Why the Fossen model is so popular for marine vehicles' dynamic analysis?

Paweł Piskur, Piotr Szymak, Rafał Kot, Łukasz Marchel, Krzysztof Naus
Polish Naval Academy
Smidowicza 69, 81-127 Gdynia, Poland
Email: p.piskur@amw.gdynia.pl, p.szymak@amw.gdynia.pl, r.kot@amw.gdynia.pl, l.marchel@amw.gdynia.pl, k.naus@amw.gdynia.pl

KEYWORDS

Dynamic models simulation; Fossen model; Marine vehicles; mas-spring-dumper.

ABSTRACT

Among various approaches, the Fossen model has gained significant popularity due to its comprehensive representation of nonlinear hydrodynamic forces, stability, and modular structure. This paper compares different simulation models using a one-degree-of-freedom (1-DOF) system, demonstrating the benefits of the Fossen model over conventional mass-spring-damper-based approaches. By incorporating added mass, nonlinear damping, and restoring forces, the Fossen model provides a more accurate representation of marine dynamics. Additionally, its difference equation formulation ensures efficient computation while preserving stability, making it suitable for both real-time applications and high-fidelity simulations. The study highlights the impact of different numerical integration techniques and simulation environments, providing a foundation for extending the analysis to multi-degree-of-freedom systems.

INTRODUCTION

A system model describes the relationship between input and output variables, typically using differential equations [1], state-space representations or transfer functions. Dynamic models can be represented in both continuous and discrete time domains [2], making them essential for analyzing and designing control systems [3].

One of the most widely recognized dynamic models in marine vehicle dynamics and control is the Fossen model, which has been a standard since the 1990s. Its foundation was laid by Thor I. Fossen in his Ph.D. dissertation (1991) and later formalized with the publication of *Guidance and Control of Ocean Vehicles (1994)* [4]. This work established a systematic mathematical framework for modeling marine craft, integrating hydrodynamic forces, control strategies, and nonlinear dynamics.

Throughout the late 1990s and 2000s, the model gained widespread adoption for various marine applications, including Autonomous Underwater Vehicles (AUVs) [5], Remotely Operated Vehicles (ROVs), and ships. Significant contributions from the Marine Cybernetics Group at NTNU further enhanced its applicability. The release of Fossen's second book, *Handbook of Marine Craft Hydrodynamics and Motion Control (2011)* [6], introduced refined modeling techniques and advanced control strategies, reinforcing its status as a benchmark in the field.

The success of the Fossen model is attributed to its modularity, compatibility with modern control systems, and extensive real-world validation. Its influence extends beyond academia, as it has been integrated into widely used simulation platforms such as Matlab (Marine Systems Simulator) [7], Simulink, and Python-based simulation frameworks. Today, it remains an industry and research standard in dynamic modeling, simulation, and control of marine vehicles [8], [9], [10].

The Fossen model for six degrees of freedom (6-DOF) is formulated mathematically as:

$$\mathbf{M}(\boldsymbol{\eta})\dot{\mathbf{v}} + \mathbf{C}(\mathbf{v},\boldsymbol{\eta})\mathbf{v} + \mathbf{D}(\mathbf{v})\mathbf{v} + \mathbf{g}(\boldsymbol{\eta}) = \boldsymbol{\tau}, \tag{1}$$

where $\mathbf{M}(\boldsymbol{\eta})$ is the inertia and added mass matrix, $\mathbf{C}(\mathbf{v},\boldsymbol{\eta})$ is the Coriolis and centripetal matrix, $\mathbf{D}(\mathbf{v})$ is the damping matrix, $\mathbf{g}(\boldsymbol{\eta})$ is the vector of hydrostatic forces and moments, and $\boldsymbol{\tau}$ represents the control input. Vectors $\mathbf{v}$ and $\boldsymbol{\eta}$ denote velocity and position/orientation in the body-fixed and inertial frames, respectively.

A one-degree-of-freedom (1-DOF) dynamic model, such as a mass-spring-damper system, provides a simplified way to introduce the main principles of marine system dynamics modeling. This approach makes it easier to understand how hydrodynamic forces, damping, and environmental interactions are represented in the Fossen model before moving to the full six-degree-of-freedom (6-DOF) system.

Using a 1-DOF model allows for the step-by-step examination of the effects of mass and added mass (influence of inertia), hydrodynamic forces (linear and nonlinear damping), restoring forces and stability (e.g., restoring moments), external forces (e.g., control forces, waves, and ocean currents) [11]. Modeling a 1-DOF system enables validation and testing of the equations of motion under controlled conditions [12], [13]. It gives comparison of simulation results with analytical solutions or experimental data, helping to understand both the limitations and benefits of the Fossen model. This structured approach helps clarify dynamic mechanisms before introducing more complex interactions.

Many control algorithms (e.g., PID, LQR, MPC) are first tested on 1-DOF models before being implemented in full-scale systems. This approach allows for a better understanding of vehicle dynamics and facilitates the identification of the main parameters, such as hydrodynamic damping and the influence of control forces.

Once the principles of the Fossen model are explained using a 1-DOF system, it becomes logical to extend the discussion to the full 6-DOF model, where all forces and moments act simultaneously. This hierarchical presentation prevents misunderstandings caused by overly complex mathematical dependencies in the initial learning phase.

This paper focuses on comparing different modeling techniques applied to a one-degree-of-freedom mass-spring-damper (MSD) system.

1-DOF SYSTEM SIMULATION

A second-order differential equation describes the dynamics of a MSD model in a 1-DOF, which can be presented as in Fig. 1.

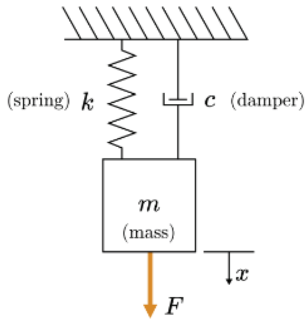


Fig. 1. Mass-spring-damper (MSD) single degree of freedom system

The equation of motion for a 1-DOF in a single direction is given by the equation (2):

$$(m + m_a)\ddot{x} + b\dot{x} + cx = F_{ext}(t) \tag{2}$$

where:

m – mass of the floating body $[kg]$,
m_a – added mass due to hydrodynamic effects $[kg]$,
$\ddot{x}$ – acceleration $[m/s^2]$,
$\dot{x}$ – velocity $[m/s]$,
x – displacement $[m]$,
b – hydrodynamic damping coefficient $[Ns/m]$,
c – hydrostatic restoring coefficient (depends on buoyancy and gravity) $[N/m]$,
$F_{ext}(t)$ – external forces acting on the body, e.g., wave excitation force $[N]$.

When a marine vehicle is analysed, the mass term $(m + m_a)$ accounts for the system's inertia, including the added mass effect from the surrounding water. The damping term $b\dot{x}$ represents the hydrodynamic resistance to motion, which can be linear or nonlinear. The restoring force cx arises due to buoyancy and gravity effects, bringing the system back to equilibrium. Additionally, the external force $F_{ext}(t)$ represents disturbances such as wave forces or control inputs. Depending on the chosen reference frame, the motion can occur in either the horizontal or vertical direction. Solving this equation provides the system's response in terms of displacement $x(t)$ under external forces $F(t)$.

I. Methods of Modeling

A. Block diagram in simulink

The equation (2) can be used to build the block diagram as presented in Fig. 2. For that reason, the equation (2) can be modified as presented in the equation (3). The initial values for displacement and velocity are defined as v0 and x0.

$$\ddot{x} = \frac{F_{ext}(t) - b\dot{x} - cx}{m + m_a} \tag{3}$$

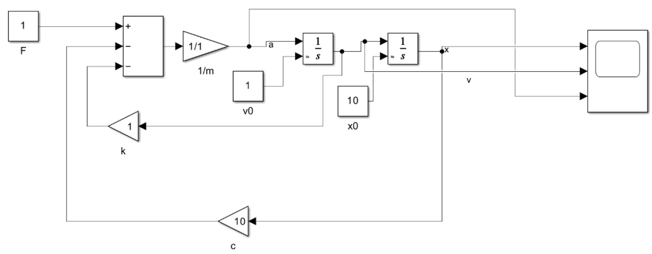


Fig. 2. Simulation model with 1-DOF designed in Matlab-Simulink as a block diagram

The equation (3) can be directly defined in the function block and added to the block diagram as presented in Fig. 3. The significant difference is the lack of initial position and speed values, while the constant values are defined in separate blocks.

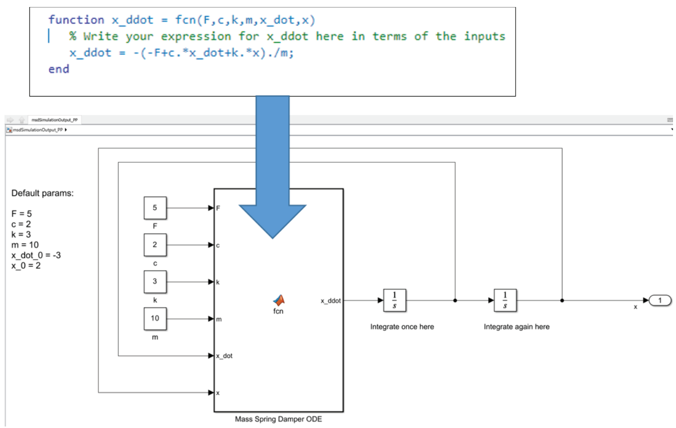


Fig. 3. Simulation model with 1-DOF designed in Matlab-Simulink as a block diagram with build in function

Also, if we assume that the initial conditions for the linear model are equal to zero, the transmittance can be used. The transmittance can be represented as equation (4):

$$G(s) = \frac{1}{ms^2 + cs + k} \tag{4}$$

It can be implemented in Simulink model with using the `Transfer Function` block (see Fig. 4). In all the above models, different ODE solvers can be used

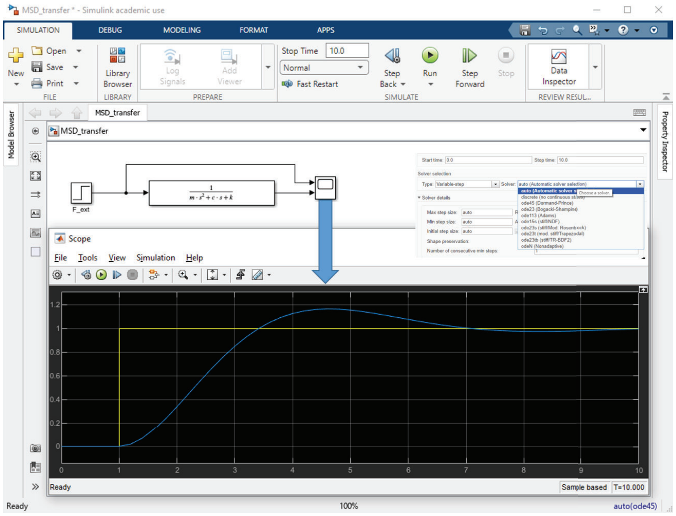


Fig. 4. Simulation model with 1-DOF designed in Matlab-Simulink with transmittance function

(see Fig. 4). Solvers available in Matlab-Simulink can be categorized into variable-step and fixed-step methods. Variable-step solvers, such as `ode45` (Dormand-Prince method), `ode15s` (stiff solver), and `ode23` (low-order method), dynamically adjust the step size to balance accuracy and computational efficiency. Fixed-step solvers, such as `ode4` (Runge-Kutta 4th order), `ode1` (Euler method), and `FixedStepDiscrete`, are useful for real-time applications and discrete-time systems. The choice of solver depends on system stiffness, accuracy requirements, and computational constraints. Another way of modelling the dynamic model is through direct definitions of differential or difference equations in Matlab script as presented in the next paragraphs.

B. Modeling with Ordinary Differential Equations (ODE)

ODE-based modeling involves converting the second-order differential equation into a system of first-order equations:

$$x_1 = x, \tag{5}$$
$$x_2 = \dot{x}, \tag{6}$$
$$\dot{x}_1 = x_2, \tag{7}$$
$$\dot{x}_2 = \frac{-kx_1 - cx_2}{m}. \tag{8}$$

This system can be solved numerically using `ode45` function, with using MATLAB code for ODE-based modeling:

```
% System parameters
m = 1.0; k = 10; b = 1;
% Initial conditions
x0 = 10; v0 = 1;
% Time span
t0 = 0; tf = 10;
% ODE function
odefun = @(t, x) [x(2); (-k * x(1) - b * x(2))
    / m];
% Solve the system
[T, X] = ode45(odefun, [t0, tf], [x0, v0]);
```

This method provides an accurate continuous-time solution but requires computational resources for numerical integration.

C. Modeling with Difference Equations

A discrete-time approximation using difference equations is effective for numerical simulations. The system is discretized using:

$$a_k = \frac{-kx_k - bv_k}{m}, \tag{9}$$
$$v_{k+1} = v_k + a_k \cdot \Delta t, \tag{10}$$
$$x_{k+1} = x_k + v_{k+1} \cdot \Delta t. \tag{11}$$

The MATLAB code for difference equation-based modeling implementation is:

```
% Time discretization
dt = 0.05; t = t0:dt:tf;
% Initialize state variables
x = zeros(size(t)); v = zeros(size(t)); a =
    zeros(size(t));
% Initial conditions
x(1) = x0; v(1) = v0;
% Discrete-time simulation
for i = 1:length(t)-1
    a(i) = (-k * x(i) - b * v(i)) / m;
    v(i+1) = v(i) + a(i) * dt;
    x(i+1) = x(i) + v(i+1) * dt;
end
```

The discrete method is computationally efficient but introduces numerical errors due to discretization and requires careful parameter tuning. The choice of Δt directly affects the simulation's numerical stability and resolution. A significant time step may cause numerical instability and loss of detail in the system's dynamic response. Small time step improves accuracy but increases computation time and may be infeasible for real-time applications. As a practical guideline, one often starts with a medium time step (e.g., 0.01 s) and performs sensitivity analyses to examine the trade-off between accuracy and computational cost.

The Fossen model describes both low-frequency ship motions (e.g., surge, sway, yaw) and potentially higher-frequency phenomena (e.g., wave-induced vibrations). To capture all relevant frequencies:

$$\Delta t \leq \frac{1}{10\, f_{\max}}, \tag{12}$$

where $f_{\max}$ is the highest frequency of interest. Longer time steps can disregard certain fast dynamics, whereas overly short time steps can be computationally expensive.

RESULTS OF SIMULATIONS

The results of the simulations using different modeling methods provide insights into the accuracy, computational efficiency, and applicability of each approach. The ODE-based method, implemented with MATLAB's `ode45`, offers high accuracy and smooth solutions but requires greater computational resources due to its adaptive time-stepping mechanism. In Fig. 5 the results of acceleration, velocity and displacement are presented, while in Fig. 6 the comparison of solvers (ode45, ode23, ode115, ode23s) are presented for the maximal value of each acceleration, velocity and displacement. It can be shown that value are similar, and adequate enough for dynamic simulation and analysis. Simulation provided in all presented methods gives almost the same results, but the time calculation and complexity of formulation varies. The discrete-time difference equation method is computationally efficient and straightforward to implement, making it suitable for real-time applications; however, it introduces discretization errors that must be carefully managed to maintain numerical stability. Simulations in MATLAB Simulink allow for a graphical block-based approach, offering flexibility in integrating various solver settings, control strategies, and real-time execution scenarios. The choice of method depends on the application: high-precision simulations favor ODE solvers, whereas real-time implementations benefit from discrete-time formulations.

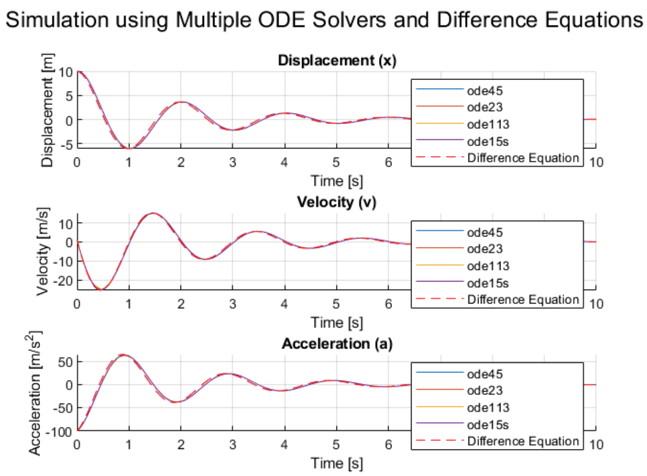


Fig. 5. Acceleration, velocity and displacement as a function in time

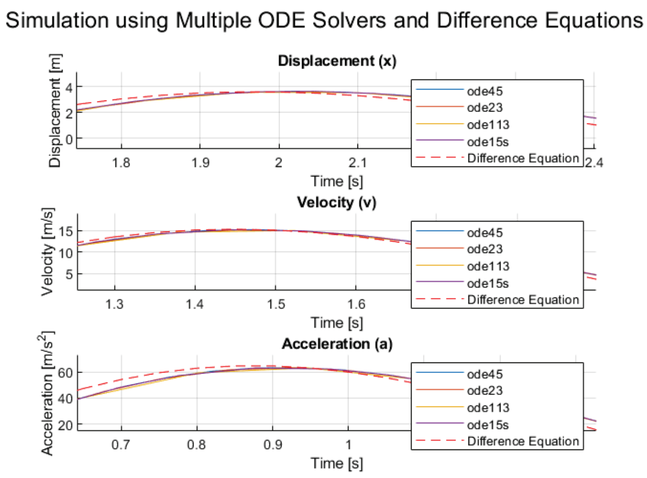


Fig. 6. Comparison of different method and solver used during the simulation

If high-frequency phenomena (e.g., wave-induced vibrations) are critical, smaller steps on the order of 0.001 s or even less may be required.

CONCLUSIONS

In practice, models are often linearized around specific conditions for control design tasks. However, a fully nonlinear system is often necessary to accurately capture large-amplitude motions or wave interactions. With its ability to account for nonlinear effects and hydrodynamic forces, the Fossen model remains the most robust approach for marine vehicle dynamics, ensuring reliable performance across various operating conditions. The Fossen model is inherently nonlinear, and its difference equation formulation enables fast computation while preserving the stability of solutions, with matrix notation further enhancing the model's readability.
In the next step, the analysis will be extended to systems with three degrees of freedom.

Appendices

I. ODE (ordinary differential equation)

Below is the MATLAB code that solves the differential equations numerically using the ODE solvers and plots the results.

```
% Parameters
m = 1.0;          % mass (kg)
k = 10;           % spring stiffness coefficient
    (N/m)
b = 1;            % damping coefficient (Ns/m)

% Initial conditions
x0 = 10;          % initial displacement (m)
v0 = 1;           % initial velocity (m/s)

\% Simulation time and time step
t0 = 0;           % initial time
tf = 10;          % final time
dt = 0.05;        % time step
t = t0:dt:tf;     % time vector

% ODE function definition
odefun = @(t, y) [y(2); (-k * y(1) - b * y(2))
    / m];

% Solvers for differential equations
solvers = {'ode45', 'ode23', 'ode113', 'ode15s
    '};
ode_results = struct();

for s = 1:length(solvers)
    solver = solvers{s};
    [T, X] = feval(solver, odefun, [t0, tf], [
        x0, v0]);
    X(:,3) = (1/m) * (-k * X(:,1) - b * X(:,2)
        );
    ode_results.(solver).T = T;
    ode_results.(solver).X = X;
end

% Difference equation solution
x = zeros(size(t)); % displacement
v = zeros(size(t)); % velocity
a = zeros(size(t)); % acceleration

% Setting initial conditions
x(1) = x0;
v(1) = v0;

% Simulation of mass-spring-damper system
    using difference equations
```

```
for i = 1:length(t)-1
    % Calculate acceleration
    a(i) = (-k * x(i) - b * v(i)) / m;
    % Calculate velocity
    v(i+1) = v(i) + a(i) * dt;
    % Calculate displacement
    x(i+1) = x(i) + v(i+1) * dt;
end

% Plotting results
figure;

% Displacement comparison
subplot(3,1,1);
hold on;
for s = 1:length(solvers)
    solver = solvers{s};
    plot(ode_results.(solver).T, ode_results.(solver).X(:,1), 'DisplayName', solver);
end
plot(t, x, 'r--', 'DisplayName', 'Difference Equation');
title('Displacement (x)');
xlabel('Time [s]');
ylabel('Displacement [m]');
legend;
grid on;
hold off;

% Velocity comparison
subplot(3,1,2);
hold on;
for s = 1:length(solvers)
    solver = solvers{s};
    plot(ode_results.(solver).T, ode_results.(solver).X(:,2), 'DisplayName', solver);
end
plot(t, v, 'r--', 'DisplayName', 'Difference Equation');
title('Velocity (v)');
xlabel('Time [s]');
ylabel('Velocity [m/s]');
legend;
grid on;
hold off;

% Acceleration comparison
subplot(3,1,3);
hold on;
for s = 1:length(solvers)
    solver = solvers{s};
    plot(ode_results.(solver).T, ode_results.(solver).X(:,3), 'DisplayName', solver);
end
plot(t, a, 'r--', 'DisplayName', 'Difference Equation');
title('Acceleration (a)');
xlabel('Time [s]');
ylabel('Acceleration [m/s^2]');
legend;
grid on;
hold off;

% Display results
sgtitle('Simulation using Multiple ODE Solvers and Difference Equations');
```

References

[1] A. Grzadziela and M. Kluczyk, "Shock absorbers damping characteristics by lightweight drop hammer test for naval machines," *Materials*, vol. 14, no. 4, p. 772, 2021.

[2] B. Szturomski and R. Kiciński, "Simulation of the impact resistance of kilo type submarine loaded with non-contact mine explosion," *Zeszyty Naukowe Akademii Marynarki Wojennej*, vol. 58, 2017.

[3] M. Przybylski, "Selection of the depth controller for the biomimetic underwater vehicle," *Electronics*, vol. 12, no. 6, 2023. [Online]. Available: https://www.mdpi.com/2079-9292/12/6/1469

[4] T. I. Fossen, *Guidance and Control of Ocean Vehicles*. John Wiley & Sons, 1994.

[5] J. Zalewski and S. Hożyń, "Computer vision-based position estimation for an autonomous underwater vehicle," *Remote Sensing*, vol. 16, no. 5, p. 741, 2024.

[6] T. I. Fossen, *Handbook of Marine Craft Hydrodynamics and Motion Control*. John Wiley & Sons, 2011.

[7] ——, "Marine systems simulator (mss)," 2020, available at: https://www.fossen.biz/mss.

[8] P. Szymak, T. Praczyk, K. Naus, B. Szturomski, M. Malec, and M. Morawski, "Research on biomimetic underwater vehicles for underwater isr," in *Ground/Air Multisensor Interoperability, Integration, and Networking for Persistent ISR VII*, vol. 9831. SPIE, 2016, pp. 126–139.

[9] P. Szymak and M. Przybylski, "Thrust measurement of biomimetic underwater vehicle with undulating propulsion," *Maritime Technical Journal*, vol. 213, no. 2, pp. 69–82, 2018.

[10] M. Akdağ, T. A. Pedersen, T. I. Fossen, and T. A. Johansen, "A decision support system for autonomous ship trajectory planning," *Ocean Engineering*, vol. 292, p. 116562, 2024.

[11] K. Jaskolski, "Two-dimensional coordinate estimation for missing automatic identification system (ais) signals based on the discrete kalman filter algorithm and universal transverse mercator (utm) projection," *Zeszyty Naukowe Akademii Morskiej w Szczecinie*, no. 52, pp. 82–89, 2017.

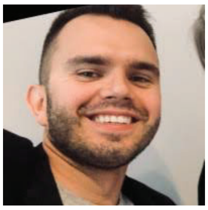
[12] M. Morawski, T. Talarczyk, and M. Malec, "Depth control for biomimetic and hybrid unmanned underwater vehicles," *Technical Transactions*, vol. 118, no. 1, 2021.

[13] T. Talarczyk, "A dynamic submerging motion model of the hybrid-propelled unmanned underwater vehicle: simulation and experimental verification," *International Journal of Applied Mathematics and Computer Science*, vol. 33, no. 2, pp. 207–218, 2023.

Paweł Piskur. Graduated from the Military University of Technology in Warsaw with a degree in Airplane Studies in 2004. He worked for 13 years in the Marine Aviation Base in Polish Army. In 2010 he received PhD degree from Koszalin University of Technology in mechanical engineering. Since 2017 he is working at Polish Naval Academy. In 2024 he obtained the academic degree of habilitated doctor at Air Force Institute of Technology in Warsaw. His research area is strictly connected with underwater unmanned vehicles, especially biomimetic propulsion systems.

Piotr Szymak. He is a scientist and navy officer. He has been graduated from the Military University of Technology in Warsaw with an MSc degree in electronics and telecommunication in 1999. He served for 2 years in the 6th Radioelectronic Centre in Polish Navy. Since 2001 he has been serving in Polish Naval Academy (PNA) in the following positions: assistant, lecturer, adjunct, head of department, assistant professor, and associate professor. He was a director of the Institute of Electrical Engineering and Automatics in PNA in the years 2015-2019. In the years 2019 - 2022, he was the Polish naval Captech National Coordinator at European Defence Agency. In 2004 he received a PhD degree from Polish Naval Academy, and in 2016 he obtained the academic degree of habilitated doctor at Cracow University of Technology. His research area is strictly connected with the modelling and control of marine objects, especially the unmanned surface and underwater vehicles, including recently biomimetic underwater vehicles.

Rafał Kot. Graduated from the Military University of Technology in Warsaw with a degree in Electronics and Telecommunication Studies in 2016. He worked for almost 6 years in the Marine Aviation Base in the Polish Army where he gained experience in servicing and maintenance of aircraft in the avionics area. From the beginning of 2022, he is working at Polish Naval Academy. His main research includes underwater obstacle detection systems, path planning algorithms, and collision avoidance systems for Autonomous Underwater Vehicles.



Lukasz Marchel.

He is a researcher and lecturer at the Polish Naval Academy in Gdynia. He obtained his Ph.D. in engineering sciences in 2020. His research interests focus on the modeling of physical phenomena, extreme optimization, parallel computing, and their applications in marine engineering. He has authored and co-authored numerous scientific publications in these fields.

Krzysztof Naus

He is a professor and researcher at the Polish Naval Academy in Gdynia. He obtained his Ph.D. in 2003 and his habilitation at the AGH University of Science and Technology in Kraków. His research focuses on signal processing, especially in radar and sonar systems. He has published extensively in the areas of software design for swarming underwater vehicles and path-planning algorithms for autonomous marine units.