Theoretical Part

1.

a. Due to the rotation being with respect to the fixed frame (frame O) we need to apply the transformations in reverse.

$$ {}^1_0R = \begin{bmatrix} \begin{bmatrix}1&0&0\\0&0&-1\\0&1&0\end{bmatrix} & \begin{bmatrix}0\\0\\0\end{bmatrix} \\ 0\quad 0\quad 0 & 1 \end{bmatrix}, {}^2_1R = \begin{bmatrix} \begin{bmatrix}1&0&0\\0&1&0\\0&0&1\end{bmatrix} & \begin{bmatrix}-1\\2\\5\end{bmatrix} \\ 0\quad 0\quad 0 & 1 \end{bmatrix}, {}^3_2R = \begin{bmatrix} \begin{bmatrix}0&-1&0\\1&0&0\\0&0&1\end{bmatrix} & \begin{bmatrix}0\\0\\0\end{bmatrix} \\ 0\quad 0\quad 0 & 1 \end{bmatrix} $$

$$ \Longrightarrow R_3^0 = R_3^2\, R_2^1\, R_1^0 = \begin{bmatrix} \begin{bmatrix}0&-1&0\\0&0&-1\\1&0&0\end{bmatrix} & \begin{bmatrix}-1\\-5\\2\end{bmatrix} \\ 0\quad 0\quad 0 & 1 \end{bmatrix} $$

$$ \Longrightarrow P^0 = R_3^0\, P^3 = \begin{bmatrix} \begin{bmatrix}0&0&1\\1&0&0\\0&1&0\end{bmatrix} & \begin{bmatrix}-2\\-1\\5\end{bmatrix} \\ 0\quad 0\quad 0 & 1 \end{bmatrix} \times \begin{bmatrix} \begin{bmatrix}4\\2\\0\end{bmatrix} \\ 1 \end{bmatrix} = \begin{bmatrix} \begin{bmatrix}-2\\3\\7\end{bmatrix} \\ 1 \end{bmatrix} $$

b. In this part we will reverse the order of applying the transformations in the part a. As shown below, the result is not the same as the result of the part a because the order of applying is important.

$$ R_3^0 = R_1^0 R_2^1\, R_3^2 = \begin{bmatrix} \begin{bmatrix}0&-1&0\\0&0&-1\\1&0&0\end{bmatrix} & \begin{bmatrix}-1\\-5\\2\end{bmatrix} \\ 0\quad 0\quad 0 & 1 \end{bmatrix} $$

$$ \Longrightarrow P^0 = R_3^0\, P^3 = \begin{bmatrix} \begin{bmatrix}0&-1&0\\0&0&-1\\1&0&0\end{bmatrix} & \begin{bmatrix}-1\\-5\\2\end{bmatrix} \\ 0\quad 0\quad 0 & 1 \end{bmatrix} \times \begin{bmatrix} \begin{bmatrix}4\\2\\0\end{bmatrix} \\ 1 \end{bmatrix} = \begin{bmatrix} \begin{bmatrix}-3\\-5\\6\end{bmatrix} \\ 1 \end{bmatrix} $$

c. With this part being fixed frame too, the transformations should be applied in reverse.

$$ {}^1_0R = \begin{bmatrix} \begin{bmatrix}1&0&0\\0&1&0\\0&0&1\end{bmatrix} & \begin{bmatrix}-1\\2\\5\end{bmatrix} \\ 0\quad 0\quad 0 & 1 \end{bmatrix}, {}^2_1R = \begin{bmatrix} \begin{bmatrix}0&-1&0\\1&0&0\\0&0&1\end{bmatrix} & \begin{bmatrix}0\\0\\0\end{bmatrix} \\ 0\quad 0\quad 0 & 1 \end{bmatrix}, {}^3_2R = \begin{bmatrix} \begin{bmatrix}1&0&0\\0&0&-1\\0&1&0\end{bmatrix} & \begin{bmatrix}0\\0\\0\end{bmatrix} \\ 0\quad 0\quad 0 & 1 \end{bmatrix} $$

$$ \Longrightarrow R_3^0 = R_3^2\, R_2^1\, R_1^0 = \begin{bmatrix} \begin{bmatrix}0&-1&0\\0&0&-1\\1&0&0\end{bmatrix} & \begin{bmatrix}-2\\-5\\-1\end{bmatrix} \\ 0\quad 0\quad 0 & 1 \end{bmatrix} $$

$$ \Longrightarrow P^0 = R_3^0\, P^3 = \begin{bmatrix} \begin{bmatrix}0&0&1\\1&0&0\\0&1&0\end{bmatrix} & \begin{bmatrix}-1\\2\\5\end{bmatrix} \\ 0\quad 0\quad 0 & 1 \end{bmatrix} \times \begin{bmatrix} \begin{bmatrix}4\\2\\0\end{bmatrix} \\ 1 \end{bmatrix} = \begin{bmatrix} \begin{bmatrix}-4\\-5\\3\end{bmatrix} \\ 1 \end{bmatrix} $$

The result makes it clear that the order of applying the transformation matrices is directly influential for the result.

2.

a. In this case, the rotation is with respect to the current frame so we apply the transformation matrices in the order they are being implemented.

$${}_0^1R = \begin{bmatrix} \begin{bmatrix}1&0&0\\0&0&-1\\0&1&0\end{bmatrix} & \begin{bmatrix}0\\0\\0\end{bmatrix} \\ 0\quad 0\quad 0 & 1 \end{bmatrix}, {}_1^2R = \begin{bmatrix} \begin{bmatrix}1&0&0\\0&1&0\\0&0&1\end{bmatrix} & \begin{bmatrix}-1\\2\\5\end{bmatrix} \\ 0\quad 0\quad 0 & 1 \end{bmatrix}, {}_2^3R = \begin{bmatrix} \begin{bmatrix}0&-1&0\\1&0&0\\0&0&1\end{bmatrix} & \begin{bmatrix}0\\0\\0\end{bmatrix} \\ 0\quad 0\quad 0 & 1 \end{bmatrix}$$

$$\Rightarrow R_3^0 = R_1^0 R_2^1\, R_3^2 = \begin{bmatrix} \begin{bmatrix}0&-1&0\\0&0&-1\\1&0&0\end{bmatrix} & \begin{bmatrix}-1\\-5\\2\end{bmatrix} \\ 0\quad 0\quad 0 & 1 \end{bmatrix}$$

$$\Rightarrow P^0 = R_3^0\, P^3 = \begin{bmatrix} \begin{bmatrix}0&-1&0\\0&0&-1\\1&0&0\end{bmatrix} & \begin{bmatrix}-1\\-5\\2\end{bmatrix} \\ 0\quad 0\quad 0 & 1 \end{bmatrix} \times \begin{bmatrix} \begin{bmatrix}4\\2\\0\end{bmatrix} \\ 1 \end{bmatrix} = \begin{bmatrix} \begin{bmatrix}-3\\-5\\6\end{bmatrix} \\ 1 \end{bmatrix}$$

b. As done in the Question 1.b, the order of applying the transformation matrices is important and applying them from right to left is different from applying them from left to right.

$$R_3^0 = R_3^2\, R_2^1\, R_1^0 = \begin{bmatrix} \begin{bmatrix}0&-1&0\\0&0&-1\\1&0&0\end{bmatrix} & \begin{bmatrix}-1\\-5\\2\end{bmatrix} \\ 0\quad 0\quad 0 & 1 \end{bmatrix}$$

$$\Rightarrow P^0 = R_3^0\, P^3 = \begin{bmatrix} \begin{bmatrix}0&0&1\\1&0&0\\0&1&0\end{bmatrix} & \begin{bmatrix}-2\\-1\\5\end{bmatrix} \\ 0\quad 0\quad 0 & 1 \end{bmatrix} \times \begin{bmatrix} \begin{bmatrix}4\\2\\0\end{bmatrix} \\ 1 \end{bmatrix} = \begin{bmatrix} \begin{bmatrix}-2\\3\\7\end{bmatrix} \\ 1 \end{bmatrix}$$

3. The order of applying the matrices is different in the questions 1.a and 1.b due to the respect of rotation being to the fixed frame vs. current frame.

The source of this difference is in the rotation performance being about a fixed frame or being performed continuously about the frames in order.

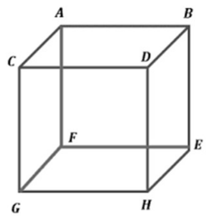
The first situation (the performance being about a fixed frame) is called fixed frame and in the result, the transformation matrices should be implemented in reverse the order but the second situation (with the frame being chased by the previous frame) is called current frame and the order of transformation matrices must be kept the way it is.

4.

a. For the prismatic joints (linear joints) the z axis should be in the direction of the linear relative motion and for the revolute joints (rotary joints) the z axis is in the same direction of the rotation axis.

b. Joint parameters described as θ_i (for revolute joints) and d_i (for prismatic joints) are parts of the D-H parameters. θ_i is the angle between x_i and x_{i-1} about the z_i axis and the d_i is the distance between x_i and x_{i-1} along the z_i axis.

c. Yes! Skew lines are lines which have no intersection but also, they are not parallel. These lines only exist in 3 or more dimensions and must be placed in different planes. There can be a perpendicular line to both and a simple example can be shown below.



$\overrightarrow{AB}$ and $\overrightarrow{FG}$ are two skew lines and $\overrightarrow{AF}$ is perpendicular to both of them.

d. When determining the x_i we need to pay attention to the z_i and z_{i+1}. x_i must be perpendicular to both z_i and z_{i+1} and cannot be parallel to z_{i+1}.

e. Link parameters are associated with the physical parts of the robot and α_i describes the twist of the link and the a_i describes the length of the link.

α_i is the angle between z_i and z_{i+1} about the x_i axis and the a_i is the distance between z_i and z_{i+1} along the x_i axis.

f. The difference between the two naming frames, ${}_{0}^{1}R$ and ${}^{1}R_{0}$ is not that much! With both the transformation is from previous frame (frame 0) to the current frame (frame 1) and the only difference is the placement of the previous frame's number.

5. In general, I will explain how z and x axis and each parameter can be calculated and the result of the a and b parts are shown below.

Z: This axis is described as the direction of linear relative motion of prismatic joints and in the direction of rotation axis of revolute joints.

X: This axis can be defined as any direction while it must be perpendicular to current z axis and previous z axis.

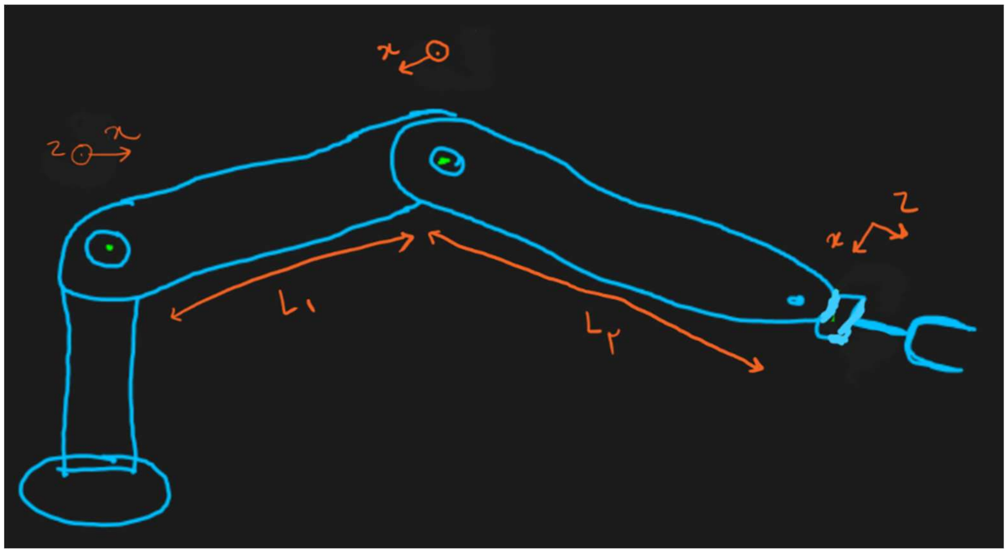
α_{i-1}: Is the angle between z_i and z_{i-1} along x_i.

a_{i-1}: Is the distance between z_i and z_{i-1} along x_i.

d_i: Is the distance between x_i and x_{i-1} along z_i.

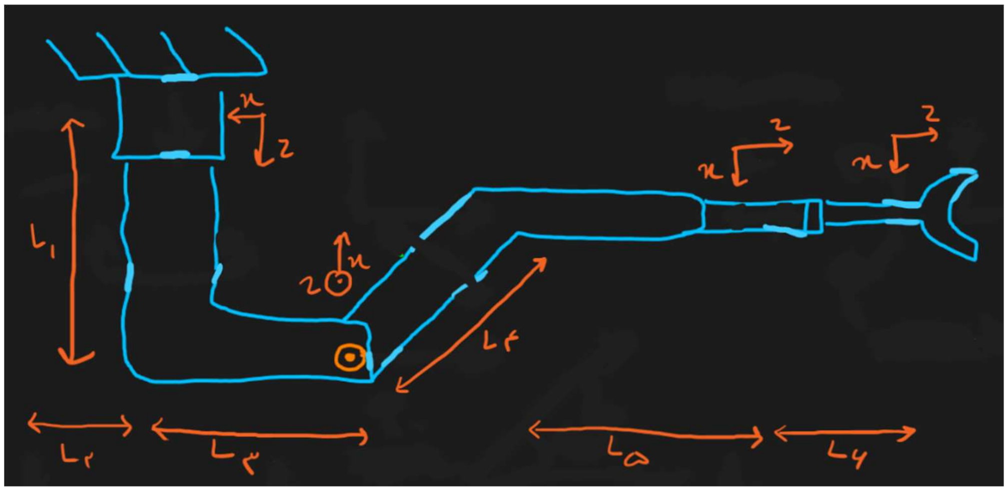
θ_i: is the angle between x_i and x_{i-1} about z_i.

a.



i	α_{i-1}	a_{i-1}	d_i	θ_i
1	0	0	0	θ_1
2	0	l_1	0	θ_2
3	90	0	l_2	θ_3

b. B



i	α_{i-1}	a_{i-1}	d_i	θ_i
1	0	0	l_1	θ_1
2	90	l_3	0	θ_2
3	90	$\frac{\sqrt{2}}{2}l_4$	$\frac{\sqrt{2}}{2}l_4 + l_5^*$	0
4	0	0	0	θ_3

6.

a.

- Linear & Angular Motion: Using Jacobian matrix describing the motions in linear world and angular world can be possible.
- Velocity Propagation: Based on the application, in most cases the velocity of end effector must be calculated and controlled. With Jacobian matrix being constrained to time, talking about velocity can be reached.
- Explicit Form
- Static Forces: Force and torque concept can be involved and used with Jacobian matrix.

b. Jacobian matrix is described as below:

$$J = \left(\frac{J_v}{J_\omega}\right) = \left(\frac{[\epsilon_1 Z_1 + \bar{\epsilon}_1(Z_1 \times P_{1n}) \quad \epsilon_2 Z_2 + \bar{\epsilon}_2(Z_2 \times P_{2n}) \quad \cdots]}{[\bar{\epsilon}_1 Z_1 \quad \bar{\epsilon}_1 Z_1 \quad \cdots \quad \bar{\epsilon}_n Z_n]}\right)$$

In this matrix, the J_v is the linear velocity and the J_ω is the angular velocity of the joints of the robot. The matrix is a $6 \times n$ dimensional matrix with n being the number of joints.

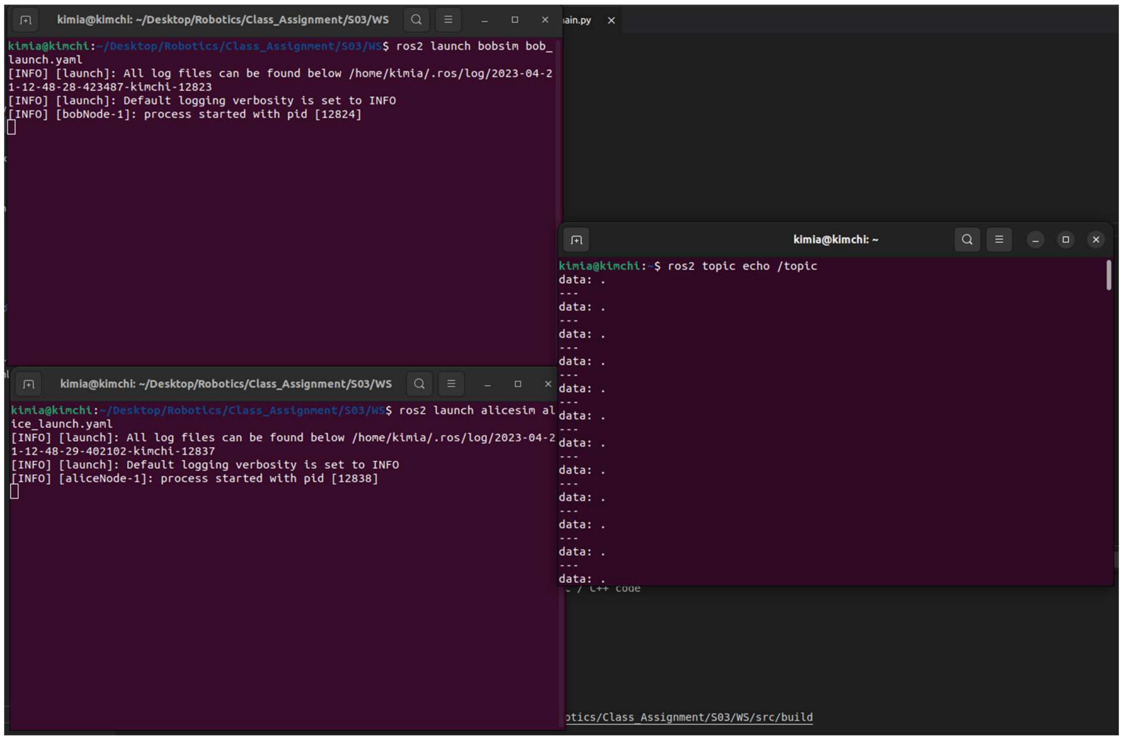
For each joint the linear velocity is described as $[\epsilon Z + \bar{\epsilon}(Z \times P)]\dot{q}$ with $Z\dot{q}$ being the linear velocity of a prismatic joint and $Z\dot{q}P$ being the linear velocity of a revolute joint. The coefficients (ϵ and $\bar{\epsilon}$) decide the type of the joint because a joint cannot be prismatic and revolute at the same time and as the result, only one of these values can be replaced by the linear velocity of a joint.

The angular velocity of a joint can only be defined for the revolute joints and once again the coefficient describes the type of the joint.

c. Linear velocity can be described as the speed of an object in a straight line while the angular velocity is the change of the angle over time. They both can be described and calculated for a robot as mentioned in the part 6.b.

Coding Part

1.
   a. Using publishers and subscribers, a dot can be echoed with topics and adding the launch file will result the terminals below.

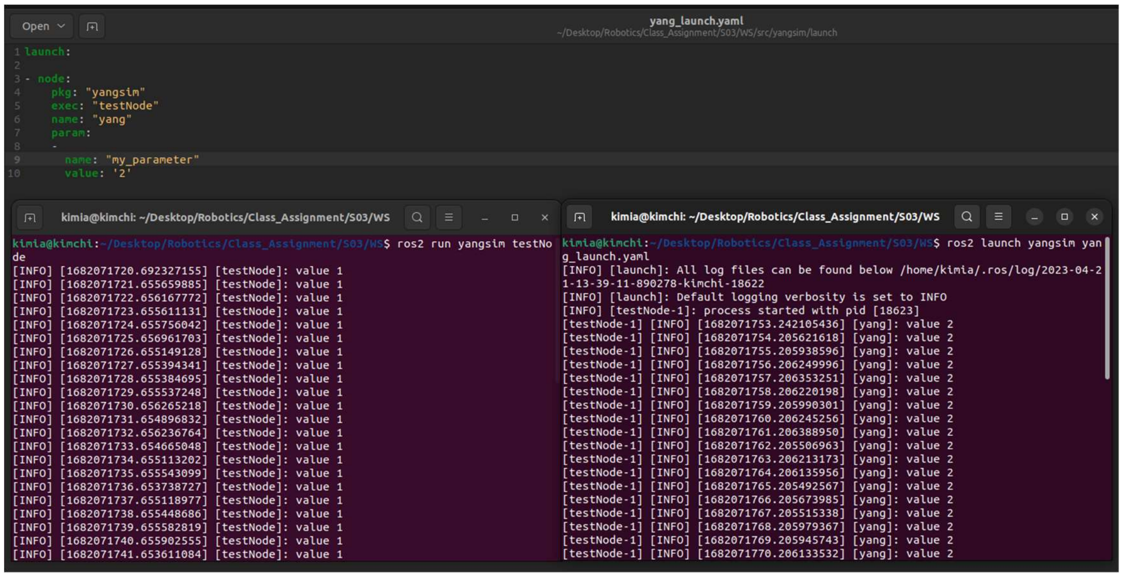


```
kimia@kimchi:~/Desktop/Robotics/Class_Assignment/S03/WS$ ros2 launch bobsim bob_
launch.yaml
[INFO] [launch]: All log files can be found below /home/kimia/.ros/log/2023-04-2
1-12-48-28-423487-kimchi-12823
[INFO] [launch]: Default logging verbosity is set to INFO
[INFO] [bobNode-1]: process started with pid [12824]
```

```
kimia@kimchi:~/Desktop/Robotics/Class_Assignment/S03/WS$ ros2 launch alicesim al
ice_launch.yaml
[INFO] [launch]: All log files can be found below /home/kimia/.ros/log/2023-04-2
1-12-48-29-402102-kimchi-12837
[INFO] [launch]: Default logging verbosity is set to INFO
[INFO] [aliceNode-1]: process started with pid [12838]
```

```
kimia@kimchi:~$ ros2 topic echo /topic
data: .
---
data: .
---
data: .
---
data: .
---
data: .
---
data: .
---
data: .
---
data: .
---
data: .
---
data: .
---
data: .
---
data: .
```

   b. With changing the launch file as below, we can set a parameter in the code.



```yaml
launch:

- node:
    pkg: "yangsim"
    exec: "testNode"
    name: "yang"
    param:
    -
      name: "my_parameter"
      value: '2'
```

```
kimia@kimchi:~/Desktop/Robotics/Class_Assignment/S03/WS$ ros2 run yangsim testNo
de
[INFO] [1682071720.692327155] [testNode]: value 1
[INFO] [1682071721.655659885] [testNode]: value 1
[INFO] [1682071722.656167772] [testNode]: value 1
[INFO] [1682071723.655611131] [testNode]: value 1
[INFO] [1682071724.655756042] [testNode]: value 1
[INFO] [1682071725.656961703] [testNode]: value 1
[INFO] [1682071726.655149128] [testNode]: value 1
[INFO] [1682071727.655394341] [testNode]: value 1
[INFO] [1682071728.655384695] [testNode]: value 1
[INFO] [1682071729.655537248] [testNode]: value 1
[INFO] [1682071730.656265218] [testNode]: value 1
[INFO] [1682071731.654896832] [testNode]: value 1
[INFO] [1682071732.656236764] [testNode]: value 1
[INFO] [1682071733.654665048] [testNode]: value 1
[INFO] [1682071734.655113202] [testNode]: value 1
[INFO] [1682071735.655543099] [testNode]: value 1
[INFO] [1682071736.653738727] [testNode]: value 1
[INFO] [1682071737.655118977] [testNode]: value 1
[INFO] [1682071738.655448686] [testNode]: value 1
[INFO] [1682071739.655582819] [testNode]: value 1
[INFO] [1682071740.655902555] [testNode]: value 1
[INFO] [1682071741.653611084] [testNode]: value 1
```

```
kimia@kimchi:~/Desktop/Robotics/Class_Assignment/S03/WS$ ros2 launch yangsim yan
g_launch.yaml
[INFO] [launch]: All log files can be found below /home/kimia/.ros/log/2023-04-2
1-13-39-11-890278-kimchi-18622
[INFO] [launch]: Default logging verbosity is set to INFO
[INFO] [testNode-1]: process started with pid [18623]
[testNode-1] [INFO] [1682071753.242105436] [yang]: value 2
[testNode-1] [INFO] [1682071754.205621618] [yang]: value 2
[testNode-1] [INFO] [1682071755.205938596] [yang]: value 2
[testNode-1] [INFO] [1682071756.206249996] [yang]: value 2
[testNode-1] [INFO] [1682071757.206353251] [yang]: value 2
[testNode-1] [INFO] [1682071758.206220198] [yang]: value 2
[testNode-1] [INFO] [1682071759.205990301] [yang]: value 2
[testNode-1] [INFO] [1682071760.206245256] [yang]: value 2
[testNode-1] [INFO] [1682071761.206388950] [yang]: value 2
[testNode-1] [INFO] [1682071762.205506963] [yang]: value 2
[testNode-1] [INFO] [1682071763.206213173] [yang]: value 2
[testNode-1] [INFO] [1682071764.206135956] [yang]: value 2
[testNode-1] [INFO] [1682071765.205492567] [yang]: value 2
[testNode-1] [INFO] [1682071766.205673985] [yang]: value 2
[testNode-1] [INFO] [1682071767.205515338] [yang]: value 2
[testNode-1] [INFO] [1682071768.205979367] [yang]: value 2
[testNode-1] [INFO] [1682071769.205945743] [yang]: value 2
[testNode-1] [INFO] [1682071770.206133532] [yang]: value 2
```

   c. With tf2 using DAG graphs to traverse the target frame to the source frame, it can be ensured that no loop can be created in the tree structure. When moving along in the tree, the package uses parent-child relationship so it never looks up a transform to create a loop. Also, with detecting a loop, the package throws a TransformException so we can wait for the transformation to become available again or skip the transformation altogether.

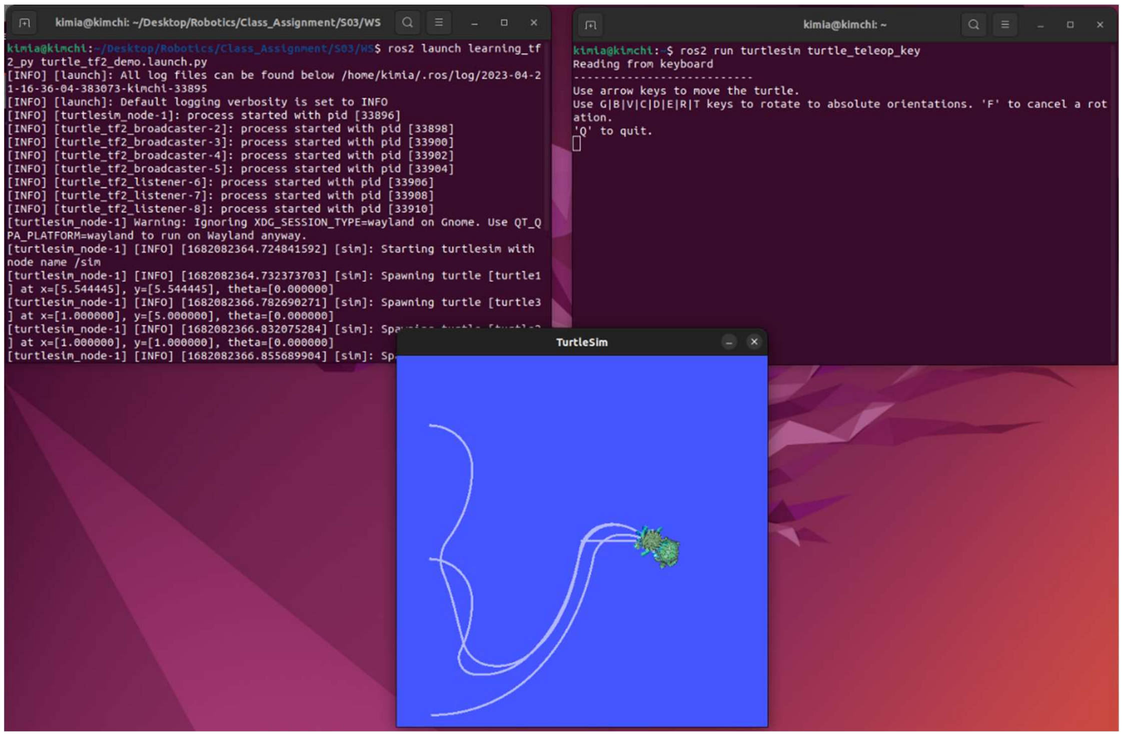
Additionally with using thread-safe structures and synchronizing mechanisms it can be guaranteed that updating in multithreading programs can be done in a safe manner.

d. Static_transform_publisher is a part of tf2 package and publishes a static transformation between target and source frames. Without this tool it can be challenging to calculate transformations between frames (especially in robots with multiple moving parts) and in this case, the programmers have to use their own code to calculate transformation matrices.

e. Static transformations are the transformations that remain fixed with the robot operating vs dynamic transformations that will change over time with parts of the robot moving and operating.
Static transformations are typically used for the parts of the robot that doesn't move such as sensors or fixed parts and the parts that are fixed relative to the robot like the outside world.
On the other hand, dynamic transformations are used for the components of the robot that moves throughout the operation of the robot, like joints and end-effector and calculating transformation between frames that are calculated with the entry data such as cameras or sensors data.

f. Tf2 trees represent the coordinate frames and the transformations between them with nodes being the coordinates and the edges being the transformations while the URDF trees describe the physical structure of the robot with nodes being links or joints and edges being transformations between links and joints.
The relation between these two trees can be defined as that the coordinate frames of the tf2 trees are created using physical descriptions of the robot described in URDF trees.

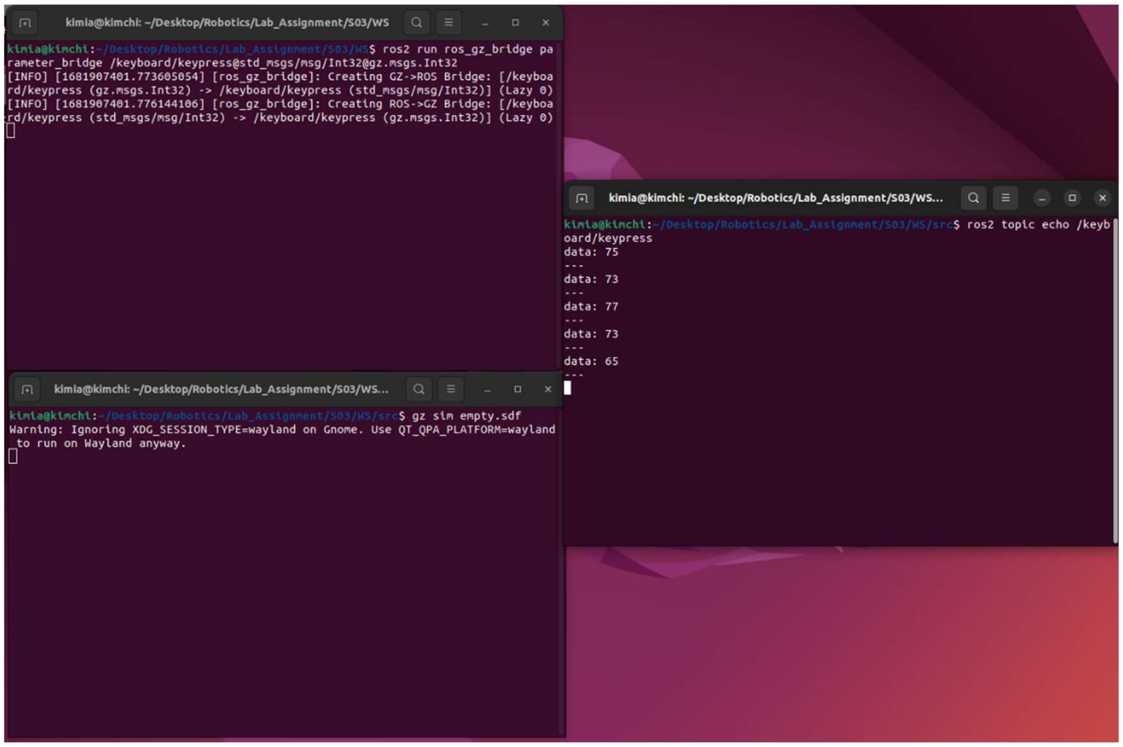
2. We can see the result of using tf2 library and launch files to achieve our goal.



```
kimia@kimchi: ~/Desktop/Robotics/Class_Assignment/S03/WS
kimia@kimchi:~/Desktop/Robotics/Class_Assignment/S03/WS$ ros2 launch learning_tf
2_py turtle_tf2_demo.launch.py
[INFO] [launch]: All log files can be found below /home/kimia/.ros/log/2023-04-2
1-16-36-04-383073-kimchi-33895
[INFO] [launch]: Default logging verbosity is set to INFO
[INFO] [turtlesim_node-1]: process started with pid [33896]
[INFO] [turtle_tf2_broadcaster-2]: process started with pid [33898]
[INFO] [turtle_tf2_broadcaster-3]: process started with pid [33900]
[INFO] [turtle_tf2_broadcaster-4]: process started with pid [33902]
[INFO] [turtle_tf2_broadcaster-5]: process started with pid [33904]
[INFO] [turtle_tf2_listener-6]: process started with pid [33906]
[INFO] [turtle_tf2_listener-7]: process started with pid [33908]
[INFO] [turtle_tf2_listener-8]: process started with pid [33910]
[turtlesim_node-1] Warning: Ignoring XDG_SESSION_TYPE=wayland on Gnome. Use QT_Q
PA_PLATFORM=wayland to run on Wayland anyway.
[turtlesim_node-1] [INFO] [1682082364.724841592] [sim]: Starting turtlesim with
node name /sim
[turtlesim_node-1] [INFO] [1682082364.732373703] [sim]: Spawning turtle [turtle1
] at x=[5.544445], y=[5.544445], theta=[0.000000]
[turtlesim_node-1] [INFO] [1682082366.782690271] [sim]: Spawning turtle [turtle3
] at x=[1.000000], y=[5.000000], theta=[0.000000]
[turtlesim_node-1] [INFO] [1682082366.832075284] [sim]: Spa
] at x=[1.000000], y=[1.000000], theta=[0.000000]
[turtlesim_node-1] [INFO] [1682082366.855689904] [sim]: Sp
```

```
kimia@kimchi: ~
kimia@kimchi:~$ ros2 run turtlesim turtle_teleop_key
Reading from keyboard
---------------------------
Use arrow keys to move the turtle.
Use G|B|V|C|D|E|R|T keys to rotate to absolute orientations. 'F' to cancel a rot
ation.
'Q' to quit.
```

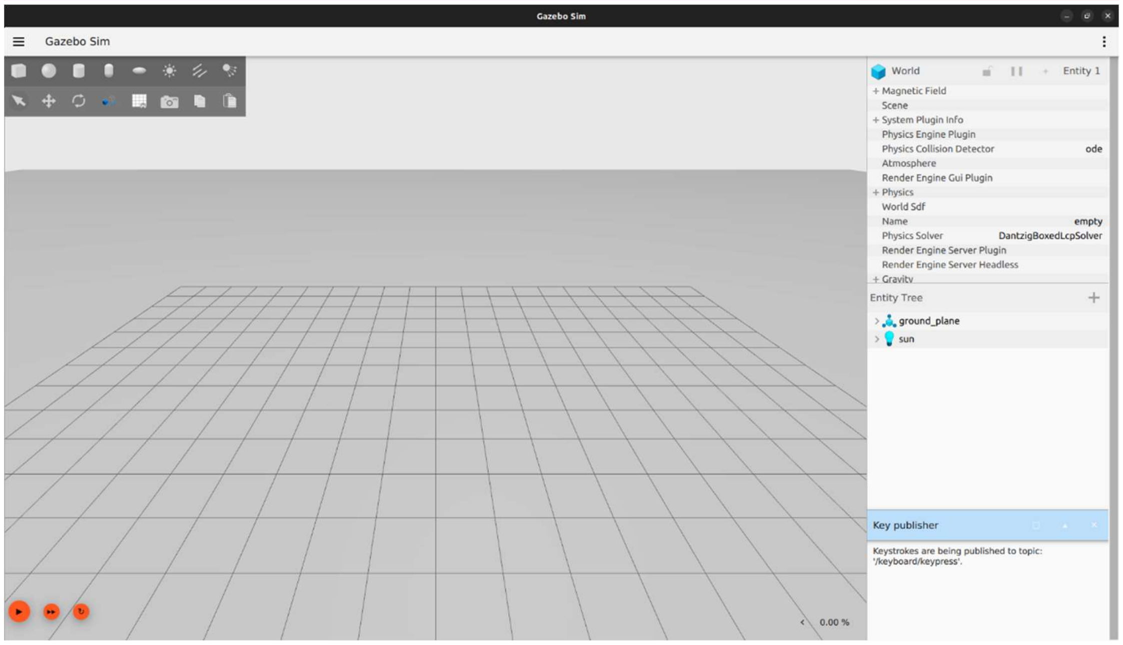
3. Using ROS-Gazebo bridge, the result and terminals can be shown below.



```
kimia@kimchi: ~/Desktop/Robotics/Lab_Assignment/S03/WS
kimia@kimchi:~/Desktop/Robotics/Lab_Assignment/S03/WS$ ros2 run ros_gz_bridge pa
rameter_bridge /keyboard/keypress@std_msgs/msg/Int32@gz.msgs.Int32
[INFO] [1681907401.773605054] [ros_gz_bridge]: Creating GZ->ROS Bridge: [/keyboa
rd/keypress (gz.msgs.Int32) -> /keyboard/keypress (std_msgs/msg/Int32)] (Lazy 0)
[INFO] [1681907401.776144106] [ros_gz_bridge]: Creating ROS->GZ Bridge: [/keyboa
rd/keypress (std_msgs/msg/Int32) -> /keyboard/keypress (gz.msgs.Int32)] (Lazy 0)
```

```
kimia@kimchi: ~/Desktop/Robotics/Lab_Assignment/S03/WS...
kimia@kimchi:~/Desktop/Robotics/Lab_Assignment/S03/WS/src$ ros2 topic echo /keyb
oard/keypress
data: 75
---
data: 73
---
data: 77
---
data: 73
---
data: 65
---
```

```
kimia@kimchi: ~/Desktop/Robotics/Lab_Assignment/S03/WS...
kimia@kimchi:~/Desktop/Robotics/Lab_Assignment/S03/WS/src$ gz sim empty.sdf
Warning: Ignoring XDG_SESSION_TYPE=wayland on Gnome. Use QT_QPA_PLATFORM=wayland
 to run on Wayland anyway.
```



Gazebo Sim

World	Entity 1
+ Magnetic Field	
Scene	
+ System Plugin Info	
Physics Engine Plugin	
Physics Collision Detector	ode
Atmosphere	
Render Engine Gui Plugin	
+ Physics	
World Sdf	
Name	empty
Physics Solver	DantzigBoxedLcpSolver
Render Engine Server Plugin	
Render Engine Server Headless	
+ Gravity	

Entity Tree
- ground_plane
- sun

Key publisher

Keystrokes are being published to topic: '/keyboard/keypress'.