Duckiepond: A Reproducible, Flexible, and ML-Compatible Education and Research Platform for a Fleet of Autonomous Maritime Vehicles

Ni-Ching Lin[1], Yu-Chieh Hsiao[1], Yi-Wei Huang[1], Ching-Tung Hung[2], Tzu-Kuan Chuang[1], Pin-Wei Chen[1], Jui-Te Huang[1], Chao-Chun Hsu[1], Andrea Censi[4], Michael Benjamin[3], Chi-Fang Chen[2], and Hsueh-Cheng Wang[1,*]

Abstract— **Duckiepond is an education and research platform of a fleet of autonomous surface vehicles "Duckieboats" designed to be *reproducible* in hardware, and yet *flexible* to develop in software and machine learning (ML) compatible. The Duckiepond platform is built upon the Duckietown and AI Driving Olympics platform: Duckieboats rely only on one monocular camera, IMU, and GPS, and perform all ML processing using onboard embedded computers. It supports commonly used middleware (ROS and MOOS-IvP) and containerize software packages in Docker, making it easy to deploy. The powerful learning-based together with classic methods enable important maritime missions: track and trail, navigation, and coordinate among Duckieboats to avoid collisions. The Duckieboat has been operating in a man-made lake, reservoir, river, and seashore environments. The platform also includes autonomy education materials for maritime missions. All hardware and software materials are openly available, and we wish that the communities across related domains will adopt the platform for education and research.**

I. INTRODUCTION

A. Motivations

According to the US Coast Guard 4,291 accidents were reported in 2017 among 11,961,568 registered recreational vessels. Those accidents involved 658 deaths, 2,629 injuries and approximately 46 million US dollars of damage to property, and over 1,500 of reported accidents were caused by collisions to other vessels or fixed objects [1]. Self-driving vehicles have become one of the most influential applications, either on the road or on maritime domains such as harbor security. Developing such a technology may have the potential to reduce accidents due to operator inattention, inexperience, violation of navigation rules, or lack of a proper lookout - the major factors causing the reported accidents.

Autonomous robotic software and middleware have been developed for education and research for decades. Mission Oriented Operating Suite - Interval Programming (MOOS-IvP) [2] is an open source project for multi-objective optimization between competing behaviors. It is well-suited in the fields of unmanned surface vehicle and underwater acoustics, and has profound influence in marine robotics for years. Recently the rise of Robot Operation System (ROS) [3] facilitates the development across academia, industry, and government organizations. Since there are majority of users and thousands of modules have been developed, ROS become one of the most commonly-used middleware for robotics. Lots of robots such as PR2, Atlas, UR5, and Turtlebot use it as their software framework. Gazebo, a 3D simulation for general robots applications, is integrated and available to ROS users.

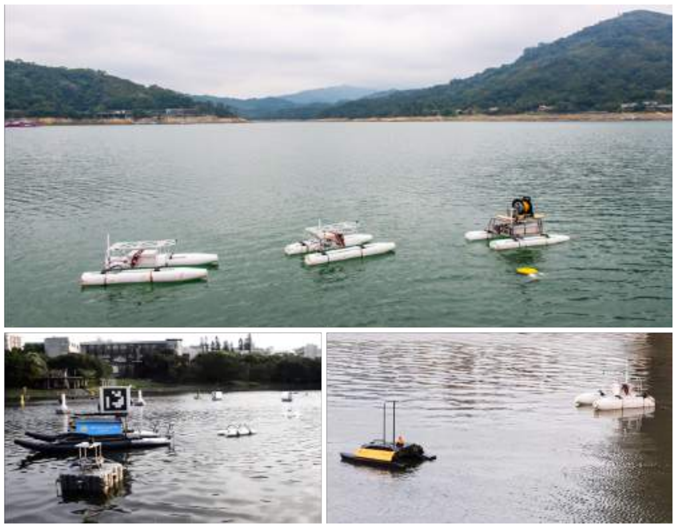


Fig. 1: Duckiepond: a research and education platform for a fleet of homogeneous or heterogeneous autonomous maritime vehicles operating in outdoor fields. The platform includes: *reproducible* hardwares "Duckieboat", *flexible* software and systems that supports heterogeneous vehicles, a set of maritime multi-vehicle behaviors of *machine-learning based* and classic approaches, and education materials for autonomy educations.

A testbed is very much needed to make progress for the safety considerations, and is involving how to deploy the state-of-the-art technologies in a fleet of real robots to avoid collisions. The recent success of the Duckietown platform was one of the miniaturized testbed to develop autonomy education and research [4], [5]. The inexpensive platform includes a team of vehicles built upon ROS, and each vehicle includes only an onboard monocular camera and an embedded computer. A miniaturized city (Duckietown) with roads, signage, and obstacles is designed to tackle the problems of autonomous driving. The Duckietown platform started to adopt Docker and deep learning, such as Convolution Neural Network (CNN) imitation learning and deep reinforcement learning for lane following tasks, and was used in the competition of the AI Driving Olympics (AI-DO) in NeurIPS 2018 [6], [7]. Here, we wish to adapt some of the principles here for the domains of the autonomous surface vehicles, and establishing a safe testbed called Duckiepond.

[1]Department of Electrical and Computer Engineering, National Chiao Tung University, Taiwan. [2]Department of Engineering Science and Ocean Engineering, National Taiwan University, Taiwan. [3]Department of Mechanical Engineering, Massachusetts Institute of Technology, USA. [4] Department of Mechanical and Process Engineering, ETH Zürich, Switzerland. *Corresponding author email: hchengwang@g2.nctu.edu.tw

B. Challenges

Building such an autonomous unmanned marine system for outdoor fields is challenging in many aspects. The hardware design considerations should include waterproofing, compact electronics, as well as a cooling system to handle a variety of weather conditions including heavy rain, strong winds and blistering sun. Due to the resource constrained of power and computation onboard, realtime algorithms should be adapted to *support power-hungry, computation-intensive deep neural networks* for overall performance. The design considerations should also include reproducibility and ideally low-cost to be widely adopted.

For software system, many maritime robots have been developed to support MOOS-IvP and well-tested in fields, including algorithms in software packages and hardware. Nevertheless, reproducing and deploying software packages of autonomous navigation to real robots in general are still challenging, especially for *a fleet of heterogeneous robots to cooperate one another*. Although MOOS-ROS Bridge [8] has been develop to communicate these two robotics middleware, the compatibility issues remain due to various software dependencies in the *fast evolving Ubuntu and ROS distributions*.

C. Contributions

This work contribute the follows:

- *Reproducible in Hardware.* This work aims to develop a safe testbed Duckiepond and a fleet of surface vehicles Duckieboats with essential functionalities of autonomous navigation in real-world maritime environments. The platform and designs will be open source in hardware and software, and we wish the developed functionalities and research could be reproducible.
- *Flexible in Software Develpments.* With the principles of containerization (Docker), we designed and built an autonomous unmanned marine system with the software that is compatible for two commonly-used middleware: ROS and MOOS-IvP. The containerization allows to a) develop under different middleware, b) deploy on real and simulation environments, such as the Virtual Maritime RobotX Challenge (VMRC).
- *Machine-Learning (ML) Compatible.* The hardware and software are designed to support deep learning framework, either on a low-cost Raspberry Pi3 with an accelerator Neural Compute Stick (Caffe and Tensorflow), or a Jetson TX2 embedded computer (PyTorch). We demonstrate that the learning-based algorithms together with classic approaches were able to achieve important maritime tasks including track and trail, avoid obstacles, and navigation with multiple vehicles.
- *Evaluations in Different Environments and Cross-Domain Applications* We have deployed the Duckiepond platform in the following environments: a) an artificial lake in NCTU campus together with a full-sized surface vehicles WAM-V for the practice of the RobotX competition in Hawaii in 2018; b) a reservoir with an underwater robot for inspections; c) a river environments with underwater sound experiments. We wish the proposed platform could be adopted cross domains to facilitate interdisciplinary research.

II. Related Works

A. Existing Platforms of Maritime Vehicles

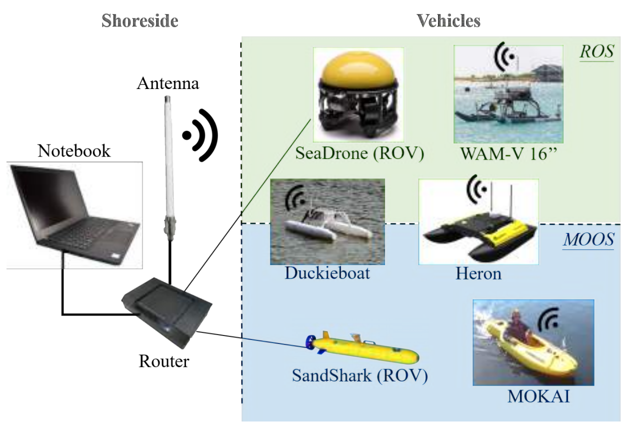
Autonomous or unmanned surface vehicles have been proposed in a variety of considerations, including mechanical systems, capabilities, sensors, for industrial, academic, and military applications. [9]–[13]. Some are extended from commercial crafts such as fishing trawler [14] and kayak [15] by mounting sensors, computing units, and propulsions. Depending on the application scenarios, the buoyance, size, sensor choices, and strength of mechanical structure of the vehicles are designed [16]–[24], including: 1) kayak-style [15], [18], speedboat [25], catamaran [16], [20], [23], [24], and small waterplane twin hull (SWATH) [19], [21]. We summarize some of the existing platforms in Table I in terms of:

- middlewares: the supported system: ROS (R), MOOS (M), or other customized applications, such as Seaware (Sea) or smartphone Apps (App).
- open hardware: the hardware designs are openly available to be reproducible.
- flexible software: the system architecture supports commonly used middlewares such as ROS and MOOS-IvP.
- simulation models: the platform includes simulation such as Gazebo,
- ML-compatible: the hardware and software architecture supports recent deep learning framework such as Caffe, Tensorflow, and PyTorch.

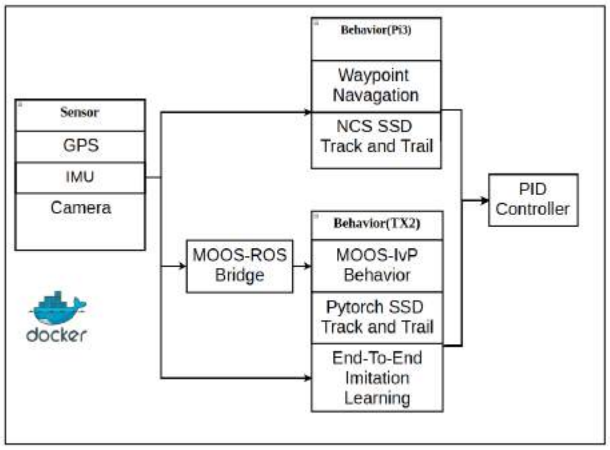
B. Autonomy Education and Competitions and of Maritime Robots

Platforms for autonomy education in robotics are proposed in the past decades (see the review in [4]). In particular our work is inspired by Duckietown initiated in MIT in 2016 and is now offered as courses in over 10 universities globally. The hands-on materials have been adopted in the courses in our campus National Chiao Tung University (NCTU) since spring 2016, the first international branch. The materials are then used in Spring 2017, Fall 2018, Summer 2018, accumulated more than 300 enrolled students and thousands of visitors in outreach efforts in Taiwan. The materials include classic vision-based solutions for lane following task, including hardware assembly, ROS, programming in Python, OpenCV, state estimation (Bayes' filter), Docker, and deep learning frameworks [4], [6].

For the maritime autonomy education, MIT 2.680 [26] covers software and algorithms for autonomy involving decision making. In particular the MOOS-IvP autonomy software infrastructure is introduced for developing integrated sensing, modeling and control solutions. The simulation environments and a field testbed with small autonomous surface craft and underwater vehicles operated on the Charles River. There are also educational certificate programs [27] for unmanned maritime vehicles that provide students with the knowledge operating of a variety of autonomous marine survey vehicles. The program aims at covering the decision making and mission planning based on the environment factors gathered from sensor data.



(a) Duckiepond Architecture.



(b) Duckieboat Systems.

Fig. 2: (a) Duckiepond includes ; (b) The Duckieboat is an easy setup platform by using docker and base on ROS. There is GPS, IMU, and Pi camera on this platform and is able to run machine learning. With a bridge, MOOS is also available in this platform.

Simulations in Gazebo have also widely adopted in robotics education and competitions [28]–[30]. The VMRC (2019) includes rich sensors, objects, and robot models, and is an open source to support developments and competitions of autonomous surface vehicles. Some tutorials are available including the qualification task in the RobotX competition [31], and a baseline solution using given GPS locations and PID controllers. There are also maritime robotics competitions in real environment to motivate students, such as RoboBoat [32]. We wish the proposed Duckiepond platform could be adopted in the community for autonomy education.

TABLE I: Platforms and Robot Boats Comparisons.

	Middlewares	Open Hardware	Multi-Vehicle	Simulation Model	ML-competible
Education and Research Platform					
Duckietown [4], [6]	R	✓	✓	✓	✓
Duckiepond	R+M	✓	✓	✓	✓
Robot Boats					
Duckieboat	R+M	✓	✓	✓	✓
Heron [33]	R+M	✗	✓	✓	✗
SwordFish ASV [16]	Sea	✗	✓	✓	✗
Hydrometra USV [34]	App	✗	✗	✗	✗
GeoSwath 4R USV [35]	App	✗	✗	✗	✗
SCOUT ASV [15]	M	✗	✓	✓	✗
Calypso [36]	M	✗	✗	✓	✗
Datamaran [37]	M	✗	✓	✓	✗

Note: Middlewares: supported middleware API or systems, Open Hardware: hardware design sufficiently released for reproduction, Multi-Vehicle: the capability of performing multi-agent missions, Simulation Model: vehicle model in a simulation environment. ML-compatible: initially set up machine learning hardware and software system. The short form of middleware, R represents ROS, M represents MOOS-IvP, Sea represents Seaware, and App represents smartphone Apps.

III. Duckiepond: System Architecture

A. Overview

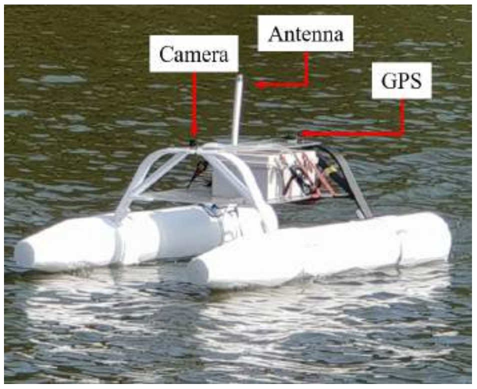
The overall multi-robot system, presented in Fig 2a, is designed to support a fleet of homogeneous and heterogeneous maritime vehicles. We consider the proposed Duckieboat (details in IV) is capable of carrying out multi-vehicle behaviors with other Duckieboats, or heterogeneous robots, such as commercially available Clearpath Heron, a 16' Wave Adaptive Modular Vessel (WAM-V), and a tethered underwater vehicle SeaDrone on the top payload (see Fig. 1).

B. Containerization for Algorithms in ROS & MOOS

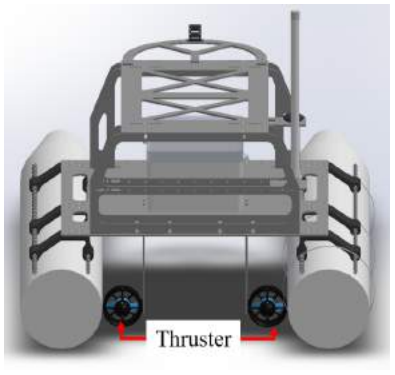
The Duckieboat system is built on Ubuntu 16.04 and ROS Kinetic as host, and packaging software into standardized units in Docker containers [38] for development (laptop) and deployment (NVIDIA TX2, and Raspberry Pi3). Such settings ensure the software integrations of required dependent libraries, machine learning tools, and the robot hardwares. Based on the underlying system, we could take advantage of the packages such as Caffe, Tensorflow, PyTorch, and Intel Neural Compute Stick (NCS) SDK to achieve learning-based capabilities. Our goal is to support two of the commonly used middlewares: ROS and the MOOS-IvP (mission-oriented operating system IvP helm) [2]. In specific, we consider two major reasons to contain *both* middlewares. First, the education needs support from the communities, since many existing educational courses can be maintained with minimal additional work for integration. Second, some competitions such as Virtual Maritime RobotX Challenge (VMRC) only provides Gazebo and ROS protocols, and hence the MOOS community could use the MOOS-ROS bridge [8] to participate the competition by applying established algorithms MOOS over ROS systems and simulations.

C. Depolyments in Simulation (Gazebo) and Real Robots

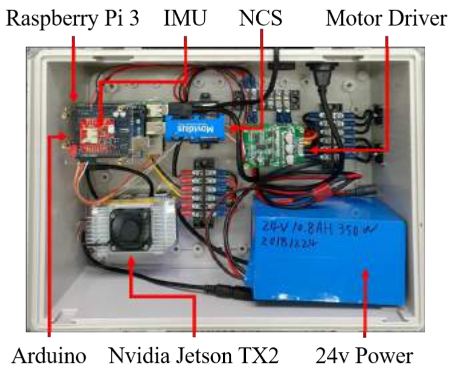
Gazebo is a simulator which can simulate robots in complex indoor and outdoor environments, besides, it is well-supported via ROS. By using the simulator, it is possible to rapidly design



(a) Duckieboat



(b) Rear view of Duckieboat.



(c) Embedded computers and battery.

Fig. 3: Duckieboat designs. (a) Duckieboat includes a camera, IMU, and GPS sensors; (b) differintial drive propulsion; (c) the computing units include a Raspberry Pi3, neural compute stick, and NVidia Jetson TX2 inside a waterproof container.

robots, test the algorithms and train AI system. The models of Duckieboats have already built and experimented in Gazebo.

IV. The Duckieboat

A. Duckieboat Overview

”Duckieboats” are autonomous surface vehicles with the flexibility, capability, machine learning compatibility and objectives of affordability. The Duckieboat system is working on Linux with Ubuntu 16.04 and ROS Kinetic. Based on the underlying system, append PyTorch, MOOS-IvP, NCS SDK to achieve more capability. For the unified and cross-platform needs, we construct the system in Docker [38] container including all software above, dependent libraries and machine learning tools (see Fig 2b). The structure of Duckieboat is build up with 3D printed materials and laser cut boat. The dimensions are $1.5 \times 0.75 \times 0.6m$. With a weight in air of $20kg$ and can carry $10kg$. The system time of autonomy approximately four hours and the power is supply by a 24 voltage Li-battery.

B. Design Considerations

Duckiepond is developed as an education platform of the Duckieboat autonomous surface vehicle. It must have the main sensors, computation, and motor to provide automation behaviors, such as navigation and detection. Rainproof for outdoor experiments, a dry and waterproof environment is needed to protect all the electrical components. Additionally, shore side and vehicle side wireless communication allow information exchange while giving a command or remote control within $100m$.

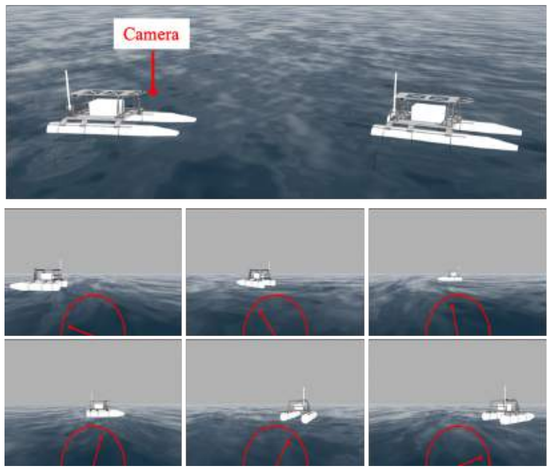
1) Waterproof: The Duckieboat uses a watertight box (see Fig 3b) with a waterproof connector to provide a dry environment because most of the electrical devices would be damaged by water and the sensors beyond the box are protected by waterproof spray. This could allow Duckieboats to work in several water environments, such as lakes and rivers.

2) Sensors: The Duckieboat claim as a minimal navigation system of an autonomous surface vehicle with a GPS and an IMU. To receive high accuracy GPS data, the GPS antenna is set on the top plate to avoid positioning error and the IMU is an electrical module set in the waterproof box. By combining the heading data from the IMU and position data from the GPS, we obtain rough navigation information for the Duckieboat. However, there is precision error through GPS. In this case, Duckieboat uses a Raspberry Pi Camera for gaining local information. With a camera, the Duckieboat could avoid collision (see Fig 3a).

3) Communications: The Duckiepond environment is set to $80m \times 80m$ in water and there is a shoreside beside the water within $20m$ because of the communication system. The shoreside station which includes a computation device, a joystick, a router, and an antenna. The shoreside station serves as a master device and connects to the clients (Duckieboats). The Duckieboat has its antenna wired to an onboard client router which can communicate to the shoreside router (master) via wireless 2.4G WiFi. The antenna provides a communication range to approximately $100m$. This antenna could be upgraded to longer distance and could extend Duckiepond environment.

4) Propulsions: The Duckieboat uses two brushless DC motors in a differential drive configuration. Brushless motors are the most commonly used motor for marine robotics because of their high torque, small size, and its low electrical noise. The motors are mounted under the floating ski to provide a small radius of curvature (approximately 0.75 meter) while rotating. The Duckieboat max speed is about $1.8m/s$ come from two thruster of force $5.5lbs$ (see Fig 3b).

5) Computing Units: Embedded systems are used for Duckieboat. The basic system provides GPS and IMU navigation with waypoints via Raspberry Pi3. To add machine learning capabilities, we have two solutions. The first one is to connect an NCS to the Raspberry Pi3. The NCS is a neural network hardware accelerator with USB interface on Raspberry Pi3 designed to run deep learning algortihms. It is compatible with Caffe or Tensorflow by converting deep learning models to graph files. Pi3 with NCS performs ten times faster than only use Pi3 while using MobileNet [39]. Additionally, it is much cheaper than other embedded system that includes a GPU. For the reasons above, we believe NCS could be the primary and guide materials of deep learning in the Duckieboat education platform. The second solution is to connect the Raspberry Pi3 to a Jetson TX2, which provides a powerful AI computing GPU. Several deep learning algorithms are tested such as single shot multi-box detector (SSD). The Jetson TX2 is connected to Raspberry Pi3 via ethernet which extends the limits of NCS, such as model size, specific framework, memory, and computing



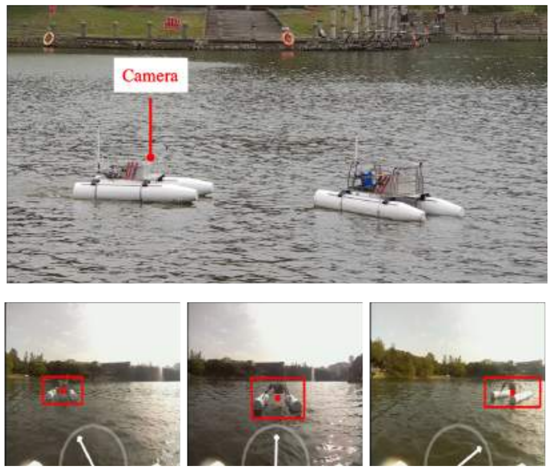


Fig. 4: The Duckieboats are performing track and trail mission. The left image is the gazebo simulation of two Duckieboats and the right image is an real boats running track and trail in Bamboo Lake.

TABLE II: Duckieboat is designed for low-cost and ML-compatiable.

Subsystem	Item	Cost
Essentials		Total: **1,085**
Computation	Raspberry Pi 3 B + 8GB SD card & Arduino	$47
Sensing	Camera, GPS & IMU	$78
Power	Battery	$90
Communication	2 Routers & 2 Antennas	$180
Propulsion	2 motors & 2 motor drivers	$350
Backbone	Lasercut chassis and 3D print mount	$300
Misc.	Cables, Screws, Nuts & Waterproof connector	$40
ML Extensions		Total: **498**
Computation	NVIDIA Jetson TX2	$399
Computation	Intel Neural Compute Stick (NCS)	$99

time.

V. Multi-Vehicle Behaviors - Learning and Classic Approaches

To prove that duckieboats not only can process robotic vision algorithms with machine learning but can also implement multi-vehicles behaviors, we choose the Track and Trail mission and deal with the task by three different approaches.

A. Object Tracking with Single-Shot Multibox Detector (SSD)

To avoid the localization error caused by sensors, the most intuitive way is by only using the camera to follow the lead boat. Therefore, we choose the MobileNet [39] Single-Shot Multibox Detector to approach the mission. MobileNet-SSD is modified based on MobileNet referred to VGG-SSD. To train the model, we collected 3183 images from a follower Duckieboat in the real world and labeled bounding boxes of the boats in PASCAL-VOC format. In order to increase performance, we use Caffe to train our deep learning model for 30000 iterations with pretrained caffemodel based on VOC datasets VOC2007 and VOC2012.

After collected and trained, the vehicle is able to find the bounding box of target duckieboat through the camera. Then, we compute the center point and the height of the bounding box as (C_x, C_y) and H_{bbx}, and we also name the image height and width as H_{img} and W_{img}. The tracking control system is divided into two parts: angular and distance control. We define the angular error as $\frac{C_x - W_{img}/2}{W_{img}/2}$ and the distance error is defined as $\frac{K - H_{bbx}}{H_{img}}$, which K means the height of the bounding box that we would like to pursue. After defining these two errors, we use the PID controller on these two systems.

B. Classic Approaches using GPS and IMU

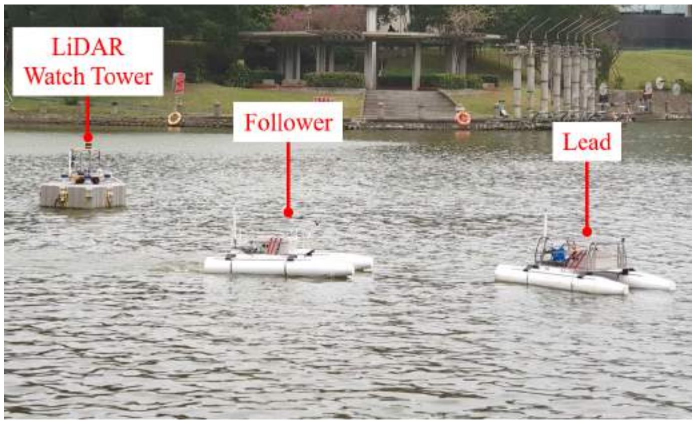
By using GPS and IMU, the duckieboat is able to estimate the position and orientation of itself. To reduce the error of sensors, we apply the Kalman Filter to obtain more precise estimations. After communicating with all other duckieboats through the shoreside station, the boat can track the target and complete missions.

C. Combine Learning and Classic Approach

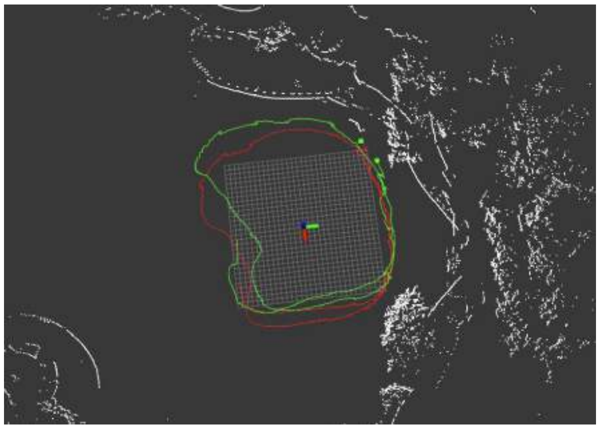
We supposed that the classic approach would cause an unstable trajectory due to the sensor error of localization, but it can complete the task in any distance and orientation. On the other hand, the learning approach would smoothly track the lead boat, but it may fail while wrong prediction or out of the camera field of view and range. Therefore, we combine both commands from different methods and assign the weights which rely on the distance of two Duckieboats. While two boats are near, the learning approach will get more weight, and vice versa.

D. End-to-end Track and Trailing

Owing to the convenience of collecting large training data without the label processing and behavior cloning by human policy, imitation and end-to-end learning are becoming more popular. Therefore, we refer the Tensorflow imitation-learning tutorials from AI Driving Olympics to approach Track and Trail mission. By using both manual joystick control and autonomous tracking vehicle, the duckieboat can quickly get the amount of training data. By this deep learning model, the duckieboat can get the boat command prediction by input images without any other processing.



(a) Experiment Environments.



(b) Visualizations of trajectories from LiDAR watchtower.

Fig. 5: Experiment setup.

TABLE III: Education Materials.

Modules	**Duckietown**	**Duckiepond**
Hardware Assembly	Duckiebot	Duckieboat
	Camera	Camera, GPS, IMU
	RaspPi3 + NCS	RaspPi3 + NCS + TX2
Computer Vision	Edge/Color Detection	
	Ground Projection	
State Estimation	Bayes' Filter	EKF
Deep Learning	End-to-end IL	End-to-end IL
	Deep RL	SSD
Tasks	Lane Following	Waypoint Navi.
	Vehicle Following	Track & Trail
	Intersection Coord.	

VI. Education Materials

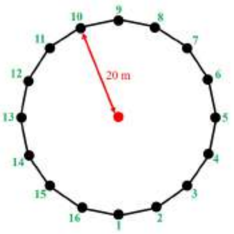
All materials related to the project are released under an open-source license and available at https://robotx-nctu.github.io/duckiepond. The step by step tutorial includes basic knowledge about autonomous surface vehicles and state-of-the-art deep learning image processing algorithms. By using Docker containers, it is possible for students to play around arbitrarily but still easy to recover from mess. The hardware is also open-sourced so students can build the Duckieboat from scratch and learn the components when building robots. There is also extendibility for academic use.

VII. Experiments

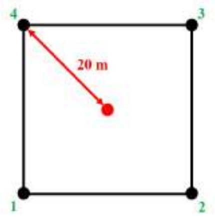
Track and trail task is selected for evaluations, shown in Fig 5a, to demonstrate the capabilities: 1) the lead vehicle runs waypoint navigation or is tele-operated; 2) the follower vehicle keeps a certain distance from the lead; 3) both vehicles should avoid collisions to each other or fixed objects in the experiment site.

A. Experiment Site and Settings

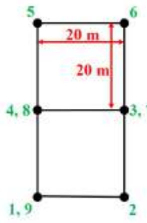
The experiments took place at Bamboo Lake, an artificial reservoir in the campus of the National Chiao Tung University (NCTU), Taiwan. The experiment site is relatively still compared to river and ocean environments, but with occasional strong winds. The site is around $60\ m \times 80\ m$ meters. We designed 3 routes, including different numbers of waypoints and angular difference in each route, shown in Fig 6. (a) Circle-shaped trail is based on 16 waypoints. This trail is the easiest task for track and trail because it has less angular difference



(a) Circle-shaped.



(b) Rectangle-shaped.



(c) 8-shaped.

Fig. 6: Routes for track and trail experiments.

and the speed of motors output should be stable. (b) Rectangle-shaped has 4 waypoints with 90° turn, considering moderate in difficulty. (c) 8-shaped has 8 90° turn waypoints and have both turns right and turn left with shorter straight route (20m), which is the most difficult route.

B. Methods

The lead vehicle is automatically operated around $1.3m/s$ through each waypoint, followed by the follower vehicle using the proposed methods: 1) a vision-based learning approach via SSD and an arbitrary controller, 2) classic approach developed in MOOS-IvP using GPS and IMU fusion, and 3) a combined method of 1) and 2), in which the control commands (velocity and angular velocity) were combined. For above 3 methods, we set 5 meters as the ideal distance of the lead and follower vehicles, but should avoid collisions or unsafe scenarios (1.5 meters). This distance is setup by the duckieboat max length to include any collision chance.

C. Evaluation Matrics

We record the positions of two vehicles from 1) the estimated locations from GPS and IMU fusion in each vehicle, and 2) a watchtower on a anchored platform with a 3D LiDAR Velodyne VLP-16 is set at the center of experiment (see Fig 5a). The point clouds from two vehicles are then clustered to estimate the positions of two vehicles. LiDAR-based evaluation is more accurate than GPS-based, although the former might be difficult in other experiment site without anchored floating platform.

The *trailing distances* between the lead and the follower vehicles are then measured according to the position and trajectory data, shown in Fig 7. The ideal distance between

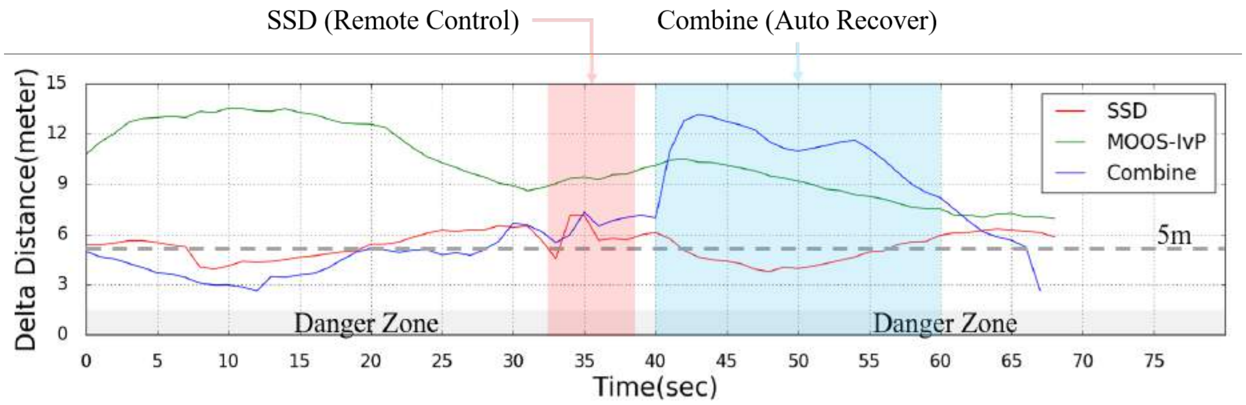


Fig. 7: The measurement of three methods trailing distances between two vehicles. Classic approach is a way far to the following distances (5 meters). SSD approach has the best trailing distance. However, there is a switch from auto to remote control because the follower vehicle can't see the lead vehicle shown in red. Combine approach could recover while the follower couldn't see the lead vehicle shown in blue.

two vehicles should maintain 5 meters but must not be shorter than 1.5 meter. We further record the number of collisions as *safety* measure, and the number of human interventions as *autonomy* measure.

D. Results and Discussions

The overall results are shown in Table IV.

1) Trailing Distance: We demonstrate that vision-based learning is well-suited for the short distance (5 meters) track and trail, given the average distance between two vehicles is the closest to 5 meters compared to the other two methods. GPS-based method provides is larger because of the low-cost GPS sensor error. Combined vision-based learning and classic method shows a way in between: we set the follower with a higher weight to use vision-based learning method in short trailing distance, whereas a higher weight of classic method while the trailing distance is longer.

2) Safety: The learning approach is in general good to maintain a certain trailing distance, but misclassification (false detection of leader vehicle) leads to collisions. Currently the SSD method does not include the watchtower, and therefore is not able to perform collision avoidance around unrecognized objects. In classic method the GPS errors and the low framerate (1 frame per second) of GPS may cause that the follower vehicle getting too close to the lead one. More importantly, we observed unreliable network communications that caused missing GPS data, resulting in 1 collisions.

3) Automony: For a more challenging 8-shaped route, we observed 4 times of human interventions of the vision-based learning method, due to that the lead vehicle is out of the field of view (FOV) of the follower vehicle. The classic method from GPS can mostly maintain autonomous mode. The combined method seems a good way for auto-recovery of the vision-based learning disadvantage of limited FOV.

VIII. CONCLUSIONS

We propose a platform "Duckiepond," which supports a fleet of autonomous maritime vehicles for research and education. The Duckieboat is designed as reproducible, ML-compatible, and low-cost, to carry out homogeneous multi-vehicle research as well as hetergeneous vehicle teams, such as underwater vehicles or a larger vehicle WAM-V equipped more sensors. The openly available educational materials , including Duckieboat specification, hands-on modules, and tutorials are available for hardware assembly, middleware, computer vision, state estimation, and machine learning. Our experiments show performance of track and trail task using vision-based learning, classic, and combined methods.

TABLE IV: Track and trail performance in a navigation task.

Trail Routes	Learning	Classic	Combined
Circle-shaped			
No. of Trials	6	6	6
Distance (avg.)(m) GPS	**5.268**	9.999	9.309
Distance (std.)(m) GPS	**0.986**	2.972	4.485
Distance (avg.)(m) LiDAR	5.05	-	7.607
Distance (std.)(m) LiDAR	1.235	-	2.998
N of Collision	0	0	**0**
N of Manual Intervention	0	0	**0**
Rect-Shaped			
No. of trails	6	6	6
Distance (avg.)(m)	6.089	11.216	7.62
Distance (std.)(m)	4.137	3.634	1.171
Collision	0	1	**0**
Manual	1	2	**0**
8-Shaped			
No. of Trials	6	6	6
Distance (avg.)(m)	5.161	5.87	6.967
Distance (std.)(m)	2.005	1.972	2.554
Collision	1	0	**1**
Manual	4	0	**1**

The Duckieboats have been tested in artificial lake, reservoir, and river, and we wish that such platform could be used to facilitate basic scientific research and investigate complex phenomena in nature and engineering in fields. Future work will include: 1) track and trail with underwater vehicle and more multi-vehicle behaviors with machine-learning approaches; 2) a "super Duckieboat" equipped with more sensors and advanced waterproof and self-sustained for a couple of months for river and ocean scenarios for climate observatory or bio-acoustic detection platform; 3) we plan to use Duckieboat for a large-scale environment on navigation, mapping and coordination with multiple vehicles such as harbor security.

ACKNOWLEDGMENT

We are thankful for the inspiration by Liam Paull and Duckietown community. We are also grateful for the help of Chih-Ping Lin, Hsiao-Sheng Chen and Lim Sin Kiat. The research was supported by Ministry of Science and Technology, Taiwan (grant 107-2221-E009-156, 107-2218-E007-051, 105-2511-S-009-017-MY3).

REFERENCES

[1] (2017) Recreational boating statistics. [Online]. Available: https://www.uscgboating.org/library/accident-statistics/

[2] M. R. Benjamin, J. J. Leonard, H. Schmidt, and P. M. Newman, "An overview of MOOS-IvP and a brief users guide to the ivp helm autonomy software," 2009.

[3] Robot Operation System. [Online]. Available: http://www.ros.org/

[4] L. Paull, J. Tani, H. Ahn, J. Alonso-Mora, L. Carlone, M. Cap, Y. F. Chen, C. Choi, J. Dusek, Y. Fang, *et al.*, "Duckietown: an open, inexpensive and flexible platform for autonomy education and research," in *Robotics and Automation (ICRA), 2017 IEEE International Conference on*. IEEE, 2017, pp. 1497–1504.

[5] Duckietown MIT. [Online]. Available: http://duckietown.mit.edu/

[6] The AI Driving Olympics. [Online]. Available: https://nips.cc/Conferences/2018/CompetitionTrack

[7] The AI Driving Olympics. [Online]. Available: https://AI-DO.duckietown.org

[8] K. DeMarco, M. E. West, and T. R. Collins, "An implementation of ros on the yellowfin autonomous underwater vehicle (auv)," in *OCEANS 2011*. IEEE, 2011, pp. 1–7.

[9] V. Bertram, "Unmanned surface vehicles-a survey," *Skibsteknisk Selskab, Copenhagen, Denmark*, vol. 1, pp. 1–14, 2008.

[10] R.-j. Yan, S. Pang, H.-b. Sun, and Y.-j. Pang, "Development and missions of unmanned surface vehicle," *Journal of Marine Science and Application*, vol. 9, no. 4, pp. 451–457, 2010.

[11] Z. Liu, Y. Zhang, X. Yu, and C. Yuan, "Unmanned surface vehicles: An overview of developments and challenges," *Annual Reviews in Control*, vol. 41, pp. 71–93, 2016.

[12] J. E. Manley, "Unmanned surface vehicles, 15 years of development," in *OCEANS 2008*. IEEE, 2008, pp. 1–4.

[13] J. Yuh, G. Marani, and D. R. Blidberg, "Applications of marine robotic vehicles," *Intelligent service robotics*, vol. 4, no. 4, p. 221, 2011.

[14] T. W. Vaneck, C. D. RODRIGUEZ-ORTIZ, M. C. Schmidt, and J. E. Manley, "Automated bathymetry using an autonomous surface craft," *Navigation*, vol. 43, no. 4, pp. 407–419, 1996.

[15] J. Curcio, J. Leonard, and A. Patrikalakis, "Scout-a low cost autonomous surface platform for research in cooperative autonomy," in *OCEANS, 2005. Proceedings of MTS/IEEE*. IEEE, 2005, pp. 725–729.

[16] H. Ferreira, R. Martins, E. Marques, J. Pinto, A. Martins, J. Almeida, J. Sousa, and E. Silva, "Swordfish: an autonomous surface vehicle for network centric operations," in *Oceans 2007-Europe*. IEEE, 2007, pp. 1–6.

[17] M. Dunbabin, A. Grinham, and J. Udy, "An autonomous surface vehicle for water quality monitoring," in *Australasian Conference on Robotics and Automation (ACRA)*, 2009, pp. 2–4.

[18] H. K. Heidarsson and G. S. Sukhatme, "Obstacle detection and avoidance for an autonomous surface vehicle using a profiling sonar," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE, 2011, pp. 731–736.

[19] M. Dunbabin and A. Grinham, "Experimental evaluation of an autonomous surface vehicle for water quality and greenhouse gas emission monitoring," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. IEEE, 2010, pp. 5268–5274.

[20] M. Fahad, N. Saul, Y. Guo, and B. Bingham, "Robotic simulation of dynamic plume tracking by unmanned surface vessels," in *Robotics and Automation (ICRA), 2015 IEEE International Conference on*. IEEE, 2015, pp. 2654–2659.

[21] M. Dunbabin, B. Lang, and B. Wood, "Vision-based docking using an autonomous surface vehicle," in *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*. IEEE, 2008, pp. 26–32.

[22] M. Breivik and J.-E. Loberg, "A virtual target-based underway docking procedure for unmanned surface vehicles," in *Proceedings of the 18th IFAC World Congress*, vol. 18, 2011, pp. 13 630–13 635.

[23] A. Martins, H. Ferreira, C. Almeida, H. Silva, J. M. Almeida, and E. Silva, "Roaz and roaz ii autonomous surface vehicle design and implementation," in *International Lifesaving Congress 2007*, 2007.

[24] J. E. Manley, A. Marsh, W. Cornforth, and C. Wiseman, "Evolution of the autonomous surface craft autocat," in *OCEANS 2000 MTS/IEEE Conference and Exhibition. Conference Proceedings (Cat. No. 00CH37158)*, vol. 1. IEEE, 2000, pp. 403–408.

[25] [Online]. Available: https://maritimerobotics.com/mariner-usv/

[26] Mit 2.680 - unmanned marine vehicle autonomy, sensing and communications. [Online]. Available: http://oceanai.mit.edu/2.680/pmwiki/pmwiki.php?n=Main.HomePage

[27] Unmanned maritime systems (ums) certificate program. [Online]. Available: https://www.usm.edu/school-ocean-science-and-technology/unmanned-maritime-systems-ums-certification

[28] DARPA Robotics Challenge simulator. [Online]. Available: https://bitbucket.org/osrf/drcsim

[29] DARPA Subterranean challenge (subt). [Online]. Available: https://bitbucket.org/osrf/subt/wiki/Home

[30] Virtual Maritime RobotX Challenge (VMRC). [Online]. Available: https://bitbucket.org/osrf/vmrc

[31] Maritime RobotX Challenge. [Online]. Available: https://www.robotx.org/

[32] RoboBoat Challenge. [Online]. Available: https://www.auvsifoundation.org/competition/roboboat

[33] [Online]. Available: https://www.clearpathrobotics.com/heron-unmanned-surface-vessel/

[34] [Online]. Available: http://www.siralab.com/hydrometra-usv/

[35] [Online]. Available: https://www.km.kongsberg.com/ks/web/nokbg0240.nsf/AllWeb/99773800A886356BC1257EF800369BDB?OpenDocument

[36] [Online]. Available: https://www.dotocean.eu/products/autonomous-systems/calypso/

[37] [Online]. Available: http://www.automarinesys.com/datamaran/

[38] D. Merkel, "Docker: lightweight linux containers for consistent development and deployment," *Linux Journal*, vol. 2014, no. 239, p. 2, 2014.

[39] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.