CET3004 – Robotics

Assignment-2

Autonomous Follow-and-Stop Robot in a Dynamic Environment

LEVEL: Individual work

WEIGHT: 75% of the total course mark

DEADLINE: 09.01.2026 at 11:59

SUBMISSION: Code + Report + Demonstration (Canvas in a ZIP folder)

LANGUAGE: Python

SYSTEM REQUIREMENTS:

- ROS 2 Jazzy
- Gazebo Harmonic
- A 2D LiDAR as the primary perception sensor

SCENARIO OVERVIEW

Your team has been commissioned by an industrial automation company developing mobile robots for shared workspaces, such as logistics hubs, hospitals, and manufacturing floors. The company requires an autonomous differential-drive robot that can:

1. Track a designated moving target (a human worker or another robot),
2. Follow it safely using onboard sensors only, and
3. Stop and maintain a safety buffer whenever the target becomes too close.

The main focus of the project is on ensuring safety, ethical responsibility, and reliable operation in dynamic and uncertain environments. This will be accomplished without relying on global planning, mapping, or advanced navigation systems, ensuring that the project aligns with the company's goals and requirements. Your task is to develop, test, and critically evaluate a prototype of this system.

The company is particularly interested in understanding how factors such as autonomous behaviour, safety constraints, and ethics concerning human-robot interaction (HRI) will influence your design and implementation decisions. They also require comprehensive documentation that demonstrates how you assessed the system's performance and ensured its compliance with ethical and legal standards.

LEARNING OUTCOMES:

LO2. Develop and assess autonomous robotic behaviours using various sensor data and planning techniques to enable robots to navigate and operate independently.

LO3. Critically evaluate the ethical, legal, and societal challenges associated with robotics and autonomous systems, including safety, accountability, and human–robot interaction.

LO4. Synthesise a solution to a problem domain for autonomous robots in realistic scenarios.

IMPORTANT INFORMATION:

All work must be completed individually, unless stated otherwise. You will only receive marks for your own work. You are responsible for the security and integrity of your files, and you must not allow others access to your assignments. Plagiarism or paraphrasing without proper accreditation will be handled seriously, as outlined in the University Infringement of Assessment Regulations and detailed in the Programme Handbook.

You are allowed to use AI tools in a supportive role during assessments. However, you must declare in your submission which tools you used and how you utilised them. Examples of acceptable use of AI tools include:

- Drafting and organising content
- Providing limited support in the writing process
- Acting as a support tutor
- Assisting with specific processes, such as translating content
- Offering feedback on content or proofreading

However, you cannot use AI tools to complete the project on your behalf; all work must be done by yourselves. This is to ensure that the work accurately reflects your understanding and effort, which are significant parts of your learning process. You should validate any AI-generated content before using it.

You are expected to upload your work directly. Submitting links to files saved in the cloud will not be accepted and will result in a zero mark. Ensure that the actual files are uploaded to Canvas and are readily accessible to the assessor. After submitting your files, it is essential to verify that you can retrieve and open them. It is your responsibility to ensure that the files are not corrupted at the time of submission, as this could result in the loss of your work. If you encounter any issues, report them immediately to the help desk, copying your lecturer, and seek alternative arrangements if necessary.

Project Requirements (Scenario-Driven)

Based on the industrial scenario, your project must include:

A. Robot Platform

Using the Week-7 lab model as a foundation, you must:

- Use the differential-drive robot (URDF/Xacro from the lab)
- Add a 2D LiDAR sensor in URDF
- Use ros2_control with DiffDriveController
- Use cmd_vel-based behaviour

B. Follow-and-Stop Behaviour

Your robot must:

1. Detect a moving target in front using the LiDAR (e.g., actor, simple robot, or box on a trajectory).
2. Follow the target using a reactive, sensor-driven algorithm.
3. Stop when the target enters a predefined safety zone (e.g., < 0.8 m)
4. Resume following when the target moves away.
5. Operate fully reactively (no planners, maps, or cost maps).

Recommended components:

- Distance estimation from LiDAR ranges
- Sector-based nearest-object detection
- Behaviour switching (Follow → Stop → Follow)
- Proportional controller for speed
- Safety override behaviour

C. Dynamic Environment

Your Gazebo world must include:

- A moving target (Gazebo actor or scripted model).
- At least one additional dynamic obstacle (to force perception decisions).
- Static elements (e.g., walls, warehouse layout, corridors).

D. ROS 2 Software Architecture

Your system must include at least two custom ROS 2 nodes:

A. Perception Node

- Subscribes to /scan
- Extracts target distance
- Processes noisy sensor data
- Publishes perception states, e.g.:
  - /target_distance
  - /target_detected (boolean)
  - /safe_to_move

B. Control Node

- Implements the follow-and-stop logic
- Subscribes to perception topics
- Publishes /cmd_vel
- Contains reactive state machine:
  - SEARCH (if target not detected)
  - FOLLOW (if target detected & safe)
  - STOP (if too close)

Deliverables

➢ ROS 2 Workspace Package

Must include:

- URDF/Xacro of robot + LiDAR
- World file with dynamic obstacles and target
- Launch files
- ros2_control YAML
- All custom nodes

➢ Demonstration Video (3–5 mins)

Show:

- Robot launched
- Target movement
- Robot following behaviour
- Robot stopping behaviour

NOTE: A screen recording is sufficient.

➢ Technical & Analytical Report (3000 words)

The report template has been provided.

Marking Criteria:

Criteria	Excellent (90–100%)	Very Good (70–89%)	Good (60–69%)	Satisfactory (50–59%)	Limited (40–49%)	Inadequate (<40%)
Scenario Understanding & System-Task Alignment (10%)	Demonstrates exceptional insight into the industrial follow-and-stop scenario. The system design demonstrates complete conceptual alignment with safety-critical, human-aware robotics. All choices are fully justified and tightly integrated with the scenario constraints.	Strong understanding of the scenario with well-reasoned design decisions. The system mostly aligns with requirements and constraints, with only minor gaps in justification or integration.	Adequate scenario understanding: system design meets requirements, but with limited rationale. Some scenario elements are applied mechanically rather than analytically.	Basic understanding demonstrated, but the scenario link is weak or inconsistent. Several design decisions lack justification.	Shallow or flawed understanding of the scenario; weak or incorrect alignment between system design and task requirements.	No meaningful understanding of the scenario. The system does not address the required Task follow-and-stop behaviour or violates core constraints.
Technical Implementation (30%)	Fully functional ROS 2 Jazzy + Gazebo Harmonic system. Diff-drive robot, LiDAR integration, nodes, launch files, ros2_control, and dynamic world work flawlessly. Implementation shows advanced understanding and exceptional robustness.	The system operates reliably, with only minor issues. All required components were implemented correctly, demonstrating good robustness and clear debugging evidence.	The system builds and runs with some imperfections. Most core components are functional, but they may exhibit partial failures or limited robustness.	Basic functionality is present, but it is accompanied by significant bugs, instability, or the absence of non-critical elements.	Significant features are missing or not working. Behaviour inconsistent or unstable; world or LiDAR not correctly integrated.	System does not function; cannot demonstrate autonomous behaviour; major components missing; violates assignment constraints.
Autonomous Behaviour Quality (20%)	Follow-and-stop behaviour is smooth, stable, reactive, and safe. State transitions (FOLLOW → STOP → FOLLOW) are reliable under dynamic scenarios. Robot consistently maintains a safe distance using LiDAR, handles rapid changes, and avoids unsafe movements.	Behaviour works reliably with minor inconsistencies. Shows good responsiveness and generally maintains safety.	Behaviour is functional but with noticeable delays, oscillations, or occasional unsafe approaches.	Minimal reactive behaviour: robot follows, but performance is inconsistent, slow, or poorly tuned. Safety distance is only partly maintained.	Behaviour frequently fails; the robot stops or follows incorrectly; unsafe distances are common.	No meaningful autonomous behaviour: the robot cannot detect or follow the target; unsafe or uncontrolled motion.

Ethical, Legal, and Societal Evaluation (20%)	Deep, critical, scenario-specific ethical/ legal/ social analysis. Expert use of standards (ISO 13482, 10218), safety principles, accountability, surveillance/privacy concerns. Insightfully linked to system design, with clear arguments and evidence.	Strong analysis with good use of relevant ethical and legal frameworks. Clear connection to system behaviour and scenario.	Solid discussion, but lacks depth or strong linkage to design decisions. Some concepts are underdeveloped.	Basic awareness of ethical issues, but mostly descriptive. Limited application to the scenario.	Weak or superficial; fails to address multiple dimensions (ethical/legal/social).	No meaningful evaluation; missing, generic, or incorrect references to ethics or legal frameworks.
Solution Synthesis & Integration (10%)	The system functions as a coherent whole. Perception, control, robot model, and world integrate seamlessly. Clear evidence of design thinking and strong justification for architectural choices.	Good integration and logical design. Minor inconsistencies, but overall system is cohesive.	System elements are mostly integrated, but with some disjointed or unclear connections.	Weak integration: Several elements appear bolted on rather than designed together.	Poor integration: components do not meaningfully interact or require manual intervention.	No coherent system: components do not work together; LO4 unmet.
Report Quality & Communication (10%)	Exceptionally clear, well-structured, technically rigorous report. Excellent diagrams, academic writing, and scenario framing. All requirements met.	Well-written, well-structured report with minor issues. Good diagrams and explanations.	Adequate clarity and structure; diagrams are functional; some inconsistencies in writing or detail.	Writing is understandable, but it lacks clarity, depth, and good structure. Missing elements or weak diagrams.	Poor clarity, weak or missing diagrams, unclear arguments, and structural issues.	Incomplete, unclear, or poorly presented report; missing major sections or unintelligible writing.

PACKAGE STRUCTURE:

```
my_diffbot/
├─ package.xml
├─ setup.py
├─ setup.cfg
├─ CMakeLists.txt
├─ resource/my_diffbot
├─ my_diffbot/
│  ├─ __init__.py
│  ├─ nodes/
│  │  ├─ perception_node.py #Processes /scan, publishes /target_distance and
/target_detected
│  │  ├─ follow_stop_node.py #Reactive state machine: SEARCH -> FOLLOW -> STOP
│  │
│  └─ launch/
│     ├─ bringup.launch.py  #Open the world and spawn the robot
│     └─ spawn_and_follow.launch.py #Launches gazebo + robot + perception +
control nodes
├─ urdf/
│  ├─ diffbot.urdf.xacro
│  └─ diffbot.urdf   #Generated by xacro
├─ config/
│  └─ diff_controllers.yaml
├─ worlds/
   └─ follow_world.sdf #Simple world with flat ground and a moving “target”
model can be added
```