `lense`. Set the orientation between `lense` and `eye` so that when you move AR tag away from the camera, you would see the same behavior in RViz with the AR tag markers. Make sure that movement in up-down and left-right are also working properly.

>>> Show the result to your lab instructor! <<<

Clean up your workspace

NB! Before you leave the lab, make sure you have pushed all the files in your catkin workspace to your git cloud service.

In terminal, **cd** to **<yourname>-rtech**

Type

```
git config user.email "youremail@example.com"
```

Type

```
git status
```

You should now see all the new and modified files in red.

Prepare the relevant files for the commit.

```
git add file_name_in_red1 file_name_in_red2
```

When you now type

```
git status
```

you should see all the added files in green. You are now ready to commit changes. Type

```
git commit –m “Insert a brief explanation”
```

Your changes have now been committed but not yet uploaded to the cloud. To upload your files, type

```
git push
```

In your web browser, **verify that all the files** have been uploaded to the **<yourname>-rtech** repository.

Delete the ***<yourname>-rtech*** folder and any other files you created from the lab’s computer.