7th International Conference on Innovation in High Performance Sailing Yachts and Wind Assisted Ships.
InnovSail 2026, Gothenburg, Sweden, 3-5 June 2026.

Sail-CV : an open-source project for embedded computer vision measurement of sails aerodynamic

Esteban Foucher[1 1], Valentin Foucher[2], and Ulysse Dhomé[3]

[1]Independent, France, former French America's Cup performance engineer
[2]Independent, France, former French America's Cup performance engineer
[3]Wallenius Marine, Sweden

Abstract. This paper presents a fully automated computer vision framework for real-time aerodynamic analysis and shape monitoring of sails. The framework (`https://github.com/estebanfoucher/Sail-CV`) is composed of two primary, interoperable modules designed for embedded hardware:

Tell-Tale Tracking: A machine-learning pipeline that detects, tracks, and classifies the aerodynamic state of tell-tales using a single onboard camera. It utilizes a real-time detection transformer (RT-DETR-L) for single-class localization, a layout-aware tracker grounded by the Hungarian algorithm, and a decoupled state-determination module. This design ensures generalization across various sail materials, colors, and lighting conditions.

3D Sail Reconstruction: A stereo-vision pipeline that generates metric 3D point clouds of the sail surface without requiring applied textures or physical markers. By leveraging MASt3R foundation model for dense pixel-to-pixel matching and triangulation, the system accurately captures surface curvature. On-the-ground precision validation tests are presented, alongside with real sail reconstruction example.

Both modules operate continuously with no human-in-the-loop once initially calibrated. At low update rates, the 3D reconstruction module can dynamically update the geometric grounding for the tell-tale tracker, compensating for sail deformation. To lower the barrier to entry and encourage further development, the purpose-built datasets and pre-trained models are published as an open-source base for the community.

Keywords: computer vision; 3d reconstruction; detection; tracking; sailing performance.

1 Introduction

Sail performance depends strongly on two quantities that are still difficult to measure continuously on board: local flow attachment and full sail shape. In practice, sailors mainly rely on visual inspection of tell-tales and qualitative interpretation of sail deformation. This works for expert intuition, but it does not produce structured, time-aligned data that can be analyzed live or after navigation or used by automated control systems.

This paper presents **Sail-CV**, an embedded computer-vision framework designed to make sail analysis quantitative and automated. The system combines two complementary modules: (i) real-time tell-tale detection, tracking, and state estimation from onboard video, and (ii) metric 3D reconstruction of sail geometry from calibrated stereo cameras. The objective is to provide a practical tool for live monitoring while also establishing an experimental and academic baseline for future work.

For tell-tale monitoring, prior approaches in sailing are mostly based on heuristic image processing pipelines (color thresholding, morphology, hand-tuned rules), which are sensitive to lighting and setup

[1]estefoucher@gmail.com

changes (Dhomé et al., 2023). In parallel, machine-learning progress in vision has made object detection and tracking robust across various environments. The presented approach leverages this progress to produce per-tell-tale state time series at usable embedded rates, with explicit identity grounding to sail layout positions.

For sail-shape measurement, existing solutions typically sit at two extremes: low-cost monocular and heuristic approaches with strong case dependance (specific pattern on sail), and high-end sensing systems (e.g., LiDAR) that remain costly, hence difficult to integrate on small or medium platforms. Recent 3D vision foundation models (e.g., MASt3R (Leroy et al., 2024), VGGT (Wang et al., 2025)) open a new path by improving dense correspondence and reconstruction robustness. Sail-CV builds on these advances with a calibration-first, fixed-camera pipeline tailored to marine constraints and embedded operation.

Both modules are designed for GPU acceleration, with the evaluations in this article executed on an NVIDIA Jetson Nano. Driven by the rapid expansion of the robotics industry, embedded computing power is currently experiencing exponential growth. The marine and sailing sectors will be able to capitalize on this GPU-driven hardware evolution to enable robust, real-time onboard measurement systems.

Beyond methodology, the project is released as an **open-source suite** (code, datasets, and pre-trained models) to lower adoption barriers and support reproducible research. The long-term goal is a reusable instrumentation base for sailors, engineers, and researchers.

2 Tell-tale tracking

2.1 Introduction

Tell-tales are lightweight ribbons affixed to a sail surface whose orientation reflects the local boundary-layer state: a tell-tale aligned with the flow indicates attached flow, while a fluttering or lifted tell-tale signals separation. Despite their ubiquity in sailing practice, no tool currently records tell-tale behavior in a quantitative, continuous form suitable for live or post-navigation analysis.

The framework presents a modular, ML-based pipeline that detects, tracks, and classifies tell-tales from a single onboard camera. The system is designed for embedded, real-time inference with minimal hardware, and generalizes across sail types, colors, and environmental conditions. The trained model and dataset are published to enable fine-tuning on new setups with minimal effort.

Previous attempts at automating tell-tale or tuft analysis fall into two categories. In the sailing domain, Dhomé et al. (2023) used a classical pipeline — HSV color thresholding, morphological operations, and a hand-crafted edge-crossing heuristic — to classify tell-tale states on a rigid wing sail. This approach is inherently tied to a specific color and lighting setup and requires per-installation tuning. In addition, electronic tell-tales were developed but showed a practical barrier in the cost and installation. In aerospace, Lee et al. (2023) proposed a learning-based system for tuft segmentation on helicopters and UAVs, combining active-learning detection, uncertainty-driven label propagation, and weakly supervised segmentation to minimize annotation cost. Their system targets offline post-processing and per-case MLOps loop training rather than having generalist foundational approach.

The approach presented here differs in three respects. First, a generalist detector, classifier and a purpose-built dataset are published, enabling direct use for practitioners. Second, fine-tuning pipelines are proposed for improvement with minimal data on specific setups. Third, the **decoupling** of detection and state determination allows each component to be developed and swapped independently — for instance, the state-determination method can be adapted to a new use case without retraining the detector. In an end-to-end automated deployment, no human intervention is required once the tell-tale layout is established.

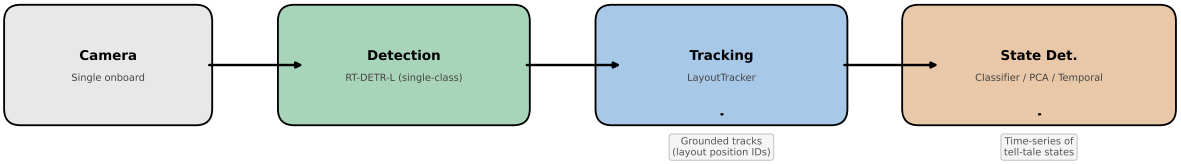


Figure 1. Pipeline diagram — Detection → Tracking → State determination

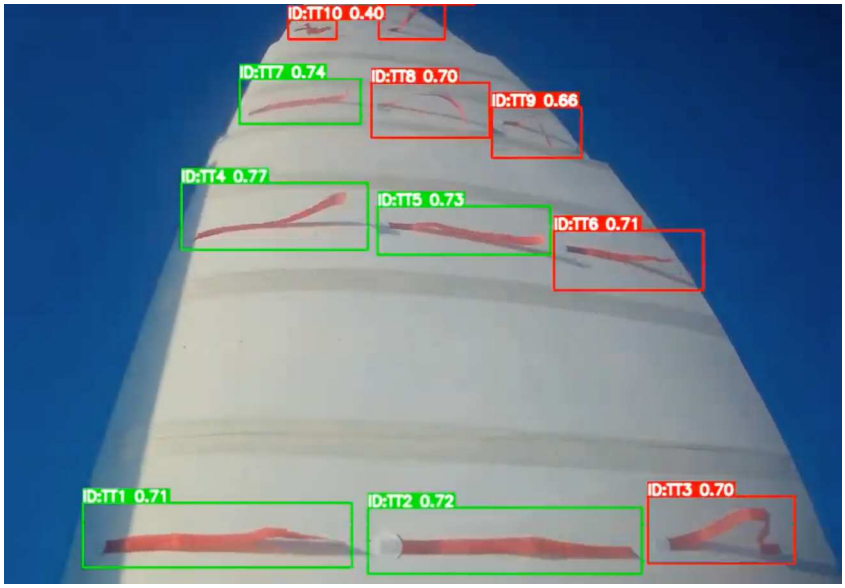


Figure 2. Full-sail inference example — tell-tale detections grounded to layout IDs. Colors are state : green is attached, red detached.

2.2 Pipeline overview

The pipeline follows a three-stage architecture: **detection**, **tracking**, and **state determination** (Fig. 1). A key design choice is that the detector is single-class — it locates all tell-tales regardless of their aerodynamic state, leaving state determination to a separate, downstream module. This decoupling was motivated by the inherent ambiguity in defining tell-tale states. Because the transition between "attached" and "detached" is a physical continuum, the visual boundary between these labels is highly subjective. This lack of a clear visual limit inevitably introduces inconsistencies during the human annotation process. When a multi-class detector is trained to jointly localize and classify, the contradictory training signals in this ambiguity zone degrade the model's overall detection performance. Decoupling the tasks ensures that the subjective noise of state classification does not penalize the detector's fundamental ability to locate the objects.

Figure 2 shows an on-water example: detections are grounded to layout IDs and color-coded by state using a classifier trained on attached/detached only.

2.3 Detection

RT-DETR-L, a real-time detection transformer, is used as the backbone detector. The model is trained in single-class mode at 640×640 input resolution.

2.3.1 Dataset

Composition The training dataset comprises two complementary sub-datasets, as shown in Tab. 1. Each instance is labeled with one of three state classes — attached, detached, leech — used for monitoring balance and for downstream state determination (not for detection training).

Table 1. Composition of the training dataset for the tell-tale detection and tracking.

Sub-dataset	Images	Instances	Attached	Detached	Leech	Description
Sailboat	524	1850	1170	485	195	Tell-tales on sails, primarily online imagery.
Home	249	1416	542	809	65	Tell-tales on non-sail surfaces (furniture, walls, appliances).
Fused	**773**	**3266**	1712	1294	260	Union of both sub-datasets

The fused dataset ensures generalization: a model trained on both sub-datasets performs at least as well on sails as one trained on sailboat images alone, while additionally detecting tell-tales on unseen surface types. Figure 3 shows annotated sample crops from the sailboat sub-dataset.

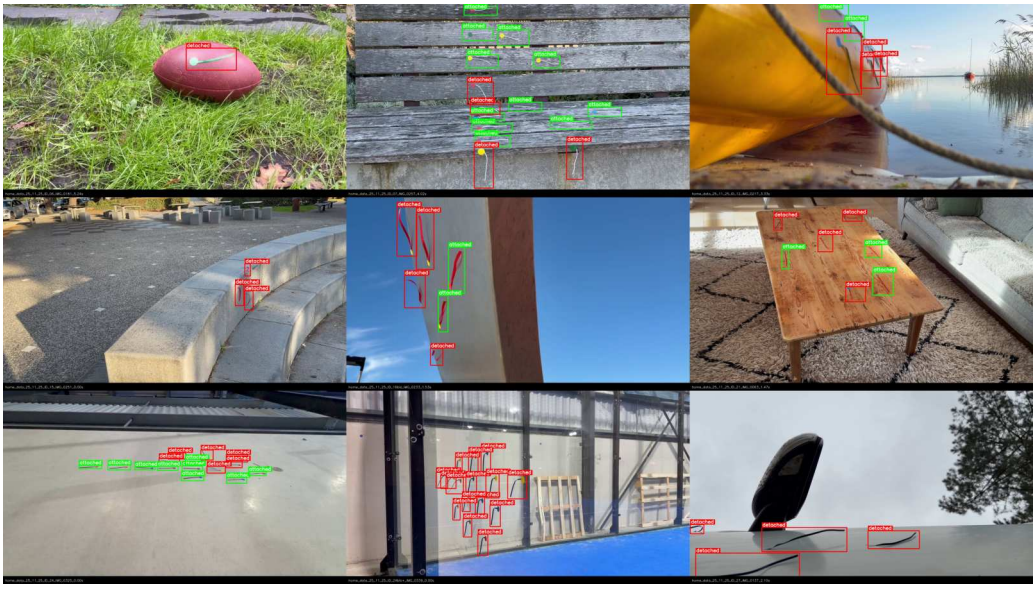


Figure 3. Dataset overview — bounding boxes overlaid on sailboat dataset images (green attached, red detached). All images come from our Home dataset.

Annotation protocol Bounding boxes are annotated directly with the three state classes (attached, detached, leech). For detector training, the labels are narrowed to a single class (tell-tale) whereas the three-class labels can be used to train the state classifier and to monitor dataset balance.

The annotation guidelines define a tell-tale as **detached** if it exhibits any of: wavy form, non-horizontal orientation, curvature, spacing from the surface, or significant motion blur. A tell-tale is labeled **attached** otherwise. **Leech** denotes a tell-tale that is not stuck to the surface but attached to an edge (e.g. trailing edge) and floating in the air.

Dataset augmentation To enhance dataset's volume and representativity of a wider set of real conditions, training images are augmented offline. A pipeline (random chain of transforms) emits a fixed number of variants per image and propagates bounding boxes through every transform. The chain combines geometric operations (flips, rotations, affine scale/translate, random scale), photometric operations (HueSaturationValue, RandomBrightnessContrast, RGBShift, CLAHE), sensor noise (ISONoise, GaussNoise, CoarseDropout), light global blurs, and outdoor/on-water effects (RandomShadow, RandomFog, RandomRain, RandomSunFlare, RandomSnow, RandomGravel). To mimic the rapid flapping of telltales without softening the rest of the scene, we further apply a bbox-local blur. Figure 4 shows a harsh-preset augmentation grid on a representative test-boat frame.

A small-object preset drops global blur, fog, snow, dropout and noise, and keeps at most one weather effect via OneOf(sun, shadow, rain) to preserve readability for tiny targets. One can typically use the small-object preset on the general sail-home-fused dataset and apply harsher presets on its own custom finetuning data.

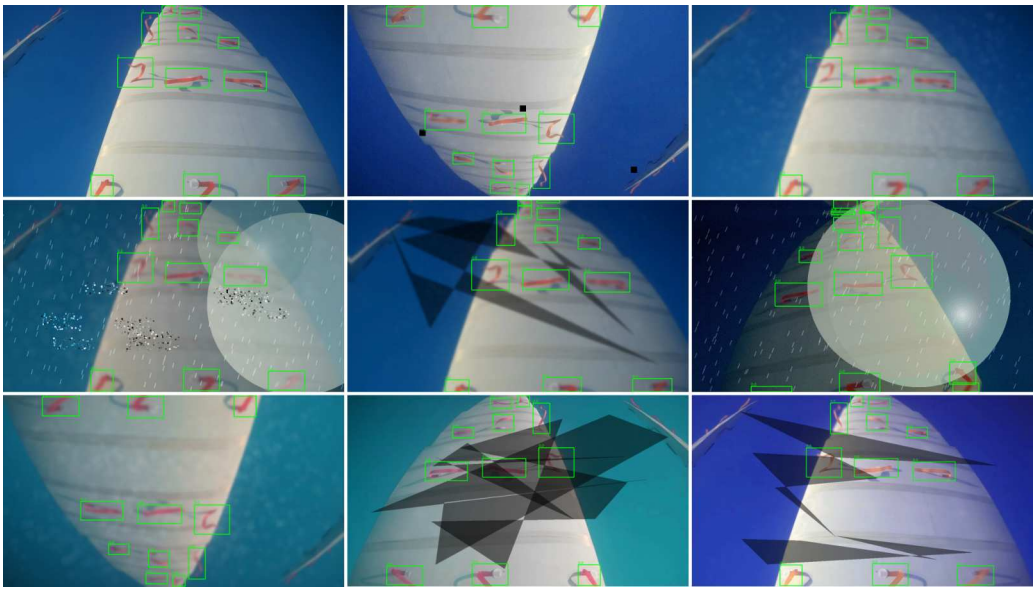


Figure 4. Example of one image (from test case boat) augmentation in a harsh preset. Upper left is original image, all others are augmentation variants.

Train/validation split Images are split into training and validation sets using group-based splitting: images originating from the same video or scene are assigned to the same group to prevent data leakage. The training dataset contains 688 images (496 from the sailboat sub-dataset, 192 from the home sub-dataset), with a total of 2694 instances of individual tell-tales. The instances contain 1462 attached tell-tales, 984 detached and 248 are leech. The validation is done on 85 images (28 from sailboat, 57 from home), with a total of 572 instances (250 attached, 310 detached and 12 leech).

2.3.2 Results

Training on raw sail-home-fused-dataset ran for 84 epochs before early stopping (patience of 40 epochs). On the fused validation set (572 instances), the best checkpoint achieves a precision of 0.723, a recall of 0.555, and a mAP50 of 0.576. For readers unfamiliar with detection metrics: **precision** is the fraction of detections that are correct (here 72.3%); **recall** is the fraction of ground-truth tell-tales that the detector finds at that confidence threshold (here 55.5%); **mAP50** (mean average precision at 50% overlap) is a standard object-detection metric that rewards both finding most targets

and localizing them with bounding boxes that overlap ground truth by at least 50%.

The resolution challenge Tell-tales are thin, elongated objects: one dimension (width) is typically much smaller than the other (length). This aspect ratio makes them particularly vulnerable to information loss at low spatial resolution — unlike compact objects, a tell-tale can be long enough to be visually apparent yet too thin for the detector backbone to resolve. This effect is quantified on the validation set (572 ground-truth bounding boxes at 640 px inference resolution). Counting a ground-truth box as detected if it has at least one matching prediction at an Intersection over Union (IoU) ≥ 0.5 (regardless of confidence), 434 are correctly detected and 138 are missed (24% missed; recall 75.9%). The missed detections concentrate overwhelmingly at small minimum bounding-box dimensions, as seen in Fig. 5). Filtering out ground-truth boxes with a minimum dimension below 8 pixels raises this recall from 0.759 to 0.826 and mAP50 from 0.677 to 0.747. Extending this threshold to 12 pixels yields a mAP50 of 0.753, confirming that small bounding boxes set a performance ceiling.

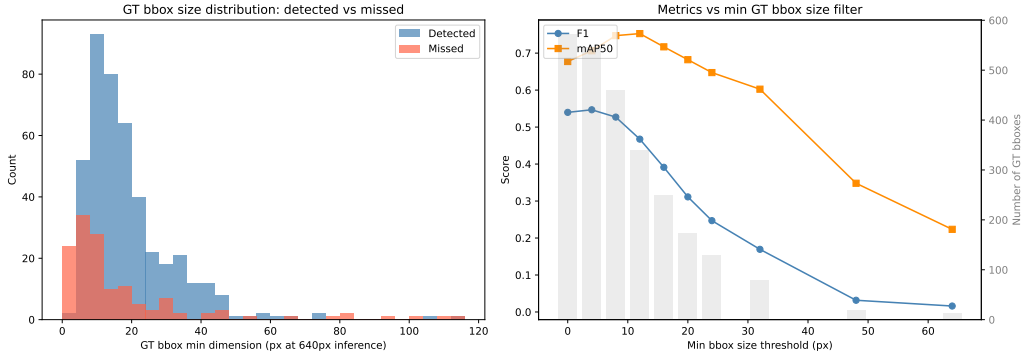


Figure 5. Resolution analysis — (a) GT bbox size histogram, detected vs missed; (b) mAP50 and F1 vs min size threshold

To isolate whether this ceiling stems from model capacity or input resolution, the same architecture trained at 1088 px is compared: mAP50 rises from 0.577 to 0.708 (+13 points). This confirms that the bottleneck is spatial resolution, not model expressiveness. Nevertheless, the 640 px model is recommended for deployment: computational complexity in RT-DETR scales quadratically with input resolution due to self-attention, and the local context at 640 px is sufficient to identify a tell-tale. When the camera setup produces tell-tales smaller than approximately 10 pixels on the thin dimension at inference resolution, inference on a crop of the image is a way to increase inference resolution.

2.4 Tracking

Standard multi-object trackers assign arbitrary, floating identifiers to detected objects. In the tell-tale monitoring use case, however, the scene contains a known, fixed set of tell-tales at predefined positions on the sail. The requirement is not merely to maintain identity across frames, but to **ground each track to a specific layout position** — enabling the user to associate measurements with physical tell-tale locations.

A **LayoutTracker** is used, which uses the Hungarian algorithm to assign detections to predefined layout positions at each frame. The assignment cost combines spatial proximity and detection confidence:

$$c_{ij} = \alpha \cdot d_{ij} + \beta \cdot (1 - s_j)$$

where d_{ij} is the normalized distance between layout position i and detection j, and s_j is the detection confidence. Track identifiers correspond directly to layout positions, producing stable, semantically

meaningful labels throughout a sequence.

Two variants handle sail deformation: a **1D tracker** compensates for deformation along one axis (boom angle), while a **2D tracker** accounts for axis and twist deformation. At lower update rates, the 3D reconstruction module (Sec. 3) can provide geometric grounding for layout position estimates.

2.5 State determination

Once a tell-tale is detected and tracked, its aerodynamic state must be determined. The visual definition of "attached" versus "detached" varies across use cases — sail material, tell-tale color, viewing angle, and wind conditions all affect appearance. The pipeline therefore offers several pluggable state-determination methods:

- **Classifier**: a classifier trained on cropped tell-tale regions. Simple to deploy and finetune, with limited control over the decision boundary due to its binary output.
- **PCA with segmentation**: the tell-tale is segmented (using SAM, morphological snakes, or Gaussian mixture models), its principal axis is computed via Principal Component Analysis (PCA), and a criterion on the axis angle determines the state. Heavier but more interpretable and tunable.

2.6 Application to the aerodynamic qualification of the flow on the rigid wing of a model wind powered vessel

2.6.1 Test Case and Method

The tell-tale tracking was tested on the data of Dhomé et al. (2023). It consists of videos of tell-tales fitted to the four rigid wings of the scaled model of the *Oceanbird* wind powered car carrier project developed at KTH Royal Institute of Technology (Dhomé et al., 2025), as visible in Fig. 6. Each wing is fitted with one camera at each side of the root of the wing, recording the 11 tell-tales per side. In (Dhomé et al., 2023), two examples of maneuvers were presented. First a "double sweep" of the wing, where the angle of attack of one wing at a time was varied by $\pm 25^{\circ}$ from its starting angle of attach twice. Second, a zigzag maneuver of the ship with the wings fixed at a constant angle, which makes the ships heading vary by approximately $\pm 15^{\circ}$. The same "double sweep" and zigzag examples as presented in (Dhomé et al., 2023) are used for comparison with Sail-CV.

Two methods were tested for the state determination: the PCA analysis and the classifier. From an aerodynamic point of view, accessing the tell-tale angles via the PCA method is very interesting, since it would enable for a more precise flow analysis that only deciding between and "attached" or "detached" state of the tell-tales. However, the tell-tales used in these experiments have a rectangular shape, which prevents them to simply "follow" the flow on the wing like the tufts traditionally used in wind tunnel tests. Their shape tends to make them snap from attached to detached, therefore for the PCA analysis method, a threshold of the angle value was used to determine the state of the tell tales as between attached and detached. Different values of threshold were tested to determine a global value for all tell-tales that matched the visual determination of the tell-tale state.

The classifier was first used with the "basic" training, however, since it was mostly made of tell-tales that were not rectangular and seen from a very different point of view, it didn't perform well. A fine-tuning of the classifier with a few images taken from the the experiments with the scaled ship model was hence performed. The measurements described in (Dhomé et al., 2023) happened over several days in different weather conditions, therefore, 13 random snapshots of some of the cameras with tell-tales either attached or detached were taken from different days to get different lighting conditions (Fig. 7). The training used 10 of these images and 3 were used for validation, totalling 143 individual

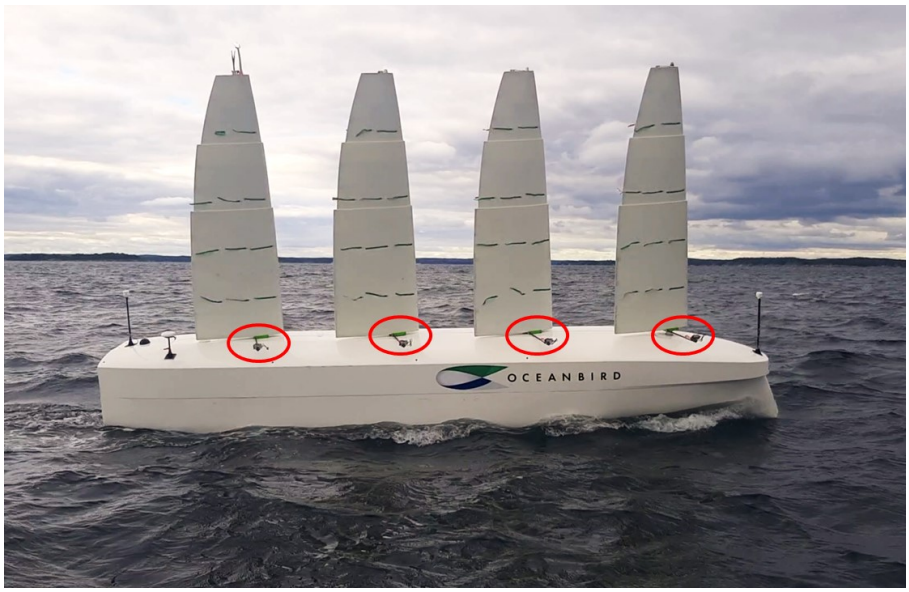


Figure 6. Picture of the test boat while sailing in Stockhom's archipelago. The cameras are highlighted by the red ellipses. Reproduced from (Dhomé et al., 2023).

tell-tales images, with 80 detached and 63 attached. The classifier was trained for 50 epoch with these extra images. This was enough to make the classifier visually good at detecting the state of the tell tales.

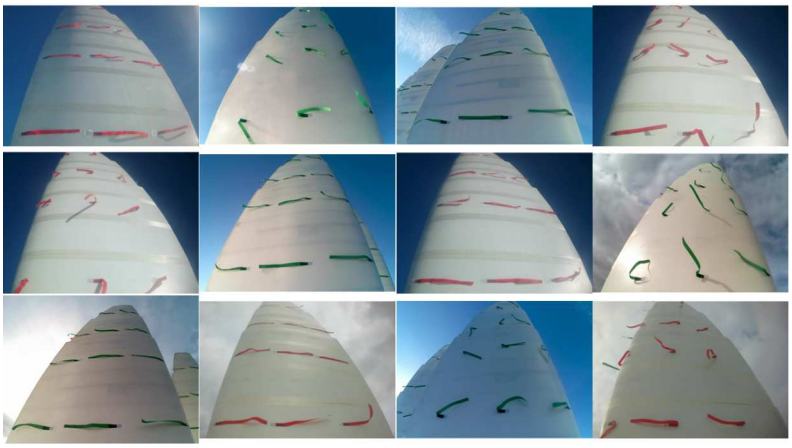


Figure 7. Some of the images used for the fine-tuning of the classifier. The missing image contains red, detached tell-tales.

This application differs in what Sail-CV was thought for to begin with in the sense that the cameras are fixed to the wing, so the tell-tales are always at the same position. However, since there are several wings, sometime one camera see the tell-tales of another wing. The tracker of the tell-tale position was sometimes confused by the apparition of "new" tell-tales when the wing was moving, which lead to wrongly moving the tell-tales to another wing. To avoid tuning Sail-CV too much to this case, the tracker was kept the same, but a mask was applied to the videos in order to hide the other wings.

2.6.2 Results

The performance of Sail-CV to determine the sate of tell-tales is compared using the same metric, called "Attached Flow Ratio", as what was presented by Dhomé et al. (2023), hereafter referred to as "Box method". Each tell-tale is assigned the value 1 when it is attached and 0 when it is detached.

The mean of 11 tell-tales over the wing is then calculated to determine an overall estimate of the state of the flow over each wing.

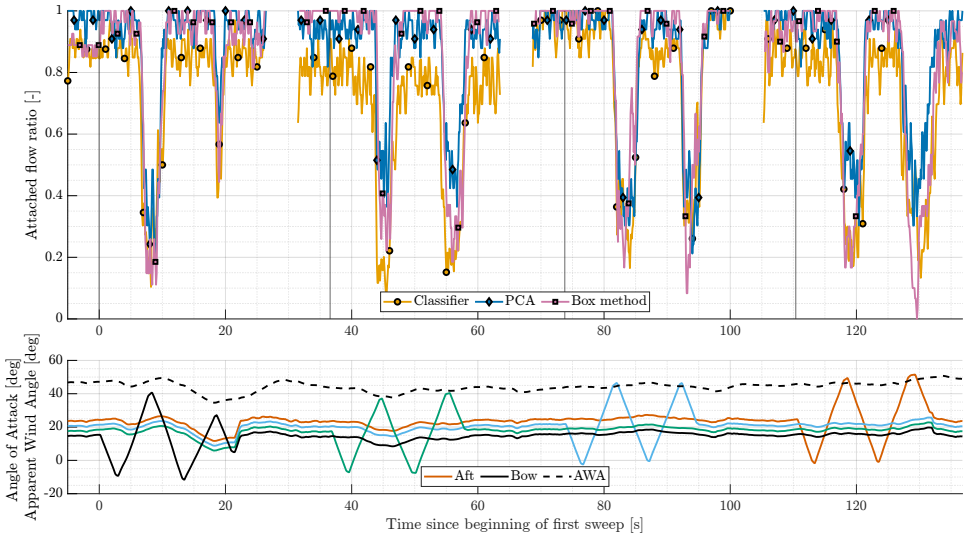


Figure 8. Comparison of the Attached Flow Ratio during a double-sweep of each wing one after the other. In the upper plot, The blue lines with rhombus markers show the attached flow ratio based on the PCA analysis; the yellow ones with round markers are for the Classifier method and the pink lines with square markers are for the Box method of Dhomé et al. (2023). The lower plot shows the angle of attack variation of each wing, together with the apparent wind angle at which the boat was sailing. The wings were swept in order starting from the one near the bow and finishing with the aft one.

Figure 8 shows a comparison, for the sweep example, of the box method from (Dhomé et al., 2023), the PCA-analysis method and the direct classifier. A clear result visible in Fig. 8 is that all three method have overall comparable results, as they all determine correctly the almost fully detached state of the flow over the wing when the angle of attack is too large, as well as the subsequent reattachment when the angle is lowered again. The results obtain with direct classification differ more from the two other methods, particularly for the first, second and last wing, where an offset of the attached flow ratio is visible. This shift corresponds to one or two tell-tales that are always detected as detached, while the other methods detect them as attached. Upon closer observation of the videos, it is clear that some tell-tales were mounted in such a way that they are partially lifted form the wing surface, even when attached. The corresponding image looks therefore much alike when a tell-tale lifts from the wing because it is detached, which explains that the classifier mislabels these tell-tales. This discrepancy is inherent to the fact that the definition of attached/detached tell-tales is a arbitrary and does not represent the continuum of states that the tell-tales adopt. This highlights the fact that the manual labelling of the tell-tales influences the training of the classifier. However, regardless of this shift, the trends and timing are well preserved, showing the robustness of the method.

One limitation of the tell-tale detection for aerodynamic studies is highlighted in Fig. 8, where, when reducing angle of attack of the wings, the Attached flow ratio does not vary (as expected), even though the force on the wing does. Such tell-tale analysis can only give information near stall.

Figure 9 shows the comparison of the three methods on the Zigzag maneuver. The top four panels show the Attached flow ratio computed with the three methods for the wings from aft to bow in this order, whereas the lower panel shows the variation of true heading with respect to the value at the start of the zigzag (orange line with triangle markers) and corresponding apparent wind angle (AWA, in blue with star markers). There is no clear difference of the performance of the three methods, which clearly capture even relatively small variations of the flow state.

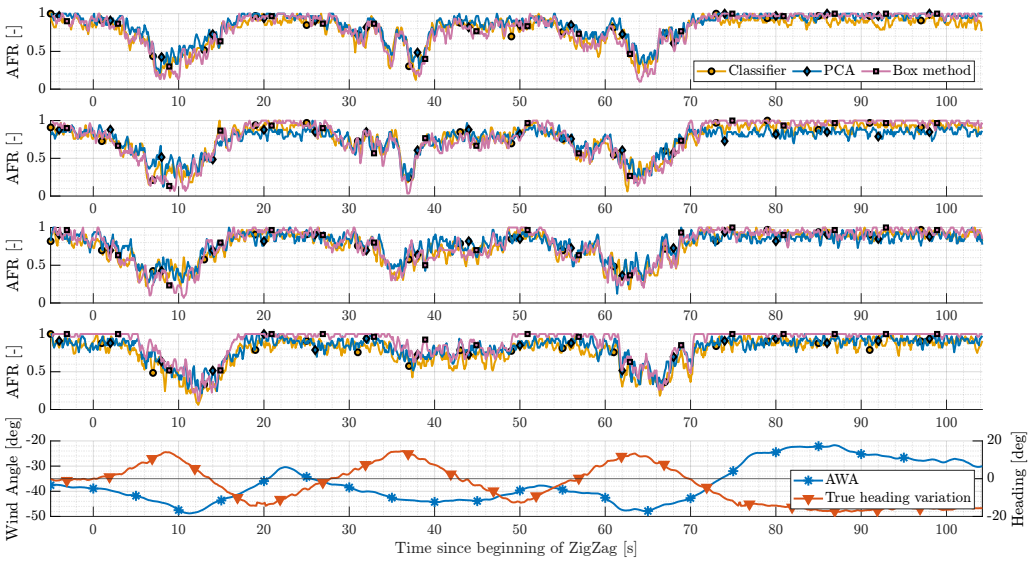


Figure 9. Attached flow ratio for the four wings during a zigzag maneuver. The first four top panels are for the wings from aft to bow, top to bottom with the same colour coding and line style as in Fig. 8. The lowest panel shows the variation of true heading (red line with triangle markers) and apparent wind angle (blue line with star markers).

2.7 Spreading

The model, training dataset, and training notebooks are published to facilitate adoption. A practitioner can fine-tune the published checkpoint on a small set of images from their own setup, optionally fused with the published dataset to avoid catastrophic forgetting. As a recommendation, tell-tales should occupy at least approximately 10 pixels on their thin dimension at inference resolution. In such an end-to-end setup, the pipeline runs live on embedded hardware with no human in the loop after the tell-tale layout is done.

3 3D sail reconstruction

3.1 Introduction

The Evolution of Sail Measurement Sail shape is the primary determinant of aerodynamic performance, yet measuring it in dynamic operating conditions remains a significant challenge. Early vision-based systems, such as SailSpy (Olsson et al., 1992), relied on physical markers and masthead cameras; deck-mounted stripe-based systems like V-SPARS (Le Pelley and Modral, 2008) similarly required instrumented sails (striping) to recover sail and rig geometry. More recently, Maciel et al. (2021) demonstrated monocular 3D reconstruction using passive markers and bundle adjustment (minimizing reprojection error across views). While effective, these methods require intrusive physical modifications to the sails. Conversely, while LiDAR has been used in the America's Cup, its high cost makes it inaccessible for smaller budgets.

The Shift to Learned Dense Matching Recent advances in "foundation models" for 3D vision have changed the game. Models like MASt3R (Leroy et al., 2024)—an evolution of the DUSt3R transformer—don't just look for corners or edges. Instead, they treat the image as a grid of patches and use attention mechanisms to understand the global context of the scene.

By predicting 3D pointmaps and dense matching features simultaneously, MASt3R can establish correspondences even on textureless surfaces where classical algorithms fail. While other models like VGGT (Wang et al., 2025) focus on predicting 3D directly from large image collections, the pipeline leverages MASt3R's ability to find precise pixel-to-pixel matches across a calibrated stereo pair.

A Streamlined, Universal Pipeline The system exploits this AI capability within a fixed, calibrated stereo setup. Unlike standard AI pipelines that must "guess" camera positions and geometry at the same time, the fixed-camera configuration allows for a one-time, reliable calibration. This reduces the problem to dense matching followed by triangulation.

The result is a universal system with the following advantages:

- Non-intrusive: No need for applied textures, markers, or sensors on the rigging.
- Hardware Agnostic: Works with standard consumer-grade cameras.
- Metric Accuracy: Produces a real-world 3D point cloud of the surface.

The pipeline is designed for live operation and can run on embedded hardware; in an end-to-end automated deployment, no human is in the loop once calibration is complete. Figures 10–13 demonstrate the pipeline in action: reconstructing a mainsail during trimming (Figs 10–11) and a jib during a tacking maneuver (Figs 12–13).



Figure 10. Mainsail — stereo view

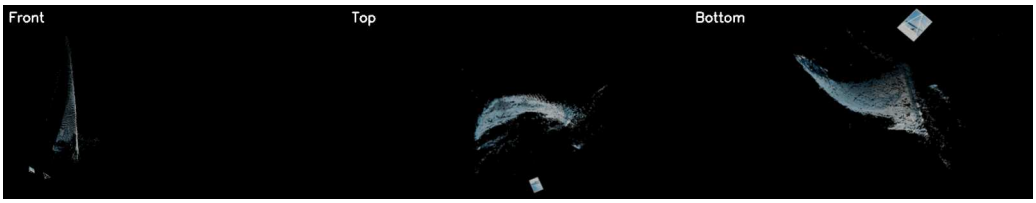


Figure 11. Mainsail — front, top, and bottom views



Figure 12. Jib — stereo view

3.2 Calibration

3.2.1 Problem statement

Stereo calibration estimates the geometric relationship between two cameras. Each camera is described by its **intrinsic** parameters — the camera's internal geometry: focal lengths (f_x, f_y), principal

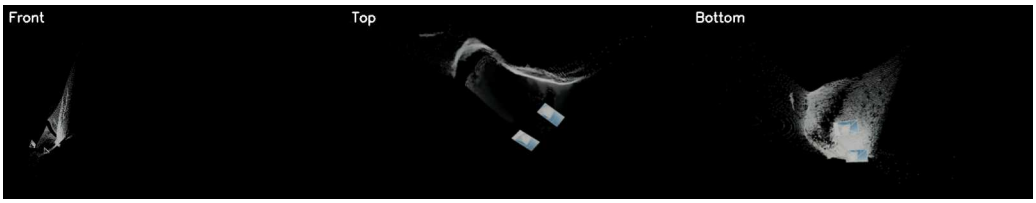


Figure 13. Jib — front, top, and bottom views

point (c_x, c_y), and lens distortion coefficients — and the system is described by **extrinsic** parameters: the relative pose between the two cameras, i.e. a rotation R and translation T such that

$$X_{\text{cam}_2} = R \cdot X_{\text{cam}_1} + T$$

With intrinsics known (from a prior monocular calibration), the stereo calibration problem has **6 degrees of freedom**: 3 for rotation and 3 for translation. These are estimated from matched 2D–3D point correspondences observed in both views simultaneously.

3.2.2 Intrinsics (per camera)

Each camera is calibrated separately using a **chessboard** pattern. A video (or image sequence) of the board is recorded at varying poses and distances. Frames are sampled at a fixed time step (e.g. 1 s) to obtain many views while limiting redundancy. The inner corner positions of the checkerboard are detected in each frame; the 3D positions of those corners are known from the board geometry (square size). OpenCV's `calibrateCamera` minimizes the reprojection error between the observed 2D corners and the projections of the 3D board points, yielding the camera matrix K (focal lengths and principal point) and distortion coefficients. Figure 14 shows an example of chessboard corner detection used for this step.



Figure 14. Chessboard corner detection for monocular intrinsics

3.2.3 Extrinsics (stereo)

For **stereo** calibration a **ChArUco** board is used, which combines the robustness of ArUco marker detection with the precision of chessboard corner localization. Corner detection is performed at multiple resolutions (1080p and full 4K) to maximize the number of reliable correspondences. To populate all degrees of freedom and avoid degeneracy, corner observations from multiple image pairs taken at varying distances and viewing angles are stacked. The calibration is computed using OpenCV's `stereoCalibrate` with fixed intrinsics. Figure 15 shows a synced pair with the ChArUco board detected in both views. The calibration flow from intrinsics to scaled extrinsics is summarized in Fig. 16.

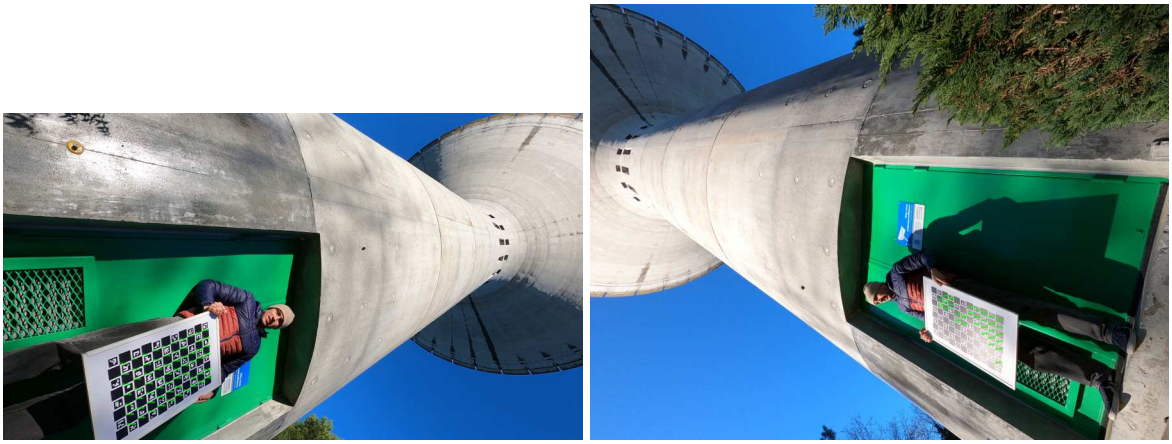


Figure 15. ChArUco stereo pair — Camera 1 (left) and Camera 2 (right) detection

3.2.4 Calibration results

The calibration is validated by comparing the computed baseline (the norm of the translation vector $\|T\|$) against a physical measurement. The scene identifiers in Tab. 2 refer to: **3** — mainsail steady; **7** — mainsail trimming; **8** — jib tacking; **10–11** — Naujac water castle (far, near); **12–13** — Pouch water castle (far, near); **14–15** — St-Médard water castle (far, near). The baseline errors show sub 1% relative error consistently achieved in high resolution calibration, as visible in Tab. 2.

3.3 Dense correspondence

The reconstruction process from image pair to point cloud is summarized in Fig. 17.

MASt3R (Leroy et al., 2024), a pair-based dense stereo model is used. It infers at a fixed resolution of **512 pixels** on the longest image dimension; input images are resized and center-cropped to align with a patch size of 16 pixels. Dense correspondences are extracted using fast reciprocal nearest-neighbor matching on the model's feature maps.

MASt3R does not require explicit texture or markers on the surface, but it does rely on visual granularity — microscopic surface features that serve as implicit anchors for matching. On a perfectly uniform surface, matching fails. This constraint sets a **critical distance**: beyond a certain range, the surface

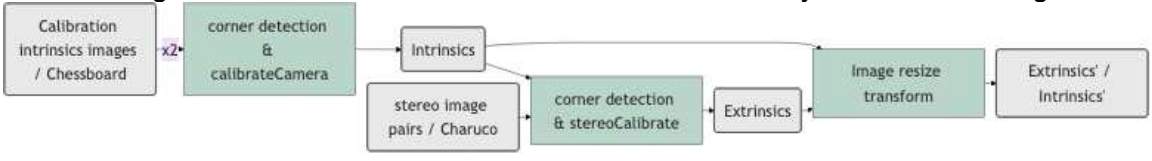


Figure 16. Calibration computational flow

Scene	Computed baseline	Measured baseline	Error	Reprojection (px)	Point pairs	Resolution	Pattern
3	1.657 m	1.640 m	1.0%	0.41	60	1080p	April tag
7	1.498 m	1.470 m	1.9%	0.68	52	1080p	April tag
8	0.806 m	0.785 m	2.6%	0.97	100	1080p	April tag
10	2.779 m	2.800 m	-0.8%	3.45	304	4K	ChArUco
11	2.048 m	2.058 m	-0.5%	4.89	993	4K	ChArUco
12	2.757 m	2.759 m	-0.1%	2.80	190	4K	ChArUco
13	2.092 m	2.110 m	-0.8%	4.96	1441	4K	ChArUco
14	3.395 m	3.408 m	-0.4%	2.11	125	4K	ChArUco
15	1.962 m	1.948 m	0.7%	7.92	1881	4K	ChArUco

Table 2. Results of calibrations for all scenes.

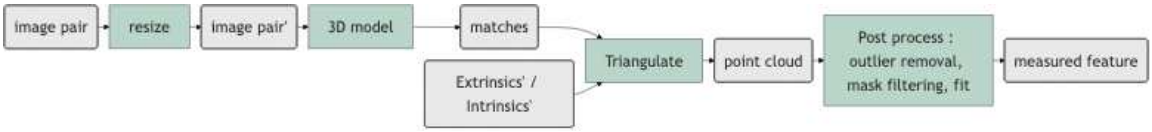


Figure 17. Process flow

granularity at 512px inference resolution is insufficient for reliable matching. When finer resolution is needed, a coarse-to-fine strategy — processing overlapping 512px windows with 50% overlap — allows the model to "see nearer" the surface.

The subsampling rate of the correspondence map is configurable, trading density against computation time.

3.4 Triangulation

Given dense 2D correspondences from MASt3R and the calibrated camera parameters, *triangulation* recovers 3D point positions: for each pair of matching 2D points (one in each image), it finds the 3D point at the intersection of the two viewing rays. The correspondences are first undistorted using each camera's intrinsic parameters. Projection matrices are then constructed:

$$P_1 = K_1[I \mid 0], \quad P_2 = K_2[R \mid t]$$

where K_1, K_2 are the intrinsic matrices and $[R \mid t]$ encodes the extrinsic pose of the second camera relative to the first. OpenCV's `triangulatePoints` produces homogeneous 4D coordinates, converted to Euclidean 3D. The calibration parameters are scaled from the original image resolution to MASt3R's 512px working resolution to ensure geometric consistency. The result is a metric 3D point cloud in the reference frame of camera 1.

3.5 Validation: water castle

To validate the reconstruction pipeline before deployment on sails, water castles were used — cylindrical concrete structures with no specific pre-applied texture, at a working scale comparable to sails: 10 to 20 m height, few meter radius. The cylinder geometry provides a reliable reference shape for quantitative evaluation of curvature radius, though it does not replicate the precision of a laboratory artifact.

This validation constitutes the **pre-analysis required before taking the system to sea**; robustness

in real sailing conditions (spray on optics, mast vibration, sudden changes in lighting and reflections) is left for **future work**.

A stereo camera setup is placed 2 to 3 meters from the cylinder surface, closely simulating an onboard configuration where deck-mounted cameras observe the sail.

3.5.1 Post-processing

The 3D point cloud is post-processed using a cylinder-fitting pipeline:

1. Statistical outlier removal (3D k-nearest-neighbor filtering).
2. PCA to identify the principal axis of the cylinder.
3. Radial filtering to remove points far from the fitted axis.
4. Slicing along the axis.
5. For each slice: projection to the orthogonal plane, RANSAC circle fitting.

This yields radius estimates as a function of height, with uncertainty bounds ($\pm 2\sigma$) — here σ is the fit uncertainty in radius space (**sigma-fit**).

3.5.2 Coarse-to-fine inference

Standard MASt3R inference at 512px can lose surface granularity at far distances. A coarse-to-fine strategy — processing overlapping 512px windows at an effective resolution of 720px — allows the model to resolve finer surface detail. On the **Naujac** far capture, this reduces the radius error from 18.3% to **1.7%** (fitted 4.06 $\pm$0.35 m vs measured 4.13 m), demonstrating that the remaining error in standard inference is primarily a resolution limit, not a fundamental matching failure.

3.5.3 Cylinder fitting process (illustrated on Pouch far)

Figure 18 shows the full analysis pipeline on the best-performing point cloud (Pouch, far, measured radius 3.35 m): outlier removal, PCA axis extraction, slicing, and circle fitting per slice. Figure 19 shows the detail projection of the water castle circle at different heights.

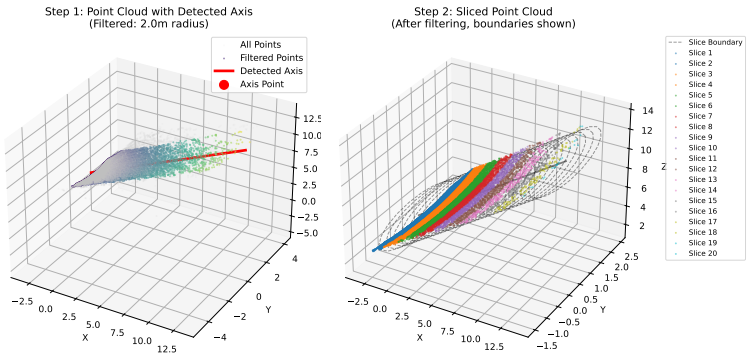


Figure 18. Pouch far — cylinder fitting process

Typical error in mm. The slice-wise projections and three related quantities are used (Fig. 20). The **green circle** is a fixed-radius circle with radius equal to the physically measured radius; it serves

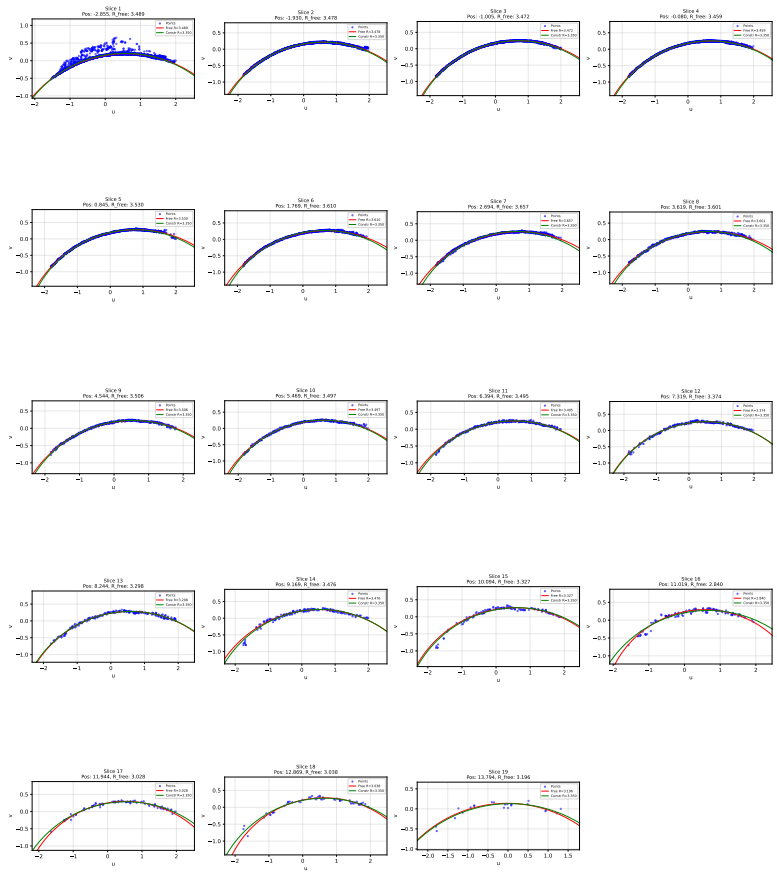


Figure 19. Pouch far — circle fits per slice (2D projections orthogonal to PCA axis)

as a **ground-truth proxy**. The free circle fit is not used as reference because it is sensitive to the arc fit itself (partial arc, noise, limited observability of curvature). **(1) Sigma-fit** is the radius-space standard deviation of the fit (reported in the results table as the $\pm$ term in fitted radius). **(2) Sigma-dist** is the standard deviation of the radial distance from each point to the green circle, in metric space — the **green curve** in the plot; it reflects the spread of the point cloud around the reference circle (sensor and matching noise; at sufficient length scale such noise has zero mean and averages out in the fit). **(3)** The **red curve** is the standard deviation of the distance between the free-fit (red) and fixed (green) circles, evaluated only along the arc inside the data bounding box — it represents **systematic/structural error**, i.e. the **precision of the pipeline after noise that averages out over the fit is removed** (the main precision demonstration of this study). In typical conditions (e.g. Pouch far), both curves give an order of magnitude for error in the range of a few mm to a few tens of mm; the red curve indicates what accuracy can be expected from the circle fit itself in that regime.

3.5.4 Overall results

The pipeline is evaluated on water castles at three sites (Pouch, Naujac, St-Médard), each captured at far and near cam-to-surface distances (scenes 12–13, 10–11, and 14–15 respectively). Far distance corresponds approximately to 2–3 m cam-to-surface gap while near corresponds to approximately 1 m. Due to the concave nature of the cylinder, in the near-distance setup the common area of the surface seen by both cameras was too thin to generate exploitable results on cylinder curvature. The following therefore focuses on the far-distance setup.

Table 3 shows the overall radius fitting results of the different water castles. It reports **sigma-fit** in the fitted-radius column and **sigma-dist** (point-to–ground-truth circle, metric) in the Mean 2σ column. Far-distance captures yield usable fits (1–18% error depending on arc coverage and surface conditions).

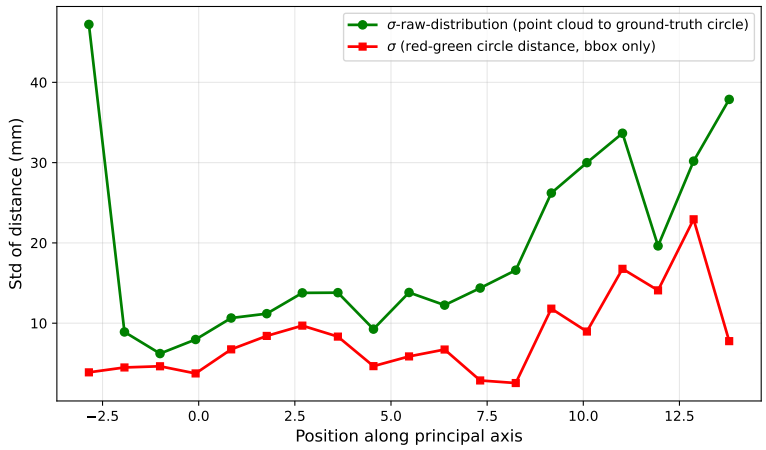


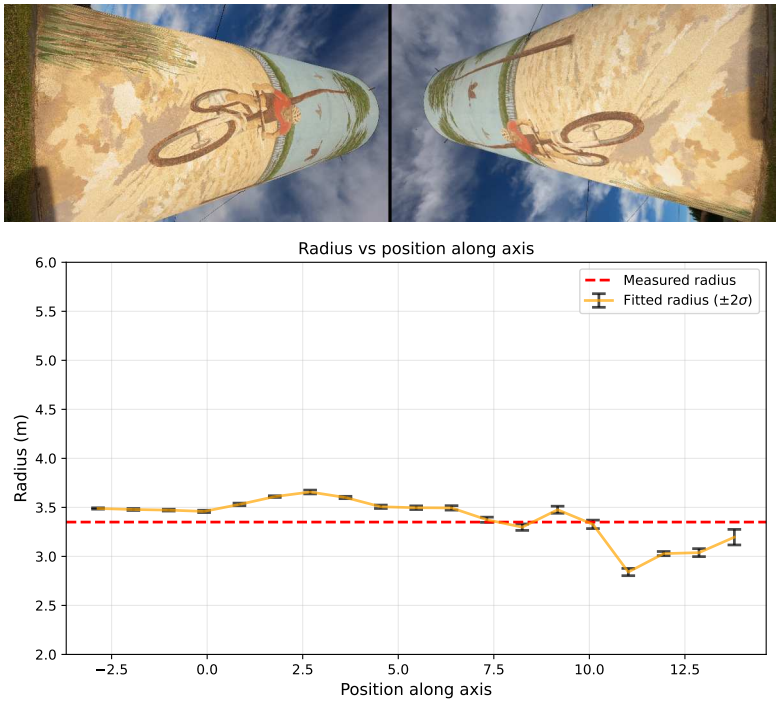
Figure 20. Pouch far — per-slice sigma: sigma-dist (green, point-to-ground-truth) and systematic error (red, circle-to-circle in bbox)

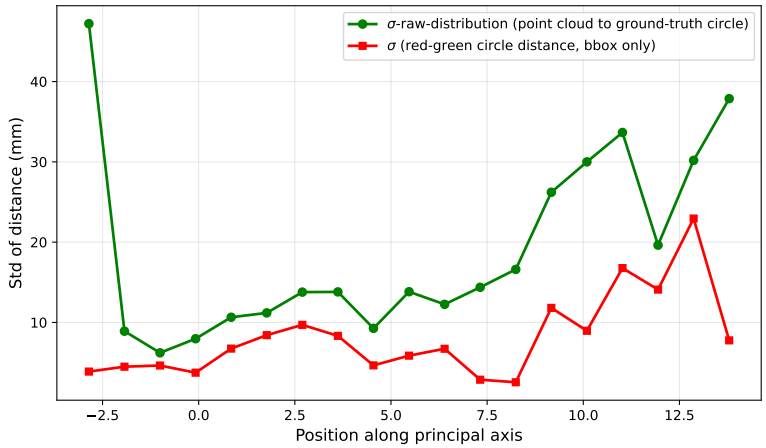
Table 3. Results of radius measurements of water castles.

Site	Distance	Measured Radius	Fitted radius (mean ± sigma-fit)	Error	Mean 2σ (raw data-to-fixed radius circle fit)
Pouch	far	3.35 m	3.39 ± 0.21 m	1.2%	0.025 m
St-Médard	far	3.07 m	3.30 ± 0.26 m	7.5%	0.033 m
Naujac	far	4.13 m	4.89 ± 2.69 m	18.3%	0.037 m
Naujac	far c2f 720	4.13 m	4.06 ± 0.35 m	1.7%	0.038 m

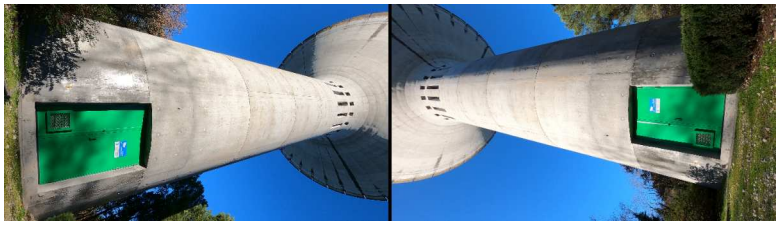
3.5.5 Per-site results

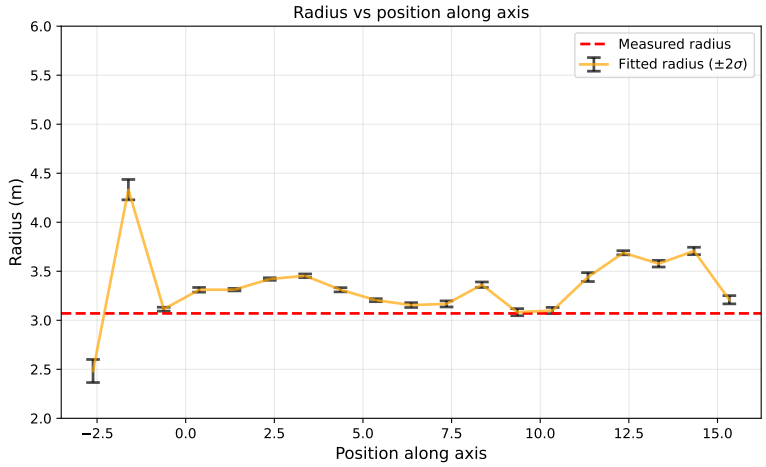
Pouch — far Measured 3.35 m, error 1.2%.

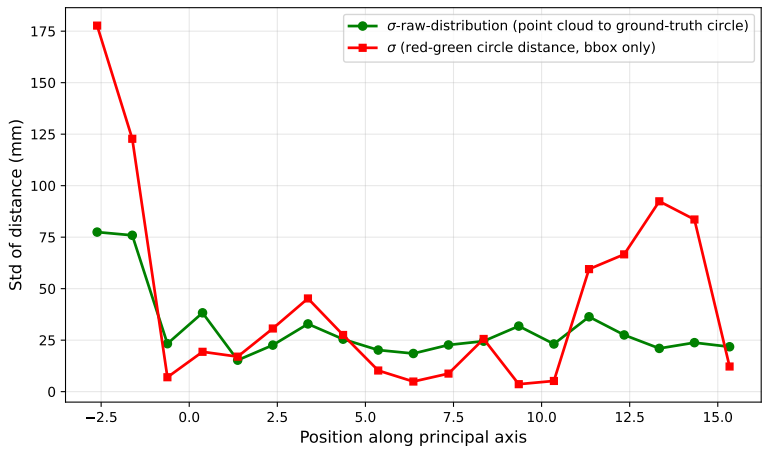




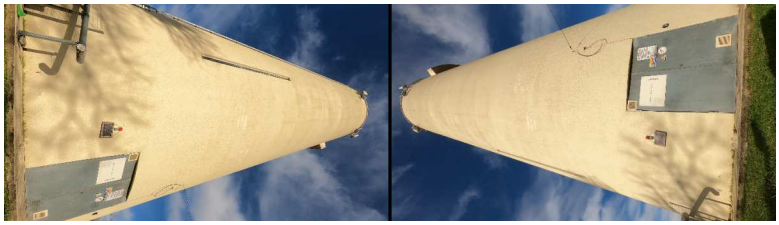
St-Médard — far Measured 3.07 m, error 7.5%.

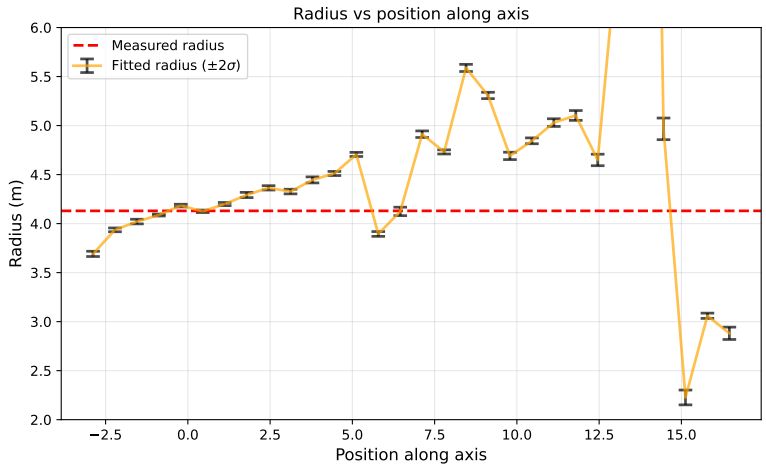


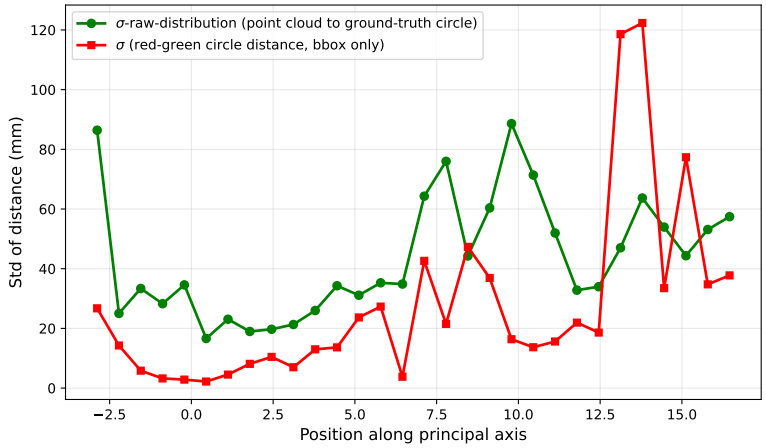




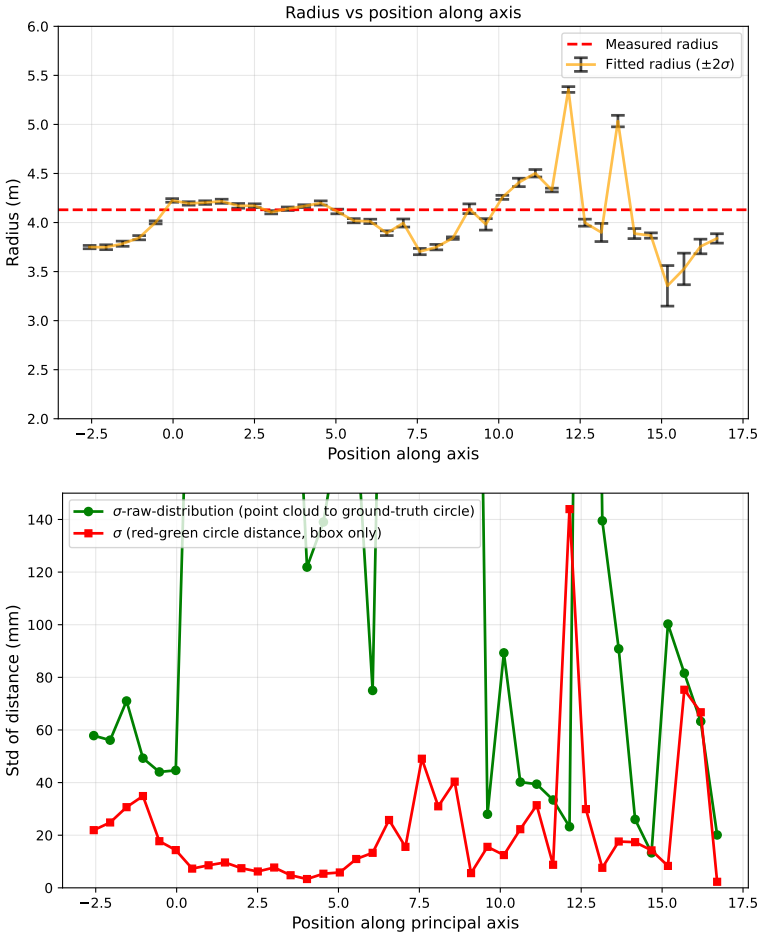
Naujac — far Measured 4.13 m, error 18.3%.







Naujac — far, coarse-to-fine at 720px Measured 4.13 m, error 1.7%. Same stereo pair as above (Naujac far); coarse-to-fine inference only.



3.5.6 Observations

Concavity effect. The concavity of the surface is a limit for a stereo reconstruction setup where cameras are too close to the surface. Moreover on concave surfaces (e.g., the inside of the cylinder), two cameras observe the same physical point at markedly different surface angles compared to a convex surface. In particular cases it can violate the Brightness Constancy Constraint — the assumption that a surface point appears similarly in both views (Lambertian reflectance). Matching performance degrades accordingly. The method is therefore **better suited for intrados (leeward) measurement** than extrados (windward).

Reflections. Specular reflections on the measured surface can catastrophically degrade matching, as reflected features create false correspondences. Matte or diffuse surfaces are strongly preferred.

Conclusions from water castle analysis. Pouch far yields the best fit: drawings on the surface provide anchor points that make pair matching easier for the model. The coarse-to-fine improvement on Naujac far shows that patch-based inference allows the model to see nearer the surface and refine precision on poorly textured or distant surfaces. Taken together, the Pouch and coarse-to-fine Naujac results are those where the method performs well and demonstrate that the pipeline is effective when the combination of surface texture and resolution has sufficient signal for the model to resolve.

3.5.7 Conclusion on 3d reconstruction pipeline

The pipeline produces metric 3D point clouds of sail surfaces from a calibrated stereo pair and learned dense matching (MASt3R) plus triangulation, without markers or special texture. Water-castle validation (cylinder fits, sigma-dist and systematic error) and sail reconstructions (mainsail and jib, Figs. 10–13) show that it is effective when surface texture and resolution are sufficient; coarse-to-fine inference extends the usable range where standard resolution is limiting. The method is best suited to intrados (leeward) measurement and matte or diffuse surfaces; robustness in real sailing conditions remains future work. The pipeline is designed for live execution on embedded hardware, with no human in the loop once calibration is done.

3.6 Cross-module link

At low update rates, the 3D reconstruction module can provide geometric grounding for the tell-tale layout tracker (Module 1): the recovered sail surface geometry compensates for sail deformation that would otherwise drift the assumed layout positions. Together, the two modules and the published datasets and models aim to provide an open, reusable base for sail instrumentation without markers or rig modification. During deployment tests, an inference rate of 1 fps was achieved for direct inference on the Jetson Nano.

4 Conclusion

This paper introduced **Sail-CV**, an embedded, non-intrusive computer vision framework that translates qualitative visual sailing cues into quantitative aerodynamic data. By combining a decoupled tell-tale tracking pipeline with markerless 3D sail reconstruction powered by foundation models, the system extracts continuous aerodynamic states and metric surface geometries. Regarding model training and MLOps, while the currently achieved performance is already promising and sufficient for many practical applications, the primary contribution of this study is presenting this framework as a cohesive, unified toolchain capable of running on shared edge hardware with minimal energy consumption (such as the Jetson Nano).

However, this project is only at its beginning. While the framework has been validated on reference structures and preliminary sail captures, deploying optical systems at sea introduces complex challenges that must be addressed to ensure rugged reliability. Furthermore, leveraging the low-frequency 3D reconstruction to dynamically update the geometric grounding of the high-frequency tell-tale tracker remains an exciting, early-stage frontier for fully automated analysis.

By releasing the code, datasets, and pre-trained models as an open-source suite, the authors establish Sail-CV not as a finalized product, but as an experimental baseline. It is the authors' hope that this foundational work lowers the barrier to entry for intelligent sail instrumentation. The authors invite sailors, engineers, and the computer vision community to adopt, stress-test, and contribute to the project, collaboratively evolving it into a robust, shared tool for marine aerodynamics.

4.1 Acknowledgements

The authors would like to express their sincere gratitude to Eric Foucher for his high-quality contribution to the experimental work. Sincere thanks are due to colleagues from the WeDeployScience team for their support during the early stages of this project. Special thanks are also due and to Simon Jégou for his essential support with GPU hardware.

REFERENCES

Dhomé, U., El Khoury, C., and Kuttenkeuler, J. (2023). Evaluation of the flow state over a rigid wingsail through tell-tale detection using computer vision. *International Conference on Innovation in High Performance Sailing Yachts (Innovsail)*.

Dhomé, U., Hillenbrand, A., Kuttenkeuler, J., and Rolleberg, N. (2025). Unsteady pressure measurements at sea on the rigid wings of a model wind propelled ship. Part A: Measurement system development. *Ocean Engineering* 331, p. 121229. DOI: https://doi.org/10.1016/j.oceaneng.2025.121229.

Le Pelley, D. J. and Modral, O. (Dec. 2008). V-SPARS: A combined sail and rig shape recognition system using imaging techniques. *Proceedings of the 3rd High Performance Yacht Design Conference*. Auckland, 2–4 December 2008. Auckland, New Zealand.

Lee, J., Olsman, W. F. J., and Triebel, R. (2023). Learning Fluid Flow Visualizations from In-flight Images with Tufts. *IEEE Robotics and Automation Letters*. DOI: 10.1109/LRA.2023.3270746. URL: https://doi.org/10.1109/LRA.2023.3270746.

Leroy, V., Cabon, Y., and Revaud, J. (2024). *Grounding Image Matching in 3D with MASt3R*. arXiv: 2406.09756. URL: https://arxiv.org/abs/2406.09756.

Maciel, L., Marroquim, R., Vieira, M., Ribeiro, K., and Alho, A. (2021). Monocular 3D reconstruction of sail flying shape using passive markers. *Machine Vision and Applications* 32.26. DOI: 10.1007/s00138-020-01149-3. URL: https://doi.org/10.1007/s00138-020-01149-3.

Olsson, O., Power, W., Bowman, C., Palmer, T., and Clist, R. (1992). SailSpy: a vision system for yacht sail shape measurement. *SPIE Vol. 1823*. Vol. 1823. SPIE, pp. 256–265.

Wang, J., Chen, M., Karaev, N., Vedaldi, A., Rupprecht, C., and Novotny, D. (2025). *VGGT: Visual Geometry Grounded Transformer*. arXiv: 2503.11651. URL: https://arxiv.org/abs/2503.11651.