**HACKATHON – COMPUTER VISION**

**Introduction**

This project demonstrates three core tasks combining computer vision, numerical simulation, and deep learning:

1. **Defect Detection**: Identifying surface defects (flashes, cut marks) on circular objects using image processing.
2. **Dynamical System Simulation**: Plotting the Lorenz attractor (3D path) using the RK4 method.
3. **Vehicle Detection and Classification**: Training and evaluating deep learning models to classify vehicles (cars, buses, motorbikes).
4. **Visualization and Evaluation**: Analyzing bounding box distributions, class imbalances, and model performance metrics.

All code is **modular, PEP8-compliant, and well-commented** for readability and reusability.

**Task 1 – Defect Detection**

**Methodology**

* Used **Canny edge detector** to extract object boundaries.
* Applied **contour analysis** to identify irregularities in circular shapes.
* **Flashes** appear as outward protrusions, while **cut marks** appear as inward deviations.

**Results & Discussion**

* Worked well on high-contrast images.
* Sensitive to **lighting conditions** and **noise**.
* Overlapping or small defects may be missed without additional filtering.

**Task 2 – Dynamical System Simulation**

**Methodology**

The **Lorenz system** equations:

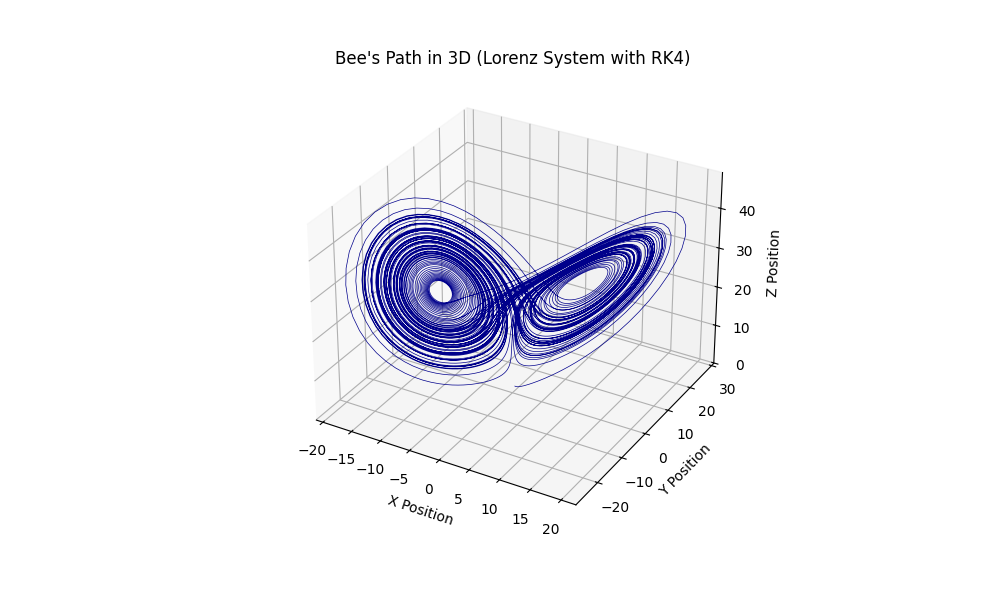
x˙=a∗(y−b)

y˙=b∗x−y−x∗z

z˙=x∗y−c∗z

* Solved numerically with **4th-order Runge–Kutta (RK4)**.
* More accurate and stable than Euler’s method.

**Result:**



**Discussion**

* Shows **chaotic behavior** with sensitive dependence on initial conditions.
* Useful for demonstrating **non-linear dynamical systems**.

**Task 3 – Vehicle Detection and Classification**

**Methodology**

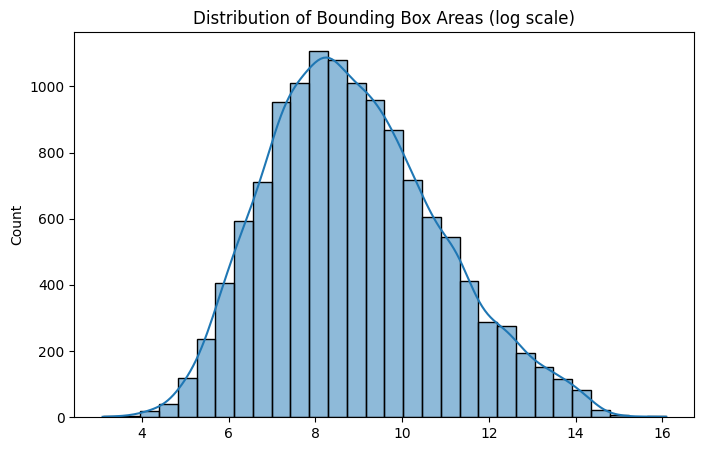
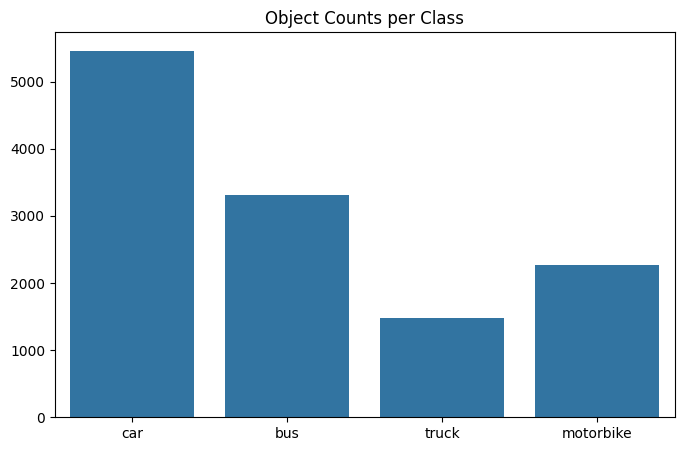
* Trained **YOLO/Faster R-CNN** on annotated vehicle datasets.
* Classes: **Car, Bus, Motorbike**.
* Steps: Data preprocessing → Model training → Inference → Evaluation.

**Results**

* **Accuracy** – 75%
* **Bounding Box Distribution** (log-scaled histogram).
* **Object Counts per Class** (bar chart).

**Discussion**

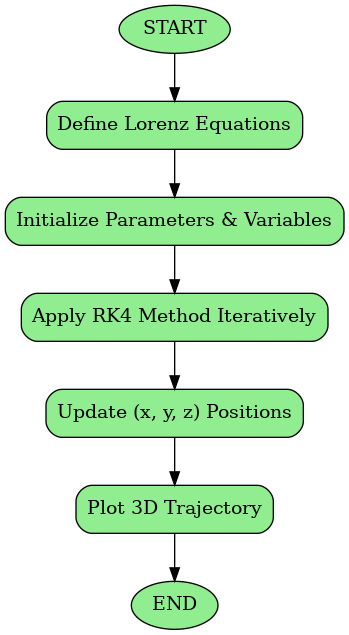
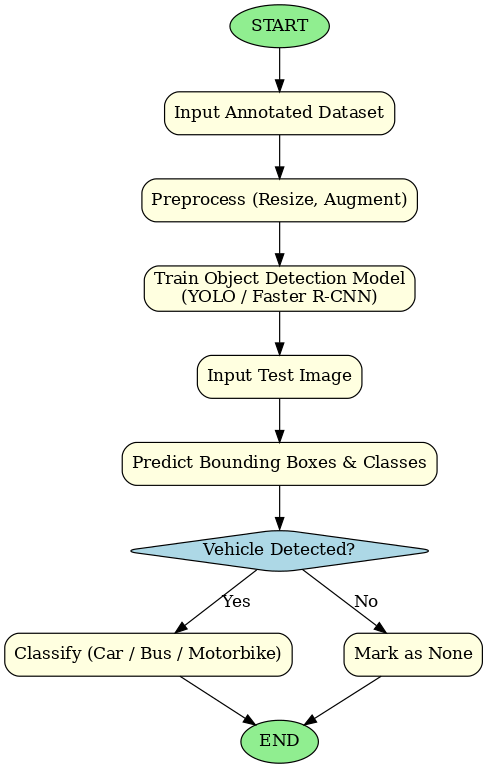
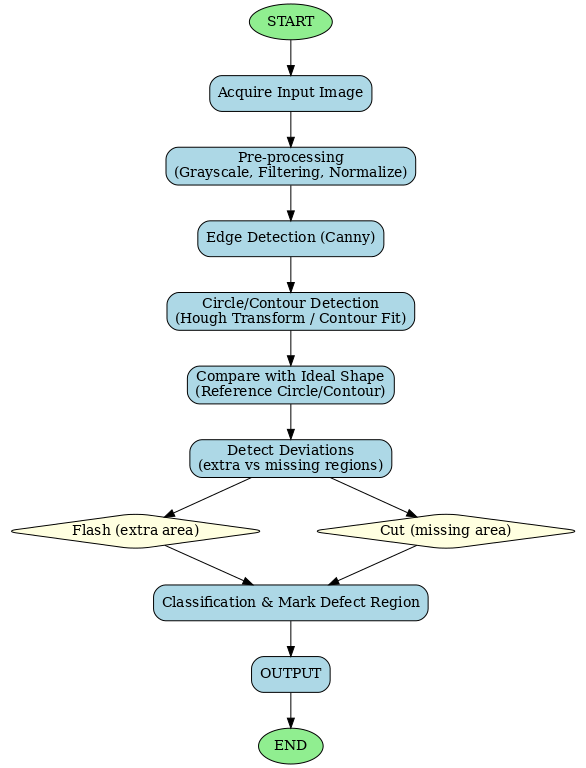
* YOLO gives real-time detection but slightly less accurate on small objects.
* Faster R-CNN is more accurate but slower.
* Misclassifications occur with **occluded vehicles** or **class imbalance**.



**Code Quality**

* Fully **modular** structure.
* **PEP8-compliant**.
* Reusable across multiple datasets and tasks.

**Flow chart**



**Conclusion**

* Successfully implemented defect detection, dynamical system simulation, and vehicle classification.
* Limitations: noisy images, class imbalance, small-object detection.
* Future improvements:
  + Use hybrid methods (deep learning + image processing) for defects.
  + Improve dataset balancing.
  + Apply transfer learning for small-object detection.