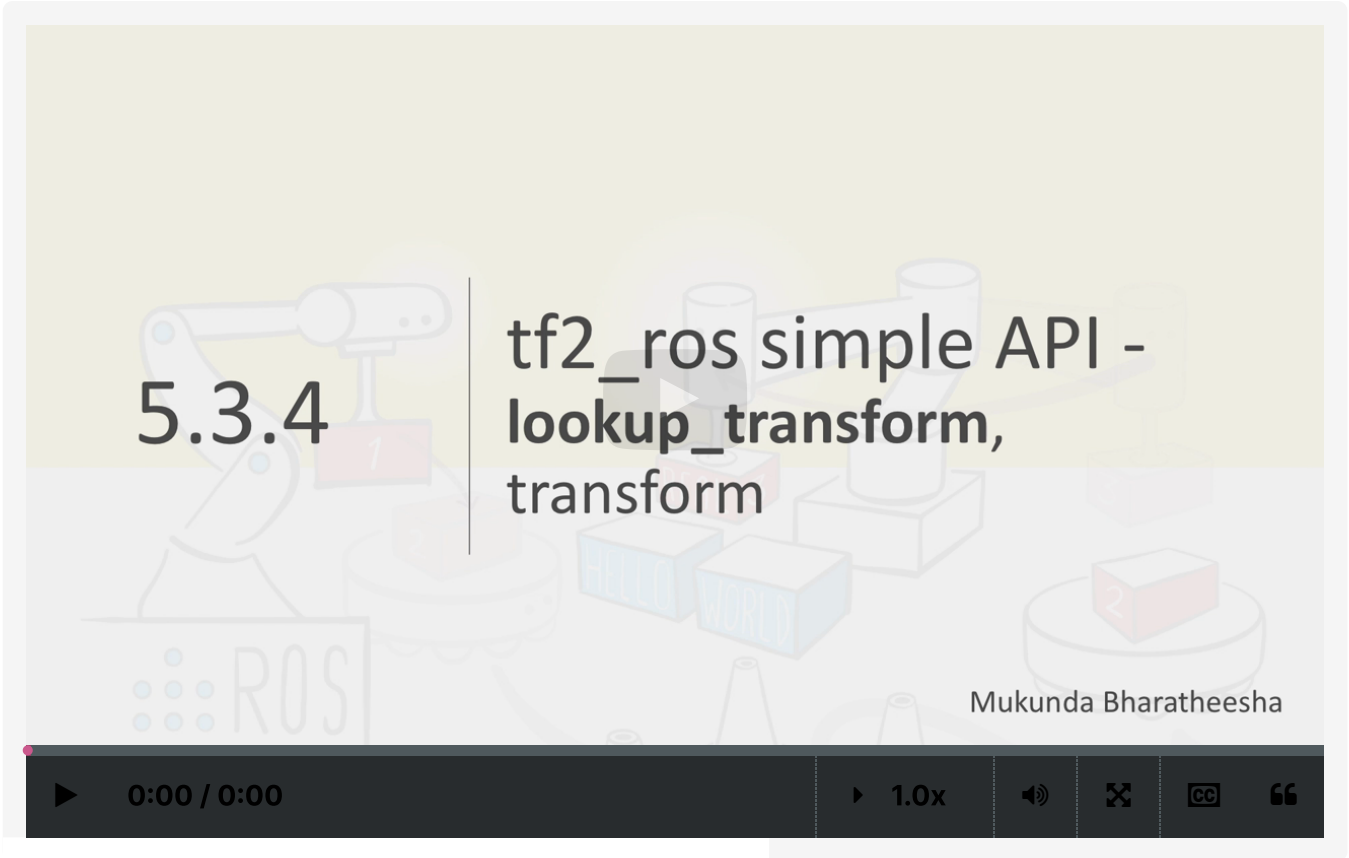
In this video lecture, we will go over an example to help understand the API called `lookup_transform`, one of the most important and helpful transform APIs.

Important Note:
In the video, the instructor mentions that the `lookup_transform` API returns translation and rotation information separately. That was the case in the `tf` package.
In `tf2_ros`, the `lookup_transform` API actually returns a complete transform message which has
the `geometry_msgs/TransformStamped` message type.

Video



Video ▼ Subtitles ▼

Subtitles (captions) in other languages than provided can be viewed at YouTube. Select your language in the CC-button of YouTube

---

Let's inspect the *robot squabblers* script from the start of this week.

You can find it on the scripts folder for `hrwros_week5` package.

That script actually uses the tf2 API to find the relative positions of the TurtleBot, so, let's take a look at the `robot_squabblers.py` script file to find out how it was done.

- We have the necessary `rospy` and `tf2_ros` imports.
- There are two lists of strings, that actually contain the conversation lines between the robots.
- But let's dive into the `robot_squabblers()` function:
  - It first initializes a ROS node called robot_squabblers.
  - It then creates a tf buffer, which we learned about in the last lecture. If you don't remember exactly, don't be afraid to go back and re-watch it if you need to! Since we don't create any specific buffer length, it will go with the default length of 10 seconds.
  - Then, we associate the buffer to a tf listener, so that transform information on the tf topic is constantly updated to the tf buffer, as soon as these topic values become available.
  - Then, we have some helper variables for updating information regarding the transforms, and for displaying the little conversation in the console.
  - The meat is in the while loop:
    - We look up the transforms between the robot 1 and 2 base links, and the TurtleBot base link. The last argument is a way to indicate that we would like the last available information from the tf buffer.
    - The API returns both translation and rotation information separately, but in this script, we only use the translation information. This used to be the case in the `tf` package. With `tf2_ros`, the `lookup_transform` API returns a `geometry_msgs/TransformStamped` message which contains both translation and rotation information in one message.

- The rest of the script assembles the text lines the robots speak, and prints them to the terminal.
- The rate_pub and rate_listener helper variables help keep the conversation speed right: The robots will wait a bit between sentences, and wait until at least one bit of information is inside the tf buffer.
- It is always good practice to put code that looks up transform information in a try-catch block, even if you check if there are messages available beforehand! This prevents warnings and errors.
- Finally, keep in mind that the transform lookup API returns a `transform` message. This looks similar to a `pose` message, but is not the same!

Question 1

1 point possible (ungraded)

The third argument to the lookup_transform("world","base_link", rospy.Time()) API indicates:

- ◯ The time at which the API lookup_transform is being called.
- ◯ System time.
- ◯ Look for the latest available transform in the buffer between world and base_link.

Submit