# 2D-3D LiDAR-Camera Calibration

Team Information:
1) Vineeth Bhat (2021101103)
2) P Gnana Prakash (2021111027)
3) Mitansh Kayathwal (2021101026)

# 2D LiDAR Camera Calibration

# Literature Review

- Fiduciary based 2D lidar-camera based calibration (checkerboard)
  - Plan to use improved, modern methods
- Addressing drawbacks in 2D LiDAR:
  - sensitivity to checkerboard poses, laser points on the checkerboard are obtained from both the originally positioned LiDAR and the same LiDAR vertically rotated from the original position, in the proposed calibration
  - construct the additional constraints with the normal vector of the checkerboard, and strengthen the basic point-on-plane constraints in the LiDAR calibration
- [~ 5 hours]

# Problem Formulation

$$\arg\min_{R_{lc},t_{lc}} \sum_{i=1}^{N_b} \frac{1}{N_i} \sum_{j=1}^{N_i} \left( \binom{r_i^3}{T} (R_{lc} P_{i,j} l + t_{lc}) - \binom{r_i^3}{T} t_i \right)^2$$

Here, ri(3) is obtained from Rotation Matrix Rl and the number of target poses is denoted as Nb. Geometrically, the target plane's normal vector is represented by the vector ri3, and the position of the target plane's origin, or the origin of the world coordinate system, in the camera coordinate system is represented by the vector ti.

$$P_{i,j}^c = R_{lc} P_{i,j} l + t_{lc}$$

Here, Pc(i,j) represents the jth LiDAR point on the target in the ith pose.

$$N_i^C = R_{lc} N_i^L$$

Here, NC(i) represents the Normal vector of the target plane in the ith target plane.

$$\arg\min_{R_{lc}} \sum_{i=1}^{N_b} \left\| r_i^3 - R_{lc} N_i^L \right\|_2^2$$

$$\arg\min_{R_{lc},t_{lc}} \sum_{i=1}^{2N_b} \left(\frac{1}{N_i}\right)^p \sum_{j=1}^{N_i} \left(\left(\frac{r_i^3}{T}\right)(R_{lc}P_{i,j}l + t_{lc}) - \left(\frac{r_i^3}{T}\right)t_i\right)^2$$

The laser points in the vertically rotated LiDAR coordinate system are also effective in the basic point-on-plane constraints. Since the vertical laser points on the target cannot be obtained from LiDAR measurement in the original LiDAR coordinate system. Thus,
in we utilize the vertical laser points as above.

$$\arg\min_{R_{lc},t_{lc}} \left(\sum_{i=1}^{2N_b} \left(\frac{1}{N_i}\right)^p \sum_{j=1}^{N_i} \left(\left(\frac{r_i^3}{T}\right)(R_{lc}P_{i,j}l + t_{lc}) - \left(\frac{r_i^3}{T}\right)t_i\right)^2 + \sum_{i=1}^{N_b} \left\|r_i^3 - R_{lc}N_i^L\right\|_2^2\right)$$

We conduct the non-separate estimation of the LiDAR extrinsic rotation and translation parameters by eventually adding the previous optimization equations.

$$P_{i,j}^l = R_V P_{i-N_b,j}^{lv} + t_l$$

Final modified equation

# 3D LiDAR Camera Calibration

# Literature Review

- Use adaptable method
  - Works without extensive 3D geometry understanding
- Calls for the camera to focus on a planar pattern that is displayed in at least two distinct directions

*You can easily move either the camera or the planar pattern. One need not be aware of the motion. It models radial lens distortion. The suggested method starts with a closed-form solution and then refines nonlinearly using the maximum likelihood criterion.*

- [~ 3 hours]

# Problem Formulation

$$s\tilde{\mathbf{m}} = \mathbf{A}\left[\mathbf{R} \quad \mathbf{t}\right]\widetilde{\mathbf{M}}$$

, where s is an arbitrary scaling factor
$R_{3x3}$ is the rotation estimate,
$t_{3x1}$ is the translation estimate

$$\mathbf{m} = [u, v]^T$$

$$\tilde{\mathbf{m}} = [u, v, 1]^T$$

$$\mathbf{M} = [X, Y, Z]^T$$

$$\widetilde{\mathbf{M}} = [X, Y, Z, 1]^T$$

$$\mathbf{A} = \begin{bmatrix} \alpha & \gamma & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{bmatrix}$$

$\mathbf{m}_{2x1}$ is the 2D image coordinate

$\mathbf{M}_{3x1}$ is the 3D world coordinate

$\mathbf{A}_{3x3}$ is the camera intrinsics matrix

$\tilde{\mathbf{m}}_{3x1}$ is the augmented 2D vector

$\widetilde{\mathbf{M}}_{.x1}$ is the augmented 3D vector

# Problem Formulation

$$s\tilde{\mathbf{m}} = \mathbf{A}[\mathbf{R} \quad \mathbf{t}]\widetilde{\mathbf{M}}$$

, where s is an arbitrary scaling factor
$R_{3x3}$ is the rotation estimate,
$t_{3x1}$ is the translation estimate

Our task is to estimate **R** and **t** using correspondences between the camera and LiDAR across various observed points

# Intrinsic Parameter Constraints

We assume the world coordinate frame is on **Z = 0**

$$s \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \mathbf{A} [\mathbf{r}_1 \quad \mathbf{r}_2 \quad \mathbf{r}_3 \quad \mathbf{t}] \begin{bmatrix} X \\ Y \\ 0 \\ 1 \end{bmatrix} = \mathbf{A} [\mathbf{r}_1 \quad \mathbf{r}_2 \quad \mathbf{t}] \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix}$$

$$s\tilde{\mathbf{m}} = \mathbf{H}\widetilde{\mathbf{M}}$$

where $\mathbf{r_i}$ is the $\mathbf{i^{th}}$ column of $\mathbf{R_{3x3}}$

3x1

$$\mathbf{H}_{3x3} = \mathbf{A} [\mathbf{r}_1 \quad \mathbf{r}_2 \quad \mathbf{t}] = [\mathbf{h}_1 \quad \mathbf{h}_2 \quad \mathbf{h}_3] = \lambda \mathbf{A} [\mathbf{r}_1 \quad \mathbf{r}_2 \quad \mathbf{t}]$$

# Intrinsic Parameter Constraints

We assume the world coordinate frame is on **Z = 0**

$$s\begin{bmatrix}u\\v\\1\end{bmatrix} = \mathbf{A}\begin{bmatrix}\mathbf{r}_1 & \mathbf{r}_2 & \mathbf{r}_3 & \mathbf{t}\end{bmatrix}\begin{bmatrix}X\\Y\\0\\1\end{bmatrix} = \mathbf{A}\begin{bmatrix}\mathbf{r}_1 & \mathbf{r}_2 & \mathbf{t}\end{bmatrix}\begin{bmatrix}X\\Y\\1\end{bmatrix}$$

$$s\tilde{\mathbf{m}} = \mathbf{H}\widetilde{\mathbf{M}}$$

where $\mathbf{r}_{i\,3\times1}$ is the $\mathbf{i}^{\text{th}}$ column of $\mathbf{R}_{3\times3}$

$$\mathbf{H}_{3\times3} = \mathbf{A}\begin{bmatrix}\mathbf{r}_1 & \mathbf{r}_2 & \mathbf{t}\end{bmatrix} = \begin{bmatrix}\mathbf{h}_1 & \mathbf{h}_2 & \mathbf{h}_3\end{bmatrix} = \lambda\mathbf{A}\begin{bmatrix}\mathbf{r}_1 & \mathbf{r}_2 & \mathbf{t}\end{bmatrix}$$

We also obtain the following equations from the fact that $r_1$ and $r_2$ are orthogonal

$$\mathbf{h}_1^T\mathbf{A}^{-T}\mathbf{A}^{-1}\mathbf{h}_2 = 0$$

$$\mathbf{h}_1^T\mathbf{A}^{-T}\mathbf{A}^{-1}\mathbf{h}_1 = \mathbf{h}_2^T\mathbf{A}^{-T}\mathbf{A}^{-1}\mathbf{h}_2.$$

# Closed Form Solution

$$\mathbf{B} = \mathbf{A}^{-T}\mathbf{A}^{-1} \equiv \begin{bmatrix} B_{11} & B_{12} & B_{13} \\ B_{12} & B_{22} & B_{23} \\ B_{13} & B_{23} & B_{33} \end{bmatrix} = \begin{bmatrix} \frac{1}{\alpha^2} & -\frac{\gamma}{\alpha^2\beta} & \frac{v_0\gamma - u_0\beta}{\alpha^2\beta} \\ -\frac{\gamma}{\alpha^2\beta} & \frac{\gamma^2}{\alpha^2\beta^2} + \frac{1}{\beta^2} & -\frac{\gamma(v_0\gamma - u_0\beta)}{\alpha^2\beta^2} - \frac{v_0}{\beta^2} \\ \frac{v_0\gamma - u_0\beta}{\alpha^2\beta} & -\frac{\gamma(v_0\gamma - u_0\beta)}{\alpha^2\beta^2} - \frac{v_0}{\beta^2} & \frac{(v_0\gamma - u_0\beta)^2}{\alpha^2\beta^2} + \frac{v_0^2}{\beta^2} + 1 \end{bmatrix}$$

$$\mathbf{b} = [B_{11}, B_{12}, B_{22}, B_{13}, B_{23}, B_{33}]^T$$

This vector **b** represents the symmetric matrix **B**

$$\mathbf{h}_i^T \mathbf{B} \mathbf{h}_j = \mathbf{v}_{ij}^T \mathbf{b} = [h_{i1}h_{j1}, h_{i1}h_{j2} + h_{i2}h_{j1}, h_{i2}h_{j2}, h_{i3}h_{j1} + h_{i1}h_{j3}, h_{i3}h_{j2} + h_{i2}h_{j3}, h_{i3}h_{j3}]\mathbf{b}$$

$$\begin{bmatrix} \mathbf{v}_{12}^T \\ (\mathbf{v}_{11} - \mathbf{v}_{22})^T \end{bmatrix} \mathbf{b} = \mathbf{0}$$

for one point

$$\mathbf{V}\mathbf{b} = \mathbf{0}$$

for n points, where **V** is obtained by stacking the previously obtained equation for each of the n points

# Closed Form Solution

- The equation **Vb = 0** can be solved using SVD, from which we obtain **b** and hence, the matrix **B**
- From this matrix **B**, we can calculate **A**, the intrinsics matrix using the formula **B = λ$\mathbf{A}^{-T}$A**
- We can extract these parameters using the following equations:

$$
\begin{aligned}
v_0 &= (B_{12}B_{13} - B_{11}B_{23})/(B_{11}B_{22} - B_{12}^2) \\
\lambda &= B_{33} - [B_{13}^2 + v_0(B_{12}B_{13} - B_{11}B_{23})]/B_{11} \\
\alpha &= \sqrt{\lambda/B_{11}} \\
\beta &= \sqrt{\lambda B_{11}/(B_{11}B_{22} - B_{12}^2)} \\
\gamma &= -B_{12}\alpha^2\beta/\lambda \\
u_0 &= \gamma v_0/\alpha - B_{13}\alpha^2/\lambda.
\end{aligned}
$$

# Closed Form Solution

- Once we have the intrinsics matrix **A**, we can obtain the extrinsics parameters using the following equations:

$$\mathbf{r}_1 = \lambda \mathbf{A}^{-1}\mathbf{h}_1, \quad \mathbf{r}_2 = \lambda \mathbf{A}^{-1}\mathbf{h}_2, \quad \mathbf{r}_3 = \mathbf{r}_1 \times \mathbf{r}_2, \quad \mathbf{t} = \lambda \mathbf{A}^{-1}\mathbf{h}_3$$

where $\lambda = 1/\|\mathbf{A}^{-1}\mathbf{h}_1\| = 1/\|\mathbf{A}^{-1}\mathbf{h}_2\|$

- Finally, using these estimated parameters as our initial estimate, we apply the Levenberg-Marquardt optimisation to minimise the following cost function to get the best estimates of the parameters **A**, $\mathbf{R}_i$ and $\mathbf{t}_i$, where $\mathbf{i = \{1, 2, ..., n\}}$

$$\sum_{i=1}^{n}\sum_{j=1}^{m} \|\mathbf{m}_{ij} - \hat{\mathbf{m}}(\mathbf{A}, \mathbf{R}_i, \mathbf{t}_i, \mathbf{M}_j)\|^2$$

# Ros Simulations

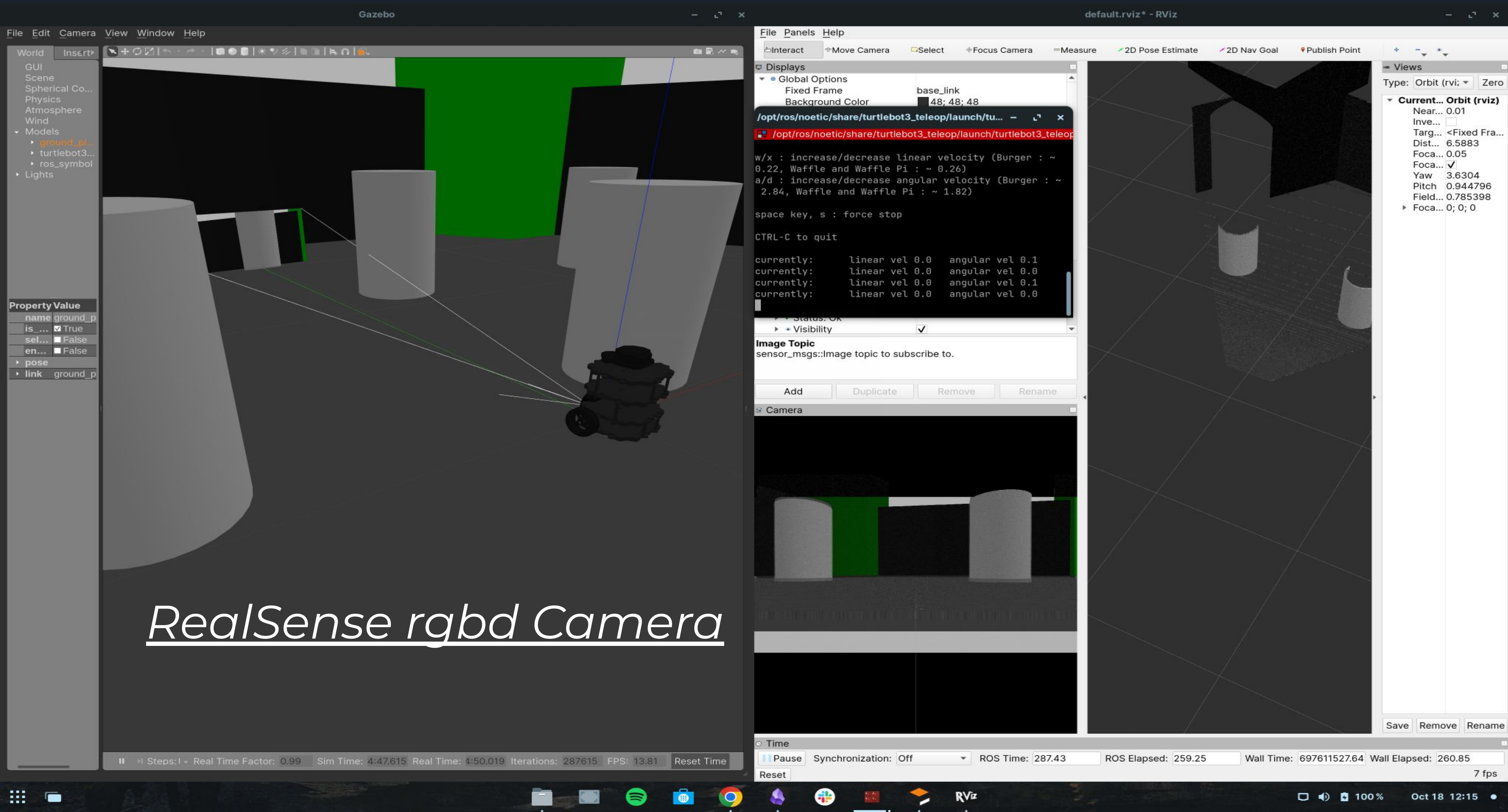
- We learnt ROS *(ros-simulations/basic_ros_running_instructions.md)* [~5 hours]

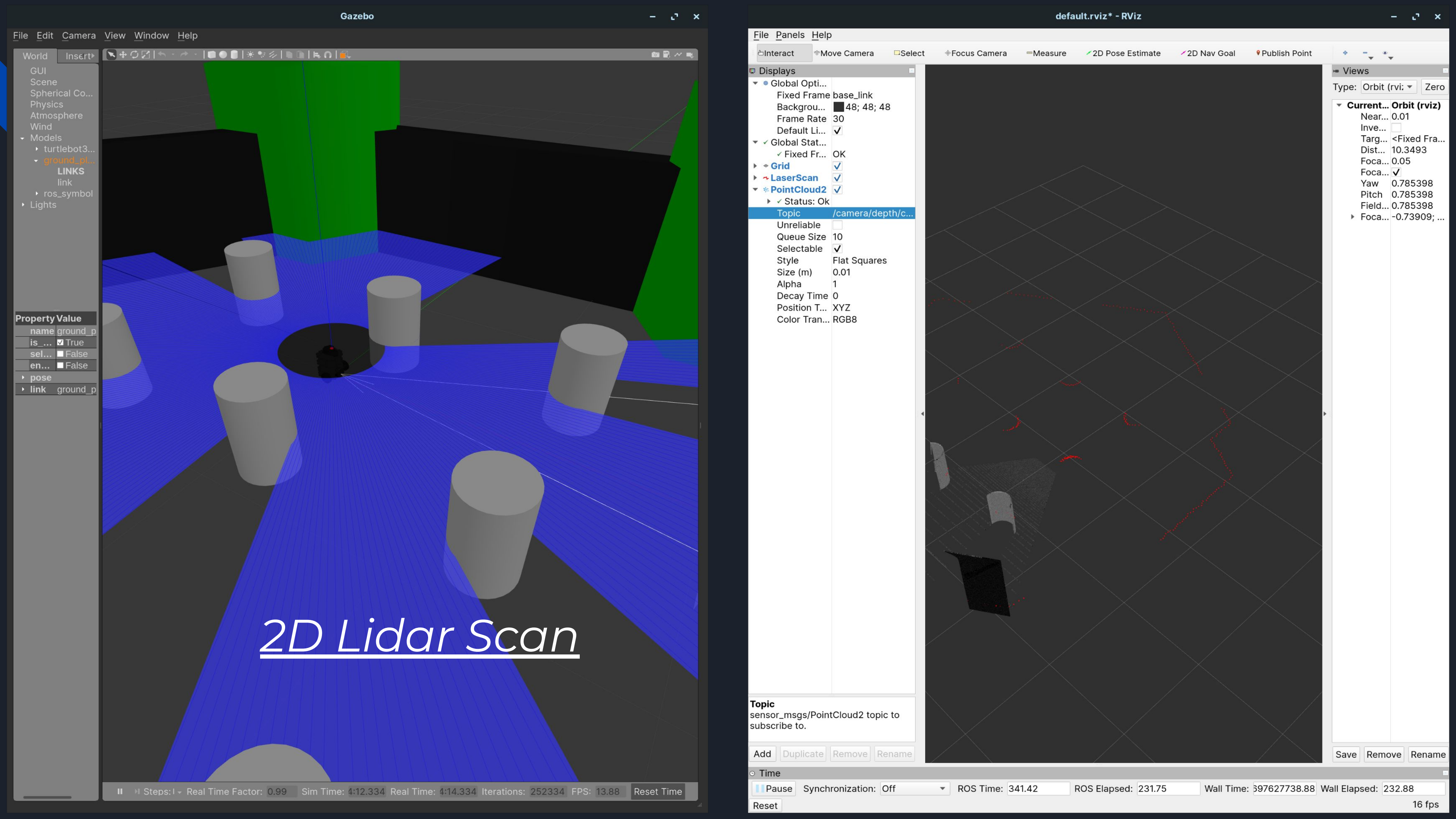
- We implemented simulations [~12 hours]
    - Simulation workspace in *(ros-simulations/catkin_ws)*

    - Documentation is present in *(ros-simulations/turtle_bot_sim_using_gazebo.md)*
        - Base robot - turtlebot3
        - Added intel realsense camera - gives raw image and pointcloud
        - Added 2D Lidar - gives laserscan

    - Future steps:
        - Added fiduciary image on plane
        - Add 3D Lidar

- Demonstrations:


RealSense rgbd Camera
Gazebo
File Edit Camera View Window Help
World Insert
GUI
Scene
Spherical Co...
Physics
Atmosphere
Wind
Models
ground_pl...
turtlebot3...
ros_symbol
Lights
Property Value
name ground_p
pose
link ground_p
Real Time Factor: 0.99 Sim Time: 4:47.615 Real Time: 4:50.019 Iterations: 287615 FPS: 13.81 Reset Time
default.rviz* - RViz
File Panels Help
Interact Move Camera Select Focus Camera Measure 2D Pose Estimate 2D Nav Goal Publish Point
Displays
Global Options
Fixed Frame base_link
Background Color 48; 48; 48
/opt/ros/noetic/share/turtlebot3_teleop/launch/turtlebot3_teleop
w/x : increase/decrease linear velocity (Burger : ~ 0.22, Waffle and Waffle Pi : ~ 0.26)
a/d : increase/decrease angular velocity (Burger : ~ 2.84, Waffle and Waffle Pi : ~ 1.82)
space key, s : force stop
CTRL-C to quit
currently: linear vel 0.0 angular vel 0.1
currently: linear vel 0.0 angular vel 0.0
currently: linear vel 0.0 angular vel 0.1
currently: linear vel 0.0 angular vel 0.0
Visibility
Image Topic
sensor_msgs::Image topic to subscribe to.
Add Duplicate Remove Rename
Camera
Views
Type: Orbit (rviz) Zero
Current... Orbit (rviz)
Near... 0.01
Targ... <Fixed Fra...
Dist... 6.5883
Foca... 0.05
Yaw 3.6304
Pitch 0.944796
Field... 0.785398
Foca... 0; 0; 0
Save Remove Rename
Time
Pause Synchronization: Off ROS Time: 287.43 ROS Elapsed: 259.25 Wall Time: 697611527.64 Wall Elapsed: 260.85
Reset 7 fps
100 % Oct 18 12:15


Gazebo
File Edit Camera View Window Help
World Insert
GUI
Scene
Spherical Co...
Physics
Atmosphere
Wind
Models
turtlebot3...
ground_pl...
LINKS
link
ros_symbol
Lights
Property Value
name ground_p
is_... True
sel... False
en... False
pose
link ground_p
2D Lidar Scan
Steps: Real Time Factor: 0.99 Sim Time: 4:12.334 Real Time: 4:14.334 Iterations: 252334 FPS: 13.88 Reset Time
default.rviz* - RViz
File Panels Help
Interact Move Camera Select Focus Camera Measure 2D Pose Estimate 2D Nav Goal Publish Point
Displays
Global Opti...
Fixed Frame base_link
Backgrou... 48; 48; 48
Frame Rate 30
Default Li...
Global Stat...
Fixed Fr... OK
Grid
LaserScan
PointCloud2
Status: Ok
Topic /camera/depth/c...
Unreliable
Queue Size 10
Selectable
Style Flat Squares
Size (m) 0.01
Alpha 1
Decay Time 0
Position T... XYZ
Color Tran... RGB8
Topic
sensor_msgs/PointCloud2 topic to subscribe to.
Add Duplicate Remove Rename
Views
Type: Orbit (rviz) Zero
Current... Orbit (rviz)
Near... 0.01
Inve...
Targ... <Fixed Fra...
Dist... 10.3493
Foca... 0.05
Foca...
Yaw 0.785398
Pitch 0.785398
Field... 0.785398
Foca... -0.73909; ...
Save Remove Rename
Time
Pause Synchronization: Off ROS Time: 341.42 ROS Elapsed: 231.75 Wall Time: 597627738.88 Wall Elapsed: 232.88
Reset
16 fps

# Project Timeline

- Completely Implement ROS Simulation (October 25th)
- Implement basic 2D and 3D lidar-camera target based methods (November 3rd)
- Implement better methods as proposed by papers - might be done simultaneously with basic implementation (November 13th)
- *If time permits*, run on hardware

# References


- An Improved Method for the Calibration of a 2-D LiDAR With Respect to a Camera by Using a Checkerboard Target, Fumio Itami, Takaharu Yamazaki, 2020

- A Flexible New Technique for Camera Calibration, Zhengyou Zhang, 2000

# Thanks for watching :)

Like, share, subscribe.