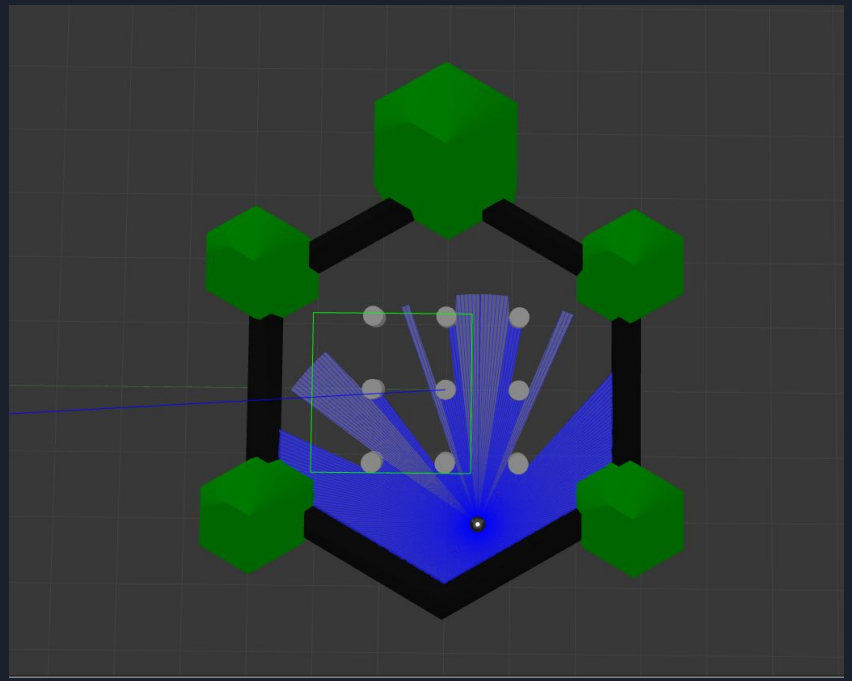
Navegação com turtlebot e ROS2: World Default

Anna Bozzi;
Pamella Mariano.

Mapa Escolhido

- **World Default Gazebo:** dros2 launch turtlebot3_gazebo turtlebot3_world.launch.py



Trabalho Final

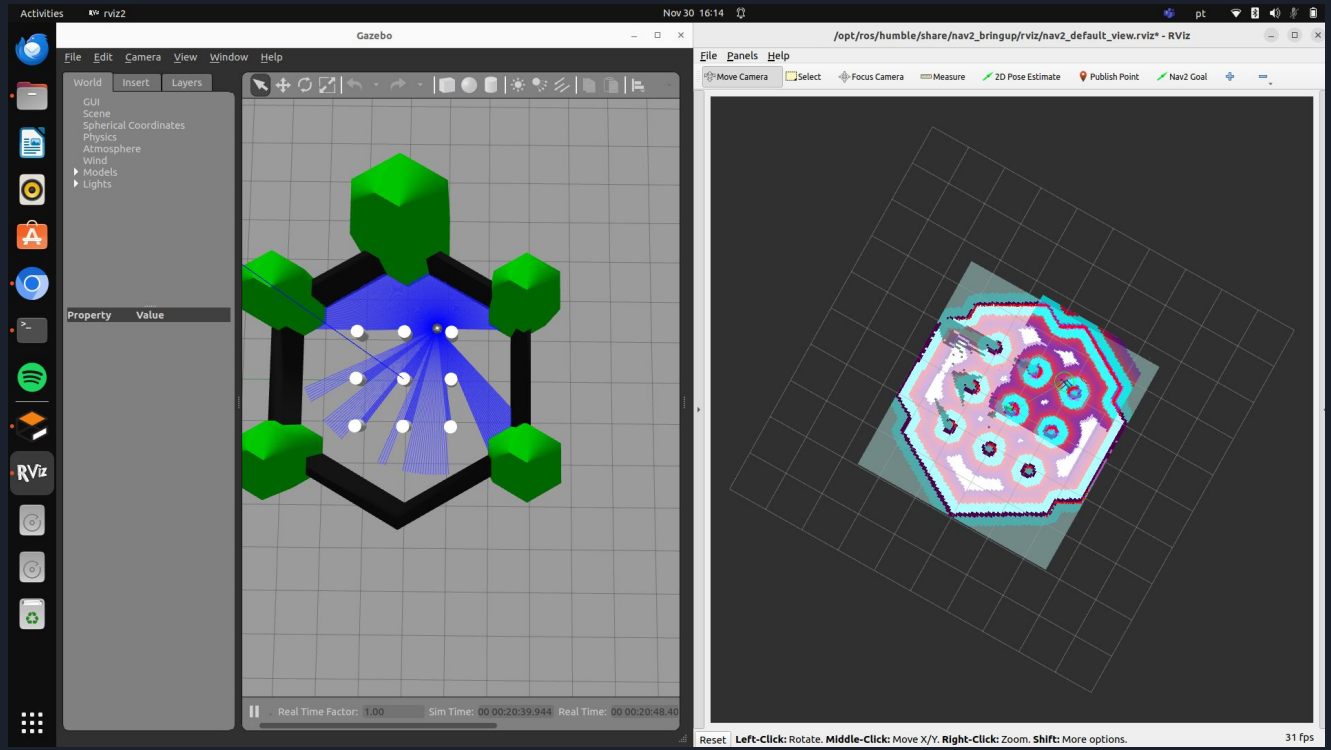
- **Desafio Escolhido:** Habilitar o Turtlebot3 para navegar dinamicamente de um ponto a outro no ambiente simulado do Gazebo;

- **Divisão em duas partes distintas:**
  - **Primeira etapa:** Implementação de código Python para controle básico do Turtlebot3 no ambiente padrão do Gazebo;
  - **Segunda etapa:** Tratamento de obstáculos no ambiente simulado, buscando navegação autônoma e evitando obstáculos no percurso do Turtlebot3;

Primeira Etapa

- **Funcionalidades:**
  - Inicialização e configuração do nó ROS2.
  - Recebimento de dados de posição e sensor de laser.
  - Lógica de movimento baseada nas leituras do sensor.
  - Publicação de comandos de velocidade para o Turtlebot3.
- **Detalhes Importantes:**
  - Utilização de lógica para evitar obstáculos.
  - Uso de mensagens específicas para comunicação entre nós ROS2.
  - Encerramento adequado do sistema ROS2 após a execução.

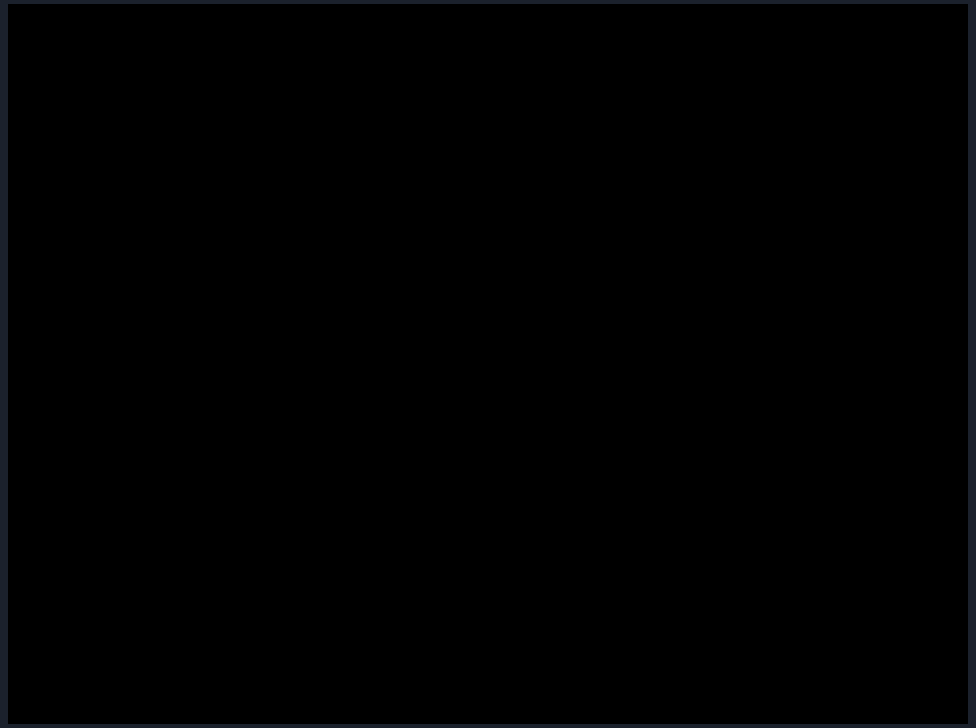
Primeira Etapa



Segunda Etapa

- **Ajuste Dinâmico da Velocidade Linear:** Sensor laser detecta obstáculos à frente do Turtlebot3.
- Código ajusta dinamicamente a velocidade linear de forma aleatória para permitir uma navegação mais flexível ao redor dos obstáculos.
- **Manutenção da Velocidade Angular:** Velocidade angular mantida constante para garantir uma rotação eficaz quando necessário, independentemente da detecção de obstáculos.
- **Condições de Navegação:** Na ausência de obstáculos detectados, as velocidades originalmente definidas são mantidas para movimento normal do Turtlebot3.
- **Inicialização do ROS2:** rclpy.init() movido para fora da função principal para ser executado antes do controle do Turtlebot3, garantindo a inicialização adequada do sistema ROS2.

Segunda Etapa



Resultados

- **Desafio de Tempo de Espera:**
  - Tempo de espera no loop principal causou lentidão nas respostas do sistema ROS2 aos eventos de detecção de obstáculos.
- **Ajuste de Timeout:**
  - Valor de timeout_sec em rclpy.spin_once(node, timeout_sec=0.01) foi reduzido para 0.001 para acelerar a resposta do sistema ROS2.
- **Persistência da Lentidão:**
  - Apesar do ajuste para um valor menor, a lentidão persiste, afetando a capacidade de resposta do sistema ROS2.

Conclusão

- **Adaptação Inteligente:**
  - Lógica adaptativa ajusta dinamicamente a velocidade linear, melhorando a navegação do Turtlebot3.
- **Desafio da Lentidão:**
  - A lentidão resultante do ajuste dinâmico da velocidade prejudica a agilidade, especialmente em ambientes que demandam respostas rápidas.
- **Conclusão:**
  - Melhoria significativa no controle do Turtlebot3 para uma navegação mais inteligente e adaptável.
  - Necessidade de explorar estratégias para otimizar o desempenho e resolver a lentidão, mantendo um equilíbrio entre eficiência e agilidade.