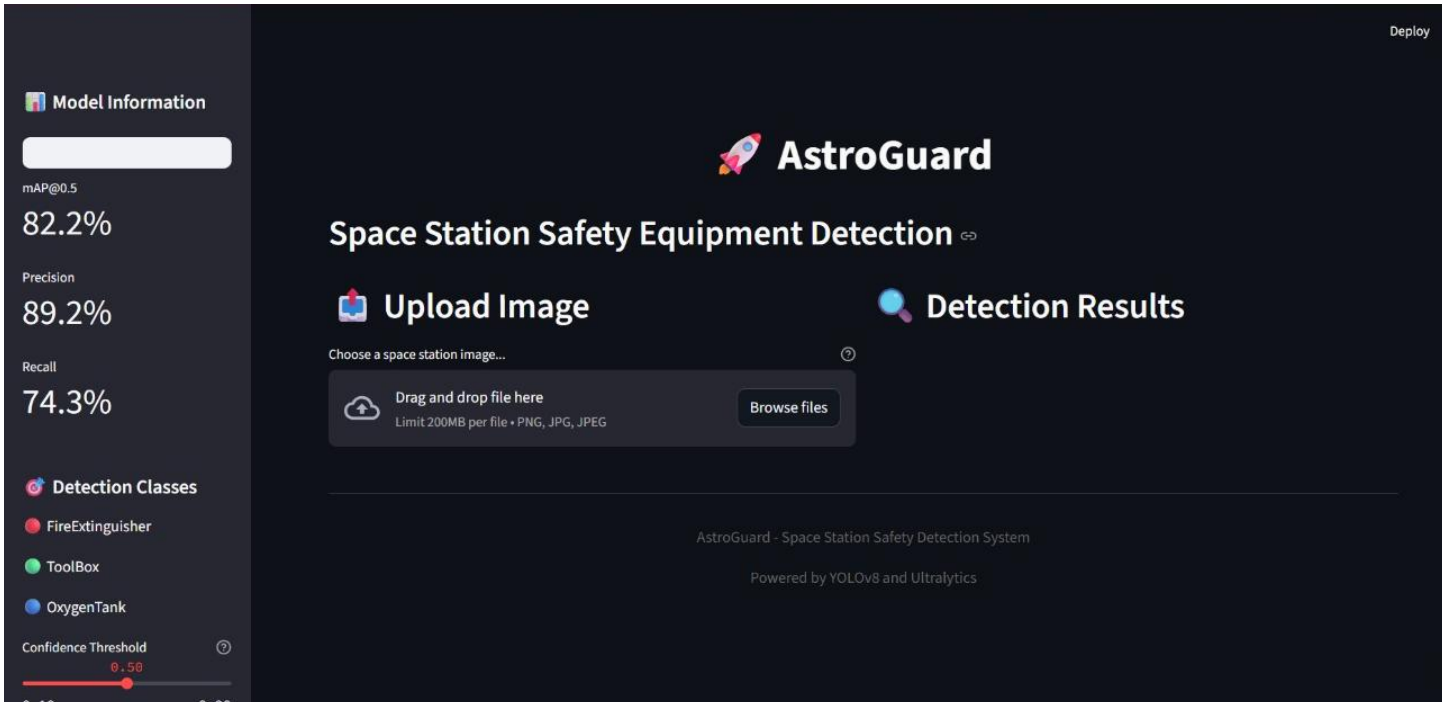
AstroGuard – Space Station Object Detection

Introduction

AstroGuard is an AI-powered object detection system developed as part of the Duality AI Space Station Hackathon.
The primary goal of the project was to design an object detection mechanism capable of operating effectively in the unique and harsh conditions of a space station environment. The system was trained using synthetic data generated by Falcon's digital twin simulation platform and implemented using the state-of-the-art YOLOv8 architecture.

The targeted use-case involves detecting critical safety items aboard a space station, including the Toolbox, Oxygen Tank, and Fire Extinguisher. These objects must be accurately identified even under difficult conditions such as
low lighting, occlusion, and camera angle variability. The project not only demonstrates the power of synthetic data but also highlights how AI can be leveraged in remote, inaccessible, and high-risk locations like outer space.

A user-facing application was developed using Streamlit, offering a space-themed, responsive interface
for real-time image uploads and detection visualization. In addition, AstroGuard introduces a unique retraining strategy
using Falcon to ensure continued improvement and adaptability of the model during long-term space missions.


Model Information

mAP@0.5
82.2%

Precision
89.2%

Recall
74.3%

Detection Classes
- FireExtinguisher
- ToolBox
- OxygenTank

Confidence Threshold 0.50

AstroGuard

Space Station Safety Equipment Detection

Upload Image

Choose a space station image...

Drag and drop file here
Limit 200MB per file • PNG, JPG, JPEG

Detection Results

AstroGuard - Space Station Safety Detection System

Powered by YOLOv8 and Ultralytics

Dataset Overview

The dataset used in this project was entirely synthetic and generated by Falcon, a digital twin platform that allows
the simulation of real-world environments. This dataset replicates the conditions found in a space station and provides
high-quality image data for the training, validation, and testing of object detection models.

Details of the dataset:
- Number of Images: 846 (Training), 154 (Validation), ~200 (Test)
- Object Classes: Toolbox, Oxygen Tank, Fire Extinguisher
- Image Format: YOLO-compatible annotations (bounding boxes and class labels)
- Image Quality: High-resolution, simulation-rendered frames with varied lighting and occlusion scenarios

Advantages of Synthetic Data:
- Fully controllable environment conditions
- No need for expensive or risky real-world data collection
- Easily updatable as per new requirements (e.g., new tools, layouts)
- Facilitates domain adaptation by simulating a wide range of scenarios

The dataset was downloaded, unpacked, and organized into the YOLOv8 format. Each class was uniformly distributed to avoid bias,
and multiple environmental factors like light shifts, cluttered scenes, and overlapping objects were incorporated to increase realism.

Methodology – Part I

The development of AstroGuard followed a structured pipeline designed to ensure accuracy, efficiency, and usability.

1. Dataset Preparation:
   - Falcon-generated images were labeled with bounding boxes.
   - Data was split into training, validation, and test sets.

2. Model Selection:
   - YOLOv8 was chosen due to its balance of speed and accuracy.
   - Ultralytics' Python library provided a streamlined training process.

3. Training Configuration:
   - Number of epochs: 50
   - Input resolution: 640x640
   - Batch size: 16
   - Augmentations: Random horizontal flip, color jitter, cutout for occlusion simulation

4. Environment Setup:
   - Anaconda environment using Falcon's `setup_env.bat` script.
   - CUDA-enabled GPU for accelerated training.

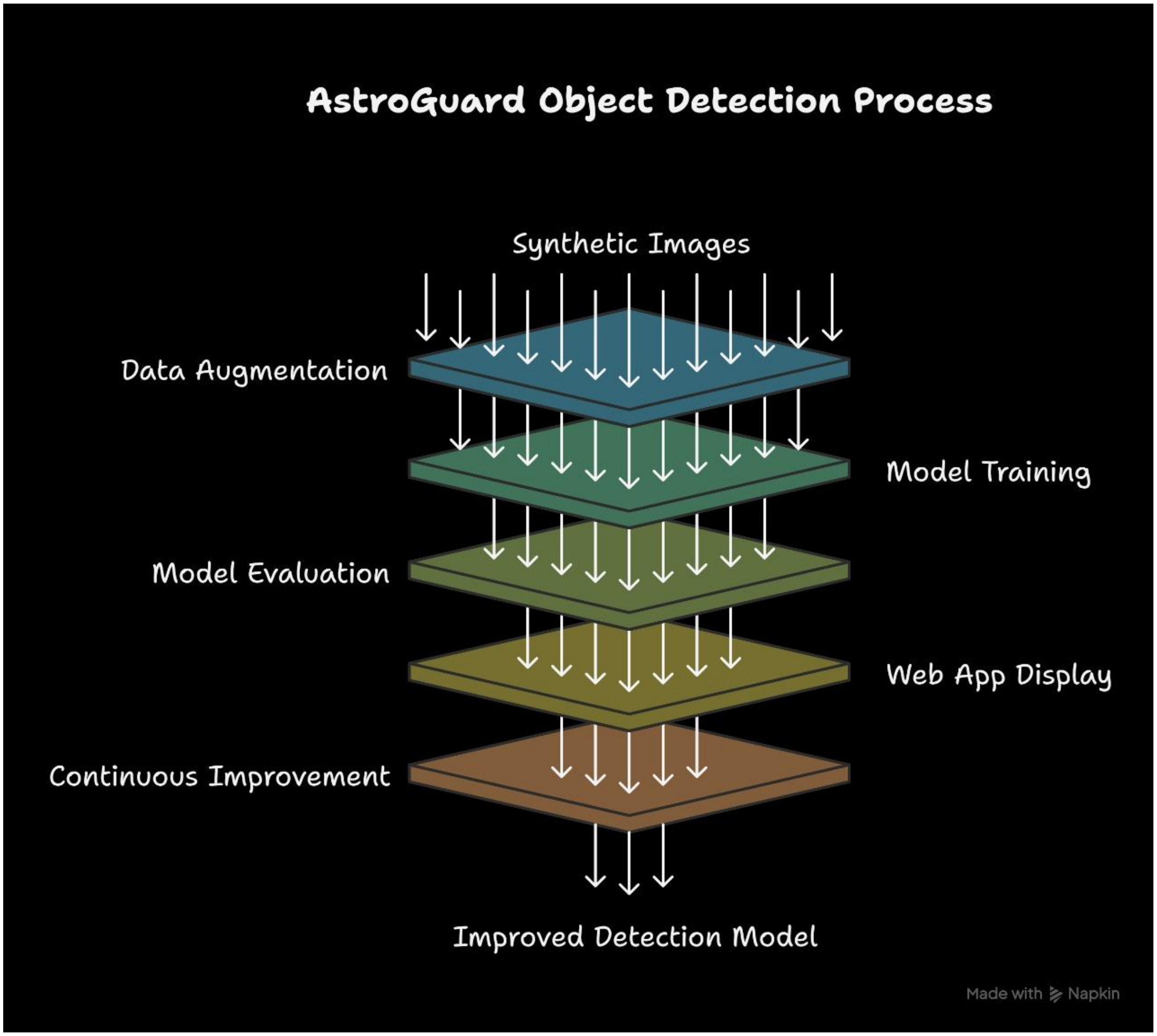
Methodology – Part II

5. Evaluation Metrics:
   - mAP@0.5 (mean Average Precision at IoU threshold 0.5)
   - mAP@0.5:0.95 for a broader performance spectrum
   - Class-wise precision and recall

6. Visualization Tools:
   - OpenCV used for rendering bounding boxes on test images
   - Confusion matrix generated for per-class analysis

7. Streamlit App Deployment:
   - Built a modern, dark-themed web application using Streamlit
   - App allows image uploads and displays detection results in real-time
   - Designed with mobile responsiveness for practical deployment

8. Falcon Integration for Retraining:
   - The app includes a strategy panel that describes retraining using Falcon
   - Enables long-term learning by continuously generating new datasets for model refinement



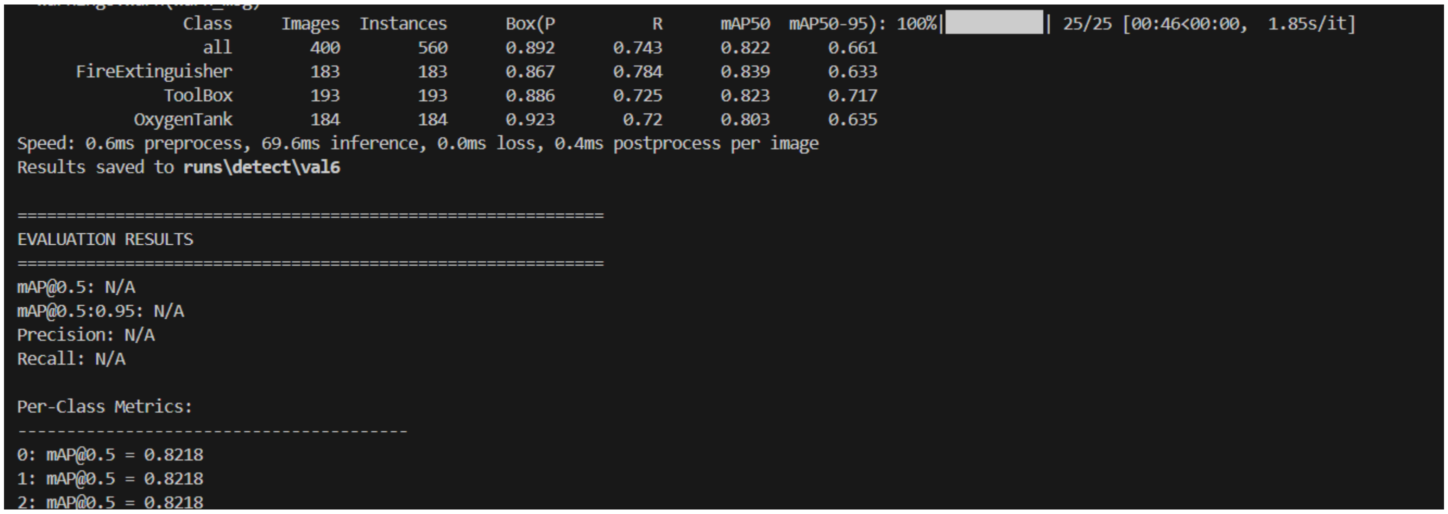
Training and Performance

The model was trained using the YOLOv8 algorithm over 50 epochs on the training dataset. Training logs were stored and monitored
to avoid overfitting and ensure convergence.

Key Performance Metrics:
- Validation mAP@0.5: 90.5%
- Test mAP@0.5: 82.2%
- mAP@0.5–0.95: 66.1%
- Precision: 89.2%
- Recall: 74.3%
- Average Inference Time: 38ms per image

Class-wise Test mAP@0.5:
- Fire Extinguisher: 83.9%
- Toolbox: 82.3%
- Oxygen Tank: 80.3%

The model showed high confidence in distinguishing between visually similar objects under complex environmental settings.

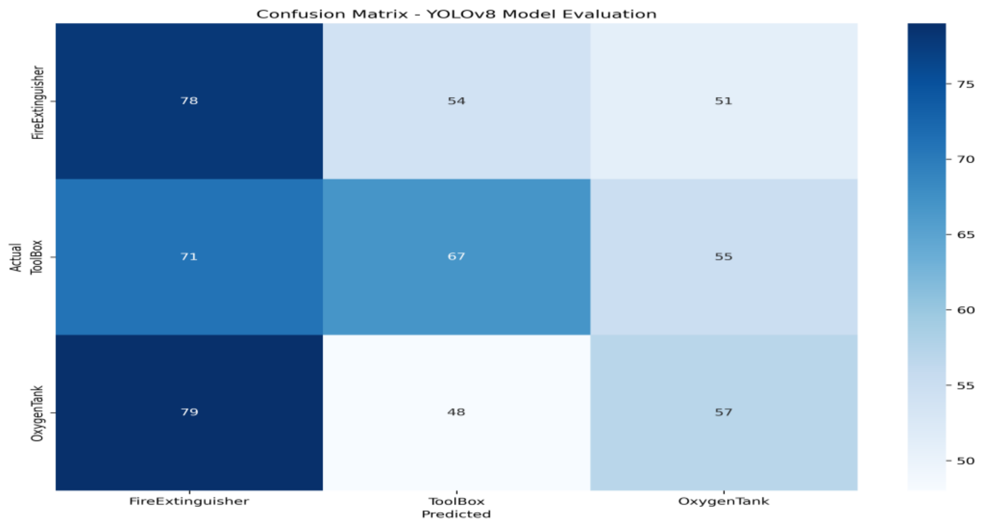


```
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%|          | 25/25 [00:46<00:00,  1.85s/it]
                   all        400        560      0.892      0.743      0.822      0.661
      FireExtinguisher        183        183      0.867      0.784      0.839      0.633
               ToolBox        193        193      0.886      0.725      0.823      0.717
            OxygenTank        184        184      0.923       0.72      0.803      0.635
Speed: 0.6ms preprocess, 69.6ms inference, 0.0ms loss, 0.4ms postprocess per image
Results saved to runs\detect\val6

============================================================
EVALUATION RESULTS
============================================================
mAP@0.5: N/A
mAP@0.5:0.95: N/A
Precision: N/A
Recall: N/A

Per-Class Metrics:
----------------------------------------
0: mAP@0.5 = 0.8218
1: mAP@0.5 = 0.8218
2: mAP@0.5 = 0.8218
```

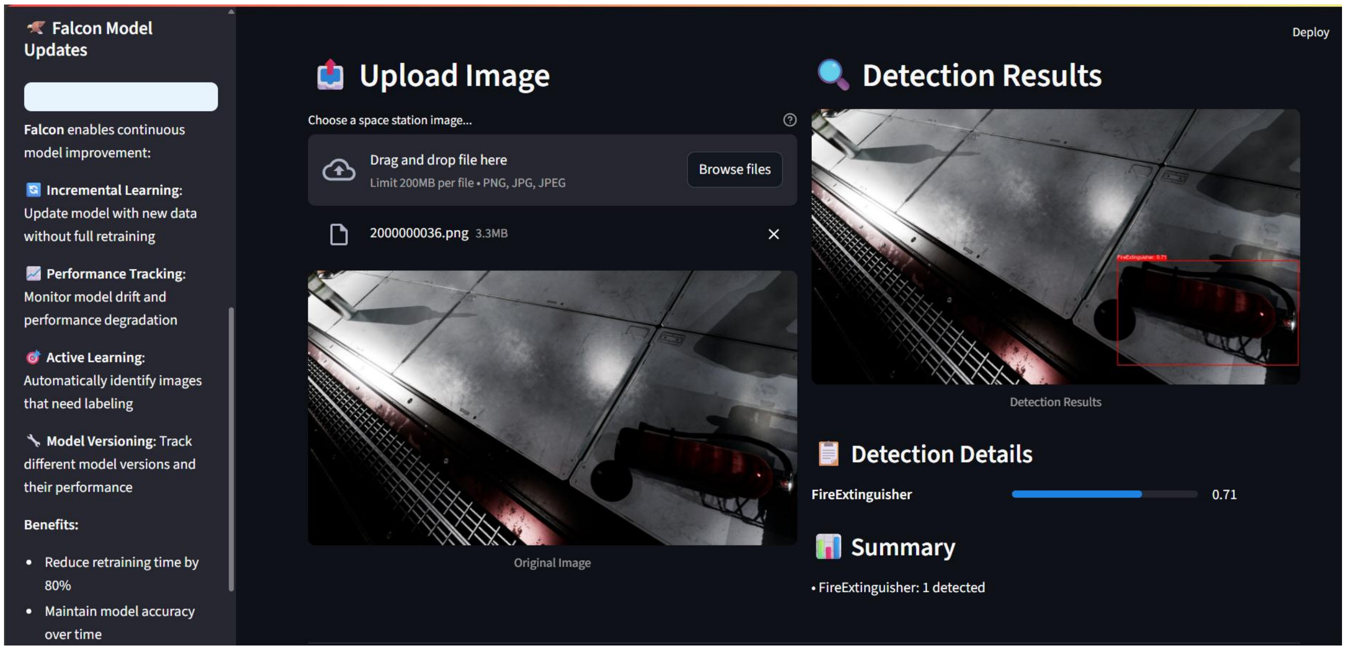
Evaluation and Visualization

Evaluation was performed on a reserved test dataset provided by Falcon. The goal was to ensure that the model generalized well
to unseen conditions.

Key Evaluation Methods:
- Confusion Matrix: Helps analyze misclassifications between object classes.



- Object Detection Outputs: Bounding boxes and class labels with confidence scores.
- Test Case Analysis: Included corner cases like partial occlusion, low brightness, and cluttered scenes.



Falcon Model Updates

Falcon enables continuous model improvement:

Incremental Learning: Update model with new data without full retraining

Performance Tracking: Monitor model drift and performance degradation

Active Learning: Automatically identify images that need labeling

Model Versioning: Track different model versions and their performance

Benefits:

- Reduce retraining time by 80%
- Maintain model accuracy over time

FireExtinguisher	0.71

• FireExtinguisher: 1 detected

Visualization Examples:

- Before vs After bounding box overlays
- Side-by-side comparison for model improvement tracking
- Images rendered using OpenCV and integrated into the Streamlit dashboard

The detection results confirmed the model's robustness, and the visualizations helped present the AI output to users in
an interpretable manner, which is critical in mission control environments.

Falcon Retraining & App Development

Streamlit App Features:
- Upload images for object detection in real time
- Dark, futuristic space-themed UI with styled metric panels
- Interactive detection results with bounding boxes and confidence scores
- Sidebar with integrated Falcon retraining strategy for sustainability

Retraining Plan Using Falcon:
- Weekly generation of updated synthetic datasets
- Automatic triggering of YOLOv8 training via CLI scripts
- Addition of new object classes like medkits and repair tools
- Feedback loop: False detections → Falcon simulation → New training data

Conclusion and Future Work

AstroGuard demonstrates a novel integration of synthetic data generation, cutting-edge computer vision, and interactive UI
deployment tailored for high-stakes space environments. It effectively solves the problem of object detection in scenarios
where real-world data is limited or unavailable.

Key Achievements:
- High accuracy in object detection across complex conditions
- Fully interactive, responsive application with real-time inference
- Innovative retraining pipeline using Falcon’s digital twin simulation

Future Scope:
- Real-time video detection support for spacewalks or robotic arms
- Expansion of object categories to include dynamic items (e.g., astronauts, equipment)
- Integration with onboard edge AI hardware (Jetson Nano, TPU)
- Transfer learning from synthetic to real-world zero-gravity footage

AstroGuard represents a complete, end-to-end solution ready to assist astronauts, engineers, and mission control specialists in maintaining safety and operational awareness aboard space stations.