Journal of Science and Technology – Data Science and Artificial Intelligence
Volume 36, Issue 1, October 2026, 029-039

A Neural Processing Unit-Accelerated Monocular Perception Pipeline for Remotely Operated Underwater Vehicle Station Keeping

Gyan Louisin[1,2], Vinh La The[1], Trung Ngo Lam[1], Nga Nguyen Thi Thanh[1], Dung Nguyen Anh[1], Duc Duong Minh[1], Hiep Hoang Van[1,*]

[1]*Hanoi University of Science and Technology, Ha Noi, Vietnam*
[2]*Grenoble Institute of Technology, Grenoble, France*
[*]*Corresponding author email: hiephv@soict.hust.edu.vn*

Abstract

Holding a stable viewpoint is a prerequisite for observation tasks performed by remotely operated underwater vehicles, yet it is difficult to achieve when the only sensor is a monocular camera and all computation must run on board. This work reports the deployment and open-loop characterisation of the perception and range-estimation chain of a visual stationkeeping system on a Qualcomm QCS8550 platform, in which two networks-an INT8 object detector and a dense monocular depth model—share the platform's Hexagon HTP/NPU accelerator. Stage-level profiling shows that accelerating depth inference yields a 27-fold speedup, increasing throughput from 0.46 to 8.47 FPS, although depth inference still consumes two-thirds of the frame budget and accelerator occupancy remains at least 69%. We document two deployment failure modes that are silent rather than loud: an output polarity that must be established empirically, and a normalisation choice that destroys the range signal. The two available range cues are combined by an analytic fusion rule-the square root of the tracked bounding box area and the depth output both vary inversely with distance, so the two are directly combinable once anchored on a reference captured at target selection, without training data or knowledge of the target dimensions. On the rigid targets evaluated here the zero-cost geometric term carries more of the distance signal than the depth network that consumes two thirds of the frame budget, and the two cues degrade unequally on immersion, which makes their relative weighting an operating parameter rather than a constant. Closed-loop stationkeeping is not demonstrated: validation was prevented by a degenerate thruster configuration, and the results characterise the measurement chain rather than the controlled vehicle.

Keywords: Computer vision, embedded AI, NPU, ROV.

1. Introduction

Remotely operated vehicles (ROVs) are widely used for observation and inspection tasks, where image quality depends directly on the ability of the vehicle to hold a stable position in front of the target. An underwater platform cannot rely on satellite positioning, and acoustic localisation is costly, bulky, and often insufficiently accurate at the short ranges typical of close-up inspection. Station keeping must therefore be derived from onboard sensing, and the camera already carried for the observation task is an attractive source of feedback.

A monocular image does not determine the position of an object in three dimensions, because absolute scale is unobservable. For station keeping, however, absolute range is not required: what is needed is a signal varying monotonically with the displacement from a reference viewpoint captured at the start of the task. Prior work exploits this by tracking the apparent area of a detected bounding box, which grows as the vehicle approaches the target [1]. The cue is computationally negligible, but it assumes the target keeps a constant apparent size at constant range—violated whenever the target rotates, deforms, or becomes partially occluded. Monocular depth estimation offers a complementary cue, estimating relative range from image structure rather than from the extent of a single box, and is in principle insensitive to target orientation—a property we motivate but do not test experimentally, since our sequences involve rigid targets undergoing pure approach-and-retreat motion (Section 7.4).

Combining the two raises two questions. First, can the signals be combined in a principled way rather than heuristically? Second, can the pipeline run on board in real time, and at what cost? Adding a dense depth network to a system already running detection and tracking increases the per-frame inference cost by more than an order of magnitude, which a general-purpose processor cannot absorb without dropping the loop rate below usefulness. The method thus relies on hardware acceleration as a core contribution rather than an implementation detail.

We address the second question by deploying both networks—the detector quantised to 8-bit integers, the depth model in floating point—on the Hexagon neural processing unit of an embedded system-on-chip through

p-ISSN 3093-3331
e-ISSN 3093-3323
https://doi.org/10.51316/jst.194.dsai.2026.36.1.4
Received: Aug 20, 2026; Revised: Sep 28, 2026;
Accepted: Oct 2, 2026; Online: Oct 15, 2026

a vendor inference delegate, which exhibited behaviour documented by neither the model documentation nor the runtime documentation. We address the first by expressing both cues in a common space: the square root of the bounding box area and the output of the depth network both vary inversely with distance, so a single weight interpolates between them.

The contributions are threefold:

- Real-time deployment of two perception networks on a shared embedded accelerator, with stage-wise latency analysis, counter-free utilisation estimation, and identification of two silent depth-model failure modes.
- Contrary to the pipeline's design intuition, the low-cost geometric cue provides a stronger and more stable distance signal (though not demonstrably higher accuracy) than the depth network, which consumes two thirds of the frame budget. Under immersion, detector degradation also appears to contribute substantially to poorer estimation, beyond the limitations of the depth backbones.
- A depth-scale fusion estimator anchored on a run-time reference, requiring neither training data nor knowledge of the target dimensions, generalising the bounding-box-only formulation of prior work as one limiting case of its weight. We characterise that weight, on the air sequence, as a predictable interpolation between two regimes whose sensitivity to immersion differs, and do not claim an interior value outperforms both components.

Section 2 reviews related work; Section 3 describes the system; Section 4 derives the fusion method; Section 5 details the NPU deployment; Section 6 presents the evaluation; Section 7 discusses limitations.

2. Related Work

2.1. Vision-Based Underwater Station Keeping

Station keeping from vision alone predates deep learning by two decades. Lots et al. regulate a ROV's position by tracking sea-bed feature points and inferring depth from a planar homography [2], establishing the principle later work has refined: hold an image-plane error near zero rather than reconstruct a metric pose. Closer to the present work, Safa et al. benchmark online trackers for underwater position locking and adopt bounding-box growth as the sole range cue driving the depth axis of the controller [1], which sidesteps underwater depth estimation entirely at the cost of assuming the target's size and orientation remain constant relative to the camera. We adopt the same cue as one input to our estimator, and treat their formulation as the special case our fusion reduces to at one extreme of its weight.

2.2. Monocular Depth Estimation and Scale Ambiguity

A single camera cannot recover absolute scale from a static scene, a consequence of the pinhole model [3]. Modern monocular depth networks predict depth or disparity only up to an unknown affine transform, trained through relative rather than metric supervision across mixed datasets [4]; Depth Anything V2 [5] and V3 [6] extend this line with improved sharpness and cross-view consistency. All three share the property that motivates our fusion: their output is a relative disparity, directly interpretable as an inverse-distance signal within a frame, provided the polarity of the specific export is verified rather than assumed (Section 5). The affine ambiguity is per-image, and its consequences for a cross-frame measurement are discussed in Section 4.1. None of these models were evaluated by their authors on underwater imagery, where turbidity and attenuation reduce the contrast and texture such networks rely on [7].

Underwater monocular depth estimation has also emerged as a domain-specific problem. Recent methods incorporate underwater image-formation priors, including wavelength-dependent attenuation and scattering, into the network architecture or training objective to address the domain gap between terrestrial and underwater imagery [8], [9]. UDepth [8] targets computationally efficient depth prediction for visually guided underwater robots by exploiting underwater light-attenuation priors, while UMono [9] explicitly incorporates characteristics of the underwater image-formation model into its depth-estimation architecture. These works further motivate treating the application of terrestrial monocular-depth models to underwater imagery as a domain-transfer problem rather than a direct deployment setting.

2.3. Embedded Deployment of Deep Perception Models

The computational cost of dense monocular depth estimation has motivated a parallel line of work on embedded deployment. FastDepth demonstrated that architecture and decoder optimisation can enable real-time monocular depth inference on embedded hardware, while later lightweight models such as Lite-Mono explicitly target the accuracy–complexity trade-off required by edge devices [10], [11]. These studies primarily optimise the depth model itself. Our focus is complementary: we examine the deployment of a dense depth model alongside detection and tracking within a complete on-board perception pipeline, with both networks sharing the same accelerator.

The standard route to acceleration is post-training integer quantisation, which reduces both memory footprint and latency by replacing floating-point arithmetic with 8-bit integer operations, and which has been shown effective on mobile accelerators including the Hexagon DSP family used here [12]. Vendor inference delegates expose this through a standard

framework API; we use the Qualcomm QNN delegate for TensorFlow Lite. We follow the routine practice of quantising the detector, but deploy the depth network in floating point and find that the accelerator brings it within the real-time budget regardless—which suggests that on this class of device quantisation is a further optimisation rather than a precondition for dense prediction models. Dense depth estimation on mobile accelerators has been studied as a throughput benchmark in its own right; we are not aware of reports of such a model running inside an on-board perception loop feeding a station keeping controller on an underwater vehicle, together with the deployment issues that setting exposes.

3. System Overview

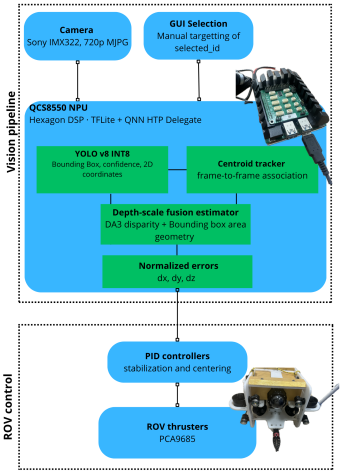


Fig. 1. Overview of the visual stationkeeping pipeline. The perception modules run on the Qualcomm QCS8550 platform, while the PID controller and thruster blocks represent the intended closed-loop system architecture

The system is a single feed-forward pipeline (Fig. 1): a monocular stream is processed by a detection and tracking front-end, the selected target is converted into a scalar range estimate by the depth-scale fusion module of Section 4, and a PID controller translates the resulting error signals into thruster commands.

3.1. Hardware Platform

The vehicle is a ROV configured for observation. Vision is provided by a single forward-looking camera based on a Sony Exmor IMX322 sensor with a native resolution of 1920×1080. The pipeline operates on the MJPG stream at 1280×720, natively supported by the sensor, which avoids the latency of on-board H.264 encoding and decoding.

All inference runs on-board on a Qualcomm QCS8550 platform, whose Hexagon DSP exposes a neural processing unit accessible through the TFLite runtime with the QNN HTP delegate, provided by the QAIRT toolchain version 2.48.0. The two networks differ by more than an order of magnitude in size—3.4 MB for the quantised detector against 143.7 MB for the floating-point depth model—and this asymmetry is reflected in the latency breakdown of Section 6.1.

3.2. Perception Front-End

Target detection uses a YOLOv8n network [13], from the YOLO family of real-time detectors [14], quantised to INT8. The lightweight variant is selected because detection is only one of two networks sharing the NPU, and because the task requires localising a single salient target rather than dense multi-class recognition. The confidence threshold is higher than the customary low default threshold, since in the scenes considered here low thresholds admit many spurious detections that destabilise the association step. It is 0.25 in the operational configuration of Section 6.1, whereas Section 6.2 treats it as a study variable at 0.05 and 0.15; the two studies are therefore not run at a common threshold, and we state the value with each result.

Detections are associated across frames by a lightweight centroid tracker assigning a persistent identifier by greedy nearest-neighbour matching of box centroids between successive frames, subject to a gating radius; a track receiving no detection for a fixed interval is discarded. The expiry is defined in seconds rather than frames, so tracker behaviour does not change with the loop rate and remains comparable across the configurations of Section 6.1. This deliberately simplifies the standard formulation, which augments association with a Kalman filter and a globally optimal assignment [15]: with a single target designated for the whole sequence and no requirement to maintain identity through occlusion, the additional state estimation was not required. In the operational pipeline the target is designated by an operator through the graphical interface; the tracker then outputs, for that identifier, the bounding box and its centroid. For the experiments of Section 6, where no operator is in the loop, the target is designated automatically after a fixed warm-up interval, which prevents the tracker from locking onto transient background detections.

3.3. Control Loop

The vision pipeline does not actuate the vehicle directly. Per frame it transmits normalised error signals over UDP—centroid offsets from the image centre, a scale error derived from the fused range estimate, a validity flag and the detection confidence—and the on-board controller closes the loop locally, keeping actuation independent of the latency of the perception process. Three quantities are regulated: the horizontal offset by lateral translation, so the target is recentred without changing heading; the scale error by surge, holding the apparent size of the target and hence the standoff distance at its value at selection time; and

heading separately, with its setpoint tracking the current attitude so yaw only rejects disturbances. Each loop is a discrete PID with dead-band, derivative filtering, anti-windup, rate limiting and a per-axis minimum command magnitude, mapped onto eight thrusters through a fixed mixing matrix. Closed-loop trials were attempted and are reported in Section 7.1; they could not validate the estimator, for reasons lying in the actuation hardware rather than the perception chain. The evaluation of Section 6 is therefore open-loop, with the pipeline exercised on the vehicle in real time and its output logged, but the resulting commands not applied.

4. Depth-Scale Fusion Estimator

4.1. Problem Formulation

Neither prior knowledge of the target geometry nor an external range reference is available: the target is designated at run time and its dimensions are unknown. The task is stationkeeping rather than absolute ranging, so what the controller requires is a signal monotonic in distance and stable enough to be regulated around the value it took at designation.

Two signals are available. The first is the depth network output, read at a single pixel at the target centroid, whose coordinates are exponentially smoothed before sampling so that detector jitter does not translate into sampling a different part of the scene between frames; we denote it z_{model}. The second is the area A of the bounding box, also exponentially smoothed, since the raw box fluctuates and the square root taken below would propagate that fluctuation. Neither is reliable alone: the network output exploits scene context but is defined only up to an unknown scale and offset, while the bounding box is a direct geometric measurement that degrades whenever the detector is imprecise or the target partially occluded.

Two assumptions underlie the use of z_{model} as a cross-frame measurement, and we state them because neither is guaranteed by the models we deploy. First, relative depth networks predict disparity up to a per-image affine transform, so the raw output at frame t has the form $a_t/d + b_t$ with a_t and b_t unobservable. Treating z_{model} as comparable between frames presumes these drift slowly relative to the distance signal—plausible in stationkeeping, where consecutive frames differ little, but not enforced; Section 5.3 shows what happens when a processing step makes that variation large, and Section 7.2 returns to the residual. Second, sampling the disparity map at a single pixel is the most noise-sensitive estimator available and is chosen for its negligible cost; a robust statistic over the box interior expected to be more robust at comparable cost, and is a refinement we did not evaluate.

Combining geometric and learned cues for per-object distance is usually realised by a trained regressor; the formulation below is instead analytic and anchored on a run-time reference, requiring no training data, no knowledge of the target size, and a negligible number of operations per frame.

4.2. Geometric Scale Term

Under a pinhole projection, an object of fixed physical extent at distance d projects onto an image extent inversely proportional to d, so

$$A \propto \frac{1}{d^2} \tag{1}$$

and $\sqrt{A} \propto 1/d$: the square root of the area is expressed in the same space as the raw network output. Let $(z_\star, A_\star)$ be the reference pair captured at designation. For the current frame, with smoothed area A,

$$z_{\text{bb}} = z_\star \sqrt{\frac{A}{A_\star}} \tag{2}$$

where the ratio expresses by what factor the inverse distance has changed relative to the reference. Neither the physical size of the target nor the focal length is required, both being absorbed into $(z_\star, A_\star)$; this distinguishes the formulation from classical size-based ranging, which assumes object dimensions known or constant within a class. The reference pair is taken as the median of several samples collected after a settling interval rather than read from one frame, since a single unlucky sample offsets the reference permanently and the controller never converges even against a stationary target.

4.3. Fusion Rule

Both estimates now live in the same space:

$$z_{\text{fused}} = \alpha\, z_{\text{model}} + (1-\alpha)\, z_{\text{bb}} \qquad \alpha \in [0,1] \tag{3}$$

The convex form is not arbitrary, though the argument for it is weaker than independence would allow. Because z_{bb} is anchored on $z_\star$, itself sampled from the depth network at designation, the two terms share a common scale by construction but are not independent: an error in the reference propagates multiplicatively into z_{bb} and hence into any combination. What $w_1 + w_2 = 1$ guarantees for a general linear combination $w_1 z_{\text{model}} + w_2 z_{\text{bb}}$ is that the fused value stays on that common scale, so $z_\star$ remains the setpoint of the loop for every α. Restricting α to $[0,1]$ further keeps the fused value between the two estimates rather than extrapolating beyond them, bounding the effect of a transient failure of either signal.

One condition invalidates the geometric term outright. When the box is truncated by an image border its visible area no longer reflects the apparent size of the target and (1) no longer holds; the estimator detects this from the box coordinates and falls back to $\alpha = 1$ for that frame. The proportion of frames triggering the fallback is reported for the underwater sequence in Section 6.3, since it bounds how much of that sequence the fusion can act on.

Under independent errors the variance-minimising weight would favour whichever signal is less noisy. We do not compute it: the two terms are not independent, as noted, and underwater turbidity and illumination degrade the depth network and the detector simultaneously, so the error terms are not stationary either. The weight is instead selected empirically over a discrete set of values, where $\alpha = 1$ and $\alpha = 0$ recover the depth network alone and the geometric term alone and serve as the two baselines. An online alternative weights each cue by the inverse of its recent increment variance. Replayed offline over a 15-frame causal window, this rule does not improve stability over the best fixed weight, but it recovers the environment dependence without being told: its mean weight is 0.06 in air, favouring the geometric term, and rises to 0.45 under water, favouring the depth term.

The fused value is finally mapped back to a distance-like signal and smoothed. What is transmitted is not the deviation from a single setpoint but the excursion outside a tolerance band centred on it, normalised to a dimensionless interval, the band half-width being derived from the dispersion of the samples collected during reference capture rather than fixed as a tuning constant. Since the evaluation is open-loop, this stage is not exercised in the results.

5. Deploying a Monocular Depth Network on the Neural Processing Unit

Running a dense depth network on an embedded accelerator inside a control loop raises problems absent when the same model is evaluated offline. This section documents the ones we encountered, because they are properties of the deployment rather than of the model, and are therefore unlikely to be reported by the authors of the model itself.

5.1. Toolchain and Operator Coverage

The depth network runs through the TFLite runtime [16] [17] with the QNN HTP delegate. The delegate version is not a free choice: the operator set used by the network is not covered by earlier toolchain releases, and an insufficient release does not report an explicit error and instead silently falls back to CPU execution for the unsupported subgraphs, removing most of the benefit of the accelerator while leaving the pipeline apparently functional. A second constraint is ordering rather than versioning: the search path locating the DSP libraries must be set before the runtime is imported, not merely before the interpreter is constructed. Setting it later leaves the delegate unable to initialise the accelerator, again without an explicit error.

5.2. Output Convention

Monocular depth networks do not agree on output polarity: some predict depth directly, a larger value denoting a more distant point, others an inverse-depth or disparity-like quantity, where a larger value denotes a nearer point. The distinction is invisible in a colour-mapped visualisation but inverts the sign of every downstream error signal.

The polarity of the deployed export was therefore established empirically, by two independent checks. The first compares the value sampled at the target against the extrema of the same frame: since the target is the nearest object in view, the convention follows from which extremum it approaches. The second exploits the geometric term, whose polarity is fixed by construction, and checks whether the network output varies with it or against it. Both agree: the export behaves as a disparity. The pipeline treats the raw output as such throughout and inverts it only once, after fusion. This matters beyond a sign convention, because the fusion presupposes both terms monotonic in the same direction; had they been opposed, the combination would have subtracted rather than reinforced, and the ablation of α would have measured a cancellation rather than a trade-off. The check is inexpensive and is a precondition of any such fusion, not a troubleshooting step.

5.3. Output Scaling

The most damaging issue we encountered was not a performance problem but a silent corruption of the signal. A natural way to bring the raw output into a fixed numerical range is to normalise each frame by its own extrema. This is standard practice for display, and exactly wrong when a single pixel of the output is used as a measurement.

The extrema are set by whichever scene elements happen to be nearest and furthest in that frame. In a stationkeeping task the foreground is largely static: the target is held near a fixed standoff and the structure of the vehicle occupies the same part of the image throughout. The extrema are therefore determined by nearly the same elements in every frame, so the normalised value at any point is pinned to a nearly constant position within the range irrespective of how far the target actually is. In the terms of Section 4.1, the procedure does not remove the per-image affine ambiguity but amplifies it, by making the effective a_t and b_t depend on the frame's own extrema. The signal reaching the controller carried little of the distance variation it was supposed to convey, while remaining plausible enough in a colour-mapped visualisation to go unnoticed.

The fix is a fixed scale factor, so the mapping from raw output to working range is identical across frames and relative variation is preserved. This removes the ambiguity the processing introduces, not the one intrinsic to the model (Section 7.2). The upper bound of the working range must then be set from observation rather than assumption: our initial bound was exceeded by the raw output during operation, clipping the signal precisely for the nearest targets, and had to be raised.

5.4. Measuring Utilisation Without a Hardware Counter

The platform exposes no counter reporting accelerator occupancy. We derive utilisation as the ratio between the summed inference time of the two networks and wall-clock time,

$$U = \frac{t_{\text{det}} + t_{\text{depth}}}{t_{\text{frame}}} \tag{4}$$

a lower bound, since it attributes to the accelerator only the intervals during which an inference call is outstanding and ignores dispatch overhead. Thermal state is likewise not exposed under the names one would expect: the accelerator zones are grouped under a neural-signal-processor prefix rather than under any name containing the vendor's designation for the accelerator itself, and several exposed zones return constant invalid readings that must be discarded. Both utilisation and thermal behaviour are reported in Section 6.1. All latency measurements in this paper come from instrumentation embedded in the pipeline rather than an external profiler, so timings correspond to the code path actually executed in operation; the cost of that instrumentation is quantified below and not subtracted from any figure.

6. Experimental Evaluation

Two studies are reported. The first characterises the pipeline as a real-time system: its throughput, where that throughput comes from, and how it changes when the depth network runs on the CPU. The second concerns the estimator: a comparison of three depth backbones against ground truth, then an ablation of the fusion weight α. The two use different detection thresholds and their configurations are not directly comparable. Ablation runs are performed offline on recorded sequences, which makes them exactly repeatable. Both studies are open-loop, for the reason given in Section 7.1.

6.1. Real-Time Performance

Table 1 reports per-stage median latency over the same recorded sequence in two configurations. Video encoding is disabled in both, so the figures describe the perception and estimation path only. The detection threshold is 0.25 throughout this section.

Table 1. Median per-stage latency (ms) over 1345 frames

Stage	NPU	CPU	Ratio
Frame acquisition	13.14	8.52	0.65
Detector pre-process	5.81	4.12	0.71
Depth pre-process	7.38	5.80	0.79
Detector inference	2.22	2.48	1.12
Depth inference	79.65	2150.17	27.00
Detector post-process	5.32	1.26	0.24
Remaining stages	1.27	0.36	0.28
Total accounted	114.79	2172.71	18.93
Frame time	118.1	2173.4	18.40
Throughput (frame/s)	8.47	0.46	—
Utilisation U (%)	69.3	0.11*	—

*In the CPU configuration only the detector runs on the accelerator, so U counts detector time alone.

Three observations follow. First, the configurations differ in one *intended* factor: the depth network runs on the accelerator in the first and on the CPU in the second, the detector on the accelerator throughout. Depth inference is twenty-seven times faster on the accelerator, and this single stage accounts for essentially the whole difference in throughput: at 0.46 frames per second, the perception rate is too low for the intended stationkeeping application, whereas at 8.47 frames per second the target is tracked continuously; whether this rate suffices for closed-loop control is discussed in Section 7.1. The detector, quantised to INT8, is essentially unaffected.

Second, offloading is not free. The non-depth stages cost 35.1 ms with the network offloaded against 22.5 ms without, an overhead of 12.6 ms per frame. Part is the transfer of the depth map back from accelerator memory (depth pre-processing 7.38 against 5.80 ms, output retrieval 5.32 against 1.26 ms), but frame acquisition, which touches no depth tensor, also rises from 8.52 to 13.14 ms: it measures the wait for a new frame from the separate capture thread, and a loop cycling eighteen times faster contends with that thread more often. We did not separate this effect from governor behaviour, so the comparison isolates the depth stage to within this residual; the net 18× gain is obtained despite the overhead, not in its absence.

Third, acceleration reduces the bottleneck without removing it. Depth inference still accounts for 67% of the frame budget, so further improvement must come from the depth model rather than the rest of the pipeline, which motivates Section 6.2. The utilisation metric of Section 5.4 evaluates to 69.3% with the network offloaded and 0.11% on the CPU. Both are lower bounds excluding dispatch overhead. The offloaded value is consistent with the 67% attributed to depth inference, which checks the metric rather than adding a result; the contrast between the two figures is the substantive point, since the CPU configuration does not merely run slowly but leaves the accelerator idle for essentially the

whole loop while the application processor saturates. Frame time exceeds the sum of measured stages by 3.3 ms when offloaded, which bounds the unaccounted portion of the loop and confirms the throughput is a sustained rate rather than an artefact of rate limiting. The instrumentation costs 266 µs per frame, or 0.23% of the frame budget.

Sustained operation was characterised over a continuous run of approximately twenty minutes in the accelerated configuration. The accelerator zones rise from about 59 °C to a plateau of 64–68 °C within roughly four minutes (values read from the plotted trace, hence approximate), several degrees below the large application cores at about 70 °C: offloading moves the dominant load onto the coolest-running unit of the system-on-chip, which matters on a sealed vehicle. We log temperatures and not cooling-device state, so we make no claim about thermal limitation; the trace establishes only that the throughput figures above are not obtained during a start-up transient.

6.2. Depth Backbone Comparison

Since depth inference dominates the frame budget even after acceleration, the backbone determines both throughput and estimate quality. Three backbones were compared against measured ground truth at six distances, in air and in water. The deployed backbone was fixed early in development, before this comparison was carried out.

Table 2. Pearson correlation between the estimated and measured distance over six range steps from 10 to 150 cm

	Air		Water
	$c = 0.25$	$c = 0.05$	$c = 0.15$
DA-V2	−0.878	−0.406	−0.685
DA-V3	−0.661	−0.204	−0.566
MiDaS	+0.327	−0.040	−0.115
Retained frames	251	1557	142
Range steps	6	6	6

Coefficients are computed over per-frame samples, but frames within a range step are strongly autocorrelated: the number of independent observations behind each is closer to the six range steps than to the frame count, and the retained-frame row indicates how much data survived the threshold rather than a degrees-of-freedom count. They are accordingly reported without confidence intervals, and we draw only ordinal conclusions.

The ordering of the three models is consistent across all three conditions, and the deployed backbone—DA-V3—is not the strongest: DA-V2 correlates more strongly in every condition tested. Within the two water columns, which share a sequence and differ only in threshold, raising it from 0.05 to 0.15 recovers a substantial part of the correlation, by a factor of 1.7 to 2.9 depending on the model, and removes the non-monotonic orderings of range steps seen at the lower threshold. Since the threshold acts on the detector and not on the depth models, and since the improvement is common to all three, the limiting factor in immersion is detector noise rather than an intrinsic limitation of the depth models—a result holding regardless of backbone, resting on the two water columns alone, and bearing directly on the ablation below. It costs sample count: retained frames fall from 1557 to 142. The positive sign for MiDaS in air is anomalous and reflects a signal largely lost in noise; that model is the weakest in every condition tested.

6.3. Fusion Weight Ablation

6.3.1. Premise check and source comparison

The fusion assumes the two terms estimate the same underlying quantity. On the air sequence, their per-frame correlation is weak ($r = 0.31$) but increases monotonically under smoothing—0.40, 0.50 and 0.63 over windows of 5, 15 and 45 frames—which is consistent with two estimators sharing a common low-frequency signal while carrying largely independent high-frequency noise, the condition under which averaging is worthwhile. Establishing that the shared component is the distance itself, rather than another slow trend common to both, would require ground truth on this sequence. The target box area varies by a factor of about four over the sequence, so the distance does genuinely vary.

Comparing the two terms against each other depends on neither the weight nor the border fallback. A direct comparison would be unfair as implemented, since the geometric term derives from a low-pass filtered box area whereas the network output is unfiltered; we therefore apply the same filter to the network output before comparing, and report alongside the jitter both the dispersion of each term and their ratio, a non-standard signal-to-noise proxy used here in the absence of ground truth. Frame-to-frame increments are taken only between consecutive frames: under water, valid frames are interleaved with detection dropouts, and differencing across those gaps would measure accumulated motion rather than jitter.

Jitter is defined as the standard deviation of frame-to-frame increments computed only between consecutive frames, while dispersion is the standard deviation over the sequence. Their ratio provides a proxy for signal variation relative to noise. For the underwater sequence, only fusion-active frames are considered (57 frames and 21 consecutive increments), excluding border-truncated detections.

Table 3. Jitter, dispersion, and dispersion-to-jitter ratio of the equally smoothed depth and geometric cues in air and underwater

			Air			Water
	Jit.	**Disp.**	**Rat.**	**Jit.**	**Disp.**	**Rat.**
Depth term	16.4	62.3	3.8	15.2	57.6	3.8
Geometric term	6.1	200.4	32.6	21.1	159.2	7.6

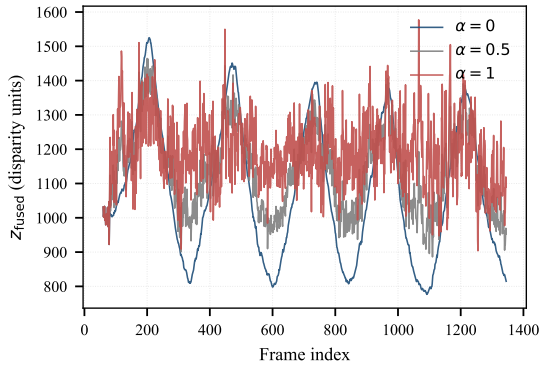


Fig. 2. Fused disparity estimates over four approach-and-retreat cycles for three fusion weights ($\alpha = 0$, 0.5, and 1)

In air the geometric term dominates on both counts: it jitters less by a factor of 2.7 and varies more over the sequence by a factor of 3.2, giving a ratio about eight times higher than the depth term. Figure 2 shows why: over four approach-and-retreat cycles the geometric term traces the motion clearly, whereas the depth term follows with markedly reduced amplitude and substantial frame-to-frame noise. The low dispersion of the depth term is therefore not stability but under-responsiveness—on this sequence it carries less of the distance signal than the box area does—which is notable for a pipeline in which depth inference consumes two thirds of the frame budget while the geometric term costs nothing. Two mechanisms plausibly contribute and our data do not separate them: the per-image affine ambiguity of Section 4.1, which injects a frame-varying offset unrelated to distance, and the single-pixel sampling of the disparity map.

Under water the picture changes. On jitter the ranking reverses, the geometric term becoming the less stable, and on the ratio the ranking is preserved but the margin shrinks from about eight to about two. Immersion penalises the geometric term far more than the depth term: its jitter rises by a factor of about 3.4, while that of the depth term is essentially unchanged. The same asymmetry appears, with factors of 4.2 and 1.4, when border-truncated frames are included, so it does not depend on the frame subset; its magnitude, however, rests on few consecutive increments and is indicative only. The asymmetry has a clear mechanical explanation: the geometric term depends entirely on the detector, whose boxes degrade in turbid, low-contrast water, while the depth network degrades less. This is corroborated independently by the threshold sensitivity of Section 6.2, which would not follow if the depth networks rather than the detector were the limiting factor.

6.3.2. Weight ablation

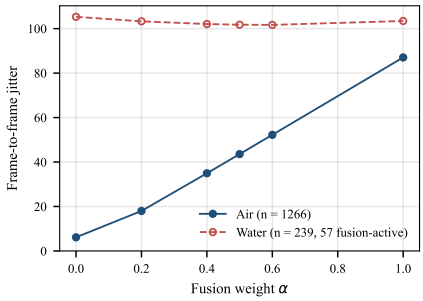


Fig. 3. Frame-to-frame jitter of the fused estimate versus fusion weight α in air and underwater

Six values of α were evaluated, each repeated twice, on recorded sequences replayed through the unmodified pipeline. Repetitions produced identical statistics to three decimal places, establishing that inference on the accelerator is deterministic and that any difference between configurations is attributable to α alone; it also means they give no estimate of variability across runs, the effective replication being one sequence per environment.

In air the jitter rises monotonically from 6.1 to 87.1, a factor of about fourteen, while dispersion falls monotonically from 200.4 to 96.1 (Fig. 3). The endpoints coincide with the source terms taken alone: at $\alpha = 0$ with the geometric term of Table 3, and at $\alpha = 1$ with the unsmoothed depth output, confirming that the ablation recovers both baselines exactly at its extremes and interpolates smoothly between them. This is the property the formulation predicts: α moves the estimate predictably between two noise regimes, with no discontinuity or cancellation, which is the practical precondition for treating the weight as an operating parameter. It does not establish an interior optimum, and none should be expected from these metrics: every curve is monotonic, so the best setting lies at a boundary on each metric taken separately, and the two disagree about which. Lower dispersion is not in itself preferable, since a signal that varies less may be compressing the true variation rather than tracking it faithfully; choosing between the boundaries is an accuracy question requiring ground truth on the ablation sequences.

The underwater ablation is reported for completeness but is not evidential. Of 239 frames only 57 have the fusion active, and they form 36 disjoint segments, the longest spanning four frames; elsewhere the box is border-truncated, forcing $\alpha = 1$ irrespective of the configured value. Accordingly the jitter is essentially independent of the weight, between 101.7 and 105.3 over the whole range, while dispersion falls from 235.7 to 155.0. This is what a predominantly-fallback run would produce whether or not the underlying trend matched the air result. Evaluating the weight under water requires a sequence framed to keep the target clear of the image borders.

7. Limitations

7.1. Actuation

The evaluation is open-loop, and this is not a methodological preference. Closed-loop trials were attempted, and characterisation of the horizontal thruster set carried out afterwards established that the command subspace is degenerate: one unit produces no motion in either direction at the amplitudes tested; two that should counter-rotate turn in the same direction, indicating reversed wiring or propeller; and a fourth is noticeably more powerful at equal command. A subsequent repair left one unit disproportionately sensitive.

The consequence is not merely reduced authority but an uncompensated parasitic torque. In those trials the horizontal error converged initially and then drifted monotonically, the expected signature of a constant disturbance the heading loop cannot observe: its setpoint tracks the current attitude passively, so a steady bias produces no heading error and therefore no correction. Since the behaviour is explained by the actuation fault, these runs say nothing about the range estimate and we report no closed-loop metrics from them. Validation is deferred until the hardware is restored, and the results here characterise the measurement chain rather than the controlled system.

Whether the measured rate would suffice once actuation is restored can be argued but not settled here. Two properties of the architecture bear on it. First, perception and control are decoupled (Section 3.3): the controller runs locally at its own rate and acts on the latest error between perception updates, so the perception rate bounds the measurement bandwidth rather than the actuation rate. The same structure has closed the loop on a small eight-thruster ROV, a BlueROV2 in its Heavy configuration, with visual inference at 2 Hz and a 10 Hz on-board controller [18], and underwater visual servoing has been demonstrated at camera rates of 5 Hz [19]; the 8.47 Hz measured here lies above both, although neither vehicle shares the dynamics of ours. Second, the usual guideline of four to ten samples per closed-loop rise time [20] admits rise times down to roughly 0.5–1.2 s at this rate.

Processing latency is not the dominant delay in the loop as deployed. Over a separate run of about 3900 frames, the median latency from frame retrieval to transmission of the error signal is about 100 ms, to which the age of the retrieved frame adds at most one camera period, about 33 ms at the nominal 30 Hz capture rate, plus a capture and decoding delay we did not instrument. The error signals, however, are exponentially smoothed before transmission, with equivalent time constants of about 0.7 s on the lateral offset and 1.1 s on the range error at the measured rate; at a crossover of 1 rad/s these filters cost roughly 36° and 48° of phase margin, against about 7° for the measured latency. The smoothing was tuned for signal stability in open loop and is a design parameter rather than a property of the platform, and it is the first element a closed-loop design would have to retune or compensate. These arguments indicate that the perception rate is unlikely to be the factor limiting closed-loop performance; they do not establish that the loop would be stable or well damped, which depends on gains, disturbance spectrum and thruster dynamics, and belongs to the validation deferred above.

7.2. Affine Ambiguity of the Depth Term

Relative depth networks predict disparity up to a per-image affine transform, and the estimator treats z_{model} as comparable across frames, which presumes the transform drifts slowly relative to the distance signal. We do not verify this: doing so requires an in-frame reference of known and constant range, such as a static structural element of the vehicle in view, against which the transform could be re-anchored every frame. The normalisation failure of Section 5.3 is the extreme case of the same phenomenon, introduced by our processing and removed by a fixed scale factor; the residual drift intrinsic to the model remains, and is a candidate explanation for the under-responsiveness measured in Section 6.3. Per-frame re-anchoring is inexpensive and does not change the fusion rule; we regard it as the first improvement to attempt. A partial check is possible: the frame-to-frame increments of the target sample share 22% (water) to 30% (air) of their variance with those of the scene-mean disparity, an upper bound on frame-wide drift since genuine scene motion also contributes, so most of the variation is local to the sampled pixel.

7.3. Model Choice and Domain Gap

The backbone was fixed early, when the priority was to establish that a dense depth network could run on the accelerator at all; the comparison of Section 6.2 indicates that DA-V2 would have been preferable, and re-deploying requires no change to the estimator or controller. We did not measure the latency of the alternative backbones on the accelerator, and since depth inference accounts for two thirds of the frame budget, a joint accuracy–latency comparison is the natural next measurement.

Replacing the backbone would leave the underlying cause untouched. Both networks are pre-trained on terrestrial imagery: the detector on a general-purpose object dataset containing no underwater category [21], so the target is matched to an approximate class at intrinsically low confidence, and the depth networks on scenes whose appearance statistics—attenuation, colour cast, contrast, texture density—differ substantially from turbid water. That all three backbones degrade in water, including the one degrading least, suggests this is a property of the model family rather than of any particular model. Fine-tuning on underwater imagery is the approach most directly addressing this limitation.

7.4. Scope of the Evaluation

Four restrictions bound how far these results generalise, sharing a cause worth stating once: access to the vehicle and to the embedded platform ended before the analysis was complete, so several measurements identified as desirable during writing could not be acquired.

First, a single sequence was recorded per environment and the repetitions are numerically identical, so the ablation is exactly repeatable but not replicated.

Second, the targets in both sequences are rigid and undergo pure approach-and-retreat motion—the regime in which the constant-apparent-size assumption holds best, and therefore the one least favourable to the depth term, whose motivation is precisely robustness to rotation, deformation and partial occlusion. That the geometric term carries more of the distance signal in air is a valid result on these sequences, not evidence that the depth term is superfluous in general. The only occlusion-like condition exercised is border truncation, and there the estimator does not degrade gracefully but removes the geometric term altogether through the fallback of Section 4.3. Rotation and deformation are harder: they change the apparent area without any change in range, biasing the geometric term in a way the box coordinates cannot reveal. A disagreement between the two cues beyond the dispersion observed at reference capture would flag such cases and could trigger the same fallback; we have not tested this, and the robustness of the depth term to these conditions remains a motivation rather than a result.

Third, the underwater ablation is dominated by border truncation: the 57 fusion-active frames form 36 disjoint segments of at most four frames, so the data yield no usable measurement of the weight under water; relatedly, the air and water columns of Table 2, recorded at different detection thresholds, cannot be compared directly.

Fourth, and most importantly, the ablation measures stability, not accuracy: the sequences carry no distance ground truth, so the metrics describe how much the estimate moves, not how close it is to the true distance, and a signal that is stable and wrong would score well on both. Accuracy is addressed only indirectly, through the backbone comparison, which uses ground truth but evaluates the depth models in isolation rather than the fused estimator.

8. Conclusion

We have described an on-board perception pipeline for underwater stationkeeping in which a detector and a dense depth model share one embedded accelerator and feed an analytic, training-free fusion rule.

Three results follow from deploying the system rather than simulating it. The depth stage alone accounts for the whole difference in throughput between the two configurations tested: offloading it makes it twenty-seven times faster and carries the pipeline from 0.46 to 8.47 frames per second, from a rate at which closing a control loop is not viable to one at which it becomes plausible, a question the open-loop evaluation leaves open, with depth inference still occupying two thirds of the frame budget and the accelerator occupied at least 69% of wall-clock time against 0.11% on the CPU. Deploying a depth network on such a target exposes failure modes that are silent rather than loud: an output polarity that must be established empirically, and a normalisation choice that is standard for display and destroys the signal when a single sample is used as a measurement. And on the rigid targets evaluated here the term derived from the bounding box, which costs nothing, carries more of the distance signal than the depth network consuming two thirds of the frame budget, while the two cues degrade very unequally on immersion—the detector-derived term becoming three to four times less stable while the depth-derived term is essentially unaffected—which is what makes their relative weighting an operating parameter rather than a constant.

The evaluation is open-loop, and the honest summary is that this work characterises a measurement chain rather than a controlled system. Beyond restoring the actuation hardware, the improvements identified in Section 7 need no new research; the experiment that would most change the assessment of the estimator, however, is a sequence with synchronised distance ground truth and a target that rotates or is partially occluded, which is the case the fusion was designed for and the only one our data do not cover.

Acknowledgements

This research was funded by the Ministry of Education and Training of Vietnam under project code CT2025.EA.BKA.18.

References

[1] A. Safa, W. Aman, A. Al-Zawqari and S. Al-Kuwari, Benchmarking 0nline 0bject trackers for underwater robot position locking applications, in IEEE Journal of Oceanic Engineering, vol. 50, no. 4, pp. 2770-2781, Oct. 2025. https://doi.org/10.1109/JOE.2025.3590074

[2] J.-F. Lots, D. M. Lane, E. Trucco, and F. Chaumette, A 2-D visual servoing for underwater vehicle station keeping, in Proceedings 2001 IEEE International Conference on Robotics and Automation, Seoul, Korea, May 2001, vol. 3, pp. 2767–2772.

[3] R. Hartley and A. Zisserman, Multiple View Geometry in Computer Vision, 2nd ed. Cambridge, UK: Cambridge University Press, 2003.

[4] R. Ranftl, K. Lasinger, D. Hafner, K. Schindler, and V. Koltun, Towards robust monocular depth estimation: mixing datasets for zero-shot cross-dataset transfer, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 44, iss. 3, pp. 1623–1637, Mar. 2022. https://doi.org/10.1109/TPAMI.2020.3019967

[5] L. Yang, B. Kang, Z. Huang, Z. Zhao, X. Xu, J. Feng, and H. Zhao, Depth anything V2, Advances in Neural Information Processing Systems, vol. 37, pp. 21875–21911, Dec. 2024. https://doi.org/10.52202/079017-0688

[6] H. Lin, S. Chen, J. Liew, D. Y. Chen, Z. Li, Y. Zhao, S. Peng, H. Guo, X. Zhou, G. Shi, J. Feng, and B. Kang, Depth Anything 3: Recovering the Visual Space from Any Views, in Processing International Conference on Learning Representations (ICLR), Jan. 2026.

[7] R. Schettini and S. Corchs, Underwater image processing: state of the art of restoration and image enhancement methods, EURASIP J. Adv. Signal Process., vol. 2010, art. 746052, 2010.

[8] B. Yu, J. Wu, and M. J. Islam, "UDepth: Fast monocular depth estimation for visually-guided underwater robots," in Proc. IEEE Int. Conf. Robotics and Automation (ICRA), 2023.

[9] X. Wu, J. Wang, J. Wang, S. Rong, and B. He, "UMono: Physical-model-informed hybrid CNN–Transformer framework for underwater monocular depth estimation," IEEE Journal of Oceanic Engineering, vol. 51, no. 1, pp. 780–793, 2026, doi: 10.1109/JOE.2025.3606045.

[10] D. Wofk, F. Ma, T.-J. Yang, S. Karaman, and V. Sze, "FastDepth: Fast monocular depth estimation on embedded systems," in Proc. IEEE International Conference on Robotics and Automation (ICRA), 2019.

[11] N. Zhang, F. Nex, G. Vosselman, and N. Kerle, "Lite-Mono: A lightweight CNN and transformer architecture for self-supervised monocular depth estimation," in Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2023, pp. 18537–18546, doi: 10.1109/CVPR52729.2023.01778.

[12] B. Jacob, S. Kligys, B. Chen, M. Zhu, M. Tang, A. Howard, H. Adam, and D. Kalenichenko, Quantization and training of neural networks for efficient integer-arithmetic-only inference, in Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition, 2018, pp. 2704-2713.

[13] G. Jocher, A. Chaurasia, and J. Qiu, YOLO by Ultralytics, Ultralytics, 2023. [Online]. Available: https://github.com/ultralytics/ultralytics

[14] C. -Y. Wang, A. Bochkovskiy and H. -Y. M. Liao, "YOLOv7: Trainable Bag-of-Freebies Sets New State-of-the-Art for Real-Time Object Detectors," 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Vancouver, BC, Canada, 2023, pp. 7464-7475, doi: 10.1109/CVPR52729.2023.00721.

[15] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, Simple online and realtime tracking, in Proc. IEEE Int. Conf. Image Processing (ICIP), 2016, pp. 3464-3468.

[16] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek G. Murray, Benoit Steiner, Paul Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2016. TensorFlow: a system for large-scale machine learning. In Proceedings of the 12th USENIX conference on Operating Systems Design and Implementation (OSDI'16). USENIX Association, USA, 265–283.

[17] TensorFlow Lite, Google. [Online]. Available: https://www.tensorflow.org/lite

[18] J. Yang, M.-Q. Le, M. Gong, and Y. Pu, "DUViN: Diffusion-based underwater visual navigation via knowledge-transferred depth features," arXiv:2509.02983, 2025.

[19] S. Krupiński, G. Allibert, M.-D. Hua, and T. Hamel, "An inertial-aided homography-based visual servo control approach for (almost) fully actuated autonomous underwater vehicles," IEEE Trans. Robotics, vol. 33, no. 5, pp. 1041–1060, 2017.

[20] K. J. Åström and B. Wittenmark, Computer-Controlled Systems: Theory and Design, 3rd ed. Upper Saddle River, NJ, USA: Prentice Hall, 1997.

[21] Lin, TY. et al. (2014). Microsoft COCO: Common Objects in Context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds) Computer Vision – ECCV 2014. ECCV 2014. Lecture Notes in Computer Science, vol 8693. Springer, Cham. https://doi.org/10.1007/978-3-319-10602-1_48.