From simulation to the field: Learning to swim with the AQUA robot

Juan Camilo Gamboa Higuera,
David Meger and Gregory Dudek

School of Computer Science
Centre for Intelligent Machines
McGill University

Work in our lab

Adaptive systems for autonomous scientific data collection

Meger et al, 3D Trajectory Synthesis and Control for a Legged Swimming Robot, IROS 2014

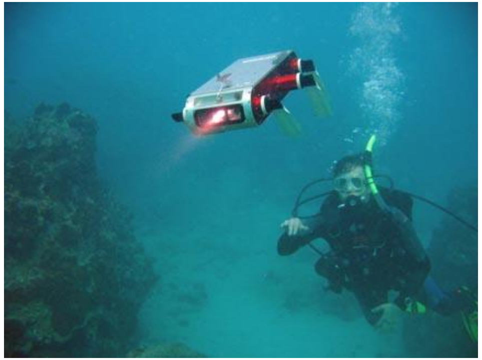
Shkurti et al, Underwater Multi-Robot Convoying using Visual Tracking by Detection, IROS 2017

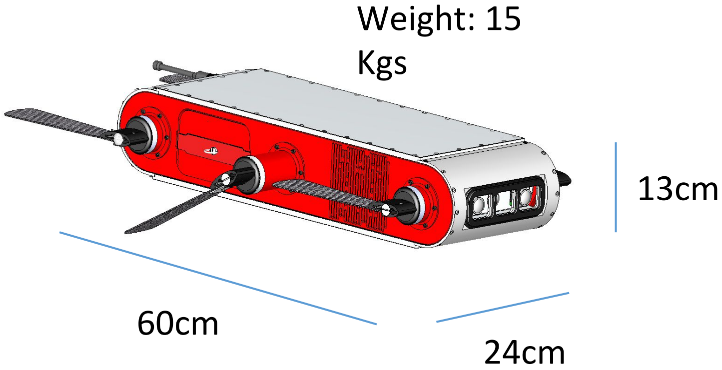
Outline of this talk

1. The Aqua Robot
   a. Hardware Overview
   b. Software Overview
2. The aqua_description and aqua_gazebo packages
   a. Hardware emulation
   b. Hydrodynamics simulation
3. Architecture for motor control learning
   a. Implementation of Model-Based RL algorithms (kusanagi)
   b. The RL glue code (aqua_rl/kusanagi_ros)

1. The **AQUA** robot

Portable Underwater Autonomous Vehicle





Based on RHex walking platform

- Developed at McGill University
- Commercialized by Independent Robotics

Who is using the AQUA robot?

Mobile Robotics Laboratory @ McGill
(Greg Dudek and Dave Meger)

VGR Lab @ York U
(Michael Jenkin)

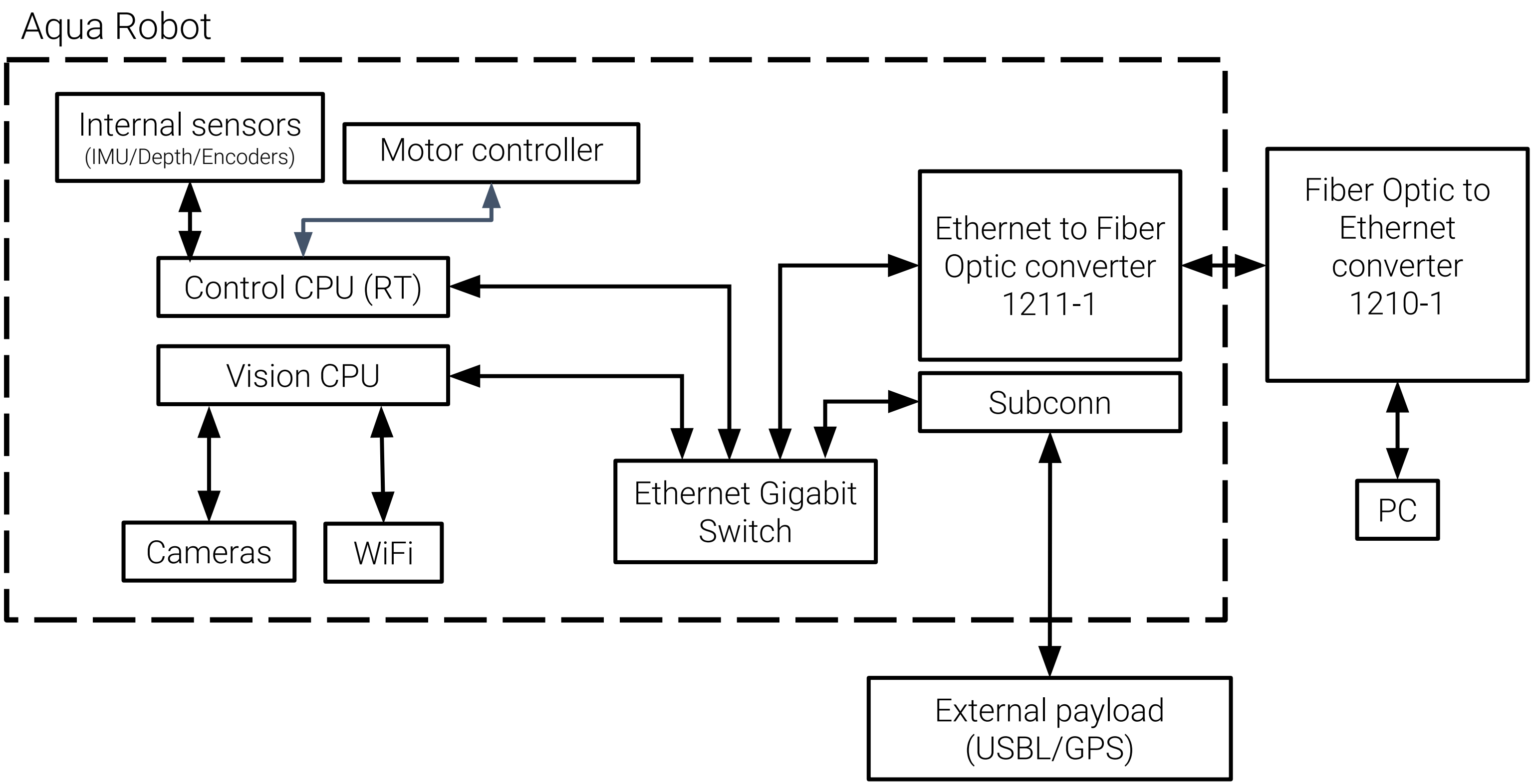
Autonomous Field Robotics Lab @ U South Carolina
(Yiannis Rekleitis)

Vision and Robotics Lab @ CINVESTAV
(Luz Abril Torres Mendez)

Interactive Robotics Laboratory @ U Minnesota
(Junaed Sattar)

WARPLab @ Woods Hole
(Yogesh Girdhar)

AQUA robot hardware



Software Overview (Control only)

Control Stack (RT)

RoboDevel

- Low-level Motor Ccontrol
- Power Management
- Internal Sensors

IMU driver

1 khz loop

Vision Stack

Hardware interface (ROS)

- aquahw
- aquacore

Navigation (ROS)

- aquaautopilot
- aqua_depth
- aqua_positioning

User interface (ROS)

- aquajoy

External PC

Controller Learning (ROS)

- Task Marshalling (aqua_marshall)
- Feedback Controllers and Dataset generations (kusanagi_ros)

General Purpose RL

- kusanagi
- rllab

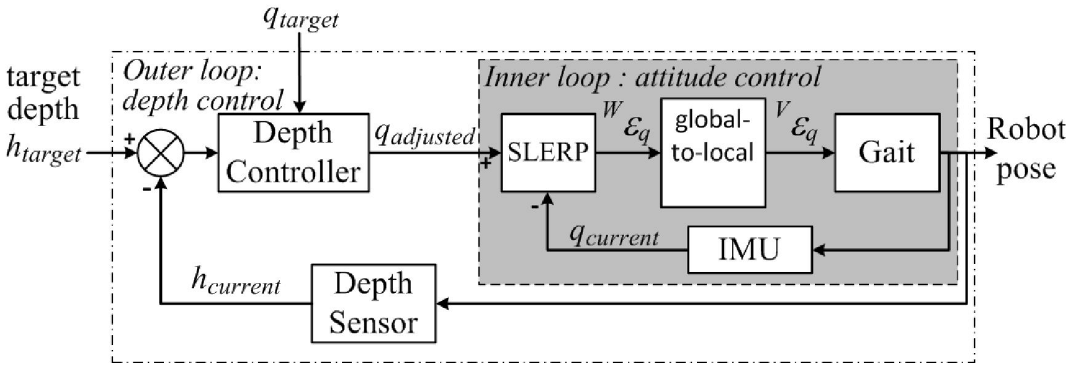
Hardware Interface

aquacore specifies common messages for aqua state and control

aquahw UDP interface exposing robot state and commands through ROS

Navigation

aquaautopilot implements trajectory tracking via waypoints



Meger et al, [3D Trajectory Synthesis and Control for a Legged Swimming Robot](), IROS 2014

aqua_positioning and **aqua_depth** transform sensor data to ROS convention

User interface

aquajoy is the joypad interface to **aquaautopilot**

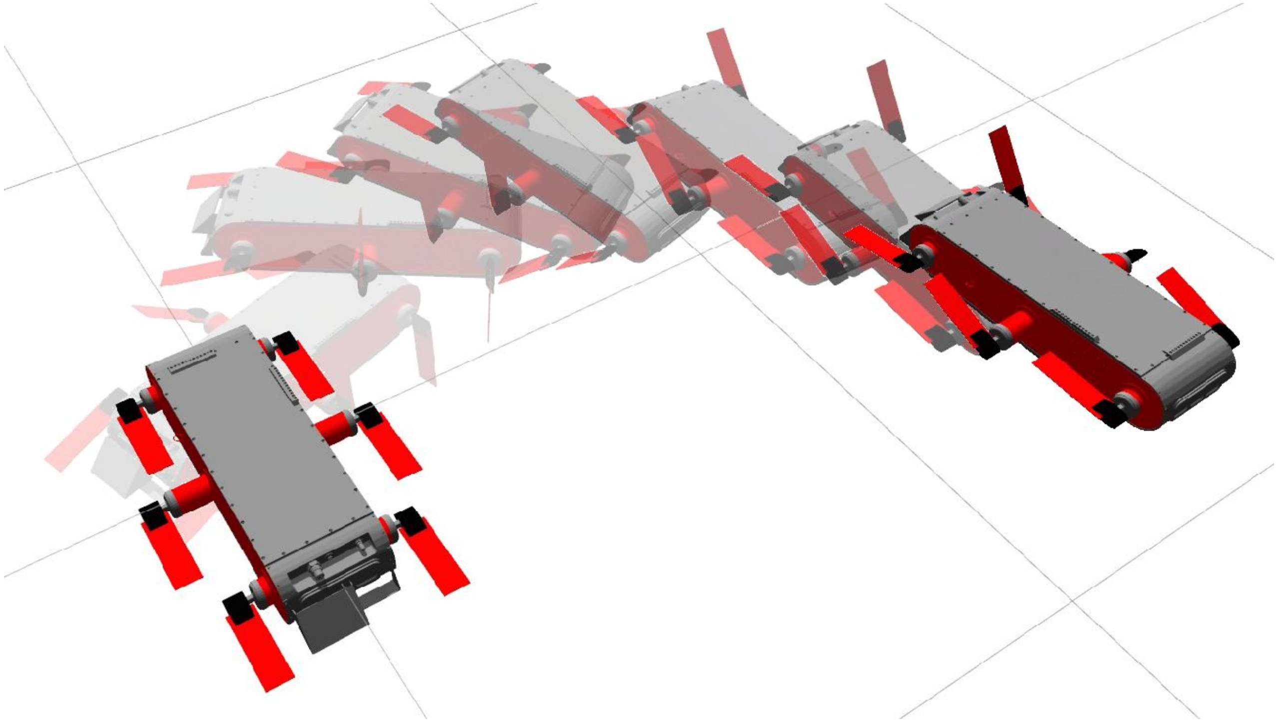
Sets waypoints and swimming modes

- Flat swim
- Constant depth
- 3D pose control

Meger et al, [3D Trajectory Synthesis and Control for a Legged Swimming Robot], IROS 2014

Shkurti et al, Underwater Multi-Robot Convoying using Visual Tracking by Detection, IROS 2017

2. **aqua_description** and **aqua_gazebo**



Software Overview

Gazebo

aqua_hardware_emulator

- Low-level Motor Control
- Internal Sensors

imu_sensor_plugin

hydrodynamics_plugin

flippers_plugin

1 khz loop

Development PC

Emulated interface (ROS)

- gazebo_ros
- aqua_description

Navigation (ROS)

- aquaautopilot
- aqua_depth
- aqua_positioning

User Interface (ROS)

- aquajoy

Controller Learning (ROS)

- Task Marshalling (aqua_marshall)
- Feedback Controllers and Dataset generations (kusanagi_ros)

General Purpose RL

- kusanagi

Emulating the AQUA hardware

aqua_hardware_emulator provides the same interface as **aquahw**, running as a Gazebo plugin

imu_sensor_plugin comes from gazebo_ros_plugins

hydrodynamics_plugin simulates additional underwater forces

flippers_plugin implements a PID controller and simulates propulsive forces for each leg

hydrodynamics_plugin

Do regular rigid body dynamics simulation on gazebo (e.g. with ODE)

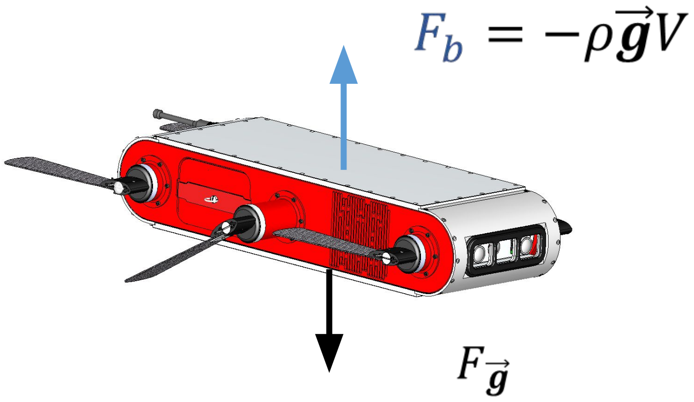
Add hydrodynamic effects and hydrostatic buoyancy as applied forces (addRelativeForce, addRelativeTorque):

$$\begin{bmatrix} \dot{\mathbf{p}} \\ \dot{l} \end{bmatrix} = \left(\begin{pmatrix} \mathbf{M} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{pmatrix} + K_f \right) \begin{bmatrix} \dot{\mathbf{v}} \\ \dot{\boldsymbol{\omega}} \end{bmatrix} = F_{drag} + F_{buoyancy} + F_{gravity} + F_{collisions}$$

Added mass

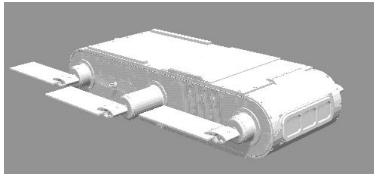
Buoyancy

Compute volume V for each link and apply force at center of buoyancy, of each link, opposite to gravity

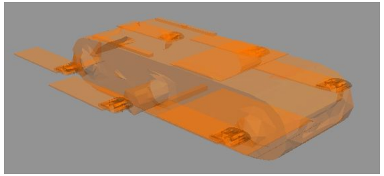


Added Mass and Drag

Precompute drag tensor D and added mass tensor K_f using a simplified mesh geometry (see Weissman and Pinkall (2013) for details)



47k vertices



500 vertices

$$F_{drag} = D\begin{bmatrix}\mathbf{v}\\ \boldsymbol{\omega}\end{bmatrix}, \qquad \begin{bmatrix}\mathbf{p}_f\\ \boldsymbol{l}_f\end{bmatrix} = K_f\begin{bmatrix}\mathbf{v}\\ \boldsymbol{\omega}\end{bmatrix}$$

Added mass modelled as external force

$$F_{added} = \begin{bmatrix}\dot{\mathbf{p}}_f\\ \dot{\boldsymbol{l}}_f\end{bmatrix} = \begin{bmatrix}\boldsymbol{l}_f \times \boldsymbol{\omega} + \mathbf{p}_f \times \mathbf{v}\\ \mathbf{p}_f \times \boldsymbol{\omega}\end{bmatrix}$$

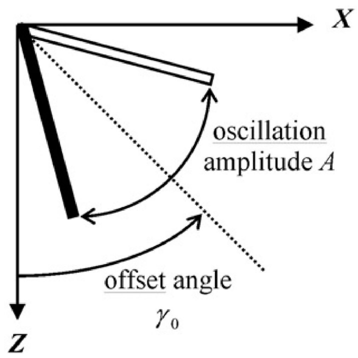
flippers_plugin

Hydrodynamics alone do not model propulsion due to turbulence

We use the empirical model of Plamondon and Nahon, 2013, for each leg

$$\theta_i = A_i \sin(2\pi f_i\, t + \phi_i) + \gamma_{0i}$$

$$F_{propulsive} = k_1(a_i)\frac{A_i}{f_i} + k_2$$

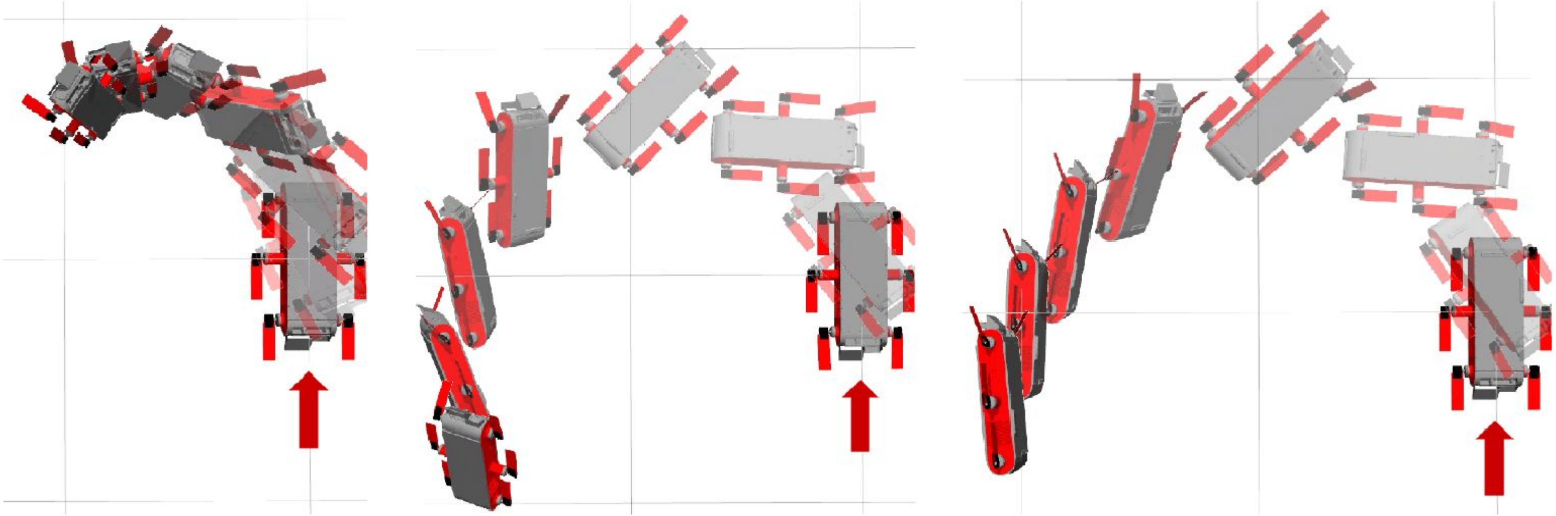



Steps: 1
Real Time Factor:
Sim Time:
Real Time:
Iterations:

Is this simulator accurate?

- It gives a reasonable how the system might behave
- We may improve it with models learned from real-world data

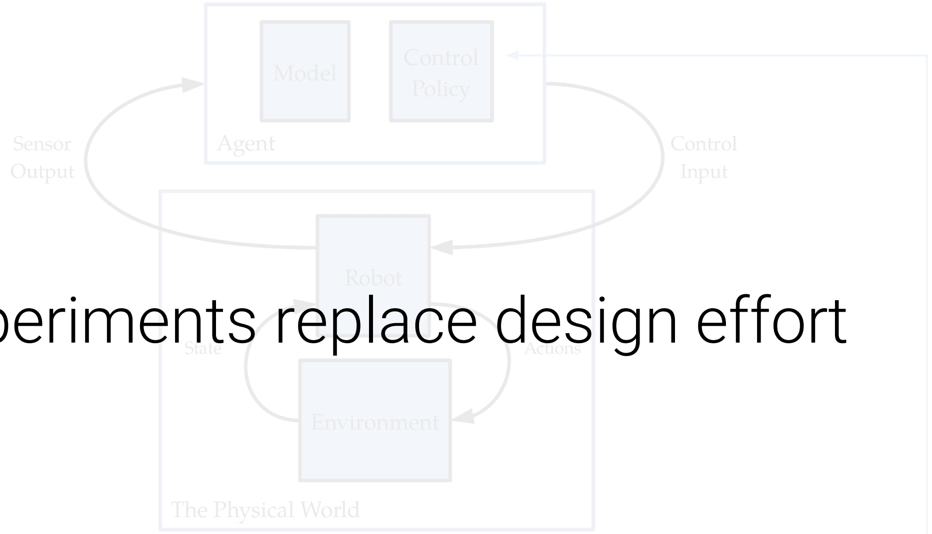
3. Architecture for motor control learning



(a) Episode 1 (b) Episode 5 (c) Episode 10

Adaptive systems realized via learning

Design is never finalized and depends on interactions with the environment (i.e. trial and error)



Experiments replace design effort

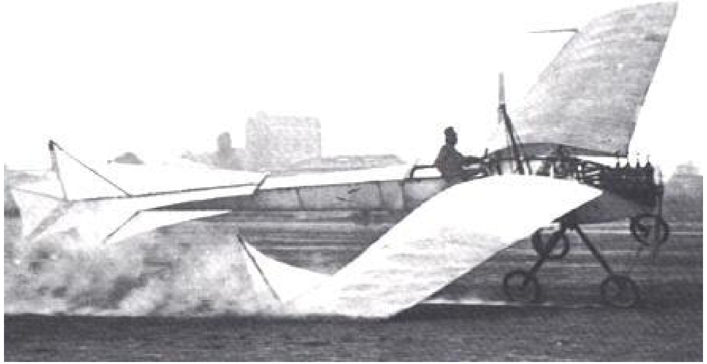
For a given a task, the agent must find a controller that realizes it

The problem with trial and error…

An alternative to robot experiments

Learn basics on a low cost simulator

Adjust skills on the target platform



ROS and robot learning

ROS provides a flexible way of developing robot software

Reinforcement Learning (RL) is a powerful paradigm for solving robotics tasks

We have **generic tools** for developing and testing RL algorithms, on **idealized environments** (e.g. OpenAI Gym)

Controller Learning (ROS)

- Task Marshalling (aqua_marshall)
- Feedback Controllers and Dataset generations (kusanagi_ros)

General Purpose RL

- kusanagi

How can we glue all of this together?

kusanagi library overview

Algorithms
- PILCO
- MC-PILCO
- PDDP
- Policy Adjustments

Utilities
- Experience datasets
- Applying and evaluating controllers
- Plotting

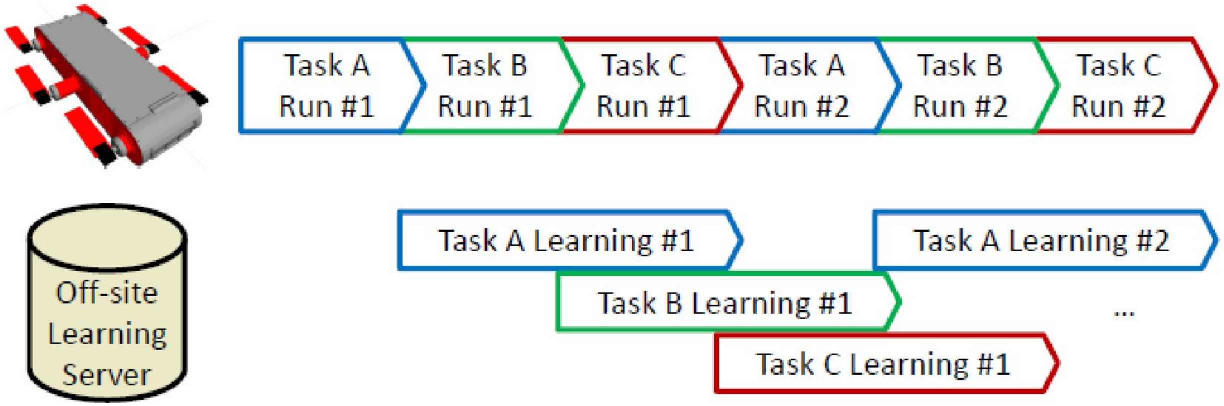
Controllers
- Linear Controllers
- Radial Basis Functions
- Neural Networks

Regression
- Gaussian Process Regression
- Sparse Gaussian Process Regression
- Bayesian Neural Networks

Coming soon at: https://github.com/juancamilog/kusanagi

aqua_marshall and **kusanagi_ros**

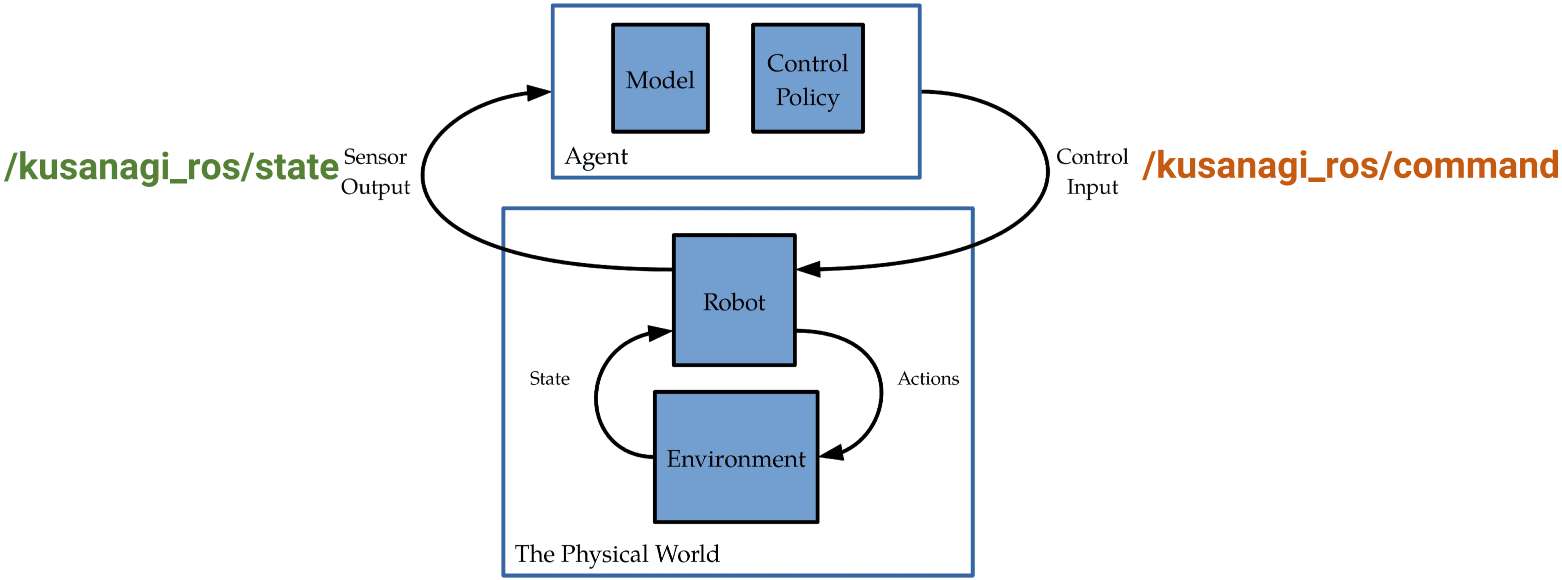
Pipelining of multiple learning tasks



Sequential execution of learned controllers on the robot

Parallel learning of models and controllers on off-site server

Building learning datasets



Aggregate data from multiple sources (ROS topics), into robot-agnostic sensor and control streams

topics_to_rl_streams yaml configuration

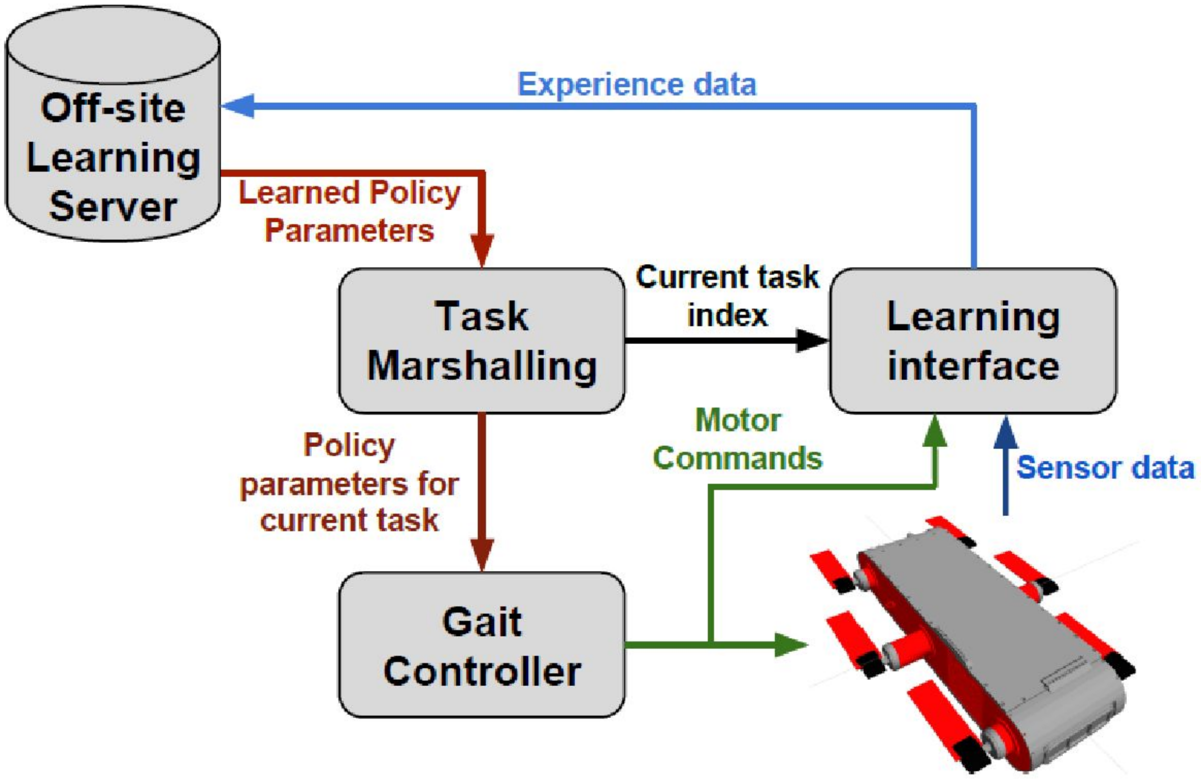
- Specifying state stream (published as **/kusanagi_ros/state**)

```yaml
experience_state_topics:
- topic_name: /aqua/state
type: {package: aquacore, name: StateMsg}
filter: ["RollAngle","PitchAngle","YawAngle", "Depth"]
- topic_name: /aqua/imu_data
type: {package: sensor_msgs, name: Imu}
filter: ["angular_velocity.x","angular_velocity.y","angular_velocity.z"]
```

- Specifying action stream (published as **/kusanagi_ros/command**)

```yaml
experience_command_topics:
- topic_name: /aqua/periodic_leg_command
type: {package: aquacore, name: PeriodicLegCommand}
filter: ["amplitudes[2]","amplitudes[5]","leg_offsets[2]","leg_offsets[5]"]
default_values: { frequencies: 2.5 }
```

Motor Control Learning on AQUA



Controller Learning (ROS)

- Task Marshalling : aqua_marshall
- Feedback Policies and Dataset generation: kusanagi_ros

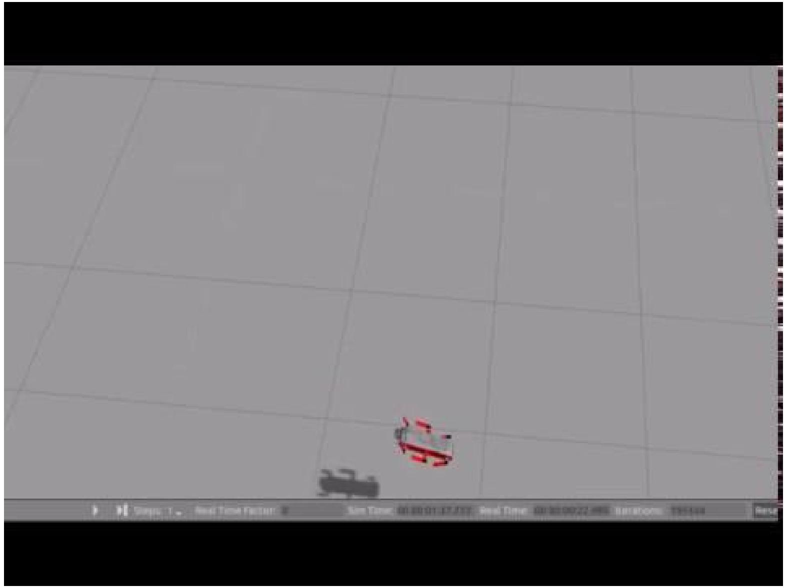
General Purpose RL

- kusanagi

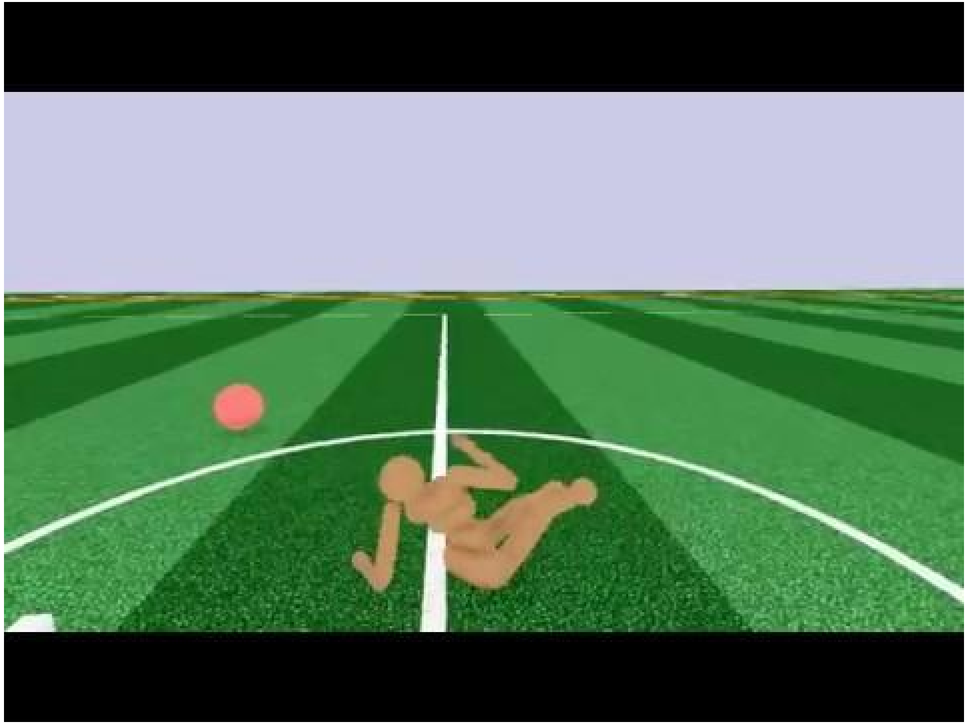
Meger et al, Learning Legged Swimming Gaits from Experience, ICRA 2015

Meger et al, Learning Legged Swimming Gaits from Experience, ICRA 2015

What’s next?



Is this robotics?



Summary

- Gave an overview of ROS packages for motor control on the AQUA robot.
  - **aquahw, aquaautopilot, aquajoy**
- Described gazebo plugins for underwater rigid body dynamics simulation
  - **aqua_gazebo**
- Introduced a generic model-based RL pipeline, and its application to the AQUA robot
  - **aqua_marshall**
  - **aqua_rl/kusanagi_ros**
  - **kusanagi**

https://github.com/mcgillmrl

Thanks!

Some of the contributors:

- Anqi Xu
- Alex Chatron
- Chris Prahacs
- David Meger
- Florian Shkurti
- Joanna Hansen
- Juan Camilo Gamboa Higuera
- Junaed Sattar
- Jimmy Li
- Malika Meghjani
- Nikolaos Pateromichelakis
- Nikhil Kakodkar
- Philippe Giguère
- Sandeep Manjanna
- Travis Manderson
- Victor Barbarosh
- Yogesh Girdhar

https://github.com/mcgillmrl

McGill

NCFRN NSERC Canadian Field Robotics Network

RCCRT Réseau canadien CRSNG pour la robotique de terrain

MRL Mobile Robotics Lab at McGill University