Initial Task
UNIVERSITY CATEGORY

IESL ROBOCON 25/26

COMPUTER SCIENCE & ENGINEERING
UNIVERSITY OF MORATUWA

IESL

SLTMOBITEL
The Connection

Table of Contents

01. About RoboGames ---------- 02

02. Overview of the University Category ----- 03

03. Event Timeline ---------- 04

04. General Rules ---------- 05

05. Task Introduction ---------- 06

06. The Task ---------- 07

07. The Arena ---------- 10

08. The Robot & Setup ---------- 12

09. Installation & Use ---------- 14

10. Violations ---------- 16

11. Judging criteria ---------- 17

12. Submission ---------- 18

13. Contact Details ---------- 20

About Robogames

IESL RoboGames is an annual robotics competition organized by the Department of Computer Science and Engineering at the University of Moratuwa, together with the ITCE Sectional Committee of the Institution of Engineers, Sri Lanka (IESL) and in partnership with SLT Mobitel. The event aims to support and develop young talent by encouraging interest and skills in engineering and technology.

The competition has three categories: School, Undergraduate, and Open. For the School and Undergraduate categories, workshops and awareness programs are conducted to introduce students to robotics in a simple and practical way. These programs help improve learning, boost creativity, and motivate students to become future engineers and innovators

MISSION

The mission is to bridge the gap in robotics education by providing all students around the country with hands-on training and guidance, empowering them to apply their knowledge in real-world challenges, and grow as future engineers and technologists.

Overview of the **University Category**

This Category of the competition encourages teams to **explore innovative and creative solutions in autonomous drone technology**. This category provides participants with the freedom to design and implement their own approaches using custom hardware, software, and algorithms.

The focus is on demonstrating **full autonomy, intelligent decision-making, and practical application of drone systems in real-world inspired scenarios such as search and rescue missions**. Teams are evaluated not only on task completion but also on originality, system integration, robustness, and overall effectiveness of their solution.

This category highlights cutting-edge developments in autonomous navigation, computer vision, and aerial robotics.

University Category **Timeline**

DEC 2025	15-12-2025	Launch of Robogames
	22-12-2025	Registrations Open
JAN 2026	05-01-2026	Awareness Session
	19-01-2026	Initial Task Revealed
	23-01-2026	Registrations Close
	24-01-2026	Submission Open
	TBD	Initial Task - Workshop 1
	TBD	Submission Deadline
FEB 2026	22-02-2026	Semifinalists Revealed & Task Released
	TBD	Semifinal Task - Workshop 2
	09-03-2026	Semifinal Task Deadline
MAR 2026	17-03-2026	Finalists Revealed & Final Task Released
	25-04-2026	Grand Finale

General Rules

- A team can consist of a maximum of 5 members and a minimum of 1 member. All members must be from the same school/institute.
- Teams that successfully complete this simulation round within the constraints will be considered for the next phase.
- Plagiarism is a serious offense and will cause a team to be disqualified. The judge panel may carry out a viva if solutions provided by a team are suspected to be plagiarized.

Note: The decision of the judges will be final.

Task Introduction

Drone Challenge
Aerial Autonomy

We are thrilled to announce the Phase 1 Task of the IESL RoboGames 25/26 competition. Our primary objective is to push the boundaries of autonomous systems and provide a platform for undergraduates to demonstrate their skills in aerial robotics and machine vision.

In this phase, participants are tasked with developing an autonomous control algorithm for a quadcopter using the ArduPilot SITL combined with the Webots simulation environment.

The goal is to program the drone to autonomously take off, navigate a path, detect markers, and land.

The TASK

Sequence of Events

1. **Takeoff**
    - The drone starts on the initial landing pad.
    - It must arm and takeoff to a stable hover altitude less than 3m.
2. **First Path & Detection:**
    - The drone must detect the Yellow Line and follow it autonomously.
    - It must navigate to the position directly above the Second Landing Pad (marked with an AprilTag).
    - Action: Hover above the pad, detect the AprilTag ID using the camera, and print the value to the console.

The TASK CTD.

3. **Turn & Second Path:**
    - After detecting the first tag, the drone must perform a **Right Turn** (90 degrees).
    - It must acquire the new Yellow Line and follow it to the **Third Landing Pad**.
4. **Final Detection & Landing:**
    - Upon reaching the third pad, the drone must detect the AprilTag ID and print the value to the console.
    - Goal: The drone must autonomously land on the center of the third AprilTag landing pad.
    - The motors must disarm to complete the run.

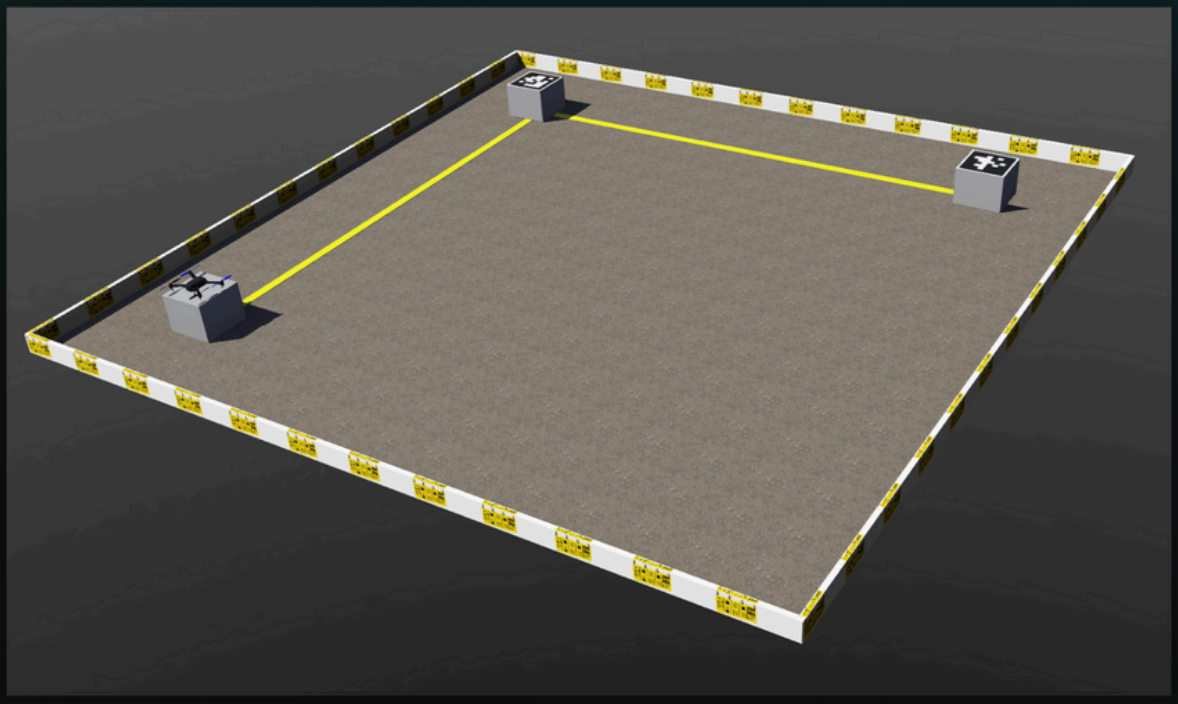
The TASK CTD.

Technical Approach

- The primary challenge of this task is **Computer Vision**.
- You will likely need to process the video to calculate the drone's position relative to the yellow line (e.g., finding the center of the line in the image frame) and send flight corrections (roll/pitch/yaw) to ArduPilot based on that data.
- While not required for submission, we highly recommend displaying your processed video feed (e.g., using cv2.imshow) during development to debug your vision algorithms.

The Arena

Competitors should not modify the provided Webots environment file while programming the task unless explicitly instructed for configuration purposes.



The Arena CTD.

Arena Specifications:

- **Dimensions**: 8m × 8m Square area.
- **Surface**: Flat terrain with specific landing markers.

Visual Elements:

- Landing Pads: Elevated blocks marked with AprilTags.
- Navigation Lines: Bright Yellow lines connecting the landing pads.
- Visual Markers: AprilTags are used on the landing pads to denote target destinations.

The Robot & Setup

Robot Platform:

- Model: IRIS Quadcopter
- Flight Controller: ArduPilot SITL
- Sensors: Bottom facing camera, IMU, GPS (simulated)

Environment Setup:

- We have provided a Docker container to simplify the setup of Webots, ArduPilot, and the necessary dependencies.

Operating System & Compatibility:

- **Linux (Recommended):** The provided Docker container is designed and tested on Linux systems (e.g., Ubuntu 20.04 or 22.04).
- **Windows/macOS:** It is also possible to run the Docker container on Windows (e.g., via WSL2) or macOS. While we have not officially tested the environment on these platforms, you are welcome to configure it on Windows if you find it easier or more convenient.

The Robot & Setup CTD .

Disclaimer :

We cannot guarantee the container will work flawlessly on every device configuration. It is provided as a convenience, not a requirement.

Manual Installation

If you choose to install the environment manually, you will need to set up the following components:

- **Webots R2025a** : The robot simulation platform.
- **ArduPilot SITL** : The autopilot firmware source code
- **MAVProxy** : The Ground Control Station software for communication.
- **Python 3** : Required for running your control scripts, along with necessary libraries (e.g., pymavlink, opencv-python, numpy).

Installation & Use CTD.

All installation files, the container setup, and the base project structure are available in the official RoboGames University Repository:

https://github.com/IESL-RoboGames/IESL-RoboGames-Uni-Phase1

Instructions

Please clone this repository and follow the instructions in the README.md file to set up your environment. You can also refer the README for helpful resources and guides.

Repository Updates

Please note that the repository may be updated during the competition to fix bugs or improve the environment. We recommend keeping your local clone connected to git so you can pull the latest commits if strictly necessary.

Technical Issues: If you encounter bugs or technical problems with the simulation environment or setup, open a new Issue in the GitHub Repository using the provided Issue Template. This allows our technical team to track and resolve errors efficiently.

Installation & Use CTD .

Programming Location

- **CRITICAL** : All your control logic and scripts must be written inside the Task folder located in the repository.
- The system is designed so that the script in the Task folder acts as the brain of the drone, communicating with ArduPilot to control the flight while processing camera data.

Constraints

- Writing a Python script that runs inside Webots as a direct Robot Controller (using the internal Webots API to bypass the flight controller) is strictly prohibited. The drone must be flown by commanding the ArduPilot firmware via MAVLink.

Tuning

- Participants are allowed to tune the PID values of the drone to achieve stable flight.
- The default PID values are provided in the Docker/Repo. If these are modified, the new parameter file must be submitted.

Violations

- **Environment Tampering**
    - Modifying the position of pads, lines, or the drone's start location in the world file.
- **Video Manipulation**
    - Editing the submission video to hide crashes or cuts. The submitted code must be capable of reproducing the video performance.
- **Illegal Control Methods**
    - Using keyboard/joystick control during the recording, or using internal Webots physics hacks instead of valid ArduPilot flight commands.

Any violation will cause the submission to be rejected

Judging Criteria

The winner will be decided based on a total score derived from task completion accuracy and efficiency.

Scoring Breakdown

Points will be awarded based on the degree of success for each action. The values below represent the **maximum** points achievable for perfect execution in each category:

- Takeoff Success: **20 Points**
- Line Following Accuracy: **30 Points**
- AprilTag Detection Correctness: **20 Points**
- Landing Accuracy: **30 Points**

Maximum Time Limit

The drone must complete the entire mission (from Takeoff to final Disarm) **within 4 Minutes on Webots simulation time**.

Submission

Submission Components

1. **Video Demonstration** : A screen recordings showing:
    - The simulation window (Webots).
    - The terminal/console window showing the script running and the AprilTag values being printed.
2. **Project Source Code (Zip File)**: A zip file containing only your source code and essential project files.

INCLUDE:

- The Task folder (this is the most important part).
- Any custom parameter files (.param) if you tuned the drone.
- A requirements.txt file (if you used external Python libraries outside the standard ones).

Submission CTD.

DO NOT INCLUDE:

- Virtual environments (venv, env, .conda).
- Docker/Podman files, images, or containers.
- The .git folder.
- __pycache__ folders.
- The Webots application or ArduPilot source code itself.

Technical **Team**

Nadeesha
Jayamanne
+94 71 903 1816
nadeeshaj.23@cse.mrt.ac.lk

Tharaka
Jayasena
+94 77 964 9008
tharakaj.23@cse.mrt.ac.lk

Thisen
Ekanayake
+94 71 065 0979
thisene.23@cse.mrt.ac.lk

Contact **Details**

Chairperson

Praveen Nawaratne

+94 77 385 3091

praveenn.23@cse.mrt.ac.lk

Vice Chairperson

Thilakshan Balakrishnan

+94 77 107 6556

thilakshanb.23@cse.mrt.ac.lk

Delegate Handling Committee Lead

Hasini Lawanya

+94 70 153 0016

lawanyakkhg.23@cse.mrt.ac.lk

Selection & Technology Committee Lead

Nadeesha Jayamanne

+94 71 903 1816

nadeeshaj.23@cse.mrt.ac.lk