BEHAVIOR TREE BASED KNOWLEDGE REASONING FOR INTELLIGENT VESSELS IN MARITIME TRAFFIC SIMULATIONS

Volker Gollücke
Daniel Lange
Axel Hahn
Department of Computer Science
CvO University of Oldenburg
26131, Oldenburg, Germany
E-mail: volker.golluecke@uni-oldenburg.de

Sören Schweigert

Cooperative Mobile Systems
OFFIS e.V.
26131, Oldenburg, Germany

KEYWORDS

Maritime Knowledge Representation, Behavior Modelling, Behavior Trees, Maritime Traffic Simulation, S-100

ABSTRACT

For simulation based verification and validation (V&V) of maritime system designs, the system under analysis is exposed to a variety of traffic scenarios. Usually bridge and shipping simulators do not provide intelligent behavior for the simulated ships. Instead, they use simple route following techniques, or just follow a given direction. In automated V&V scenarios, a lot of different simulation runs must be executed e.g. to test new assistance systems in various situations. To cover the needed number of important situations, an automated behavior of target ships is needed.

This paper presents a technique to configure and calculate realistic and intelligent ship behavior. Each ship has its own knowledge about the environment and uses this knowledge to decide what kind of behavior the ship shows using the Behavior Tree technique.

INTRODUCTION

Many V&V scenarios in the maritime domain need realistic ship behavior. Under different environmental conditions, different ship behavior is required. As in other domains, assistance systems as well as their functional safety are becoming increasingly important.

For verification and validation in engineering assistance systems, German maritime industry and research institutes have launched a test bed for e-Maritime applications (eMIR – eMaritime Integrated Reference Platform) (Hahn and Noack 2016). eMIR provides, among others, services for research and industry projects to ensure the functional safety of new systems. For this purpose, eMIR is divided into a virtual simulation based platform (named HAGGIS) (Schweigert et al. 2014) as well as a physical platform (called LABSKAUS) for field testing.

In this paper, we will address the virtual simulation part of eMIR that is capable of testing new assistance systems like collision avoidance systems. Core element of HAGGIS is a maritime traffic simulation (MTS) (Hahn 2015). For engineering, complex systems, validation and verification we will make use of realistic simulated system environments. For this purpose, HAGGIS and in particular the MTS offer a framework to create a variety of scenarios to be simulated with the system under test. While maritime simulation systems for bridge crew training usually use a simple route following approach for target ships, V&V scenarios need target ships that behave realistically and adopt to 'relevant' parts of the environment for test automation. In terms of realism, the distinct behavior (amongst target ships) such as evasive or overtaking maneuvers, can be simulated. Therefore, a structured way of configuring and simulating target ships' behavior is required.

A target ship's environment e.g. comprises of dynamic terrain, with its bathimetry, currents and waves in water and air, as well as movable or static objects, such as other ships, landmass and wind farms. Maritime safety information is issued from shore based systems. Additionally, regularitions, e.g. anti-collision regulations (ColRegs (Organization 2003)) have to be followed.

This paper proposes to simulate each vessel behavior by an agent which uses an extended Behavior Tree technique (Ogren 2012; Colledanchise et al. 2016).

In the following, an overview on behavior modelling for V&V of maritime systems is given and requirements for behavior simulation are described. Then, we propose the concept of applying Behavior Trees and usage of ships own belief about the environment. An application example in a ship's overtaking condition is covered for evaluation of the concept.

CURRENT STATE OF BEHAVIOR MODELING AND SIMULATION IN THE MARITIME DOMAIN

Like mentioned in the introduction, for many traffic simulations, it is often sufficient to specify the course or a route to be followed. In addition to these simulators, there are other scientific approaches for describing the behavior of target ships.

Köse et al. (Köse et al. 2003) presented the simulation of maritime transport in Istanbul's Strait. Their simulator contains several assumptions. The ship's time of arrival is evenly distributed, it is forbidden to overtake each other

Proceedings 32nd European Conference on Modelling and Simulation ©ECMS Lars Nolle, Alexandra Burger, Christoph Tholen, Jens Werner, Jens Wellhausen (Editors)
ISBN: 978-0-9932440-6-3/ ISBN: 978-0-9932440-7-0 (CD)

and the intended speed for the ship in the strait is fixed at 10 knots.

Fan and Cao (Fan and Cao 2000) presented a model that calculates the throughput of a waterway from the average ship size, the average ship speed, the average separation distance between ships and the probability that each type of ship appears in the waterway. While different parts of a waterway may have different average speeds due to factors like the physical environment or other. The average speed of vessels based on the entire waterway may only be a good estimate for calculating capacity of small sea areas.

The SMARTS (Ship-with-a-captain MARine Traffic Simulation System) of Osaka University is a multi-agent model for the simulation of shipping traffic in the Bay of Osaka and the Bay of Tokyo (Japan) (Hasegawa et al. 2001). It can automatically generate a maritime traffic flow based on statistical data to control the simulation runs. Hasegawa et al., the inventors of the system, used a fuzzy expert system to navigate ships through the waterways.

In addition to these examples from the maritime domain, two newer technologies for describing behavior in the domain of artificial intelligence have emerged in recent years. On the one hand, the Utility AI concept ((Rabin 2013),p. 113ff) which carries out the selection of behavioral patterns via an evaluation function and the Behavior Tree concept, which decides which behavioral pattern is carried out by means of conditions within a tree hierarchy. Since Behavior Trees were developed in the commercial sector (created by the company BUNGIE for the development of HALO 2 (Isla 2005)) and were subsequently adapted by many developers for their own purposes, there is no industrial standard. Due to the flexible possibilities for behavioral modeling, Behavior Trees have already been scientifically investigated and partially defined in different publications (Colledanchise et al. 2016; Ogren 2012).

Both approaches cover different objectives. While the AI Utility concept uses the scoring function to offer a bigger variance in the choice of behavioral patterns, the Behavior Tree concept offers a design option that is easier to understand for non-technicians.

REQUIREMENTS FOR SIMULATION BASED V&V

From the given motivation, different requirements for a V&V simulation system can be derived.

The most obvious requirement is for accelerated execution of simulation runs to be able to create a meaningful coverage of simulated state spaces, required by the V&V methods in reasonable time.

Related to this, there is the demand for a high coverage of the simulated state space by different test vectors. This means that a large number of different scenarios must first be created and then simulated. Classical simulations in the maritime domain are designed for interaction with a human user, who defines the exact scenarios and, if necessary, adjusts the scenario during runtime of the simulation. This can be, for example, manually changing the trajectory of a ship by which it reacts to the behavior of the system under investigation. While this works out fine for training of bridge crews, it is not feasible for automatic verification and validation of new assistance systems, as it limits the number of executed simulations.

To summerize, simulation based V&V requires the simulations to behave as realistically as possible, adapting dynamically to the behavior of the system under test and under consideration of the applicable regulations (e. g. collision prevention rules and maritime traffic regulations). Agent based approaches have been proven to fulfil this requirement. However, within a simulated environment, it can be assumed that all necessary information is available to an agent. This is not the case in reality. Therefore, it has to be possible to filter the existing knowledge to restrict the knowledge of the simulated agents. This initially refers to the declarative knowledge of the agent, but also holds for the methodical knowledge of the agent. An example of this is forgetting or ignoring traffic rules. However, this is not further covered in this paper.

With regard to the modelling and simulation of vessel behavior in a maritime traffic simulation we found some additional requirements to be considered.

Based on the investigation of the collision prevention rules (Organization 2003) and the work "Distribution of debts in case of ship collisions" (Bierwirth 2004),"Collisions and their Causes" (Cahill and Britain) 2002) as well as "Managing Collision Avoidance at Sea: A Practical Guide" (Lee and Parker 2007) we derived three basic behavioral aspects that need to be mapped within a behavioral component.

1.) **General driving behavior**
    Describes the behavior of the vessel with regard to preferences such as efficiency and forward-looking driving.
2.) **Goal achievement behavior**
    Describes the behavior of the goal achievement of set target coordinates. Examples would be a frequent correction of the course.
3.) **Collision avoidance behavior**
    Describes the the ship's rules for collision avoidance. The ship could thus assess, if it is necessary to keep the vehicle in the event of an imminent collision and plan and carry out necessary overtaking maneuvers independently.

These behavioral aspects have in common that they can be expressed with a set of rules, which decide, based on information from the surrounding environment, what kind of actions are executed next. Also, these behavioral aspects may be composed of a series of simpler tasks. The overtaking task, as part of the collision avoidance behavior for example consists of: detecting the demand for overtaking, the overtaking manoeuvre itself and to trace back to the original route. During each of these partial tasks, additional task such as general collision prevention must be considered. This leads to the requirement that different behavior tasks need be combined and executed in conjunction while maintaining a hierarchical, defined order.

As a maritime traffic simulation consist of many vessels and while they should behave realistically, they also should show differences when handling a situation like overtaking another ship. Some captains target safety while other target efficiency, in terms of the time to reach their goal. Therefore, another requirement is to have the possibility to give different behavior aspects different weights while designing and executing them.

In addition, the possible large number of vessels leads to the need to setup different behaviors or different weights as easily as possible, for example by reusing previously created behaviors.

Within the next sections, we will present an approach that fulfils those requirements. First, we introduce our maritime simulation system MTS and how it represents the available knowledge about the environment, and how each individual agent build up its internal believes about this environment. Later, we will present, how Behavior Trees can be used to represent the procedural knowledge required to execute intelligent behavior and how they take advantage of the chosen environment representation.

KNOWLEDGE REPRESENTATION FOR MARITIME SIMULATIONS

The Maritime Traffic Simulation (MTS) as part of the virtual testbed HAGGIS, consists of two main components (see Figure 1). Those components are the World component, which represents the descriptive knowledge about the simulated surroundings and a set of vessel agents, where each agent represents one vessel inside the simulation in terms of a multi-agent simulation.

The *World* component can be seen as the ground truth of the simulated scenario that keeps all the static and dynamic knowledge about the environment. Thereby, the *World* is represented using the new hydrodynamic standard S-100, developed by the international Hydrographic Organization (IHO) in 2010 (IHO 2015). This generic standard defines a way to describe (maritime) feature types and information types in a structured and interoperable manner. Features include for example the knowledge from sea charts for static objects, water depths and traffic areas and rules for a specific sea area. Other features expressed with means of a S-100 conform standard are for example weather data and forecasts or maritime safety information (MSI) for nautical specialists. Those MSI in turn indicate among others, floating obstacles or malfunctioning of nautical equipment. Information, expressed as information types, provide additional information for features, like data quality or meta data, providing information about the data capturing process

As a standard with a very strong geographical reference, the IHO S-100 Standard is based on the ISO 19000 standard series and specializes, the product specifications from ISO 19131. Each product specification represents its own standard. For example, the nautical charts are standardized by the Product Specification S-101 (Electronic Navigational Charts), whereas the Standard S-124 will represent important navigational warnings (MSI - Maritime Safety Information) in the near future.

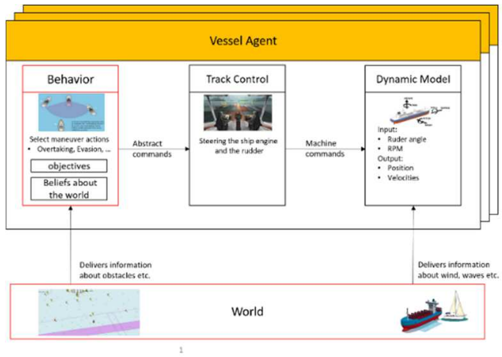

Figure 1: Common structure of vessels agents in HAGGIS

In addition to the geographical reference, the S-100 standard was also developed with the aim of interoperability between various standards in the same family. This includes a global registry in which all standard-compliant data types, associations and attributes (A) can be stored. The specialty about the S-100 is that the data types stored in the registry and especially the attributes and associations stored there can be reused in other standards of the S-100 family. This reuse of standard parts is intended to ensure that the vocabulary and the corresponding semantics are harmonized across the board. In concrete terms, this means that when two standards, as part of the description of a ship, refer to a MMSI (Maritime Mobile Service Identity), the meaning of MMSI is the same in both standards.

To express the knowledge represented by the S-100 standard, we use the following notation:

$$PS = [FT, IT, A] \quad (1)$$

Where FT represents an S100 Featuretype, IT contains the available information types and A presents a set of attributes that can be used to characterize the feature and information types.

$$A = [n, ft] \mid ft \in FT \ \vee \ ft \in IT \quad (2)$$

Those attributes can be described by their name (n) and a given type (ft). Actually, within the S-100 standard, they do contain additional information like multiplicities or descriptions, however those information are not needed for the presented approach.

Using the Product Specification (PS), we can describe the representation of the knowledge as the following tuple:

$$K^{G|A} = [F, I, AB] \quad (3)$$

Where K^G represents the global knowledge, available within the simulation system and thus the Ground truth and K^A represents the knowledge of an agent. Within this knowledge representation, set F represents the set of available features and I the set of available, additional, information described by an information type. Thereby features and information share the same representation and as we tend to mainly use features within the simulation we will combine both in the following sections.

$$F = [n, p, ft, ab] \mid p \in F \wedge ft \in FT \quad (4)$$
$$ab \in AB = [a, v] \mid a \in A \vee a = f(ab_i, ab_j) \quad (5)$$

Within the S-100 each feature and thus each information is buildup of a name (n), an optional parent (p), the feature type (ft) and a set of attribute bindings (ab), as described in equation 4. Using this representation and in particular by introducing the parent, the knowledge about the vessel can be seen as a tree structure itself.
The attribute binding, as shown in equation 5 is represented through a tuple of attributes (a) and their values (v). This represents the S-100 concept to reuse attributes (A) as well as their semantics at different positions. In addition to the normal attribute binding we also allow to conclude knowledge from two other attribute bindings.
The second part of the MTS is a set of vessel agents.
Each vessel is an independent component that interacts on its own with the environment and possibly with other independent agents. Nevertheless, all vessels agents follow a common structure as shown in the upper part of Figure 2. In the MTS, each agent is composed of three parts, which can be combined and configured almost arbitrarily with each other. This allows a large diversity of ships to be modelled and simulated.
These three parts are

1) The intelligent behavior, which can be compared to the captain of a ship. It uses its internal knowledge about its environment to reach its goals in an intelligent and realistic manner. We will discuss this part of the text in the following sections.
2) The so-called Track-Control, which is a bridge between the abstract, intelligent behavior as well as the technical - physical behavior of the ship and
3) the physical behavior of the ship in its environment. This includes a simulation of the ship's engines and rudders, as well as an application of the induced forces. Those are applied in combination with the physical properties of the water and other environmental factors such as wind and current.

As already mentioned, in this approach, the track control represents an abstraction layer between the intelligent behavior of the ship and the possibly complex physical interactions of the ship. It translates abstract commands, such as those given by the captain, into concrete machine and rudder actuations and is also able to distribute them to several machines and rudders if required.
Similar to a single ship, each behavior specifies a certain structure as shown in Figure 2. According to Figure 2, each ship has its own representation of its environment (K^A). That is usually a subset of the knowledge from the World Component (see equation 6).

$$K^A = (K^G \cap K^L) \cup K^S \cup K^E \quad (6)$$

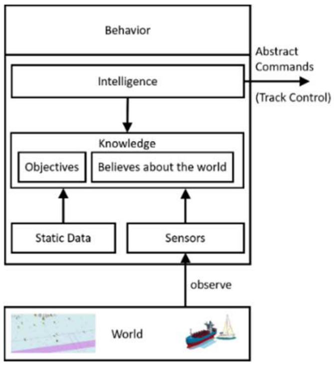

Figure 2: General structure of the behavior component

The agents knowledge is usually captured by the sensors of the agent as in the classic agent models (see (Russell and Norvig 2003)). In the maritime environment, these sensors include, in particular, radar sensors and the automatic identification system AIS, as well as sensors that monitor the internal condition of the ship. Those sensor readings are expressed through K^L. Together with the static knowledge about the ship (K^S), such as its dimensions or the knowledge from nautical charts, it represents the agent's beliefs about its current environment.

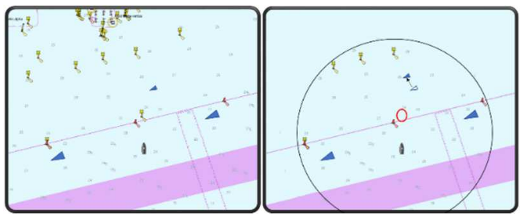

Figure 3: Ground Truth Environment (left) and believe about the current environment (right).

It is important to mention that the vessel's internal representation does not necessarily have to be a correct assumption about the surrounding, but may contain misfits (K^E) based on faulty sensor readings (as described in (Schweigert et al. 2014)) or outdated data (e. g. missing updates of the charts), and can thus deviate from the Ground Truth of the Simulation, as shown in Figure 3.
The figure displays the Ground Truth on the left side and the belief about the current environment on the right side. It shows objects like other vessels, buoys and depth information. The black circle in the figure represents the range of the belief, the red circle a missing object from the Ground Truth environment and the arrow a wrong assumption of a position of another vessel.
For the internal representation of the assumptions, the already presented S-100 based data model is also used within the vessel agent.

USING BEHAVIOR TREES AND ENVIRONMENTAL BELIEVES TO MODEL AND SIMULATE SHIP BEHAVIOR

Behavior Trees use structures in the form of directed, cycle-free graphs. Within the scope of this work, the graphical notation from the work of Ögren and Millington (Ogren 2012) are used.
Two nodes connected by edges are related to each other. A distinction is made between six types of nodes. If it is a node that is not a leaf, it can be of type Selector (executes its child nodes one by one until one of the children had success with its execution), Sequence (also executes its child nodes one by one but if one child node fails the following are not executed anymore), Parallel (Nodes of the type Parallel run all child nodes concurrently). If it is a leaf, it is either of type Action (executes a predefined action e.g. start overtake maneuver) or Condition (check whether a condition has been entered e.g. is a ship in front of us). The last type of a node, the Random Node Selector will be introduced in the section "Introduction of probabilistic selection strategies".
At runtime, the root of the used Behavior Tree generates a signal, also called tick, and sends it through the tree. The tick follows the specifications of the depth search and thus only traverses into depth, whereby the nodes can be given a fixed hierarchy. In the leafs, calculations are finally made and the defined behavior is executed.

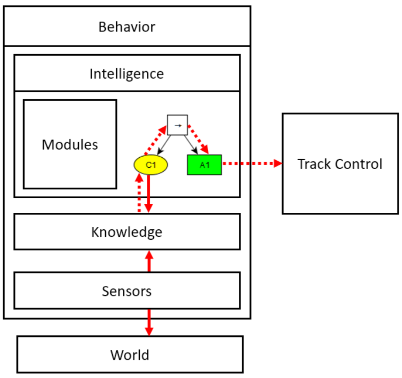

Figure 4: Behavior Tree access to ships belief

Considering the structure of the Behavior component of the Vessel Agent, Behavior Trees are the agent's rule set and are therefore classified under "Intelligence", as shown in Figure 4.
Since conditions are used to direct the flow of control, they rely on ship beliefs (solid red arrows) to make decisions based on returned information (dotted red arrows). Condition C1 uses information from the knowledge of the vessel. The information is transferred and evaluated in C1 to a decision. If the decision is negative, the subtree is canceled. Otherwise the corresponding action A1 is executed and commands are delegated to the track control component.
Using this, a Behavior Tree can be expressed using the following recursive equation 7, with $t \in$ *{Selector, Sequence, Parallel, Action, Condition, Random}* describing the type of the node, and $r \in \{Succsess, Failure, Running\}$ describing the result of the tree.

$$BT = [t, r, BT] \qquad (7)$$

At this point the Behavior Tree and especially the condition can take advantage of the modular description of the used S-100 based data model and it attribute bindings, e.g. reusing attributes in features.
For this purpose, we can further specify the conditional as well as the action node types, and how they use the agents' as well as the globally available knowledge. In case the Behavior Tree represents a conditional tree (BT_C), the result is written as a function, defined over attributes available within the agent's knowledge.

$$BT_C = [t, r(f(a^A) \to \{true, false\}), \emptyset\,] \mid a^A \in A^A \qquad (8)$$

That is: Within a Feature Catalogue, we have defined the two attributes Speed Over Ground and Course over ground which are used in different types of vehicles, like vessels, airplanes, helicopters but also within floating obstacles. Since the semantic is known for those two attributes, we do not have to care, whether we are looking onto a vessel or a floating obstacle to determine if we have to avoid this obstacle but select all objects we could find in a certain radius around the vessels current position and search for those attributes.
Naturally, a realistic behavior must consider more than just the course and speed of a possible target but also its size and possibly it's mass to determine if it could result in a thread but that information can be determined the same way, without knowing the object but knowing that it is characterized with those attributes.
If the Behavior Tree on the other hand represents an action (BT_A), the tree is formalized as follows.

$$BT_A = [Action, r(f(a^G, \Delta t) \to AB^G), \emptyset\,] \mid a^G \in A^G \qquad (9)$$

With Δt being the tick's time and a^G an attribute from the global knowledge that is actually changed by the action. By modifying the global knowledge this may also affect other agents, as they do observe their surrounding and update their internal knowledge in every time step.

Modeling and simulating a collision regulation behavior

A possible application for the Behavior Trees is the modelling of the International Regulations for Preventing Collisions at Sea (ColRegs). In the following, we present a modeling approach for a simple "drive on starboard side" behavior in the form of a Behavior Tree and then add additional parts to the tree. Attention will be given to the aspects of modularity and complexity.
In general, according to the ColRegs, ships are obliged to drive on their trajectory as far to the right as possible. This procedure is an easy way to avoid collisions. To realize this behavior, it is assumed that the ship can orientate

itself along the coastline and position itself accordingly in the fairway.

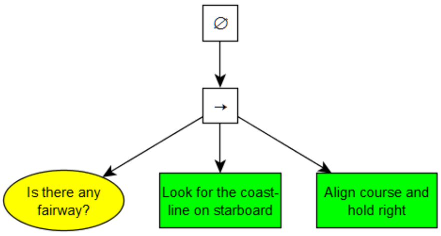

Figure 5: Behavior Tree for the “drive on starboard side” behavior

Figure 5 shows the Behavior Tree. The desired behavior is split into logical units, which can be numerically larger or smaller depending on the initial design decision. This decomposition of the behavior is already based on a divide and conquer approach during modeling and allows the developer to think in small modules and to consider sub-problems. Using our notation, we can express the condition “Is there a fairway” as follows:

$$\exists ab_f \in AB^A \wedge ab_f = [a_f, v_f] \wedge \\ a_f = [n_f, ft_f] \wedge ft_f = Fairway \qquad (10)$$

Which can be read as: There exists an attribute binding within the knowledge of the agent, whose attributes type is of type Fairway.

Since the tree first traverses into depth, it is checked if the fairway exists. If so, the condition returns a positive return value to the sequence-node and performs the first action on looking for a coastline for orientation on starboard. Once the coastline was recognized, the action returns a positive return value and the last action would start. This action is used to calculate the course based on the information collected in the last action.

Extending the Tree with additional behavioral modules

The usual driving behavior rules not only regulate the positioning in the fairway, but also describe the use of orientation aids such as buoys. Therefore, the following example will explain how Behavior Trees can be extended with additional behavioral modules.

Figure 6 shows what an extended model of the right-hand driving regulations could look like. First of all, the buoys are checked to see if they are in sight. If this is not the case, the ship orients itself at the coast and calculates the appropriate distances, as shown in Figure 5.

When buoys are in sight, the tree distinguishes between two cases. If the ship comes from the high seas, it is oriented towards the red buoys on the starboard side. If the ship is moving towards the high seas, it calculates distances to green buoys on the starboard side.

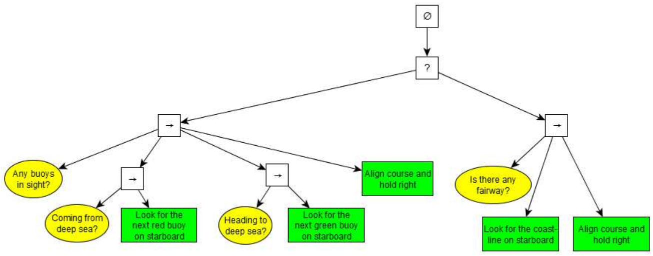

Figure 6: Navigate on starboard-side

It was shown, how simple existing rules can be supplemented by further conditions and actions without having to change the existing model. Since all ColRegs can probably be modeled and implemented as isolated rules, it should be possible to combine the behavior for arbitrary scenarios from any number and combination of ColRegs later on.

According to this introductory example, we would like to propose extensions to the Behavior Tree concept to allow more dynamic and randomness in the simulation environment. For this reason, some probabilistic selection strategies and the efficiency parameter are introduced.

Introduction of probabilistic selection strategies

For the purpose of V&V research, it is helpful to randomize behavior, to obtain a sufficient coverage of test cases. Assume a ship is supposed to overtake another ship on the left, on the right or not at all. Then it is recommended to use a "Random Node Selector" as described in (Millington and Funge 2009) and as shown in Figure 7-A. An action to be executed is selected randomly in each iteration step. However, there are also cases where some more control over the random selection is necessary. For example, if the ship should overtake on the left side with a very small probability, on the right side with a greater probability, and not at all with an even greater probability. To describe such situations, the ~? Operator, in conjunction with edge weights, is introduced as a "simple weighted random selection" (see Figure 7-B). The edge weights correspond to the probability with which the subsequent subtree is to be selected. Since the node always has to return a return value, the overall probability of all edge weights of a ~? Node is always 1.

Since the Behavior Tree iterates at regular intervals, the question arises as to how a similar situation, at the same runtime, should be treated. The vessel can now either select an action again by simple weighted random selection, or the result of the last selection can be memorized, so that the vessel reacts as before.

In order to be able to describe both variants, the "+?" Node is introduced as a "disposable weighted random selection" of the "+?" Nodes (see Figure 7-C). The "+?" Node stores the previously result and re-executes its related subtree at each iteration step.

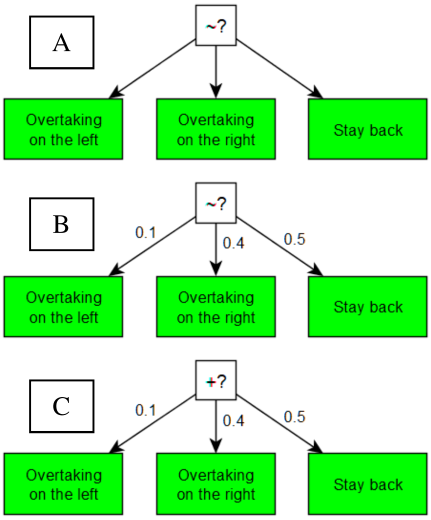

Figure 7: Different types of random selection

This additional expansion would lead to a unique behavioral profile for each vessel in the course of the simulation.

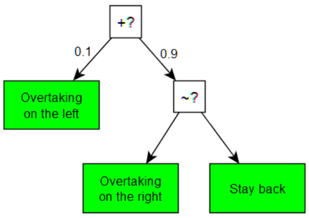

Figure 8: Combination of selection strategies

Suppose there are some ships with a behavior as shown in Figure 8. Then a small number of ships will always overtake on the left side if they encounter an obstacle and all other ships will either overtake on the right side or stay behind. These combinations allow complex dynamic behaviors to be generated even though only one tree is used. However, this will affect the readability of more complex trees.

Introduction of the efficiency parameter

As a further approach, the efficiency parameter will be introduced. It is intended to change the behavior of the ship at runtime within a predefined range. The parameter refers to the captain's driving style and indicates the tendency to have a safe, neutral, or efficient driving style. For further understanding, the overtaking behavior outlined in Figure 9 is considered.
The green vessel keeps its course and is overtaken by the blue vessel. The blue vessel is aware of the slower green one, which runs in the same direction and thus can be overtaken. It calculates an overtaking course, parallel to its own route and resumes the old course once it has gained sufficient distance to the green vessel. Applied to the overtaking behavior, the value of the efficiency parameter could affect the distance to the preceding ship, which is needed until the captain initiates an overtaking maneuver.

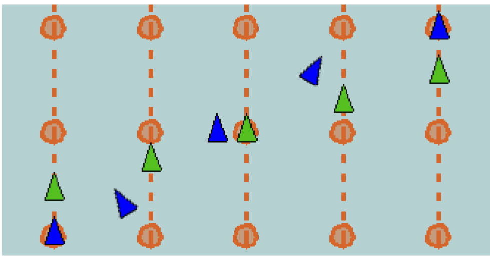

Figure 9: Different phases of overtaking with two vessels

In case of a safer driving behavior, the captain would start overtaking much earlier than a captain with efficient intentions would do. Since it depends on the application and desired behavior, in which situation a captain behaves safe or not, no general recommendations can be given here. Nevertheless, the example of the overtaking behavior is intended to show what such a parameter might look like.
The efficiency parameter is defined as the interval between [-1, 1], where -1 is the safest and passive, 0 is the neutral, and 1 the most efficient tendency. In the following example, the required distance to the preceding vehicle, so that an overtaking maneuver is initiated, is to be bound to the efficiency parameter. To achieve this, a second interval is required, on which the interval of the efficiency parameter is mapped. This additional value interval is the distance from the preceding vehicle with the minimum as the most efficient and the maximum as the safest value.

$$f(x) = \left(\frac{x+1}{2} * (\min - \max)\right) + max \qquad (11)$$

In addition, a mapping function for two intervals like shown in equation (11) is necessary, which returns a corresponding distance y for an efficiency parameter x.

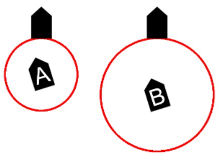

Figure 10: One vessel starting an overtaking maneuver with different efficiency parameters

Figure 10 shows a ship in two different situations A and B. In situation A, the efficiency parameter is close to 1 and the distance to the preceding ship is relatively small, as it begins to overtake. In situation B, the efficiency parameter is lower and the distance is higher. This concept is realized by defining the value intervals wherever they are used. In this case, in a condition that checks the distance to the preceding vessel using the own believe of the overtaking vessel. The efficiency parameter can be changed via an external parameter in the behavior class object. At each tick, the affected leaf node checks the parameter value and makes a corresponding decision. In this way, many ships could be equipped with the same behavior, but they would still be able to behave differently.

USE CASE

The functionality of the requirements and the efficiency parameter are shown by configuring and performing an overtaking maneuver in a maritime traffic simulation. For this purpose, a scenario with two vessels, which corresponds to Figure 9, is created in the Maritime Traffic Simulator. These are two similar vessels with the same Behavior Tree, but the ship that is passing through is faster.

Figure 11: Trajectories of an overtaking maneuver

The trajectories of the simulation are shown in Figure 11. Course t1 shows an overtaking maneuver with an efficiency parameter of -1. There are different points that can be affected by changing the efficiency parameter. Distance d1 describes the distance between the two ships before the start of the overtaking. Distance d2 describes the distance during the overtaking operation, and d3 is the distance before the overtaking maneuver is completed.
Our configuration provides identical intervals for d1 and d3, thus changing to identical values depending on the efficiency parameter. Distance d2 varies only with a very small interval. At this point, an advantage of modularity is demonstrated. By adding another node in the Behavior Tree and corresponding another class in the implementation, d1 and d3 could be separated from each other. This would make it possible to add more variability to the scenario in a simple way. Additionaly it should be mentioned that d1 and d3 might vary due to different ship sizes. Course t2 shows the same scenario with the maximum efficiency parameter of the overtaking vessel. The distances for initiating and terminating the overtaking maneuver are significantly shorter and the entire overtaking process is carried out at a shorter distance. However, the ships are getting closer, which increases the risk of collision. Since in both cases the overtaking ship first follows a route and then changes to the overtaking maneuver, it is proven, that several rules can be used and a hierarchy is followed. In this case: follow the Overtaking Behavior before the Route-Follow Behavior. The Behavior Tree can be used in several ships, and even if the configuration is identical, the efficiency parameter can be modified in a way to create different behaviors.

CONCLUSION

The attempt of modelling the overtaking scenario within the Behavior Tree concept gave an insight that the requirements towards a more realistic vessel behavior can be achieved. The strengths of the investigated approach come from the modular structure of the Behavior Tree concept, and the possibility to decide what kind of behavior is executed using the knowledge about the environment from each vessels perspective. It became visible that the S-100 based model for describing environmental knowledge and the Behavior Tree concept could be well combined. The V&V of maritime scenarios, would gain from the manifold parameterization and the ability to still maintain a non-deterministic behavior for various vessel agents. The modularity and easy access to the environmental knowledge allows the quick composition of different behaviors for many vessels. Whereas the parameter space in this attempt is constrained between safe and efficient behavior of an artificial captain.
By using global behavior-determining parameters and the possibility to set these randomly distributed, many simulated vessels can be equipped with similar Behavior Trees without these vessels behaving in the same way.
The modular structure of the Behavior Trees makes it possible to modify different behaviors. Thus behavioral building blocks, in the sense of different behavior trees that represent a sub-behavior, can be combined to model new behavior.
While Behavior Trees are easier for the designer to understand for individual behavioral aspects, they can quickly become confusing in case of larger or complicated behavior patterns. Next, we will approach the handling of large behavior patterns looking into the modelling of behavior aspects via Behavior trees and choosing the execution of these Behavior Trees via scoring functions used in the Utility AI concept.

ACKNOWLEDGMENTS

The work presented in this paper is supported by the Center for Critical System Engineering for Sociotechnical Systems at the University of Oldenburg, OFFIS and DLR. The center is funded by the Federal State of Lower Saxony, Germany under grant numbers VWZN3237 and VWZN3270.

REFERENCES

Bierwirth, Michael. 2004. “Schuldverteilung bei Schiffskollisionen.” Diplomarbeit (FH). Hochschule Bremen. https://www.hs-bremen.de/internet/hsb/projekte/maritime/studium/nautikseeverkehr/diplombachelor/diplarb_bierwirth.pdf.

Cahill, R.A., and Nautical Institute (Great Britain). 2002. *Collisions and Their Causes*. Nautical Institute. https://books.google.de/books?id=6RpFQgAACAAJ.

Colledanchise, Michele, Alejandro Marzinotto, Dimos V. Dimarogonas, and Petter Oegren. 2016. “The Advantages of Using Behavior Trees in Mult-Robot Systems.” In *ISR 2016: 47st International Symposium on Robotics; Proceedings of*, 1–8. VDE. http://ieeexplore.ieee.org/abstract/document/7558452/.

Fan, Henry SL, and Jia-Ming Cao. 2000. “Sea Space Capacity and Operation Strategy Analysis System.” *Transportation Planning and Technology* 24 (1):49–63.

Hahn, Axel. 2015. “Simulation Environment for Risk Assessment of E-Navigation Systems.” In *ASME 2015 34th International Conference on Ocean, Offshore and Arctic Engineering*, V003T02A072–V003T02A072. American Society of Mechanical Engineers. http://proceedings.asmedigitalcollection.asme.org/pdfaccess.ashx?url=/data/conferences/asmep/86128/v003t02a072-omae2015-41498.pdf.

Hahn, Axel, and Thoralf Noack. 2016. *EMartitime Integrated Reference Platform.* Deutsche Gesellschaft für Luft-und Raumfahrt-Lilienthal-Oberth eV.

Hasegawa, Kazuhiko, Go Tashiro, Seiji Kiritani, and Koji Tachikawa. 2001. “Intelligent Marine Traffic Simulator for Congested Waterways.” In *7th IEEE International Conference on Methods and Models in Automation and Robotics*, 632–636.

IHO. 2015. “S-100 – UNIVERSAL HYDROGRAPHIC DATA MODEL.” Edition 2.0.0. MONACO: International Hydrographic Organization.

Isla, Damian. 2005. “Handling Complexity in the Halo 2 AI.” In *Game Developers Conference*. http://www.gamasutra.com/gdc2005/features/20050311/isla_01.shtml.

Köse, Ercan, Ersan Başar, Emrullah Demirci, Abdulaziz Güneroğlu, and Şebnem Erkebay. 2003. “Simulation of Marine Traffic in Istanbul Strait.” *Simulation Modelling Practice and Theory* 11 (7):597–608.

Lee, G.W.U., and J. Parker. 2007. *Managing Collision Avoidance at Sea: A Practical Guide*. Nautical Institute. https://books.google.de/books?id=ZrwNNQAACAAJ.

Millington, Ian, and John David Funge. 2009. *Artificial Intelligence for Games*. 2nd ed. Burlington, MA: Morgan Kaufmann/Elsevier.

Ogren, Petter. 2012. “Increasing Modularity of UAV Control Systems Using Computer Game Behavior Trees.” In *AIAA Guidance, Navigation and Control Conference, Minneapolis, MN.*

Organization, International Maritime. 2003. *COLREG: Convention on the International Regulations for Preventing Collisions at Sea, 1972*. IMO Publication. International Maritime Organization. https://books.google.de/books?id=_ZkZAQAAIAAJ.

Rabin, Steven. 2013. *Game AI pro: Collected Wisdom of Game AI Professionals*. CRC Press.

Russell, Stuart J., and Peter Norvig. 2003. *Artificial Intelligence: A Modern Approach*. 2nd ed. Pearson Education.

Schweigert, Sören, Volker Gollücke, Axel Hahn, and André Bolles. 2014. “HAGGIS: A Modelling and Simulation Platform for E-Maritime Technology Assessment.” In , 10. Istanbul, Turkey.

AUTHOR BIOGRAPHIES

VOLKER GOLLÜCKE started to work at the University of Oldenburg in 2012. He received his PhD (Dr.-Ing.) in 2016 at the University of Oldenburg. Currently, he is working within the project CSE, where he focuses on simulative risk assessment in the maritime sector. His e-mail address is : `volker.golluecke@uni-oldenburg.de` and the homepage of his group can be found at `https://www.uni-oldenburg.de/sao/`.

SÖREN SCHWEIGERT received his PhD (Dr.-Ing.) in 2016 at the University of Oldenburg. Since 2009 he is working at the OFFIS research institude in the Transportation devision. Currently he is working in the ENABLES-S3 project, where he focuses on maritime co-simulation and scenario based V&V. His E-Mail address is: *soeren.schweigert@offis.de* and the homepage of his group is `http://www.offis.de/`.

DANIEL LANGE received his Bachelor of Science in 2017 at the University of Oldenburg and started to work at offis in the research group CMS (Cooperative mobile systems) as student research assistant. His E-Mail address is: `daniel.lange@uni-oldenburg.de`.

AXEL HAHN is full professor at the University of Oldenburg and leads the working group Systemanalysis- and Optimisation and is board member of the research institute OFFIS. His research topics are safety and efficiency in marine transportation systems. His E-Mail address is: *axel.hahn@uni-oldenburg.de* and the homepage of his group is *https://www.uni-oldenburg.de/sao/*.