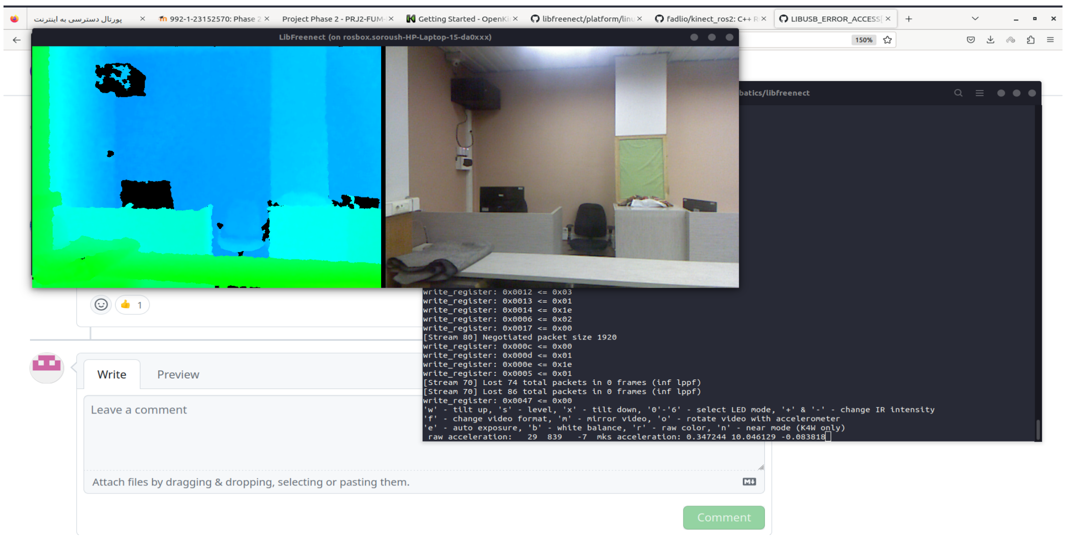
Project Phase 2

Zahra Niazi, Soroush Naseri

Setting up the Kinect

At first we have to se the kinect .firstly we installed the dirvers for kinct such as freeglut3-dev . and then clone the consider repository to our work space and then build it .When we run this command "freenect-glview" it shows the RGBD picture that catched by kinect :

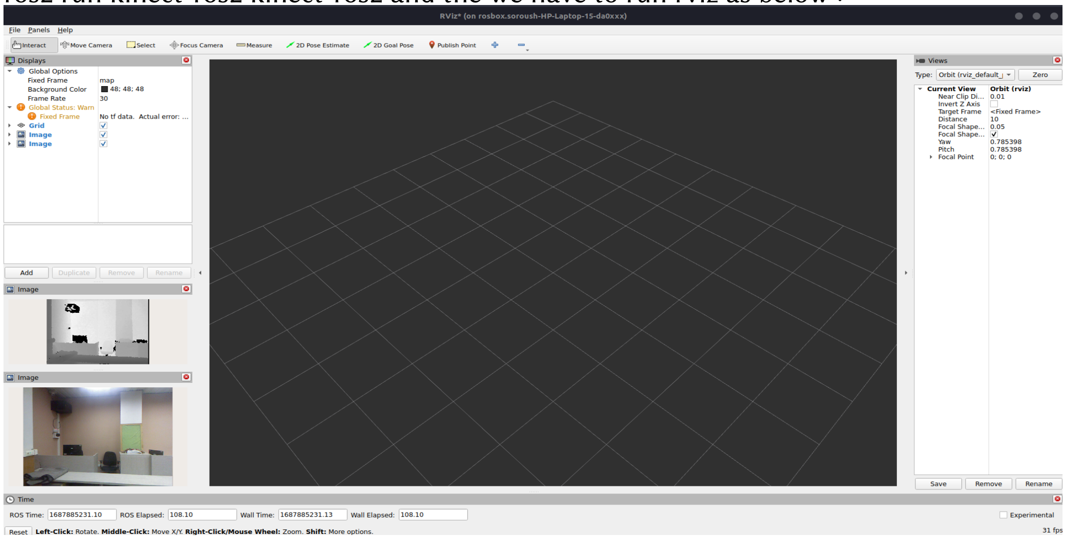

LibFreenect (on rosbox.soroush-HP-Laptop-15-da0xxx)
Write
Preview
Leave a comment
Attach files by dragging & dropping, selecting or pasting them.
Comment

```
write_register: 0x0012 <= 0x03
write_register: 0x0013 <= 0x01
write_register: 0x0014 <= 0x1e
write_register: 0x0006 <= 0x02
write_register: 0x0017 <= 0x00
[Stream 80] Negotiated packet size 1920
write_register: 0x000c <= 0x00
write_register: 0x000d <= 0x01
write_register: 0x000e <= 0x1e
write_register: 0x0005 <= 0x01
[Stream 70] Lost 74 total packets in 0 frames (inf lppf)
[Stream 70] Lost 86 total packets in 0 frames (inf lppf)
write_register: 0x0047 <= 0x00
'w' - tilt up, 's' - level, 'x' - tilt down, '0'-'6' - select LED mode, '+' & '-' - change IR intensity
'f' - change video format, 'm' - mirror video, 'o' - rotate video with accelerometer
'e' - auto exposure, 'b' - white balance, 'r' - raw color, 'n' - near mode (K4W only)
 raw acceleration:   29  839   -7  mks acceleration: 0.347244 10.046129 -0.083818
```

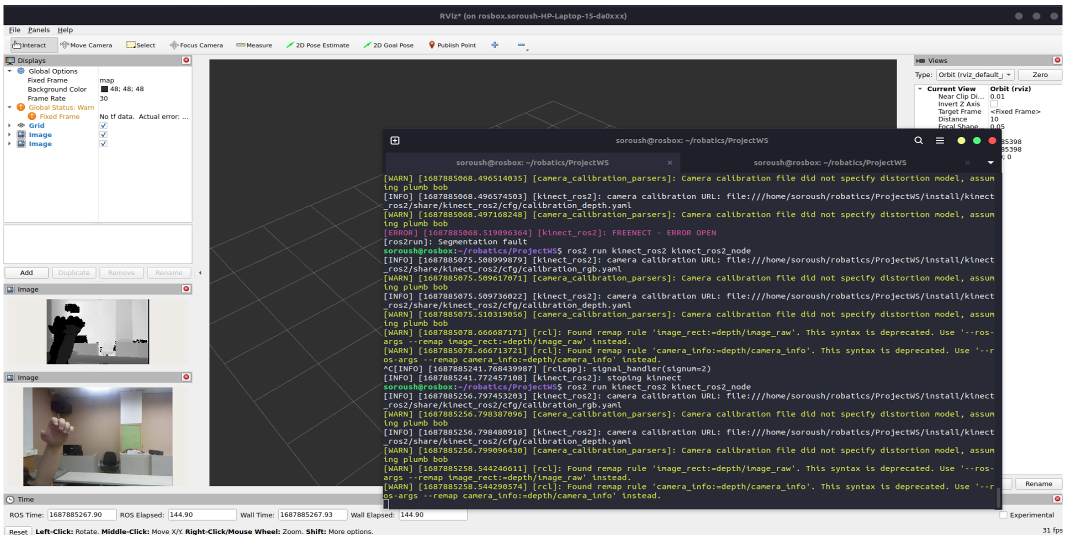
as you see in the picture we have 2 part the first one shows the depth of the objects and right one depicted a RGB picture .
one of the problems that i have faced it was that my laptap couldent connect to the kinnect , specially it dosnt aloow whane i wanted to connect it and i wrote a command and i gave it the access of 777 and the i overcame the problem .

Bringup the kinect

After the installation of drivers and clone the related repositories noe we have to set up the system and use it .
firstly we have to connect the kinect to laptop and then open a terminal and source the dependencies . now we are able to brig up the kinect :
ros2 run kinect-ros2 kinect-ros2 and the we have to run rviz as below :

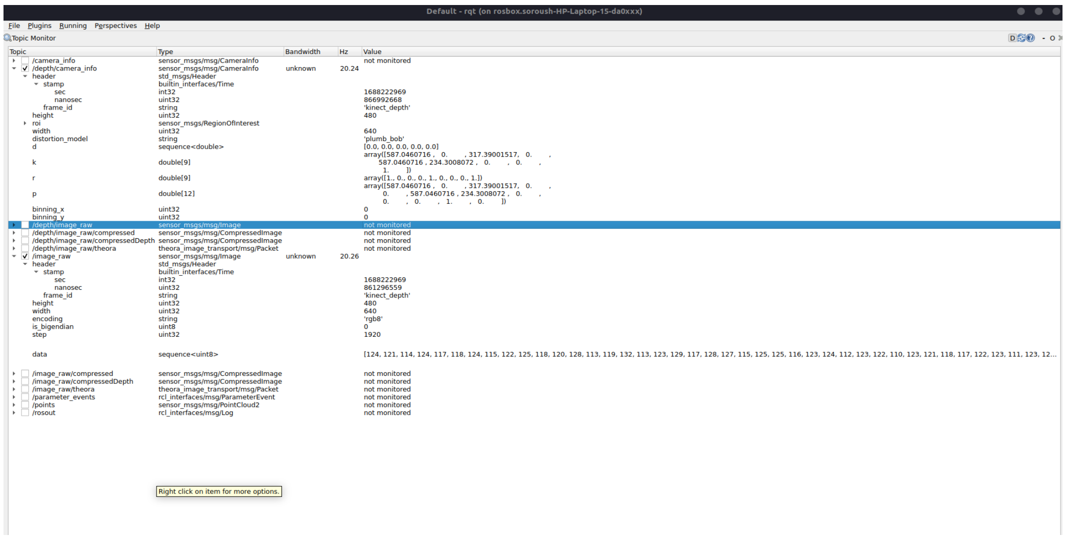


we can see the RGBD picture as mentioned befor .


RViz* (on rosbox.soroush-HP-Laptop-15-da0xxx)
soroush@rosbox: ~/robatics/ProjectWS
soroush@rosbox: ~/robatics/ProjectWS
soroush@rosbox: ~/robatics/ProjectWS
ROS Time: 1687885267.90
ROS Elapsed: 144.90
Wall Time: 1687885267.93
Wall Elapsed: 144.90
31 fps

```
[WARN] [1687885068.496514035] [camera_calibration_parsers]: Camera calibration file did not specify distortion model, assum
ing plumb bob
[INFO] [1687885068.496574503] [kinect_ros2]: camera calibration URL: file:///home/soroush/robatics/ProjectWS/install/kinect
_ros2/share/kinect_ros2/cfg/calibration_depth.yaml
[WARN] [1687885068.497168248] [camera_calibration_parsers]: Camera calibration file did not specify distortion model, assum
ing plumb bob
[ERROR] [1687885068.519096364] [kinect_ros2]: FREENECT - ERROR OPEN
[ros2run]: Segmentation fault
soroush@rosbox:~/robatics/ProjectWS$ ros2 run kinect_ros2 kinect_ros2_node
[INFO] [1687885075.508999879] [kinect_ros2]: camera calibration URL: file:///home/soroush/robatics/ProjectWS/install/kinect
_ros2/share/kinect_ros2/cfg/calibration_rgb.yaml
[WARN] [1687885075.509617071] [camera_calibration_parsers]: Camera calibration file did not specify distortion model, assum
ing plumb bob
[INFO] [1687885075.509736022] [kinect_ros2]: camera calibration URL: file:///home/soroush/robatics/ProjectWS/install/kinect
_ros2/share/kinect_ros2/cfg/calibration_depth.yaml
[WARN] [1687885075.510319056] [camera_calibration_parsers]: Camera calibration file did not specify distortion model, assum
ing plumb bob
[WARN] [1687885078.666687171] [rcl]: Found remap rule 'image_rect:=depth/image_raw'. This syntax is deprecated. Use '--ros-
args --remap image_rect:=depth/image_raw' instead.
[WARN] [1687885078.666713721] [rcl]: Found remap rule 'camera_info:=depth/camera_info'. This syntax is deprecated. Use '--r
os-args --remap camera_info:=depth/camera_info' instead.
^C[INFO] [1687885241.768439987] [rclcpp]: signal_handler(signum=2)
[INFO] [1687885241.772457108] [kinect_ros2]: stoping kinnect
soroush@rosbox:~/robatics/ProjectWS$ ros2 run kinect_ros2 kinect_ros2_node
[INFO] [1687885256.797453203] [kinect_ros2]: camera calibration URL: file:///home/soroush/robatics/ProjectWS/install/kinect
_ros2/share/kinect_ros2/cfg/calibration_rgb.yaml
[WARN] [1687885256.798387096] [camera_calibration_parsers]: Camera calibration file did not specify distortion model, assum
ing plumb bob
[INFO] [1687885256.798480918] [kinect_ros2]: camera calibration URL: file:///home/soroush/robatics/ProjectWS/install/kinect
_ros2/share/kinect_ros2/cfg/calibration_depth.yaml
[WARN] [1687885256.799096430] [camera_calibration_parsers]: Camera calibration file did not specify distortion model, assum
ing plumb bob
[WARN] [1687885258.544246611] [rcl]: Found remap rule 'image_rect:=depth/image_raw'. This syntax is deprecated. Use '--ros-
args --remap image_rect:=depth/image_raw' instead.
[WARN] [1687885258.544290574] [rcl]: Found remap rule 'camera_info:=depth/camera_info'. This syntax is deprecated. Use '--r
os-args --remap camera_info:=depth/camera_info' instead.
```

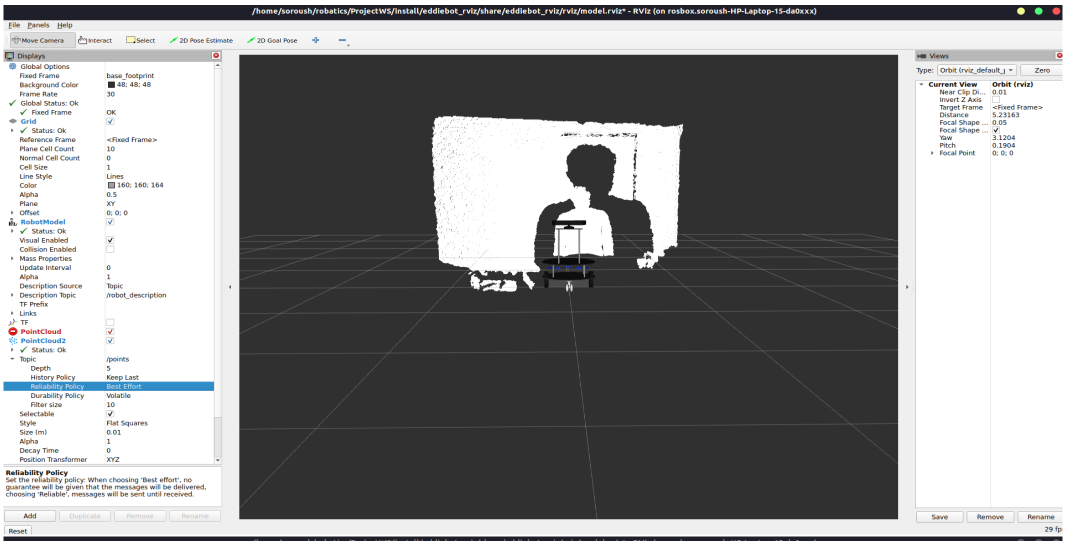
we can also open qrt and check the timestamps as below :


Default - rqt (on rosbox.soroush-HP-Laptop-15-da0xxx)
File Plugins Running Perspectives Help
Topic Monitor
Right click on item for more options.

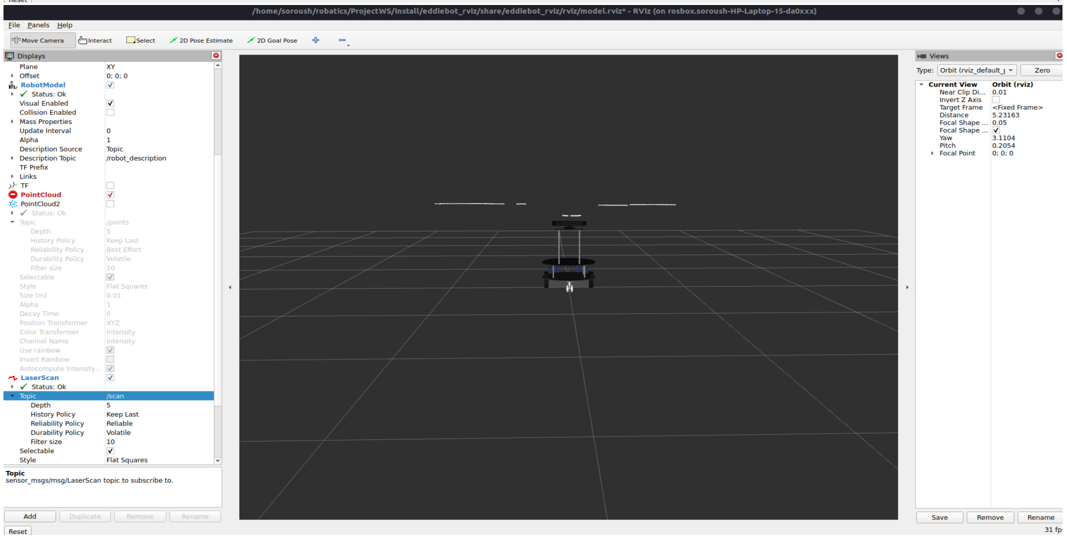
```
Topic                                Type                                  Bandwidth  Hz     Value
/camera_info                         sensor_msgs/msg/CameraInfo                              not monitored
✓ /depth/camera_info                 sensor_msgs/msg/CameraInfo            unknown    20.24
  header                             std_msgs/Header
    stamp                            builtin_interfaces/Time
      sec                            int32                                                   1688222969
      nanosec                        uint32                                                  866992668
    frame_id                         string                                                  'kinect_depth'
  height                             uint32                                                  480
  roi                                sensor_msgs/RegionOfInterest
  width                              uint32                                                  640
  distortion_model                   string                                                  'plumb_bob'
  d                                  sequence<double>                                        [0.0, 0.0, 0.0, 0.0, 0.0]
                                                                                             array([587.0460716 ,   0.        , 317.39001517,   0.        ,
  k                                  double[9]                                                  587.0460716 , 234.3008072 ,   0.        ,   0.        ,
                                                                                                  1.        ])
  r                                  double[9]                                               array([1., 0., 0., 0., 1., 0., 0., 0., 1.])
                                                                                             array([587.0460716 ,   0.        , 317.39001517,   0.        ,
  p                                  double[12]                                                   0.        , 587.0460716 , 234.3008072 ,   0.        ,
                                                                                                  0.        ,   0.        ,   1.        ,   0.        ])
  binning_x                          uint32                                                  0
  binning_y                          uint32                                                  0
/depth/image_raw                     sensor_msgs/msg/Image                                   not monitored
/depth/image_raw/compressed          sensor_msgs/msg/CompressedImage                         not monitored
/depth/image_raw/compressedDepth     sensor_msgs/msg/CompressedImage                         not monitored
/depth/image_raw/theora              theora_image_transport/msg/Packet                       not monitored
✓ /image_raw                         sensor_msgs/msg/Image                 unknown    20.26
  header                             std_msgs/Header
    stamp                            builtin_interfaces/Time
      sec                            int32                                                   1688222969
      nanosec                        uint32                                                  861296559
    frame_id                         string                                                  'kinect_depth'
  height                             uint32                                                  480
  width                              uint32                                                  640
  encoding                           string                                                  'rgb8'
  is_bigendian                       uint8                                                   0
  step                               uint32                                                  1920

  data                               sequence<uint8>                                         [124, 121, 114, 124, 117, 118, 124, 115, 122, 125, 118, 120, 128, 113, 119, 132, 113, 123, 129, 117, 128, 127, 115, 125, 125, 116, 123, 124, 112, 123, 122, 110, 123, 121, 118, 117, 122, 123, 111, 123, 12...

/image_raw/compressed                sensor_msgs/msg/CompressedImage                         not monitored
/image_raw/compressedDepth           sensor_msgs/msg/CompressedImage                         not monitored
/image_raw/theora                    theora_image_transport/msg/Packet                       not monitored
/parameter_events                    rcl_interfaces/msg/ParameterEvent                       not monitored
/points                              sensor_msgs/msg/PointCloud2                             not monitored
/rosout                              rcl_interfaces/msg/Log                                  not monitored
```

Bringup the Eddie

to do this firstly we set command as below :
"sudo chmod a+rw /dev/ttyUSB0"
and then we set :
"ros2 launch eddiebot-bringup eddie.launch.yaml "
so we run the launch file to run eddiebot-bringup package whtch can sent the commands for the framework .and it gives us the abstract command and convert it to eddies command that mentioned in docs .
next we set :
"ros2 launch eddiebot-nav eddiebot.launch.py "
that This should start the nodes for Kinect, odometry, vel-controller, some static transforms, and fake laser scan.
and we have to write :
ros2 run teleop$_{t}wist_{k}eyboardteleop_{t}wist_{k}eyboard--ros-args-r/cmd_{v}el := /eddie/cmd-$
vel
$andthenamovementwillappearinEddie.$
$herewecanadjustthespeedofediieandcontrollitviathegivencommands:$



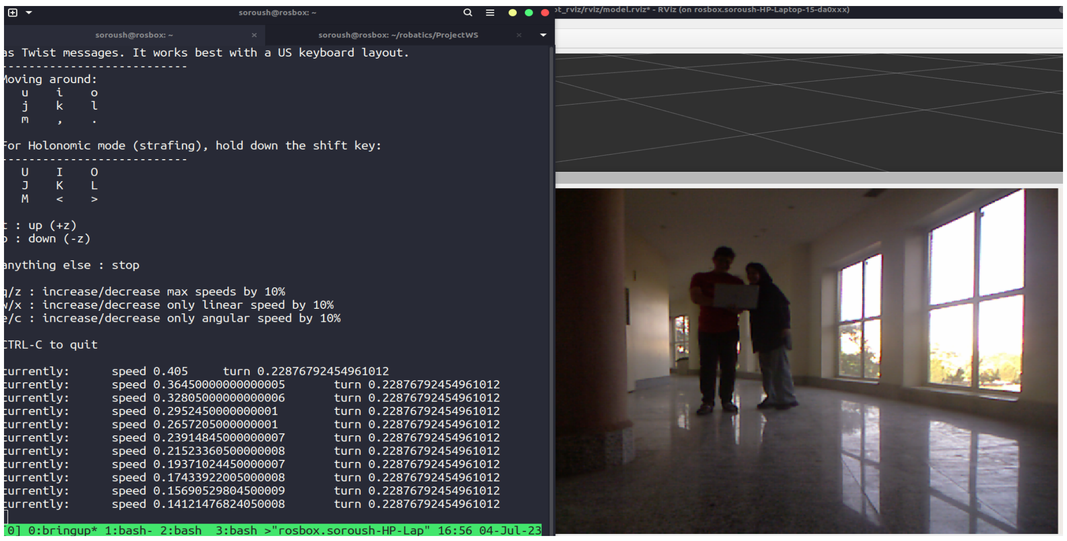
```
/home/soroush/robatics/ProjectWS/install/eddiebot_rviz/share/eddiebot_rviz/rviz/model.rviz* - RViz (on rosbox.soroush-HP-Laptop-15-da0xxx)

Displays
Global Options
  Fixed Frame               base_footprint
  Background Color          48; 48; 48
  Frame Rate                30
✓ Global Status: Ok
  ✓ Fixed Frame             OK
Grid                        ✓
  ✓ Status: Ok
  Reference Frame           <Fixed Frame>
  Plane Cell Count          10
  Normal Cell Count         0
  Cell Size                 1
  Line Style                Lines
  Color                     160; 160; 164
  Alpha                     0.5
  Plane                     XY
  Offset                    0; 0; 0
RobotModel                  ✓
  ✓ Status: Ok
  Visual Enabled            ✓
  Collision Enabled         ☐
  Mass Properties
  Update Interval           0
  Alpha                     1
  Description Source        Topic
  Description Topic         /robot_description
  TF Prefix
  Links
TF                          ☐
PointCloud                  ✓
PointCloud2                 ✓
  ✓ Status: Ok
  Topic                     /points
    Depth                   5
    History Policy          Keep Last
    Reliability Policy      Best Effort
    Durability Policy       Volatile
    Filter size             10
  Selectable                ✓
  Style                     Flat Squares
  Size (m)                  0.01
  Alpha                     1
  Decay Time                0
  Position Transformer      XYZ

Reliability Policy
Set the reliability policy: When choosing 'Best effort', no
guarantee will be given that the messages will be delivered,
choosing 'Reliable', messages will be sent until received.

Views
Type: Orbit (rviz_default_
Current View                Orbit (rviz)
  Near Clip Di...           0.01
  Invert Z Axis             ☐
  Target Frame              <Fixed Frame>
  Distance                  5.23163
  Focal Shape ...           0.05
  Focal Shape ...           ✓
  Yaw                       3.1204
  Pitch                     0.1904
  Focal Point               0; 0; 0
```



```
/home/soroush/robatics/ProjectWS/install/eddiebot_rviz/share/eddiebot_rviz/rviz/model.rviz* - RViz (on rosbox.soroush-HP-Laptop-15-da0xxx)

Displays
  Plane                     XY
  Offset                    0; 0; 0
RobotModel                  ✓
  ✓ Status: Ok
  Visual Enabled            ✓
  Collision Enabled         ☐
  Mass Properties
  Update Interval           0
  Alpha                     1
  Description Source        Topic
  Description Topic         /robot_description
  TF Prefix
  Links
TF                          ☐
PointCloud                  ✓
PointCloud2                 ☐
  ✓ Status: Ok
  Topic                     /points
    Depth                   5
    History Policy          Keep Last
    Reliability Policy      Best Effort
    Durability Policy       Volatile
    Filter size             10
  Selectable                ✓
  Style                     Flat Squares
  Size (m)                  0.01
  Alpha                     1
  Decay Time                0
  Position Transformer      XYZ
  Color Transformer         Intensity
  Channel Name              intensity
  Use rainbow               ✓
  Invert Rainbow            ☐
  Autocompute Intensity...  ✓
LaserScan                   ✓
  ✓ Status: Ok
  Topic                     /scan
    Depth                   5
    History Policy          Keep Last
    Reliability Policy      Reliable
    Durability Policy       Volatile
    Filter size             10
  Selectable                ✓
  Style                     Flat Squares

Topic
sensor_msgs/msg/LaserScan topic to subscribe to.

Views
Type: Orbit (rviz_default_
Current View                Orbit (rviz)
  Near Clip Di...           0.01
  Invert Z Axis             ☐
  Target Frame              <Fixed Frame>
  Distance                  5.23163
  Focal Shape ...           0.05
  Focal Shape ...           ✓
  Yaw                       3.1104
  Pitch                     0.2054
  Focal Point               0; 0; 0
```

Networking

in networking we want to connect each other as our goal is to move Eddie and change its position . for this firstly we have to be in the same network then the "ROS-DOMAIN-ID" has to be same . both of us assign 0 to this variable . but there was problem . we couldnt sent any message for each other. we set the command below and error solved .
$"exportROSLOCALHOSTONLY = 1"$.


soroush@rosbox: ~
soroush@rosbox: ~
soroush@rosbox: ~/robatics/ProjectWS
ot_rviz/rviz/model.rviz* - RViz (on rosbox.soroush-HP-Laptop-15-da0xxx)

```
as Twist messages. It works best with a US keyboard layout.
---------------------------
Moving around:
   u    i    o
   j    k    l
   m    ,    .

For Holonomic mode (strafing), hold down the shift key:
---------------------------
   U    I    O
   J    K    L
   M    <    >

t : up (+z)
b : down (-z)

anything else : stop

q/z : increase/decrease max speeds by 10%
w/x : increase/decrease only linear speed by 10%
e/c : increase/decrease only angular speed by 10%

CTRL-C to quit

currently:      speed 0.405     turn 0.22876792454961012
currently:      speed 0.36450000000000005       turn 0.22876792454961012
currently:      speed 0.32805000000000006       turn 0.22876792454961012
currently:      speed 0.2952450000000001        turn 0.22876792454961012
currently:      speed 0.2657205000000001        turn 0.22876792454961012
currently:      speed 0.23914845000000007       turn 0.22876792454961012
currently:      speed 0.21523360500000008       turn 0.22876792454961012
currently:      speed 0.19371024450000007       turn 0.22876792454961012
currently:      speed 0.17433922005000008       turn 0.22876792454961012
currently:      speed 0.15690529804500009       turn 0.22876792454961012
currently:      speed 0.14121476824050008       turn 0.22876792454961012

[0] 0:bringup* 1:bash- 2:bash  3:bash >"rosbox.soroush-HP-Lap" 16:56 04-Jul-23
```

after bringing up the robot we can use this command and contro the robot : ros2 run teleop twist keyboard teleop twist keyboard
and then we can remotly controll the eddie .

2D SLAM

in this section that associated to problem 5 we are going to use slam in our robot. to do this fistly we have to bring up Eddie . so firstly we run a launch eddie.launch.yaml in eddiebot-ros-bringup package, and then eddiebot.launch.py from eddiebot-ros-nav . and run :
ros2 launch eddiebot-rviz view-robot.launch.py to start rviz . after that we have to move the robot around the room and we can make our map .
now we have to set the command below to load the map in rviz :
ros2 service call /slam-toolbox/save-map slam-toolbox/srv/SaveMap "name: data: 'maze-tight'".
now if we put ros2 launch eddiebot-nav localization.launch.py we can localize the robot in the map and then we have to enter the start position of our robot and then enter the command below for nav2 to determine its goal . :
ros2 launch eddiebot-nav nav2.launch.py . now our robot using the given data and trt to planning ant compute the optimal path to goal, after that it starts to move toward ots goal .

Nav2

we adjustes some parameters in nav2.yaml file, to have a good performance . firstly we change the value of smoothing-frequency from 20 to 0.2 beacuse 20 is very large and framework can not compute and analyse the commands to find a good way. we also reduce the amount of theta and linear speed .to control it better .scale-velocity is changed to "True" .

Odometry

Motor Control

As each wheel spins the sensor will gather data about the angular position change of the wheel. Once the encoder input has been captured it must be converted to linear velocity and used by a robotic kinematic model to create an estimate of the distance traveled and possible error. Two wheel encoders are used to deduce the speed and travel direction by measuring the number of pulses registered from each wheel. Each wheel encoder has two sensors and is capable of registering a distance resolution of 1/36th of the robot's wheel circumference. The Position Controllers on Eddie use a quadrature encoder system to reliably track the position and speed of each wheel at all times. With the included plastic encoder disks, each Position Controller has a resolution of 36 positions per rotation; this equates to approximately 0.5 inches of linear travel per position using the included 6 inch tires. The Position Controllers calculate and report position and average speed data on command. Knowing that the sensor has 1/36th resolution of wheel circumference, the sensor produces 36 pulses for every complete revolution of the wheel. Based on this, the distance traveled in the duration of one pulse is given below:

$$d = \frac{2\pi r}{36} \tag{1}$$

```
// COUNTS_PER_REVOLUTION   36

// WHEEL_RADIUS    0.1524   Wheel radius in meters
// // the distance of a wheel move forward when encoder increased by 1
// DISTANCE_PER_COUNT      ((TWOPI * WHEEL_RADIUS) /
    COUNTS_PER_REVOLUTION)

// WHEEL_SEPARATION      0.3    two wheels center-to-center distance


// Called from the velocity callback.
// Set the values for the left and right wheels' speeds
// so that Eddie can do arcs
void EddieController::moveLinearAngular(float linear, float angular) {

  // Calculate wheel velocities and convert meter per second to position
    per second
  double left_command_speed  = ((linear - angular * WHEEL_SEPARATION /
    2.0) / WHEEL_RADIUS) / DISTANCE_PER_COUNT;
  double right_command_speed = ((linear + angular * WHEEL_SEPARATION /
    2.0) / WHEEL_RADIUS) / DISTANCE_PER_COUNT;

  sem_wait(&mutex_interrupt_);
  left_drive_speed  = left_command_speed;
  right_drive_speed = right_command_speed;
  // this is not a pure rotation
  rotate_ = false;
  process_ = true;
  // cancel other moving commands in favor of this new one.
  interrupt_ = true;
  sem_post(&mutex_interrupt_);
}
```

Listing 1: eddie controller

Wheel Odometry Model

With the rotation data, alongside information on the encoder, such as the radius or circumference, we can estimate the distance traveled by the wheel. Since each slit represents some angle of rotation, knowing the number of slits passed informs us about the amount of rotation between time steps. For an optical encoder, where all the slits are equally spaced, we could get the total angle of rotation between time steps by multiplying the number of slits passed by the amount of rotation represented by a single slit. After we determine the angle of rotation, we can multiply it by the circumference of the encoder to get the distance traveled by the wheel. The goal of an odometry model is to estimate the position and orientation of the robot. To achieve this, we'll leverage data from the rotary encoders, the dimensions of our robot, and geometry. The encoder will inform us of the distance traveled by each wheel at each time step. In terms of using the dimensions of our robot, the only dimension we need is the distance of the point from the left and right wheels. Since we defined our reference point to be located equidistant between the two wheels, we only need to keep track of one number.

Now let's define some variables to keep track of these ideas:

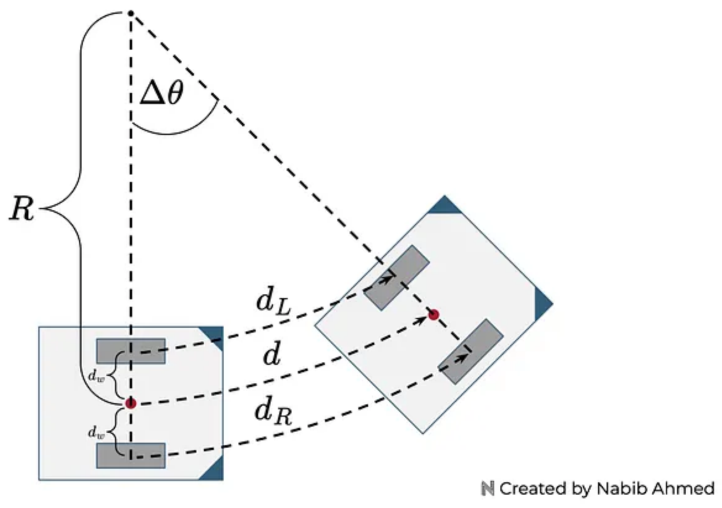


Figure 1

d_L = distance traveled by the left wheel.
d_R = distance traveled by the right wheel.
d_w = distance between the reference point and the wheels.

d = distance traveled by the reference point.
$\Delta\theta$ = A change in the angle of rotation.
R = radius of the curve containing the reference point.

d_L and d_R correspond to the distance traveled by the wheel at a certain time step. This information will come from our rotary encoder. d_w can be derived by measuring the distance between the two wheels and dividing it in half since the point is equidistant from the two wheels. The last three variables are not directly measurable — instead, we need to use geometry to relate these variables to the measurable quantities. We can start by using the arc length formula. The path for the left wheel, right wheel, and reference points are arcs. They all share the same angle and the radius for each can be expressed in terms of the radius of the curve containing the reference point and the distance between the reference point and the wheels.

$$d = R\Delta\theta \tag{2}$$

$$d_L = (R - d_w)\Delta\theta \tag{3}$$

$$d_R = (R + d_w)\Delta\theta \tag{4}$$

Now we solve for the change in the angle of rotation in terms of measurable quantities, getting the following relationship:

$$\Delta\theta = \frac{d_R - d_L}{2d_w} \tag{5}$$

Now we solve for the radius of the curve containing the reference point by rearranging equations and plugging in what we know. Then, we solve for the distance travelled by the reference point.

$$R = d_L\frac{2d_w}{d_R - d_L} + d_w \tag{6}$$

$$d = \frac{d_R + d_L}{2} \tag{7}$$

We solved for all variables in terms of measurable quantities. Since we're interested in the robot's position and orientation, the key variables of interest would be the distance traveled by the reference point and the change in the angle of rotation. The distance traveled by the reference point informs us of the position and the change in the angle of rotation informs us of the orientation. The radius of the curve containing the reference point, while useful for derivation, is not really needed anymore.
Now, we know the distance traveled, but not the direction. We know how much the orientation angle changed, but not the new orientation angle. So we start modifying our odometry model.
To simplify our model, we will represent the distance traveled by the reference point as a line instead of a curve (as shown in Figure We can make this simplification because typically, in wheel odometry with encoders, the data sampling is very high. What this means is that our encoders are able to collect data very frequently so the time window between measurements is very small. Since the time window is very small, the amount of motion captured by each time step will also be small. For our model, that means the curvature of the arc will be very

small and resemble a straight line. Thus, it's a safe assumption and simplification to represent our distance now as a straight line. We then calculate the angle of the distance in terms of a previously solved variable, and the new orientation of the robot as shown in Figure 2:

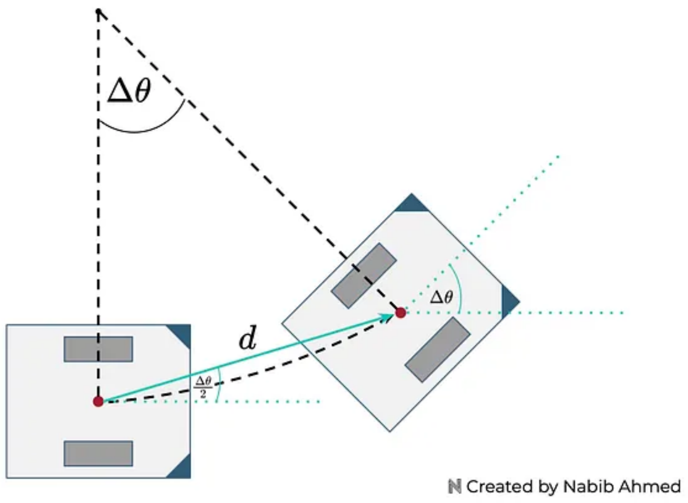


Figure 2

In Figure 3, the odometry model at time t will add the absolute orientation angle from the previous time step. Notice that adding the orientation from the previous time step won't change the distance traveled by the reference point or the change in angle of rotation as the formulas we derived from earlier don't rely on the orientation angle (only the traveled wheel distances). Instead what does change is the orientation of the robot, from being relative between time steps to now being absolute on the coordinate plane. Thus, the absolute orientation angle at any time step can be defined by:

$$\theta_t = \theta_{t-1} + \Delta\theta_t \tag{8}$$

Using the distance traveled by the reference point and the angle of orientation from the previous time step plus the angle that results from motion, the amount of distance traveled along the x and y directions can be calculated.

$$x_t = x_{t-1} + dcos(\theta_{t-1} + \frac{\Delta\theta_t}{2}) \tag{9}$$

$$y_t = y_{t-1} + dsin(\theta_{t-1} + \frac{\Delta\theta_t}{2}) \tag{10}$$

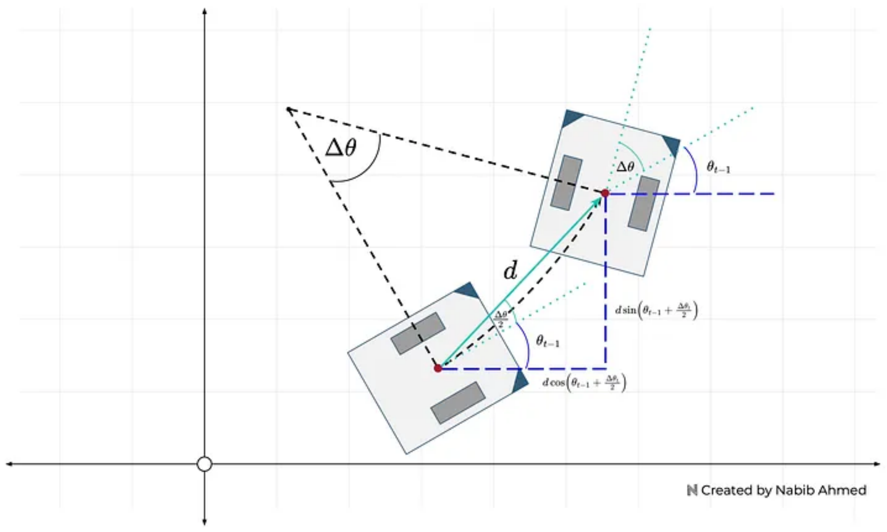

$\Delta\theta$
$\Delta\theta$
θ_{t-1}
d
$\frac{\Delta\theta}{2}$
θ_{t-1}
$d\sin\left(\theta_{t-1}+\frac{\Delta\theta_t}{2}\right)$
$d\cos\left(\theta_{t-1}+\frac{\Delta\theta_t}{2}\right)$
Created by Nabib Ahmed

Figure 3

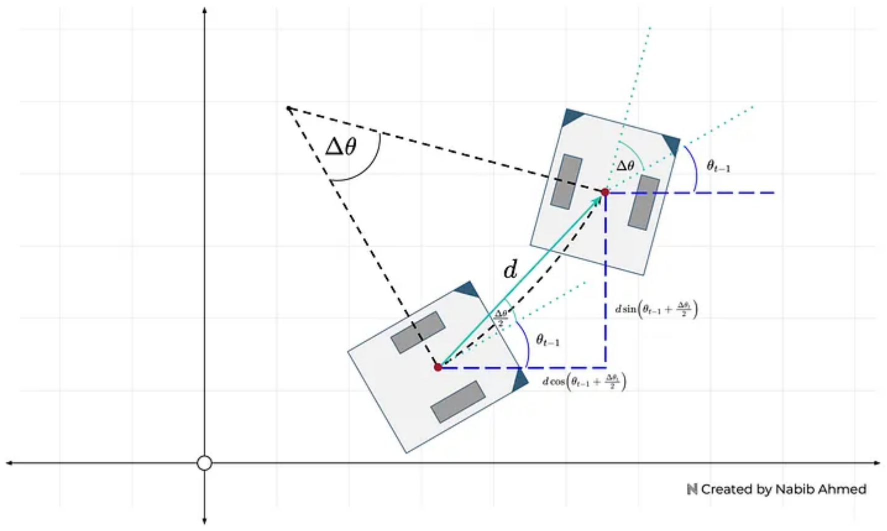

$\Delta\theta$
$\Delta\theta$
θ_{t-1}
d
$\frac{\Delta\theta}{2}$
θ_{t-1}
$d\sin\left(\theta_{t-1}+\frac{\Delta\theta_t}{2}\right)$
$d\cos\left(\theta_{t-1}+\frac{\Delta\theta_t}{2}\right)$
Created by Nabib Ahmed

Figure 4

VSLAM

RTAB-Map (Real-Time Appearance-Based Mapping) is a RGB-D Graph SLAM approach based on a global Bayesian loop closure detector. The loop closure detector uses a bag-of-words approach to determinate how likely a new image comes from a previous location or a new location. When a loop closure hypothesis is accepted, a new constraint is added to the map's graph, then a graph optimizer minimizes the errors in the map. and one its parameters is ’tf$_{d}elay' : 4.0$,