Identification of Failure Modes in the Collision Avoidance System of an Autonomous Ferry using Adaptive Stress Testing

Hanna W. Hjelmeland [**] **Bjørn-Olav H. Eriksen** [***]
Ole J. Mengshoel [****] **Anastasios M. Lekkas** [†]

Norwegian University of Science and Technology, Trondheim, 7491 Norway
[**] *e-mail: hannawh@ntnu.no*
[***] *e-mail: bjorn.olav.eriksen@zeabuz.com*
[****] *e-mail: ole.j.mengshoel@ntnu.no*
[†] *e-mail: anastasios.lekkas@ntnu.no*

Abstract:
As complex autonomous systems emerge in the maritime sector, measures must be taken in order to ensure thorough safety assessment. Real world testing can be costly and potentially dangerous, and therefore there is a need for suitable simulation-based methods. This paper presents an implementation of the Adaptive Stress Testing (AST) method applied to the collision avoidance (COLAV) system of a small passenger ferry. AST is a simulation-based technique which has shown promising results in safety assessment of aviation and automobile systems. Given a simulator of a system, AST uses reinforcement learning to optimize towards system failure, and returns the most likely failure scenarios. AST is here shown to successfully identify scenarios where the criteria for failure are met, which is when the ferry collides with an adversary vessel controlled by AST. However, the majority of the initial results exhibit failures where the COLAV system of the ferry is not responsible for the failure, which makes the results less valuable to system developers. In order to improve the relevance, augmentations are made to the optimization problem. The augmentations result in four distinct problem formulations which are presented in the paper. Finally, the results are clustered using an unsupervised machine learning method called Soft Dynamic Time Warping k-means clustering in order to present a general summary of the identified failure scenarios. Our results demonstrate the relevance and potential of AST for the maritime sector, and illustrates how common drawbacks of the AST method can be circumvented by method adjustment.

Keywords: Safety, Simulation, Autonomous Vehicles, Reinforcement Learning, Adaptive Stress Testing

1. INTRODUCTION

Autonomous solutions are beginning to make an impact on the marine industry (Torben et al., 2019). The technology to enable such solutions is complex, and the safety critical nature of maritime operations makes thorough safety validation necessary. Due to the high cost of real world testing and the complexity of these systems, simulation-based methods are required as a supplement to traditional methods in order to ensure safety (Zhao et al., 2020).

Knowledge about the potential ways a system can fail is valuable but hard to obtain in real world tests, as they can inflict damage on the physical system. Adaptive Stress Testing (AST) is a simulation-based approach to finding the most likely ways that a system can fail. AST which was first proposed by Lee et al. (2015). The AST method uses reinforcement learning (RL) to gradually learn ways to make the system fail by influencing the system in simulations and obtaining reward when failure occurs. AST has shown promising results when applied to automobile systems (Koren et al., 2018), and has also contributed in the aviation industry to the approval of a new airborne collision avoidance (COLAV) system by confirming that it outperformed the existing COLAV system (Lee et al., 2020).

ThIn this work, we investigate and demonstrate the relevance of AST to the marine sector by applying it to a two-vessel crossing scenario using a simulator of the MilliAmpere autonomous ferry. The MilliAmpere ferry is a small passenger ferry developed as a part of the Autoferry project at NTNU, and it also functions as a prototype for further development of commercial autonomous ferrys by Zeabuz. We stress test the COLAV system of the MilliAmpere ferry by letting AST control the movement of an adversary vessel, and optimize for collision between the two vessels.

We provide the following contributions:

1: We demonstrate the AST method in use for a marine control system, using a similar implementation as in Koren et al. (2018). The method quickly and successfully identifies scenarios that contain collision between the ferry and the adversary. However, the majority of the initial failure scenarios are caused by the aggressive and unrealistic behaviour of the adversary, not by failures of the COLAV system of the ferry, which decreases the relevance of the results to system developers.
2: Augmentations are made to the reward function in order to obtain more relevant failure scenarios:
 · We propose an additional training heuristic to alter the behaviour of the adversary.
 · We alter the failure criteria and reward function to optimize for both improper behaviour of the ferry as well as collision between the adversary and the ferry, by adapting the method proposed in Corso et al. (2019) to the marine setting.
 · We add noise the ferry's estimate of the adversary to investigate if this can cause the ferry to behave improperly.
3: We cluster the resulting failure trajectories using Soft Dynamic Time Warping k-Means clustering, which provides a more general summary of the behavioural patterns of the adversary.

The paper is structured as follows: Section 2 provides preliminaries for the AST method, deep reinforcement learning and marine vessel modelling. Section 3 presents the simulator and the COLAV algorithm under test, our four problem variations, the method implementation and AST hyperparameters. The results are presented and analyzed in Section 4. Section 5 concludes the paper.

2. PRELIMINARIES

2.1 Adaptive Stress Testing (AST)

AST is a simulation-based approach to identify the most likely failure modes of a system under test (SUT) (Lee et al., 2015). A failure mode is a scenario where the SUT does not adhere to a certain behaviour, such as when a COLAV system does not prevent collision. The problem of bringing the system to failure is modelled as a Markov Decision Process (MDP). An MDP is formalized as a tuple (X, U, P, R) describing the behaviour of an agent $\mathcal{A}$, where

- X is the finite set of states the agent can be in, i.e., the *state space*.
- U is the finite set of actions the agent can take, i.e., the *action space*.
- $P(\boldsymbol{x}, \boldsymbol{x}') = P(X_{t+1}{=}\boldsymbol{x}' | X_t{=}\boldsymbol{x}, U_t{=}\boldsymbol{u})$ is the *transition function*, which specifies the probability of a possible next state $\boldsymbol{x}'$ given a state-action pair $(\boldsymbol{x}, \boldsymbol{u})$.
- $R(\boldsymbol{x}, \boldsymbol{x}') = R(X_t{=}\boldsymbol{x}, X_{t+1}{=}\boldsymbol{x}', U_t{=}\boldsymbol{u})$ is the *reward function*, which specifies the reward the agent obtains as a consequence of transitioning from state $\boldsymbol{x}$ to state $\boldsymbol{x}'$ due to action $\boldsymbol{u}$.

The AST optimization problem is formulated mathematically as:

$$\begin{aligned} &\max_{\boldsymbol{u}_0,\ldots,\boldsymbol{u}_{end}} \prod_{t=0}^{t_{end}-1} p(\boldsymbol{u}_t|\boldsymbol{x}_t) \\ &\text{subject to } \boldsymbol{x}_{t_{end}} \in E \end{aligned} \tag{1}$$

where E is the *event space* $E \subset X$, which is the set of failure modes, and t_{end} is the final time step of the simulation (Lee et al., 2020, p7). The reward function R of the MDP is set to optimize Equation 1. Variations of the reward function will be presented in Section 3.

2.2 Deep reinforcement learning

AST uses reinforcement learning (RL) to optimize the MDP. RL is a machine learning method where the objective is to optimize the policy $\boldsymbol{\pi}$ of the agent $\mathcal{A}$. The agent receives rewards for wanted behaviour and penalties for unwanted behaviour in simulations, and gradually updates its policy to gain more rewards. The optimal policy $\boldsymbol{\pi}^*$ is the policy that maximizes the sum of expected rewards with respect to the MDP reward function R. Different RL solvers have been proposed in the AST literature. Similar to Koren et al. (2018), we use a deep RL (DRL) solver. The DRL solver is implemented as a policy gradient method, meaning that the policy $\boldsymbol{\pi}$ is parameterized by a vector $\boldsymbol{\theta}$ and is updated via gradient ascent steps. The parameterized policy $\boldsymbol{\pi}_{\boldsymbol{\theta}}$ is implemented as a neural network where the network weights $\boldsymbol{\theta}$ are the policy parameters. The policy maps the state $\boldsymbol{x}$ to the mean of a Gaussian distribution, from which the action $\boldsymbol{u}$ is sampled. This means that at every AST step, the state is fed into the neural network, and an action is drawn from a Gaussian distribution with the output of the neural network as mean. The sampling of actions is thus done with some stochasticity, which adds an element of exploration to the method as the actions may deviate from the policy output. The policy is updated by estimating the policy-gradient from a batch of simulations using General Advantage Estimation (GAE), and Trust Region Policy Optimization (TRPO) is used to step the policy (Koren et al., 2018).

2.3 Marine vessel modelling

The simulator used in this work considers 3-degree of freedom (3-DOF) planar motion models of marine vessels. The state and velocity of the 3-DOF vessel are described by:

$$\boldsymbol{\eta} = \begin{bmatrix} x \\ y \\ \psi \end{bmatrix}, \ \boldsymbol{\nu} = \begin{bmatrix} u \\ v \\ r \end{bmatrix} \tag{2}$$

where x, y are the vessel coordinates w.r.t the earth-fixed reference frame called North East Down (NED), and ψ is the vessel heading. The vector $\boldsymbol{\nu}$ describes the velocity of the vessel, and the following relation between $\boldsymbol{\eta}$ and $\boldsymbol{\nu}$ can be obtained:

$$\dot{\boldsymbol{\eta}} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} u\cos(\psi) - v\sin(\psi) \\ u\sin(\psi) + v\cos(\psi) \\ r \end{bmatrix} = \boldsymbol{R}_{z,\psi}\nu \tag{3}$$

where $\boldsymbol{R}_{z,\psi}$ is the basic rotation about the vertical axis z of value ψ. The dynamics of the 3-DOF vessel model can be formulated as:

$$\begin{aligned} \dot{\boldsymbol{\eta}} &= \boldsymbol{R}_{z,\psi}\nu \\ \boldsymbol{M}\dot{\boldsymbol{\nu}} + \boldsymbol{C}(\boldsymbol{\nu})\boldsymbol{\nu} \ + \boldsymbol{D}(\boldsymbol{\nu})\boldsymbol{\nu} &= \boldsymbol{\tau} \end{aligned} \tag{4}$$

where $\boldsymbol{\tau}$ is the vector of forces and torque acting on the vessel, $\boldsymbol{D}(\boldsymbol{\nu})$ is the hydrodynamic damping matrix, $\boldsymbol{M}$ is the mass matrix and $\boldsymbol{C}(\boldsymbol{\nu})$ is the Coriolis and centripetal matrix (Fossen, 2011).

In order to simulate the adversarial vessel, a simpler vessel model is used based on first order processes, described by:

$$\begin{aligned} \dot{\boldsymbol{\eta}} &= \boldsymbol{R}_{z,\psi}\boldsymbol{\nu} \\ \dot{\boldsymbol{\nu}} &= -\frac{1}{\boldsymbol{T}}\boldsymbol{\tau} \end{aligned} \tag{5}$$

where $\boldsymbol{T}$ is the vector of vessel time constants and $\boldsymbol{\tau}$ is a control input.

3. METHODOLOGY

3.1 Zeabuz COLAV simulator and the The Single path velocity planner

To simulate the ferry and adversary, the Zeabuz COLAV simulator is used. This simulator is implemented in Python, allowing for direct integration with the chosen AST framework, presented in Section 3.3. The simulator is highly parameterizable, and suitable for running batch simulations at high speed.

The COLAV algorithm under test is the Single Path Velocity Planner (SP-VP), which is based on path-time decomposition (Thyri et al., 2020). In this concept, a fixed nominal path is used for transforming obstacles modeled as moving Euclidean polygons into the path-time space. By constraining the ferry to be located somewhere on the nominal path, a search tree spanning the path-time space can be constructed and used for finding possible velocity trajectories along the nominal path. Collision avoidance is assured by requiring that the edges in the search tree do not intersect with obstacles. A cost dependent on the speed and closeness to obstacles is applied to each edge in the search tree, and the optimal trajectory is found using Dijkstra's algorithm. The optimal path-time trajectory is then transformed into a time-expanded Euclidean space and tracked by the ferry using a Dynamic Positioning (DP) controller.

Being constrained to following a nominal path, the SP-VP algorithm is only able to control the velocity of the vessel along this path, making it unable to make course changes to avoid collision. Furthermore, the algorithm is parameterized to only allow for a speed within a certain range, leading to stopping being the preferred action when the ferry is faced with a collision situation.

The simulation scenario considered in the experiments is illustrated in Figure 1. It exhibits a crossing situation where the adversary (red) is passing from the starboard side of the ferry (blue). The adversary is set up to follow a line in westward direction using a simple proportional speed-heading controller, while the ferry attempts to follow a line straight ahead to reach $(x, y) = (200, 0)$ while following the instructions from the SP-VP system. Without AST intervention, the ferry is able to navigate the situation by stopping and letting the adversary pass in front, which is evident in Figure 1. However, there may be variations of this scenario in which the ferry is unable to navigate, and the goal of AST is to uncover these, as we discuss next. In the experiments, the AST agent is able to add actuation to the adversary, s.t. the adversary dynamic becomes:

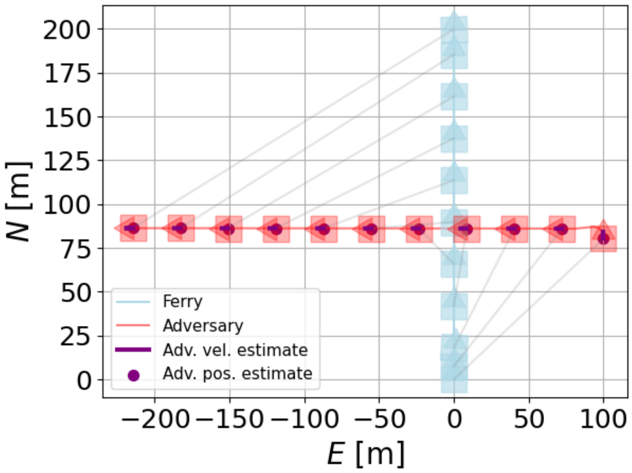

Fig. 1. Baseline experiment scenario: The ferry stops to let the adversary pass. The grey lines indicate positions for simultaneous time steps.

$$\begin{aligned} \dot{\boldsymbol{\eta}} &= \boldsymbol{R}_{z,\psi}\boldsymbol{\nu} \\ \dot{\boldsymbol{\nu}} &= -\frac{1}{\boldsymbol{T}}\boldsymbol{\tau} + \boldsymbol{u} \end{aligned} \tag{6}$$

3.2 Problem formulation variations

Building on the baseline scenario, we present four distinct problem variations. The first three include augmentations to the MDP reward function used in the AST optimization problem. In the last variation, estimation noise is introduced in the ferry's estimate of the adversary position and velocity.

Variation A: Distance heuristic In the first variation, the reward function is implemented in a similar manner to Koren et al. (2018). Three reward cases are specified:

$$R = \begin{cases} 0, & \text{if } \boldsymbol{x} \in E \\ -\alpha - \beta D, & \text{if } \boldsymbol{x} \notin E, t \geq t_{end} \\ -\log(1 + M(\boldsymbol{u}, \boldsymbol{\mu_u}|\boldsymbol{x})), & \text{if } \boldsymbol{x} \notin E, t < t_{end} \end{cases} \tag{7}$$

In the case of failure, i.e., when $\boldsymbol{x}$ is in the event space E, the RL agent obtains a zero-reward. The second case is active if a full simulation is run without failure. In that case, a large negative reward is distributed according to the user specified constants α and β, and the distance heuristic D which is the distance between the ferry and the adversary. The distance heuristic penalizes the agent additionally if the distance between the adversary and the ferry is large in the end. This guides the agent towards cases where the two vessels end up in close proximity. The last case portrays the reward distributed at every time step. The agent receives a reward using the Mahalanobis distance $M(\boldsymbol{u}, \boldsymbol{\mu_u}|\boldsymbol{x})$, which is the statistical distance between the action $\boldsymbol{u}$ and the mean action $\boldsymbol{\mu_u}$ (Mahalanobis, 1936). This part of the reward function is a heuristic measure of the probability of the AST action, which makes the AST method optimize for the most likely action sequences. The mean action $\boldsymbol{\mu_u}$ is set to zero in order to penalize high action values, as they are deemed unlikely.

Variation B: Additional training heuristic In variation B, we propose an additional training heuristic in order to make the adversary less aggressive. The value of the heuristic depends on whether or not an AST action increases the risk of collision, as risk-increasing behaviour is considered unlikely. Let r denote the measure of risk, defined as:

$$r = \begin{cases} 0, & \text{if } D > \mathrm{L} \\ \dfrac{1}{2}\dfrac{(L-D)}{L}, & \text{if } D < \mathrm{L}, |\phi| > \gamma \\ \dfrac{1}{2}\dfrac{(L-D)}{L} + \dfrac{1}{2}\dfrac{(\gamma-\phi)}{\gamma}, & \text{if } D < \mathrm{L}, |\phi| < \gamma \end{cases} \quad (8)$$

where L is a user defined look-ahead distance, i.e., the minimum value of the distance between the ferry and the adversary at which the risk measure becomes active. D is the distance between the ferry and the adversary, ϕ is the angle along the distance vector from the adversary to the ferry relative to the adversary heading and γ is a user defined adversary field of view angle relative to its heading.

The risk increases if the obstacle is close to the ferry, and it increases further if the obstacle is headed towards the ferry. The resulting reward function becomes:

$$R = \begin{cases} 0, & \text{if } \boldsymbol{x} \in E \\ -\alpha - \beta D, & \text{if } \boldsymbol{x} \notin E, t \geq t_{end} \\ -\log(1 + M(\boldsymbol{u}, \boldsymbol{\mu_u}|\boldsymbol{x})) - \zeta h(\boldsymbol{x}), & \text{if } \boldsymbol{x} \notin E, t < t_{end} \end{cases} \quad (9)$$

where ζ is a user-defined heuristic gain. The heuristic is given as:

$$h = \begin{cases} 0, & \text{if } \Delta_r \leq 0 \\ \log(1 + r_{t+1}), & \text{if } \Delta_r > 0 \end{cases} \quad (10)$$

where $\Delta_r = r_{t+1} - r_t$, making the heuristic active if the action led to increased risk. The value of the heuristic increases according to the risk after the action, causing the agent to be further penalized for high-risk maneuvers.

Variation C: Improper behaviour In variation C, the reward function is altered to optimize for improper behaviour of the ferry. As the adversary in the simulation scenario is crossing from the right, the proper behaviour of the ferry is to let the adversary pass before it continues, according to rule 15 of the Convention on the International Regulations for Preventing Collisions at Sea (COLREGS). The SP-VP system adheres to this behaviour in the scenario without AST intervention, as depicted in Figure 1. This reward augmentation is based on the work by Corso et al. (2019), where optimizing for improper behaviour showed promising results in the autonomous automobile case. The improper behaviour of the automobile was defined according to navigation rules for automobiles, analogous to the COLREGS.

We evaluate if a time step is improper based on the speed of the ferry, the proximity of the ferry to the adversary and the angle from the ferry to the adversary. Let S be true if the ferry speed U is above a maximum value U_{max}, P be true if the proximity D is less than a minimum proximity D_{min} and A be true if the angle following the vector from the ferry to the adversary relative to the ferry heading is within an angle sector $[\delta_{min}, \delta_{max}]$. We deem a time step to be improper if the following logical expression evaluates to true:

$$improper := S \wedge P \wedge A. \quad (11)$$

All of the time steps in the simulation are evaluated as proper or improper, and the fraction of improper time steps has to exceed a threshold in order for AST to deem a scenario a failure, in addition to a collision being present. The event space is thus altered to include the fraction of improper time steps f_{imp} and the improper time step threshold f_{thresh}. Let E_{imp} be the altered event space and ω be a sequence of actions, then:

$$E_{imp} = \{\omega \mid \omega \in E \wedge f_{imp}(\omega) > f_{thresh}\}. \quad (12)$$

The resulting reward function is:

$$R = \begin{cases} 0, & \text{if } \boldsymbol{x} \in E_{imp} \\ -\alpha - \beta f_{prop}, & \text{if } \boldsymbol{x} \notin E_{imp}, t \geq t_{end} \\ -\log(1 + M(\boldsymbol{u}, \boldsymbol{\mu_u}|\boldsymbol{x})), & \text{if } \boldsymbol{x} \notin E_{imp}, t < t_{end} \end{cases} \quad (13)$$

where the fraction of proper time steps, f_{prop} = 1 - f_{imp}, replaces the distance heuristic used in the other cases to optimize for improper time steps.

Variation D: Improper behaviour with estimate noise In the last variation, estimation noise is introduced in the tracking system of the ferry. This is done to examine the effect of sensory errors on the ferry behaviour. The AST actuation of the adversary is restricted to smaller values than the previous cases in order to make the adversary behave less abruptly and focus on sensory errors. The reward function and failure definition remains the same as in variation C, as we still search for improper behaviour of the ferry.

The AST agent is able to induce noise in both the position and velocity estimates of the adversary. Let the ferry's estimate of the adversary be denoted $\tilde{\boldsymbol{q}}_{\boldsymbol{a}}$, the actual measurement of the adversary $\boldsymbol{q_a}$ and estimation noise $\boldsymbol{w_n}$. The estimate is then given as:

$$\tilde{\boldsymbol{q}}_{\boldsymbol{a}} = \boldsymbol{q}_{\boldsymbol{a}} + \boldsymbol{w}_{\boldsymbol{n}} = \begin{bmatrix} x \\ y \\ \dot{x} \\ \dot{y} \end{bmatrix} + \begin{bmatrix} w_x \\ w_y \\ w_{\dot{x}} \\ w_{\dot{y}} \end{bmatrix} \tag{14}$$

where the noise is updated at every time step by the AST action vector $\boldsymbol{u}_{\boldsymbol{n}}$,

$$\boldsymbol{u}_{\boldsymbol{n}} = \begin{bmatrix} u_p \\ u_p \\ u_v \\ u_v \end{bmatrix} \tag{15}$$

such that the noise dynamics become:

$$\dot{\boldsymbol{w}}_{\boldsymbol{n}} = \frac{1}{K}\boldsymbol{w}_{\boldsymbol{n}} + \frac{1}{K}\boldsymbol{u}_{\boldsymbol{n}} \tag{16}$$

where K is a predefined constant. The added noise in position and velocity is the same in x and y to simplify the action space of the AST agent.

3.3 AST Implementation and hyperparameters

AST is implemented using the Python AST Toolbox which is developed and maintained by Stanford Intelligent Systems Lab (Koren et al., 2021). The toolbox creates a wrapper around the simulator and interfaces with it through the following functions:

- `reset(`$\boldsymbol{x}_0$`)`: reset the simulator to the fixed initial position $\boldsymbol{x}_0$
- `step_simulation(`$\boldsymbol{u}$`)`: step the simulation one step forward with action $\boldsymbol{u}$
- `is_goal()`: return true if $\boldsymbol{x} \in E$

The state $\boldsymbol{x}$ is set to consist of the state vector of the ferry and the adversary:

$$\boldsymbol{x} = \begin{bmatrix} x_f \\ y_f \\ \psi_f \\ x_a \\ y_a \\ \psi_a \end{bmatrix} \tag{17}$$

The AST wrapper steps the simulation from start to end and applies a new action for every AST step. The simulator is stepped a number of times between every AST step in order to keep the simulation smooth whilst limiting the size of the AST action space. The AST wrapper does a fixed number of steps in one *epoch*, which is user specified and referred to as the *batch size*. The resulting simulation batch is then used to train and update the DRL network. The AST hyperparameters were set according to Table 1.

In variation D, the movement of the adversary is less abrupt. This made the crossing faster and reduced the number of AST steps necessary to run a full crossing simulation from 100 to 50, as reflected in Table 1.

3.4 Clustering

The failure trajectories are clustered using soft dynamic time warping (soft-DTW) k-Means clustering in order to provide a more general overview of the behavioural patterns of the adversary. Soft-DTW was first proposed by Cuturi and Blondel (2017). The method aligns the trajectories in time before finding the Euclidean distance between them and thus it is able to identify patterns in the multidimensional trajectories even if they do not match in time. We implement a clustering method from the Python library tslearn called TimeSeriesKMeans with soft-DTW as the metric, using $k = 4$ clusters as this gives clusters that are better balanced between general and specific than other numbers of clusters.

Table 1. AST hyperparameters

Hyperparameter	Value
α	100 000
β	10 000
Batch size	20 000
Epochs	30
Collision distance threshold	10
Simulator step size	0.1 s
AST step size	4 s
Max number of AST steps	Variation A, B, C: 100, D: 50
ζ	50
γ	20°
L	50 m
U_{max}	0.2 m/s
D_{min}	30 m
δ_{min}	0 rad
δ_{max}	$\frac{\pi}{2}$ rad
f_{thresh}	2%
K	10

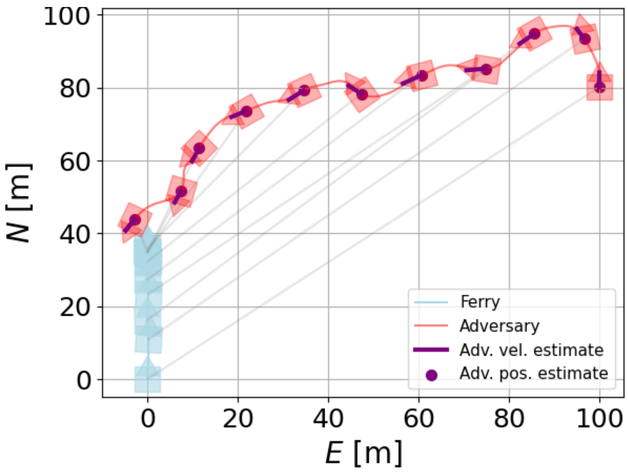

Fig. 2. Example failure mode from variation A, distance heuristic: The adversary attacks the ferry head on to provoke a collision.

4. RESULTS AND DISCUSSION

4.1 Variation A: Distance heuristic

Using the problem formulation of variation A, the AST method is able to discover many failure modes. However, they all show the adversary deliberately attacking the ferry. The SP-VP algorithm is not constructed to avoid the attack of another vessel as this is unrealistic, and thus the failures are of little concern to system developers. A failure scenario from variation A is illustrated in Figure 2.

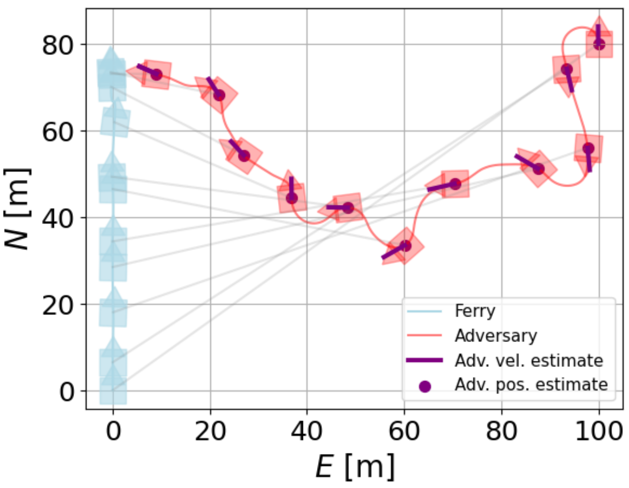

Fig. 3. Example failure mode from variation B, additional heuristic: The adversary makes detours in order to avoid the penalty of driving close and straight towards the ferry.

4.2 Variation B: Training heuristic

In the majority of the resulting failure trajectories from variation B, the adversary spends less time heading straight towards the ferry compared to the trajectories from variation A. In many of the trajectories, the adversary performs one or more detours in order to avoid the penalty of close proximity and heading straight towards the ferry, as illustrated in the failure mode shown in Figure 3. It seems, however, that the high reward of collision trumps the penalty of a little aggressive behaviour at the end, as the adversary still exhibits unrealistically aggressive behaviour in the last few time steps before collision.

4.3 Variation C: Improper behaviour

The optimization for improper behaviour of the ferry in variation C gives noticeably fewer results than the previous variations. The failure modes do indeed display scenarios where the ferry behaves improper according to our definition, as the ferry has positive speed while in close proximity of the adversary. However, most of the scenarios exhibit cases where the adversary seems to be headed in other directions until the last few time steps where it decides to cross, as illustrated in one of the failure scenarios shown in Figure 4. In this scenario, the ferry is driving forward as the adversary seem to be crossing in front of the ferry with good margin. Then the adversary turns, and the ferry continues forward as the adversary seems to be on on its way to cross astern of the ferry. In the last time steps, the adversary turns to cross ahead of the ferry. The ferry stops due to the adversary crossing, and a collision occurs. In these specific cases, a better solution for the ferry would be to continue forward to let the adversary pass astern. Hence, the scenarios does highlight a way that the SP-VP system can misunderstand the adversary trajectory. The result is still, to some extent, reassuring to the system developers due to the irrational or worst-case behavior of the adversary.

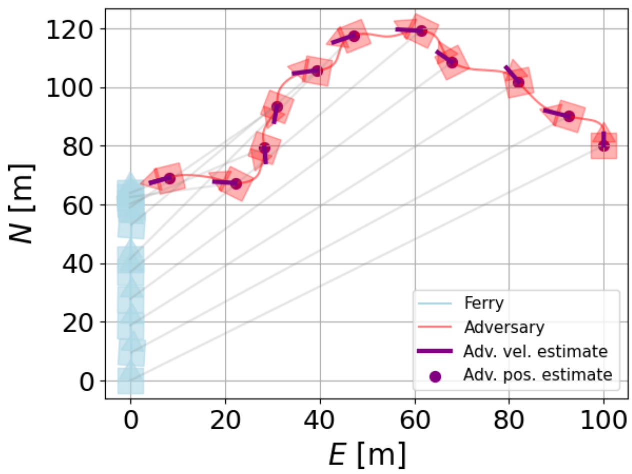

Fig. 4. Example failure mode from Variation C, improper behaviour: When the adversary misleads the ferry system by an apparent crossing astern the ferry, the ferry is instructed to continue straight ahead. Thus, the ferry has nonzero speed while in close proximity of the adversary and the behaviour is deemed improper.

4.4 Variation D: Improper behaviour with estimation noise

Adding varying noise while restricting the movement of the adversary resulted in failure modes that differs from the ones discovered in the previous variations, as illustrated in the failure scenario in Figure 5. In these failure scenarios, the adversary behaves rationally due to its restricted movement, as it attempts to cross past the ferry while complying with rule 15 of the COLREGS. The ferry proceeds to behave improper due to its inaccurate estimates of the adversary. Although the noise values are quite high and thus not necessarily probable, the results show how the SP-VP system is prone to misunderstand the trajectory of adversary vessels in cases of consistently high estimation noise. It is also evident that the system is especially impacted by inaccuracies in the velocity estimates.

4.5 Clustering

The soft-DTW clustering technique is effective for the purpose of clustering the resulting failure trajectories, as it aids the analysis process by creating an understandable overview of the patterns in the adversary behaviour. This is illustrated in Fig 6 and Fig 7, displaying the clusters from variation A and B, respectively. In these examples, the change in the behavioural pattern of the adversary due to the effect of the training heuristic from variation B is evident. In the results from variation B, the number of trajectories where the adversary attacks the ferry from north is reduced, and a new cluster arises where the adversary makes the aforementioned detour before attacking the ferry from astern.

5. CONCLUSION

This paper demonstrated the use of AST for the COLAV system of a small autonomous passenger ferry. Four variations of the AST problem formulation were presented, with results that displayed a range of different behaviors. The

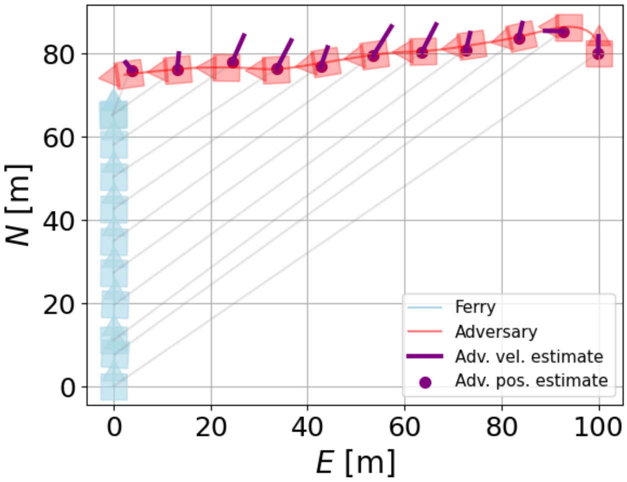

Fig. 5. Example failure mode from variation D, noisy estimates: The ferry exhibits improper behaviour when noise is added in the adversary estimates. The COLAV system is specially effected by inaccurate velocity estimates.

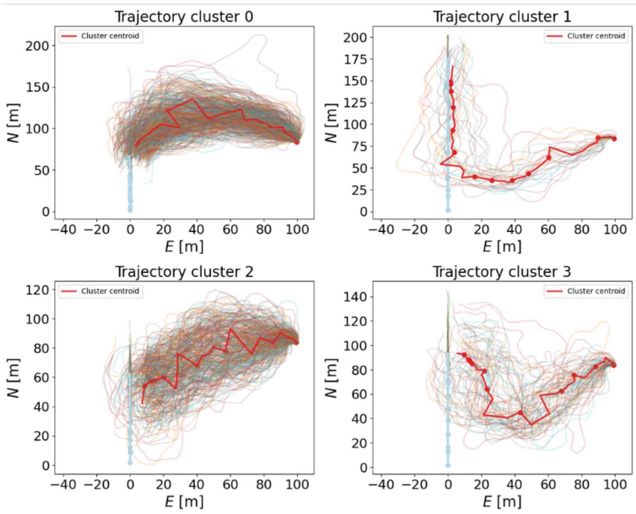

Fig. 6. Clusters of failure trajectories from variation A: All resulting trajectories are plotted. The red lines illustrate the cluster centroids.

degree of observed irrational or worst-case behaviour of the adversary changes between the different variations studied. While sensor noise needs to be set very high to induce improper behavior of the ferry, it is clear that the system is impacted by inaccurate velocity estimate. Overall, while this study is clearly not a proof that other failure modes do not exist, it builds confidence in the robustness of the SP-VP COLAV system of the ferry.

In further work, more case studies should be conducted using different scenarios and more complex COLAV systems than the SP-VP scheme. The adversary model can also be further developed into a more complex model that is more restricted in its behaviour, in order to obtain more realistic trajectories.

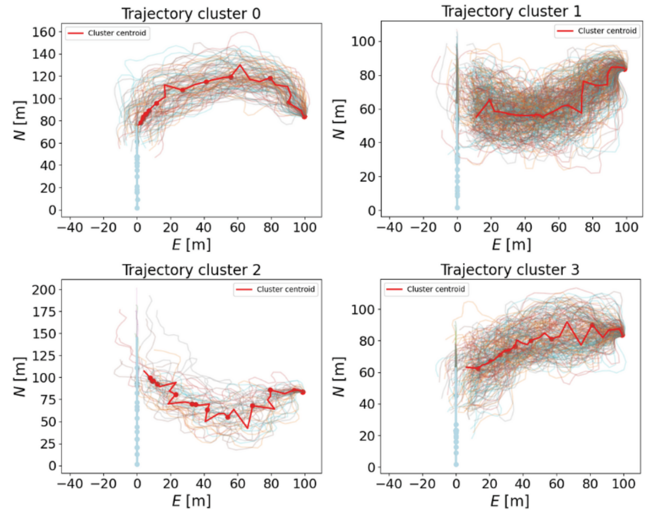

Fig. 7. Clusters of failure trajectories from variation B: The effect of the training heuristic is evident in the resulting clusters, as the most prominent cluster is the one which contains trajectories where the adversary performs one or more detours and attacks the ferry from astern.

ACKNOWLEDGEMENTS

This research could not have been conducted without the use of the Python AST Toolbox which is developed and maintained by Stanford Intelligent Systems Lab (SISL), with extra support from Ritchie Lee and Mark Koren. Resources and equipment were provided by the Department of Engineering Cybernetics at the Norwegian University of Science and Technology.

REFERENCES

Corso, A., Du, P., Driggs-Campbell, K., and Kochenderfer, M.J. (2019). Adaptive Stress Testing with Reward Augmentation for Autonomous Vehicle Validatio. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*.

Cuturi, M. and Blondel, M. (2017). Soft-DTW: a Differentiable Loss Function for Time-Series. *Proceedings of Machine Learning Research*.

Fossen, T.I. (2011). *Handbook of marine craft hydrodynamics and motion control*. Wiley, Chichester, West Sussex.

Koren, M., Alsaif, S., Lee, R., and Kochenderfer, M.J. (2018). Adaptive Stress Testing for Autonomous Vehicles. In *2018 IEEE Intelligent Vehicles Symposium (IV)*.

Koren, M., Ma, X., Corso, A., Moss, R.J., Campbell, K.D., and Kochenderfer, M.J. (2021). AST Toolbox: An Adaptive Stress Testing Framework for Validation of Autonomous Systems.

Lee, R., Kochenderfer, M.J., Mengshoel, O.J., Brat, G.P., and Owen, M.P. (2015). Adaptive stress testing of airborne collision avoidance systems. In *2015 IEEE/AIAA 34th Digital Avionics Systems Conference (DASC)*.

Lee, R., Mengshoel, O.J., Saksena, A., Gardner, R.W., Genin, D., Silbermann, J., Owen, M., and Kochenderfer, M.J. (2020). Adaptive Stress Testing: Finding Likely

Failure Events with Reinforcement Learning. *Journal of Artificial Intelligence Research*.

Mahalanobis, P.C. (1936). On the generalized distance in statistics. *Proceedings of the National Institute of Sciences (Calcutta)*.

Thyri, E.H., Breivik, M., and Lekkas, A.M. (2020). A Path-Velocity Decomposition Approach to Collision Avoidance for Autonomous Passenger Ferries in Confined Waters. *IFAC-PapersOnLine*.

Torben, T.R., Brodtkorb, A.H., and Sørensen, A.J. (2019). Control allocation for double-ended ferries with full-scale experimental results. *IFAC-PapersOnLine*.

Zhao, X., Salako, K., Strigini, L., Robu, V., and Flynn, D. (2020). Assessing safety-critical systems from operational testing: A study on autonomous vehicles. *Information and Software Technology*.