MTDD: Mosaic Tile Damage Detection

Rishith Arra, Anthony Stan, Zachary Rieth

Background

- Vandalism of public property is a persistent issue in urban environments
- Degrades public aesthetics, result in high repair costs, and reduced community morale
- Traditional surveillance often fails to provide timely detection or actionable intelligence

Mission

- Our focus: Heart & Soul Park, a park that opened in 2021 and has been subjected to vandalism
- Our goal is revitalization by consistent maintenance
- Adhering to Broken Window Theory (Wilson and Kelling, 1982) to prevent snowball effects
- Push damage reports to the police RMS systems

Approach 1 - LPIPS & OpenCV

Outline

OpenCV → Created a differencing algorithm with OpenCV to detect any changes found between two images

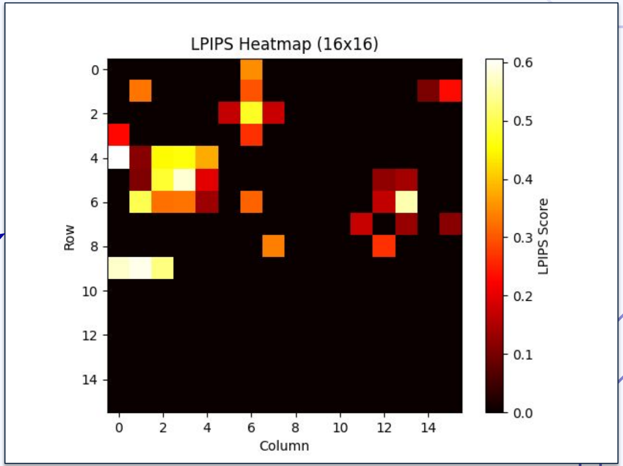
↓

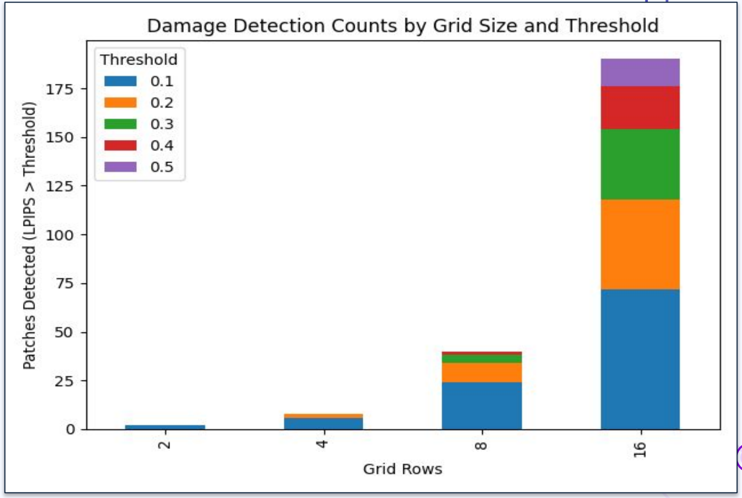
LPIPS → Used LPIPS to score the significance of differences, triggering the differencing algorithm if a threshold LPIPS value is met

↓

RF-DETR → Implemented RF-DETR model to detect people passing in front of the mosaics, delaying snapshots until no one is in the way

Results





This model excelled at identifying differences between test images including current mosaic photos and edited photos with fixed damages at a threshold around 20%.

Discussion

Improving the model:

- Problems related to brightness normalization
- Camera quality
- Easily picks up small changes → can be solved with parameter optimization
- Implementation and further testing in park with Raspberry Pi and Pi-cam

Approach 2 : RF-DETR, YOLO v11, & OpenCV

RF-DETR

RF-DETR is used to detect humans in the scene. If humans are identified, we will continue on to our next steps.

Privacy concerns were a worry, which is why all humans detected on the feed were anonymized through Gaussian blurring or cutouts.

Object Detection

Tier 1
YOLO v11

Tier 1 damage detection involves using a fine-tuned YOLO v11 model designed to identify large sections of damage. It is then passed to Tier 2 for minute detail analysis.

Tier 2
YOLO v11

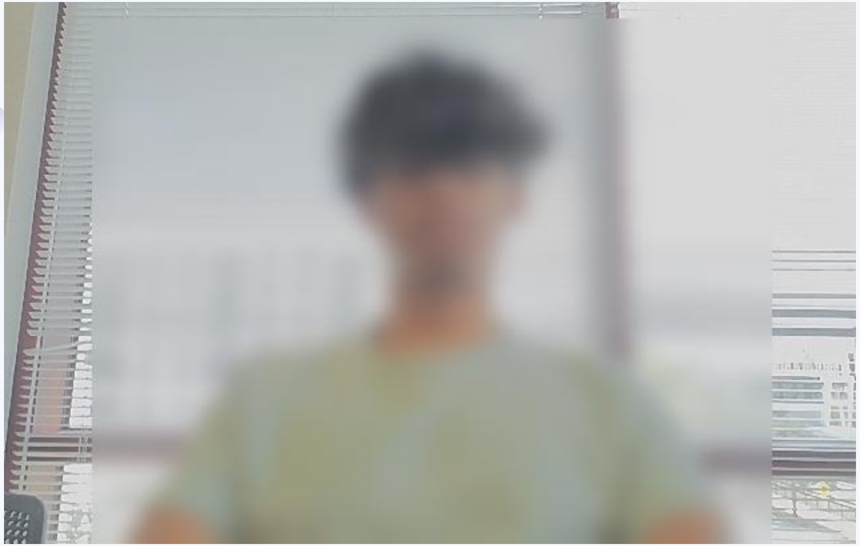
Tier 2 damage detection involves using a different fine-tuned YOLO v11 model designed to identify small sections of damage. The total results are output in a damage report.

Models Used

RF-DETR: Lightweight and high-performance compared to larger models like Microsoft Florence 2, making it helpful for constant video analysis, as it minimizes computational resources with a high detection rate. Because RF-DETR comes pre trained with weights that allow it to easily detect people, RF-DETR did not need to be fine-tuned, allowing us to save training resources.

YOLO: YOLO was chosen for damage detection because it was able to be quickly fine tuned (saving training resources) whilst also boasting a high accuracy rate. While other models like PaliGemma may have also been suitable for the task, training these other models took too much computational resources, pushing us to use YOLO over them.

Pipeline Results - Anonymization

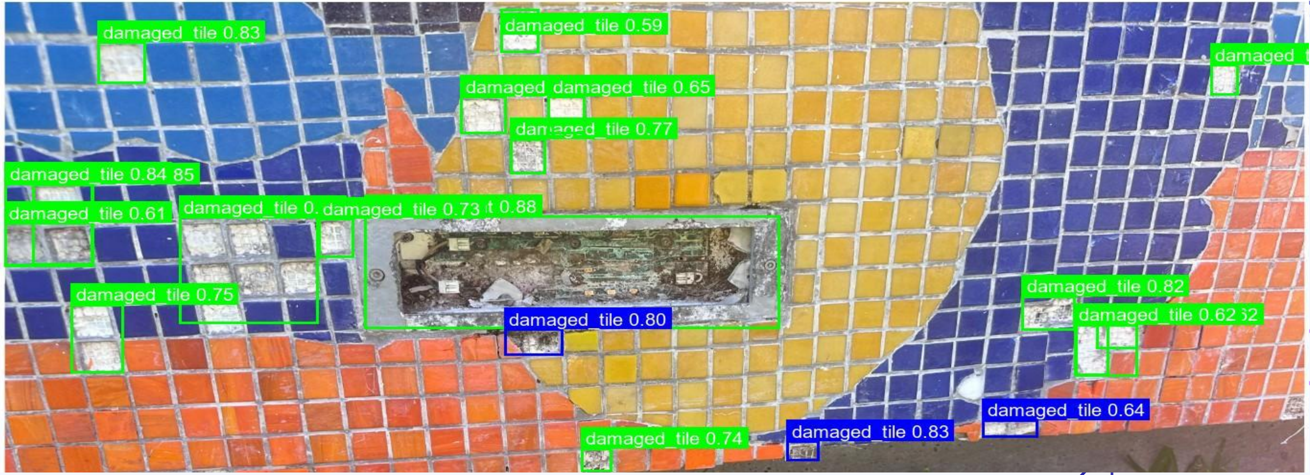


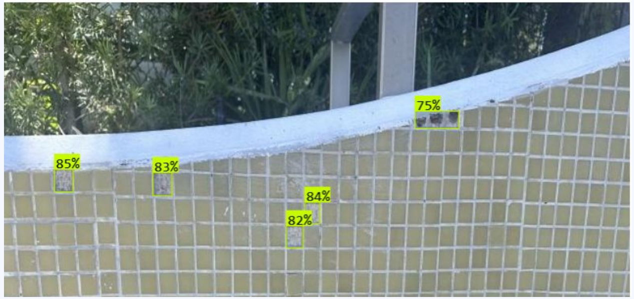
Anonymization through targeted Gaussian Blur

Anonymization through targeted human cutouts

Pipeline Results - Images

Images from field testing





Pipeline Results - Report

```
Human entered scene at 15:39:42
Human left scene at 15:40:03

============================================================
DAMAGE ANALYSIS REPORT
============================================================
Human entered scene: 2025-08-04 15:39:42.737762
Human left scene: 2025-08-04 15:40:03.202237
Scene duration: 0:00:21
============================================================

0: 480x640 (no detections), 178.4ms
Speed: 4.6ms preprocess, 178.4ms inference, 0.7ms postprocess per image at shape (1, 3, 480, 640)

0: 480x640 (no detections), 144.0ms
Speed: 2.4ms preprocess, 144.0ms inference, 0.6ms postprocess per image at shape (1, 3, 480, 640)
Damage analysis image saved to: damage_report_20250804_154003.jpg
Frame size: (640, 480)
Total damage detections: 0
No damage detected in scene
============================================================
```

Discussion

Improving the pipeline:

➔ Longer training times with more data would lead to better model damage detections

➔ Designing a system to push severe damage reports (maybe 20+ damages?) to police RMS systems, potentially using webhooks

➔ Improving damage report readability through using a Generative AI model

  - ◆ This was a big area of struggle, as high end models were severely computationally expensive, cost money per use with API's, and may misconstrue data (hallucinate)

➔ Better hardware would likely improve results

➔ More field testing would have identified any flaws

Future Work

- → Using LIDAR systems
- → Pilot testing with Raspberry Pi 5 in the park

Any Questions?

Thank you!