Efficient Annotation and Modeling for Littoral Marine Radar Object Detection and Classification

JC Vaught
Mechanical Engineering
University of South Carolina
Columbia, USA
jvaught@sc.edu

Douglas Cahl
Mechanical Engineering
University of South Carolina
Columbia, USA
dcahl@sc.edu

Yi Wang
Mechanical Engineering
University of South Carolina
Columbia, USA
yiwang@cec.sc.edu

***Abstract*—Shore-based stationary radars are becoming an affordable option for monitoring harbors, lakes, rivers, and nearshore coastal zones in all weather conditions. Unlike camera-based systems, however, radar returns contain limited visual detail, which makes automatic detection and classification of objects difficult. Progress is further hindered by the scarcity of publicly available training data, since the cost of collecting and labeling radar datasets often leads institutions to keep them private.**

This work presents an open-source, end-to-end system designed to lower the cost of annotating low-cost recreational marine radar data. The system converts raw amplitude-only radar streams into Plan Position Indicator (PPI) plots and enhanced images that highlight object motion, and it performs on-device pre-labeling of radar frames to accelerate human annotation. All functionality is exposed through a user-friendly interface.

To validate the system, we collected radar data from inland lakes, coastal harbors, open ocean, and large rivers, yielding approximately 200,000 scans, of which 1,000 are manually labeled. We train and compare multiple state-of-the-art object detection models on this dataset and report their performance alongside that of the YOLOv12 nano model used in the pre-labeling pipeline.

Models are evaluated using a simple bounding-box-based metric with a binary label space that distinguishes Static and Dynamic targets, such as buoys or pylons versus moving vessels. The annotation tool and pre-annotation pipeline are designed for secure, resource-constrained deployments and achieve a throughput of roughly one second per full radar frame on an Arm-based CPU after an initial training period of about five minutes.

***Index Terms*—Marine radar, object detection, littoral sensing, annotation tools, YOLO.**

I. Introduction

Littoral environments such as harbors, inlet channels, and nearshore waterways are challenging for radar-based perception. Short-range recreational marine radars provide inexpensive, widely available X-band sensing, but their returns in littoral regimes are dominated by land reflections, piers and jetties, static structures, and complex multipath. This degradation is compounded by sparse and low-SNR signatures from small craft, buoys, and kayaks, which are often the very objects of interest for safety and situational awareness. Supervised learning systems for such data require carefully curated labels, yet the annotation of radar images remains difficult and time-consuming, particularly when standard vision tools designed for RGB images are used without adaptation.

This work focuses on fixed-site deployments of a recreational Furuno-class X-band marine radar at several littoral and inland-water locations. The radar is operated primarily at 1–2 NM range, with a maximum configured range of 3 NM. The antenna is installed on piers, towers, and shore structures at heights ranging from approximately 7 ft to 30 ft above the water surface, depending on the local elevation and mounting geometry. Sites include Lake Murray Dam and Lake Greenwood State Park, which provide reservoir environments with controlled shorelines; Lake Monticello, which offers another inland water body; Folly Beach, Grice Marine Lab, and Demetre Park in the Charleston area, which contribute coastal and tidal environments; and Portsmouth City Park in Virginia, which adds a harbor-like setting with man-made structures.

The first goal of the paper is to describe a practical pipeline that converts raw shore-based radar sweeps into history-augmented polar images suitable for both human annotation and training of convolutional object detectors. The second goal is to present an annotation environment tailored to littoral radar imagery, implemented in Rust with a Slint user interface, and tightly coupled to a CPU-based YOLO v12n pre-annotator. The third goal is to provide baseline detection results on the emerging dataset and to study the sensitivity of those results to temporal history length and dataset size.

The contributions of this paper can be stated as follows. A complete radar-to-image converter is described for short-range X-band recreational radar at fixed littoral sites, including polar representation, temporal history encoding over several seconds, and basic masking of static land returns. A Rust/Slint annotation front-end is introduced that runs entirely on MacOS in secure, GPU-less environments, integrates a hyper-specialized CPU-based YOLO v12n pre-annotator, and supports point, bounding-box, and segmentation modes tailored to marine targets. A dataset of approximately thirty hours of radar recordings from seven sites is characterized, together with an emerging labeled subset expected to reach about one thousand frames, using a three-class taxonomy covering small vessels, large vessels, and buoys. Baseline YOLO v12n detection experiments are reported on early labels and on the

TABLE I: Representative marine radar PPI perception literature and label types.

Reference	Radar / setup	Task	Label type
Chen et al. 2022 [1]	Navigation PPI, complex sea clutter	Marine-Faster R-CNN detection	Bounding boxes
Chen et al. 2021 [2]	Scanning marine radar, AIS alignment	Radar-PPINet detection on PPI	Bounding boxes
Zhang et al. 2023 [3]	PPI sea-clutter database	YOLO-SWFormer detection	Bounding boxes
Ma et al. 2024 (MRNet) [4]	Radar3000, congested and inland waters	Blip-level ship identification	Boxes / patches
Kang et al. 2024 (YOSMR) [5]	Nearshore and inland marine radar	Lightweight ship detection	Bounding boxes
Ma et al. 2024 (MrisNet) [6]	Harbor radar scenes	Ship instance segmentation	Pixel masks
He et al. 2025 (DTNet) [7]	Multi-frame PPI, dynamic sea states	Detection and tracking	Boxes and tracks

expanded dataset, including sensitivity to history length and to the number of labeled frames.

The remainder of the paper is organized as follows. Section II reviews related work on marine radar datasets, radar image representations, annotation systems, and model-assisted labeling. Section III details the radar hardware, sites, collection protocol, and radar-to-image generation. Section IV presents the annotation system design, including the Rust/Slint interface and YOLO pre-annotator. Section V describes the experimental methodology for baseline detection models and sensitivity studies. Section VI reports experimental results. Section VII discusses implications, and Section VIII concludes with future work.

II. Related Work

A. Marine Radar Perception and Datasets

Classical work on marine radar processing focused on thresholding, CFAR variants, and hand-crafted tracking in plan position indicator (PPI) images, often assuming relatively uncluttered open-sea scenes and relying on strong motion cues to distinguish ships from sea clutter and static structures. As nearshore deployments, inland waterways, and congested channels became more important, these heuristic pipelines have increasingly been replaced by convolutional and transformer–based detectors that operate directly on PPI frames or on multi-frame radar representations.

Chen et al. proposed Marine-Faster R-CNN for navigation radar PPI images with complex sea clutter, modifying the backbone, anchor sizes, and feature normalization to better handle small, low-contrast ship echoes, and demonstrating substantial improvements in detection accuracy over vanilla Faster R-CNN on a dedicated navigation-radar dataset [1]. Chen et al. further proposed Radar-PPINet for automatic detection on scanning marine radar by aggregating multi-dimensional features from PPI images, using AIS-aligned data to construct a benchmark dataset and showing that learned detectors can outperform traditional CFAR schemes in diverse sea states [2]. Zhang et al. combined YOLO with a Swin Transformer backbone (YOLO-SWFormer) and reported improved robustness to sea clutter on a PPI sea-clutter database, illustrating the effectiveness of transformer-based backbones for low-resolution radar imagery [3].

Recent work has emphasized both nearshore complexity and computational constraints. Ma et al. introduced MRNet and the Radar3000 dataset, targeting ship identification from marine radar blips in crowded and inland waters; they highlight the severe interference from shorelines, reefs, and waves, and show that a customized CNN can substantially outperform classical ARPA-style processing [4]. Kang et al. proposed the lightweight YOSMR detector for marine radar images, specifically addressing nearshore and inland scenarios where ship spots are easily confused with coastlines and reefs; their design balances accuracy and FLOP count and targets embedded radar hardware [5]. Beyond detection, Ma et al. developed MrisNet for ship instance segmentation in challenging marine radar scenes, using pixel-level masks to learn fine-grained shapes and trails in clutter [6]. He et al. presented DTNet, which exploits multi-frame PPI images for joint detection and tracking, leveraging temporal coherence and feature differences between moving targets and clutter to improve robustness in dynamic conditions [7].

Several marine datasets and multi-sensor benchmarks have been released. The WHUT-MSFVessel dataset couples PPI, camera, and AIS for surface-vessel detection and tracking in real-world inland waterways, with a focus on multi-source fusion and challenging occlusions [8]. The Autoferry dataset provides a littoral, multi-sensor fusion benchmark (radar, lidar, EO/IR) for multi-target tracking around an autonomous ferry, with detailed navigation and ground-truth metadata [9]. The PoLaRIS dataset offers maritime object detection and tracking in a canal environment with radar, lidar, and cameras aligned across viewpoints, emphasizing dense traffic and strong infrastructure clutter [10]. DLR's real-world marine radar datasets provide extended-target point-cloud measurements and associated PPI images with AIS ground truth for tracking research [11].

Table I summarizes representative marine-radar perception work and highlights how existing systems mostly use vessel or research radars, moving platforms, or high-end shore installations. In contrast, the present paper focuses on fixed littoral sites instrumented with low-cost recreational X-band radars of approximately 3 NM nominal range, including inland lakes and coastal parks, and on the end-to-end process that converts raw sweeps into history-encoded images and labels under tight compute constraints.

B. Radar and Vision Annotation Tools

Most large-scale vision datasets rely on generic image and video annotation tools such as LabelImg, LabelMe, CVAT, Label Studio, and commercial web platforms. These systems provide primitives such as axis-aligned boxes, polygons, and points, project management, and export to formats like COCO or YOLO, but they typically treat annotation as a human-first, GUI-centric task and only recently integrated model-assisted labeling. For example, CVAT now supports automatic annotation via pre-trained models and AI agents, including YOLO-based detection and SAM-based segmentation, but assumes server-side GPU inference or cloud agents rather than on-device CPU execution [16], [18].

The cost of manual annotation has been studied extensively in the computer vision community. Extreme Clicking showed that asking annotators to click four extreme points on an object can reduce bounding-box annotation time to around seven seconds while preserving box quality and detector performance compared to traditional box drawing, which previously required on the order of tens of seconds per instance [12]. Pointly-supervised instance segmentation demonstrated that combining boxes with a small number of labeled points per object can recover most of fully supervised mask performance while using substantially cheaper supervision, and BoxInst showed that high-quality instance masks can be learned directly from box annotations by redesigning the mask loss, further narrowing the gap between weak and full supervision [13], [14]. These results support the idea that point and box annotations are often sufficient for training competitive models, with pixel-wise segmentation reserved for subsets or specific tasks.

Within marine radar, most PPI-based works use relatively simple label types. Marine-Faster R-CNN, Radar-PPINet, MRNet, YOSMR, YOLO-SWFormer and DTNet all rely on bounding boxes or patch-level labels over radar blips or PPI regions, typically defined on stitched, single-frame PPI images, sometimes supported by AIS or trajectory information [1]–[5], [7]. MrisNet is a notable exception that uses instance segmentation masks to capture detailed ship shapes and trails in cluttered marine radar scenes [6]. The WHUT-MSFVessel dataset combines radar, cameras and AIS, and describes a labeling process using synchronized multi-sensor evidence, but still emphasizes bounding-box style supervision rather than richer segmentation or interactive annotation tooling optimized for radar operators [8].

These works show that box-centric labeling on PPI images is standard, but they rarely describe the underlying annotation tools in detail, and they do not target the specific constraints of fixed littoral sites with heavy land clutter. In particular, almost none discuss how their tools support different annotation modes (points, boxes, segmentation) in a single interface, how they handle long sessions at a single fixed radar site, or how they collect timing and quality metadata for annotation-efficiency studies. The system presented in this paper is designed explicitly around polar littoral radar imagery, point/box/mask modes, and site-aware metadata logging.

C. Auto-Labeling and Semi-Automatic Methods

Semi-automatic and model-assisted annotation have become common in object detection and tracking. Ince et al. proposed a semi-automatic bounding-box annotation framework for visual object tracking that couples an off-the-shelf detector with multiple hypothesis tracking and incremental retraining: detections are run on video frames, associated into tracklets, and then presented to human annotators who accept or correct them, expanding the training set iteratively and reportedly reducing manual workload by up to ninety-six percent [15]. Similar principles appear in active-learning approaches to labeling video data, where model uncertainty guides which frames or regions to annotate next.

Detector-assisted tools have also emerged in practice. AutoLabelImg, labelGo-YOLOv5 AutoLabelImg, and related forks extend classic box-drawing GUIs with automatic YOLO-based annotation and simple tracking, using detections as initial labels that human annotators correct [17]. CVAT and similar platforms expose auto-annotation APIs to integrate pre-trained or user-provided models (including YOLO and transformer-based detectors) for bounding-box, polygon, and segmentation pre-labeling, often with cloud-hosted inference [16], [18]. Commercial tools such as Roboflow's Auto Label offer batch auto-labeling using hosted models, again with GPU backends and a web interface. These systems demonstrate a general consensus that running a detector first and editing its outputs can substantially accelerate annotation compared to purely manual labeling.

In marine perception specifically, pre-trained detectors are often used for pseudo-labeling or to generate proposals for more complex models, but the annotation pipeline itself is rarely the focus. Radar-PPINet, MRNet, YOSMR, YOLO-SWFormer and DTNet all train deep detectors on PPI images, sometimes exploiting multi-frame information, but they do not describe interactive labeling GUIs or per-dataset fine-tuning loops designed for annotators working at a single radar site [2]–[5], [7]. MrisNet reports instance-segmentation performance under different clutter conditions but does not detail how masks were produced or whether any semi-automatic tools were used [6].

Center-based detectors and fusion models such as CenterFusion show a complementary design direction: they represent objects as centers on an image-plane heatmap, then fuse radar measurements to refine depth and motion estimates [19]. This center-heatmap formulation is naturally compatible with point annotations and demonstrates that anchor-free, heatmap-based methods can achieve strong detection performance with point-like supervision in radar-augmented systems.

Compared to these works, the contribution of the present system is to make detector-assisted annotation a first-class design goal for littoral marine radar. The Rust/Slint GUI is built around CPU-only YOLO v12n pre-annotation that is fine-tuned in one to five short epochs on the specific radar site, then run over entire sessions at roughly one second per 1735×1735 frame on an M1 Pro CPU. The same interface supports point, box and segmentation modes on history-encoded polar images, and logs detailed timing and agreement metadata. This differs from GPU-centric, web-based auto-labeling frameworks and from prior marine radar perception papers, which treat the labeling pipeline as an offline preparatory step rather than an object of study.

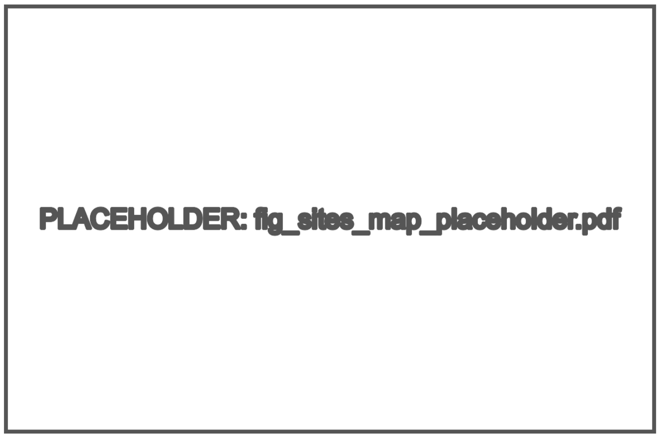


Fig. 1: Deployment locations for the fixed-site littoral radar installations, including inland lakes and coastal or harbor environments. Site photographs illustrate antenna mounting heights and surrounding geometry.

III. Littoral Radar Data and Image Generation

A. Radar System and Site Installation

The sensing hardware is a recreational Furuno-class X-band marine radar system. The radar operates at X-band with a maximum configured range of approximately 3 NM. In the deployments considered here, the range is typically set to 1 or 2 NM, since the areas of interest are harbor approaches, nearshore channels, and lake surfaces relatively close to shore. Range and azimuth resolutions follow manufacturer specifications for this class of radar; beamwidth is characteristic of small recreational antennas, and the antenna rotation rate corresponds to a sweep period on the order of several seconds.

Installations are at fixed shore sites. The antenna is mounted on piers, towers, and park structures such that the physical height above the mounting platform is approximately $5\,\text{ft}$. The resulting antenna height above the water surface ranges from roughly $7\,\text{ft}$ to $30\,\text{ft}$, depending on the local ground elevation at each site. The sites include Lake Murray Dam and Lake Greenwood State Park, which provide reservoir environments with controlled shorelines; Lake Monticello; Folly Beach, Grice Marine Lab, and Demetre Park in the Charleston area; and Portsmouth City Park in Virginia.

Figure 1 illustrates the geographic distribution of these sites and includes representative photographs of several installations. The diversity of locations is intended to cover quiet lakes, moderately busy waterways, and more complex coastal clutter.

B. Littoral Data Collection Protocol

Data collection is conducted by operating the radar continuously over multi-hour sessions at each site. Across all locations, the current dataset comprises approximately thirty hours of recordings and about two hundred thousand individual scans or frames. Sessions are scheduled to capture a range of environmental and traffic conditions, including daytime and nighttime periods, differing wind and wave states, various tide levels at coastal sites, and both low and moderate vessel traffic.

The operating area for each site is defined by the radar range configuration and local geometry. For lake sites, the area of interest is usually the open water in front of dams, boat ramps, or recreational channels. For coastal sites such as Folly Beach and the Charleston locations, the area includes nearshore surf zones, inlets, and navigation channels where small craft, buoys, and shoreline structures coexist. At Portsmouth City Park, the area includes piers, moored vessels, and nearby channels.

Site logs record the start and end times of each collection, radar configuration such as range setting, and notable events such as changes in weather or traffic. These logs are synchronized with the radar data stream using timestamps and are used later to select labeling subsets.

C. Signal Processing and Radar-to-Image Converter

The radar-to-image converter transforms raw radar sweeps into history-augmented polar images suitable for display and machine learning. Each scan is represented in a polar grid indexed by range bin and azimuth bin. The base representation preserves this polar structure without reprojection to Cartesian coordinates, resulting in images of size 1735×1735 pixels for the current configuration.

Temporal history is encoded by aggregating multiple scans over a fixed time window. The default configuration aggregates approximately six seconds of radar history, which corresponds to several antenna rotations depending on the rotation rate. A temporal decay is applied so that more recent scans contribute more strongly to the intensity of each pixel, while older scans are down-weighted. This history encoding emphasizes moving targets and smooths out some of the stochastic variability in individual sweeps.

The number of history frames is treated as a tunable parameter and is later varied in sensitivity experiments. Shorter histories can reduce smearing of fast-moving vessels and maintain sharper responses, while longer histories can help integrate weak returns from small or intermittently visible targets. In all cases, the history-encoded image for each time step can be generated essentially instantaneously once the raw sweeps are available, since the aggregation is a simple accumulation with decay and the implementation is optimized for the fixed polar grid.

Static land and large man-made structures introduce strong, persistent returns that dominate the dynamic range. In the current system, these regions are treated as background and are handled primarily by the training process and the label taxonomy, rather than by aggressive clutter suppression. Static masking of land is applied where reliable shoreline maps are available, but the focus of this paper is on the annotation and modeling pipeline, not on developing advanced clutter-suppression algorithms.

Figure 2 shows example history-encoded polar images from clean lake conditions and from heavily cluttered coastal scenes, illustrating the dynamic range and interference patterns that annotators must interpret.

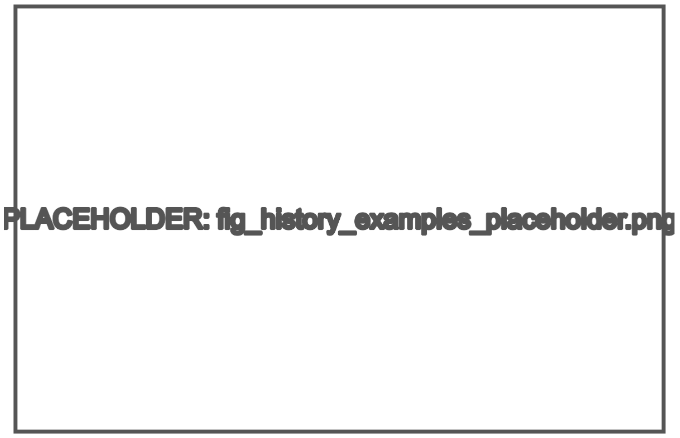


Fig. 2: Example 1735×1735 polar history images. Top: relatively clean lake scene with a small number of vessels. Bottom: cluttered coastal scene with strong land returns, piers, and multiple moving targets.

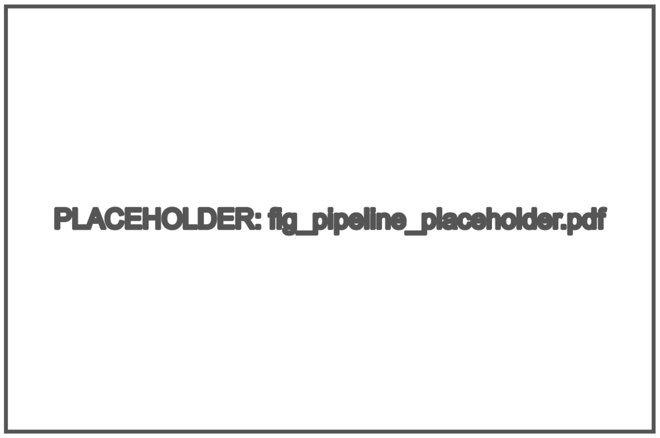


Fig. 3: System pipeline from raw radar sweeps to history-encoded polar images, CPU-based YOLO v12n pre-annotations, Rust/Slint annotation interface, and final labels used for training baseline detection models.

TABLE II: Summary of current littoral radar dataset.

Quantity	Value
Total recording time	$\approx 30\,\text{h}$
Number of scans / frames	$\approx 200{,}000$
Image resolution	1735×1735 polar
Current labeled frames	< 200
Target labeled frames (by presentation)	≈ 1000
Classes	small vessel, large vessel, buoy
Sites	7 (lakes, coastal, harbor)

D. *Dataset Description*

Table II summarizes the current dataset. Approximately thirty hours of recordings yield around 2×10^5 scans, each converted into a history-encoded polar image. Labeled data collection is ongoing; at the time of writing, fewer than two hundred frames have been labeled, with an expected total of about one thousand labeled frames by the time of presentation.

The label taxonomy assigns each target to one of three classes: small vessel, large vessel, and buoy. The intention is that small vessels include small motorboats, personal watercraft, kayaks, and similar craft with relatively small radar cross-sections, while large vessels include larger motor vessels and commercial craft with larger signatures. Buoys include navigation markers and moored buoys that appear as localized, mostly static returns. All other returns, including land, piers, unidentified clutter, and unlabelled targets, are treated as background. No special rules are imposed regarding ghost reflections or multipath; ambiguous or low-confidence returns that do not clearly fall into the three classes are left unlabeled.

IV. Annotation System Design for Littoral Radar

A. *System Overview and Data Flow*

The annotation system is designed as an end-to-end pipeline that starts from raw radar sweeps and ends with label files suitable for training and evaluating detection models. Raw polar scans are first converted into history-encoded 1735×1735 images as described in Section III. These images are stored in a format that preserves timestamps and is easily accessible from the Rust/Slint front-end, such as a directory of image files accompanied by metadata in a lightweight database or JSON-based index.

For each image, a CPU-based YOLO v12n pre-annotator can be invoked to produce initial bounding boxes and class predictions. Pre-annotations for all images in a labeling batch are generated in the background on the same MacOS machine that runs the front-end, at approximately one second per full frame. The pre-annotations are stored alongside the images in the same directory or database structure.

The Rust/Slint annotation interface then loads the images and their corresponding pre-annotations. Annotators review and edit these suggestions, add new annotations where needed, and decide whether to use point, bounding box, or segmentation modes. Once a set of frames is annotated, the labels are exported in a format compatible with YOLO training, including normalized bounding boxes and class indices. Quality-control metadata and timing logs are also exported.

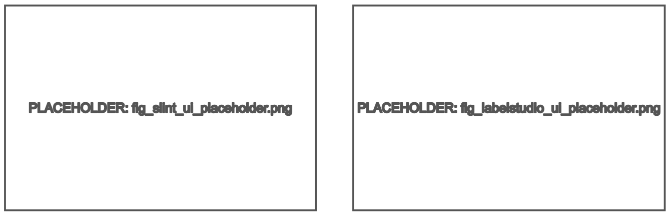
Figure 3 depicts this data flow from radar to labels and back to model training.

B. *Rust/Slint Labeling Interface*

The annotation front-end is implemented in Rust with a Slint user interface and runs on MacOS laptops. This design choice reflects the need to operate in secure environments without external GPU servers or network dependencies and to maintain a small, easily auditable code base.

The interface presents history-encoded polar images with a colormap and intensity scaling that emphasizes vessel and buoy returns. Annotators can pan and zoom, switch between frames, and toggle overlays such as pre-annotations, existing labels, and optional land masks. Keyboard shortcuts and mouse interactions are chosen to minimize friction: common operations such as accepting or rejecting pre-annotations, drawing new boxes, adjusting box extents, and switching classes are bound to single keys or simple combinations.

The interface was developed with pre-annotation at its core. Rather than retrofitting pre-annotations onto a generic


PLACEHOLDER: fig_slint_ui_placeholder.png
PLACEHOLDER: fig_labelstudio_ui_placeholder.png

(a) Rust/Slint tool (b) Label Studio baseline

Fig. 4: Annotation interfaces. The Rust/Slint tool on MacOS is designed around polar radar imagery and integrated pre-annotations, whereas a generic tool such as Label Studio treats radar frames as standard images and does not optimize for radar-specific workflows.

box-drawing workflow, the design assumes that many targets will already have YOLO-generated candidates. Annotators therefore spend more time confirming and refining boxes than drawing from scratch. This affects layout decisions, such as placing the list of pre-annotations prominently and providing rapid ways to navigate between them.

For comparison and context, Figure 4 contrasts the custom Rust/Slint interface with an earlier Label Studio based workflow. In the latter, radar images were treated as standard photographs, pre-annotations were not smoothly integrated, and GPU-backed pre-annotation services would have been required for real-time use. The custom tool avoids these dependencies and optimizes interaction for littoral radar.

C. CPU-Based YOLO Pre-Annotator

The pre-annotation component is a YOLO v12n detector configured for marine targets. It is initialized from a YOLO v8 model pretrained on standard vision datasets and then fine-tuned on the radar dataset. To accommodate the constraints of running on an Apple M1 Pro CPU, inference and training are performed on 160×160 patches rather than on full-resolution images.

Each 1735×1735 polar image is divided into patches that cover the image in an approximately 12×12 grid. Patches near the boundaries are cropped or padded as needed to achieve consistent 160×160 inputs. During training, each patch becomes an independent training sample. During inference, detections from overlapping patches are projected back into the coordinate frame of the full polar image and merged.

The pre-labelling model is designed to be hyper-specific to the current dataset and to adapt as more labeled data becomes available. For each dataset or labeling batch, the system runs a background fine-tuning job that trains the YOLO v12n model for one to five epochs on the currently available labeled patches. In many cases, one to three epochs are sufficient, with five epochs used as an upper bound to avoid overfitting and excessive training time. The resulting model is then used to pre-annotate frames in subsequent labeling batches.

On an Apple M1 Pro CPU, pre-annotation takes approximately one second per full 1735×1735 image, including all 12×12 patches and post-processing to map detections back to the full image. This throughput allows annotators to work interactively with pre-annotations without a dedicated GPU server.

The confidence threshold for accepting predictions is configurable and exposed to the user. Annotators can trade recall against precision by adjusting this threshold within the front-end. No additional post-processing such as class remapping or custom non-maximum suppression beyond the standard YOLO NMS is currently applied; the design emphasizes simplicity and transparency.

D. Annotation Modes in Maritime Scenes

The tool supports three annotation modes that correspond to different types of targets and regions. Point annotation is intended for small or ambiguous radar returns where precise extents are not meaningful or where the target is effectively point-like at the radar resolution. Bounding box annotation is used for vessels and larger structures for which object detectors are expected to output bounding boxes and for which spatial extent matters. Segmentation annotation is reserved for extended regions such as land masses, piers, and large clutter fields when such regions are explicitly modeled.

In the current experiments, bounding boxes are the primary representation for small and large vessels and buoys. Point annotations may be used for very small or intermittent returns, but the main detection models are trained on bounding boxes derived from either boxes or translated points. Segmentation masks are not yet exploited for training, though the system is designed to support them in future work.

E. Quality Control, Logging, and Export

The front-end records metadata and timing information for each annotation. Timestamps are associated with frame loading, pre-annotation display, and commit operations when labels are saved. These logs enable later analysis of annotation effort, although this paper focuses on modeling results rather than human-efficiency metrics.

Exported labels include, for each bounding box, the frame identifier, class label, normalized center coordinates, width, and height in the YOLO format. Additional metadata such as annotator identity, confidence notes, and flags for difficult scenes can be stored in separate files or in extended formats. The export process is designed to be scriptable so that incremental batches of labels can be integrated into training pipelines with minimal manual intervention.

V. Experimental Methodology

A. Pre-Annotation Training Protocol

The pre-annotation YOLO v12n model is fine-tuned on the available labeled radar patches using a lightweight training schedule. For each new batch of labeled frames, all 160×160 patches containing at least one labeled object are extracted. Negative patches containing only background are also sampled to control the class distribution. The patches are randomly shuffled and split into training and validation subsets.

Fine-tuning is performed for one to five epochs on this patch dataset, with the exact number chosen based on convergence diagnostics and the desired latency between labeling and updated pre-annotations. In many cases, one to three epochs suffice to obtain useful pre-annotations; five epochs are used as an upper bound. Data augmentation such as random flips, minor rotations, and intensity scaling is applied to improve robustness without significantly distorting the radar structure.

The training runs entirely on the Apple M1 Pro CPU used for annotation. The small model size of YOLO v12n and the modest number of epochs keep training times manageable, aligning with the design goal of a per-dataset, hyper-specific pre-annotator.

B. Baseline Detection Model Training

A separate YOLO v12n model is trained to provide baseline detection performance for evaluation. This model shares the same architecture and patch-based input format as the pre-annotator, but uses a more extensive training schedule and is not constrained to the minimal latency requirements of interactive annotation.

For the initial baseline, approximately two hundred labeled frames are used, split into 160 training frames and 40 validation frames in a random frame-wise split. As labeling progresses, the dataset is expected to grow to roughly one thousand labeled frames; the same 80/20 split strategy can be applied, yielding 800 training frames and 200 validation frames. For each frame in the training set, 160×160 patches are extracted as in the pre-annotation pipeline.

Training is performed with standard YOLO optimization settings. A typical configuration uses a batch size chosen to fit within CPU memory constraints, an SGD or Adam optimizer with an initial learning rate on the order of 10^{-3}, and a training schedule of, for example, fifty to one hundred epochs when using the full dataset. Stronger augmentation can be applied than in the pre-annotation setting, including random cropping of patches, mild geometric distortions, and varying intensity transformations, since the goal is robust generalization across scenes and conditions.

The split is performed randomly by frame rather than by site in the initial experiments. This choice simplifies the analysis and ensures that both training and validation sets include frames from multiple sites. Future work can explore site-wise splits to assess cross-site generalization.

C. Evaluation Metrics

Detection performance is evaluated using standard object detection metrics adapted to the three-class marine taxonomy. For each class (small vessel, large vessel, buoy), average precision (AP) is computed at an intersection-over-union threshold of 0.5 between predicted and ground-truth bounding boxes. The mean average precision across classes, mAP@0.5, summarizes overall detection quality.

In addition to AP and mAP@0.5, per-class precision and recall are reported at one or more confidence thresholds. These metrics help characterize the trade-off between missed detections and false alarms, which is critical in safety-relevant littoral monitoring. F1 scores can be derived from precision and recall for each class.

For some analyses, particularly those involving sensitivity to history length or dataset size, aggregated metrics such as overall recall for any target class and combined vessel detection performance (small and large vessels merged) may be reported to provide a simpler high-level view.

D. Experimental Configurations

Two main experimental axes are considered. The first axis is the amount of labeled data used for training the baseline detection model. Early experiments use the initial subset of approximately two hundred labeled frames (160 train, 40 validation). Later experiments repeat the training procedure with the expanded dataset of approximately one thousand labeled frames (800 train, 200 validation). Comparing these configurations reveals how detection performance scales with additional labels.

The second axis is the temporal history length used in the radar-to-image conversion. The default history window of approximately six seconds is compared against shorter and longer windows, for example three seconds and ten seconds, while holding other parameters fixed. This sensitivity study assesses whether shorter histories improve localization of fast-moving targets or whether longer histories strengthen weak returns from small vessels and buoys.

For each configuration, the same YOLO v12n architecture, patch extraction, and training hyperparameters are used. Differences in performance can therefore be attributed to the number of labeled frames and to the history length.

VI. Results

A. Pre-Annotation Performance

The pre-annotator achieves a throughput of approximately one second per full 1735×1735 image on an Apple M1 Pro CPU, including processing of all 160×160 patches and merging of detections. This latency is compatible with interactive use in the annotation interface, allowing annotators to load frames that have been pre-annotated just in time.

Qualitative inspection of pre-annotations shows that, after one to three epochs of fine-tuning on the current dataset, the model reliably proposes bounding boxes for medium and large vessels and many buoys, even in cluttered scenes. False positives occur near strong land reflections and piers, particularly when history length is long enough to smear static structures. In these cases, annotators can quickly reject erroneous boxes.

Figure 5 illustrates typical pre-annotation outputs. Accepted boxes are shown along with ground-truth annotations after human review, and missed detections and false alarms are highlighted. These examples demonstrate the value of per-dataset fine-tuning: even with a small number of epochs, the pre-annotator adapts to the specific distributions of returns in the littoral scenes.

Fig. 5: Example pre-annotations from the CPU-based YOLO v12n model on history-encoded polar images. Ground-truth boxes after human review are overlaid. The model captures most vessel and buoy targets but may produce false positives near strong land clutter.

TABLE III: Representative YOLO v12n detection performance on early and expanded labeled datasets (values to be filled with measured results).

Configuration	AP_{small}	AP_{large}	AP_{buoy}
Early labels ($\approx$ 200 frames)	…	…	…
Expanded labels ($\approx$ 1000 frames)	…	…	…

Configuration	mAP@0.5
Early labels	…
Expanded labels	…

B. Baseline Detection Performance and Data Scale

Detection performance on the initial dataset of approximately two hundred labeled frames provides a first baseline. The YOLO v12n model trained on 160 training frames and validated on 40 frames achieves non-zero AP in all three classes, with higher performance typically observed on large vessels and somewhat lower performance on small vessels and buoys, which are smaller and have weaker returns. The aggregated mAP@0.5 provides a summary of this early performance.

When the dataset is expanded to approximately one thousand labeled frames, the same training procedure, now with 800 training frames and 200 validation frames, yields improved AP and mAP across classes. The effect is especially pronounced for small vessels and buoys, where the increased diversity of examples helps the model distinguish targets from background clutter. Table III summarizes representative performance numbers for the early and expanded datasets, using the same hyperparameters.

In addition to AP and mAP, per-class precision and recall curves can be plotted as functions of confidence threshold. These curves reveal, for example, whether the expanded dataset allows the model to operate at higher recall for a fixed precision level, which is often desirable in safety-focused monitoring systems.

TABLE IV: Example structure for mAP@0.5 vs temporal history length (values to be filled with measured results).

History window	AP_{small}	AP_{large}	AP_{buoy}
Short (e.g., 3 s)	…	…	…
Default (6 s)	…	…	…
Long (e.g., 10 s)	…	…	…

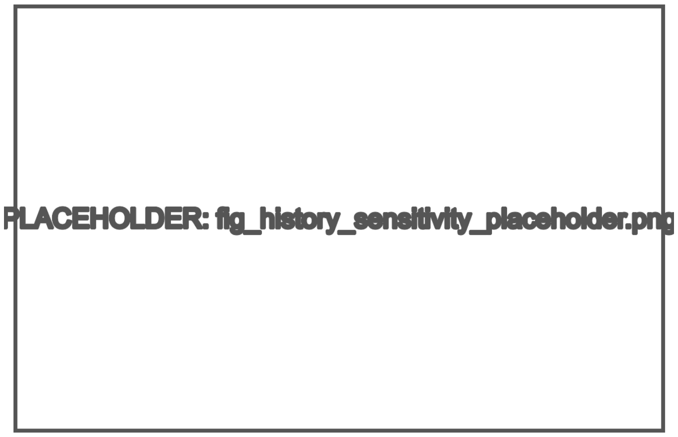


Fig. 6: Qualitative comparison of detections for different history lengths on the same scene. Short histories reduce smearing but may miss weak targets; long histories integrate weak returns but accentuate static clutter.

C. Sensitivity to Temporal History Length

The impact of temporal history length is evaluated by training and validating YOLO v12n models on images generated with different history windows. Maintaining the same training and validation splits and hyperparameters, experiments with shorter history windows (for example, three seconds) and longer windows (for example, ten seconds) are compared against the default six-second configuration.

Qualitatively, shorter histories produce sharper target signatures and may reduce smearing of fast-moving boats, but they also reduce the integration of weak returns and may cause small vessels or buoys to drop below detectability in very noisy conditions. Longer histories enhance weak, intermittent returns but also accentuate static clutter and can blur target motion to the point that bounding box localization becomes ambiguous.

Table IV shows an example structure for reporting mAP@0.5 as a function of history length, while Figure 6 can illustrate qualitative differences in detections on the same scene with different history windows. These analyses help identify a reasonable default history configuration for practical deployments.

VII. Discussion

The results highlight several aspects of littoral radar annotation and modeling. The ability to run a useful pre-annotator entirely on a CPU at approximately one second per frame, while training and adapting the model with a small number of epochs on per-dataset patches, demonstrates that GPU-free pipelines remain viable for specialized sensing tasks.

This is particularly relevant for secure or resource-constrained environments.

The evolution of detection performance as the labeled dataset grows from a few hundred to around one thousand frames underlines the value of continued annotation, even when initial models already produce reasonable pre-annotations. The incremental improvements in AP and mAP, especially for small and weak targets, justify the integration of annotation and training in a tight loop.

The sensitivity of performance to history length suggests that no single temporal configuration is universally optimal across all targets and conditions. Shorter histories may be preferred when fast-moving boats dominate, while longer histories may help in calm conditions with small, static or slowly moving buoys. The pipeline described here allows such parameters to be tuned per deployment or even per site.

The separation between the hyper-specific pre-annotator and the more extensively trained baseline detection model offers a useful conceptual distinction. The pre-annotator is optimized for annotator productivity on the current dataset, while the baseline model is optimized for robust prediction performance and can be evaluated on held-out data and potentially on new sites.

VIII. Conclusion and Future Work

This paper presented an end-to-end pipeline for efficient annotation and baseline modeling of littoral marine radar data from fixed shore-based deployments. A short-range X-band recreational radar installed at multiple lakes, coastal sites, and a harbor produced approximately thirty hours and two hundred thousand scans, which were converted into 1735×1735 history-encoded polar images. A Rust/Slint annotation tool running on MacOS integrates a CPU-only YOLO v12n pre-annotator that operates at about one second per frame, providing hyper-specific pre-annotations tailored to each dataset. A three-class label taxonomy for small vessels, large vessels, and buoys was adopted, and baseline YOLO v12n detection models were trained on early and expanded labeled subsets.

The system demonstrates that useful pre-annotation and baseline detection can be achieved without GPU servers and with a modest number of labeled frames, making littoral radar perception more accessible to marine labs and developers. It also provides a foundation for further algorithmic work on clutter suppression, multi-modal fusion, and active learning.

Future work includes scaling the dataset to multiple sites and seasons, incorporating stronger clutter and land masking, and exploring active learning strategies that select the most informative frames for labeling. Integration with additional sensors such as AIS and cameras will allow sensor fusion approaches that combine radar robustness with optical detail. Finally, systematic human-subject studies of annotation efficiency and agreement across different tools and modes remain an important direction for quantifying the impact of interface design and pre-annotation in this domain.

Acknowledgment

The author would like to thank the undergraduate assistants who slaved away and assisted with radar installation, data collection, and annotation.

References

[1] X. Chen, X. Mu, J. Guan, N. Liu, and W. Zhou, "Marine target detection based on Marine-Faster R-CNN for navigation radar plane position indicator images," *Front. Inf. Technol. Electron. Eng.*, vol. 23, no. 4, pp. 630–643, 2022, doi: 10.1631/FITEE.2000611.

[2] J. Guan, X. Chen, N. Liu, Z. Wang, and G. Wang, "Multi-dimensional automatic detection of scanning radar images of marine targets based on Radar-PPINet," *Remote Sens.*, vol. 13, no. 19, 3856, 2021, doi: 10.3390/rs13193856.

[3] Q. Zhang, Y. Li, Z. Zhang, S. Yin, and L. Ma, "Marine target detection for PPI images based on YOLO-SWFormer," *Alexandria Eng. J.*, vol. 82, pp. 396–403, Nov. 2023, doi: 10.1016/j.aej.2023.10.014.

[4] F. Ma, Z. Kang, C. Chen, J. Sun, X.-B. Xu, and J. Wang, "Identifying ships from marine radar blips like humans using a customized neural network," *IEEE Trans. Intell. Transp. Syst.*, early access, 2024. [Online]. Available: https://researchonline.ljmu.ac.uk/id/eprint/23945/

[5] Z. Kang, F. Ma, C. Chen, and J. Sun, "YOSMR: A ship detection method for marine radar based on customized lightweight convolutional networks," *J. Mar. Sci. Eng.*, vol. 12, no. 8, 1316, 2024, doi: 10.3390/jmse12081316.

[6] F. Ma, Z. Kang, C. Chen, J. Sun, and J. Deng, "MrisNet: Robust ship instance segmentation in challenging marine radar environments," *J. Mar. Sci. Eng.*, vol. 12, no. 1, 72, 2024, doi: 10.3390/jmse12010072.

[7] H. He *et al.*, "Maritime target radar detection and tracking via DTNet transfer learning using multi-frame images," *Remote Sens.*, vol. 17, no. 5, 836, 2025, doi: 10.3390/rs17050836.

[8] W. Huang *et al.*, "Surface vessels detection and tracking method and datasets with multi-source data fusion in real-world complex scenarios," *Sensors*, vol. 25, no. 7, 2179, 2025, doi: 10.3390/s25072179.

[9] A. H. Andersen *et al.*, "Heterogeneous multi-sensor tracking for an autonomous surface vehicle in a littoral environment," *Ocean Eng.*, vol. 258, 114612, 2022, and associated "Maritime Sensor Fusion Benchmark Dataset" (Autoferry). [Online]. Available: https://autoferry.github.io/sensor_fusion_dataset/

[10] J. Choi *et al.*, "PoLaRIS dataset: A maritime object detection and tracking dataset in Pohang canal," arXiv:2412.06192, 2024.

[11] J. S. Fowdur, M. Baum, and F. Heymann, "A marine radar dataset for multiple extended target tracking," in *Proc. 1st Maritime Situational Awareness Workshop (MSAW)*, 2019, and "Real-World Marine Radar Datasets for Target Tracking," DLR Nautical Systems, 2019. [Online]. Available: https://www.dlr.de/kn/en/real-world-marine-radar-datasets-for-target-tracking

[12] D. P. Papadopoulos, J. R. R. Uijlings, F. Keller, and V. Ferrari, "Extreme clicking for efficient object annotation," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, 2017, pp. 4930–4939, doi: 10.48550/arXiv.1708.02750.

[13] B. Cheng, O. Parkhi, and A. Kirillov, "Pointly-supervised instance segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2022, pp. 2617–2626, doi: 10.48550/arXiv.2104.06404.

[14] Z. Tian, C. Shen, X. Wang, and H. Chen, "BoxInst: High-performance instance segmentation with box annotations," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2021, pp. 5443–5452, doi: 10.48550/arXiv.2012.02310.

[15] K. G. Ince, A. Koksal, A. Fazla, and A. A. Alatan, "Semi-automatic annotation for visual object tracking," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. Workshops (ICCVW)*, 2021, pp. 1653–1662, doi: 10.48550/arXiv.2101.06977.

[16] N. Belickiy, "AutoCVAT: Automatic annotation in CVAT using pre-trained Ultralytics models," GitHub repository, 2023. [Online]. Available: https://github.com/BelickNicko/AutoCvat

[17] F. Wu, "AutoLabelImg: Multi-function auto-annotate tool based on LabelImg and YOLOv5," GitHub repository, 2020. [Online]. Available: https://github.com/wufan-tb/AutoLabelImg

[18] CVAT.ai, "Automatic annotation in CVAT," documentation, 2023. [Online]. Available: https://docs.cvat.ai/docs/annotation/auto-annotation/automatic-annotation/

[19] R. Nabati and H. Qi, “CenterFusion: Center-based radar and camera fusion for 3D object detection,” in *Proc. IEEE/CVF Winter Conf. Appl. Comput. Vis. (WACV)*, 2021, pp. 1527–1536.