Tianjin Daxue Xuebao (Ziran Kexue yu Gongcheng Jishu Ban)/
Journal of Tianjin University Science and Technology
ISSN (Online):0493-2137
E-Publication: Online Open Access
Vol: 56 Issue: 05: 2023
DOI10.17605/OSF.IO/FSUQZ

SAMPLING-BASED UNDERWATER AUTONOMOUS VEHICLE TRAJECTORY PLANNING THROUGH REALISTIC SIMULATORS

ALFREDO J. BAYUELO

Currently pursuing Ph.D. degree program in Systems engineering in National University of Colombia.
E-mail: ajbayuelos@unal.edu.co

LEONARDO BOBADILLA

Florida International University, Miami, FL 33199, United States.

FERNANDO NINO

Universidad Nacional de Colombia, Bogotá, Colombia.

Abstract

The fourth Industrial revolution has put robots at a starring position in every endeavor. For its correct operation, testing is a crucial phase before deployment. Running tests on robots is more challenging than testing on software, particularly when accounting for replicability, environment matching, recovery strategies, etc. In the case of robots for marine environments, running tests is even more challenging because of the features of the former such as high dynamism, poor visibility, extreme conditions, etc. New technologies such as AUVs (Aquatic Unmanned Vehicles) have gained attention since they are reliable, affordable, and highly maneuverable. Even for these platforms, sufficient trials should be conducted and several outcomes studied before deployment, to make it secure and more likely to accomplish the objectives. We propose a strategy that provides the capabilities of running trials in simulated environments. Once the trials are conducted and a plan is determined, the real-world deployment of the robot will be more reliable and more likely to achieve its goals. Our strategy is based on the RRT algorithm (Rapidly exploring Random Trees) for an AUV. We present a variation that considers kinematics, dynamics, and uncertainty of the vehicles' movement, allowing safe experimentation for the robot, environment and researchers.

Index Terms: Aquatic Environment, Motion Planning, Path Planning, Realistic Simulation, Testing, Uncertainty, Underwater Vehicles

1. INTRODUCTION

The fourth Industrial revolution has put robotic systems (robots) at a starring position in every industrial, academic, and social endeavor. These systems allow the new computing power and communications strengths to interact with the physical world. In software development, testing is a crucial phase before every deployment. Further, test-deploy activities are run automatically in a continuous integrated manner. It is a different scenario for robotic systems in which running tests accounting for replicability, environment matching, recovery strategies, etc., is more challenging. Despite these difficulties, it is still imperative to conduct sufficient trials and study several outcomes before successfully deploying a robotic system. The positive testing results increase the probability of the robotic system being safely deployed and accomplishing the intended objectives. Nevertheless, running field trials for robots can be expensive in both computational and financial sense: On the one hand, mathematically modeling a complex vehicle to plan the set of actions beforehand is a challenging task; also, reproducing a particular set of conditions or the state of the deployment environment is often impractical or impossible.

Tianjin Daxue Xuebao (Ziran Kexue yu Gongcheng Jishu Ban)/
Journal of Tianjin University Science and Technology
ISSN (Online):0493-2137
E-Publication: Online Open Access
Vol: 56 Issue: 05: 2023
DOI10.17605/OSF.IO/FSUQZ

On the other hand, failed trials can be catastrophic and stop the research projects by losing the assets and the budget to replace them, not to mention that the environment itself can also result in destruction due to the test.

In the case of marine environments, running robotic systems tests is even more challenging because of its features such as high dynamism, poor visibility, extreme temperatures, extreme pressures, and wildlife (flora and fauna) that can affect the equipment. For these reasons, research in these environments has been led by robotic systems, traditionally large equipment mounted on dedicated ships. Even though new technology such as AUVs (Aquatic Unmanned Vehicles) has lowered the budget and improved maneuverability, other requirements to conduct field experiments in this area are still difficult and sometimes unaffordable for most laboratories, especially in developing countries. A consequence of these difficulties can be evidenced in the fact that more than 80% of the oceans, which in turn comprehends around 70% of the earth's surface, are still unmapped, unexplored, and unobserved [1]. This finally translates into a poor understanding of the processes that occur in these environments, which of course, have direct and indirect repercussions on human activities: health-related, economic, and environmental.

The strategy and methods presented in this work provide the ability to do enough trials in simulated environments. Once the simulation trials are conducted and the plan for the robotic system is determined and tested, the real-world deployment would be more secure and more likely to achieve its goals.

1.1 Related Work

Computers have dramatically increased their capabilities even faster than expected by Moore's Laws [2]. By 2021 the components of computational processors are as small as 2 nm [3], which translates into smaller and faster computers, also more efficient in energy consumption. These technologies have opened the gates to a new generation of autonomous robots that use relatively small platforms to conduct exploration, exhibiting endurances of days and weeks. Also, this increase in computing power allows for on-board advanced calculation despite the size of the embedded computers.

One domain that has greatly benefited from these technologies is computer graphics. The higher computational strength allows for making calculations and generating realistic environments progressively as a character moves around in a digital world. Furthermore, artificial intelligence techniques have been used to make these generated worlds look photo-realistic [4]. These kinds of Simulations have also been used to train robotic systems for tasks in different domains such as self-driving [5], [6], warehousing [7], and house-holding [8], among others. The same technology is also used to train humans in immersive simulations for safe learning industrial equipment manipulation [9]. Following these ideas, in this work, a simulated environment is used to explore the surroundings, as a real underwater vehicle would, and find a feasible path in the form of a policy. The dynamics for the vehicle are emulated using the operating system of the actual robots, and the outcome is accounted for by a physics simulator. This setting is an instance of the strategy "Software-in-the-loop." The simulation can also be connected to the real

Tianjin Daxue Xuebao (Ziran Kexue yu Gongcheng Jishu Ban)/
Journal of Tianjin University Science and Technology
ISSN (Online):0493-2137
E-Publication: Online Open Access
Vol: 56 Issue: 05: 2023
DOI10.17605/OSF.IO/FSUQZ

robots sensors and actuators to configure “Hardware-in-the-loop” [10]; this is very convenient since the policy obtained can be tested on the real robots hardware before being transferred to the production robot; this last step is known as “sim-to-real” [11], [12], [13], i.e., a transference of the data learned during the simulation process into the real robot. There is an extra step that can be considered in between hardware-in-the-loop and sim-to-real, which is Virtual Reality for Robots (VRR) [14], in which the sensors and actuators of the actual robot are fooled into believing it is moving in the real world environment, but the data spoofed to the sensors is still being generated by the simulator. To accomplish this, it is kept in the simulation a “ghost” track of the robot and it is calculated what the sensor would “see” at each particular position and orientation in the simulated world. The notation presented in the next section is based on [14] with some slight modifications.

The ultimate objective is to obtain a path that the vehicle can follow. Selecting the set of actions needed to accomplish a translation from one point in the world to another is called Motion Planning. The classical approach divides the process into four steps [15]: first, a feasible path is found; second, the path is smoothed to account for constraints in the robot movement (for example, an autonomous robotic vehicle cannot do sharp turns, that is also the case for tripulated vehicles); third, the path is revisited to account for dynamic constraints, for example, to prevent the vehicle from immediately trying to stop while moving; and finally, a controller is designed to keep the robot following the path. Each one of these steps has several challenges for research. Some strategies, such as Planning Under Differential Constraints, aim to address several of these steps in a single consideration [16],[17],[18]. In this sense, we present a strategy that accounts for feasible paths while considering kinematics and dynamics.

1.2 Problem Formulation

Given an environment. Let represent the set of locations that are inaccessible for a robot. The robot's position and pose are described by a configuration $q \in \mathcal{C}, q = (x_1, x_2, x_3, \mathbf{h})$, where is the set of all possible configurations, x_1, x_2, x_3 are coordinates and is a unit quaternion representing the orientation in space. Let represent the robot transformed to configuration, then comprises the obstacle region and is the free region, i.e., the configurations where the robot does not collide into any obstacle. The robot has a set of sensors that are modeled as, such that is an observation read by sensor obtained from the mapping. The idea behind VRR is achieved through mocking the sensors using displays, a display output induces the robot to an observation, this output can be modeled as a mapping that could also include the configuration space: such that either or holds. For the display to be able to induce the intended at a particular configuration, it must keep an internal track of the configuration called , so the display would calculate , such that or, if possible, $d = r(q, q')$ when both the simulation and the real world configuration are used, such that. This is the philosophy behind hardware in the loop, in which displays cover some of the sensors, while the rest are providing actual physical input. Finally, to obtain the observation we have:

Tianjin Daxue Xuebao (Ziran Kexue yu Gongcheng Jishu Ban)/
Journal of Tianjin University Science and Technology
ISSN (Online):0493-2137
E-Publication: Online Open Access
Vol: 56 Issue: 05: 2023
DOI10.17605/OSF.IO/FSUQZ

Problem I: Getting feasible trajectories from BlackBox simulator

Given a starting configuration and a set of goal configurations , determine if there exists a set of configurations such that each configuration can be reached from , also .

The set of configurations is called a trajectory, and when it is called a feasible trajectory.

Problem II: Selecting safer trajectories to avoid collisions.

Given a set of feasible trajectories, determine which is the least hazardous trajectory that can be followed and accomplished.

2. MATERIALS AND METHODS

It is proposed here a strategy based on the well-known RRT algorithm (Rapidly exploring Random Trees) [1]. A variation of this algorithm was considered to take into account the kinematics, dynamics, and at some degree uncertainty. The strategy presented also allows for complex landscapes with complex obstacle regions, which is especially challenging for non-sampled solutions. The one presented is a BlackBox strategy that lets a physics simulator take the heavy lifting of calculating the dynamics of the vehicle, perform the actions and deal with the uncertainty associated as described in [2]. The robot considered in this document is the torpedo shaped Light Autonomous Underwater Vehicle (LAUV) model for gazebo [3]. At each time, the vehicle has a position in the space (x,y,z) and orientation (roll, pitch, yaw) that can be determined in the simulation and that combined define the state of the vehicle uniquely, this last corresponds to the *vehicle's configuration*.

Algorithm 1: SIMULATEDRRT(S, U, T, n, q_S, Q_G)

Input: S,U,T,n, q_S, Q_G – a simulator, a set of actions, the duration of the action, the number of steps for the algorithm, the starting position, the goal region

Output: R – a graph (V,E) that might contain a feasible path to the Goal region.

$R.V \leftarrow q_S$ $\quad$ $R.E \leftarrow \emptyset$
for *step* ***in*** n **do**
  if *random()* $<$ *10* **then**
    $q_r \leftarrow random\_point(\mathrm{Q}_G)$;
  else
    $q_r \leftarrow$random_point(S);
  $q_{closest} \leftarrow$R.closestNode(q_r);
  for u ***in*** U **do**
    $q \leftarrow S.execute(u, q_{closest})$;
    if q **then**
      $R.V$.add(q);
      $R.E$.add($q_{closest}$,q);
return R

Algorithm 1: Overall algorithm

The RRT variation used is shown in Algorithm 1 and described next: Starting from the desired initial configuration, the root of the RRT tree starts growing toward the branches. The algorithm selects a random position in the environment and finds the closest node in the current tree to that position. A set of actions are performed from the mentioned closest node for a period of time, and the reached configurations are added to the tree saving:

Tianjin Daxue Xuebao (Ziran Kexue yu Gongcheng Jishu Ban)/
Journal of Tianjin University Science and Technology
ISSN (Online):0493-2137
E-Publication: Online Open Access
Vol: 56 Issue: 05: 2023
DOI10.17605/OSF.IO/FSUQZ

the parent node for each one, the action performed at the parent, and a set of way-points visited during the execution of the action. The algorithm continues selecting random positions in the environment; a portion of the time (10%) the goal position is selected to guide the search by adding a bias toward the goal; every configuration that results in a collision is disregarded. The algorithm stops when a node is added close enough to the goal configuration, or the maximum defined number of iterations is reached. The RRT algorithm is probabilistically complete, which means that if a path exists and sufficient time and iterations are performed, then the algorithm would find the path.

The simulator used is Gazebo 9 [22]. This open-source software provides physics simulation and rendering capabilities. The sensors' data and the commands are transmitted using the Robotic Operating System (ROS). ROS is a set of libraries that allow both the simulation and deployment of robotic applications.

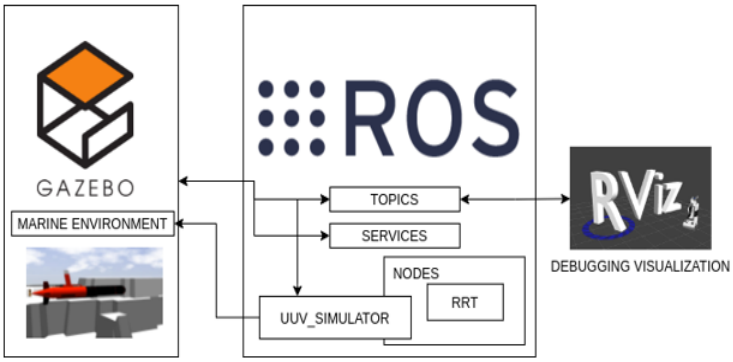

Figure 1: Overall Scheme including the technologies used and their relationship.

The algorithm described previously was implemented as a ROS Node, which is the unit execution block in the ROS environment as shown in Figure 1. The ROS Node is a script that can be written in C++ or Python; the RRT implementation described was coded in python. One of the advantages of this setting is that the simulator only knows the building blocks physics, that is: the physics of the propeller, the physics of the movement for the shape of the robot in space, the physics of the environment, and all this put together in an iterative simulation predicts realistically the outcome for the actions taken. This outcome is established without actually modeling the robot kinematics nor considering the particular dynamics of the robot; is that why this strategy is called BlackBox modeling: the simulator serves as a BlackBox and provides the outcomes.

3. RESULTS

Figure 2 shows an example result of the algorithm in action. Particularly, it shows all the outcomes after executing the set of available actions determined to reach a particular goal. One big advantage of simulated environments is replicability: based on a selected seed, the exact same simulation can be repeated several times. If a scenario is found where a modification needs to be performed to the plan or to the settings of the vehicle, it can be fully tested before configuring the actual robots. Another benefit of this property

Tianjin Daxue Xuebao (Ziran Kexue yu Gongcheng Jishu Ban)/
Journal of Tianjin University Science and Technology
ISSN (Online):0493-2137
E-Publication: Online Open Access
Vol: 56 Issue: 05: 2023
DOI10.17605/OSF.IO/FSUQZ

is exploited in this work: once a particular action is executed for the defined period, the vehicle can be set back to the initial conditions, including the position and velocities of the vehicle, to perform another action; this behavior would be impossible in real life.

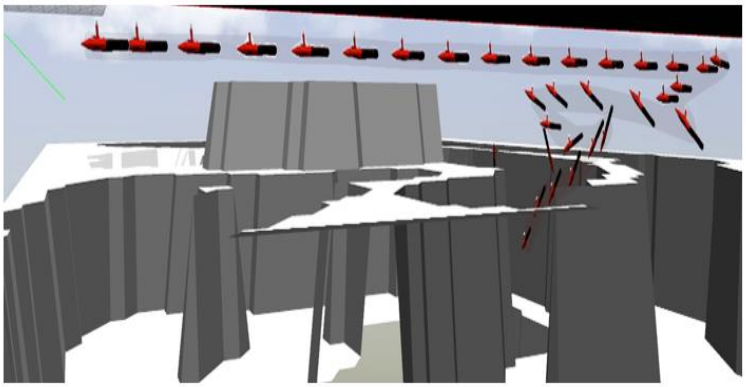

Figure 2: Partial plan executed, the trajectory is shown as the vehicle is executing the generated plan.

Figure 3 shows the tree generated in 400 epochs for the environment and target region from Figure 2. It was constructed by drawing a straight line from each parent to its corresponding node child. Nevertheless, the actual path followed from parent to child, corresponds to the natural movement of the vehicle after the action performed at the father node. It can be seen that the environment used for tests is non-trivial, rather very similar to natural underwater spaces in which caves and trenches can be found. Even though for this experimentation a single path was followed, the algorithm proposed and the tree shown in Figure 3 acts as a policy since multiple paths can be followed depending on the current position. This also can be exploited as a controller to replan whenever the intended path is deviated; if a node exists close to the actual position, the new path can be followed. If the deviation is such that no Node in the tree represents such a configuration, then the tree can be further expanded to account for the new location.

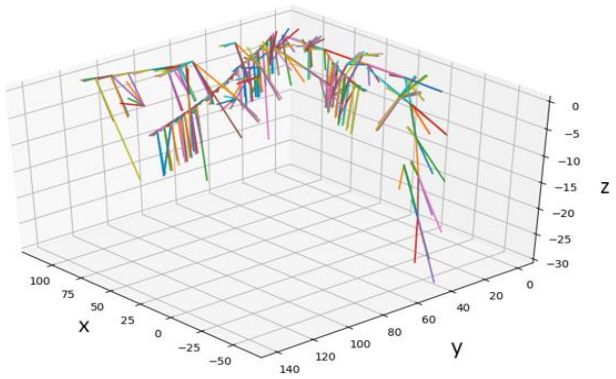

Figure 3: Node tree generated by the augmented RRT algorithm. Parent nodes are linked to their children.

Tianjin Daxue Xuebao (Ziran Kexue yu Gongcheng Jishu Ban)/
Journal of Tianjin University Science and Technology
ISSN (Online):0493-2137
E-Publication: Online Open Access
Vol: 56 Issue: 05: 2023
DOI10.17605/OSF.IO/FSUQZ

4. CONCLUSIONS AND FUTURE WORK

The strategy presented in this work also permits selecting safer paths since, in the simulation, the information of the world is complete, i.e., the distance to every obstacle can be calculated at all times. With this information, hazardous maneuvers that take the vehicle too close to an obstacle can be avoided. This is also the case for compliance with regulations: if a mission needs to be executed in a region where some particular restrictions apply, the feasible path can be obtained in such a way that aims to prevent accidental noncompliance.

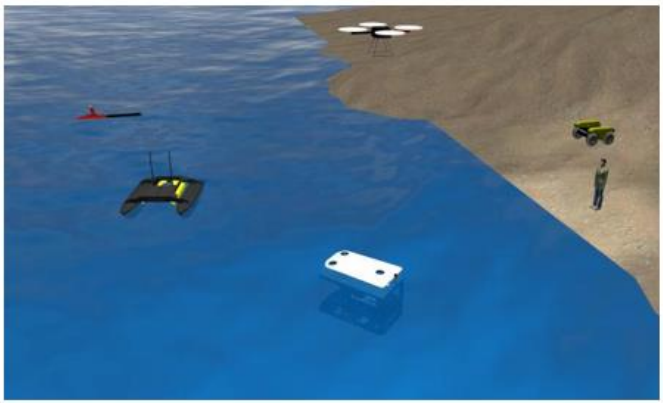

Figure 4: Simulation for multiple parties including human operators.

This safety can also be extended to other parties in the environment. As shown in Figure 4, multiple robots can be simulated, the communication among them can also be simulated, and safe zones can be defined to maneuver while preserving safety for human operators and other vehicles.

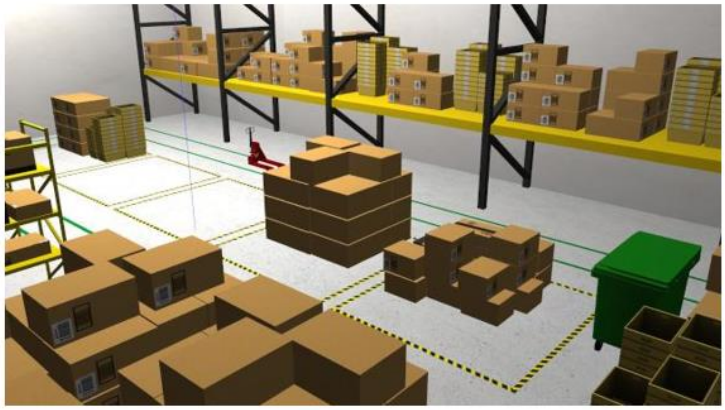

Figure 5: Warehouse setting simulation.

One common practice when doing motion planning is the discretization of the action space. This makes tracktable the problem since it is reduced to an instance in which a finite number of possible actions is considered. As mentioned in [4], this has the

Tianjin Daxue Xuebao (Ziran Kexue yu Gongcheng Jishu Ban)/
Journal of Tianjin University Science and Technology
ISSN (Online):0493-2137
E-Publication: Online Open Access
Vol: 56 Issue: 05: 2023
DOI10.17605/OSF.IO/FSUQZ

disadvantage of missing an action that could be necessary to accomplish the goal. In this work, the set of actions consist of only 12 actions combining pitch, roll, yaw and forward/reverse. Sampling from the action space as proposed in [23] would be a very interesting improvement for this work.

Finally, these simulated environments also apply to ground vehicles in the presence of delicate goods, as is the case for warehouses shown in Figure 5. Real-world disasters have been recorded in surveillance cameras showing how a minimal collision with the obstacle region can destroy the complete warehouse; safer paths are critical to prevent that kind of outcome. Also, the obstacle region can be re-defined based on multiple tests and routes on a simulated warehouse. In such a manner that design is also possible through simulations. Figures were constructed following the same strategy, for which the operating system is common for both the simulations and the real robots; this allows for completing the four steps: software-in-the-loop, hardware-in-the-loop, virtual reality for robots, and sim-to-real. The evident further step of this work is to transfer the obtained paths to real-world robots which can be gradually achieved using the mentioned VRR.

References

1. S. Nag and others, “How much of the ocean have we explored?,” WorldAtlas, 2019.
2. G. E. Moore and others, “Cramming more components onto integrated circuits.” McGraw-Hill New York, 1965.
3. B. H. McCarthy, “IBM Unveils World’s First 2 Nanometer Chip Technology, Opening a New Frontier for Semiconductors.” May 2021. [Online]. Available: https://newsroom.ibm.com/2021-05-06-IBM-Unveils-Worlds-First-2-Nanometer-Chip-Technology,-Opening-a-New-Frontier-for-Semiconductors
4. S. R. Richter, H. A. AlHaija, and V. Koltun, “Enhancing Photorealism Enhancement,” ArXiv Prepr. ArXiv210504619, 2021.
5. A. Gambi, M. Mueller, and G. Fraser, “Automatically testing self-driving cars with search-based procedural content generation,” in Proceedings of the 28th ACM SIGSOFT International Symposium on Software Testing and Analysis, 2019, pp. 318–328.
6. Musk, “Tesla AI Day.” [Online]. Available: https://www.youtube.com/watch?v=j0z4FweCy4M
7. A. A. Vieira, L. M. Dias, G. A. Pereira, J. A. Oliveira, M. D. S. Carvalho, and P. Martins, “Simulation model generation for warehouse management: Case study to test different storage strategies,” Int. J. Simul. Process Model., vol. 13, no. 4, pp. 324–336, 2018.
8. J. Tremblay, T. To, B. Sundaralingam, Y. Xiang, D. Fox, and S. Birchfield, “Deep Object Pose Estimation for Semantic Robotic Grasping of Household Objects.” arXiv, Sep. 27, 2018. doi: 10.48550/arXiv.1809.10790.
9. Q. Chen et al., “Immersive Virtual Reality Training of Bioreactor Operations,” in 2020 IEEE International Conference on Teaching, Assessment, and Learning for Engineering (TALE), 2020, pp. 873–878.
10. A. Ledin, “Hardware-in-the-loop simulation,” Embed. Syst. Program., vol. 12, pp. 42–62, 1999.
11. W. Zhao, J. P. Queralta, and T. Westerlund, “Sim-to-real transfer in deep reinforcement learning for robotics: a survey,” in 2020 IEEE Symposium Series on Computational Intelligence (SSCI), 2020, pp. 737–744.
12. R. Jeong et al., “Modelling generalized forces with reinforcement learning for sim-to-real transfer,” ArXiv Prepr. ArXiv191009471, 2019.

Tianjin Daxue Xuebao (Ziran Kexue yu Gongcheng Jishu Ban)/
Journal of Tianjin University Science and Technology
ISSN (Online):0493-2137
E-Publication: Online Open Access
Vol: 56 Issue: 05: 2023
DOI10.17605/OSF.IO/FSUQZ

13. A. A. Rusu, M. Večerík, T. Rothörl, N. Heess, R. Pascanu, and R. Hadsell, “Sim-to-real robot learning from pixels with progressive nets,” in Conference on Robot Learning, 2017, pp. 262–270.
14. M. Suomalainen, A. Q. Nilles, and S. M. LaValle, “Virtual reality for robots,” in 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2020, pp. 11458–11465.
15. S. M. LaValle, “Motion planning: Wild frontiers,” IEEE Robot. Autom. Mag., vol. 18, no. 2, pp. 108–118, 2011.
16. E. Schmerling, L. Janson, and M. Pavone, “Optimal sampling-based motion planning under differential constraints: the driftless case,” in 2015 IEEE International Conference on Robotics and Automation (ICRA), 2015, pp. 2368–2375.
17. L. Jaillet and J. M. Porta, “Path planning under kinematic constraints by rapidly exploring manifolds,” IEEE Trans. Robot., vol. 29, no. 1, pp. 105–117, 2012.
18. S. R. Lindemann and S. M. LaValle, “Multiresolution approach for motion planning under differential constraints,” in Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006., 2006, pp. 139–144.
19. M. LaValle, “Rapidly-exploring random trees: A new tool for path planning,” 1998.
20. M. LaValle, Planning Algorithms. Cambridge, U.K.: Cambridge University Press, 2006.
21. A. Sousa et al., “LAUV: The Man-Portable Autonomous Underwater Vehicle,” IFAC Proc. Vol., vol. 45, no. 5, pp. 268–274, Jan. 2012, doi: 10.3182/20120410-3-PT-4028.00045.
22. N. Koenig and A. Howard, “Design and use paradigms for gazebo, an open-source multi-robot simulator,” in 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No. 04CH37566), 2004, vol. 3, pp. 2149–2154.
23. Bayuelo, A., Alam, T., Bobadilla, L., Niño, L. F., & Smith, R. N. (2019, August). “Computing feedback plans from dynamical system composition”. In 2019 IEEE 15th International Conference on Automation Science and Engineering (CASE) (pp. 1175-1180). IEEE.