International Journal of Engineering & Technology, 7 (2.33) (2018) 960-967

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research paper

Investigating feasible tool for swarm robotic based oil skimming application

Padma Priya R [1] *, Agarwal Ruchi Sanjay [2], Yesho Vardhan Gupta [2], D. Rekha [1]

[1] Assistant Professor (senior), School of Computer Science and Engineering, Vellore Institute of Technology, India
[2] Student, School of Computer Science and Engineering, Vellore Institute of Technology, India
**Corresponding author E-mail: padmapriya.r@vit.ac.in*

Abstract

Swarm Robots, a multi-robot system (inspired from insect- groups), work in a decentralized network in a distributed manner [1]. These group of robots coordinate their activities without the need of a universal leader. Instead, they use local rules to track the behavior of the whole group and to communicate information amongst each other. Proper implementation of the swarm robotics system includes executing tasks such as aggregation, assembling, path planning, and pattern formation [2]. Various algorithms may be implemented to realize these tasks and techniques the robots carry out. These algorithms need to be exhaustively tested in terms of robustness, safety and efficiency. Testing of these systems in real-time environments is a matter of great risk to both the resources as well as life. Hence, simulators play a crucial role in the research of swarm robots, more so in applications which require close interaction with the humans [3]. It is equally important to have efficient simulators as it is to have algorithms pertaining to swarm robotics. This study presents a meticulous overview of popular robotic simulation software, some of which could also be used for interfacing in the real robotic environment. A comparison with reference to the oil skimming from oceans using swarm robots as mentioned in the paper is made between the most feasible simulation environments. This is followed by an illustration of the various tasks to be performed by individual robots or group of them.

Keywords: *Distributed System; Oil Skimming; Swarm Robots; Simulation Software.*

1. Introduction

Increasing offshore industrial activities and transportation of oil across continents have raised threats drastically in marine environments. These activities contribute to discharges that endanger marine lives. Treating and restoration of such polluted water bodies, amidst the presence of dynamic environmental parameters has always been an onerous challenge both for government and researchers. With growing research, the knowledge of swarm robotics is being applied for environmental monitoring in marine environments [4] and can be used for oil skimming and removal. Increasing robotic applications in critical areas such as the medical field where there is close interaction between humans and robots, and in restoration of marine environments from oil-spills - has created a need for accurate and robust simulation tools. Due to the varying conditions of the surrounding in most applications, it is vital to test the safety, efficiency and robustness of the new robot models and interaction algorithms in a realistic and reliable simulation of the actual environment. Software implementations are easy to produce, cheaper and does not endanger any lives. Simulations pertaining to this field provide sufficient proof of the correctness of the designs of such robots. Specific environmental conditions can be simulated to understand the working of designed models. In choosing one simulator, it should provide easy and swift code-portability to a real platform.
A budding number of software-simulation tools have been designed in view of these features and requirements. Along with the development of such commercial and open-source software simulation tools, a proper, extensive and clear documentation of them is very necessary to help researchers and students in the field of robotics. With an evident need for an extensive survey on simulators, this work aims to help readers decide the right simulator for their project and help beginners gain insight on this subject. The structure of the paper is as follows. Section II illustrates the problem which motivated us to think in this domain. The detailed discussion on various robotic simulators has been done in section III. Section IV covers the application of swarm robotics in the field of oil skimming. Concluding remarks have been provided in section V. Section VI throws light on the future scope of this project. Lastly, comparisons between the chosen simulators have been edged out in section VII.

2. Motivating problem and solution

On January 28, 2017, two cargo ships collided off the Ennore coast in Chennai [14] causing oil to spill into the sea. Fig 1 depicts the shores in Chennai polluted with oil. The oil spill spread for 34 KM and more than a thousand manual workers volunteered to remove the sludge (shown in fig. 2). The articles [15], [16] noted down the effects of this spill on marine life which include oxygen shortage, endangered turtles and fishes. These effects had direct impact on the lives of fishermen and people living in the villages by the Tamil Nadu coast. The articles also pointed out the incapability of the government to react faster to the accident. A need for a system to improve the oil skimming process in terms of speed, area coverage and robustness is evident. Later in year 2017, Two hundred and seventy students of IIT Madras come together under the institute's Centre for Innovation (CFI), with 45 robots, cleaning a 750-sqft area with the help of rotating scrubbing pads [18]. A couple of scrubbing pads, attached to the robots directed the dust towards the central suction mechanism. A filter then collected the dust in a vacuum tunnel. The robots were controlled by Bluetooth through a mobile-based application. Collision Avoidance techniques were implemented using a proximity ultrasound. This multi-robot system sparked the idea of using swarm robotic system for oil removal from oceans.

Fig. 1: The Floating Oil along the Shore of North Chennai.

Fig. 2: A Total of 208 Tonnes of Oil Sludge Was Collected Manually.

3. Robot simulation environments

In this section, an overview of available open source robotic simulation environments is given. As mentioned earlier, an attempt was made to discover simulators which can simulate water and other properties. The keywords we used in search engines to unearth these simulators were, robotic simulation environments, aquatic robots, multi-robot systems and wireless sensor networks. After investigating about the various simulators, the simulators which seemed to be the most favourable were compared to identify and classify them, based on the project requirements. All paragraphs must be justified alignment. With justified alignment, both sides of the paragraph are straight.

3.1. Player/stage

Player/stage is an extensively used simulator available as a free software package for robot and sensor research. As the name suggests, it includes a player network server and a stage which serves as a robot simulator platform. The client-side utilities are written in C++, Tcl, Java and Python. It Although it is a highly scalable simulator which can simulate up to 1000 mobile robots, it should be noted that accurate results are hard to obtain through this simulator. Also, the physics engine used is naive which is inferior when compared to the engine used in Gazebo.

3.2. UW sim

In [13], UWsim is claimed to be an underwater simulator for marine robotics research and development. 3D modelling of the environment and robot can be done using third party applications such as blender and 3D studio Max. Robot description is given through XML file according to the URDF format. It integrates physics engine bullet with OSG through osgBullet. Although most of the important sensors and features are available with UWsim, it is developed on Ubuntu and doesn't support windows and mac operating systems.

3.3. Kelpie

A modular, open-source, and ROS-Based multi-robot simulator is being developed as part of the RIVERWATCH experiment. According to [5], Kelpie is developed on top of ROS specifically for aquatic and airborne robots. It also incorporates a physics engine bullet with OSG. The description and details available seems to be promising but of no use until available. The robot simulation environments specifically for aquatic robots are still in development phase, not released and cannot be used for projects straightaway. Hence, we look upon other available simulation software packages.

3.4. RDS

Microsoft Robotics Developer Studio (RDS) is a windows-based environment. RDS includes a lightweight REST-style, Node based simulation tool. It runs DSS Nodes for each model simulation. The nodes can be run remotely or on different ports of a local computer. Extensive documentation and tutorials are available. Algorithm designing for model robots is done in C#. It also provides the end user with a visual programming language to enable non-programmers command their robots/ simulate them. An example program was designed using VPL as shown in fig. 3

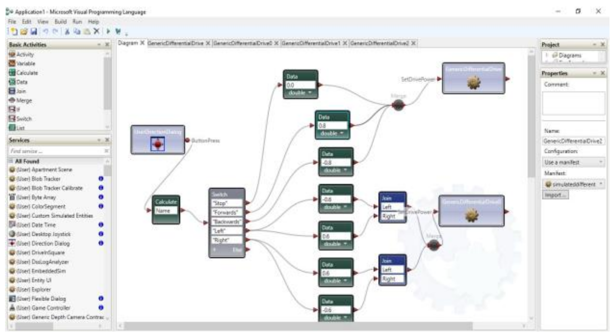


Fig. 3: Snapshot of the Example Vpl Command Created to Move the Model Robot in Four Directions Based on the Controller Input

To provide a rich simulation environment, RDS includes the following components:

- CCR - Concurrency and Coordination Runtime
- DSS - Decentralized Software Services
- VPL - Visual Programming Language
- VSE - Visual Simulation Environment
- Samples, Tutorials and Documentation

The installation and running of sample files is straightforward with the help of tutorials and guidelines available online on the Microsoft forum. However, few limitations which makes the package unlikely to be used for our purpose are as follows: the removed support for robotics by Microsoft. No further developments are being done [17]. To be precise creating custom entities (sensors, robots) in RDS is done using XNA framework, the support for which is removed in windows 10 – with the development of new framework. It doesn't support MAC and ubuntu operating systems.

3.5. MARS

Multi-agent Robotic simulator (MARS) is a MATLAB based simulator. According to [6], It is a 2D robot simulator platform for simulating a team of robots in structured/unstructured environments. The tool utilizes powerful capabilities of MATLAB to simulate complex navigation algorithms including obstacle avoidance and pattern formation. It is an easy choice to compare algorithms for multi-robotic systems with minimum robot modelling. The graphical output makes it easier to visualize the results of the simulated algorithm (fig.4). A simple text file as documentation is provided to help the user with the abstraction details. No active developments are ensured and hence, is not an appropriate choice for a long-term project.

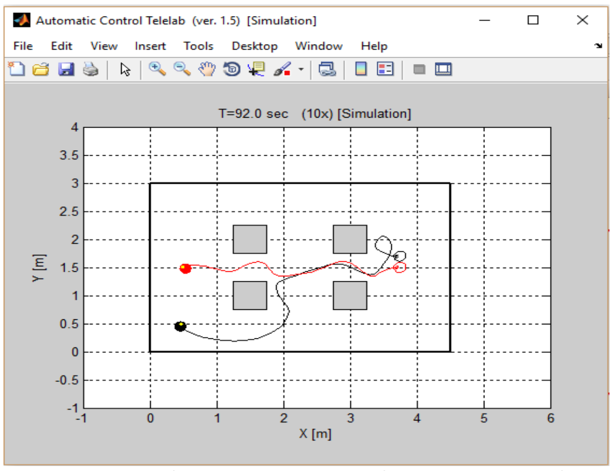

Automatic Control Telelab (ver. 1.5) [Simulation]
File Edit View Insert Tools Desktop Window Help
T=92.0 sec (10x) [Simulation]
Y [m]
X [m]
4
3.5
3
2.5
2
1.5
1
0.5
0
-0.5
-1
-1
0
1
2
3
4
5
6

Fig. 4: Screenshot of the Output Window of MARS Simulator for Two Robots Following a Path Planning Algorithm with Obstacle Avoidance on a Graphical Display Area (2D).

3.6. Net logo

Net logo is a programming language integrated with a modelling environment [7]. The environment allows the user to understand several emerging phenomena and can be used as an explanatory tool. The tool was built with the intention of serving a large audience and being able to implement in larger domains. The output can be made interactive and appealing to such an extent that it can be used by professors to explain various scientific phenomenon. With such a diverse simulator in-hand, alpha algorithm was implemented to test its features. Net-Logo contains extensive models in a variety of domain such as chemistry, biology, system dynamics and physics. Net-Logo provides the developers the options to make switches, sliders, choosers, inputs, and other interface elements which can make it more convenient for the end user to plug in the required inputs and visualize the output. The screenshots of the simulation have been included. The simulator simulates the robots in form of turtles and the environment in form of patches. The three triangles are the turtles (robots) and the red part is the environment. The robots are designed to be yellow colored when they aren't in close vicinity of one another to indicate that they are disconnected (fig. 5). When the robots enter the communication range of one another, they change color to blue and may communicate (fig. 6).

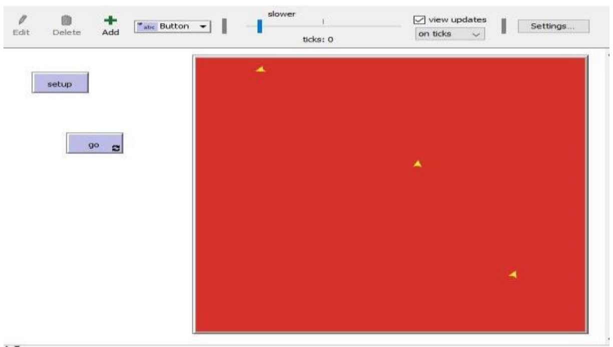


Fig. 5: The Snapshot of the Simulation Window of Net Logo with Three Turtles Not in Communication Range.

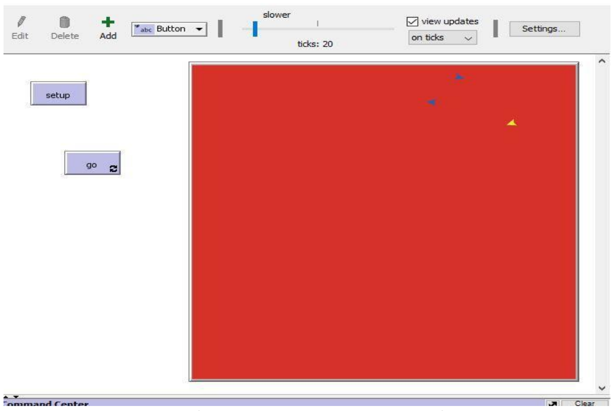


Fig. 6: The Snapshot of the Simulation Window of Net Logo with Two Turtles in Communication Range.

3.7. ARGOS

ARGoS can simulate huge heterogenous swarms of robots in real time. It is easy to use as every entity is projected as a plug-in. Multiple physical engines may run in one experiment, and the robots may move from one position to another in a clear manner. ARGoS is a customizable software and has a capability of parallel implementation. ARGoS supports most major platforms available like Windows, Linux and MacOS. Sample codes were implemented to realize that ARGoS is a one of the solutions one can use for any application of swarm robotics. It is a flexible simulator and hence, can support varied robot models. Easy installation makes it a fair choice.

Command to install ARGoS via terminal

```
brew tap ilpincy/argos3
brew install bash–completion qt lua argos3
```

After installation is complete, typing the following command will run the simulation environment as shown in fig. 7 and fig. 8

```
argos3 – c trajectory.argos ––config – file trajectory.argos
```

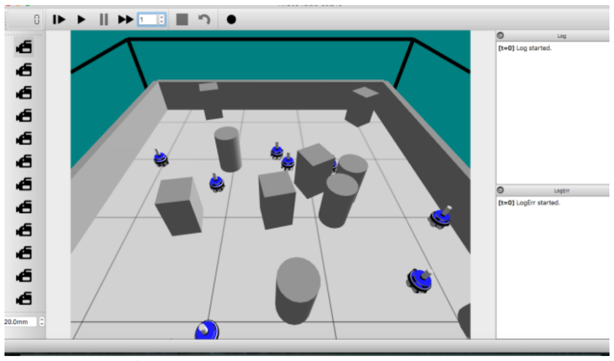


Fig. 7: A Snip of the Simulation Window of Argos with Multiple Robots and Obstacles.

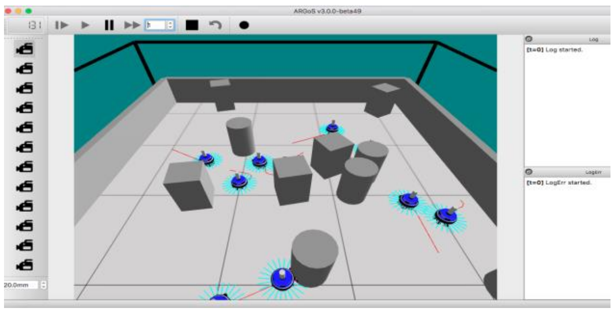
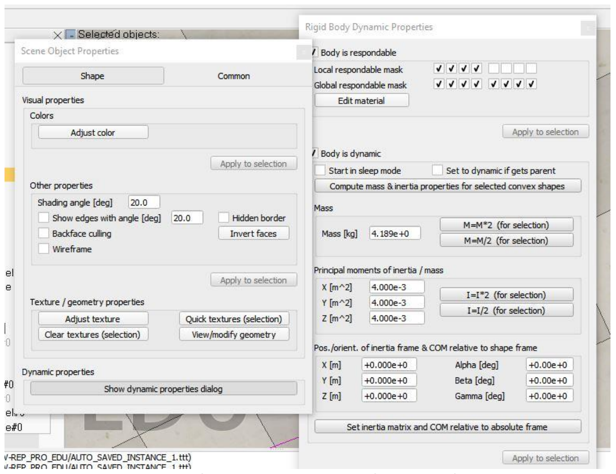


Fig. 8: A Still from the Simulation of Swarm Robots Following an Obstacle Avoidance Navigation Algorithm in Argos.

3.8. V-REP

V-REP, based on a distributed control architecture is a 3D simulator with integrated development environment which allows easy

modelling, editing, programming and simulating any robot in the environment. It can work as a standalone application or embedded with other robotic programming languages. C++, python, java, Lua, MATLAB are some of the languages which can be used to program the controllers.
Extensive documentation makes it easier to start with and understand the basic modelling. The user interface is trivial to understand and work upon, hence not requiring a lot of pre-study for using the software.
As per the user manual, it uses a lot of resources and slows down the system. However, this is of less importance in our application since swarm robotics makes use of simple robots which reduces the amount of complexity and hence bring down the resource utilization factor considerably. Therefore, VREP isn't eliminated from the list of potential simulators that can be efficiently used to model and simulate swarm robotic systems. V-REP can be downloaded for the respective OS from the official website. One can start modelling right away using the graphical user interface as shown in fig. 9


Scene Object Properties

Visual properties

Shading angle [deg] 20.0
Show edges with angle [deg] 20.0

Rigid Body Dynamic Properties

✓ Body is respondable
Local respondable mask ✓ ✓ ✓ ✓ ☐ ☐ ☐ ☐
Global respondable mask ✓ ✓ ✓ ✓ ✓ ✓ ✓ ✓

✓ Body is dynamic
☐ Start in sleep mode ☐ Set to dynamic if gets parent

Mass [kg] 4.189e+0

Principal moments of inertia / mass
X [m^2] 4.000e-3
Y [m^2] 4.000e-3
Z [m^2] 4.000e-3

Pos./orient. of inertia frame & COM relative to shape frame
X [m] +0.000e+0 Alpha [deg] +0.00e+0
Y [m] +0.000e+0 Beta [deg] +0.00e+0
Z [m] +0.000e+0 Gamma [deg] +0.00e+0

V-REP_PRO_EDU/AUTO_SAVED_INSTANCE_1.ttt)

Fig. 9: The Snapshot of GUI Dialog Box for Modifying the Shape and Other Properties of the Model Robot.

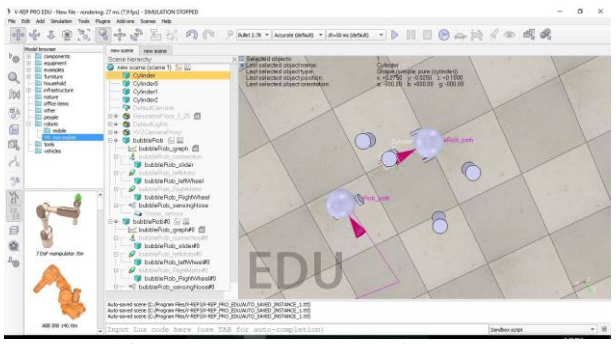

V-REP PRO EDU - New file - rendering: 27 ms (7.9 fps) - SIMULATION STOPPED
File Edit Add Simulation Tools Plugins Add-ons Scenes Help
Model browser
Scene hierarchy
bubbleRob
7 DoF manipulator.ttm
EDU
Input Lua code here (use TAB for auto-completion)

Fig. 10: A Still from the Simulation Window of V-REP with Two Robots Following an Obstacle Avoidance Strategy

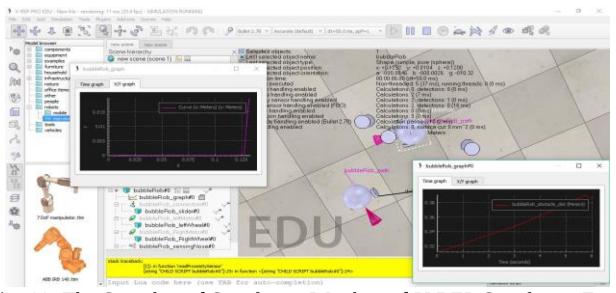


Fig. 11: The Snapshot of Simulation Window of V-REP Simulation Environment with Graphs during Simulation.

3.9. Gazebo

Gazebo is a simulator which can accurately and efficiently simulate robots in both indoor and outdoor 3D environments. The simulator contains high-quality graphics, suitable graphical and programmatic interfaces. Modelling of robots is done in URDF language which is like XML. Modelling can also be done using the graphical user interface. Gazebo is generally integrated with ROS - robot operating system language. This integration makes gazebo installation and working a little complex. Specific versions of gazebo need to be downloaded to work with specific versions of ROS. According to the documentation, gazebo can be downloaded and installed on windows. However, it doesn't work well. Specific versions of gazebo are coupled with specific versions of ubuntu. Gazebo utilizes a lot of resources and makes the system slow. However, integration with ROS and powerful physics engine are a trade off with the above-mentioned limitations. A basic model was modelled using the simulator, screenshot is attached herewith in fig. 12.

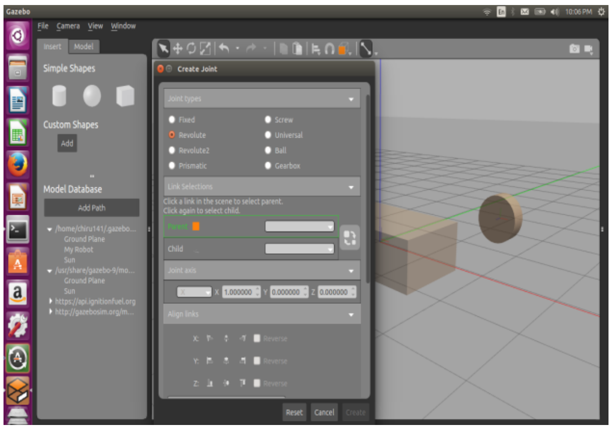


Fig. 12: Screenshot of the Simulation Environment of Gazebo with Graphical Model Modifying Features

4. Swarm robotics for oil skimming

Transportation of oil across continents is carried out with the help of ships and submarines. Casualties and accidents during transportation cause oil spills which endanger the aquatic life and pollute the water bodies [8]. Cleaning of oceans, seas and rivers has been a difficult task due to its vastness and ever-changing environment. Application of swarm robotics in this field can help overcome these difficulties and provide necessary infrastructure to the government to carry out the task with increased efficiency and reduced cost [9]. The stages of a swarm system removing oil from the water bodies is illustrated below. The procedure begins with deployment of the robots into the marine environment. The deployment may be close to or far away from where the oil spill may occur in the future. Phase A begins with the sensing stage, where the robots will sense where the oil has spilled. The robots will also be able to keep track of the presence of obstacles and other robots. In phase B, we have shown that the robots will communicate with each other to pass on information about the spill and also to stay together as a swarm. In the next phase C, the robots will plan a path and travel to the oil spill, avoiding the obstacles that come in their way. The robots make sure to travel as a swarm and not individually. Phase D demonstrates how the robots flock together and form patterns to start the process of cleaning oil spills. Lastly in phase E we see that the robots have cleaned the oil from the environment. The above-mentioned stages have been discussed below in detail [10]:

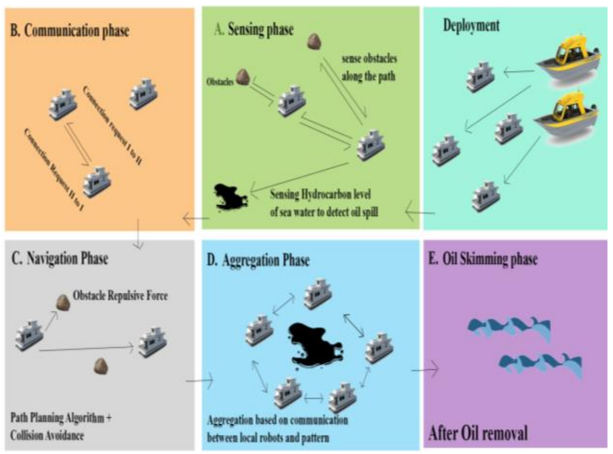

B. Communication phase
Connection request I to II
Connection Request II to I
A. Sensing phase
Obstacles
sense obstacles along the path
Sensing Hydrocarbon level of sea water to detect oil spill
Deployment
C. Navigation Phase
Obstacle Repulsive Force
Path Planning Algorithm + Collision Avoidance
D. Aggregation Phase
Aggregation based on communication between local robots and pattern
E. Oil Skimming phase
After Oil removal

Fig. 13: A Visual Representation of the Proposed Stages of Oil Skimming by Swarm Robotic System

4.1. Sensing stage

The swarm of robots may be deployed in the ocean at a location which is far from the oil spill, or the oil spill hasn't occurred, and the robots should be able to sense the spill in the future and navigate towards it. Hence, the first stage involves having a sensory module attached to the robot. Various sensors are to be included in each robot, namely a Global Positioning System (GPS) receiver, a digital compass unit, a temperature sensor, oxygen and carbon dioxide sensor. The readings from the sensors – temperature, co2, o2 sensor are to be recorded and information shall be derived after checking with the acceptable levels of these elements based on location and time. The result of this step is integral/Boolean. If the values match there is an indication of the presence of oil. The direction of the fleet of the swarm robots should be managed depending upon the result obtained and the information communicated from other bots. The robot which senses the presence of an oil spill changes its direction, communicates the information to its nearby bots using the communication model.

4.2. Communication stage

Each robot communicates its location through GPS and Wi-Fi to create an organized system for collection that can work continuously without human support. While there can be problems related to Wi-Fi and internet connectivity as the bots will work in remote areas, the communication model can be based on underwater sound properties, as SONAR may be used. The communication model is an important aspect of a decentralized network of robots, since the robots need to continuously talk with each other and carry out the required tasks and not individually.

4.3. Navigation stage

The swarm of robots needs to navigate in the ocean to the location where the oil spill has taken place. Many different algorithms may be used which the swarm would follow to reach from the initial to a final position. These algorithms should be as optimum as possible to reduce time, cost and resource consumption [11]. The navigation algorithm will also need to cater to the obstacles that the robots will encounter on the way in the form of debris, marine life and other landforms. It should be made sure that the robots don't collide with these as well as don't bump into each other. A safe distance parameter needs to be considered to obtain such collisions. The other factor is that the entire swarm, and not just individual robots, needs to coordinate and reach the final location. This process of navigation along with obstacle avoidance is known as path planning. The entire route or the path that swarm will take is decided and then followed. At times the dynamics of the ocean may cause problems to the system. Hence the swarm robotic system should be cautious of the possibility of an ever-changing environment and should be able to tackle and respond to it. Hence, we can see that the navigation process is a rather extensive and an important process.

4.4. Aggregation stage

The robots on reaching the oil spill need to arrange themselves around it. These spills may be of any shape and size. Hence, the swarm of robots need to communicate with each other and arrange themselves around the oil spill. After forming around the oil spill the robots can slowly and steadily start encroaching the spill to clean it. It should be noted that the order to be followed and the exact path is again a task of path planning. Hence, path planning is a task which is in turn needed in other tasks as well. This process also serves as a test for the control that exists in the swarm.

4.5. Oil skimming stage

The bots shall consist a conveyer belt that is made up of an oil absorbent material and is hydrophobic in nature. As the conveyer belt rotates, the bot moves forward on the surface of water. This conveyer belt absorbs the oil present on the surface of water and then the oil is separated in another chamber. There are two designs for handling the oil accumulated/collected. In one design the bot burns the oil in another chamber. And in the second design – The oil is stored at a place to be used later. Because the oil is processed locally the robots don't need to make repeated trips back to shore for maintenance, as for traditional skimmers. What's more, since the transport line is so steady – it sticks to the surface of the water, so it can't invert – the robot can work under outrageous conditions and harsh climate.
Each robot in the system will be in all the above stages mentioned. The robots are intended to be deployed in an aquatic environment. A human operator may control the bots through leader bots which can be selected based on some predetermined parameters [12]. Once the bots are deployed. The sensors and actuators measure the required parameters and make decisions to change its direction. The leaders of the system can be changed using the information obtained and derived dynamically. Once the robots reach the region of oil spill, they absorb the oil and remove it from the environment by either using it or burning it away. It is important to realize that the system is decentralized, and the leader is for the sole purpose of communicating with the human operator. In the cleaning process, human operator possesses no superior differences. The above steps and modules are illustrated in fig. 13 below which gives a clear understanding of the oil skimming process using swarm robotics.

5. Conclusion

In this paper, a comprehensive study and comparison of several robotic simulators has been presented. The simulators have been identified which best suit swarm robotic tasks and can simulate aquatic environments. It was realized that V-REP, Gazebo, Argos, Net Logo were the simulators that have enough documentation, proper development road map and support to enable researchers to use them without interruption. We attached the screenshots to present the reader with the opportunity to get a better idea of the simulator and its working. To get a concise overview of the simulators we drew a comparison table between the same. Application of swarm robotics in marine pollution monitoring and restoring has been explored. The various stages that will be involved in the process of oil-skimming have been illustrated in detail. This paper along with the detailed documentation of the various simulators present on the internet will be a stepping stone for beginners to gain extensive knowledge on the available robotic simulation software

6. Future work

Future work to the project includes expanding our knowledge about various robotic simulators. It may also include designing and developing algorithms for various stages of the oil skimming process and simulating it on the most favourable simulator.

7. Comparison tables

Comparisons table for 2D and 3D simulators are included as Table 1 and Table 2 respectively.

Table 1: Comparison Table for 2D Simulators

Comparison Parameter	Mars	Net Logo
Platform Support	• Macos • Windows • Linux	• Macos • Windows • Linux
	Basic Simulator Capabilities	
Physics Engine Available	No	No (Uses Simple Modelling And Rule-Based Coding)
Coding Language	Matlab Functions For Controller.	Own Language For Controlling Behaviour And Environment Specifications.
Graphical Simulation	Yes	Yes
Depiction Of Robots And Obstacles	Geometric Shapes	Geometric Shapes (Triangular By Default)
Output Trends	Data Plots And Text Files	Data Charts And Plots (Graphical Display)
Main Usage Of Simulator	To Visualize And Compare Complex Algorithms Without Getting Into Real Modelling Of Robots.	To Illustrate And Explain The Various Scientific And Engineering-Based Phenomenon Graphically.
	Modelling Details	
Sensors	Yes (Simple Sensors Such As Proximity Sensor And Range Finder)	No
Modelling Focus	No Precision In Modelling Of Robots. It Focuses On Simulation Of Robotic Systems (Algorithms) Rather Than Robots.	Doesn’t Model Robots But Focusses On Simulation Of The Robotic System Behaviour And The Algorithms Pertaining To It.
	Programming Methodology	
Scene Extension	It’s A Matlab File. All Scene Editing Therefore Must Be Done Using The Matlab.	.Nlogo Extension. The Editing Is Done In The Code Editor View, Again Using The Language Unique To Net Logo.
Simulation Environment	Standalone	Standalone
Language Used For Programming The Environment	Matlab Scripts	Unique Language To Net Logo
The Working Of Sample Programs Provided By The Simulator	All Demo .M Files Worked Properly	All Of Them Worked Properly
Detailed Documentation For The Simulator Understanding	Yes	Yes
Advancements	No Active Developments Announced, Hence Less Likely To Have New Features Extended.	A 3d Simulator For The Same Exists As An Advancement To The 2d Version. There Also Exists An Active Developer Community For The Same.
	User Interface Attributes	
User Interface	Matlab Interface.	The User-Interface Is A Simple Layout Of The Output Screen Along With Space To Have The Sliders, Buttons And Other Action Keys. The Page Will Be Toggled To Switch Between The Edit Mode And The Output Screen.
User Interface And Functionality Details Documentation	Exists	Exists In The Same Manual

Freezing Of The Simulator While Using It.	No	No
As A Comparison Simulator	Yes (Applicable For Comparing Various Algorithms)	No

Table 2: Comparison Table for 3D Simulators

Comparison Parameter	V-Rep	Gazebo	Argos
Platform Support	• Macos • Windows • Linux	• Macos • Windows • Linux	• Macos • Linux
	Basic Simulator Capabilities		
Physics Engine	Yes Default/Initial Physics Engines Include: Bullet 2.78, Bullet 2.83, Ode, Vortex And Newton.	Yes The Ode Physics Engine Is The Default /Initial One Present But However, One Can Build Gazebo From Scratch As Well, Using A Different Physics Engine.	Yes A Custom-Built 2d Or 3d Physics Engine Having Restricted Capabilities Are Available Initially.
Scene And Code Editors	• Graphical And Xml Scene Editors	• Scene Editor • Script Editor	• Script Editor
Scene Objects Are Fully Interact-Able During Simulation	Yes	Yes	Yes
Mesh Manipulations	Yes	No	No
Output Trends	Video, Custom Data Plots And Text Files.	Simulation Log Files, Video Frames As Pictures And Text Files.	Video Frames As Pictures And Text Files.
	Modelling Details		
Different Types And Models Of Available Robots	Wheeled, Aerial As Well As Snake-Like Robots	Flying And Wheeled Robots. Third-Party Robot Models Are Available, But Often Contain Poor Documentation.	Small Library Of Robots. E-Puck, Eye-Bot, Kilobot, Marxbot, And Spiri Robots.
Simulations Possible With The Simulator	High-Precision Simulations Possible With Highly Detailed Models Available.	Appropriate For Computationally Complex Simulations.	Appropriate For Computationally Complex Simulations.
	Programming Methodology		
Scene Extension	Vrep Format And Hence The Editing Also Needs To Be Done Using The Vrep Interface.	Xml File And Hence It Is Possible To Create A Script And Change The Scene And Run The Simulation.	Xml File And Hence It Is Possible To Create A Script And Change The Scene And Run The Simulation.
Functionality Execution	Scripts Joined To Robots, Plug-Ins, Ros Hubs Or Separate Projects That Interface With V-Rep Through The Remote Api.	C++ Modules Or By Means Of Ros Programs.	Lua Contents Or In C++.
Functionality Documentation	Scripts Can Be Incorporated Into Robot Models And Are Frequently Used To Portray The Models And Their Capacities.	It Is Hard To Perceive How An Outsider Robot Model Functions Or What Modules It Uses. The Module List Is Just Accessible In The Model Editor.	Some Documentation Of The Robots Is Given In Argos, However The Vast Majority Of How A Robot Function Should Be Deducted From Code Cases.
The Working Of Sample Scripts And Modules Provided By The Simulator	Worked Properly.	Some Didn’t Work Properly	Worked Properly.
Detailed Documentation For The Simulator Understanding	Yes Regular Updates Since 2013.	Yes Improvement Guide Also Exists On The Website.	Yes Inconsistent Advancements.
	User Interface Attributes		
Freezing Of The Simulator While Using It.	No	Yes	No
User Interface And Functionalities	All Functionality Is Genuinely Natural And Takes After General Traditions Known From Comparative Applications.	The Ui Usability Is Relatively Low. For Example, The Top Application Tool Bar Sometimes Disappears, It Is Not Possible To Copy And Paste Multiple Objects	The Ui Is Extremely Constrained; However, All Functionality Is Genuinely Instinctive And Takes After General Traditions Known From Comparative

			Applications.
Model-Library Provided Along With The Software	Yes	No Accessible On-Line.	Yes
Efficient Segregation Of Model Libraries Into Folders	Yes	No	No

References

[1] I. Navarro and F. Matía, "An Introduction to Swarm Robotics," ISRN Robot. vol. 2013, pp. 1–10, 2013.

[2] L. Bayindir, "A review of swarm robotics tasks," Neurocomputing, vol. 172, pp. 292–321, 2016.

[3] A. Staranowicz and G. L. Mariottini, "A survey and comparison of commercial and open-source robotic simulator software," Proc. 4th Int. Conf. PErvasive Technol. Relat. To Assist. Environ. - PETRA '11, p. 1, 2011.

[4] M. Duarte et al., "Application of Swarm Robotic Systems to Marine Environmental Monitoring," Proc. IEEE/MTS Ocean. pp. 1–8, 2016.

[5] R. Mendonça, P. Santana, F. Marques, A. Lourenço, J. Silva, and J. Barata, "Kelpie: A ROS-based multi-robot simulator for water surface and aerial vehicles," Proc. - 2013 IEEE Int. Conf. Syst. Man, Cybern. SMC 2013, no. January 2014, pp. 3645–3650, 2013.

[6] T. Tosik, J. Schwinghammer, M. J. Feldvoß, J. P. Jonte, A. Brech, and E. Maehle, "MARS: A simulation environment for marine swarm robotics and environmental monitoring," Ocean. 2016 - Shanghai, 2016.

[7] C. Dixon, A. F. T. Winfield, M. Fisher, and C. Zeng, "Towards temporal verification of swarm robotic systems," Rob. Auton. Syst., vol. 60, no. 11, pp. 1429–1441, 2012.

[8] R. C. Harrel, "Effects of a crude oil spill on water quality and macrobenthos of a southeast Texas stream," Hydrobiologia, vol. 124, no. 3, pp. 223–228, 1985.

[9] A. Jevtic and D. Andina, "Swarm Intelligence and Its Applications in Swarm Robotics," Proc. 6th Wseas Int. Conf. Comput. Intell. ManMachine Syst. Cybern., no. January, pp. 41–46, 2007.

[10] N. M. P. Kakalis and Y. Ventikos, "Robotic swarm concept for efficient oil spill confrontation," J. Hazard. Mater. vol. 154, no. 1–3, pp. 880–887, 2008.

[11] L. Silva and N. Nedjah, "Efficient strategy for collective navigation control in swarm robotics," Procedia Comput. Sci., vol. 80, pp. 814–823, 2016.

[12] P. Walker, S. Amirpour Amraii, N. Chakraborty, M. Lewis, and K. Sycara, "Human control of robot swarms with dynamic leaders," IEEE Int. Conf. Intell. Robot. Syst., pp. 1108–1113, 2014.

[13] Dhurandher, S. K., Misra, S., Obaidat, M. S., & Khairwal, S. UWSim: A simulator for underwater sensor networks. Simulation, 84(7), 327-338, 2008

[14] The Hindu. (2018). Chennai oil spill. [online] Available at: http://www.thehindu.com/news/cities/chennai/chennai-oil-spill/article17170741.ece1 [Accessed 15 Mar. 2018].

[15] The Hindu. (2018). A slick on the shore and the murky aftermath. [online] Available at: http://www.thehindu.com/news/cities/chennai/a-slick-on-the-shore-and-the-murky-aftermath/article17290268.ece1 [Accessed 15 Mar. 2018].

[16] Anon, (2018). [online] Available at: http://www.thehindu.com/news/cities/chennai/As-clean-up-continues-oil-spill-reaches-Mamallapuram/article17295236.ece [Accessed 15 Mar. 2018].

[17] Anon, (2018). [online] Available at: http://spectrum.ieee.org/automaton/robotics/robotics-software/microsoft-shuts-down-its-robotics-grouphtt [Accessed 15 Mar. 2018].

[18] IIT Madras&#x27 Bluetooth-controlled robots sweep area clean, e. (2018).45 Bluetooth-controlled robots by IIT Madras students swept an area clean, enter records. [online] YourStory.com. Available at: https://yourstory.com/2017/11/iit-madras-robots/ [Accessed 15 Mar. 2018.