**Multi-UAV Roundup Strategy with CEL-MADDPG**

This project implements the "Multi-UAV roundup strategy method based on deep reinforcement learning CEL-MADDPG algorithm" as described in the paper published in Expert Systems With Applications. The implementation includes:

1. **Curriculum Experience Learning (CEL)**: Divides the roundup task into three subtasks
2. **Preferential Experience Replay (PER)**: Selects more important samples for learning
3. **Relative Experience Learning (REL)**: Uses experiences similar to the current situation

**Project Structure**

* run\_simulation.py: Main script to run training, testing or demo
* environment.py: Multi-UAV roundup environment simulation
* agents.py: UAV and target agent implementations
* cel\_maddpg.py: Implementation of the CEL-MADDPG algorithm
* networks.py: Neural network models for actor and critic
* replay\_buffer.py: Experience replay buffer with PER and REL strategies
* visualization.py: Visual display and animation with military-radar style
* utils.py: Utility functions and configurations

**Installation**

**Requirements**

To run this project, you need:

* Python 3.8 or higher
* TensorFlow 2.9 or higher
* NumPy
* Matplotlib
* Pillow
* FFmpeg (optional, for high-quality video output)

**Installing Dependencies**

Place the requirements.txt file in your project folder and install the dependencies with:

bash

pip install -r requirements.txt

**Usage**

**Training**

To train the CEL-MADDPG algorithm:

bash

python run\_simulation.py --mode train --num\_uavs 3 --num\_obstacles 3 --episodes 20000

Configurable parameters:

* --num\_uavs: Number of UAVs (default: 3)
* --num\_obstacles: Number of obstacles (default: 3)
* --dynamic\_obstacles: Enable moving obstacles
* --episodes: Number of training episodes
* --save\_path: Path to save models
* --render\_interval: Interval for displaying animation during training

**Testing**

To test a trained model:

bash

python run\_simulation.py --mode test --load\_path ./saved\_models --num\_uavs 3 --num\_obstacles 3

**Demo**

To run a demo with a trained model and save the animation:

bash

python run\_simulation.py --mode demo --load\_path ./saved\_models --output roundup\_demo.mp4

**Configuration**

You can modify parameters in the utils.py file to adjust:

* Environment dimensions
* UAV and target characteristics
* Training hyperparameters
* Reward function weights

**Algorithm Details**

The CEL-MADDPG algorithm divides the roundup task into three subtasks:

1. **Target Tracking**: UAVs approach the target
2. **Encircling Transition**: UAVs form an encirclement around the target
3. **Shrinking Capture**: UAVs tighten the circle to capture the target

Each subtask has its own reward function to guide learning.

**Implementation Features**

* **Visualizer**: Graphical display with radar and military style for better illustration of the roundup process
* **Obstacle Avoidance**: UAVs have the ability to detect and avoid obstacles
* **Intelligent Target**: The target moves with an evasion strategy away from the nearest UAV
* **Custom Icons**: Display of UAVs and targets with custom icons (or creation of default icons if images are not available)

**Quick Start**

To get started quickly, follow these steps:

1. Place all files in a folder
2. Create subfolders images and saved\_models
3. First run in training mode: python run\_simulation.py --mode train
4. Then view the results in demo mode: python run\_simulation.py --mode demo --load\_path ./saved\_models

This will create an attractive animation with a military radar style where UAVs pursue and surround the target according to the CEL-MADDPG algorithm strategy.

**Support and Help**

If you encounter any issues:

* You may need to install FFmpeg for high-quality animation saving
* Make sure dependencies are installed correctly
* Use Python 3.8 or higher

**References**

This implementation is based on the paper "Multi-UAV roundup strategy method based on deep reinforcement learning CEL-MADDPG algorithm" published in Expert Systems With Applications.