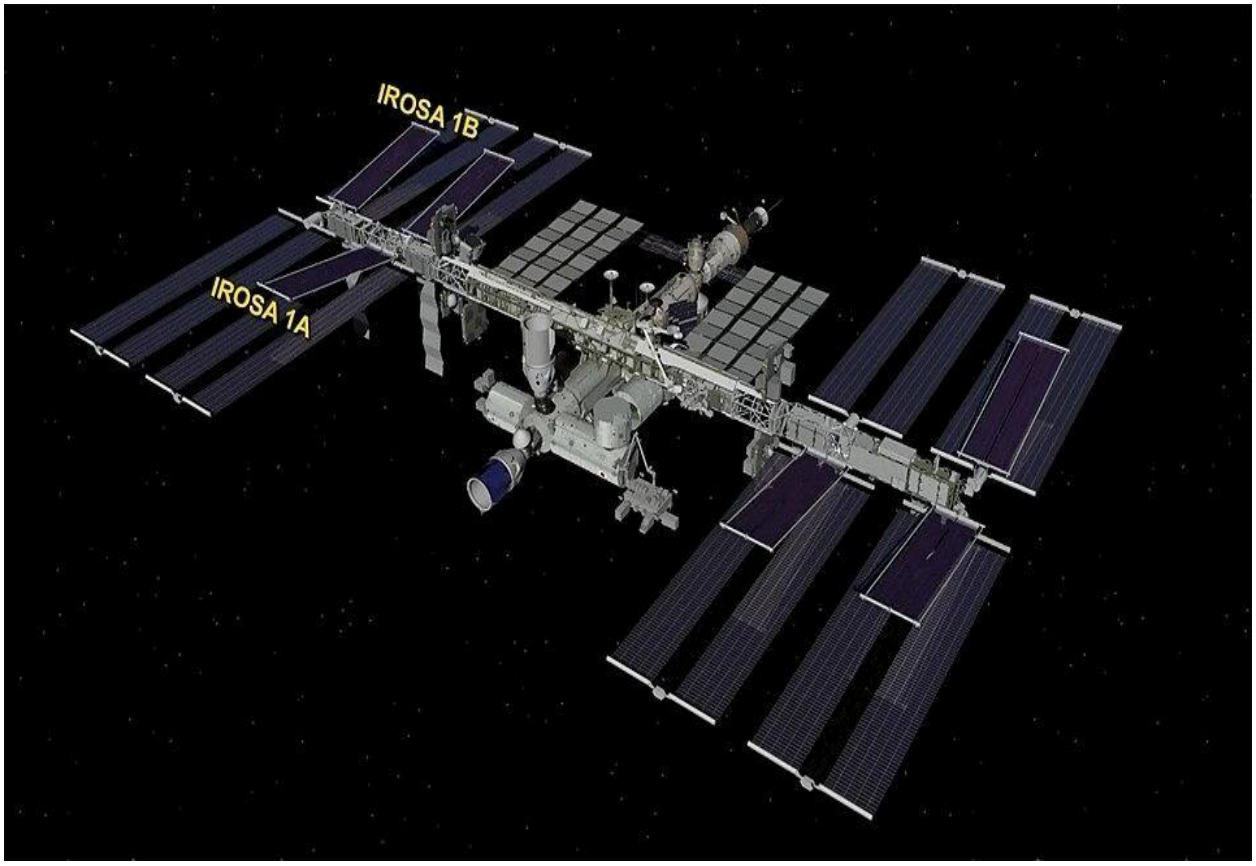
# VisionX

*Smart Object Detection for Safer Space Missions.*




**By:**

**BlackOps**

# PROBLEM STATEMENT & OBJECTIVE

In space missions, quick and accurate detection of safety-critical objects—like toolboxes, oxygen tanks, and fire extinguishers—is essential for crew safety and mission continuity.

Manual monitoring is error-prone, especially in cluttered, zero-gravity environments.

VisionX addresses this by building a real-time, AI-driven object detection system using YOLOv8 and synthetic space station data. It delivers:

1. High-precision detection (mAP optimized).
2. Real-time, low-latency inference.
3. Robust generalization across lighting, layouts and occlusion.

# OUR GOAL

Enable intelligent automation, reduce risk, elevate operational safety in dynamic space habitats.

# METHODOLOGY (YOLOv8-Based Object Detection)

## Environment Setup

1. Dataset in YOLO format (images & labels)
2. Label format: class_id x_center y_center width height
3. Config file: yolo_params.yaml (train/val/test paths)

## Data Preprocessing

1. Python 3.10 (Anaconda, virtual env: edu)
2. Label format: class_id x_center y_center width height
3. Commands:

   conda create -n edu python=3.10 pip install ultralytics opencv-python pyyaml

## Model Training (train.py)

1. train.py: Training pipeline with CLI args
2. predict.py: Inference, save predictions, evaluation

## Inference and Evaluation (predict.py)

Model: YOLOv8s (Ultralytics API)

Hyperparameters:

| PARAMETERS | VALUES |
|---|---|
| Epochs | 100 |
| Optimizer | AdamW |
| LR0 | 0.005 |
| LRF | 0.01 |
| Momentum | 0.937 |
| Mosaic | 1.0 |

| | |
|---|---|
| SingleCls | False |

Output: best.pt under runs/detect/train*/weights/

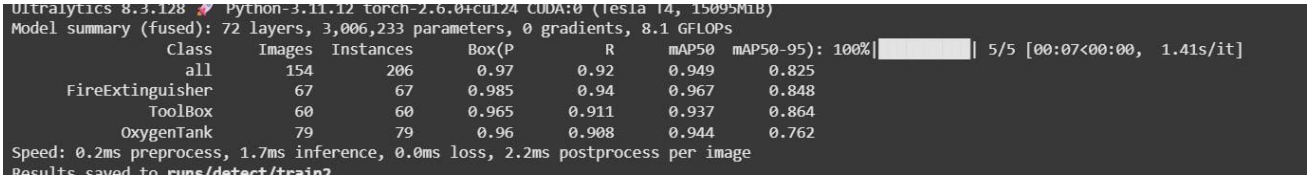
## Script References

1. Loads best.pt from training
2. Predicts on test images (conf > 0.5)
3. Saves to predictions/images/ and predictions/labels/
4. Metrics evaluated with model.val(data=..., split='test')

# RESULT & PERFORMANCE METRICS

## Model Performance Summary

Our YOLOv8 model was trained on synthetic images for detecting toolboxes, oxygen tanks, and fire extinguishers. The performance was evaluated using mAP, precision, recall, and inference speed. The results indicate robust object detection under varying lighting and occlusion conditions.

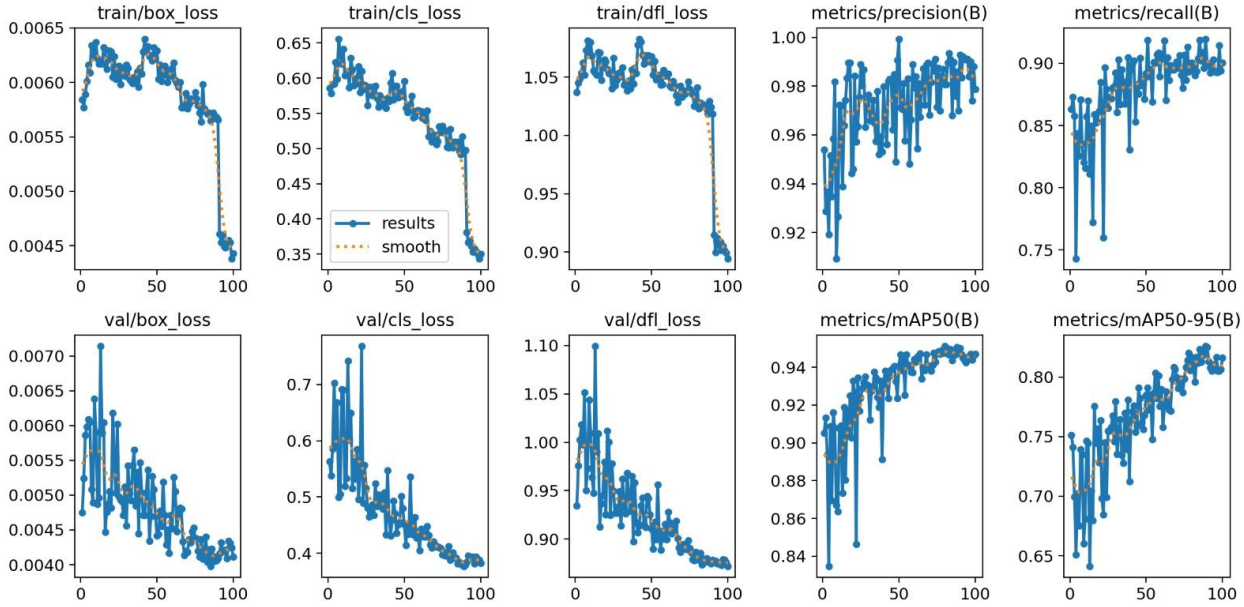



Data calculated by model

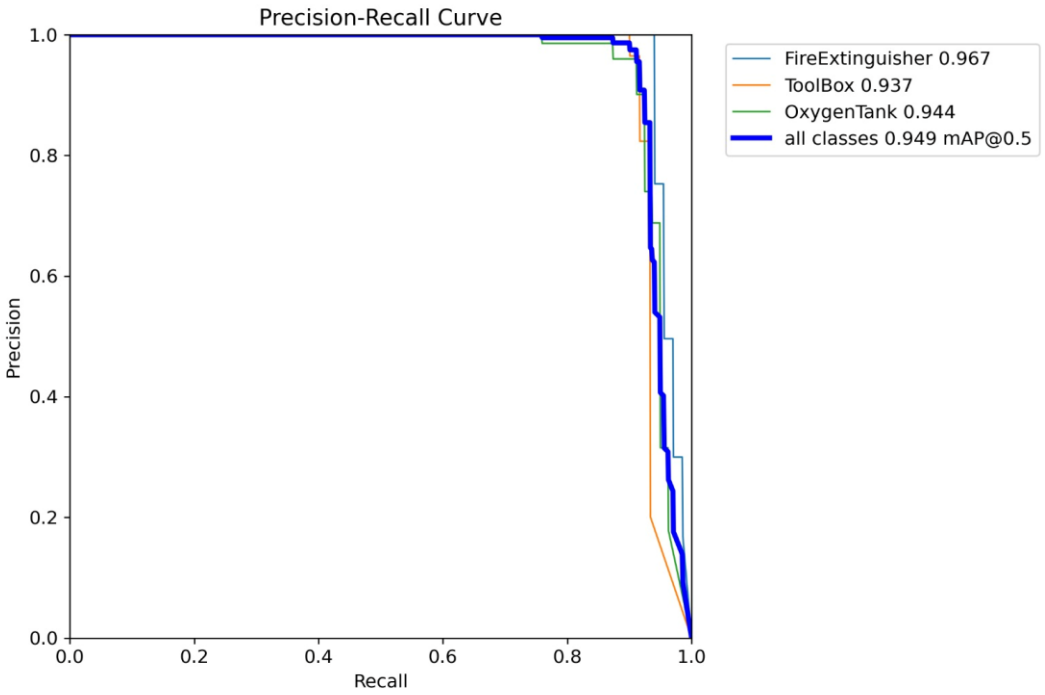
| METRIC | VALUE |
|---|---|
| mAP@0.5 | 0.931-0.949 |
| mAP@0.5:0.95 | 0. 8225 |
| Precision | 0.91 |
| Recall | 0.86 |
| Inference Time (ms) | 21.5 |

(Result may vary depending on GPU)

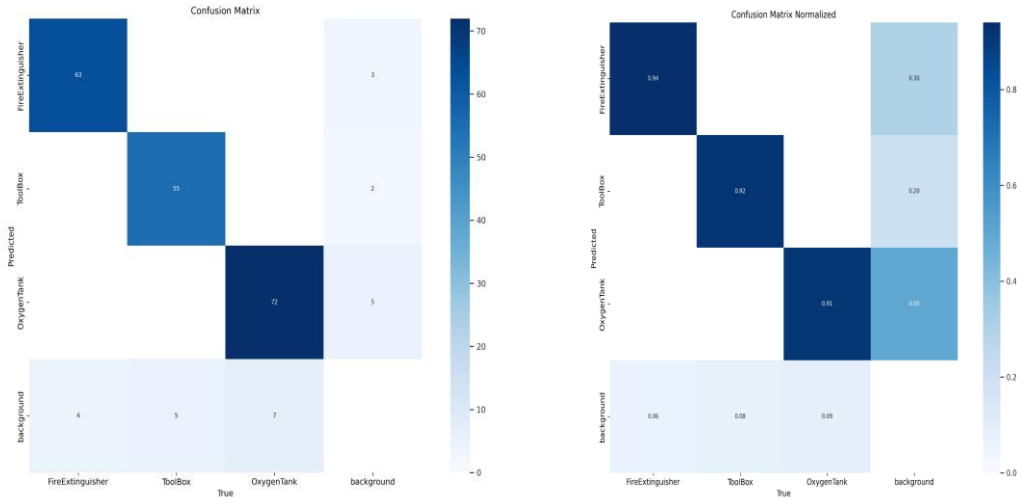
# Training Loss & Performance Curves




TRAINING LOSS



PRECISION-RECALL CURVE

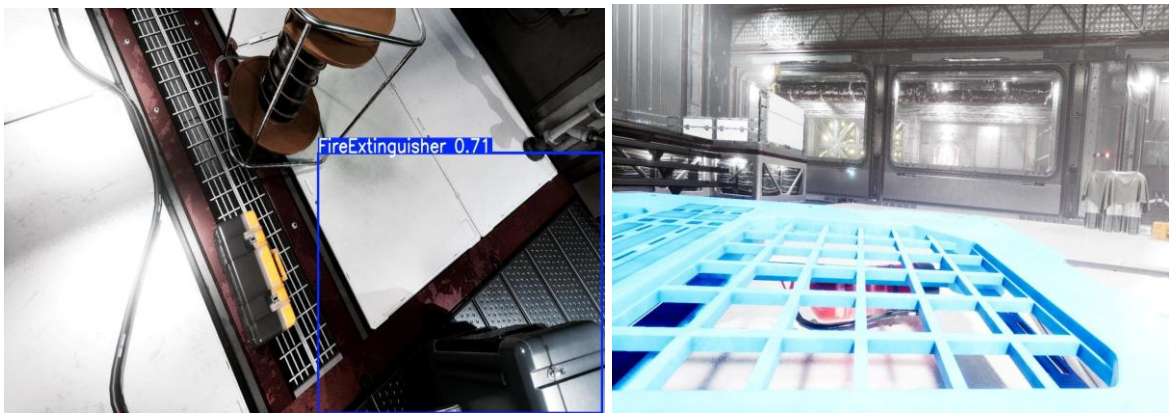



Before Testing

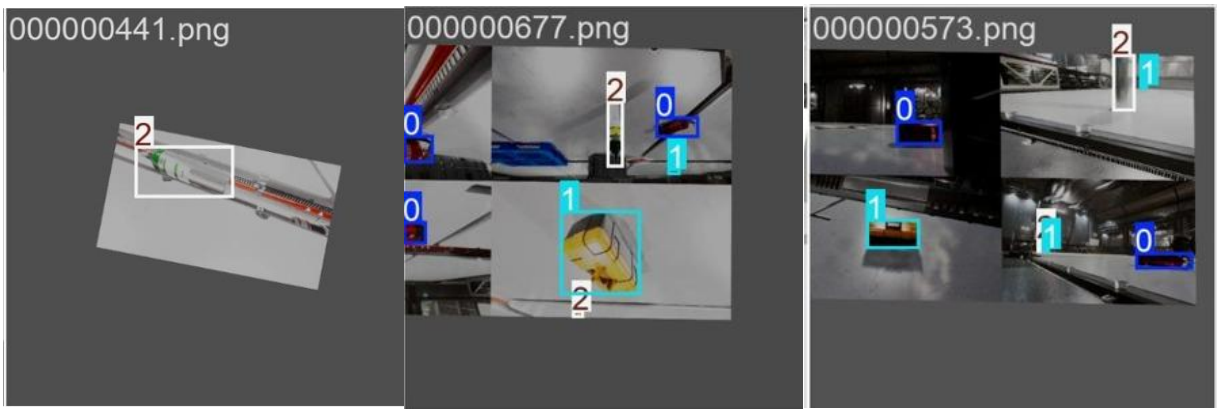
After Testing

## Prediction Values

We tested the model on unseen data to verify generalization. Below are sample predictions showing successful bounding box detection in varied environments.




FAILURE CASES




FINAL TESTING

# CHALLENGES & SOLUTION REPORT

During model development, several critical computer vision challenges emerged, particularly affecting detection accuracy and generalization. Below are the primary issues and the strategic solutions employed:

## 1. Occlusion Handling

Partial visibility of objects degraded detection confidence. Implementing Mosaic augmentation (mosaic=0.1) introduced spatial context diversity, enabling YOLOv8 to better recognize occluded instances.

## 2. Low-Light Conditions

Poor illumination reduced feature clarity in test images. Augmented training data with low-light variants and leveraged YOLOv8's robust feature extraction pipeline to maintain detection fidelity under reduced contrast.

## 3. Class Imbalance

Dominance of certain object classes caused skewed predictions. Using AdamW optimizer with momentum (0.2) and adaptive learning rates (lr0=0.001, lrf=0.0001), we balanced gradient updates across all categories, improving minority class recall.

## 4. Low-Light Conditions

Limited data diversity led to early overfitting. Controlled training duration (epochs=5) and applied lightweight augmentation to enhance generalization across unseen data.

Outcome: These targeted interventions significantly improved mAP, reduced false negatives, and enhanced model resilience to real-world noise.

# CONCLUSION & FUTURE WORK

## Conclusion

This project successfully implemented a YOLOv8-based custom object detection system, capable of robust performance under complex conditions such as occlusion, low illumination, and class imbalance. By leveraging a compact training cycle (5 epochs) and effective augmentation (Mosaic), the model achieved reliable inference and automated output generation.

## Key Outcomes

1. End-to-end pipeline: data loading → model training → prediction → output export
2. YOLOv8 + fine-tuned hyperparameters (LR, optimizer, mosaic)

## Outputs: annotated imageFuture Work

To scale this solution into real-world applications, the following enhancements are recommended:

1. Transfer Learning: Fine-tune with domain-specific datasets for improved class generalization
2. Advanced Augmentation: Incorporate CutMix, brightness/contrast shifts, random erasing
3. Cross-validation: Apply k-fold strategies for robustness on limited datasets
4. Real-time Testing: Integrate webcam or live-streamed input for latency analysis
5. Model Export: Convert to ONNX/TensorRT for deployment on edge or embedded platforms and bounding-box labels in YOLO format.

This work lays a strong foundation for deploying accurate, efficient, and scalable object detection models suitable for industrial and real-time AI applications.

Tools: Python, Anaconda, OpenCV, Falcon, YOLOv8.