From Fragmented Integration to Operational Reliability:

A Unified Marine Vehicle Operating Architecture

Chris Malzone, Beringia Marine Inc / Mission Robotics Inc.

Abstract

Marine robotics has advanced significantly at the component level over the past decade. Thrusters, enclosures, sensors, and payloads are now widely available and increasingly commoditized. The barrier to building a vehicle has never been lower. Yet the industry has not advanced at the same pace. The missing element is not better hardware. It is the platform architecture that binds components into coherent, field-deployable systems. Integration complexity, not hardware availability, is the primary constraint limiting commercial deployment in marine robotics today.

This paper presents a marine vehicle operating architecture implemented through Mission Robotics' NadirOS and Mission Dock platforms. Most control frameworks in use today were derived from aerial and ground robotics. They were not designed for acoustic positioning, six degree-of-freedom subsea dynamics, electrical fault isolation, or remote deployment. The architecture presented here replaces fragmented control stacks with an integrated vehicle operating system and a fault-observable, domain-isolated electrical and communications backbone, built on open standards, laying the foundation for expansion.

The central argument is not about incremental hardware refinement. It is about a shift in how platforms are evaluated. Vehicle acquisition cost is a narrow metric. The more relevant measure is operational reliability, mission repeatability, and total lifecycle integration overhead. Commercial scalability is determined by uptime, development velocity, and the ability to support increasingly complex missions without rebuilding the platform.

Index Terms—Marine robotics, ROV architecture, QoS-managed control, DDS, autonomy integration, electrical fault isolation, vehicle operating systems.

I. Introduction

Marine robotics has advanced significantly in its component ecosystem over the past decade. Thrusters, enclosures, lights, cameras, sensors, and payloads are available from multiple manufacturers at increasingly accessible price points. There are more options to build exactly what you want than at any prior point in the industry's history. OEMs and service providers can conceive of vehicles suited to specific mission profiles

and assemble them mechanically without significant constraint. The constraint is no longer what can be built. It is whether what is built can be reliably operated in the field, integrated without prohibitive overhead, and scaled as mission demands increase. The missing element is not another sensor or a faster data mux. It is the platform that holds all these components together.

End-user demands have increased at the same time. Operators are deploying smaller, more cost-effective vehicles to perform missions that previously required larger platforms. This reduces the size and cost of the support vessel, and in some cases eliminates the need for a crewed vessel entirely in favor of an unmanned surface vessel. Smaller vehicles doing more demanding work means greater functional demands on vehicle capabilities. Higher data quality, mission repeatability, and longer or more remote deployments are now expected from vehicles not traditionally used for work of this complexity. The gap between what can be mechanically assembled and what can be reliably operated in the field has widened.

The open-source software frameworks commonly used to manage these systems were not developed for the marine environment. They originate from aerial and ground robotics, where the control assumptions, sensor fusion requirements, and operational demands differ substantially from subsea applications. Underwater, there is no GPS. Positioning largely depends on acoustic systems (DVL, USBL) and AHRS, IMU, and depth sensors that must be fused together and fed directly into control loops. Vehicles operate across six degrees of freedom in a fluid medium that does not behave like air or ground. The electronics operate in a saltwater environment where a single shorted peripheral sensor can take down the entire vehicle if the architecture allows it. Recovery from failure is slow, expensive, and sometimes impossible mid-mission. These frameworks were not designed with any of this in mind, and the mismatch is architectural, not superficial.

Integration is where these constraints converge into real-world failure. Components designed in isolation expose failure modes that their individual specifications never anticipated. Without proper electrical architecture, a fault in one subsystem can propagate through shared power domains and take down compute, control, and payload simultaneously. Sensors on shared communication buses can interfere with or disable one another. The failure is rarely a single clean event. It is an unreliable pattern of behavior whose root cause is difficult to isolate and reproduce. These are not edge cases. They are the characteristic failure mode of systems assembled from components from different vendors that were never designed to share a system context. Building the architecture correctly from the start, with fault isolation and domain separation as design requirements rather than afterthoughts, is what allows these systems to behave predictably in the field.

In commercial operations, unreliable behavior has a direct cost. Downtime consumes vessel time and crew resources. Bad data means repeat dives and delayed deliverables. Every hour of field troubleshooting erodes mission profitability. For OEMs,

unresolved integration dependencies delay time-to-market and consume engineering capacity that should be directed at building capability. Vehicle acquisition cost is a poor basis for platform evaluation when the real costs accumulate in operations, engineering overhead, and lost mission time over a twelve to thirty-six month lifecycle.

II. Architectural Fragmentation in Existing Systems

The control frameworks most commonly deployed on inspection-class marine vehicles were not built for the subsea environment. ArduSub, the firmware at the core of many such systems including the widely deployed BlueROV2, is derived from ArduPilot, a platform developed for aerial drones. ROS and ROS2, frequently used for higher-level autonomy and sensor integration, originated in wheeled ground robotics and manipulator arm development. The assumptions baked into these frameworks reflect their origins: flat environments, GPS availability, and sensor modalities that have no direct subsea equivalent. Six degree-of-freedom motion in a dense fluid medium, acoustic positioning, water pressure as a sensor input, and sonars as a piloting tool all require workarounds rather than first-class support. These are not minor adaptations. They are fundamental mismatches between what marine robotics requires and what these frameworks were built to provide.

A common misconception is that open-source frameworks are easy to modify. In practice, core architectural decisions are made by the broader community and the organizations that drive it. Their priorities often have little to do with marine applications. Marine-specific requirements end up accommodated at the periphery, through workarounds and integration layers, rather than addressed at the architectural level. This introduces long-term maintenance burden and fragility. Upstream changes driven by unrelated use cases can break downstream integrations. Teams absorb the cost of re-integration work that delivers no new capability.

The navigation and control boundary is where the mismatch becomes most costly. Survey-grade inertial navigation systems represent significant capital investment, yet in fragmented architectures that data rarely participates directly in vehicle control. The control loop continues to rely on low-cost MEMS sensors embedded in the flight controller, while the high-grade INS runs as a parallel reporting layer. The result is degraded positioning consistency and data quality that cannot be resolved by investing in better sensors. The barrier is architectural.

Extending these systems compounds the problem. Adding new capability requires changes across multiple independent codebases: flight firmware, companion computer applications, middleware bridges, and user interfaces. None of these were designed to evolve together. Even incremental feature changes can require cross-layer regression testing. Mission-critical support relies on community forums and issue trackers. Architectural changes that would benefit marine deployments may never be prioritized by a community focused elsewhere.

Electrically, fragmented systems create failure domains with no isolation between them. A single overcurrent event, a flooded auxiliary light or a shorted-out peripheral sensor, can propagate across a shared power bus and collapse compute, control, and payload subsystems at once. In the field, this means a full vehicle reset rather than a localized fault. As vehicles operate in more remote locations and at greater depths, the costs escalate, best case is a vehicle reset, worst case is a component failure that causes the mission to be aborted. A lost dive offshore is not rescheduled the next afternoon. It means additional vessel mobilization, weather delays, and significant cost. Fault visibility is limited, root cause analysis is reactive, and repeat failures are hard to prevent. The resulting system is a collection of point solutions made to function together, not a platform designed from the start to operate reliably in commercial conditions.

III. Lifecycle Cost and Operational Risk

In fragmented integration architectures, engineering effort is distributed across many codebases. Feature expansion requires cross-layer changes across multiple repositories. This is manageable during early development. At commercial scale, it becomes a significant ongoing overhead. A particularly underappreciated risk is upstream fragility. Customizations built on open-source frameworks can be broken by upstream updates from a community with different priorities. Teams absorb the cost of re-integration work that adds no new capability and delivers no customer value.

The operational risk from fragmented architectures shows up differently for operators and OEMs, but the root cause is the same: a system that was not designed as a whole.

For Operators:

- Mission failure and aborted dives due to system resets or unreliable subsystem behavior
- Inconsistent or unusable data products requiring repeat deployments to re-acquire lost data
- Unplanned vessel time extensions when field troubleshooting cannot be resolved quickly
- Inability to diagnose root cause in the field, making repeat failures difficult to prevent

For OEMs:

- Extended commissioning and sea trial cycles due to cross-layer subsystem synchronization
- Engineering time consumed by firmware fork maintenance, middleware message maintenance, and integration debugging rather than customer-facing capability development
- Slow iteration cycles that delay time-to-market and reduce ability to respond to customer needs

- Bespoke integration work required for each new customer or vehicle configuration, limiting scalability

These costs compound over time and hidden integration overhead routinely outpaces initial bill-of-material savings. For operators, every aborted dive or repeat deployment is vessel time, crew cost, and delayed deliverables. For OEMs, every engineering cycle spent maintaining infrastructure is a cycle not spent on competitive differentiation or customer capability.

A unified operating architecture eliminates this overhead at the source. Engineering effort shifts from infrastructure maintenance to building capability. Operators complete missions more reliably and deliver better data products. Development velocity increases because control allocation, state estimation, electrical protection, and messaging do not need to be reinvented for each new platform or customer. Time-to-market becomes a function of mission design, not control-stack modification.

IV. Reliability-Focused Vehicle Operating Architecture

Mission Robotics addresses the integration challenge through two purpose-built products: Mission Dock, a fault-observable, domain-isolated electrical and communications backbone with paired edge compute (^{ec}), and NadirOS, a vehicle operating system built specifically for marine control, navigation fusion, and scalable autonomy. Both were designed for the subsea environment from the start. Neither is an adaptation of a framework built for a different domain.

NadirOS version 2 has been in commercial use for years. It has supported 24/7 over-the-horizon operations with tier-1 service providers, multi-kilometer internal pipeline inspections, aquaculture net repair, and as-built cable surveys. These are not demonstrations. They are sustained commercial deployments under real operational conditions. Version 3 extends this proven foundation into a scalable platform that shortens OEM development cycles and enables mission types that were not previously practical for vehicles of this class.

A. Mission Dock^{ec}: Fault-Observable, Domain-Isolated Electrical and Communications Backbone

Mission Dock is a single, validated board that serves as the common electrical and communications foundation across vehicle types and configurations. The edge compute module is selected to match the application, from lightweight control-only deployments to compute-intensive autonomy and vision workloads. Because the form factor and feature set are consistent, integration work done for one vehicle carries forward to the next. There is no starting from scratch with each new platform.

Mission Dock^{ec} consolidates sensor power distribution, electrical protection, and hardware interfacing into a single monitored backbone. All user ports have per-port

current limiting with fault feedback and software-controlled power enable/disable. In practice, this allows powering down thermally sensitive sensors during bench or deck testing without submerging the vehicle, or isolating a faulted peripheral without losing control or compute. Electrical anomalies are no longer silent. They are observable and recoverable.

The board includes reverse and overvoltage protection to 50V. Supercapacitors allow the system to ride through brownouts and prevent data corruption during power transients. These are not passive protections. A fault in a peripheral subsystem does not propagate to take down compute, control, or other payloads. Fault events are logged alongside state estimation and video streams, so the diagnostic loop between cause and effect is short.

The interface set covers the connectivity requirements of real marine vehicle integration:

- Built-in gigabit Ethernet switch
- Isolated RS-232 and RS-485 serial with per-port isolated 5V power
- Multiple UARTs with and without flow control
- 3× independent I2C buses, including two independently powered critical buses
- 2× independent CAN bus ports with termination
- GPIO, PWM, A/D, leak detection
- Onboard sensors: 6-axis MEMS gyro/accelerometer + 3-axis magnetometer, barometer, board temperature, and voltage and current telemetry across power rails

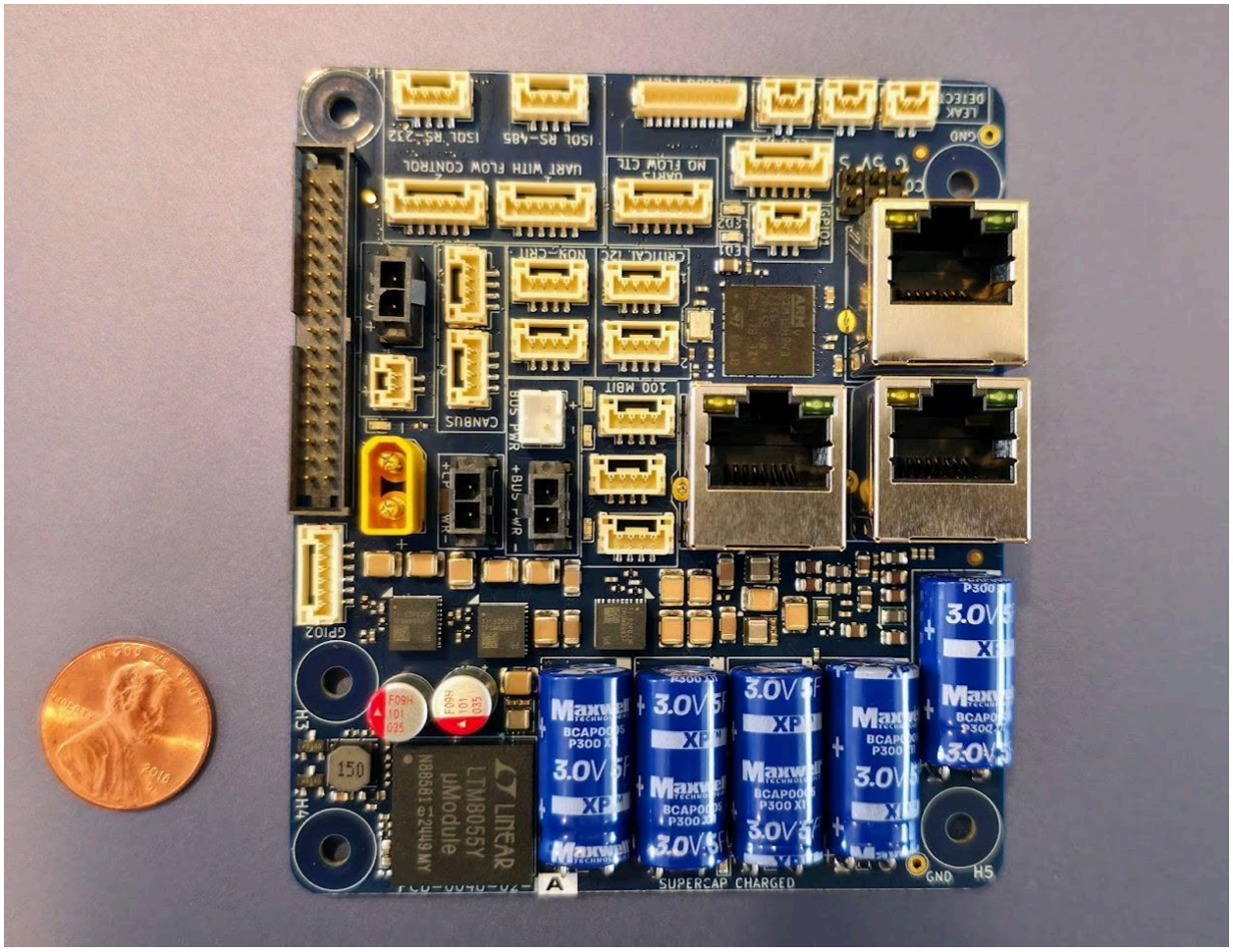

Figure 1. MissionDock; a single, validated board that serves as the common electrical and communications foundation across multiple vehicle types and configurations

B. NadirOS: Vehicle Operating System

NadirOS centralizes control allocation, state estimation, sensor fusion, recording, logging, camera and sonar data, and vehicle health into a single vehicle operating system. NavX is the vehicle control interface within NadirOS, providing operators with unified access to all control modes, sensor feeds, and mission parameters from a single UI. Vehicles are mission-ready in days, not months. The cross-layer integration dependencies that consume time in fragmented architectures are eliminated at the platform level.

Figure 2. NavX vehicle control interface within NadirOS

Control allocation is physics-based, supporting stabilization at any attitude without gimbal lock, across both body and global control frames. Full six degree-of-freedom control changes what a vehicle can do and how it is designed. The vehicle becomes the positioning mechanism. A hard-mounted photogrammetry or SLAM camera can be held at a precise angle and consistent distance from a target without mechanical articulation. A vehicle can match the curvature of a ship hull or aquaculture net and maintain a perpendicular orientation, which is directly relevant for cathodic protection measurement and net repair where geometry determines success. Basic control modes include attitude, heading, and depth hold. Advanced modes extend to station keeping, altitude hold, and velocity control.

State estimation uses a marine-specific Error-State Kalman Filter fusing AHRS, IMU, depth, and DVL sensors. Version 3 extends this to support higher-grade sensors including FOG INS and position resets via vehicle mounted GNSS or USBL systems. Drift-managed operations are native to the control network, not a workaround bolted on top. The key distinction is that navigation-grade sensor data is not a reporting layer. Any subsystem that needs it can consume it directly, whether that is the vehicle's own control loops, a survey pipeline, or a third-party process. A $100,000 fiber-optic gyro

(FOG) INS informs vehicle control, not survey data products. A vehicle conducting a pipeline survey can hold a precise heading and consistent standoff distance. The navigation quality flows directly into the data product quality.

Recording is time-synchronized across video, sonar, telemetry, and control inputs. In fragmented architectures, troubleshooting requires correlating separate logs after the fact. Here, everything is aligned. For end users this means faster reports, cleaner data for photogrammetry and as-built documentation, and a complete record of every mission parameter without manual reconstruction. It also shortens fault isolation and reduces the time from mission completion to deliverable. Beyond immediate troubleshooting, a unified record enables retrospective analysis. When a failure occurs, the full sequence of electrical, control, and sensor events leading up to it is available in a single correlated dataset. Over time, this supports pattern recognition across missions, identifying early indicators of component degradation or recurring fault conditions before they result in mission failure.

C. Open Standard Messaging and Scalable Autonomy

NadirOS is built directly on RTI Connext DDS, not on ROS2 or a ROS2-wrapped DDS layer. DDS is an open standard. It guarantees consistency, portability, and interoperability across vendors. Building on pure DDS, rather than through a higher-level abstraction, preserves the performance and QoS tunability that those abstractions compromise.

Because DDS is an open standard, OEMs and end users can subscribe and publish within defined QoS parameters without modifying the core vehicle software. This supports expansion into distributed systems, multi-vehicle coordination, and advanced features. It also enables rapid experimentation with other frameworks, including ROS2, without compromising the underlying reliability of the vehicle. Third-party autonomy processes, survey pipelines, and external systems participate natively.

Version 3 extends this into a scalable autonomy platform. Vehicle control loops run independently from autonomy and AI processes. Vision inference or waypoint generation does not degrade primary control performance. Autonomy is structured within the DDS framework rather than layered on top, so expansion into distributed systems and multi-vehicle coordination happens within the same architecture, not through rework.

V. OEM Case Studies

A. Purpose-Built Intervention ROV: 6-DOF Control Enabling New Vehicle Concepts

An OEM partner set out to develop a 1-2 man portable ROV for aquaculture net repair. The mission demands precise maneuvering near irregular net structures, perpendicular orientation to curved surfaces, and consistent tool delivery regardless of current or vehicle attitude. The vehicle concept was sound. The limiting factor was the control system. Net repair requires treating the vehicle itself as the positioning mechanism, not relying on a complex articulated arm to compensate for imprecise placement.

Six degree-of-freedom control was the enabling capability. The vehicle matches net curvature, holds perpendicular orientation, and maintains consistent standoff distance. The tooling is fixed and purpose-built rather than articulated. The same control quality that makes net repair viable carries over to cavi blasting, where operators report mission times approximately 50% lower than with legacy systems, driven by more precise positioning and less pilot-induced correction. The architecture also let the OEM move from concept to field-deployable system quickly. Development focused on the tooling and mission workflow. The result is a deployed commercial product operating in environments where diver intervention or net recovery were previously the only options.

B. Scalable OEM Platform: Inspection-Class Vehicles Doing the Work of Larger Systems

An OEM building inspection-class ROVs for professional survey and intervention markets has developed more than half a dozen distinct vehicle variants on a single operating architecture. Vehicles range in size and thruster configuration. Payloads span standard visual inspection cameras, SLAM cameras for survey work, Norbit Wingheads, and pipe tracker (TSS 660) + Sprint-Nav Mini INS for pre- and post-cable lay surveys. Each vehicle is mechanically different. All run the same software stack.

Inspection-class vehicles with survey-grade navigation and 6-DOF control can execute missions that previously required much larger, more expensive platforms. Support vessel size, crew, and operational cost all come down. When a new mission requirement comes in, the OEM does not start over. A new configuration is in the field in days, because control, state estimation, electrical protection, and recording are already solved. The customer gets a vehicle built exactly for their mission. The OEM can respond quickly without taking on new architectural debt.

C. Remote Operations: Reliability as a Prerequisite for Over-the-Horizon Deployment

A tier-1 subsea service provider operating ROVs in the 250-600kg class, deployed from unmanned surface vessels in open ocean conditions, represents the most demanding reliability environment in the industry. The vehicle is managed from a remote operations center over a satellite link. There is no crew on deck. There is no easy recovery. Bandwidth is limited. In this operational model, reliability is not a preference. It is the prerequisite for the model to work at all.

The architecture supports this directly. Predictable startup, observable fault states, and unified time-synchronized recording give remote operators full visibility into system state without physical access. Positional awareness, tether management, and subsystem health are available continuously. Unplanned resets and asynchronous fault conditions are eliminated, which removes the repeat diagnostic dives that fragmented architectures force on remote operations. Across sustained 24/7 operation, the operation shifts from reactive troubleshooting to predictable mission execution. That is the only basis on which over-the-horizon remote operations work at commercial scale. In one of their first campaigns, a planned 60-day offshore pipeline survey, they were able to complete the mission 8 days ahead of schedule.

VI. Discussion

Marine robotics is undergoing a fundamental transition. The prior generation of subsea work was defined by large vessels, large vehicles, and large crews. Market segments were primarily energy extraction and military. The systems that served them reflected those origins: high operational cost, limited adaptability, and architectures that have not kept pace with what is now technically possible. The industry broadly agrees that solutions have not advanced at the rate the technology allows.

The new era demands different things. Operators need to reduce costs by removing personnel from offshore environments, shifting to remote and semi-autonomous operations, and deploying smaller task-optimized vehicles. At the same time, the technology available is expanding rapidly. New propulsion, advanced manipulators, AI and machine learning, SLAM-based navigation, and improved communications are all becoming accessible to inspection-class platforms. The ability to adopt these technologies quickly is a competitive differentiator. Service providers and OEMs that cannot move fast are left with three options: buy existing vehicles and compromise on capability, assemble unreliable systems from fragmented stacks, or lose the opportunity.

The architectural distinction in this paper is the response to that transition. It is not about bespoke versus off-the-shelf components. Components will always come from multiple vendors and that is fine. The distinction is between loosely coupled point solutions with no shared system context and architectures designed as unified operating systems. A unified architecture does more than improve reliability. It can reduce component count, because a common data fabric means sensors do not need to be duplicated to serve multiple consumers. The same navigation data informs control loops, survey pipelines, and third-party processes simultaneously. It enables mission types that are not possible on fragmented stacks and it gives the industry the stable foundation it needs to adopt AI, SLAM, and advanced autonomy without rebuilding the vehicle platform each time.

Ultimately this comes down to time. Time to market. Time to complete missions. Time spent troubleshooting instead of operating. Each represents real cost: for OEMs, engineering burn rate and delayed revenue; for service providers, vessel time and

eroded profitability; for the industry, the pace at which new capabilities can actually reach the water.

VII. Conclusion

Mission Robotics addresses a structural problem in the industry. The components exist. The sensors, compute, and hardware to build capable vehicles are all available. What has been missing is the platform that binds them into systems that are reliable, scalable, and ready for the missions the industry now demands. Version 2 established that foundation, proven across years of commercial use, 24/7 remote operations, multi-kilometer pipeline inspections, and aquaculture intervention. Version 3 extends it into a platform capable of supporting autonomous, distributed, and survey-grade operations at scale.

The value looks different depending on who is evaluating it. For service providers and end users, it is operational: less time on missions, less time troubleshooting, better data products, and access to mission profiles that were not practical before. For OEMs, it is commercial: faster time to market, new vehicle configurations in days instead of months, a single software framework across an entire product line, and engineering time directed at differentiation rather than integration overhead.

Mission complexity is increasing. Remote and autonomous operations are expanding. Survey-grade data requirements are rising. Vehicles need a foundation that can support that trajectory without requiring an architectural rebuild at each step. The relevant measure is not hardware cost. It is the total cost of commissioning, maintaining, debugging, and scaling a platform across its operational lifecycle. Architectural coherence reduces downtime, lowers engineering burden, and improves commercial reliability. Commercial scalability is measured through uptime, repeatability, and predictable behavior under load. Moving from fragmented integration to a unified vehicle operating architecture is not an incremental improvement. It is what makes the next generation of marine robotics commercially viable.