Human-Supervised CFAR Autotuning for Maritime Sensing

Nathan Kirk*, J.C. Vaught†, Douglas Cahl‡, Yi Wang§
University of South Carolina, Columbia, SC, 29208

Radar-based target detection in cluttered environments requires robust detection under tight size, weight, power, and cost constraints. Tuning Constant False Alarm Rate (CFAR) detectors for heterogeneous scenes is difficult due to heavy-tailed clutter, terrain transitions, and multipath, yet remains essential for low-latency situational awareness. We present a human-supervised methodology and open-source browser-based tool, *CFAR Studio*, for tuning CFAR parameters of commercial-off-the-shelf radars. The operator annotates stationary reference objects on one or more radar frames using an interactive blob-tracing tool. The system then performs an exhaustive grid search over three CFAR variants – Cell-Averaging (CA), Greatest-Of (GO), and Smallest-Of (SO) – evaluating each parameter configuration against the annotated ground truth via an area-weighted coverage score. Multi-frame temporal stability is assessed by tracking detection consistency across sequential radar sweeps. An integral-image optimization enables the evaluation of hundreds of configurations in seconds entirely within the browser. The system is demonstrated on real X-band radar data collected with a Furuno NXT DRS4D radar and Raspberry Pi 5 at Lake Murray, South Carolina.

I. Introduction

Constant False Alarm Rate (CFAR) detectors are the dominant target-detection paradigm in radar systems, valued for their controllable false-alarm probability, local-statistics design, and suitability for embedded implementations [1, 2]. Under homogeneous, Gaussian clutter the threshold multiplier admits a closed-form solution and detection performance is well characterized [3, 4]. Yet real-world environments violate every assumption this theory depends on – clutter is heavy-tailed and spatially nonhomogeneous, well modeled by K-distribution and Pareto families rather than Rayleigh statistics [5, 6]; terrain transition edges introduce abrupt power discontinuities; and multipath, mixed cooperative/non-cooperative traffic, and environmental returns further corrupt the reference window. As a result, selecting appropriate CFAR parameters is notoriously scene- and sensor-dependent – a configuration that works in one region may generate excessive false alarms in another, while aggressive clutter rejection near terrain boundaries may mask small targets of interest [7]. Currently, operators tune guard cells, reference windows, and false-alarm rates by manual trial-and-error, adjusting settings until the display "looks right." This process is slow, subjective, and not reproducible.

Researchers have sought to address these limitations through adaptive and knowledge-based CFAR schemes. Variability-Index CFAR dynamically switches between CA, GO, and SO variants via hypothesis tests on the reference window [8]; knowledge-based detectors exploit GIS databases to select training samples appropriate to the local terrain [9]; and cognitive CFAR approaches compare local clutter statistics against prior performance models to adjust window length [10]. More recently, deep-learning detectors have demonstrated strong accuracy on radar data [11]. However, each approach carries practical constraints that limit operational adoption. Adaptive switching heuristics degrade when interfering targets span both halves of the reference window [8]. Knowledge-based methods require pre-built environmental databases that do not exist for ad-hoc deployments in unfamiliar environments [9]. Cognitive CFAR optimizes window length alone, leaving the choice of variant, guard-cell size, and P_{FA} to the operator [10]. Deep-learning detectors abandon the CFAR property entirely, cannot guarantee false-alarm control, and require large labeled datasets that are unavailable in most operational settings [12]. Despite decades of effort, no method systematically searches across multiple CFAR variants and the full parameter space to find the best configuration for a given scene.

The tooling landscape compounds the problem. Commercial radar processors from major vendors embed proprietary CFAR logic with limited or no parameter exposure, while third-party research tools that do expose the full parameter set – most notably MATLAB's Radar Toolbox [13] – require expensive licenses, desktop installations, and cannot be

*Undergraduate Student, Department of Mechanical Engineering, AIAA Student Member, Member ID: 1924156.
†Graduate Student, Department of Mechanical Engineering, AIAA Student Member, Member ID: 1537189.
‡Professor, Department of Mechanical Engineering.
§Professor, Department of Mechanical Engineering.

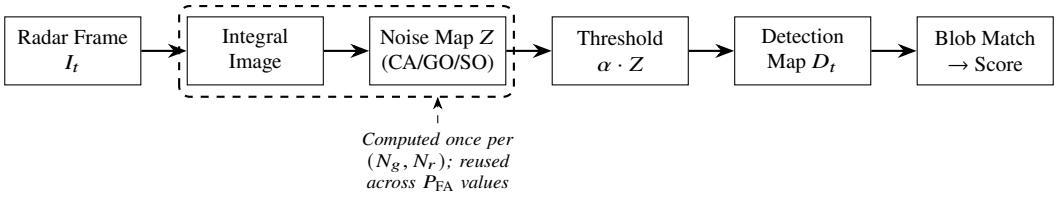

Fig. 1 CFAR detection pipeline for a single parameter configuration. The noise map depends only on guard and reference cell counts, enabling *loop hoisting*: for each (N_g, N_r) pair, the noise map is computed once and reused across all P_{FA} values, reducing redundant computation.

deployed on edge hardware or shared without infrastructure. Yet field operators, small teams, and academic researchers – the users who most need tunable detection for challenging scenes – are precisely those without access to these toolchains. No open, installation-free tool exists that allows an operator to experiment with CFAR configurations in situ.

A further gap lies in how CFAR configurations are evaluated. Standard assessment relies on probability-of-detection and probability-of-false-alarm curves computed against synthetic injected targets or single-frame detection counts [4, 14]. However, this ignores two operational realities. First, radar operators already possess direct situational awareness of which stationary objects – towers, pylons, buildings, infrastructure – should appear on the display. This knowledge constitutes a free, high-quality source of ground truth, yet no prior work formalizes operator annotation as a quantitative objective function for CFAR optimization. Second, a configuration that detects a target on one sweep but loses it on the next is operationally useless for tracking and situational awareness; nevertheless, temporal consistency across sequential radar sweeps is absent from standard CFAR evaluation metrics.

This paper presents a human-supervised CFAR autotuning framework that addresses these gaps. The contributions are: (i) a formal scoring function anchored to human-annotated ground-truth blobs, using area-weighted coverage ratios rather than binary detection indicators; (ii) an exhaustive grid search over three CFAR variants (CA, GO, SO) with integral-image loop hoisting that enables evaluation of hundreds of configurations in seconds; (iii) a multi-frame temporal stability metric that quantifies detection consistency across sequential radar sweeps; and (iv) an open-source, fully client-side browser-based tool, *CFAR Studio*, that requires no installation, server infrastructure, or specialized hardware beyond a modern web browser. The system is demonstrated on real X-band radar data collected with a Furuno DRS4D-NXT radar and Raspberry Pi 5 at Lake Murray, South Carolina.

II. Background and Related Work

The CA-CFAR family estimates background power via local training windows and compares the cell-under-test to a scaled noise estimate to maintain a desired P_{FA} [2, 3]. For N independent, identically distributed reference cells drawn from an exponential background, the threshold multiplier admits the closed-form expression [3, 4]:

$$\alpha = N\left(P_{\mathrm{FA}}^{-1/N} - 1\right). \tag{1}$$

GO- and SO-CFAR improve robustness near clutter edges and in multi-target scenes by partitioning leading and trailing reference windows and selecting the greatest-of or smallest-of their respective noise estimates [15, 16]. Other variants – including Order-Statistic (OS-CFAR), Variability-Index, and censored/trimmed-mean CFAR – address additional heterogeneous-background scenarios [1] but are beyond the scope of the current implementation.

High-resolution radar clutter is heavy-tailed and spatially nonhomogeneous. The K-distribution, a compound Gaussian model with gamma-distributed texture, has been widely adopted since Ward et al. [5]; however, Fioranelli et al. [6] showed that Pareto-plus-noise models better capture the spiky returns observed at high grazing angles in X-band data. Comparative studies across 1–10 GHz confirm that no single distribution fits all polarizations, resolutions, and sea states [17]. These non-Gaussian, site-specific statistics are particularly problematic for CFAR detectors. Training windows that span terrain transitions or nearby infrastructure can severely bias noise estimates, inflating or suppressing the detection threshold in ways that the closed-form α cannot anticipate.

Several lines of work have sought to automate CFAR parameter selection. Humphreys et al. [10] proposed a cognitive CFAR that adapts window length by comparing local statistics against prior performance models, but the approach does not address variant selection, guard-cell sizing, or P_{FA}. Variability-Index CFAR [8] switches dynamically among CA, GO, and SO modes via hypothesis tests on the reference window; yet detection probability degrades when interfering

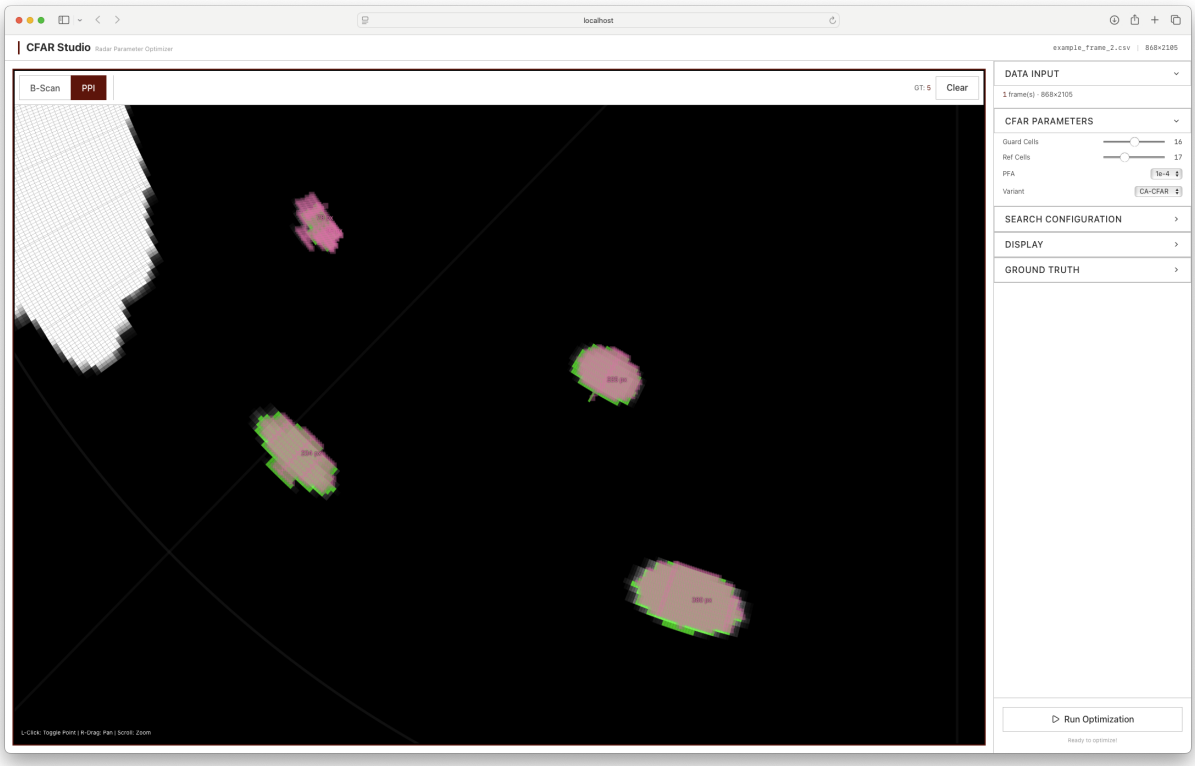

Fig. 2 Close-up of ground truth annotations on the PPI display. The operator clicks on a detected target; an 8-connected flood fill traces the contiguous detection blob, capturing its exact pixel footprint as the reference shape for scoring.

targets are not confined to a single half-window. Neural-network-based variant selectors [18] and CFAR-constrained deep-learning detectors [12] have shown accuracy improvements in simulation, but require offline training data, GPU infrastructure, and retraining when the sensor or scene changes. Cross-sensor supervision using lidar has been explored for automotive radar [19]; however, synchronized lidar is unavailable on most platforms and the resulting detector abandons the CFAR property entirely.

Despite these advances, we are not aware of prior work that jointly searches over CFAR variant, guard cells, reference cells, and P_{FA} using human-annotated ground truth as the objective function. Knowledge-based CFAR schemes [9] leverage GIS databases but require pre-built environmental maps unavailable for ad-hoc deployments. Expert-system approaches [20] automate the operator out of the loop rather than harnessing operator situational awareness. Our approach treats the operator's annotation of stationary reference objects as quantitative ground truth, defining a formal objective function that the system optimizes via exhaustive search – combining human judgment with computational efficiency in a browser-deployable tool.

III. Problem Formulation

Let $I_t \in \mathbb{R}^{M\times N}$ denote radar intensity frames in range–azimuth coordinates, $t = 1, \ldots, T$. A CFAR variant $v \in \{\mathrm{CA}, \mathrm{GO}, \mathrm{SO}\}$ with parameter vector $\boldsymbol{\theta} = (N_g, N_r, P_{\mathrm{FA}})$ – guard cells, reference cells, and probability of false alarm – is applied to each frame to produce a binary detection map $D_t(v, \boldsymbol{\theta}) \in \{0, 1\}^{M\times N}$.

A. CFAR Detection Variants

All three variants considered in this work – CA, GO, and SO – share a common two-dimensional sliding-window architecture applied independently to each pixel of the radar frame. For a given cell-under-test at pixel (i, j) in frame I_t, the detector defines two concentric rectangular regions: a *guard region* of half-width N_g that excludes the cell-under-test

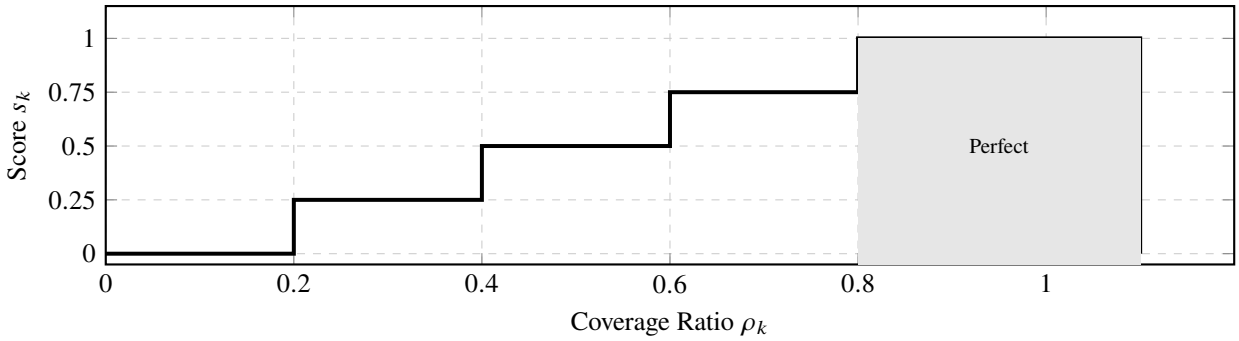

Fig. 3 Scoring function mapping area coverage ratio ρ_k to discrete score s_k. Configurations that capture 80–110% of the ground truth blob area receive a perfect score; partial detections are penalized proportionally.

and its immediate neighbors from the noise estimate, and a *reference annulus* of width N_r that surrounds the guard region and provides the training samples from which the local noise level is estimated. The guard region prevents the target's own energy from inflating the noise estimate, while the reference annulus should be large enough to yield a statistically reliable background estimate yet small enough to remain within a locally homogeneous clutter region. To avoid the computational cost of summing over the reference annulus at every pixel, the implementation pre-computes an integral image (summed area table) [21] for each frame, enabling $O(1)$ evaluation of the sum over any axis-aligned rectangular region. With the integral image in place, the three variants differ only in how they aggregate the reference cells into a single noise estimate $Z_{i,j}$.

1. CA-CFAR

Cell-Averaging CFAR computes the arithmetic mean of all $N = (2R_o + 1)^2 - (2R_i + 1)^2$ reference cells, where $R_o = N_g + N_r$ is the outer half-width and $R_i = N_g$ is the inner half-width:

$$Z_{i,j}^{\mathrm{CA}} = \frac{1}{N} \left(S_{\mathrm{outer}} - S_{\mathrm{inner}} \right), \tag{2}$$

where S_{outer} and S_{inner} are the summed-area-table queries over the outer and inner rectangles, respectively. CA-CFAR is optimal under homogeneous, exponentially distributed clutter but can suffer threshold elevation when strong interferers or clutter edges fall within the reference annulus.

2. GO-CFAR

Greatest-Of CFAR mitigates the clutter-edge problem by splitting the reference annulus into two half-windows – one leading (above) and one trailing (below) the cell-under-test – and selecting the larger of the two noise estimates. Let $\bar{Z}_1$ and $\bar{Z}_2$ denote the mean power of each half-window, each containing N_{half} cells:

$$Z_{i,j}^{\mathrm{GO}} = \max(\bar{Z}_1, \bar{Z}_2). \tag{3}$$

By adopting the higher noise estimate, GO-CFAR raises the threshold near clutter transitions, suppressing false alarms at the cost of reduced detection sensitivity for weak targets.

3. SO-CFAR

Smallest-Of CFAR takes the opposite approach, selecting the lower of the two half-window estimates:

$$Z_{i,j}^{\mathrm{SO}} = \min(\bar{Z}_1, \bar{Z}_2). \tag{4}$$

This preserves sensitivity near clutter edges – useful for detecting targets that lie adjacent to terrain or structures – but at the expense of an elevated false-alarm rate in heterogeneous regions where the quieter half-window underestimates the true background.

For both GO- and SO-CFAR, the effective reference cell count used for the threshold multiplier α (Eq. 1) is N_{half} rather than the full annulus count N, reflecting the reduced number of independent training samples.

Algorithm 1 Grid Search with Loop Hoisting

Input: frames $\{I_t\}_{t=1}^{T}$, ground truth blobs $\{\mathcal{B}_k\}$, variant v, parameter ranges
Pre-compute integral images for all frames
for each (N_g, N_r) in parameter grid **do**
  Compute noise map Z for all frames ▷ *Expensive step*
  for each P_{FA} value **do**
    $\alpha \leftarrow N_{\text{eff}}(P_{\text{FA}}^{-1/N_{\text{eff}}} - 1)$
    Apply threshold: $D_t(i,j) \leftarrow \mathbb{1}[I_t(i,j) > \alpha \cdot Z_{i,j}]$
    Compute coverage scores $\{s_k\}$, frame score, false alarms, stability
  end for
end for
Return all results sorted by score

B. Detection Rule

Given the noise estimate from any of the three variants, a detection is declared when the cell-under-test exceeds the scaled noise floor:

$$D_t(i,j) = \mathbb{1}\left[I_t(i,j) > \alpha \cdot Z_{i,j}\right], \tag{5}$$

where the threshold multiplier α is given by Eq. (1) using the appropriate effective cell count. The resulting binary detection map D_t is then evaluated against the operator's ground truth annotations as described in the following subsections.

The operator annotates stationary reference objects (e.g., towers, buildings, infrastructure) on the radar PPI display. When the operator clicks on a detected pixel, an 8-connected flood-fill algorithm traces the contiguous detection blob $\mathcal{B}_k$, collecting all connected pixels into a set of indices. This "magic wand" approach captures the exact shape and area of each reference object as it appears in the current CFAR detection, rather than requiring the operator to draw bounding boxes or estimate target extent. The resulting blob pixel set $\mathcal{B}_k = \{p_1, p_2, \ldots, p_{|\mathcal{B}_k|}\}$ serves as the ground truth reference for scoring.

For each ground truth blob $\mathcal{B}_k$ and a candidate detection map D_t, we compute the *area coverage ratio*:

$$\rho_k = \frac{\sum_{p \in \mathcal{B}_k} D_t(p)}{|\mathcal{B}_k|}. \tag{6}$$

This ratio measures what fraction of the ground truth blob's pixels are detected by the candidate configuration. The coverage ratio is mapped to a discrete score via a piecewise function that rewards configurations capturing the full target shape:

$$s_k = \begin{cases} 1.00 & \text{if } 0.8 \le \rho_k \le 1.1, \\ 0.75 & \text{if } 0.6 \le \rho_k < 0.8, \\ 0.50 & \text{if } 0.4 \le \rho_k < 0.6, \\ 0.25 & \text{if } 0.2 \le \rho_k < 0.4, \\ 0.00 & \text{if } \rho_k < 0.2. \end{cases} \tag{7}$$

The upper bound of 1.1 accommodates slight detection expansion beyond the annotated blob boundary, which is common when a slightly more sensitive configuration captures peripheral pixels. This area-based approach provides substantially more nuance than a binary "detected/not-detected" metric; a configuration that detects only a small fragment of a large target is appropriately penalized, while one that captures the full target shape is rewarded.

The per-frame score across all K ground truth targets combines the worst-case and average target scores:

$$S_{\text{frame}} = 0.7 \cdot \min_{k \in [K]} s_k + 0.3 \cdot \frac{1}{K} \sum_{k=1}^{K} s_k. \tag{8}$$

The heavy weighting on the minimum score (0.7) enforces a critical constraint – a configuration that completely misses even one annotated target is severely penalized regardless of how well it detects the others. This prevents the optimizer

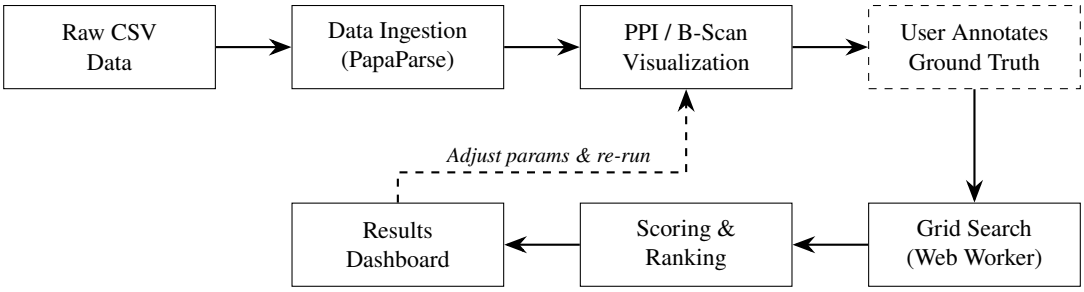

Fig. 4 CFAR Studio system pipeline. Dashed box indicates user interaction; dashed arrow shows the iterative feedback loop where the operator can adjust parameters and re-run optimization after reviewing results.

from recommending configurations that sacrifice coverage of difficult targets in favor of easy ones. In addition to target coverage, the false alarm count is computed as the number of detected pixels that fall outside all ground truth blobs:

$$\mathrm{FP} = \sum_{t=1}^{T} \left(\sum_{(i,j)} D_t(i,j) - \sum_{k=1}^{K} \sum_{p \in \mathcal{B}_k} D_t(p) \right). \tag{9}$$

This metric allows the operator to compare configurations that achieve similar coverage scores but differ in the number of spurious detections they produce.

When multiple sequential frames are available, temporal stability quantifies how consistently each target is detected across the frame sequence. For each ground truth target k, let c_k be the number of frames in which it is detected (score > 0). The stability metric is

$$\mathrm{Stability} = \frac{1}{K} \sum_{k=1}^{K} \frac{c_k}{T}. \tag{10}$$

A stability of 1.0 indicates that all targets are detected in every frame, while low stability reveals configurations that are marginally sensitive and produce intermittent detections – a failure mode that is operationally unacceptable for tracking and situational awareness even if the single-frame score is high.

The parameter space is defined by discrete ranges for guard cells N_g, reference cells N_r, and a set of P_{FA} values, with all combinations evaluated exhaustively. The key computational optimization exploits the fact that the noise map Z depends only on (N_g, N_r, v) and not on P_{FA}. By restructuring the search loop to iterate over (N_g, N_r) in the outer loop and P_{FA} in the inner loop, the expensive noise map computation is performed once per (N_g, N_r) pair and reused across all P_{FA} values (Fig. 1). Within the inner loop, only the scalar multiplication $\alpha \cdot Z_{i,j}$ and comparison in Eq. (5) are needed. This *loop hoisting* reduces the total number of integral-image queries by a factor equal to the number of P_{FA} values tested.

IV. System Architecture

CFAR Studio is implemented as a fully client-side web application using React 19 with TypeScript, built with Vite. All computation occurs in the user's browser with no server-side processing, no data upload, and no installation required. Plotly.js provides interactive PPI (Plan Position Indicator) and B-scan displays with multiple colormaps and zoom/pan controls. Radar CSV files are parsed via PapaParse, and a dedicated Web Worker thread performs the grid search without blocking the user interface.

Radar data is ingested from CSV files containing range–azimuth intensity matrices. Each CSV row corresponds to one azimuth spoke, with columns representing range bins. A companion azimuth array specifies the angular position of each spoke. The system supports multi-frame loading, enabling temporal analysis across sequential radar sweeps.

The primary display is a PPI view rendered in Cartesian coordinates, with options for multiple colormaps (grayscale, amber, green, inferno, ice). CFAR detections are overlaid as colored highlights, and ground truth annotations are displayed as distinct markers. A B-scan view (Fig. 6) provides an alternative range–azimuth representation useful for examining clutter structure and detection behavior along individual spokes.

The annotation workflow proceeds as follows: (i) the operator loads radar CSV data and selects CFAR parameters for an initial preview detection; (ii) on the PPI display, the operator clicks on detected targets representing known stationary

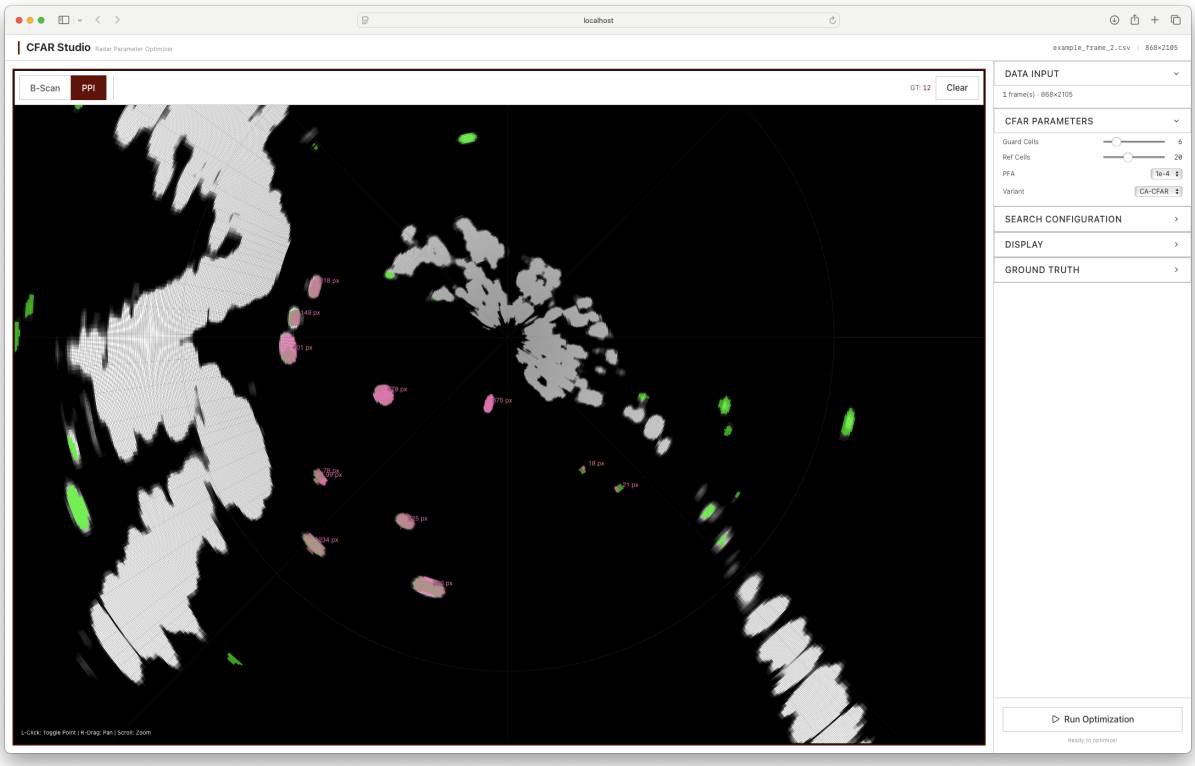

Fig. 5 PPI display showing raw radar intensity with CFAR detections (highlighted) and annotated ground truth blobs. The overlay enables visual comparison of detection coverage against known targets.

objects; (iii) the system performs an 8-connected flood fill from the click point, tracing the contiguous detection blob and recording all constituent pixel indices as the ground truth reference; (iv) the blob pixel count is displayed as a size indicator. Ground truth annotations persist across frames, with the assumption that the radar is stationary and reference objects do not move.

When the operator initiates optimization, the parameter ranges (guard cells, reference cells, P_{FA} values) and CFAR variant are dispatched to a Web Worker. The worker executes Algorithm 1, posting progress updates at regular intervals. Upon completion, results are returned to the main thread for visualization in a multi-dimensional results dashboard (Fig. 7), including sortable tables, parameter trend plots, and interactive filtering by score, false alarm count, and stability.

V. Experimental Setup and Results

Radar data was collected using a Furuno NXT DRS4D X-band (9.41 GHz) solid-state radar connected to a Raspberry Pi 5 single-board computer for raw data digitization and recording. The radar was operated in a shore-based configuration at Lake Murray, Columbia, South Carolina. Lake Murray provides a mixed environment with open water, shoreline structures, and fixed infrastructure – representative targets for CFAR tuning evaluation.

Raw radar returns were captured as CSV files, with each file representing one complete antenna rotation (360° sweep). Multiple sequential sweeps were recorded to enable multi-frame temporal analysis.

Stationary reference objects were annotated using the blob-tracing tool described in Section III.B. Targets included fixed structures and shoreline features that produce consistent radar returns across multiple sweeps. For each target, the operator clicked on the detection under an initial CFAR configuration, and the flood-fill algorithm captured the blob footprint as the ground truth reference.

The grid search swept guard cells $N_g \in [1, 10]$ in steps of 1, reference cells $N_r \in [5, 30]$ in steps of 5, and logarithmically spaced P_{FA} values in $[10^{-6}, 10^{-2}]$. Each of the three variants (CA, GO, SO) was optimized

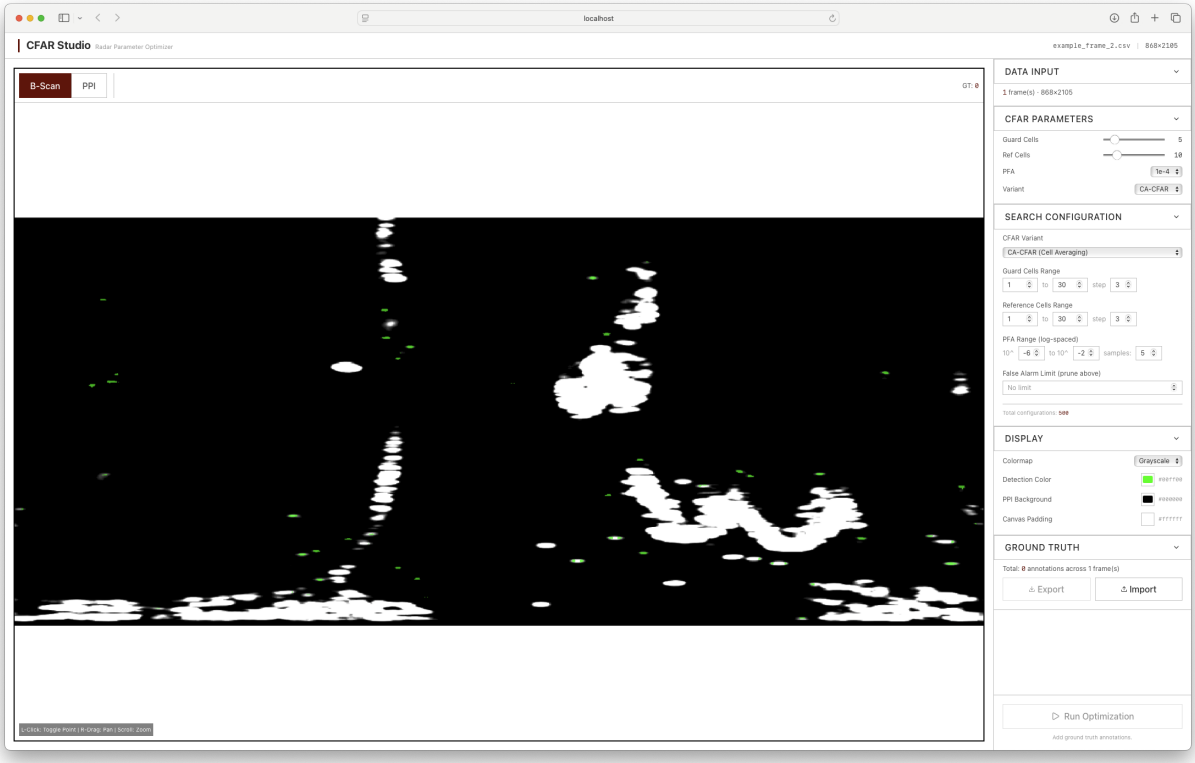

Fig. 6 B-scan (range vs. azimuth) view with CFAR detections overlaid. This representation reveals clutter structure along individual radar spokes and aids in understanding detection behavior near terrain transitions.

independently.

A typical grid of approximately 500 parameter configurations completes in under 10 seconds on a modern laptop browser, owing to the integral-image loop hoisting described in Section III.B.

The optimization results reveal several general trends across the tested CFAR variants. Larger guard cell counts ($N_g \geq 3$) consistently improve scores for extended targets such as dock structures, as they prevent the target's own return from biasing the noise estimate; for point-like targets such as buoys, guard cell sensitivity is lower. Moderate reference window sizes ($N_r \in [10, 20]$) provide the best trade-off between noise estimation accuracy and spatial homogeneity, while very large reference windows ($N_r > 25$) degrade performance near terrain transitions where the training window spans heterogeneous clutter regions.

Lower P_{FA} values reduce false alarms but risk missing weaker targets. The optimal P_{FA} depends strongly on the scene – open-water targets tolerate lower P_{FA}, while targets near clutter edges benefit from moderate P_{FA} to maintain detection. GO-CFAR tends to produce fewer false alarms near clutter edges at the cost of slightly reduced detection sensitivity, consistent with its conservative noise selection. SO-CFAR provides higher detection rates but generates more false alarms in heterogeneous regions, while CA-CFAR offers a middle ground and often produces the highest overall scores in relatively homogeneous scenes. Across multi-frame sequences, the top-scoring configurations for each variant typically achieve stability values above 0.9, indicating consistent detection of annotated targets across sequential sweeps.

VI. Discussion and Limitations

The method assumes quasi-stationary reference objects and a fixed radar platform, so that ground truth annotations remain valid across frames. Extension to moving targets or mobile radar platforms would require frame-to-frame registration or track-before-detect association. Running entirely in the browser imposes additional constraints: single-threaded JavaScript execution (mitigated by Web Worker offloading), no GPU acceleration, and a practical file size limit of approximately 50 MB per CSV. For the dataset sizes typical of X-band radar sweeps, these constraints are not limiting.

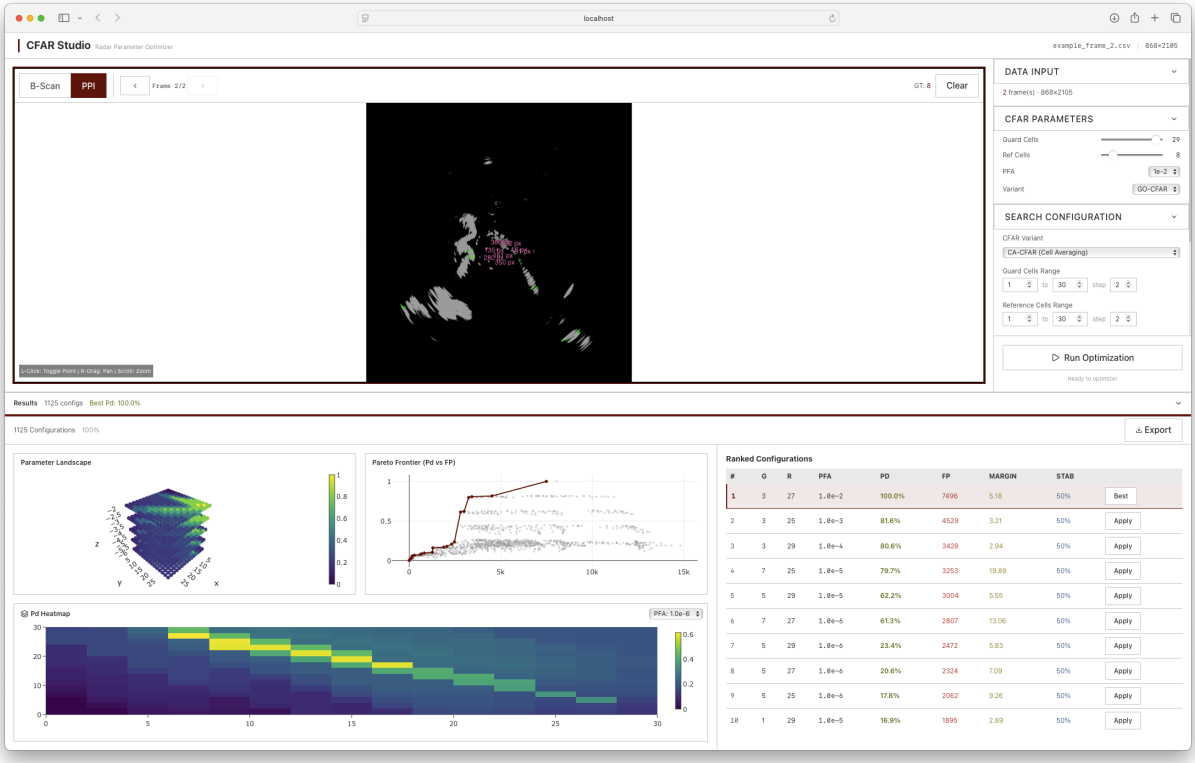

Fig. 7 Full CFAR Studio interface showing the PPI radar display (left), parameter controls (center), and optimization results dashboard (right) with score rankings and parameter trend visualizations.

An exhaustive grid search is computationally feasible for the three-variant, three-parameter space addressed here, with typical grids containing a few hundred configurations evaluable in seconds. More sophisticated approaches such as Bayesian optimization or evolutionary algorithms would become necessary if the parameter space were expanded to include additional CFAR variants with extra parameters (e.g., OS-CFAR order statistic k, censoring fractions). The area-weighted coverage score (Eq. 7) provides substantially more information than a binary detected/not-detected metric, since a configuration that detects only a small fragment of a large target receives a low score whereas binary scoring would assign full credit. This distinction is important for targets where shape fidelity affects downstream processing such as target classification or extent estimation.

The tool currently supports three CFAR variants (CA, GO, SO); incorporating OS-CFAR or censored variants would require sorting operations that increase per-pixel cost. Data ingestion is limited to the CSV format produced by the Furuno/RPi5 digitization pipeline, and support for additional radar formats would require parser extensions.

Despite these limitations, the framework addresses several operational needs. Rapidly configurable CFAR detection directly improves on-board detection of fixed reference targets during autonomous mapping, approach monitoring and deconfliction, and search-and-rescue operations in cluttered environments where environmental returns generate excessive false alarms. Because the browser-based architecture requires no software installation, no specialized computing hardware, and no network connectivity during operation, an operator can run CFAR Studio on a laptop alongside the radar feed in the field. This zero-infrastructure deployment model is particularly relevant for ad-hoc missions where pre-installed signal processing toolchains are unavailable and radar parameters must be adapted to an unfamiliar scene on short notice.

VII. Conclusion

We presented CFAR Studio, an open-source, browser-based tool for human-supervised CFAR parameter tuning in heterogeneous radar environments. The system enables an operator to annotate stationary reference objects via

blob tracing, then exhaustively searches over three CFAR variants (CA, GO, SO) and their parameter spaces using an integral-image-accelerated grid search. An area-weighted scoring function evaluates each configuration against the annotated ground truth, penalizing partial detections and rewarding full-shape coverage. Multi-frame temporal stability identifies parameter settings that produce consistent detections across sequential radar sweeps.

Demonstrated on real X-band radar data from a Furuno NXT DRS4D at Lake Murray, South Carolina, the tool evaluates hundreds of configurations in seconds within a standard web browser. The results identify scene-specific parameter trends and variant trade-offs that would be difficult to discover through manual trial-and-error tuning. The fully client-side architecture, requiring no installation or server infrastructure, makes the tool suitable for rapid field deployment across a range of radar sensing applications.

Acknowledgments

This work was supported by the Office of Naval Research (ONR) under contract No. N00014-23-C-1052. The support of the ONR is gratefully acknowledged. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the United States Navy.

References

[1] Farina, A., and Studer, F. A., "A Review of CFAR Detection Techniques in Radar Systems," *Optimised Radar Processors*, IET, 1987. doi:10.1049/PBRA001E_ch18, URL https://digital-library.theiet.org/doi/10.1049/PBRA001E_ch18.

[2] Gandhi, P. P., and Kassam, S. A., "Analysis of CFAR Processors in Nonhomogeneous Background," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 24, No. 4, 1988, pp. 427–445. doi:10.1109/7.7185, URL https://ieeexplore.ieee.org/document/7185.

[3] Finn, H. M., and Johnson, R. S., "Adaptive Detection Mode with Threshold Control as a Function of Spatially Sampled Clutter Level Estimates," *RCA Review*, Vol. 29, 1968, pp. 414–463. URL https://www.semanticscholar.org/paper/Adaptive-detection-mode-with-threshold-control-as-a-Finn/1d775905f818bf867029ba6b80ab9974a5566090.

[4] Richards, M. A., *Fundamentals of Radar Signal Processing*, 3rd ed., McGraw-Hill, 2022. URL https://www.accessengineeringlibrary.com/content/book/9781260468717.

[5] Ward, K. D., Tough, R. J. A., and Watts, S., *Sea Clutter: Scattering, the K Distribution and Radar Performance*, IET, 2006. doi:10.1049/PBRA025E, URL https://digital-library.theiet.org/doi/book/10.1049/pbra025e.

[6] Rosenberg, L., and Bocquet, S., "The Pareto Distribution for High Grazing Angle Sea-Clutter," *Proceedings of the IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, 2013, pp. 4209–4212. doi:10.1109/IGARSS.2013.6723762, URL https://ieeexplore.ieee.org/document/6723762.

[7] U.S. Department of Homeland Security, "Small Boat Intrusion Detection Systems with Radar," Technote, System Assessment and Validation for Emergency Responders (SAVER), 2009. URL https://www.dhs.gov/publication/small-boat-intrusion-detection-systems-radar.

[8] Smith, M. E., and Varshney, P. K., "Intelligent CFAR Processor Based on Data Variability," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 36, No. 3, 2000, pp. 837–847. doi:10.1109/7.869503, URL https://ieeexplore.ieee.org/document/869503.

[9] De Maio, A., Farina, A., and Foglia, G., "Design and Experimental Validation of Knowledge-Based Constant False Alarm Rate Detectors," *IET Radar, Sonar & Navigation*, Vol. 1, No. 4, 2007, pp. 308–316. doi:10.1049/iet-rsn:20060113, URL https://digital-library.theiet.org/doi/10.1049/iet-rsn:20060113.

[10] Humphreys, E., Antoniou, M., Baker, C. J., and Clark, P., "Cognitive CFAR Parameter Optimization for Detection in Sea Clutter," *Proceedings of the International Conference on Radar Systems (RADAR)*, 2022, pp. 148–153. doi:10.1049/icp.2022.2307, URL https://digital-library.theiet.org/doi/10.1049/icp.2022.2307.

[11] Wang, J., and Li, S., "SALA-LSTM: A Novel High-Precision Maritime Radar Target Detection Method Based on Deep Learning," *Scientific Reports*, Vol. 13, 2023, p. 12125. doi:10.1038/s41598-023-39348-3, URL https://www.nature.com/articles/s41598-023-39348-3.

[12] Diskin, T., Beer, Y., Okun, U., and Wiesel, A., "CFARnet: Deep Learning for Target Detection with Constant False Alarm Rate," *Signal Processing*, Vol. 223, 2024, p. 109543. doi:10.1016/j.sigpro.2024.109543, URL https://www.sciencedirect.com/science/article/pii/S0165168424001762.

[13] MathWorks, "Constant False Alarm Rate (CFAR) Detection," Online Documentation, 2025. URL https://www.mathworks.com/help/phased/ug/constant-false-alarm-rate-cfar-detection.html.

[14] Skolnik, M. I., *Radar Handbook*, 3rd ed., McGraw-Hill, 2008. URL https://www.accessengineeringlibrary.com/content/book/9780071485470.

[15] Hansen, V. G., and Ward, H. R., "Detection Performance of the Cell Averaging LOG/CFAR Receiver," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. AES-8, No. 5, 1972, pp. 648–652. doi:10.1109/TAES.1972.309580, URL https://ieeexplore.ieee.org/document/4104055.

[16] Trunk, G. V., "Range Resolution of Targets Using Automatic Detectors," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. AES-14, No. 5, 1978, pp. 750–755. doi:10.1109/TAES.1978.308634, URL https://ieeexplore.ieee.org/document/4102054.

[17] Angelliaume, S., Rosenberg, L., and Ritchie, M., "Modeling the Amplitude Distribution of Radar Sea Clutter," *Remote Sensing*, Vol. 11, No. 3, 2019, p. 319. doi:10.3390/rs11030319, URL https://www.mdpi.com/2072-4292/11/3/319.

[18] Rohman, B. P., and Kurniawan, D., "Neural Network-Based Adaptive Selection CFAR for Radar Target Detection in Various Environments," *International Journal of Intelligent Systems Technologies and Applications*, Vol. 18, No. 4, 2019, pp. 377–390. doi:10.1504/IJISTA.2019.10021691, URL https://www.inderscienceonline.com/doi/abs/10.1504/IJISTA.2019.10021691.

[19] Roldan, I., Palffy, A., Kooij, J. F. P., Gavrila, D. M., Fioranelli, F., and Yarovoy, A., "See Further Than CFAR: A Data-Driven Radar Detector Trained by Lidar," *Proceedings of the IEEE Radar Conference (RadarConf)*, 2024. doi:10.1109/RadarConf2458775.2024.10548165, URL https://arxiv.org/abs/2402.12970.

[20] Wicks, M. C., Baldygo, W. J., Jr., and Brown, R. D., "Expert System Constant False Alarm Rate (CFAR) Processor," U.S. Patent 5,499,030, 1996. URL https://patents.google.com/patent/US5499030.

[21] Crow, F. C., "Summed-Area Tables for Texture Mapping," *Proceedings of the ACM SIGGRAPH*, 1984, pp. 207–212. doi:10.1145/800031.808600, URL https://dl.acm.org/doi/10.1145/800031.808600.