Available online at www.sciencedirect.com

ScienceDirect

IFAC Papers Online CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 55-31 (2022) 70–77

Particle Swarm Optimization for Dynamic Risk-Aware Path Following for Autonomous Ships ⋆

Simon Blindheim * **Tor Arne Johansen** *

* *Centre for Autonomous Marine Operations and Systems, Department of Engineering Cybernetics, Norwegian University of Science and Technology (NTNU), 7491 Trondheim, Norway*

Abstract: This work presents the use of particle swarm optimization (PSO) for dynamic risk-aware path following, or waypoint re-planning during autonomous surface navigation in a maritime environment with polygonal grounding obstacles. Although recent research on control and local or global path planning for maritime autonomous surface ships (MASS) is considerable, few deal with the concept of risk during dynamic path following along a pre-planned path. The proposed method introduces risk-based terms in the PSO fitness function for dynamic (online) adjustment or re-planning of intermediate waypoints during path following of a pre-planned route, where the control of the vessel in this work is left to a standard line-of-sight PID controller. Moreover, the results are compared to the performance of an analogous implementation of risk-aware model predictive control (MPC) using a gradient-based solver. The suggested method yields adequate performance similar to that of the MPC algorithm. Ultimately, the PSO approach may in future works allow for more complex and dynamic risk-aware behavior related to e.g. weather conditions implemented through lookup tables, or discrete machinery modes that are non-smooth.

Keywords: Autonomous navigation systems, ANS, autonomous surface vessels, ASV, bathymetry, decision-making, dynamic path planning, electronic navigational charts, ENC, line of sight guidance, maritime autonomous surface ships, MASS, maritime systems, online optimization, optimal control, particle swarm optimization, PSO, risk-aware control.

1. INTRODUCTION

In this work, a standard path following problem for maritime autonomous surface ships (MASS) is formulated, given a pre-planned route in a maritime environment with several isles or islands nearby. This is considered a common MASS control problem of moderate difficulty, with high safety requirements. Moreover, due to increasing complexity of various additional operating modes and emergent behaviors in larger composite systems, the act of formulating, handling and optimizing more complex cost functions accommodating the safety requirements of MASS should be advanced accordingly. Thus, this work presents an approach which allows future works to mix and incorporate discrete and non-smooth cost objectives together with smooth penalties or optimization variables to be provided to the solver, which may make autonomous navigation systems (ANS) more risk-aware during online guidance or control in uncertain or changing environments.

1.1 Literature review

Path planning and obstacle avoidance for MASS remains a challenging problem (Vagale et al., 2021a) (Vagale et al., 2021b). There exists a wide range of methods for long-distance (global) autonomous path planning such as the A*-algorithm (Shah and Gupta, 2016) (Song et al., 2019), Voronoi diagrams (Candeloro et al., 2017) or hybrid trajectory planning (Bitar et al., 2020) (Martinsen et al., 2021). Moreover, research on adaptive or dynamic path planning for autonomous surface vessels (ASV) or MASS including environmental disturbances (i.e. wind, waves and currents) have accelerated in recent years, such as energy optimization and path smoothing with respect to sea currents (Niu et al., 2018) (Ma et al., 2018), mixed-integer linear programming (Vitus et al., 2008), A* and repulsive vector fields (Song et al., 2017), harmonic potential fields (Shi et al., 2007) for obstacle avoidance and increased safety (Singh et al., 2018), as well as grounding safety domains (Bakdi et al., 2020) and COLREG-compliant collision avoidance for safe autonomous navigation (Zhou et al., 2020) (Zaccone, 2021).

However, the notion of *safe navigation* noted in the literature is largely dependent on the definition of inherent system *risks*, and is still a somewhat ambiguous concept with respect to quantitative path planning and optimal control in an environment with grounding hazards. It is

⋆ This work was supported by the Online risk management and risk control for autonomous ships (ORCAS) project funded by the Research Council of Norway (grant number 280655), Kongsberg Maritime, DNV, and Centre for Autonomous Marine Operations and Systems (AMOS) at NTNU funded by the Research Council of Norway (grant number 223254).

10.1016/j.ifacol.2022.10.411

suggested that risk perception should largely be based on system uncertainty or lack of knowledge, rather than probabilities (Utne et al., 2017). Thus, several recent works utilize risk-based or *risk-aware* approaches for autonomous navigation, such as risk-based model predictive control (MPC) for anti-grounding (Blindheim et al., 2020), rapidly-exploring random trees (RRT) (Enevoldsen and Galeazzi, 2021) for grounding-aware collision avoidance, and verification of COLREG-compliant vessel behavior for collision avoidance (Dallolio et al., 2022) (Kufoalor et al., 2020). There is however much potential left to be explored within the domain of safe navigation, with respect to the utilization of risk within (dynamic) path planning and optimal control.

Significant research efforts on PSO for safe autonomous maritime navigation has emerged in recent years. In this paper, however, the scope of the risk-aware PSO approach is limited to grounding or allision avoidance only, for the purpose of a proof-of-concept study. Nevertheless, it is argued that the approach discussed with respect to risk-based anti-grounding in general may be extended to additionally consider collision avoidance of dynamic obstacles (e.g. nearby vessels in motion) in future research.

Particle swarm optimization (PSO) has since its initial introduction in 1995 loosely based on the social flock behavior of birds, evolved into a wide array of algorithm variations and methodologies (Poli et al., 2007). Due to the computational nature of the PSO method, the cost or fitness function provided to the algorithm may in general take any form. This is most notably in contrast to the gradient-based MPC method which – while computationally fast for feasible problems – requires a sufficiently smooth cost function to achieve satisfactory convergence rates and robust performance. Thus, non-smooth and mixed-integer cost or fitness functions may be utilized in the PSO algorithms, enabling a larger and more complex range of optimization objectives both related to resource consumption and formulation of risk or safety concerns.

Within this particular scope, similar recent works on standard or extended PSO methods for robot path planning or MASS navigation with respect to static obstacles include the following:

- **a)** Adaptive PSO for distance-based path planning toward an end target while avoiding static obstacles (Dewang et al., 2018)
- **b)** PSO path planning using distances between the start and target location as well as intermediate waypoints, including static obstacle avoidance (Alam et al., 2020)
- **c)** Energy-efficient PSO path planning in which the energy consumption with respect to environmental forces is weighted against the path length (Ding et al., 2018)
- **d)** PSO combined with the sine cosine algorithm for safe MASS navigation using a cost function which includes the terms path length, grounding risk, bathymetry (depths), and path smoothness (Xue, 2022)
- **e)** Two-part global path planning and multi-objective path following using a cost function also including path length, path smoothness and grounding risk, as well as economic cost and safety risk (Guo et al., 2020)

1.2 Novel contribution

This paper presents a dynamic waypoint planning algorithm for path following based on PSO, using a fitness function with risk-related objectives. The approach may be summarized as follows: With heavy focus on modularity, simplicity and verifiability within each control module in the hierarchy of a larger ANS, autonomous navigation is presumed split into several independent algorithms or control loops. Hence, it is assumed that a coarse (global) pre-planned path or route is already defined and provided to the algorithm. However, no assumptions related to the efficiency, safety or optimality of the provided route is made.

Moreover, the output of the method is a new risk-aware sequence of waypoints, distributed along the pre-planned path (in this work distinctly denoted as the original *route vertices*) defined by the provided route of consecutive line segments. The distribution of the resulting waypoints is optimized with respect to a fitness function including explicit risk-based terms. These waypoints are subsequently used by a standard guidance algorithm, i.e. path following using a line-of-sight PID controller. This PSO-based structure may allow for both computation parallelization for performance increases, and more complex and dynamic risk-aware behavior utilizing discrete parameters or measurements of e.g. weather conditions or similar environment variables such as non-smooth operational modes, compared to solvers which require continuous or differentiable cost functions.

In order to further demonstrate the novel contribution of this paper with respect to the current literature, the method is compared to those of the listed similar works: Though based on the same principle, the simple PSO path planning approaches of **a)**, **b)** and **c)** do not consider risks nearby grounding obstacles, which may lead to unacceptable (i.e. dangerous) solutions with respect to safe navigation. In contrast, the approaches of **d)** and **e)** consider aspects of risk and safety through the use of linear distance weighting, with respect to static obstacles as well as bathymetry data in the PSO fitness function. However, several improvements may be made to the structure of both the fitness function and the utilization of electronic navigational charts (ENC) data, which will be presented and discussed throughout this paper. Thus, the main goal of this work is to introduce a problem formulation which improves upon the approaches presented in Xue (2022) and Guo et al. (2020), for increased performance and safety within MASS navigation.

2. METHOD

Particle swarm optimization is an iterative computational method based on weighted individual and social behavior among a swarm (collection) of candidate solution particles, given a set of defined learning parameters. The two main groups of PSO alternatives consist of variations using a *local* neighborhood (local best), and those using a *global* neighborhood (global best). Though there is no conclusive research declaring one PSO algorithm superior to the other for constrained problems (Engelbrecht, 2013), the global approach is chosen in this paper due to the problem structure of the presented case study (see Section A.2).

2.1 Algorithm initialization

The variables of the PSO algorithm are denoted as follows:

- ρ = the ship route (path) to be followed, i.e. a collection of J line segments L_j defined by static vertices (pre-planned route vertices), $j \in \{1, ..., J\}$.
- K = the number of new waypoints along each path segment $L_j \in \rho$, which partitions each line segment into K line subsegments l_{jk}, $k \in \{1, ..., K\}$.
- w_{jk} = a dynamic waypoint to be optimized, corresponding to each line (sub)segment l_{jk}.
- $\phi_{jk}(w_{jk})$ = a specific objective function for each waypoint w_{jk}, defined $J \cdot K$ times.
- G = a collection of convex grounding obstacle polygons, provided by an ENC module (Section 3.1).
- B = the spatial lower (minimum) and upper (maximum) bounds of all particles in the two-dimentionsal (2D) North-East (NE) plane, in this work equal to the boundaries of the visual environment constructed by the ENC module.
- S = an integer denoting the particle swarm size, i.e. the number of particles contained within each swarm.
- P_k = the set of S particles p for each waypoint, where each p contains a 2D waypoint particle w_p (point), a 2D particle velocity v_p, the local best particle w_b, and the local best cost c_b of w_b, calculated from $\phi_{jk}(w_{jk})$.
- σ = the set of all $J \cdot K$ particle swarms s_w, where each s_w contains all particles P_k, the cost function $\phi_{jk}(w_{jk})$, the global best swarm cost c_g, and the global best 2D swarm point w_g corresponding to each waypoint.
- N = an integer denoting the number of iterations to be repeated by the main loop of the PSO algorithm.
- $\theta = [w, c_1, c_2]$, a vector of PSO hyper-parameters.

Algorithm 1 and the notation of Fig. 1 outline the procedure for constructing the particle swarms σ for PSO in the context of the case study presented in this paper. The resulting collection of optimizable particle swarms σ are

Algorithm 1 PSO-Initialization

Input: ρ, ϕ_{jk}, B, G, K, S
Output: σ particle swarms for $J \cdot K$ waypoints
  procedure PSO-INIT($\rho, \phi_{jk}, S, B, G, K$)
    $\sigma \leftarrow \varnothing$ empty set of particle swarms
    for each line segment $L_j \in \rho$ **do**
      for each route partition $l_{jk} \in L_j$ **do**
        $\phi_{jk}(w_{jk}) \leftarrow \phi_{jk}$ wrt. $w_{jk\text{-}1}, w_{jk+1}, L_j, G$
        $P_k \leftarrow \varnothing$ empty set of particles for each w_{jk}
        for each s between 1 and S **do**
          $w_p \leftarrow$ random waypoint particle wrt. B
          $v_p \leftarrow$ random particle velocity wrt. B
          $c_b \leftarrow \infty$ as minimum known local cost
          $w_b \leftarrow w_p$ as best known local particle
          $P_k \leftarrow$ add particle $p(w_p, v_p, c_b, w_b)$
        end for
        $c_g \leftarrow \infty$ initial minimum global swarm cost
        $w_g \leftarrow \varnothing$ best global swarm state with c_g
        $s_w \leftarrow$ swarm with $(P_k, \phi_{jk}(w_{jk}), c_g, w_g)$
        $\sigma \leftarrow$ add waypoint swarm s_w
      end for
    end for
  end procedure

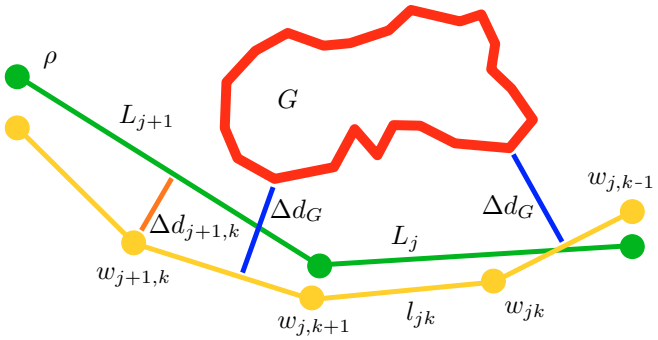

Fig. 1. An overview of the notation used in Algorithm 1.

along with θ and N provided to the general main loop PSO detailed in Algorithm 2 in the Appendix.

During initialization, any line segment of a ship route ρ is considered. Next, a distinct objective (fitness) function is defined with respect to the nearby grounding obstacles G and each line segment partition l_{jk}, given as

$$\phi_{jk}(w_{jk}) = \lambda(\Delta d_{k-1} + \Delta d_{k+1}) + \varepsilon \Delta d_{jk} + \mu e^{-\zeta \Delta d_G} \quad (1)$$

where Δd_{k-1} and Δd_{k+1} are the absolute distances (minus half of the line segment length l_{jk}) between w_{jk} and the previous and next waypoints $k-1$ and $k+1$, respectively, Δd_{jk} is the distance between w_{jk} and the original route line segment L_j, Δd_G is the minimum distance between the nearest individual polygon in G, and λ, ε, μ and ζ are tuning parameters. Thus, a unique objective function is constructed for each individual waypoint w_{jk} to be optimized along each line segment of the original ship route ρ, as well as the exponential cost of (1) for anti-grounding.

An initially empty set to hold the set of particle swarms is defined as σ, and is expanded as follows: For each route partition $j, k \ldots$, S number of candidate particles P_k are constructed, with each particle p containing the attributes w_p, v_p, c_b and w_b. Here, w_p is the initial waypoint location of the particle, v_p is the initial particle velocity, both uniformly randomized within the spatial bounds B defined by the ENC environment. Moreover, c_b is the best known local fitness (i.e. minimum cost) of the particle yet seen and w_b is the corresponding best known local waypoint location w_p which produced the best known fitness, to be updated throughout the PSO iterations.

In addition to the best known local fitness and waypoint seen by each particle p, the best known *global* fitness and waypoint location are denoted as c_g and w_g, respectively. These variables are shared by all candidate particles P_k of each particle swarm s_w, and are also updated during the main loop. Each of the constructed swarms s_w are added to the set of all swarms corresponding to each waypoint w_{jk} to be optimized. Thus, the initial variables of the PSO procedure are defined.

2.2 PSO main loop

From Algorithm 1, the collection of $J \cdot K$ particle swarms σ of size S with distinct objective functions ϕ_{jk} and candidate particles P_k, are used to optimize all corresponding waypoints for each line segment partition l_{jk} such that a complete global solution $W_g = \{w_1^g, w_2^g, ..., w_{JK}^g\}$ is returned after N iterations. The general procedure for a

standard PSO implementation for waypoint planning is presented as Algorithm 2 in the Appendix.

The result of the PSO is returned as the set W_g of all best known global states w_g from every swarm s_w in σ, which is the optimal solution after N iterations. See Section A.3 for more detailed discussion related to PSO initialization and tuning with respect to the hyper-parameters of θ.

2.3 Guidance and control

The overall ANS structures for risk-aware navigation as discussed in this work are shown in Fig. 2, and the simulations of Section 3 demonstrate and discuss the performance of the system alternatives.

The speed of the ship is held constant, for simplicity. Thus, the control problem at the lower level is course keeping, which is readily controllable using well-established methods such as PID control. Note however that due to the modular structure of this approach, a supervisory ANS may also employ an independent ship speed controller to ensure that the ship is able to sufficiently follow the risk-aware path resulting from the optimized waypoints.

The optimal waypoints of W_g computed by the PSO algorithm are subsequently used as inputs to a simple kinematic line-of-sight PID controller for MASS path following, based on the introductory geometric equations of Fossen et al. (2003):

$$\delta(t, w_{jk}) = K_p e(t) + K_i \int_0^t e(\tau)\, d\tau + K_d \dot{e}(t) \qquad (2)$$

where $e(t) = \chi(t) - \chi_\Theta(w_{jk})$, and $\delta(t, w_{jk})$ is the controlled rudder angle of the ship. $\chi(t)$ is the current course angle of the ship, and $\chi_\Theta(w_{jk}) = arctan(y_w - y_s, x_w - x_s)$ is the current target course calculated from $w_{jk} = P_w(x_w, y_w)$ and the current ship center position $P_s(x_s, y_s)$. K_p, K_i and K_d are the proportional, integral and derivative tuning coefficients of the controller, tuned such that the ship automatically steers toward the target position w_k along ρ. It may be noted that no assumption in general is made by the PSO method regarding the *selection* of the chosen target waypoint at any given time during simulation and/or operations, with respect to the path following algorithm subsequently applied to the resulting waypoints. Thus, the independent and modular structure assumed within the ANS furthermore promotes the utilization of well-known schemes for verifiable control problems and solutions, such as the ones mentioned in this work.

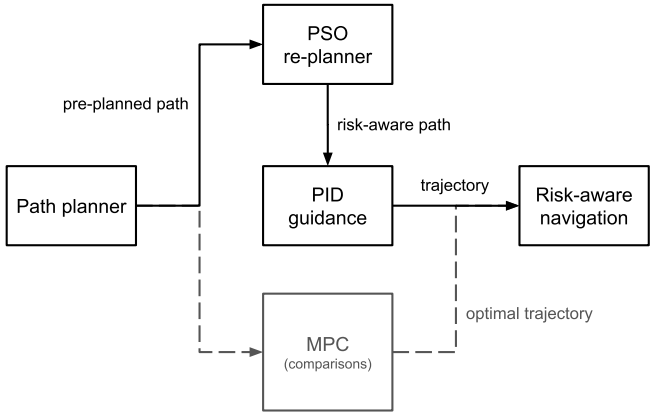

Fig. 2. Block diagram of the two ANS alternatives.

3. RESULTS

Several example simulations of PSO runs produced by Algorithms 1 and 2 implemented in Python are presented in this section, applied to a specific ENC environment located north-west of the Norwegian city of Ålesund, for the purpose of demonstration. The ENC data as well as methods for geometric computations with respect to points, lines and polygons, as well as environment and vessels visualizations are provided by the *SeaCharts* ENC package (Blindheim and Johansen, 2021).

3.1 ENC environment and visualization

In contrast to the three-dimensional (3D) environments used in the similar works of **d)** and **e)** from Section 1.1, the ENC environment is in this work structured as a 2D plane for computational efficiency. It is argued that this structure vastly reduces the computational load within the PSO algorithm, as the problem of safe navigation for a *surface* vessel by definition is naturally restricted to a 2D domain. Consequently, there is no practical reason to perform 3D distance calculations on ocean depths deeper than some static safety threshold (with respect to the maximum ship draft and shallows), if the ocean bed gradients or topography is not otherwise considered with respect to e.g. prediction of ocean current dynamics. The computational complexity required for such considerations should not be included within the main loop of the PSO algorithm, and may rather be moved to a separate and independent ANS module for e.g. analysis and/or prediction of environmental forces in future works.

Similarly, it is argued that it is not necessary to differentiate between too shallow waters, the shoreline or other static obstacles with respect to grounding or allision risks. For this purpose, the SeaCharts ENC package provides a significantly more efficient environment in which *any* (polygon) area of insufficient depth for navigation are consolidated into a single set of 2D grounding obstacles without loss of relevant information, constructed in its entirety before the PSO initialization. Thus, the minimum distance Δd_G of (1) to any nearby grounding obstacle is readily computed by a simple call of the **distance** function provided through SeaCharts by the dependency library *Shapely* (Gillies et al., 2007–) for efficient geometric calculations.

3.2 Risk-aware waypoint re-planning

In this section, a visualization of an optimal waypoint distribution produced by the PSO algorithm is presented, as well as a comparison visualization showing the end result trajectories of the PSO algorithm and PID controller versus those of a gradient-based MPC solver. Additional discussions concerning the initial values and convergence rates of the PSO run are presented in the Appendix.

Fig. 3 demonstrates a PSO run using the objective function (1), i.e. including the risk cost term $\mu \cdot \exp(-\zeta \Delta d_G)$.

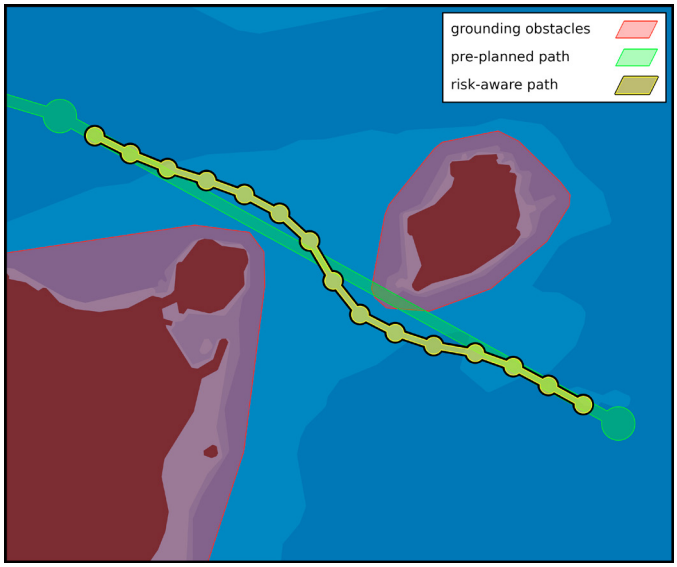

Fig. 3. Example demonstration of risk-aware waypoint re-planning, in which a risk-based cost term is added to the objective function.

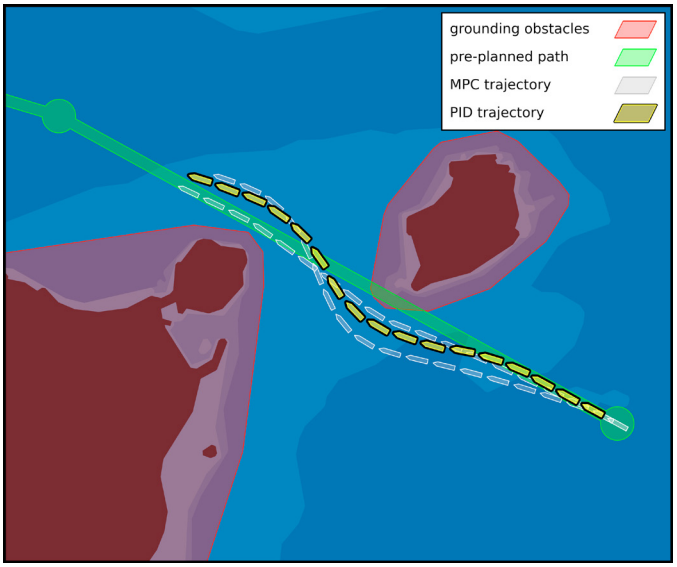

Fig. 4. A trajectory generated by the risk-aware waypoint planning and PID guidance approach of Fig. 3, compared with two other trajectories produced by an MPC scheme using a gradient-based solver.

As noted, the purpose of the risk term is to enable anti-grounding behavior with respect to the set of grounding obstacles G provided by the ENC module. The original ship route path ρ is shown with green disks as the pre-planned vertices, and green straight line segments drawn between them. The optimal re-planned waypoints solution with respect to the grounding obstacles are similarly shown as light yellow disks with straight line segments between them, highlighted by a black border. The grounding obstacles considered in this example are shown in red as two convex polygons surrounding two islands nearby the ship route.

Notice how the weighting of the distances between each waypoint in yellow and the grounding obstacles generates behavior similar to evasive maneuvers, shown by the significant deviations from the original straight line segment. This is considered an appropriate and indeed the desired result, ultimately verifying the effect of the risk term.

Next, Fig. 4 shows a comparison visualization of a yellow trajectory with black borders generated by the path following controller when given the re-planned path shown in Fig. 3, against two additional ship trajectories shown in white. These are produced by a gradient-based MPC scheme utilizing a similar risk-based cost function for anti-grounding (Blindheim et al., 2020). The two white trajectories are in general generated with the same parameters, with the only difference being the values given to the sensitivity constant ζ of the exponential risk cost term, assessed as edge case demonstrations for sufficiently safe autonomous navigation in this environment.

Moreover, Fig. 4 shows actual trajectories, i.e. the resulting simulation runs which utilize the line-of-sight PID controller (with constant speed) and a receding horizon MPC for comparison. Thus, in this context, the optimal path generated by the PSO based on the original ship route is provided as the input to the PID controller, which together produce a trajectory output (Fig. 2).

The MPC, however, calculates an optimal trajectory directly from the original ship route. The first white example demonstration of the MPC trajectory closest to the original pre-planned path is given $\zeta = 0.3$, which generates a trajectory in which the ship only barely avoids the convex polygons in red. The second example is given $\zeta = 0.07$, effectively reducing the rate of decay (weighting) applied to the distance measure between the ship center and the grounding obstacle, which in turn results in higher risk cost evaluations closer to the obstacles. It is apparent that the optimal waypoint placements computed by the PSO method as well as the resulting trajectory produced by the PID guidance controller primarily are located between the two MPC boundary trajectories, providing further confirmation of the validity of the risk-aware PSO approach.

Note, however, that the objective function (1) does not contain an explicit term related to path or waypoint smoothness, in contrast to the similar works of **d)** and **e)** of Section 1.1. The reason for this is two-fold: The inherent distance weighting between each consecutive waypoint acts as a self-smoothing mechanism, due to the iterative PSO cost updates in which each waypoint are adjusted with respect to its previous and subsequent neighbor. Moreover, it is argued that the significance of path smoothness of an (emergency) evasive maneuver should be negligible compared to the importance of simply avoiding the grounding obstacle in its entirety – particularly if the approach is extended to include collision avoidance of dynamic obstacles such as moving vessels. No distributive priority weighting is thus shared between path smoothness and grounding avoidance. Rather, the parameters of the exponential term should be appropriately tuned with respect to the ship dynamics and environment resolution, such that the trajectory generated by the path following algorithm given the resulting PSO waypoints is sufficiently smooth during normal operations.

3.3 Problem parameterization and alternatives

Several additional comments may be made with respect to the choice of parameterization and general performance of the approach compared to available alternatives.

The cost function (1) essentially formulates the problem to be solved, i.e. safe navigation along a path. Next, the problem solution is parameterized as $K = 20$ two-dimensional waypoints, resulting in 40 parameters to be optimized along the original route. In this work, the structure of these waypoints is also utilized by the PSO solver itself, i.e. 20 independent particles of 2 dimensions which are weighted with individual cost functions with respect to its neighbors. However, the parameterization used by the PSO may also have been structured as a single solution of a *waypoint collection* with 40 dimensions. In this case, the resulting particle swarms would consist of complete path solutions, and may be subject to be solved by other methods such as genetic algorithms (GA). It is however argued that this structure is less intuitive (see the Appendix), and the utilization of such parameterizations and e.g. GA is left for future works.

The PSO algorithm is not guaranteed to find the optimal solution, and is indeed subject to potential convergence to local minima, similar to nonlinear MPC and artificial potential fields. However, the chosen PSO structure alleviates this issue through a diversified initialization of particles that should cover the expected solution space. The "best known global particle" is shared across all particles in the swarm, which may help steer a larger selection of candidate solutions toward the true global minimum if it is found.

Moreover, the initialization of the PSO particles (the initial positions of the waypoints) has a critical impact on the optimal solution generated by both the PSO and the MPC solvers. However, the consideration of alternative paths e.g. around the island due to different initial conditions are considered outside the scope of this work. See the Appendix for more comments on aspects of the PSO performance.

4. CONCLUSION

In this paper, a method for dynamic risk-aware waypoint planning based on PSO was presented. The resulting sequence of optimal waypoints with respect to grounding obstacles as computed by the PSO algorithm, were subsequently used as inputs to a simple PID course controller for path following. This two-part modular structure is employed in order to facilitate a standardized and interchangeable hierarchy of potentially parallel, verifiable and robust controllers within a larger supervisory ANS for safe MASS navigation. Moreover, the performance of the PSO path re-planner was considered well-suited for the purpose of anti-grounding, compared to that of an analogous implementation using a gradient-based MPC solver including the same exponential risk term from previous works. Ultimately, the approach may be regarded as an adequate alternative to gradient-based controllers for safe MASS navigation, when utilizing more complex (i.e. mixed-integer or non-smooth) cost functions for risk-aware control.

ACKNOWLEDGEMENTS

This work was carried out with the helpful guidance of associates within the Online risk management and risk control for autonomous ships (ORCAS) project funded by the Research Council of Norway (grant number 280655), Kongsberg Maritime and DNV GL, and Centre for Autonomous Marine Operations and Systems (AMOS) at NTNU funded by the Research Council of Norway (grant number 223254).

REFERENCES

Alam, M.S., Rafique, M.U., and Khan, M.U. (2020). Mobile robot path planning in static environments using particle swarm optimization. *arXiv preprint arXiv:2008.10000*.

Bakdi, A., Glad, I.K., Vanem, E., and Engelhardtsen, Ø. (2020). Ais-based multiple vessel collision and grounding risk identification based on adaptive safety domain. *Journal of Marine Science and Engineering*, 8(1), 5.

Bitar, G., Martinsen, A.B., Lekkas, A.M., and Breivik, M. (2020). Two-stage optimized trajectory planning for asvs under polygonal obstacle constraints: Theory and experiments. *IEEE Access*, 8, 199953–199969.

Blindheim, S., Gros, S., and Johansen, T.A. (2020). Risk-based model predictive control for autonomous ship emergency management. *IFAC-PapersOnLine*, 53(2), 14524–14531.

Blindheim, S. and Johansen, T.A. (2021). Electronic navigational charts for visualization, simulation, and autonomous ship control. *IEEE Access*.

Candeloro, M., Lekkas, A.M., and Sørensen, A.J. (2017). A Voronoi-diagram-based dynamic path-planning system for underactuated marine vessels. *Control Engineering Practice*, 61, 41–54.

Dallolio, A., Bergh, T.K., Pedro, R., Øveraas, H., and Johansen, T.A. (2022). Enc-based anti-grounding and anti-collision system for a wave-propelled usv. *IEEE/MTS OCEANS Conference, India*.

Dewang, H.S., Mohanty, P.K., and Kundu, S. (2018). A robust path planning for mobile robot using smart particle swarm optimization. *Procedia computer science*, 133, 290–297.

Ding, F., Zhang, Z., Fu, M., Wang, Y., and Wang, C. (2018). Energy-efficient path planning and control approach of usv based on particle swarm optimization. In *OCEANS 2018 MTS/IEEE Charleston*, 1–6. IEEE.

Enevoldsen, T.T. and Galeazzi, R. (2021). Grounding-aware RRT* for Path Planning and Safe Navigation of Marine Crafts in Confined Waters. *IFAC-PapersOnLine*, 54(16), 195–201.

Engelbrecht, A.P. (2013). Particle swarm optimization: Global best or local best? In *2013 BRICS congress on computational intelligence and 11th Brazilian congress on computational intelligence*, 124–135. IEEE.

Fossen, T.I., Breivik, M., and Skjetne, R. (2003). Line-of-sight path following of underactuated marine craft. *IFAC proceedings volumes*, 36(21), 211–216.

Gillies, S. et al. (2007–). Shapely: Manipulation and analysis of geometric objects. URL `https://github.com/Toblerity/Shapely`.

Guo, X., Ji, M., Zhao, Z., Wen, D., and Zhang, W. (2020). Global path planning and multi-objective path

control for unmanned surface vehicle based on modified particle swarm optimization (pso) algorithm. *Ocean Engineering*, 216, 107693.

Kufoalor, D.K.M., Johansen, T.A., Brekke, E.F., Hepsø, A., and Trnka, K. (2020). Autonomous maritime collision avoidance: Field verification of autonomous surface vehicle behavior in challenging scenarios. *Journal of Field Robotics*, 37(3), 387–403.

Ma, Y., Hu, M., and Yan, X. (2018). Multi-objective path planning for unmanned surface vehicle with currents effects. *ISA transactions*, 75, 137–156.

Martinsen, A.B., Lekkas, A.M., and Gros, S. (2021). Optimal model-based trajectory planning with static polygonal constraints. *IEEE Transactions on Control Systems Technology*.

Niu, H., Lu, Y., Savvaris, A., and Tsourdos, A. (2018). An energy-efficient path planning algorithm for unmanned surface vehicles. *Ocean Engineering*, 161, 308–321.

Poli, R., Kennedy, J., and Blackwell, T. (2007). Particle swarm optimization. *Swarm intelligence*, 1(1), 33–57.

Shah, B.C. and Gupta, S.K. (2016). Speeding up A* search on visibility graphs defined over quadtrees to enable long distance path planning for unmanned surface vehicles. In *Twenty-Sixth International Conference on Automated Planning and Scheduling*.

Shi, C., Zhang, M., and Peng, J. (2007). Harmonic potential field method for autonomous ship navigation. In *2007 7th International Conference on ITS Telecommunications*, 1–6. IEEE.

Singh, Y., Sharma, S., Sutton, R., Hatton, D., and Khan, A. (2018). A constrained A* approach towards optimal path planning for an unmanned surface vehicle in a maritime environment containing dynamic obstacles and ocean currents. *Ocean Engineering*, 169, 187–201.

Song, R., Liu, Y., and Bucknall, R. (2017). A multi-layered fast marching method for unmanned surface vehicle path planning in a time-variant maritime environment. *Ocean Engineering*, 129, 301–317.

Song, R., Liu, Y., and Bucknall, R. (2019). Smoothed A* algorithm for practical unmanned surface vehicle path planning. *Applied Ocean Research*, 83, 9–20.

Utne, I.B., Sørensen, A.J., and Schjølberg, I. (2017). Risk management of autonomous marine systems and operations. In *International conference on offshore mechanics and arctic engineering*, volume 57663, V03BT02A020. American Society of Mechanical Engineers.

Vagale, A., Bye, R.T., Oucheikh, R., Osen, O.L., and Fossen, T.I. (2021a). Path planning and collision avoidance for autonomous surface vehicles II: a comparative study of algorithms. *Journal of Marine Science and Technology*.

Vagale, A., Oucheikh, R., Bye, R.T., Osen, O.L., and Fossen, T.I. (2021b). Path planning and collision avoidance for autonomous surface vehicles I: a review. *Journal of Marine Science and Technology*.

Vitus, M.P., Waslander, S.L., and Tomlin, C.J. (2008). Locally optimal decomposition for autonomous obstacle avoidance with the tunnel-MILP algorithm. In *2008 47th IEEE Conference on Decision and Control*, 540–545. IEEE.

Xue, H. (2022). A quasi-reflection based sc-pso for ship path planning with grounding avoidance. *Ocean Engineering*, 247, 110772.

Zaccone, R. (2021). Colreg-compliant optimal path planning for real-time guidance and control of autonomous ships. *Journal of Marine Science and Engineering*, 9(4), 405.

Zhou, J., Wang, C., and Zhang, A. (2020). A COLREGs-Based Dynamic Navigation Safety Domain for Unmanned Surface Vehicles: A Case Study of Dolphin-I. *Journal of Marine Science and Engineering*, 8(4), 264.

Appendix A. APPENDIX: THE PSO ALGORITHM

Algorithm 2 presents the general procedure of a standard PSO, and Algorithm 3 presents the standard subroutine for updating the velocities of each swarm particle within the main loop.

For each iteration $i = 1, ..., N$, the fitness (i.e. objective cost) of every particle $p \in P_k$ within each particle swarm $s_w \in \sigma$ is evaluated with respect to its distinct objective function ϕ_{jk}, denoted as c_p. If c_p is less than the current best known local cost c_b, the new cost replaces it as the best known local cost for that particle. Moreover, if c_p is less than the current best known *global* cost c_g, the new cost value also replaces the global best known cost for any particle across all swarms in σ. Both the local and global best known waypoints w_b and w_g are thus dynamically updated during runtime.

After each objective function evaluation, the current velocity and state of each particle p is updated. The velocity is computed through the *ParticleVelocity* procedure presented in Algorithm 3, in which the hyper parameters w,

Algorithm 2 PSO-MainLoop

Input: σ, θ, N
Output: best global solution W_g after N iterations
procedure PSO-MAIN(σ, θ, N)
  while iteration $< N$ **do**
    for each waypoint swarm $s_w \in \sigma$ **do**
      $P_k \leftarrow$ current particles of swarm s_w
      $\phi_{jk} \leftarrow$ specific cost function of swarm s_w
      for each waypoint particle $p \in P_k$ **do**
        $w_p \leftarrow$ current waypoint location
        $c_p \leftarrow$ evaluate local cost $\phi_{jk}(w_p)$
        if $c_p < c_b$ of p **then**
          $c_b \leftarrow c_p$ new best local fitness
          $w_b \leftarrow w_p$ new best local waypoint
          $p \leftarrow$ update particle with c_b, w_b
        end if
        if $c_p < c_g$ of s_w **then**
          $c_g \leftarrow c_p$ new best global fitness
          $w_g \leftarrow w_p$ new best global waypoint
          $s_w \leftarrow$ update swarm with c_g, w_g
        end if
      end for
      for each particle $p \in P_k$ **do**
        $v_p \leftarrow$ update $ParticleVelocity(p, w_g, \theta)$
        $w_p \leftarrow$ add particle velocity v_p to w_p
        $p \leftarrow$ update particle with w_p, v_p
      end for
    end for
  end while
  $W_g \leftarrow$ complete solution of all K global bests w_g
end procedure

Algorithm 3 ParticleVelocity

Input: p, w_g, θ
Output: updated weighted particle velocity v_{p+1}
procedure PV(p, w_g, θ)
$w \leftarrow$ particle inertia weight $\in \theta$
$c_1 \leftarrow$ cognitive weight $\in \theta$
$c_2 \leftarrow$ social weight $\in \theta$
$w_p \leftarrow$ state vector of particle p
$w_b \leftarrow$ local best state vector of particle p
$r_1 \leftarrow$ first uniformly random value $\in [0, 1]$
$r_2 \leftarrow$ second uniformly random value $\in [0, 1]$
$\Delta b \leftarrow w_b - w_p$ local best difference
$\Delta g \leftarrow w_g - w_p$ global best difference
$v_{p+1} \leftarrow wv_p + c_1 r_1 \Delta b + c_2 r_2 \Delta g$
end procedure

c_1 and c_2 are used as weights to adjust the direction and magnitude of the new particle velocity, with respect to the previous value. The difference between the local and global best known state w_b and w_g are weighted with the *cognitive* constant c_1 and the *social* constant c_2 respectively, as well as two uniformly random variables r_1 and r_2 between 0 and 1, with both terms added to the inertia constant w multiplied by the previous particle velocity v_p to produce the updated velocity v_{p+1}. The updated state of the particle is simply the sum of the previous state w_p and the calculated particle velocity v_p. The global best waypoint particles W_g are lastly returned as the PSO solution.

A.1 Choice of parameterization and algorithm

Several alternative methods for risk-aware path adjustment or re-planning were considered. Though the well-established A* algorithm and RRT* are suitable for fast optimal path planning in large search spaces from a start point toward an end goal, they are considered less appropriate for path adjustment or re-planning in the case where an optimal or near-optimal pre-planned path is given.

The structure and principles of GA compared to that of PSO are more similar, and are particularly suited for combinatorial problems. However, GA are dependent on a valid and efficient function for recombinations and mutations between candidate solutions in order to explore the search space. Consequently, it is considered more appropriate to utilize the inherent concepts of PSO particle velocities and fitness for this particular application.

A.2 PSO performance and convergence

The convergence rate of the computed PSO solutions may be evaluated with respect to the resulting global best solution, in order to further assess the performance of the PSO method. The initial *global* best known waypoints are in this work given completely random start values for demonstration of robustness. It may also be noted that some of these values can be initialized *onto* obstacles, as the cost rises even higher inside obstacles due to negative distances within the exterior boundary of a polygon. Though the cost is non-negligible if grounding polygons are nearby, the increased cost of the risk term is relatively insignificant with respect to the initial best global costs present in the system during PSO initialization. Nevertheless, the algorithm is considered feasible, with respect to computational time during real-time navigation: The PSO optimization for Fig. 3 and 4 with $N = 100$ and $S = 300$ was on a standard desktop computer applied to the considered line segment of length 2.3 km, and was computed within 13 seconds – using the random waypoint initialization approach for the purpose of demonstration.

Though the nominal route is assumed optimal or near-optimal, grounding risks are implicitly always present in an environment with inherent uncertainties, such as e.g. weather conditions, tides, winds or currents, and the potential for unexpected events such as e.g. the sudden presence of unmapped grounding hazards such as moored vessels, and failures or faults onboard the vessel. Thus, the main application of the proposed system is to continuously adapt the nominal route with small alterations, in order to adhere to changes that were introduced into the environment after the route was originally planned. Rigorous handling of uncertainty aspects such as e.g. reduced safety margins with respect to estimated time to grounding due to winds or currents will be addressed in future works.

A.3 PSO initialization and hyper-parameters

The random initialization of particle candidates of each particle swarm (recall that one swarm corresponds to one waypoint) does in general favor *particle exploration*. However, the PSO consequently has a large initial cost due to being located far away from the low-cost regions around the pre-planned path (as defined through the defined cost function), and the initial values of two subsequent runs will by construction always be random, potentially leading to unstable results between runs and higher convergence rates. The monotonic nature of the cost gradient is nevertheless easily verified during runtime through visual inspection of e.g. a cost graph, and is a direct consequence of the fact that new global best particles by definition always have a lower cost or better fitness compared to the last.

Another approach is to give the initial global best particles e.g. the exact optimal placements along the original straight path segment of the ship route (analogous to the concept of "warm start" of e.g. gradient-based solvers). Assuming no grounding obstacles are present, the initial cost would in this case be vastly reduced, and the convergence rate accordingly improved. Moreover, it is argued that particle exploration is not the primary objective of the PSO re-planner, as the main function of the risk-aware re-planned path is merely to lightly adjust a presumed optimal or near-optimal pre-planned path due to online risk-related influences such as changing weather conditions, nearby vessels, unforeseen events or disturbances. Due to this obvious increase in performance with the above scope in mind, this approach is recommended as the favored initialization method for future works.

Ultimately, one may achieve satisfactory performance for a large selection of optimization problems by careful tweaking of the PSO hyper-parameters of Algorithm 3. Candidate particles are during optimization thus allowed to probe previously not yet evaluated fitness values of unvisited states through semi-random exploration, while simultaneously approaching both previously seen local and global minima. In the simulation of Fig. 3, $w = 0.75$, $c_1 = 1.0$, and $c_2 = 2.0$.