Robot Operating System

Chapter 4

TF

Pyae Soan Aung
RHCSA
Rom Robitics Co.ltd

Outline

1. What is tf?

2. Robot State Publisher

3. Static Transform Publisher

4. tf Broadcaster

5. tf listener

6. adding frame

7. tf and time

8. tf time travel

What is tf?

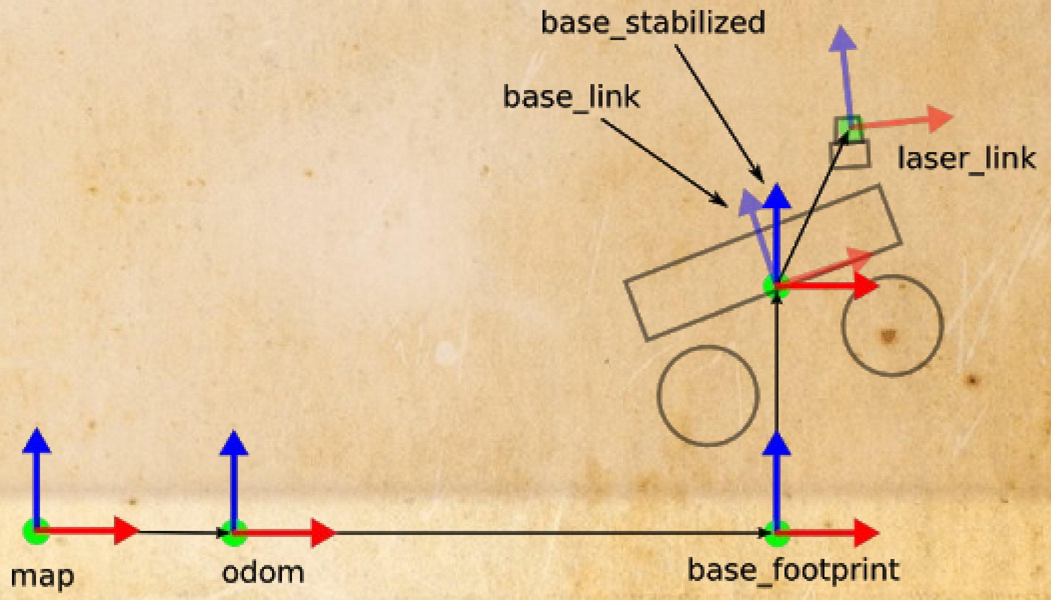
၁) ROS မှာ အချိန်နဲ့အမျှ ပြောင်းလဲနေတဲ့ 3D frame (ဥပမာ /world, /base_link, /camera_link) frame တွေရဲ့ Position နဲ့ Orientation ကို record လုပ်ပြီး publish လုပ်ပေးပါတယ်။

၂) Publish လုပ်ရာမှာ central server ဆိုတာမရှိဘဲsystem ထဲကဘယ် node မဆိုရယူအသုံးပြုနိုင်တယ်။

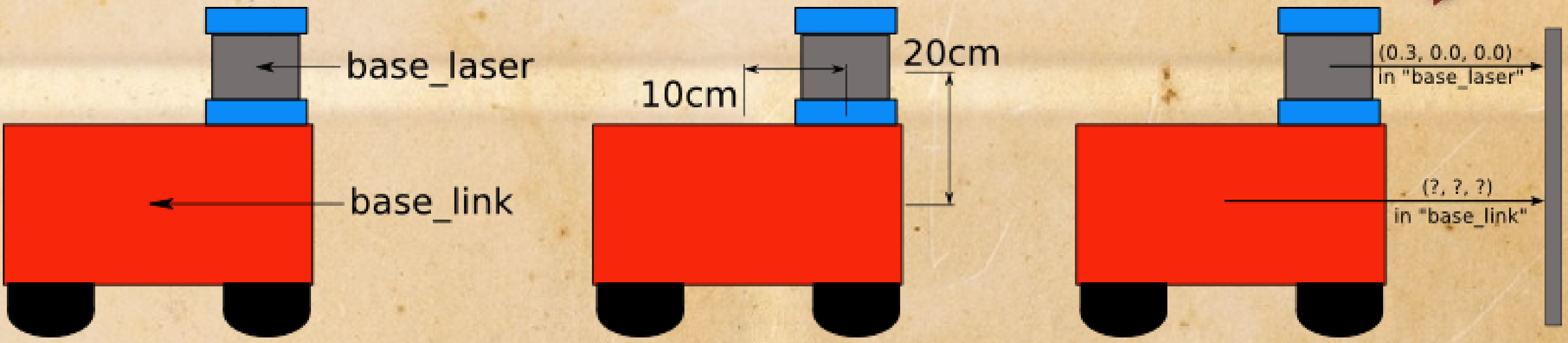
ဥပမာ ပြောရရင် လွန်ခဲ့တဲ့ ၅ စက္ကန့်က Robot arm ရဲ့ end effector သည် world ရဲ့ ဘယ်နားမှာလဲ စတဲ့ မေးခွန်းတွေကို အဖြေရှာဖို့ tf ကို သုံးရပါတယ်။

၄) tf အတွက် command line tools တွေကတော့ tf package ထဲက node များကို အသုံးပြုနိုင်ပါတယ်။

```
~$ rosrun tf view_frame
~$ rosrun tf tf_monitor /source_frame /target_frame
~$ rosrun tf tf_echo /source_frame /target_frame
```



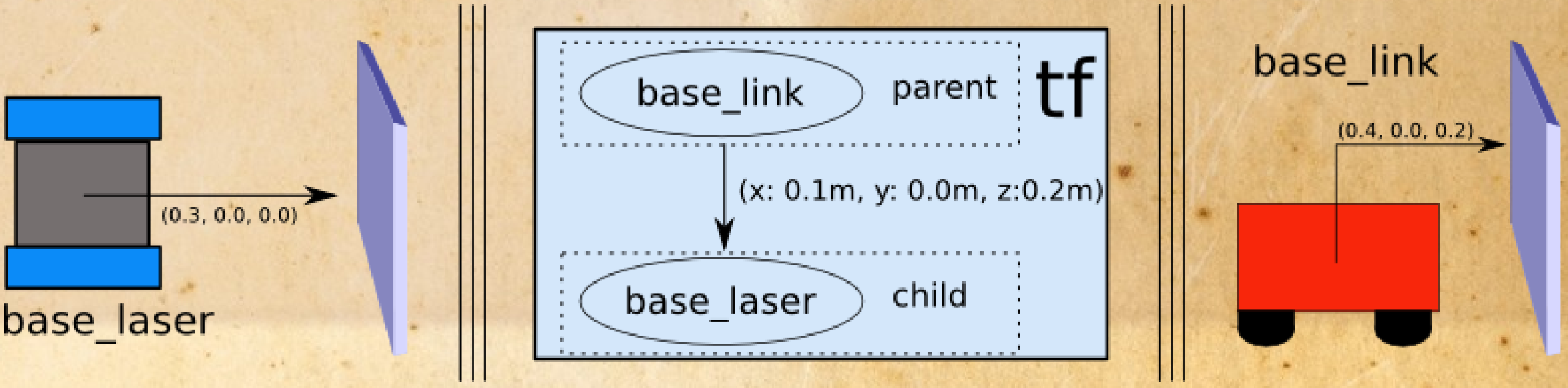
Transform at Mobile Robot Sample



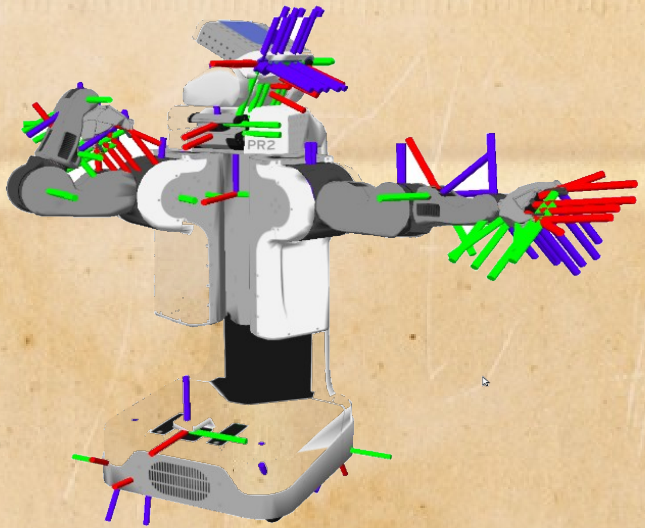
base_link ==> base_laser (x=20, y=0, z=10, r=0, p=0, y=0)

base_laser ==> wall (x=0.3, y=0, z=0, r=0, p=0, y=0)

base_link ==> wall (? ? ? ? ? ?)



Robot State Publisher



- Robot State Publisher ဆိုတဲ့ node သည် parameter server ပေါ်က urdf ရဲ့ joint state များကို ရယူသည်။
- Runtime မှာ ပြောင်းလည်းနေတဲ့ sensor_msgs/JointState type ဖြစ်တဲ့ /joint_state ကိုလည်း Subscribe လုပ်သည်။
- ရရှိတဲ့ အချက်အလက်များအပေါ်မူတည်ပြီး forward kinematic တွက်ပြီး tf ကိုထုတ်လုပ်ပေးပါတယ်။
- Api ရှိပြီး program ရေးသားနိုင်ပေမဲ့ Node အနေနဲ့သာ အသုံးပြုကြပါတယ်။
- လိုအပ်တဲ့ tf အများစုကို developer ကိုတိုင် ထုတ်လုပ်ပေးစရာမလိုတာမို့ အချိန်ကုန်သက်သာပါတယ်။

```
<launch>
        <node pkg=”robot_state_publisher” type=”robot_state_publisher” name=”some”/>
</launch>
```

Static Transform Publisher

- သူကတော့ Code ရေးစရာမလိုလဲ static transform တွေကို publish လုပ်ပေးပါတယ်။ လိုအပ်တဲ့ အတိုင်းအတာ နဲ့ frame name တွေတော့ ထည့်ပေးရမည်။

- position သည် meter unit ဖြစ်ပြီး orientation part ကို rpy သို့မဟုတ် quaternion သုံးနိုင်သည်။

static_transform_publisher x y z yaw pitch roll frame_id child_frame_id period_in_ms

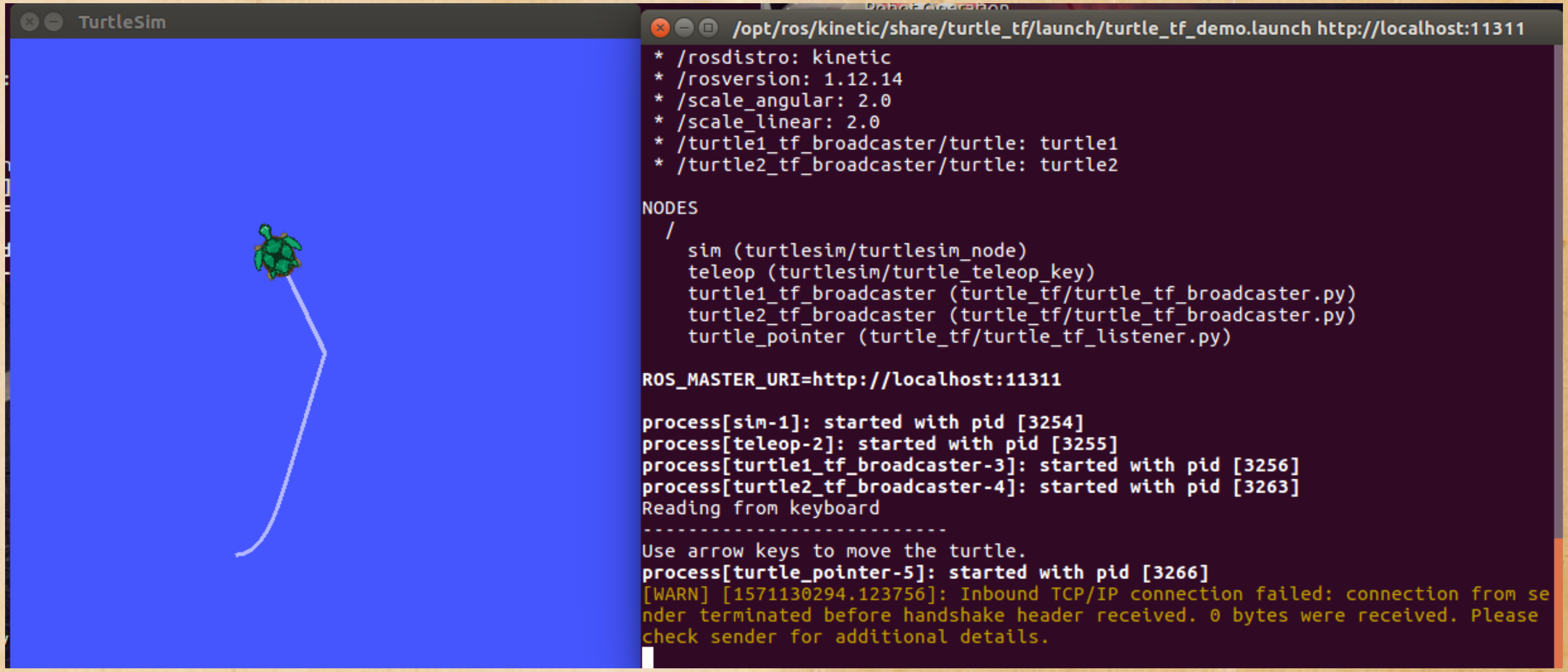
static_transform_publisher x y z qx qy qz qw frame_id child_frame_id period_in_ms

- launch ဖိုင်ကနေသုံးသင့်ပါသည်။

```
<launch>
        <node pkg=”tf”
                    type=”static_transform_publisher”
                    name=”laser_broadcaster”
                    args=”1 0 0 0 0 0 1 base_link laser_link 100”    />
</launch>
```

TF tutorial Examples

```
~$ roslaunch learning_tf start_demo.launch
```


TurtleSim
/opt/ros/kinetic/share/turtle_tf/launch/turtle_tf_demo.launch http://localhost:11311

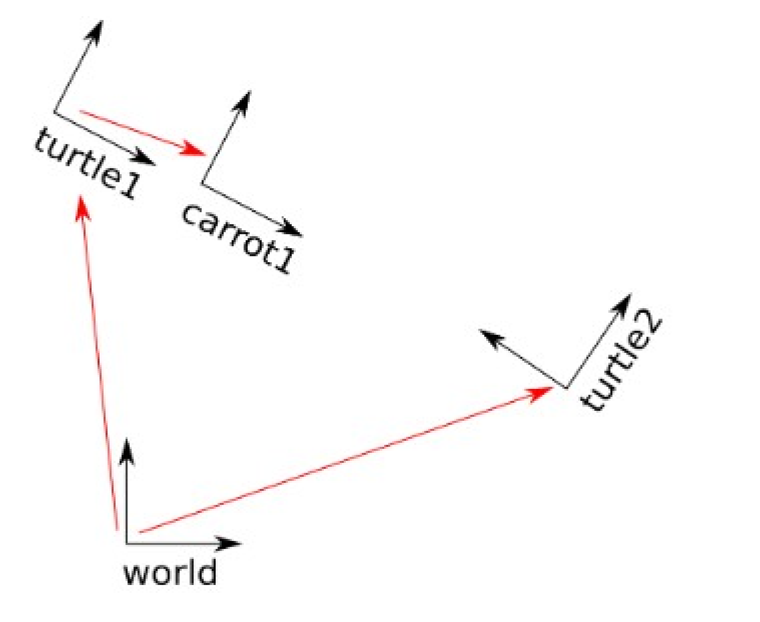
```
 * /rosdistro: kinetic
 * /rosversion: 1.12.14
 * /scale_angular: 2.0
 * /scale_linear: 2.0
 * /turtle1_tf_broadcaster/turtle: turtle1
 * /turtle2_tf_broadcaster/turtle: turtle2

NODES
  /
    sim (turtlesim/turtlesim_node)
    teleop (turtlesim/turtle_teleop_key)
    turtle1_tf_broadcaster (turtle_tf/turtle_tf_broadcaster.py)
    turtle2_tf_broadcaster (turtle_tf/turtle_tf_broadcaster.py)
    turtle_pointer (turtle_tf/turtle_tf_listener.py)

ROS_MASTER_URI=http://localhost:11311

process[sim-1]: started with pid [3254]
process[teleop-2]: started with pid [3255]
process[turtle1_tf_broadcaster-3]: started with pid [3256]
process[turtle2_tf_broadcaster-4]: started with pid [3263]
Reading from keyboard
---------------------------
Use arrow keys to move the turtle.
process[turtle_pointer-5]: started with pid [3266]
[WARN] [1571130294.123756]: Inbound TCP/IP connection failed: connection from se
nder terminated before handshake header received. 0 bytes were received. Please
check sender for additional details.
```

ခုနေ လိပ်သွားနေပေမဲ့ transform frame မရှိလို့ rviz ဖွင့်ကြည့်ရင် turtle1 ရဲ့ frame မတွေ့ရပါ။

How to Broadcast tf?

```
#include <ros/ros.h>
#include <tf/transform_broadcaster.h>
#include <turtlesim/Pose.h>

std::string turtle_name;
void poseCallback(const turtlesim::PoseConstPtr& msg)  {

  static tf::TransformBroadcaster br;
  tf::Transform transform;
  transform.setOrigin( tf::Vector3(msg->x, msg->y, 0.0) );
  tf::Quaternion q;
  q.setRPY(0, 0, msg->theta);
  transform.setRotation(q);
  br.sendTransform(tf::StampedTransform(transform, ros::Time::now(), "world",
turtle_name));
}

int main(int argc, char** argv){
  ros::init(argc, argv, "my_tf_broadcaster");
  if (argc != 2){ROS_ERROR("need turtle name as argument"); return -1;};
  turtle_name = argv[1];

  ros::NodeHandle node;
  ros::Subscriber sub = node.subscribe(turtle_name+"/pose", 10, &poseCallback);

  ros::spin();
  return 0;
};
```

Adding new frame



```
#include <ros/ros.h>
#include <tf/transform_broadcaster.h>

int main(int argc, char** argv){
  ros::init(argc, argv, "my_tf_broadcaster");
  ros::NodeHandle node;

  tf::TransformBroadcaster br;
  tf::Transform transform;

  ros::Rate rate(10.0);
  while (node.ok()){
    transform.setOrigin( tf::Vector3(0.0, 2.0, 0.0) );
    transform.setRotation( tf::Quaternion(0, 0, 0, 1) );

    br.sendTransform(tf::StampedTransform(transform, ros::Time::now(),"turtle1",
"carrot1"));

    rate.sleep();
  }
  return 0;
};
```

How to Listen tf?

```
  While ( node.ok() )
{
    tf::StampedTransform transform;

    Try
      {
              listener.lookupTransform("/turtle2", "/turtle1", ros::Time(0), transform);
      }
    catch (tf::TransformException ex)
      {
            ROS_ERROR("%s",ex.what());
            ros::Duration(1.0).sleep();
       }
}
```

For tf, time 0 means "the latest available" transform in the buffer.

TF and Time

```
listener.lookupTransform("/turtle2", "/turtle1", ros::Time::now(), transform);
```

So, all of the sudden lookupTransform() is failing, telling you repeatedly:
Why is that? Well, each listener has a buffer where it stores all the coordinate transforms coming from the different tf broadcasters. When a broadcaster sends out a transform, it takes some time before that transform gets into the buffer (usually a couple of milliseconds). So, when you request a frame transform at time "now", you should wait a few milliseconds for that information to arrive

```
ros::Time now = ros::Time::now();

listener.waitForTransform("/turtle2", "/turtle1",now,ros::Duration(3));

listener.lookupTransform("/turtle2", "/turtle1",now, transform);
```

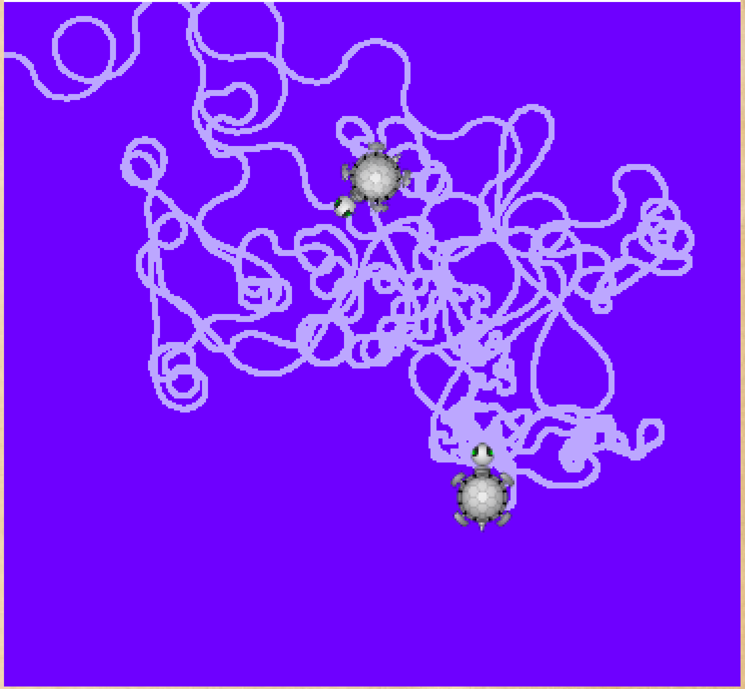
So waitForTransform() will actually block until the transform between the two turtles becomes available (this will usually take a few milliseconds),

OR –

if the transform does not become available-- until the timeout has been reached.

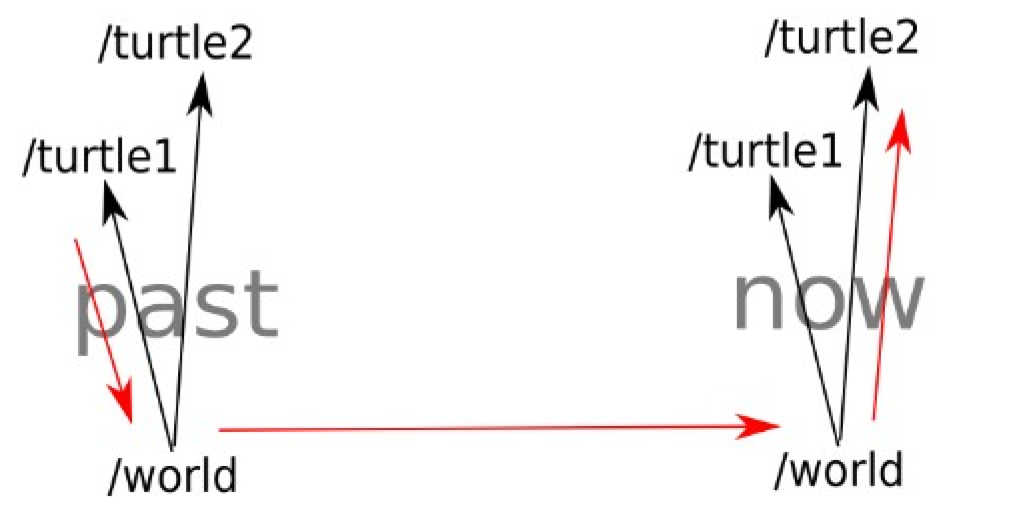
TF and Time Travel

```
try{
    ros::Time past = ros::Time::now() - ros::Duration(5.0);

    listener.waitForTransform( "/turtle2", "/turtle1", past, ros::Duration(1.0) );

    listener.lookupTransform("/turtle2", "/turtle1", past, transform);
```



TF and Time Travel

```
try{
  ros::Time now = ros::Time::now();
  ros::Time past = now – ros::Duration(5.0);

  listener.waitForTransform("/turtle2",now,"/turtle1",past,"/world",ros::Duration(1));

  listener.lookupTransform("/turtle2", now, "/turtle1", past, "/world", transform);
```



1) Give the transform from this frame,

2) at this time ...

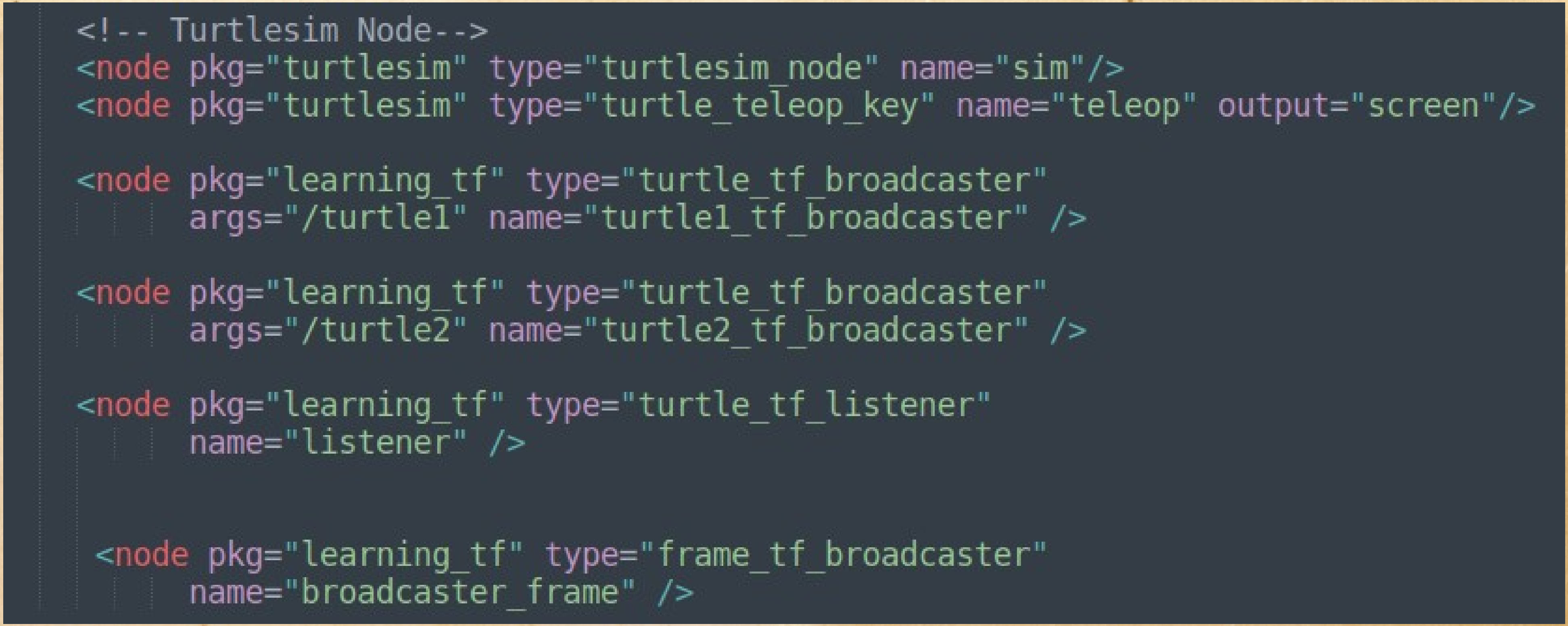
3) ... to this frame,

4) at this time.

5) Specify the frame that does not change over time, in this case the "/world" frame, and

6) the variable to store the result in.

Launch File



```
<!-- Turtlesim Node-->
<node pkg="turtlesim" type="turtlesim_node" name="sim"/>
<node pkg="turtlesim" type="turtle_teleop_key" name="teleop" output="screen"/>

<node pkg="learning_tf" type="turtle_tf_broadcaster"
      args="/turtle1" name="turtle1_tf_broadcaster" />

<node pkg="learning_tf" type="turtle_tf_broadcaster"
      args="/turtle2" name="turtle2_tf_broadcaster" />

<node pkg="learning_tf" type="turtle_tf_listener"
      name="listener" />


 <node pkg="learning_tf" type="frame_tf_broadcaster"
      name="broadcaster_frame" />
```

~$ rosmsg show tf/tfMessage

Thank you!

This work is licensed under
Rom Robotics.
It makes use of the works of
GhosTmAN.

Facebook	**- https://www.facebook.com/ROMROBOTS/**
Website	**- http://romrobots.com/**
YouTube	**- https://www.youtube.com/channel/UCq6OHEpbQ9RQIHV2U6W8qAQ**
GitHub	**- https://github.com/GreenGhostMan**
Phone	**- 09259288229 , 09259288230**