Poseidon Stage-1 Master Plan

The 10/10 Maritime Domain Awareness Solution

Executive Summary

This plan delivers a **production-ready, competition-winning** Stage-1 solution through strategic simplicity, bulletproof execution, and surgical optimization of the three evaluation metrics (AP50: 50%, F1: 30%, RMSE: 20%).

Core Strategy: Progressive Sophistication

Phase 1 (Weeks 1-3): Rock-solid baseline that works perfectly **Phase 2 (Weeks 4-6):** Strategic enhancements targeting specific metrics
Phase 3 (Weeks 7-8): Competition-grade polish and failsafe systems

---

Technical Architecture

1. Detection Pipeline (Targeting AP50 = 50% weight)

Baseline Detector:

- **YOLOv8x** fine-tuned separately on EO and SAR
- Proven architecture, extensive documentation, robust inference
- Pre-trained on COCO → transfer learning advantage

EO Processing Chain:

Raw Sentinel-2 → Cloud masking (SCL band) → Normalization → 640x640 tiles → YOLOv8-EO → NMS

SAR Processing Chain:

Raw Sentinel-1 → Speckle filtering → dB conversion → Normalization → 640x640 tiles → YOLOv8-SAR → NMS

Advanced Enhancement (Phase 2):

- **Ensemble averaging** of EO and SAR predictions (when both available)
- **Test-Time Augmentation (TTA)** - 4 rotations + flips
- **Multi-scale detection** - 3 different input resolutions
- **Confidence calibration** using temperature scaling on validation set

2. Vessel Classification (Missing requirement addressed)

Three-class system:

1. Commercial vessels (cargo, tanker, passenger)
2. Vessels of interest (fishing, military, unknown)
3. Other (recreational, service)

Implementation:

- **Classification head** added to YOLOv8 (detection + classification in one pass)
- Training on vessel type annotations from xView dataset
- **Aspect ratio + size features** as backup classifier for edge cases

3. AIS Correlation (Targeting F1 = 30% weight)

Smart Matching Algorithm:

```
def correlate_ais_detections(detections, ais_data, image_timestamp):
    # Phase 1: Temporal gating (±30 minutes)
    ais_candidates = filter_by_time(ais_data, image_timestamp, window=30)

    # Phase 2: Spatial gating (adaptive radius based on vessel speed)
    matches = []
    for detection in detections:
        for ais_point in ais_candidates:
            distance = haversine(detection.lat_lon, ais_point.lat_lon)
            max_distance = calculate_max_distance(ais_point.speed, time_window=30)

            if distance <= max_distance:
                confidence = calculate_match_confidence(distance, time_diff,
speed_consistency)
                matches.append((detection, ais_point, confidence))

    # Phase 3: Hungarian algorithm for optimal assignment
    return hungarian_assignment(matches)
```

Advanced Enhancement (Phase 2):

- **Kalman filter tracking** for vessel state estimation
- **Speed/course consistency** scoring in match confidence
- **Multi-hypothesis tracking** for ambiguous cases

4. Path Interpolation (Targeting RMSE = 20% weight)

Physics-Based Interpolation:

```
def interpolate_vessel_path(sparse_ais_points):
    # Phase 1: Linear interpolation with speed constraints
    base_path = linear_interpolation_with_physics(sparse_ais_points)

    # Phase 2: Smooth with cubic spline (enforces realistic accelerations)
    smooth_path = cubic_spline_smooth(base_path, tension=0.1)

    # Phase 3: Land avoidance using coastline buffer
    final_path = avoid_land_intersections(smooth_path, coastline_buffer=1km)

    return final_path
```

Advanced Enhancement (Phase 2):

- **Constant Turn Rate and Speed (CTRS) model** for realistic vessel dynamics
- **Weather/current compensation** using historical patterns (offline lookup tables)
- **Uncertainty quantification** to weight interpolation confidence

5. Land Masking & Geo-Processing

Robust Land Masking:

- **Natural Earth coastline** → 100m buffered polygons
- **Pre-computed mask tiles** for all possible Sentinel scenes
- **Conservative masking** - exclude detections within 200m of coast

Coordinate Handling:

- **Geodesic calculations** for all distance measurements
- **CRS transformation pipeline** with error checking
- **Bounding box validation** - ensure no invalid geometries

Data Strategy

Training Data Preparation

Datasets (in order of priority):

1. **xView vessel detection** - 190k vessel annotations (primary)
2. **Sentinel vessel detection** - Zenodo dataset (domain-specific)
3. **HRSC2016** - ship classification labels
4. **Marine Cadastre AIS** - real-world correlation examples

Data Augmentation Pipeline:

- **Geometric:** rotation (±45°), flip, scale (0.8-1.2x)
- **Photometric:** brightness (±20%), contrast (±15%), gamma (0.8-1.2)
- **SAR-specific:** speckle simulation, multiplicative noise
- **Atmospheric:** synthetic cloud/haze overlay for EO

Validation Strategy:

- **Stratified 5-fold CV** on training data
- **Geographic holdout** - reserve specific regions for validation
- **Temporal holdout** - reserve recent dates for validation
- **Mock test simulation** - identical format to competition evaluation

---

Implementation Timeline

Week 1-2: Foundation (Baseline that works)

- [ ] Data loaders for all formats (SAFE, TIFF, CSV)
- [ ] YOLOv8 training pipeline for EO and SAR
- [ ] Basic land masking implementation
- [ ] Export format compliance (GeoJSON + Shapefile)
- [ ] Local evaluation metrics (AP50, F1, RMSE)
- [ ] **Milestone: End-to-end pipeline runs on sample data**

Week 3-4: Core Algorithms

- [ ] AIS correlation algorithm with Hungarian assignment
- [ ] Path interpolation with physics constraints
- [ ] Vessel classification integration
- [ ] Docker containerization
- [ ] **Milestone: All three evaluation metrics implemented**

Week 5-6: Optimization & Enhancement

- [ ] Model ensemble and TTA implementation
- [ ] Hyperparameter optimization using Optuna
- [ ] Advanced interpolation with CTRS model
- [ ] Confidence calibration and uncertainty quantification
- [ ] **Milestone: Performance optimization complete**

Week 7: Competition Preparation

- [ ] Mock test participation and analysis
- [ ] Submission format automation
- [ ] Error handling and edge case coverage
- [ ] CPU fallback mode for resource constraints
- [ ] **Milestone: Competition-ready system**

Week 8: Polish & Failsafe

- [ ] Final mock test and tuning
- [ ] Documentation and code cleanup
- [ ] Stress testing on IIT Delhi hardware specs
- [ ] Multiple backup strategies
- [ ] **Milestone: Bulletproof competition submission**

Risk Mitigation

Technical Risks

Risk	Impact	Mitigation
Model overfitting	High	5-fold CV + early stopping + dropout
Runtime timeout	High	CPU fallback + model pruning + batch optimization
Format compliance	Critical	Automated validation + extensive testing
Memory overflow	Medium	Streaming inference + garbage collection
Coordinate errors	High	Comprehensive CRS testing + validation

Competition Risks

Risk	Impact	Mitigation
Late rule changes	Medium	Modular design + rapid adaptation capability
Hardware differences	Medium	Docker + deterministic seeds + multiple test environments
Data corruption	Low	Robust file handling + checksums
Network issues	Low	Offline operation + local caching

Performance Engineering

Optimization Targets

Speed: Complete processing within 2-hour IIT Delhi demo slot

- **GPU path:** 5 min/image (A100 optimized)
- **CPU path:** 15 min/image (guaranteed completion)

Memory: Fit within 512GB RAM constraint

- **Streaming inference:** Process 1 image at a time
- **Memory pools:** Pre-allocate and reuse tensor memory
- **Garbage collection:** Explicit cleanup between images

Accuracy: Target metrics for top-5 finish

- **AP50:** >0.75 (aim for 0.80+)
- **F1 Score:** >0.85 (aim for 0.90+)
- **RMSE:** <500m (aim for <300m)

Hardware Utilization

```
# GPU Configuration (A100 80GB)
batch_size = 16  # Maximizes GPU utilization
precision = "mixed"  # FP16 + FP32 for speed + accuracy
compile = True  # PyTorch 2.0 compilation

# CPU Configuration (64 cores)
num_workers = 32  # Parallel data loading
```

```
thread_count = 64  # NumPy/OpenCV threading
```

Quality Assurance

Testing Strategy

1. **Unit tests** for all core functions (>90% coverage)
2. **Integration tests** for pipeline components
3. **Format validation** for all output files
4. **Regression tests** against known good outputs
5. **Performance benchmarks** on reference hardware

Code Quality

- **Type hints** throughout (mypy compliance)
- **Docstring coverage** for all public functions
- **Linting** with black, isort, flake8
- **Git hooks** for pre-commit validation
- **Configuration management** with Hydra

Documentation

- **API documentation** (Sphinx)
- **Deployment guide** for IIT Delhi demo
- **Troubleshooting runbook** for common issues
- **Performance tuning guide** for different hardware

Competitive Edge

What Makes This Plan Unbeatable

1. **Metric-First Design:** Every component optimized for AP50/F1/RMSE
2. **Bulletproof Reliability:** Extensive testing + multiple fallback modes
3. **Production Quality:** Enterprise-grade code quality + documentation
4. **Strategic Simplicity:** Proven techniques over experimental approaches
5. **Perfect Compliance:** Automated format validation + extensive testing

Secret Weapons

1. **Adaptive Matching Radius:** AIS correlation radius adjusts based on vessel speed
2. **Physics-Constrained Interpolation:** Impossible vessel movements rejected
3. **Confidence Calibration:** Accurate uncertainty estimates for better thresholding
4. **Geographic Stratification:** Training/validation splits respect spatial distribution
5. **Multi-Scale Ensemble:** Different model scales combined intelligently

---

Success Metrics

Stage-1 Goals

- **Primary:** Advance to Stage-2 (top 15-20 teams)
- **Target:** Top-5 finish in Stage-1 evaluation
- **Stretch:** #1 overall score in Stage-1

Technical KPIs

- **AP50:** >0.80 (target: 0.85)
- **F1 Score:** >0.90 (target: 0.95)
- **RMSE:** <300m (target: <200m)
- **Processing Speed:** <2 hours for full evaluation set
- **Memory Usage:** <400GB peak (safe margin under 512GB)

Quality Metrics

- **Zero format errors** in all submissions
- **100% successful** docker deployments
- **<5 minute** setup time at IIT Delhi demo
- **Zero crashes** during evaluation period

This plan combines **battle-tested techniques** with **surgical optimizations** to dominate Stage-1. Every component has been selected for maximum reliability while targeting the specific evaluation metrics. The progressive sophistication approach ensures we have a working system early, with systematic improvements that compound our competitive advantage.

The key insight: This competition rewards **perfect execution** over **novel research**. Our plan delivers both.