Short Report: Object Detection and Change Detection in Simulated Environment

Solution Overview

This solution involves developing software to detect and log changes in a specific environment using a simulated dataset. The environment is represented by a series of images recorded at different time steps. Have used YOLOv8, a state-of-the-art object detection model, to identify and classify objects in the images. The solution includes the following steps:

1. **Data Loading** Load images and associated metadata from the provided `base.npz` and `test.npz` files.
2. **Object Detection** Use YOLOv8 to detect and classify objects in each image.
3. **Change Detection** Compare detected objects between base and test images, identifying appeared, disappeared, and moved objects.
4. **Visualization** Display base and test images side-by-side, highlighting detected changes and logging these changes in a separate display window.

Rationale behind Design Choices

1. **YOLOv8 for Object Detection**
   YOLOv8 was chosen for its balance between accuracy and speed, which is essential for real-time applications. It is well-suited for detecting multiple objects in various scenarios, providing both bounding boxes and class labels.
2. **Background Subtraction**
   Initially included to highlight moving objects but later replaced by YOLOv8 for more robust and comprehensive object detection.
3. **Proximity Threshold for Change Detection**
   Introduced to handle slight variations in object positions between frames, ensuring the same objects are not misclassified as appeared or disappeared due to minor positional shifts.
4. **Visualization and Logging**
   Visual feedback is crucial for validating the detection and change detection processes. The side-by-side display helps in immediate visual comparison, while the log display provides a detailed account of changes, aiding in debugging and performance evaluation.

Potential Improvements with Extended Time

Given six more months, the prototype can be significantly enhanced through the following ideas:

1. **Advanced Tracking Algorithms**
   Implement and integrate advanced tracking algorithms to maintain object identities across frames, improving the accuracy of detecting moved objects.
2. **Improved Proximity Measures**
   Develop more sophisticated methods for comparing objects, such as using Intersection over Union thresholds, to further reduce false positives and negatives in change detection.
3. **Data Augmentation and Training**
   Collect additional data and perform data augmentation to retrain the YOLOv8 model, tailoring it specifically to the target environment for better detection performance.
4. **User Interface Enhancements**
   Develop a comprehensive user interface for better interaction and visualization, including features like dynamic log updates, interactive frame navigation, and detailed change reports.