ISSN 2229-5518

Design and Simulation of Underwater Motion Sensing

V Renuga, Student at Dr.G.R.Damodaran College of Science.
renugaragavan94@gmail.com
Dr K Vanitha, Associate Professor, Dr.G.R.Damodaran College of Science.
vanitha.k@grd.edu.in

Abstract— Underwater sensing is a challenging domain in which the acoustic signals to be captured and processed. Setting up a motion capturing environment in underwater supplies the water body movement, water habited monitoring and ample number of applications. This project is aimed to deploy a motion sensor in a UAN designated to observe and monitor the fish movements. The observed data is processed for the classification, fishing and feeding of fishes. This can be attached to the software to watch the movements of the fishes. Through this we can observe the movements of the fishes lively.

Keywords— Simulation, UnetSim, Motion sensing, Aloha, Node mobility, Plot tracking.

1 Introduction

Experimentation with underwater robots is normally very difficult due to the high number of resources required. A water tank – high enough for the systems to be tested– is normally needed, which requires significant space and maintenance. Another possibility is to access to open environments such as lakes or the sea, but this normally involves high costs and requires special logistics. In addition, the nature of the underwater environment makes it very difficult for researchers (operating in the surface) to observe the evolution of the running system. As a consequence, experimental validation of these systems is highly laborious. In order to facilitate the development of underwater robots, it is of utmost importance to develop suitable simulators that allow to

1) Develop and test the systems before they are deployed, and
2) Supervise a real underwater task where the developers do not have a direct view of the system.

Fish detection is not a new one; most of the researches have tried to develop fish detection system. They even expand their ideas to localization of schooling fish which is similar with fish finder. Variety of techniques and algorithms have been developed and implemented, where some of them use ultrasound based fish finder combined with Global Positioning System (GPS). A simple smart algorithm embedded into a small light weight device has to be used for a small fish robot or a miniature submarine, which has limited area for equipments. Hence, the fish finder features can still be improved to have the ability to classify the type of fishes, by adding pattern classification algorithm such as Hidden Markov Model, clustering, or artificial neural network to the system. The current fish detection devices are expensive and cannot be deployed in many small underwater or submarine robots because they have limited space (Tidd and Wilder, 2001). Therefore, it is necessary to develop new type of fish detection devices which are cheap and compact in size.

An important characteristic of the underwater sensor network we built is mobility. The sensor network includes both static and mobile nodes. Mobility enhances the performance of this sensor network in several ways.

1) It provides a means for deploying, reconfiguring, and retrieving the nodes in the network.
2) It permits large area coverage with sparse networks which is especially important in an underwater environment - a much harder space to access than terrestrial space. The mobile nodes can move across the field to ensure the necessary connectivity.
3) Mobile nodes can act as data mules and travel from node to node across a sparsely deployed sensor network to collect data.

Communications is enabled only when the sensors and the mobile mules are in close proximity. Transmitting data over these shorter distances reduces the power consumption on each sensor and alleviates the hot spot problem on the sensors near the destination. Moreover, since underwater acoustic communication is characterized by low data rates, and optical underwater communication is subject to short ranges, mobility enables a time-e_cient and more energy-efficient means to collect and transmit the data.

2 Literature Review

Implementation of moving object detection in field programmable gate array (FPGA) is presented. This monitors movement of animals, humans or vehicles across a desired area. The system can be used for automatic under water vision system for monitoring moving objects to avoid potential human errors [1]. A machine vision system capable of analyzing underwater videos for detecting, tracking and counting fish is presented.

ISSN 2229-5518

The video processing system consists of three subsystems: the video texture analysis, fish detection and tracking modules. Fish detection is based on two algorithms computed independently, whose results are combined in order to obtain a more accurate outcome. The tracking was carried out by the application of the CamShift algorithm that enables the tracking of objects whose numbers may vary over time [2].

Underwater sensor networks to be used for long-term monitoring of coral reefs and fisheries. The nodes communicate point-to-point using a novel high-speed optical communication system integrated into the TinyOS stack, and they broadcast using an acoustic protocol integrated in the TinyOS stack. The nodes have a variety of sensing capabilities, including cameras, water temperature, and pressure. They can perform network maintenance functions such as deployment, relocation, and recovery [3].

Proximity ping sensor widely used in mobile robot with some dedicated signal preconditioning and processing of extracted features with the proposed algorithm, a fish detection and classification system has been realized [4]. Fish finders have already been widely available in the fishing market for a number of years.

However, the sizes of these fish finders are too big and their prices are expensive to suit for the research of robotic fish or mini-submarine. The goal of this research is to propose a low-cost fish detector and classifier which suits for underwater robot or submarine as a proximity sensor. With some pre-condition in hardware and algorithms [5].

This compares different sensor network architecture designs that can be used for monitoring underwater pipeline infrastructures. These architectures are underwater wired sensor networks, underwater acoustic wireless sensor networks, RF (Radio Frequency) wireless sensor networks, integrated wired/acoustic wireless sensor networks, and integrated wired/RF wireless sensor networks. It also develops and evaluates a hierarchical sensor network framework for underwater pipeline monitoring [6].

3 Problem Specification

ALOHA Simulation: The ALOHA used a new method of experimental ultra high frequency (UHF) for its operation, since frequency assignments for communications to and from a computer. By using ALOHA simulation, we can get the frequency of the fishes. The application of an ALOHA channel are cables and satellites.
Node Mobility: If nodes change their location over time, it will update their location. Node mobility is used to get the location and movement of the fishes.
Package org.arl.unet.addr: Through this package we can stimulate the robotic fish and get address location of other fishes.
Package org.arl.unet.net: This package is used to discover route of the fish and gives the route trace notification.

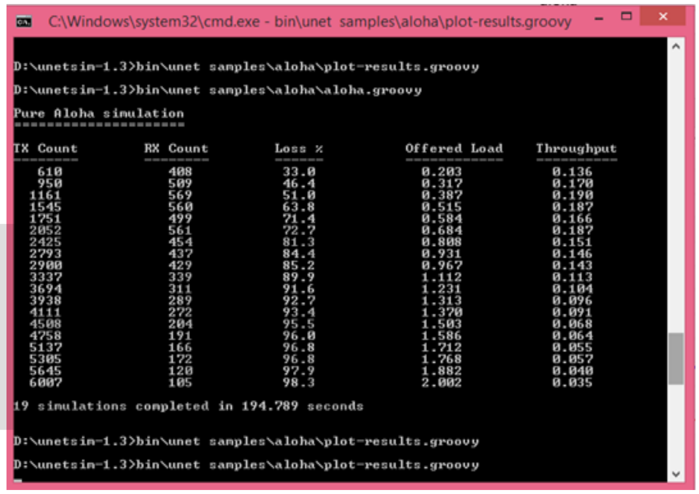
Package org.arl.unet.nodeinfo: This package is used to detect the robotic fish attribute such as address, location, speed etc.

4 Proposed System

The Underwater motion sensing robot has similar features like roaming robot. But it has more additional features than roaming robot. The Underwater motion sensing robot is used to capture the actions which performs in the underwater and retrieves the data by fixing the camera into artificial fish. The artificial fish records the movements of other fishes in aquarium and provides the data streaming technology. Through this we can able to see the movements of the fishes lively.

5 Experimental Results

5.1 ALOHA Simulation


C:\Windows\system32\cmd.exe - bin\unet samples\aloha\plot-results.groovy

D:\unetsim-1.3>bin\unet samples\aloha\plot-results.groovy

D:\unetsim-1.3>bin\unet samples\aloha\aloha.groovy

Pure Aloha simulation

TX Count	RX Count	Loss %	Offered Load	Throughput
610	408	33.0	0.203	0.136
950	509	46.4	0.317	0.170
1161	569	51.0	0.387	0.190
1545	560	63.8	0.515	0.187
1751	499	71.4	0.584	0.166
2052	561	72.7	0.684	0.187
2425	454	81.3	0.808	0.151
2793	437	84.4	0.931	0.146
2900	429	85.2	0.967	0.143
3337	339	89.9	1.112	0.113
3694	311	91.6	1.231	0.104
3938	289	92.7	1.313	0.096
4111	272	93.4	1.370	0.091
4508	204	95.5	1.503	0.068
4758	191	96.0	1.586	0.064
5137	166	96.8	1.712	0.055
5305	172	96.8	1.768	0.057
5645	120	97.9	1.882	0.040
6007	105	98.3	2.002	0.035

19 simulations completed in 194.789 seconds

D:\unetsim-1.3>bin\unet samples\aloha\plot-results.groovy

D:\unetsim-1.3>bin\unet samples\aloha\plot-results.groovy

Figure 1: Simulation of ALOHA

ALOHA have two types: Pure aloha and Slotted aloha. The simulation of an ALOHA has been executed as show in Figure 1. It has the process of finding the frequency of the fishes and it will be plotted in both pure aloha and slotted aloha. Simulation of aloha wireless network are as follows: To run simulation bin/unet samples/aloha/aloha.groovy. Set the modem settings for a simple modem with 1 second frame duration by setting up each node at origin to ensure no propagation delay between nodes. Drop any ongoing TX/RX and then send frame to random node and display the statistics to show TX count, RX count, throughtput and so on.

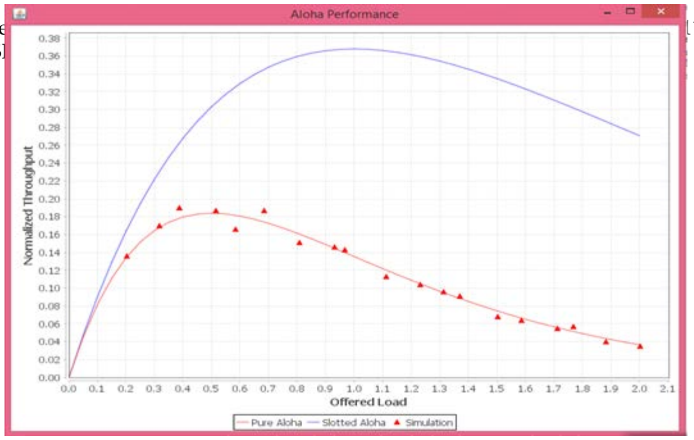


Figure 2: Plotting the nodes of ALOHA simulation

```
//! Simulation: Aloha wireless network
/// To run simulation:
///   bin/unet samples/aloha/aloha
/// Output trace file: logs/trace.nam
/// Plot results: bin/unet samples/aloha/plot-results
import org.arl.fjage.*
import org.arl.unet.*
import org.arl.unet.phy.*
import org.arl.unet.sim.*
import org.arl.unet.sim.channels.*
import static org.arl.unet.Services.*
import static org.arl.unet.phy.Physical.*
println '''
Pure Aloha simulation
=====================
TX Count\tRX Count\tLoss %\t\tOffered Load\tThroughput
--------\t--------\t------\t\t------------\t----------'''
// modem settings for a simple modem with 1 second frame duration
channel.model = ProtocolChannelModel
modem.dataRate = [2400, 2400].bps
modem.frameLength = [2400/8, 2400/8].bytes
modem.headerLength = 0
modem.preambleDuration = 0
modem.txDelay = 0
// simulation settings
def nodes = 1..6                    // list of nodes
def loadRange = [0.2, 2.0, 0.1]       // min, max, step
def T = 1.hours                     // simulation horizon
trace.warmup = 10.minutes           // collect statistics after a while
// simulation details
for (def load = loadRange[0]; load <= loadRange[1]; load += loadRange[2])
{
simulate T, {
 // setup each node at origin to ensure no propagation delay between
nodes
nodes.each { myAddr ->
def myNode = node("${myAddr}", address: myAddr, location: [0, 0, 0])
myNode.startup = {
def phy = agentForService PHYSICAL
add new PoissonBehavior(1000*nodes.size()/load, {
// drop any ongoing TX/RX and then send frame to random node, except
myself
phy << new ClearReq()
phy << new TxFrameReq(to: rnditem(nodes-myAddr), type: DATA)
})
}
}
} // simulate
 // display statistics
float loss = trace.txCount ? 100*trace.dropCount/trace.txCount : 0
println sprintf('%6d\t\t%6d\t\t%5.1f\t\t%7.3f\t\t%7.3f',
[trace.txCount, trace.rxCount, loss, trace.offeredLoad, trace.throughput])
} // for
```

Plot the nodes of ALOHA simulation bin/unet samples/aloha/plot-results.groovy. It will plot the nodes in both pure aloha and slotted aloha. This can be plotted in normalized throughput and offered load as shown in Figure 2.

5.2 Node Mobility

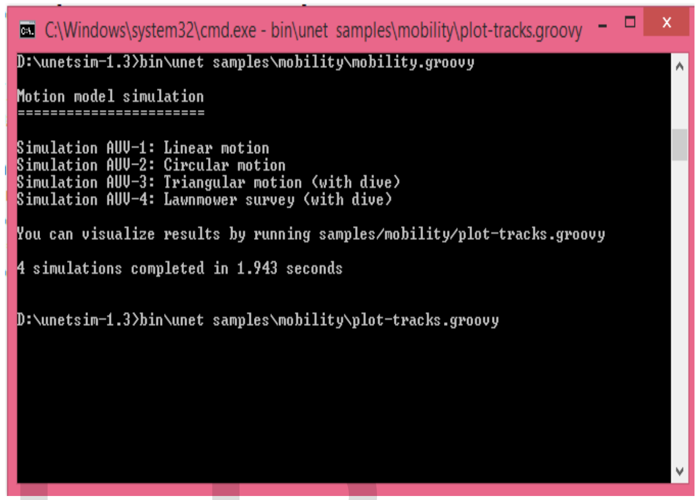


Figure 3: Simulation of Node Mobility

ALOHA have two types: Pure aloha and Slotted aloha. The simulation of an ALOHA has been executed as show in Figure 3. It has the process of finding the frequency of the fishes and it will be plotted in both pure aloha and slotted aloha. Simulation of aloha wireless network are as follows: To run simulation bin/unet samples/mobility/mobility.groovy. Set the utility closure to log AUV locations every 10 seconds or as we want. Set the Linear motion, Circular motion, Triangular motion and Lawnmower survey to plot them for a simulation. Plot the nodes of mobility bin/unet samples/mobility/plot-tracks.groovy. It will plot the nodes in meters as shown in Figure 4. It is used to get the location of the fishes (i.e nodes of the fishes).

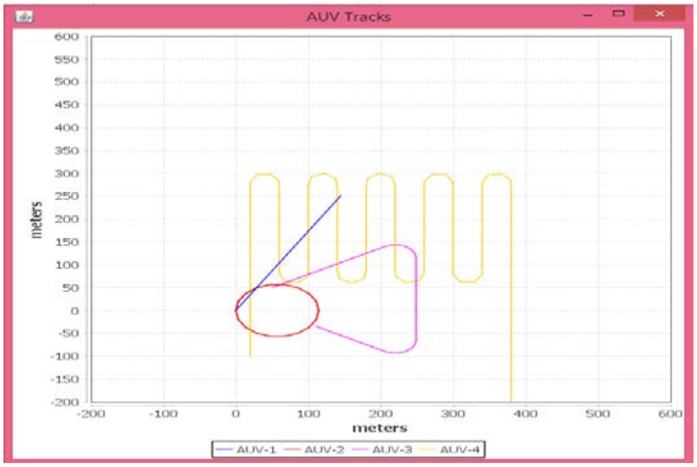


Figure 4: Plotting the node mobility simulation

```
//! Simulation: AUV motion patterns
/// To run simulation:
///   bin/unet samples/mobility/mobility
/// Output trace file: logs/trace.nam
/// Plot results: bin/unet samples/mobility/plot-tracks
import org.arl.fjage.*
import org.arl.unet.sim.MotionModel
import static org.arl.unet.Services.*
println '''
Motion model simulation
=======================
'''
////// Utility closure to log AUV locations every 10 seconds
trackAuvLocation = {
def nodeinfo = agentForService NODE_INFO
trace.moved(nodeinfo.address, nodeinfo.location, null)
add new TickerBehavior(10000, {
trace.moved(nodeinfo.address, nodeinfo.location, null)
})
}
////// Linear motion
println 'Simulation AUV-1: Linear motion'
simulate 5.minutes, {
def n = node('AUV-1', location: [0, 0, 0], mobility: true)
n.startup = trackAuvLocation
n.motionModel = [speed: 1.mps, heading: 30.deg]
}
////// Circular motion
println 'Simulation AUV-2: Circular motion'
simulate 7.minutes, {
def n = node('AUV-2', location: [0, 0, 0], mobility: true)
n.startup = trackAuvLocation
n.motionModel = [speed: 2.mps, turnRate: 2.dps]
}
////// Triangular motion (with diving)
println 'Simulation AUV-3: Triangular motion (with dive)'
simulate 10.minutes, {
def n = node('AUV-3', location: [50.m, 50.m, 0], mobility: true)
n.startup = trackAuvLocation
n.motionModel = [[time:  0.minutes, heading:  60.deg, speed:      1.mps],
 [time:  3.minutes, turnRate:  2.dps, diveRate:  0.1.mps],
 [time:  4.minutes, turnRate:  0.dps, diveRate:    0.mps],
 [time:  7.minutes, turnRate:  2.dps],
 [time:  8.minutes, turnRate:  0.dps],
 [time: 11.minutes, turnRate:  2.dps, diveRate: -0.1.mps],
 [time: 12.minutes, turnRate:  0.dps, diveRate:    0.mps]]
}
////// Lawnmower survey (with diving)
println 'Simulation AUV-4: Lawnmower survey (with dive)'
simulate 1.hour, {
def n = node('AUV-4', location: [20.m, -100.m, 0], heading: 0.deg, mobility: true)
n.startup = trackAuvLocation
// dive to 30m before starting survey
n.motionModel = [[duration: 3.minutes, speed: 1.mps, diveRate: 0.2.mps], [diveRate: 1.mps]]
// then do a lawnmower survey
n.motionModel += MotionModel.lawnmover(speed: 2.mps, leg: 200.m, spacing: 20.m, legs: 10)
// finally, come back to the surface and stop
n.motionModel += [[duration: 6.minutes, speed: 3.mps, diveRate: 0.4.mps], [diveRate: 2.mps, speed: 3.mps]]
}
////// Done!
println "\nYou can visualize results by running samples/mobility/plot-tracks.groovy"
```

6 Conclusion and Future Work

The experimental shows the plots of pure and slotted aloha simulation of the nodes (i.e, fishes in the aquarium). And also it simulates the mobility of the fishes in the aquarium. Then it will be plotted in graphical view of fish's node. In the future research, the variation of mobility, aloha and ping daemon can be quantized by observing thoroughly a directional vector. Hence, it can become a good starting point to have complex classification of more than two types of fish. More aggressive fish with high agility swim ability will also be investigated.

References

[1] **AlexRaj S. M and Supriya M. H**, " Hardware Co-simulation of Underwater Moving Object Detection using Xilinx System Generator", International Journal of Oceans and Oceanography ISSN 0973-2667 Volume 10.

[2]**Yun-Heh Chen-Burger, Gayathri Nadarajan, Robert B. Fisher,** "Detecting ,Tracking And Counting Fish In Low Quality Unconstrained Underwater Videos.

[3] **I. Vasilescu, K. Kotay and D. Rus, M. Dunbabin and P. Corke,** "Data Collection, Storage, and Retrieval with an Underwater Sensor Network".

[4] **Yeffry Handoko, Bambang Riyanto, Yul.Y.Nazaruddin, Edi Leksono,** "Designing embedded fish sensor for underwater robot", Indian Journal of Marine Sciences Vol. 38(3), September 2009, pp. 308-315.

[5] Yeffry Handoko, Yul.Y. Nazaruddin, Huosheng Hu, "Using Echo Ultrasound from Schooling Fish to Detect and Classify Fish Types", Copyright © 2009, Elsevier Limited and Science Press.

[6] **Nader Mohamed *, Imad Jawhar, Jameela Al-Jaroodi and Liren Zhang,** "Sensor Network Architectures for Monitoring Underwater Pipelines", *Sensors* **ISSN 1424-8220, 2011**, *11*, 10738-10764; doi: 10.3390/s111110738

[7] **John Heidemann, Wei Ye, Jack Wills, Affan Syed, Yuan Li,** "Research Challenges and Applications for Underwater Sensor Networking", Information Sciences Institute, University of Southern California.

[8] **Ian F. Akyildiz *, Dario Pompili, Tommaso Melodia**, “Underwater acoustic sensor networks: research challenges”, Broadband and Wireless Networking Laboratory, School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332, Ad Hoc Networks 3 (2005) 257–279.

[9] **James R. Solberg, Kevin M. Lynch, Malcolm A. MacIver,** “Active Electrolocation for Underwater Target Localization”, Department of Mechanical Engineering, THE INTERNATIONAL JOURNAL OF ROBOTICS RESEARCH, May 2008.

[10] **Miquel Massot-Campos * and Gabriel Oliver-Codina**, “Optical Sensors and Methods for Underwater 3D Reconstruction”, Department of Mathematics and Computer Science, University of the Balearic Islands, Sensors **2015**, 15, 31525–31557.