NTNU – Trondheim
Norwegian University of
Science and Technology

Specification and First Prototype of Simulated Environment for Autonomous USV

Kjetil Svae Børs-Lind

August 2016

TTK4550 - Specialization Project

Department of Engineering Cybernetics

Norwegian University of Science and Technology

Supervisor: Morten Breivik, ITK

Co-supervisor: Rein Anders Apeland, Kongsberg Maritime

Preface

Some preface including acknowledgements.

Trondheim, 2012-12-16

(Your signature)

Ola Nordmann

Summary

...

Contents

Chapter 1

Introduction

Introduction to the problem, some background about the Survey Explorer project..

1.1 Background

Where do we stand? What has been done before?

Problem Formulation

Literature Survey

What Remains to be Done?

...

1.2 Objectives

The main objectives of this project are

1. Investigate existing solutions for HIL testing of autonomous boat
2. Describe implementation of sensors and data processing on Odin and/or Jolner for situational awareness above the surface.

3. Discuss complexity and solutions related to simulation of raw sensor data from Radar, Lidar and camera versus simulation of pre-processed data.

4. Specify interface between simulator and autonomous navigation system.

5. Specify system for logging and visualization of simulation in real-time and for post simulation analysis.

6. Necessary protyping in C++ and MATLAB to verify assumptions.

7. Investigate which other agents (ships, small boats etc.) that can be interesting to implement as a part of the simulation environment.

8. Discuss methods for using the model as a part of an automated test environment related to ROS, MROS, scripting, repetition and regression testing.

1.3 Limitations

1.4 Approach

1.5 Structure of the Report

The rest of the report is organized as follows. Chapter 2 gives an introduction to ...

Chapter 2

Existing Solutions and Framework

This chapter goes through some of the existing tools used for HIL simulations of marine systems. The key properties of each tool are listed, and an evaluation is made about the possibilities of using the tools in the implementation of our own simulation environment.

2.1 Robot Operating System (ROS)

Short about ROS and how we can utilize it in the project. Will write this later as i gain more experience with ROS.

2.1.1 Marine Robotics Operating System (MROS)

Kongsberg Maritimes own version of ROS, with some extensions. Need to learn more about this (Rein will send a paper on it by the end of November).

2.1.2 Gazebo

Looks interesting. Even has done some research on this...

2.2 CyberSea Simulator

The CyberSea Simulator developed by Marine Cybernetics is an advanced simulator for HIL testing of Dynamic Positioning (DP) systems. It is *probably* super expensive (how can i investigate this?) and mainly focused on motion dynamics of big vessels at low speed (less than 3kts).

Key properties of the CyberSea Simulator (Johansen et al. (2005)):

- Capabilities for real-time presentation of results.
- Emphasis on vessel dynamics and accurate simulation of vessel motion during DP.
- Advanced simulation of wave, wind and current loads in six degrees of freedom.
- Several options for practical interfaces between simulator and computer control system, both analog and digital using for example NMEA protocol or normal network protocol.
- Generation of realistic signals from all the common sensors and position reference systems such as *"Gyro-compasses, VRUs, wind sensors, thruster feedback [...], power feedback from thrusters, switchboard and generator sets"* (Johansen et al. (2005)) used in modern DP technology. The signals can also be contaminated with noise levels typical for the sensors in use.
- Advanced generation of GNSS signals with possibility of simulating a broad specter of common failure modes.

The CyberSea Simulator, although powerful and highly customizable, is probably too expensive to use as a part of our simulation environment. It is also not certain to which extent the simulator can simulate other active agents such as ship traffic for testing of collision avoidance.

2.3 Marine Systems Simulator (MSS)

The Marine Systems Simulator (MSS) is a free toolbox for MATLAB/Simulink developed by several professors, MSc and PhD students at the Norwegian University of Science and Technology (NTNU). It is a merge of 3 previously existing toolboxes: Marine GNC Toolbox, Marine Cybernetics Simulator (MCSim) and DCMV. The toolbox contains possibilities for modeling of the dynamics of ships, underwater vehicles and floating structures under different wave, wind and current conditions.

Key properties of MSS (Perez et al. (2006)):

- Good modularity in Simulink.
- Emphasis on vehicle dynamics and thereby well suited for developing good motion control of such vehicles.
- Can be set up to do HIL simulations.
- Possibility of 3D animation using Marine Visualization Toolbox.

The Marine Systems Simulator is free software[1] under the terms of GNU General Public License[2]. Its also well suited for easy simulation of advanced marine vehicle dynamics. It is likely that this toolbox can be used as a part of our simulation environment to simulate Odin's motions and possibly for 3D animations of such dynamics.

2.4 MCSim (Marine Cybernetics)

Trenger denne artikkelen: *Simulation-Based Design and Testing of Dynamically Positioned Marine Vessels.* Spør Roger Skjetne.

[1] http://www.marinecontrol.org/license.html
[2] http://www.gnu.org/licenses/gpl.html

Chapter 3

Implementation and Simulation of Sensors

Using electronic nautical charts (ENC), information about other simulated agents and 3D models of installations in sea it is possible to generate realistic sensor data for HIL simulations. This section contains an overview of the implementation of important sensors used on Odin and a brief discussion about how sensor data can be simulated. The Inertial Measurement Unit (IMU) is documented in (**Evens rapport, trenger referanse?**).

3.1 Sensors for Environmental Analysis Implemented on Odin

In section 3.1 a brief overview of the sensors on Odin used for situational awareness above the surface are presented. The overall layout of sensors and system architecture are visualized in Figure 3.1.

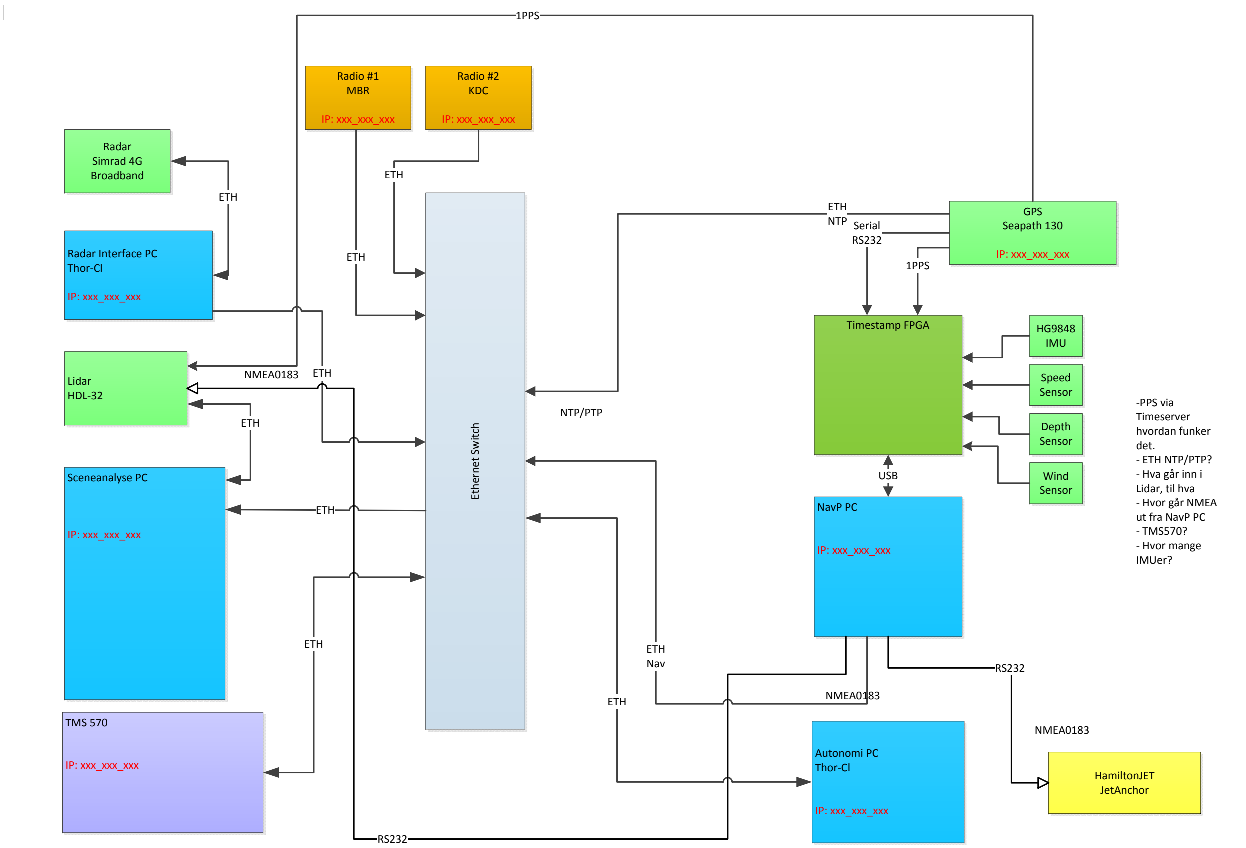

Figure 3.1: *Visualization of system architecture and network layout on Odin, including sensors and processing units.* **A better figure should be made when the final layout is determined!**

3.1.1 Seapath 134 GPS

The GPS used on Odin is the Seapath 134 developed by Kongsberg Seatex. Combining GNSS signals and IMU data the Seapath 134 gives highly accurate heading, position, heave, roll and pitch measurements (Kongsberg Seatex (2016)). Data is transmitted via standard NMEA 0183 protocol (SiRF (2005)) over a serial RS232 cable for interpretation by the control system. **Short introduction and small example of NMEA protocol should come here.**

Figure 3.2: *The Seapath 134 developed by Kongsberg Seatex is used on Odin for position, heading and attitude measurements.*

3.1.2 Radar

Figure 3.3: *Simrad Broadband 4G radar used on Odin.*

The radar used on Odin is a Simrad Broadband 4G with range from 200 feet to 32 nautical miles and 48 RPM sweep rate (SIMRAD (2012)). The radar communicates directly with the *Radar Interface PC* as seen in Figure 3.1. Automatic Radar Plotting Aid (ARPA) (Bole et al. (2005)) is used for target tracking and analysis in the radar interface. It is assumed that the radar interface including the ARPA functionality is well tested and that the output from *Radar Interface PC* is predictable and well documented, possibly following some NMEA like protocol. At the current time the details of the transaction of this information is not yet decided. Processed target and surroundings data is transmitted via Ethernet and is assumed to at least include information about target position, heading and speed. Other possibly available data might be time and point of collision, radar cross section (or other target size information) and current noise conditions.

3.1.3 Velodyne LiDAR HDL-32E

Figure 3.4: *Velodyne LiDAR HDL-32E used on Odin for above-the-surface 3D analysis.*

For analysis of the nearby environment above the surface a LiDAR is used to create a 3D point cloud. The model used on Odin is a Velodyne HDL-32E as seen in Figure 3.4. A LiDAR can measure the distance to points around the sensor by firing a laser and measure the time it takes for the light to return. The distance is then saved along with the horizontal and vertical angle of the laser, so that the positions of the measured points can be used to generate a 3D model of the nearby environment. Velodyne HDL-32E generates 700,000 points per second with ±2cm accuracy at 80m-100m range. The LiDAR spins around the vertical axis to achieve a 360° horizontal field of view (FOV), and a combination of 32 lasers stacked vertically yields a 40° vertical FOV (+10° to −30°). An external GPS should be connected to the LiDAR for time pulse synchronization.

Uncalibrated point cloud data packets are transmitted from the LiDAR over a standard Ethernet cable using UDP. The packet format is well documented in the user manual so that they should be easy to decompose by a custom made point cloud processing unit. A calibration table must be used for vertical correction for each laser. This table is included on a CD delivered with the HDL-32E.

The LiDAR sweeps are processed by *Sceneanalys PC* as seen in Figure 3.1. The data is stored in a ROS structure called `grid_map` and sent to the motion planning module in *Autonomi PC* (Figure 3.1). Detected objects are planned to be put in ROS messages containing information about relative position, speed, heading, latitude/longitide and detection probability.

3.2 Simulation of Sensor Data from Virtual Environment

The feasibility of simulating realistic sensor data from a virtual environment will be discussed in this section, as well as complexity and benefits regarding simulation of raw versus preprocessed sensor data. Information, hardware and software needed to generate data from each sensor will also be discussed.

3.2.1 Simulating Data from GPS

As the data output of the Seapath 134 follows the well documented NMEA0183 protocol it should be feasible to generate these data based on knowledge of position, speed, heading and attitude. This information should be a result of simulating the boats motion in the virtual environment and is thereby easily accessible.

Remark: In the system architecture (Figure 3.1) of Odin the GPS is used for time synchronization of the entire system. It is assumed that the HIL setup is synchronized in time using either NTP or PTP network timing. A real or simulated GPS time synchronization signal is therefore not considered necessary in the scope of this project.

3.2.2 Simulating Data from Radar

Assuming predictable and well documented data output from *Radar Interface PC* (Figure 3.1) it should be feasible to simulate output from this subsystem given information about simulated targets around Odin. Simulating raw data from the radar, i.e simulating the exact radio signal reflected from the target, is considered too complex to stay within the scope of this project. This is because it requires too detailed information about everything from the target shape, weather conditions and the specific radar in use as well as a deep understanding of radar technology. It is thereby determined that only preprocessed data from the *Radar Interface PC* will be simulated. To simulate realistic output the following points should be considered:

- Radar blind zones minimum detection range.
- Radar range being dependent on weather conditions.
- Target detection dependent on the shape and orientation of target.
- During precipitation clutter a target may or may not be detected. A solution could be to use a randomize function to determine if a target should be detected or not. In that case, reproducibility of simulation might be more difficult.

- Accuracy of the implemented radar and how to model errors resulting from inaccuracy. Bole et al. (2005) gives great details about possible errors.

3.2.3 Simulating Data from LiDAR

The protocol of the data transfer from Velodyne HDL-32E is well documented. Given a 3D model of the surrounding environment with easy access to angle and distance calculations it should be feasible to generate a realistic point cloud. The point cloud can be represented as data packets using the HDL-32E protocol and sent over Ethernet to the interface between HIL simulator and the control system. This way it would be possible to simulate raw sensor readings from the HDL-32E. This is assumed to be beneficial to the developers of the simulated vehicle as a larger chain of HW/SW can be tested in the simulated environment. A challenge with this approach is to simulate realistic sensor readings during rainy conditions and other environmental disturbances. A solution could be to use randomized noise added to the simulated measurements. A deterministic randomize function should be used to ensure reproducibility of the simulations. 3D modeling of the surrounding environment might also be computationally challenging. More research should be done to investigate the feasibility of this.

As shown in Figure 3.1, the data from the LiDAR is processed by the *Sceneanalyse PC*. If this module is well tested, or the 3D processing of the surrounding environment is too computationally heavy, it might be more convenient to simulate the output from this PC directly without going through the steps of generating a point cloud. **The output from *Sceneanalyse PC* is what? Possibly strings with information about nearby targets? Would be easy to generate.**

Chapter 4

Interface between Simulator and Control System

This chapter will aim to decide an overall layout of the interface between the HIL Simulator and the USV control system to be tested. It will also be suggested an overview of the information flow between important modules of the simulator. Only a general layout will be suggested as the primary simulation target (Odin) is still under development and many details about the HW and SW solutions on board the USV are yet to be decided.

4.1 Overview of the Simulation Setup

The HIL Simulator should be run on a single laptop. The operating system (OS) should be Ubuntu to be able to utilize ROS functionality during simulation. The task of developing the simulator is divided in two projects- and master thesis's: one for the simulation of the surrounding environment around the USV (this project) and one for the simulation of the USV dynamics (Ødegaard (2016)). The dividing in two tasks makes it natural to split the simulation software in two main modules: Dynamics and Environment as seen in Figure 4.1. The boats attitude and movement at any time are calculated in the Dynamics module. This information should be sent to the Environment module which decides the USV's global position in the simulation environment. Weather conditions such as wind and current are decided in the Environment module and sent to the Dynamics module which calculates how these factors influences the motion of the USV. The two modules, knowing everything about the surroundings and attitude of the USV, generates appropriate sensor data and sends this to the physical interface connecting the HIL Simulator to the USV control system.

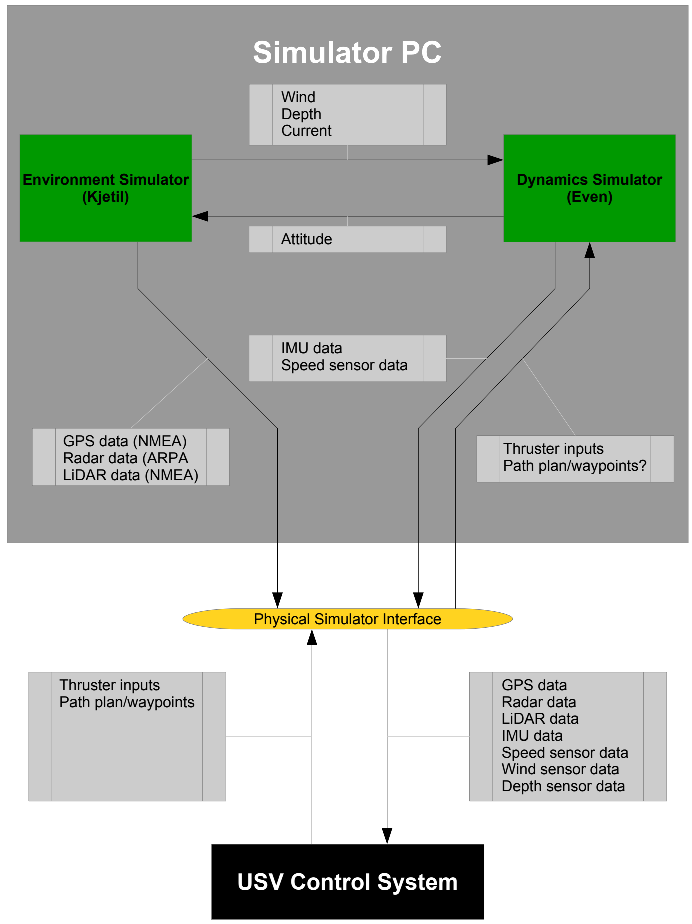

Figure 4.1: *General overview of the planned design of the HIL Simulation setup. The overview includes general information flow between the USV control system, simulator and main software modules.*

4.1.1 Information Flow Between Software Modules

4.2 Physical Interface

Too early to decide the details of this.

Chapter 5

Logging and Visualization of Simulation

5.1 Logging

5.2 Visualization

5.3 C++ example?

Chapter 6

Use of Model in Autonomous Testing Environment (ROS)

Chapter 7

Other Agents and their Behavior as Part of Simulated Environment

7.1 Possible Agents

7.1.1 Ships

7.1.2 Boats

7.2 Pros and Cons of Agents Behavior

7.2.1 Reactivity?

7.2.2 Predictability

7.2.3 Possibility of Repeating Scenario

Chapter 8

Summary and Recommendations for Further Work

8.1 Summary and Conclusions

8.2 Discussion

8.3 Recommendations for Further Work

Appendix A

Acronyms

FTA Fault tree analysis

MTTF Mean time to failure

RAMS Reliability, availability, maintainability, and safety

Appendix B

Additional Information

This is an example of an Appendix. You can write an Appendix in the same way as a chapter, with sections, subsections, and so on.

B.1 Introduction

B.1.1 More Details

Bibliography

Bole, A., Dineley, B., and Wall, A. (2005). *Radar and ARPA Manual.* Elsevier Butterworth-Heinemann, 2 edition.

Johansen, T. A., Fossen, T. I., and Vik, B. (2005). Hardware-in-the-loop testing of dp systems. Technical report, Marine Cybernetics, Norwegian University of Science and Technology, Trondheim, Norway.

Kongsberg Seatex (2016). *Seapath 130 Datasheet.*

Ødegaard, E. (2016). Specification and first prototype of simulated dynamics of autonomous usv. Master's thesis, Norwegian University of Science and Technology (NTNU).

Perez, T., Smogeli, Ø., Fossen, T., and Sørensen, A. (2006). An overview of the marine systems simulator (mss): A simulink toolbox for marine control systems. Technical report, Norwegian University of Science and Technology (NTNU), Trondheim, Norway.

SIMRAD (2012). *Simrad 4G Brochure.* SIMRAD.

SiRF (2005). *NMEA Reference Manual.* SiRF Technologies, Inc.