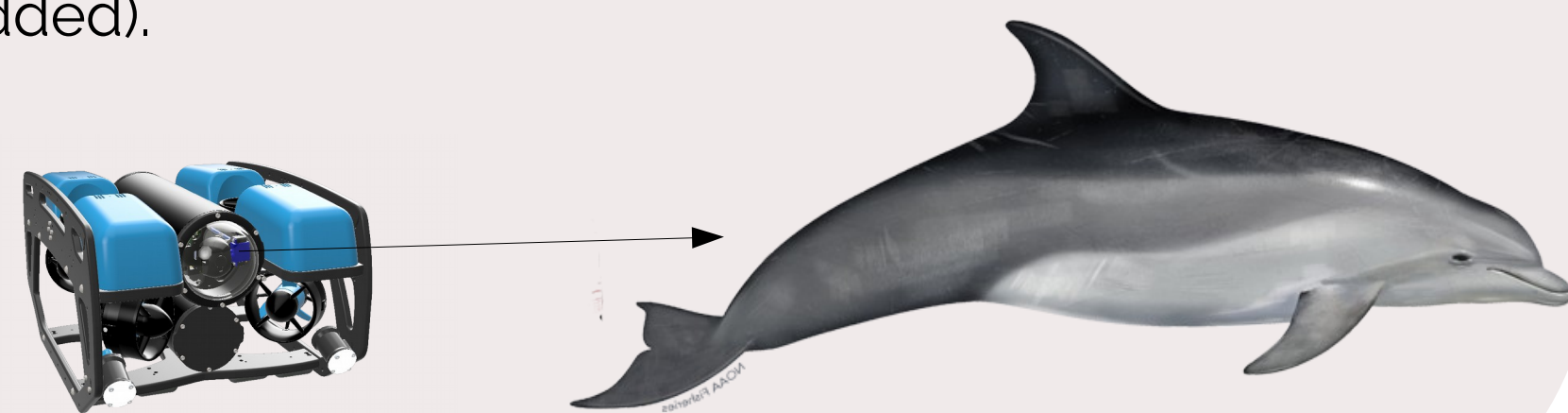


¹ School of Computing, ² School of Engineering,
³ School of Natural and Environmental Sciences

Can we train a **visuomotor** policy, purely with **machine learning**, to do **target tracking** “in-sim” ... and then use it to control a real vehicle like an ROV from camera input alone?

My research is just the **first building block** toward this goal. It is so far entirely **“in-sim”** (real-world transfer is to come), it is **“single object tracking”** (multiple objects would be the next challenge), and in **2 dimensions** (the vertical 3rd dimension will then be added).



Generational Autoencoder

The diagram illustrates a Generational Autoencoder architecture. It starts with an input image of a car on a road. This image is processed by an **Image Encoder** (green box), which consists of three orange vertical bars representing layers. The encoder's output is a latent space representation Z (a square of black dots). This latent space Z is then processed by two decoders: an **Image Decoder** (green box) which outputs a reconstructed image of the car, and a **State Decoder** (green box) which outputs parameters r , θ , and ψ . The Image Decoder also has a feedback loop from its output back to the latent space Z .

Overall Framework

The diagram illustrates the overall framework for the car tracking task, showing the interaction between the environment, the model, and the reinforcement learning components.

Environment Interaction:

- Unity exe** sends a_t to **Gym env** via `take action()`.
- Gym env** sends o_t to **Unity exe** via `observe()`.
- Gym env** sends o_t to **CM-VAE**.
- Gym env** sends z_t to **Replay buffer**.
- Gym env** sends $z_{t:n}$ to **Replay buffer**.
- Gym env** sends $o_{t:n}$ to **Replay buffer**.
- Gym env** sends **reward + done flag** to **Replay buffer**.

Model Components:

- CM-VAE** (pretrained model) takes o_t as input and outputs **feature vector z** .
- SAC** (Soft Actor-Critic) consists of:
 - Critic**: Takes **obs** and **action** as input and outputs **Q value** (x2).
 - Actor**: Takes **obs** as input and outputs **action** (x2).
 - Replay buffer**: Stores data for training.

Diagram Details:

- The **Replay buffer** and **CM-VAE** are marked as **not used at inference time**.
- The **Replay buffer** provides data to the **Actor** and **Critic**.
- The **Actor** outputs **action** to the **Gym env**.

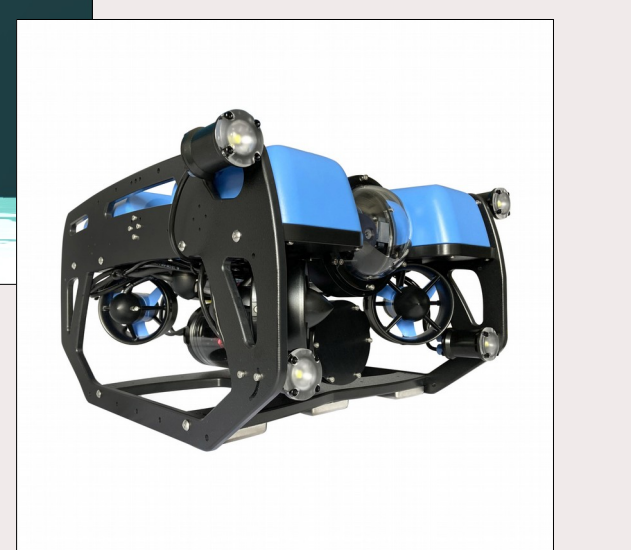
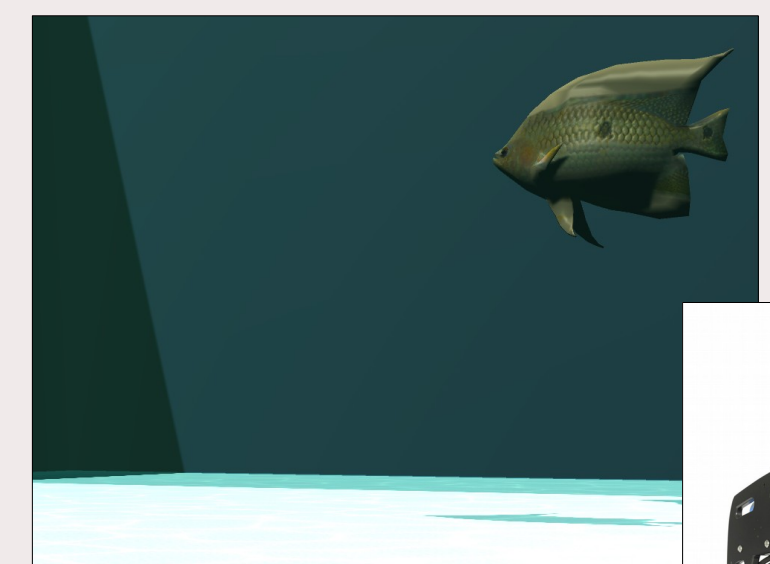
Diagram Illustration:

The diagram shows a **TARGET** car and a **TRACKER** car. The **TRACKER** car is positioned at a distance d from the **TARGET** car, with a heading angle a . The **TRACKER** car is also at a distance r from the origin.

Reward Formula:

$$r = A - \left(\frac{\sqrt{x^2 + (y - d)^2}}{c} + \lambda |a| \right)$$

Termination of episode:
collision / > max distance /
> max timesteps



Newcastle
University

Scan for
videos of my
agents and for
talks on the
same work

