BEST 2026 · ENSTA BRETAGNE · GUERLÉDAN LAKE, BRITTANY

Autonomous Navigation of a Differential-Drive Boat on Guerlédan Lake

Project report: compass calibration, GPS guidance, digital twin and lake trials

TEAM NAVYSEALS
Natalia Lach · Jakub Grygierczyk · Samuel Mastroscianni · Vamshidhar Reddy Enugala

SUPERVISORS
Prof. Luc Jaulin · Prof. Benoît Zerr · Antoine Morvan

DATE
October 2026

Presentation slides
vamshidharre.github.io/BEST_Brest/navyseals_presentation.html

Source code (GitHub)
github.com/vamshidharre/BEST_Brest

Boat code, logs & KML
github.com/vamshidharre/BEST_Brest/tree/main/NavySeals

ABSTRACT

We programmed a DD-Boat, a small autonomous surface vessel steered only by the speed difference of two thrusters, to navigate GPS waypoints on Guerlédan Lake. The work covered magnetometer calibration, projection of GPS fixes to a local metric frame, a proportional heading controller with a spin-in-place mode, and a digital twin used to test the real mission scripts before deployment. On the lake, the boat completed a 15 m box mission (4/4 waypoints, 111 s, loop closed within 2.8 m) and a high-speed sprint in which it reached a buoy 166 m away within 3.7 m. The return leg was cut 84.6 m before the finish when Wi-Fi dropped. We analyse 3,476 telemetry samples, compare them with the simulation, and list the changes these results call for.

1 Introduction and objectives

A DD-Boat has no rudder: turning comes from running the left and right motors at different speeds. Its only navigation sensors are a 9-axis IMU, used as a compass, and a GPS receiver giving about one fix per second. All control runs on board. The laptop is only used to start missions and download logs over the lake Wi-Fi. We set three missions of increasing difficulty:

MISSION	GOAL	SCRIPT	STATUS
1 · 15 m box	Visit four waypoints on a 15 m square and return to the start	`waypoint_mission.py`	**Completed on the lake**
2 · Sprint	Dock → A → B → A at full speed between two buoys 165.7 m apart	`waypoint_mission.py`	**B reached; return cut by Wi-Fi drop**
3 · Regatta	Cross two gates in the scoring direction as often as possible in 5 min	`regatta_loop.py`	**Built and validated in simulation**

2 Platform

- **Propulsion:** two brushless thrusters driven by an Arduino Mega 2560; no rudder.
- **Computer:** Raspberry Pi 3, Raspbian Stretch, Python 3.5.3 (no f-strings, no SciPy: NumPy only).
- **Sensors:** Pololu 9-DOF IMU (LIS3MDL magnetometer, LSM6DS33 accel/gyro) and a serial NMEA GPS.
- **Operations:** SSH over the lake Wi-Fi; CSV logs at about 9 Hz and KML tracks for Google Earth.
- **Vessels:** DD-Boat 11 for the lake trials, then DD-Boat 18, where the compass axis problem of §3.2 appeared.



3 Method

3.1 Boat model

We use the nautical convention (θ = 0 north, clockwise positive; x east, y north). The digital twin integrates this model; u_L, u_R are motor PWM commands and (w_x, w_y) a constant wind/current drift.

$$\dot{x} = v\sin\theta + w_x \qquad \dot{y} = v\cos\theta + w_y \qquad \dot{\theta} = k_1(u_L - u_R) \qquad \dot{v} = k_2(u_L + u_R)/2 - c\,v|v|$$

3.2 Compass calibration and the axis fix

Nearby iron (screws, motors, battery) shifts and squashes the magnetic field, so a full turn of the boat traces an off-centre ellipse instead of a circle. From one slow 360° turn we fit the offset b and the matrix A, then correct each reading:

$$m = A^{-1}\cdot(m_{raw} - b)$$
$$\theta = \operatorname{atan2}(-m_y, m_x) \quad \text{heading, marine convention}$$

Field problem. After moving our code to Boat 18, North and South were right but East and West were swapped. The magnetometer's Y axis points the opposite way to the nautical frame. Negating m_y fixed it in `calib.py`, `waypoint_mission.py` and `regatta_loop.py`. Calibration is specific to each hull.

3.3 GPS to local metres

Over a few hundred metres the Earth is flat enough for an equirectangular projection around a fixed origin (lat_o, lon_o) = (48.199111° N, 3.014930° W), with R = 6 371 km:

$$x = R\cdot\cos(lat_o)\cdot(lon - lon_o) \quad \text{east}$$
$$y = R\cdot(lat - lat_o) \quad \text{north}$$

BUOY	LATITUDE, LONGITUDE	X, Y (M)
A shore	48.198863, -3.015070	-10.4, -27.6
B lake	48.198259, -3.017114	-161.9, -94.7

A → B is 165.7 m on a bearing of 246.1° (return 66.1°). To avoid acting on stale data, the GPS serial buffer is drained every cycle and only the newest fix is used.

3.4 Guidance and control

The bearing to the active waypoint is compared with the compass heading; the error is wrapped to ±180° with a sawtooth function and fed to a proportional law on the two motors:

$$\theta^* = \operatorname{atan2}(x_t - x,\ y_t - y) \qquad e = \operatorname{sawtooth}(\theta^* - \theta) \qquad u_L = v_0 + K_p\cdot e, \quad u_R = v_0 - K_p\cdot e$$

Both commands are clipped to [0, 115]. When |e| exceeds 40° the boat switches to a **spin-in-place** turn ($u_L = -u_R$, ±75 in the box run and ±95 in the sprint) and stays in it until |e| < 20°. The hysteresis stops it from switching back and forth. A waypoint is reached when the boat is within the arrival radius, or when it has come within 3.5 m and crossed the line through the waypoint perpendicular to the leg ($\langle b - a,\ m - b\rangle > 0$). This catches near misses without circling back. Box settings: v_0 = 70, K_p = 28, radius 2.5 m. Sprint settings: v_0 = 100, K_p = 35, radius 3.5 m.

3.5 Digital twin

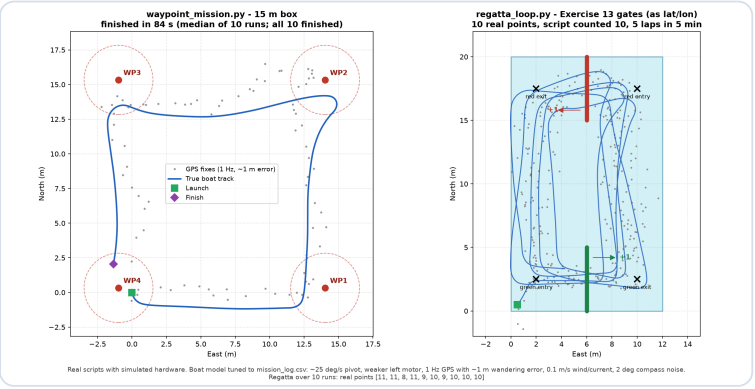


Fig. 1. The unmodified mission scripts driving a simulated boat (1 Hz GPS with ~1 m error, 2° compass noise, 0.1 m/s drift, weaker left motor). Left: 15 m box. Right: regatta gates.

Following A. Morvan's method, we swapped the hardware drivers for the model above with realistic sensor noise and ran the **real** mission scripts against it. In 10 runs every box mission finished (median 84 s). The regatta loop scored 8–11 points per 5-minute race, its own count matching the true gate crossings. On the real buoy coordinates, the twin predicted about 140 s for A → B.

4 Field results

4.1 Mission 1: 15 m box (DD-Boat 11)

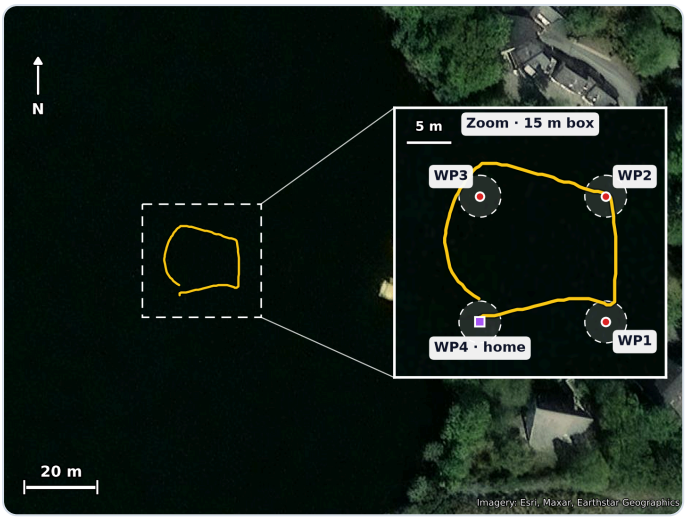


Fig. 2. Logged GPS track of the box mission on satellite imagery, with a zoom on the 15 m square. Dashed circles: 2.5 m arrival radius. Data: box_mission_track.kml, box_mission_log.csv.

4/4
waypoints reached

111 s
total time (sim: 84 s)

2.8 m
finish to start distance

0.59 m/s
mean speed, 65 m path

LEG	TIME (S)	CLOSEST (M)
Start → WP1 (15 m east)	26.7	2.5
WP1 → WP2	19.6	2.6
WP2 → WP3	30.7	3.4
WP3 → WP4 (home)	32.5	2.8

WP3 was validated by the line-crossing rule (closest pass 3.4 m, outside the 2.5 m circle). Corners are rounded because the boat turns while still moving.

4.2 Mission 2: Dock → A → B → A sprint

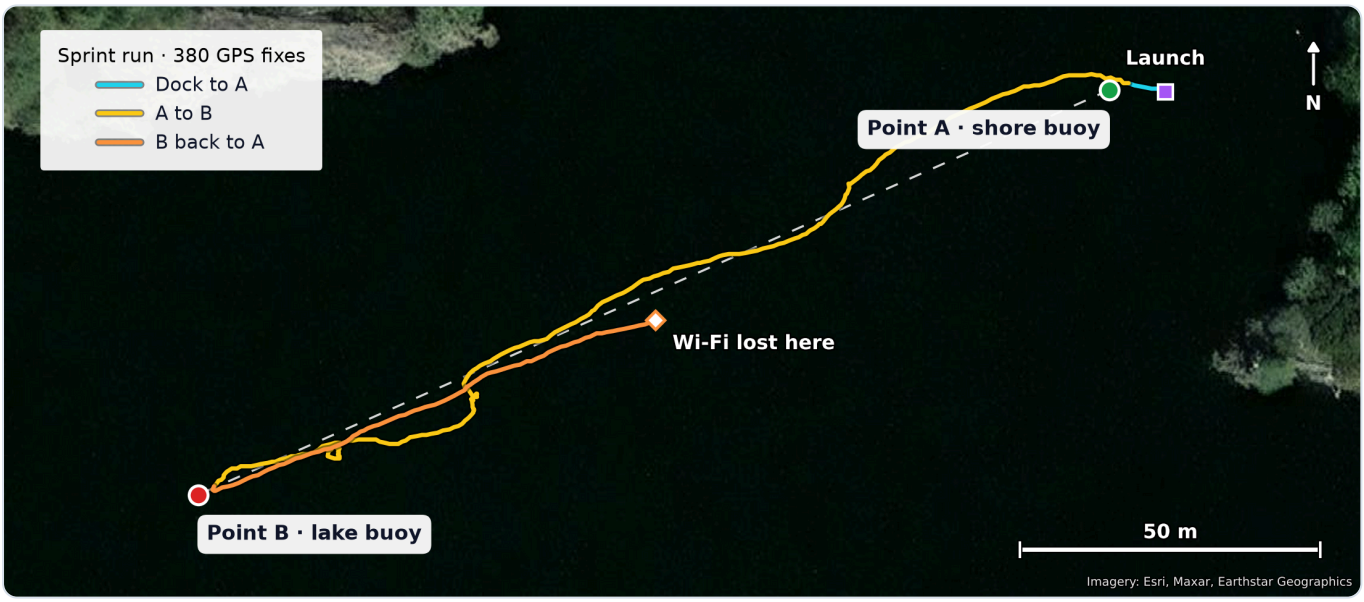


Fig. 3. Sprint track from 380 GPS fixes (mission_track.kml), coloured by leg. White dashes: the ideal A–B line. Imagery © Esri, Maxar, Earthstar Geographics.

LEG	STRAIGHT-LINE	TIME	SPEED MADE GOOD	SPIN TURNS	OUTCOME
Dock → A	9.4 m	11.7 s	0.44 m/s	1	**Reached**
A → B	165.7 m	280.9 s	0.59 m/s	72 (81 s)	**Reached B within 3.7 m**
B → A	165.7 m	88.2 s	0.89 m/s	10	**Stopped 84.6 m from A**
Total		**381 s**			280 m travelled · 3,476 log rows

Why the run stopped. The mission was started directly in the SSH session. When Wi-Fi dropped on the return leg, the script and its log stopped with it, 84.6 m before A. The boat was still on course at near full power (u ≈ 101/98). Running missions inside `screen -S sprint` keeps them alive through a disconnect. The planned re-run was cancelled because of heavy wind and rain.

4.3 Telemetry analysis

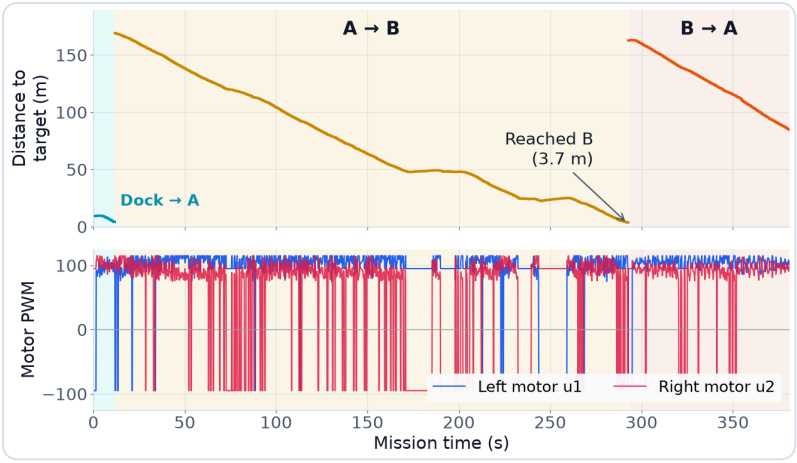


Fig. 4. Sprint telemetry (**mission_log.csv**). Top: distance to the active waypoint. Bottom: motor commands; drops to −95 mark spin-in-place turns.

The outbound leg was the weak point. The heading error crossed the 40° threshold 72 times, and the boat spent 81 s (29% of the leg) spinning in place. At 0.59 m/s made good, the leg took 281 s, twice the 140 s the twin predicted. The two plateaus in the distance curve, around 170–200 s and 230–260 s, are where the boat stalled, spinning and drifting. On the way back, the same controller made only 10 spin turns and held 0.89 m/s with both motors near full power. The code and gains were identical in both directions, which suggests the outbound heading was fighting the wind and waves rather than a software fault.

5 Reliability and safety

- **GPS watchdog:** no fix for 3 s sets both motors to zero.
- **Track autosave:** the KML is rewritten every 5 s, so a power cut loses almost nothing.
- **Emergency stop:** `stop.py` forces (0, 0) immediately.
- **Auto download:** a laptop script waits for the boat to reconnect, then copies the KML and CSV.
- **Detached sessions** (lesson from the sprint): starting missions inside `screen` keeps them running through a Wi-Fi drop.

6 Discussion

Simulation vs lake. The twin was right about behaviour (all waypoints reached, correct gate logic) but optimistic about time: 84 s vs 111 s for the box, 140 s vs 281 s for A → B. Its wind and wave model is too gentle, and spin turns cost more on real water.

Sensors. One sign error swapped East and West on a new boat: each hull needs its own calibration and a four-point compass check.

Operations. The sprint was limited by how the script was launched, not by the controller.

7 Conclusion and future work

We built and tested a full GPS navigation stack for the DD-Boat: compass calibration, local projection, a hysteretic proportional controller, line-crossing waypoint validation, safety watchdogs and logging. It completed the 15 m box on the lake and reached the far buoy of the 166 m sprint within 3.7 m. Next steps: (i) re-run the sprint inside `screen` to record the full A → B → A time; (ii) cut spin turns on long legs by tuning the 40°/20° thresholds and adding cross-track or wind-drift correction; (iii) race the regatta loop on the water; (iv) feed the measured speed and turn rate back into the twin (k_1, k_2) so its time predictions match the lake.

8 Reproducibility

Every figure is generated from the logged data in the repository. `python make_slide_images.py` rebuilds Fig. 2–4 from `NavySeals/box_mission_log.csv` and `NavySeals/mission_log.csv`; `python report/build_report.py` rebuilds this PDF. On the boat (Python 3.5.3), missions are started with `screen -S sprint python3 waypoint_mission.py`, and `python3 stop.py` is the emergency stop.

9 Resources and references

1. Slides: vamshidharre.github.io/BEST_Brest/navyseals_presentation.html · Code: github.com/vamshidharre/BEST_Brest
2. L. Jaulin, *DD-Boat introductory course* (best_ddboat.pdf), ENSTA Bretagne, BEST 2026: calibration, control law and finish-line test.
3. L. Jaulin, *Mobile robotics / drones* textbook, ch. 4 (DD-Boats), p. 28: local GPS projection.
4. A. Morvan, *Digital twins and observers*, lecture slides, ENSTA Bretagne, 2026.
5. Esri World Imagery basemap (Esri, Maxar, Earthstar Geographics), used for Fig. 2–3.