howest
hogeschool

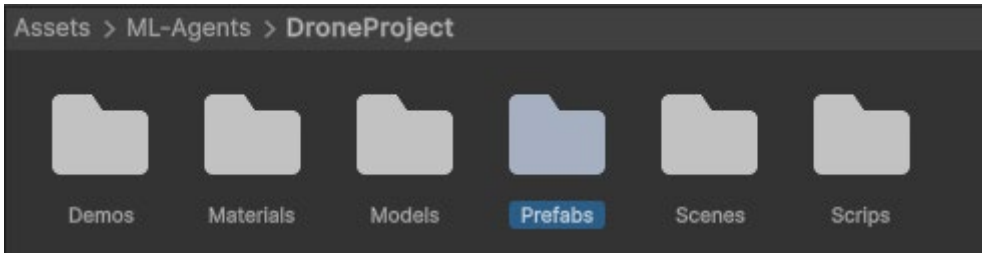
Gebruikershandleiding

Hoe kan een drone autonoom een persoon volgen met behulp van Reinforcement Learning?

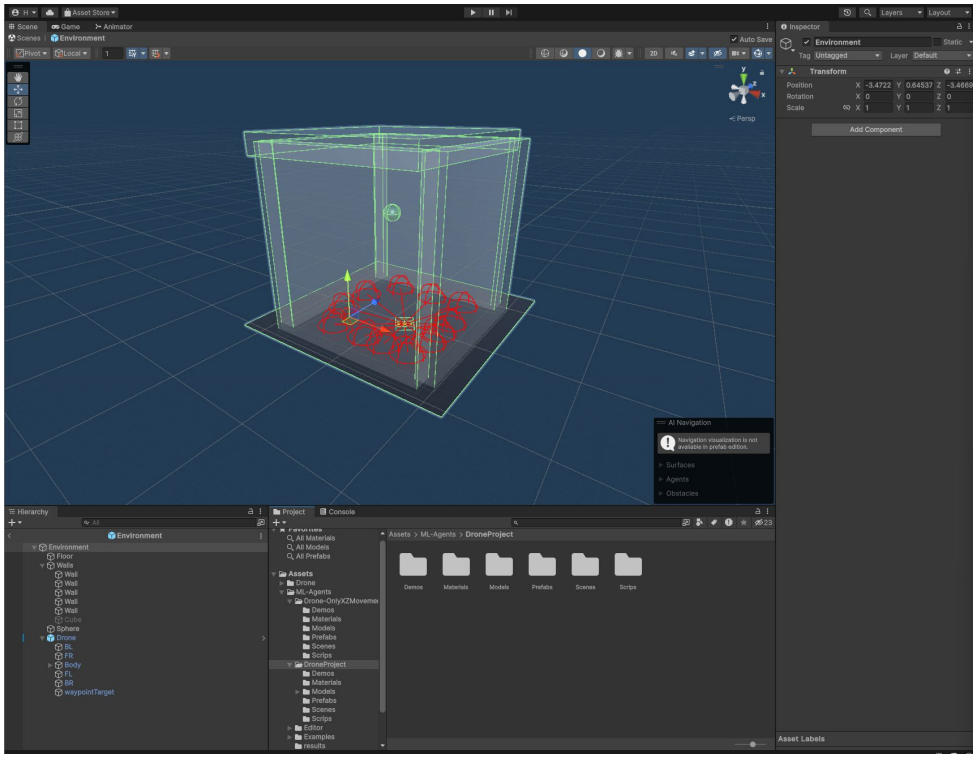
Olivier Cardoen

Gebruik van een bestaand Model

Open het Unity project en ga vervolgens naar de folder “Prefabs” in de map DroneProject

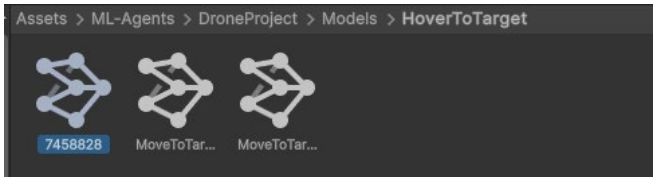


Open de environment.



Om een bestaand model in te laden, kun je de map “Models“ openen.

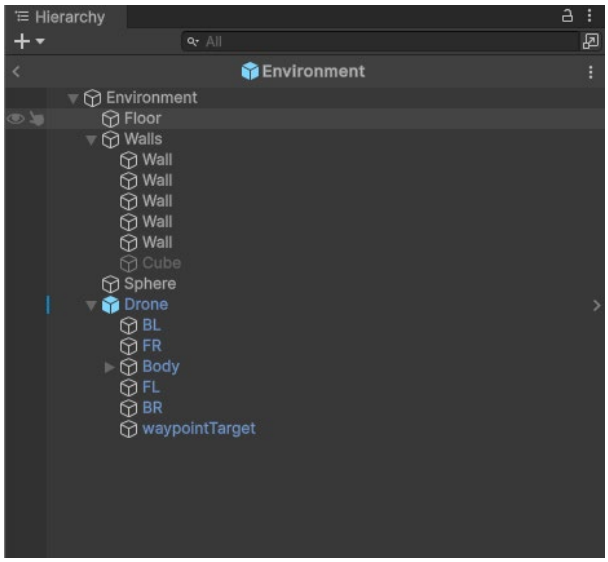
Kies een model uit “MoveToTarget”. Ik gebruik bv: 7458828



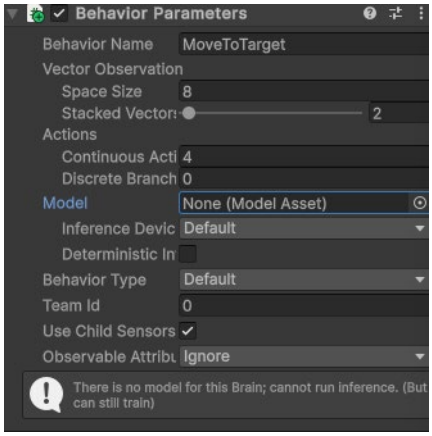
Het model laden in de Drone

Nu je een model gekozen hebt, kunnen we deze slepen in de Drone.

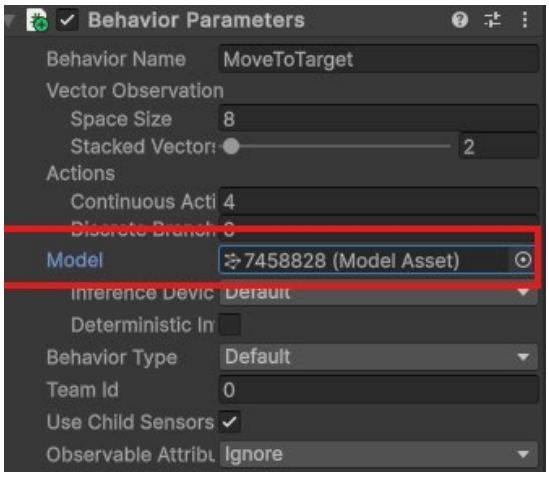
Dit doe je door de drone aan te klikken in het “Hierarchy” venser.



We focussen ons op het rechtse venster dat verschijnt. In het “Inspector” venster zijn er heel wat parameters. We scrollen naar beneden tot we het “Behavior Parameters’ script te zien krijgen.



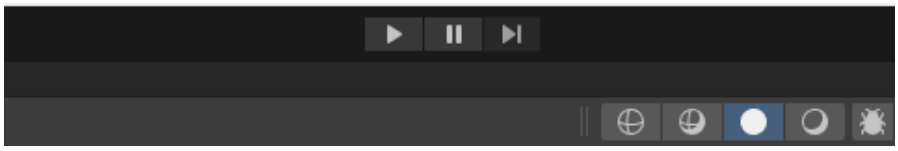
Om het gekozen model in te laden, slepen we het model in de Project folder naar het “Model” Tab in het “Behavior Parameters” venster.



Vervolgens kan je de Prefab opslaan en naar de “Scenes” map gaan in het Project venster.

Klik op “Drone_Hover”.

Boven op het scherm staat er een Play knop.



Dit zal de simulatie beginnen.

Een nieuw model trainen

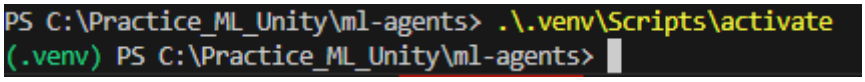
Een nieuw model trainen gaat als volgt:

Open de ml-agents folder in een terminal.

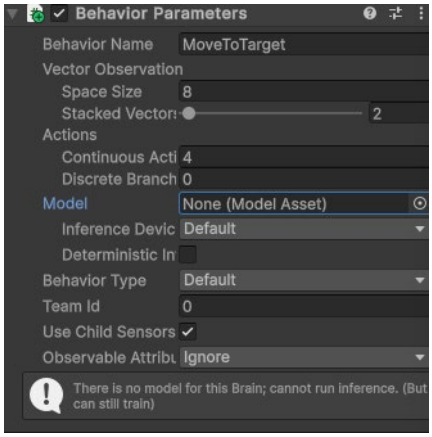
Vervolgens start je de virtual environment door dit commando in te vullen:

```
.\.venv\Scripts\activate
```

Nu zou je dit moeten te zien krijgen:



Vooraleer we iets anders doen, maak je eerst dat in het “behavior Parameters’ venster, de Model tab op “Default” staat.



Sla de prefab op en ga naar de Scene.

Voor het nieuw model te trainen hoeven we maar 1 commando te gebruiken.

```
mlagents-learn config/ppo/Drone.yaml --run-id=ExampleId
```

Dit commando zal nu wachten tot je de simulatie start in Unity zelf.

```
 Version information:
  ml-agents: 1.1.0,
  ml-agents-envs: 1.1.0,
  Communicator API: 1.5.0,
  PyTorch: 2.2.2+cu121
[INFO] Listening on port 5004. Start training by pressing the Play button in the Unity Editor.
```

Eenmaal de simulatie start, zou je deze output te zien moeten krijgen:

```
[INFO] Connected to Unity environment with package version 3.0.0 and communication version 1.5.0
[INFO] Connected new brain: MoveToTarget?team=0
[INFO] Hyperparameters for behavior name MoveToTarget:
        trainer_type:   ppo
        hyperparameters:
          batch_size:   2024
          buffer_size:  20240
          learning_rate:        3e-05
          beta: 0.005
          epsilon:      0.2
          lambd:        0.95
          num_epoch:    3
          shared_critic:        False
          learning_rate_schedule:       linear
          beta_schedule:        linear
          epsilon_schedule:     linear
        checkpoint_interval:    500000
        network_settings:
          normalize:    True
          hidden_units: 512
          num_layers:   3
          vis_encode_type:      simple
          memory:       None
          goal_conditioning_type:       hyper
          deterministic:        False
        reward_signals:
          extrinsic:
            gamma:      0.995
            strength:   0.99
            network_settings:
              normalize:        False
              hidden_units:     128
              num_layers:       2
              vis_encode_type:  simple
              memory:   None
              goal_conditioning_type:   hyper
              deterministic:    False
        init_path:      None
        keep_checkpoints:       5
        even_checkpoints:       False
        max_steps:      10000000
        time_horizon:   1000
        summary_freq:   30000
        threaded:       False
        self_play:      None
        behavioral_cloning:     None
```

Als je tevreden bent met het resultaat, kan je de simulatie stoppen in Unity.

Dit zal dan het getrainde model opslaan in de hooft ml-agents map onder “Results”.

Je kan het dus altijd hergebruiken door het te slepen naar de “Models” folder in het Project Venster.

howest
hogeschool

howest.be