Unity3D and Robot Operating System applied to the development of a multi-UAV strategy to identify oil spills around offshore platforms

Kelen C. Teixeira Vivaldini[1,2], Tatiana F. P. A. T. Pazelli[3] and Team Flying U2

***Abstract*— This paper presents a comprehensive approach to oil spill detection, inspection, and monitoring using Unmanned Aerial Vehicles (UAVs). A simulated environment was developed to illustrate and provide dynamic and realistic data of a case of oil spill around an offshore platform based on Unity 3D and ROS frameworks. The validation of the multi-UAV collaborative system was conducted to detect, inspect, and monitor oil spills in this environment. The results demonstrate the effectiveness of using simulated environments, which allows the system to be tested and validated faster and more cost-effectively, thus accelerating the development of the research and facilitating the gaining of useful knowledge about the issue in question.**

I. INTRODUCTION

Nowadays, UAVs are being adopted for a variety of applications. Simulators are a highly valued research tool in robotic systems, as they accelerate the development process of the project, reduce cost, and mitigate failures and problems that could only be verified in the real environment. Simulated environments also allow users to emulate obtaining accurate and real-time information, which is essential for effective decision-making in validating the project development process.

Due to the specific needs of each application, customization of simulated environments is becoming increasingly necessary. Some works have been dedicated to the development of tools for the creation of custom environments [1], [2], [3], [4], [5], [6].

The application of environment monitoring using UAVs requires high-quality visual simulators. To ensure accuracy and effectiveness, UAVs can be equipped with specific sensors to detect the problems in each application. And then, this allows researchers to identify specific points that need attention and to develop solutions accordingly. Simulated environments have been widely adopted in research to improve proposed solutions. These simulations provide a safe and controlled environment to test and evaluate different strategies and algorithms, allowing researchers to gain valuable insights into the problem at hand.

[1]Kelen Vivaldini - Flying U2 coordinator is with Department of Computer, Federal University of São Carlos - UFSCar, Brazil vivaldini@ufscar.br

[2]Kelen Vivaldini is with the Department of Cybernetics, Faculty of Electrical Engineering, Czech Technical University in Prague teixekel@fel.cvut.cz

[3]Tatiana Pazelli - Flying U2 coordinator, is with the Department of Electrical Engineering, Federal University of São Carlos - UFSCar tatianapazelli@ufscar.br

[4] Team Flying U2 (UFSCar/USP) is formed by researchers from the Federal University of São Carlos (UFSCar) and the University of São Paulo (USP). flying.ufscar.usp@mail.com

In this article, we present a simulated environment aimed to illustrate and provide dynamic and realistic data of a case of oil spill around an offshore platform. Furthermore, we discuss the application of multi-UAVs as a proposal to detect and map the slicks and highlight the importance of having simulators with superior visual quality to improve the development of research in the area of monitoring.

II. ENVIRONMENTS - UNITY SIMULATOR AND ROS

The Unity 3D Engine [7] was used to generate a realistic simulated environment in conjunction with the Robot Operating System (ROS) framework [8]. ROS is a popular framework for robotics, and the Unity 3D community provides many examples and solutions for game design [9], providing high-end graphics, GPU utilization for parallelizable computation, and simple test scene design.

The communication between ROS ($rosbridge - suite$ [10]) and Unity3D ($rosbridgelib$ [11]) makes it an invaluable asset for robotics development. Figure 1 shows an illustration of the abstraction layers in data transfer between the Unity3D engine features and ROS framework modules.

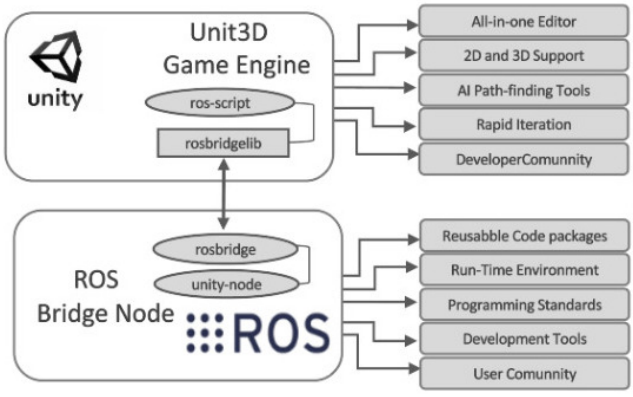


Fig. 1. ROS and Unity 3D communication diagram based on [9]. This diagram illustrates the abstraction layers in data transfer from the unity3D engine features to ROS framework modules and vice versa.

The environment proposed[1] in Unity 3D features a physics engine with fluid motion, which allows for the simulation of realistic water movement and oil spills in real-time, an example of the high level of visual fidelity achieved can be viewed in Figures 2 and 3. The textures and lighting provide a high level of visual fidelity, creating a stunningly realistic environment. The combination of these elements

[1]https://github.com/kleberandrade/flying-u2-2021

created an immersive experience that allowed multiple tests to be conducted for testing the oil-spill monitoring in this simulated environment.

As can be observed in Figures 2 and 3, the oil slick spreads out from its source in all directions, forming a dark stain on the surface of the water. The oil is a deep, iridescent black, reflecting the light of the sun on the surface of the water as it moves outward. The slick gradually covers more and more of the water, creating a stark contrast between the clear water and the oil.

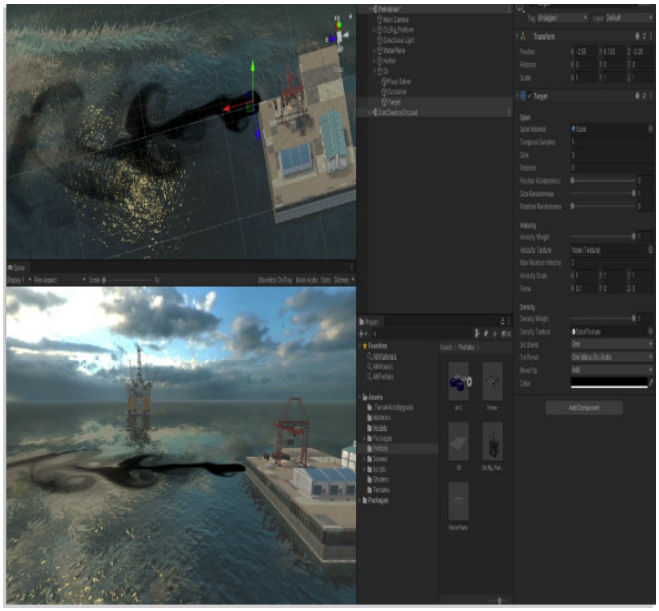


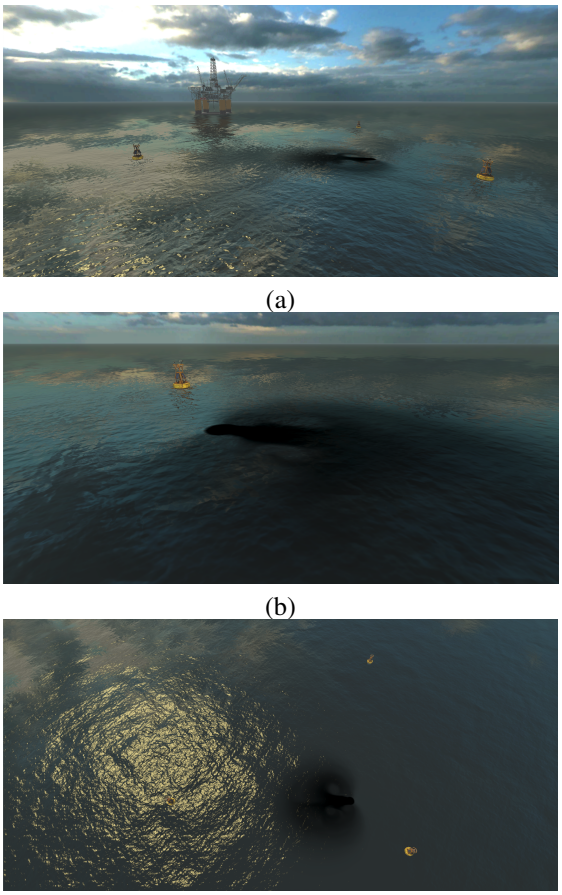
Fig. 2. A simulated oil spill from a storage tank. An oil offshore platform, a base, and an oil slick spread out from it in all directions. The slick is a dark, murky color and spreads across the surface of the water.

Figure 2 shows a simulated oil spill from a storage tank. This type of spill is caused by a breach in the tank wall, which can be caused by corrosion, mechanical damage, or other factors. Figure 3 shows a simulated oil spill that can be from a pipeline or a wellhead. The pipeline spill is caused by a rupture in the pipeline, which can be caused by corrosion, mechanical damage, or other factors. The wellhead spill is caused by a failure in the wellhead, which can be caused by corrosion, mechanical damage, or other factors.

The oil flows into the seawater considering a realistic fluid dynamics model, which was also applied to seawater and wind movements. All of these types of spills can be simulated in a controlled environment allowing the development of strategies to prevent, monitor and deal with them.

III. Detecting and Monitoring Oil Spills Using Multi-UAVs

Oil spills can have devastating environmental consequences, including the death of marine animals and the contamination of water and soil. To minimize the damage caused by oil and derivative spills, companies must invest in prevention, preparation, and rapid response. This includes having an Individual Emergency Plan approved by



(a)

(b)

(c)

Fig. 3. Different perspectives of oil spills: a simulated oil spill that can be from a pipeline or a wellhead.

the environmental agency of each country, such as Brazil's Resolution CONAMA No. 398/2008 (Conselho Nacional do Meio Ambiente - National Council for the Environment) and the United States of America's EPA (United States Environmental Protection Agency). As well as having trained and well-equipped teams with appropriate equipment and materials. The emergency plan should include procedures for the alert system, monitoring, and operational procedures for oil spill response, such as visual monitoring and sample collection. Additionally, detailed mapping for monitoring should be done to characterize the base condition and identify changes as well as incidents.

A. Multi-UAV Collaborative System

In this context, the Flying U2 Team proposed a Multi-UAV collaborative and autonomous system for the detection, inspection, and monitoring of oil spills, adopting this realistic simulated scenario.

1) Framework Architecture : The Multi-UAV collaborative system works in a leader-follower, heterogeneous architecture as explained in Figure 4.

The leader system is a customized F450 UAV, chosen for its larger processing and storage capacity compared to the other UAVs in the team, and it includes a neural

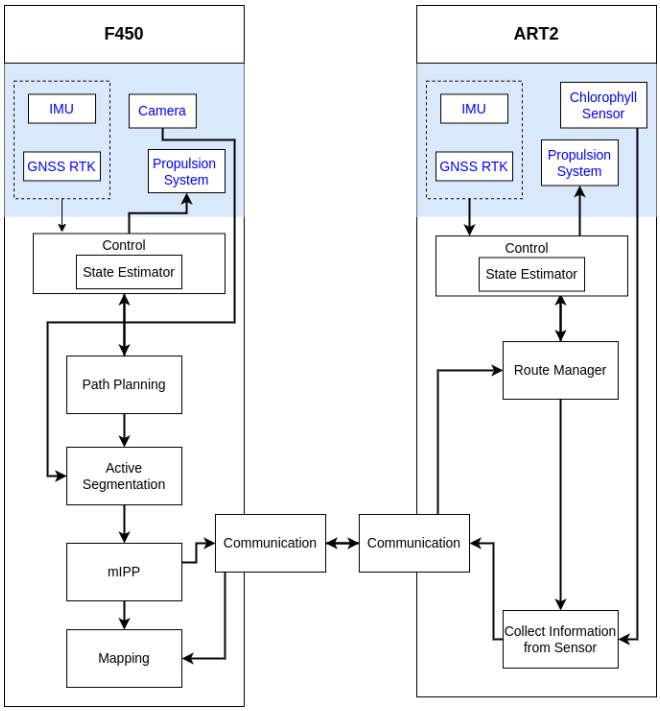


Fig. 4. Framework Architecture

network-based vision system for real-time target detection. The follower system is composed of a set of UAVs named ART2, designed to land on water, immerse an optical sensor, and analyze samples in situ. This system is capable of performing real-time target detection, coordinating path planning, and assigning tasks online for the team of heterogeneous UAVs. In addition, the system includes hardware integration, communication, path planning, decision-making, control, mapping, active segmentation, and a Multi-UAV Informative Path Planning (MIPP).

Figure 5 shows the F450 and ART2 used as the leader and the follower drones, respectively. From the simulated environment, we adopted the Software in the Loop strategy, which benefited the development of the system. This strategy allows the development of the system without the need for physical hardware.

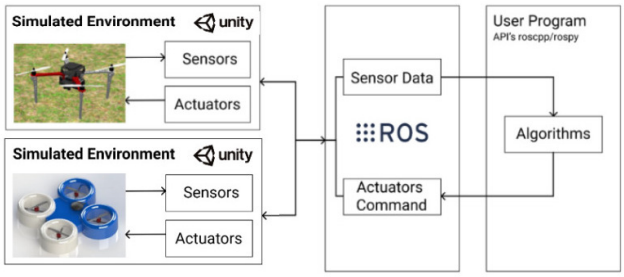


Fig. 5. Software in the Loop Diagram - UAVs F450 and ART2

2) Active Segmentation System : To identify the oil spill problem, we first analyzed real images from an open-source dataset to gain an understanding of the issue. Semantic segmentation was deemed the best way to classify an oil spill because it allows for a detailed analysis of the size and coverage of the spill. This type of segmentation uses deep learning algorithms to identify and classify objects in an image, as the identification requires both speed and accuracy. The UNet fully convolutional neural network was chosen for the image segmentation and classification task due to its suitability for the training data, its simple training procedure, and its fast inference time [12].

The UNet was structured with MobileNet as the encoder [13], which was pre-trained on ImageNet. The CNN utilized categorical cross-entropy as the loss function and Adam optimizer for optimization. To enhance the dataset, we improved the model equations for the spill expansion in the simulated environment to compose various scenarios for training the network. And to expand the dataset used for training, data augmentation techniques such as rotation, translation on the X and Y axes, scaling, gamma contrast, flipping, and median blur were utilized, thus enhancing the model's robustness to deformations.

The metrics were calculated using the Intersection over Union (IoU) method, obtaining an IoU of 94% from a total of 241 images. From the training, testing, and validation, the active segmentation algorithm can be embedded in the UAV to identify regions of interest, providing information to MIPP.

IV. RESULTS

The functionality of the simulated environment with the collaborative multi-UAV architecture was tested using the structure of the diagram presented in Figure 6. The F450 UAV uses path planning to minimize time and distance traveled while maximizing the amount of data collected in real time. During coverage of the area, it collects images (Fig. 7 a) and data produced by the State Estimator, and then send them to the Active Segmentation system Section III-A.2). The Active Segmentation system identifies the target (Fig. 7 b) and sends the center and extreme points of the coordinates to the MIPP. These points are grouped and evaluated to ensure that it is a contaminated area. The coordinates of the point of interest are sent to the Online Route Planning algorithm, which calculates the best ART2 route.

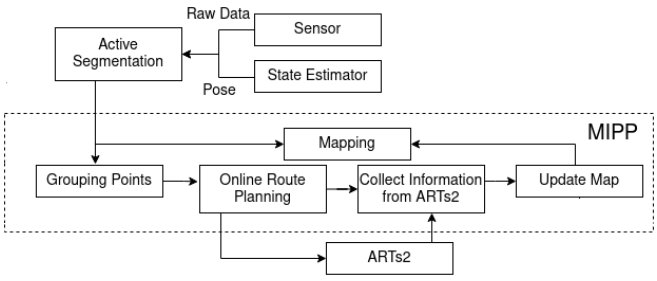
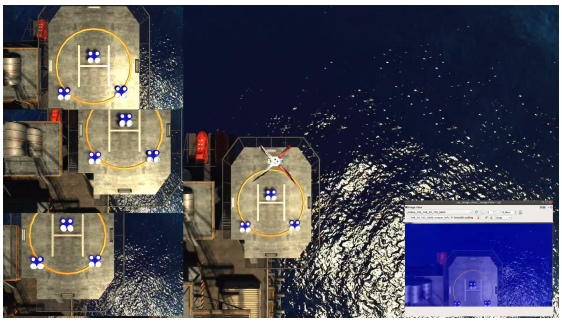
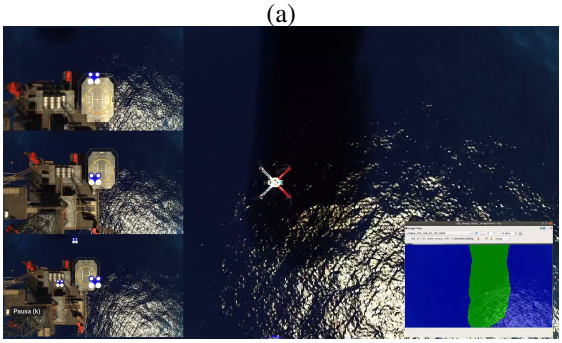


Fig. 6. Multi-UAV Informative Path Planning Architecture

Each ART2 executes the route and collects samples in each specified point (Fig. 8). Thus, the leader quickly and



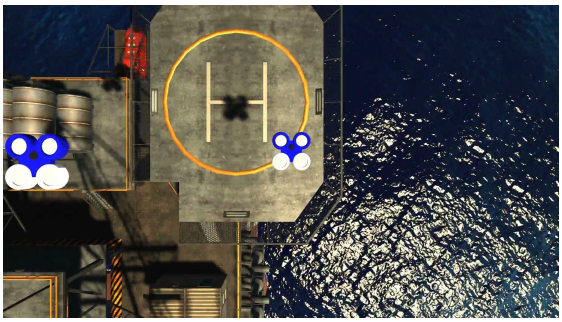
(a)



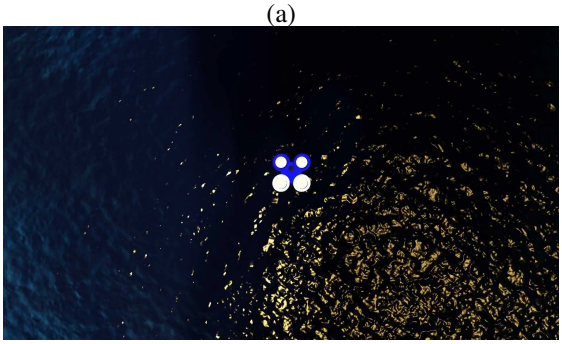
(b)

Fig. 7. Multi-UAV Platform - F450

accurately assesses the environment and sends reports that ensure the safety and success of the mission.



(a)



(b)

Fig. 8. Multi-UAV Platform - ART2s

V. CONCLUSIONS

In conclusion, the realistic simulated environment proposed in Unity 3D provides a powerful tool for testing and evaluating strategies and algorithms for the detection, inspection, and monitoring of oil spills. The combination of the physics engine, textures, and lighting creates an immersive experience that allows for multiple tests to be conducted in a safe and controlled environment. From the ROS Framework was possible to use the multi-UAV collaborative system enabling the reconfiguration and expansion of specific objectives, in addition to minimizing the task completion time by executing different processes in parallel. The simulated environment provides a safe and controlled environment to test and evaluate different strategies and algorithms, allowing researchers to gain valuable insights into the problem at hand, thus accelerating the development of the research and reducing the cost of the experiments.

ACKNOWLEDGMENT

Thank you to the Flying U2 team for the work and experience shared together. The authors would like to acknowledge the contribution of team member Kleber Andrade to the environment design. The authors would like to thank financial support provided by the Brazilian National Science and Technology Institute for Autonomous Cooperative Systems (INSAC), São Paulo Research Foundation (FAPESP) (grants #2014/50851-0 and #2016/21220-7), National Council for Scientific and Technological Development (CNPq) (grant #465755/2014-3 and #421131/2018-7) and Coordination for the Improvement of Higher Education Personnel (CAPES).

REFERENCES

[1] A. Chaudhary, R. Mishra, B. Kalyan, and M. Chitre, "Development of an underwater simulator using unity3d and robot operating system," in *OCEANS 2021: San Diego – Porto*, 2021, pp. 1–7.

[2] I. Lončar, J. Obradović, N. Kraševac, L. Mandić, I. Kvasić, F. Ferreira, V. Slošić, Na, and N. Mišković, "Marus - a marine robotics simulator," in *OCEANS 2022, Hampton Roads*, 2022, pp. 1–7.

[3] P. Foehn, E. Kaufmann, A. Romero, R. Penicka, S. Sun, L. Bauersfeld, T. Laengle, G. Cioffi, Y. Song, A. Loquercio, and D. Scaramuzza, "Agilicious: Open-source and open-hardware agile quadrotor for vision-based flight," *Science Robotics*, vol. 7, no. 67, p. eabl6259, 2022.

[4] Y. Song, S. Naji, E. Kaufmann, A. Loquercio, and D. Scaramuzza, "Flightmare: A flexible quadrotor simulator," in *Conference on Robot Learning*. PMLR, 2021, pp. 1147–1157.

[5] A. Brunel, A. Bourki, O. Strauss, and C. Demonceaux, "Flybo: A unified benchmark environment for autonomous flying robots," in *2021 International Conference on 3D Vision (3DV)*, 2021, pp. 1420–1431.

[6] A. Merci, C. Anthierens, N. Thirion-Moreau, and Y. Le Page, "A simulator of underwater glider missions for path planning," *Ocean Engineering*, vol. 269, p. 113514, 2023.

[7] J. K. Haas, "A history of the Unity game engine," 2014.

[8] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, A. Y. Ng, *et al.*, "ROS: an open-source Robot Operating System," in *ICRA workshop on open source software*, vol. 3, no. 3.2. Kobe, Japan, 2009, p. 5.

[9] A. Hussein, F. García, and C. Olaverri-Monreal, "ROS and Unity based framework for intelligent vehicles control and simulation," in *2018 IEEE International Conference on Vehicular Electronics and Safety (ICVES)*, 2018, pp. 1–6.

[10] J. Mace, "Rosbridge suite," 2022. [Online]. Available: http://wiki.ros.org/rosbridgesuite

[11] M. Ciarlo, "Rosbridge library," 2015. [Online]. Available: http://wiki.ros.org/rosbridgelibrary

[12] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, 2015, pp. 1234–241.

[13] 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," 2017.