A Virtual Environment for Modelling, Planning and Simulation of Autonomous Vehicles

Volker Zerbe, Mike Bott, Torsten Radtke, Horst Salzwedel
Faculty of Computer Science and Automation
Ilmenau Technical University
98684 Ilmenau, Germany
e-mail: {zerbe,radtke}@theoinf.tu-ilmenau.de

Abstract: Today embedded and robot systems are designed at system level. With increasing complexity of systems it is necessary to move the main focus of the development to the early phase of the system design process (system design at mission level). In the specification of the system not only functional, architectural, and environmental descriptions but also operational requirements are included. We developed a distributed mission level design framework which allows modelling, planning and simulation of autonomous underwater vehicles in a virtual environment. At the same time we can test the implementation in a 3-D virtual world.

KEYWORDS: Modeling, Simulation, Virtual Environment, Autonomous Vehicles

1 INTRODUCTION

AUVs (autonomous underwater vehicles) [3] are unmanned underwater vehicles, which operate autonomously. In opposition to ROVs (remote operating vehicles) they do not have a physical link to a control station and contain their own power supply. At unforeseen situations, they have to be able to react, and eventually redesign a planned mission. To react under extreme environmental situations, in total isolation, and without the possibility of human interaction is similar to unmanned space missions. AUVs create new application fields in science and industry (e.g. inspection of underwater telecommunication cables, exploration of the seabed).

The usage of simulations and the possibilities of computer graphics and virtual reality makes it possible to validate and verify the design and reduce tests of completed devices at the integration phase of the design processes. A concept of a distributed simulation environment is presented. It consists of a 3-D visualisation client and a simulation server [1].

2 MISSION LEVEL DESIGN

With increasing complexity of systems it is necessary to move the main focus of the development to the early phase of the system design process. System level design is achieved by a consequent formalisation of the whole design process. The goal is to create a model of the whole system which consists of functional model, environmental model, and architectural model [2].

To create a design with the right behaviour of the system under all operational conditions, they must be included as operational requirements in the validation process. The definition of possible operational conditions can be made by of use cases (missions). They serve as test cases for the implementation of the system.

If the design of system is made with the help of operational requirements through use cases or scenarios, we talk about system design at mission level. At this level all components and their interaction in a scenario is tested. This means that defined scenarios are used for the validation of the specification and the design process.

The task of an AUV is to perform a predefined mission in a real environment (an example environment used for test purposes is depicted in Figure 1), e.g. to go from one location to another, or to perform a certain action. For this simulation it is necessary to know the behaviour of the AUV and all other objects.

Figure 1: Overview of the firth of Kiel

In case of the AUV, the simulation server [4] calculates position and orientation from required depth, course, and number of revolutions of the propeller (Figure 2).

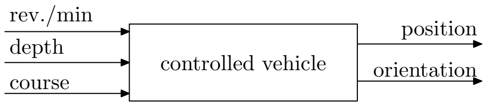

Figure 2: Block diagram of the AUV model

3 ARCHITECTURE

The requirements to a software system are the following:

- modelling and simulation of manually controlled underwater vehicles and obstacles (obstacles can be created dependent on a special situation or place),
- visualization of all simulated objects in a virtual environment,
- integration of seamaps into the system to create a visualization as close as possible to reality,
- recording of simulation results to acquire and evaluate the human behaviour of collision avoidance, and
- a flexible communication interface to enable the use of the virtual environment by other simulators.

The structure of the distributed simulation system depicted in Figure 3 originates from the above requirements.

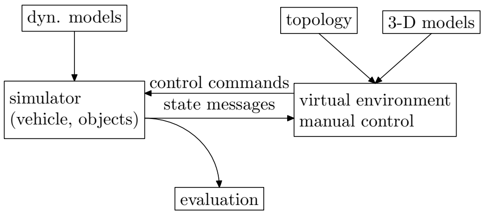

Figure 3: Structure of the simulation system

The task of the simulation server is to compute the next state of all existing objects based on their modelled dynamics. In case of a manually controlled AUV the server uses the inputs of a user. For this purpose a visual feedback is necessary. Therefore the manual control (and the acquisition of the target values) is integrated into the visualization client.

The visualization client acquires the user's target values and sends these data to the simulation server. It recomputes the actual states of all simulated objects and broadcasts the new states to all (visualization) clients.

4 VISUALIZATION CLIENT

The visualization client (Figure 4) manages the following tasks:

- reception of state data of all simulated objects and their visualization in the virtual environment, and
- evaluation of user inputs, transformation into target values and transmission to the simulation server.

The appearance of the virtual environment is characterized by a number of objects which are located in this virtual environment. An object is described by a set of properties like global position and shape as well as visual properties (e.g. color, texture, transparency).

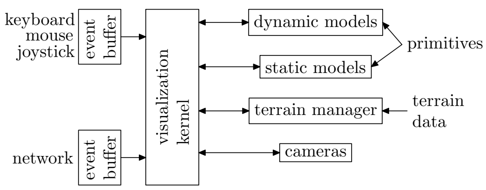

Figure 4: Structure of the visualization client

The visualization client distinguishes between *static* and *dynamic* objects. Dynamic objects are created because of messages from the simulation server, thus their number is indefinite. Their internal state change due to actualised state messages from the simulation server. Static objects (landscape, trees) are created when the virtual environment is initialized, their internal state does not change at simulation time.

Although landscape data belong to static objects, they are managed separately from other static objects. Landscape data can become large dependent on the mission area – often too large to be kept in memory or visualize it as a whole. For this reason, landscape data are divided and managed in so-called terrain tiles. Only those tiles are kept in memory which are visible in the current field of view.

The visualization of the virtual environment is done from freely adjustable cameras. These cameras can be positioned at certain positions and can have various properties.

OpenGL, an industry standard, is the foundation of the visualization. The main reason for this is it's the well structured and documented programming interface.

4.1 3-D Objects

In order to use 3-D objects in the visualization client they have to be created. Objects were created using 3D-Studio MAX, whereby orientation and position are transfered to the visualization client.

To secure a "smooth" visualization, the number of polygons of each objects should be as small as possible. The number of polygons can be reduced at construction time (approximation of non-planar surfaces by different numbers of triangles) or later by a separate tool.

To import constructed 3-D models a simple file exchange format has to be found. The file format of DirectX from Microsoft is well documented and text based, so a loading routine can be implemented easily.

4.2 Terrain

Seamaps are the source for the landscape data. Starting point for terrain modelling is the mapping of an equidistant grid (e.g., grid distance is 100m) to a certain seamap. Grid lines are along longitude and latitude lines. At each intersection altitude is sampled, resulting in a matrix of altitude values.

The number of height data depend on the operational area of the AUV and can become very large. Hence preprocessing is necessary to achieve visualization in realtime.

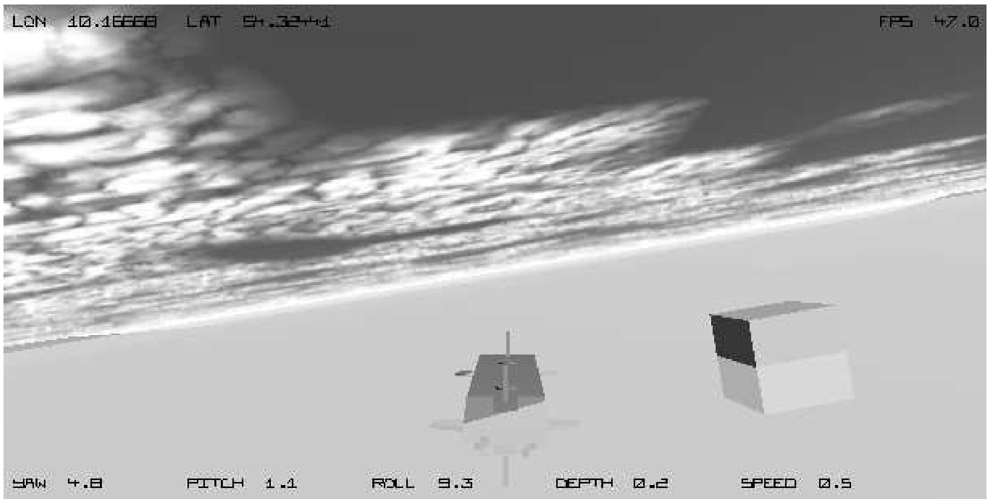

Figure 5: AUV in 3-D virtual environment

Terrain data are divided into tiles of a finite length of 30 arc seconds to the east and south. Each tile is stored in a separate file which is named by the coordinates of the tile reference point.

4.3 3-D Background

Generally, a most realistic and spectacular visualization is not necessary for scientific and technical applications, but presentations require a certain amount of realism.

A realistic sky can be constructed by different techniques:

- a textured plane is placed in the scene at a certain height,
- a textured half-sphere is placed above the visitor, or
- a textured box is placed above the visitor (sky box).

In the two latter cases it is important to move box or sphere together with the visitor. A disadvantage of the sky box method is the enormous consumption of texture memory. High texture resolutions are necessary to avoid artefacts at high screen resolutions. Today's graphic cards have a limited memory (AGP cards allow the swapping of graphic memory to the main memory and vice versa at a lower performance). The result is a box without floor which is textured with a map of 512 x 256 texels.

5 CONCLUSION

The distribibuted mission level design framework allows the planning of missions in a 3-D virtual environment. With a connected joystick the collision avoidance behaviour of a manually controlled AUV can be recorded. These parameters are important input data for the development of an intelligent and knowledge based guidance control which itself will be developed with the help of the framework. Due to the modelling of the real world in a virtual environment, the simulation results in a visualization of simulated objects (Figure 5).

To expand the capabilities of the visualization, various sensors (e.g. a sonar) will be integrated. With sonar data generated out of the virtual environment it is possible to test the control algorithms more accurately.

REFERENCES

[1] M. Bott. *Konzeption einer verteilten Simulationsumgebung für autonome Unterwasserfahrzeuge*. Diplomarbeit, Technische Universität Ilmenau, 1999.

[2] U. Freund. *Integrierte Simulationstechniken auf Systemebene*. Dissertation, Technische Universität Ilmenau, 1999.

[3] T. I. Fossen. *Guidance and Control of Ocean Vehicles*. Wiley & Sons Ltd. Chichester, 1994.

[4] J. Wernstedt, et al. *Abschlussbericht: Manuelle Führung von Unterwasserfahrzeugen in virtuellen Welten*. Technische Universität Ilmenau, 1999.