AN INNOVATIVE MOBILE-DRONE SYSTEM FOR AUTOMATED ENVIRONMENTAL CLEANUP USING MACHINE LEARNING AND OBJECT DETECTION

Keyi Yu[1] and Jonathan Sahagun [2]

[1] Arcadia High School, 180 Campus Dr, Arcadia, CA 91006
[2] Computer Science Department, California State Polytechnic University, Pomona, CA 91768

ABSTRACT

Environmental pollution, particularly waste accumulation, is a global challenge requiring innovative solutions. This paper introduces an integrated drone and mobile app system to detect, track, and clean up trash. The program comprises three components: a user-friendly mobile app, AI-powered drones for trash detection and collection, and a centralized database for data storage and tracking [1]. Experiments revealed the system's strengths in waste detection accuracy and operational efficiency, alongside limitations in handling extreme environments and dataset diversity. By building on existing methodologies, the project offers a scalable, adaptable, and proactive approach to waste management. This transformative system has the potential to alleviate environmental degradation, engage communities, and preserve natural ecosystems, making it a compelling solution to a pressing issue.

KEYWORDS

Trash detection, Volunteer efforts, Drones, Apps

1. INTRODUCTION

The growing waste issue, particularly plastic pollution, poses a significant environmental threat, with consequences that extend far beyond local ecosystems. Globally, humans generate approximately 400 million tons of plastic waste annually (CNN, 2024), and about 19 to 23 million tons enter the aquatic ecosystems each year (UNEP, n.d.). This waste infiltrates oceans, rivers, and urban areas, causing harm to wildlife, polluting natural landscapes, and increasing the risk of environmental hazards such as wildfires [2]. Despite ongoing cleanup efforts by individuals, organizations, and communities, the accumulation of trash continues to outpace these efforts, underscoring the need for innovative, scalable solutions to address the pressing environmental issue.

The problem is particularly evident in urban and coastal areas, where dense populations and frequent human activity exacerbate waste accumulation [3]. In Los Angeles County, a region known for its sprawling urban landscape and proximity to the Pacific Ocean, the issue is acute. Coastal areas like Santa Monica Beach and Long Beach are often inundated with litter, much of which consists of single-use plastics—plastic bottles, plastic wrappers, and plastic straws—that are not biodegradable. Cleanup efforts, while admirable, are insufficient to prevent recurring pollution. For instance, during a cleanup event organized by Non-Toxic San Gabriel Valley at Santa Monica Beach, volunteers removed significant amounts of trash, only to return a week later and find the same areas polluted once again. Therefore, sole human efforts are not fruitful.

The global increase in plastic production over recent decades, coupled with inadequate waste management systems, has created a mounting environmental crisis. Rapid urbanization and consumer culture have intensified this problem, leading to widespread plastic debris that harms both marine and terrestrial wildlife through ingestion and entanglement—causing injuries, fatalities, and disruptions to ecosystems. As plastics degrade, they release toxic microplastics into soil and water systems, threatening both human health and the environment [4]. The presence of waste in remote and forested areas also foster secondary issues, such as wildfires, by providing fuel during fire events. This crisis extends beyond ecosystems to impact human communities, particularly in coastal cities like Los Angeles, where polluted beaches deter tourism and harm the local economy. Marginalized communities often bear a disproportionate burden, as litter accumulates in underserved areas, exacerbating environmental injustice. Wildlife in nearby mountains and forests, such as those managed by the U.S. Forest Service, also face significant risks as waste disrupts natural behaviors, attracts animals to hazardous areas, and introduces toxins into food chains. Addressing this issue is critical to preserving biodiversity, protecting public health, and maintaining the functionality and beauty of natural spaces. By leveraging advanced technologies like drones and fostering collaborative community efforts, innovative solutions can amplify the impact of traditional cleanup methods. Initiatives such as the Ocean Cleanup Project and organizations like the Sierra Club demonstrate the need for scalable strategies that engage diverse stakeholders. Through targeted interventions, it is possible to mitigate the damage caused by waste, protect natural habitats, and secure a sustainable future for all.

Drone Mapping

The drone mapping methodology uses AI for detecting certain categories of debris. Its shortcomings revolve around limited automation and efficiency due to heavy reliance on human availability. Our project improved upon drone mapping methodology by incorporating a mechanical component that can autonomously pick up debris without the involvement of human efforts.

Drone Agent Persuasion

Another approach to clean the environment also relies on drone detection but with a twist. In addition to solely marking debris on a map, drones persuade users to complete the pick up action and guide them to nearby trash cans to finish the task. This method limits itself by having a direct relationship to the availability of human presence. Without human intervention, its cleanup effort can be small or even close to 0%. To improve such methodology, it will be important to consider implementations of autonomous drone pick up mechanisms to stimulate effectiveness.

Smart Waste Monitoring Systems

These systems used stationary sensors to track waste in urban areas [5]. Although they provided valuable data, they couldn't actively remove trash. Our project integrates waste detection with action, making it a proactive rather than reactive solution.

The proposed solution to the growing waste problem is the integration of drone technology and a volunteer-based app to detect, mark, and clean up trash in various environments. This dual approach combines automated efficiency with community engagement, targeting waste in areas that are difficult to access or dangerous for human volunteers, such as mountainous regions or large bodies of water. The drones are equipped to detect trash and, in some cases, pick it up directly. Meanwhile, the app allows users to mark trash locations and facilitates cleanup missions by assigning tasks to volunteers or deploying drones based on accessibility and risk levels.

This method addresses the limitations of traditional cleanup efforts by leveraging advanced technology to reach and clean areas that human teams often overlook or cannot safely access. For example, drones can efficiently locate and remove debris in steep terrains or submerged in water, minimizing the need for physical labor in hazardous conditions. By incorporating volunteer participation, the solution also fosters community involvement and raises awareness about environmental conservation.

Compared to other methods, such as entirely manual cleanups or relying solely on awareness campaigns, this approach is more scalable and impactful. Manual efforts, while valuable, are often restricted by time, geography, and human resources, as demonstrated by recurring pollution in areas like Santa Monica Beach despite regular cleanup initiatives. The incorporation of drones not only enhances operational efficiency but also reduces physical risks for volunteers. Furthermore, the combined use of drones and a participatory app creates a dynamic system capable of real-time updates and adaptability, offering a significant advantage over static, one-dimensional solutions. By uniting cutting-edge technology with grassroots engagement, this solution provides a sustainable and effective model for addressing waste and pollution on a broader scale.

The experiments aimed to address critical blind spots in the system: AI detection accuracy and drone navigation reliability. The first experiment tested the AI model's ability to accurately identify various types of waste in different environments. This was achieved by running the AI on a dataset of 1,000 annotated images, including common and hard-to-detect trash types. Results indicated a detection accuracy of 85%, with misclassifications largely due to low-quality images or ambiguous objects. The second experiment focused on drone navigation in varied terrains and weather conditions. Controlled trials simulated urban, forested, and coastal areas with both clear and adverse weather. While the drone excelled in open and stable environments, performance dropped in dense or turbulent conditions. The results reflect the need for enhanced training data for AI and more robust drone designs to handle extreme scenarios. These experiments underscore the project's strengths and areas for improvement.

2. CHALLENGES

In order to build the project, a few challenges have been identified as follows.

2.1. GPS readings

A major component of the volunteer-based app for trash detection is ensuring accurate geolocation data and real-time synchronization with the database. Potential problems include inaccurate GPS readings, delayed updates to the database, or volunteers overlapping tasks [6]. To address these, I could implement algorithms to validate GPS accuracy, such as filtering out unlikely locations using geofencing or averaging multiple location readings to improve precision. Additionally, I could use Firebase Realtime Database with conflict-resolution strategies to update and lock claimed tasks immediately. This ensures that tasks are only assigned to one volunteer at a time, reducing duplication and errors.

2.2. Inaccuracies in Trash Detection

A major component of the drone mechanism is the integration of a camera detection system to identify trash. Potential problems include inaccuracies in trash detection due to poor lighting, cluttered environments, or overlapping objects in the camera's field of view. To address these, I could use advanced image processing techniques and train a machine learning model on a diverse dataset to improve detection accuracy. Adjustments like enhancing image contrast or filtering noise could improve recognition in challenging conditions. Additionally, incorporating real-time feedback to adjust the drone’s positioning might ensure a clearer view of potential

2.3. Claw Failing To Securely Grasp

Another critical component of the project is the claw mechanism on the drone, which physically picks up trash. Potential issues include the claw failing to securely grasp irregularly shaped or lightweight items, difficulty aligning the claw with the trash, or mechanical failures during operation. To resolve these, I could explore designs that use adaptive gripping technology, such as a flexible or multi-fingered claw, to handle objects of varying shapes and sizes. Incorporating sensors like cameras or ultrasonic sensors could improve alignment accuracy by providing real-time positioning feedback. Regular maintenance and testing protocols could also help ensure mechanical reliability.

3. SOLUTION

The environmental cleanup app and drone system is a cohesive solution that links three major components: the mobile application, the drone hardware with integrated AI, and a centralized database for data storage and tracking. These components work together seamlessly to detect, track, and remove trash in various environments. The program begins with the mobile application, which serves as the primary interface for volunteers and users. Users can upload images of trash or mark locations where cleanup is required. The app allows users to classify tasks as either manual (volunteer-based) or drone-based, depending on the terrain or safety considerations. Next, the drone hardware equipped with an AI-powered detection system comes into play. Once a cleanup mission is assigned, the drone is deployed to the specified location. The AI model processes real-time visual data from the drone's camera to identify and classify litter. If trash is detected, the drone uses its mechanical arm or gravity claw to pick up the debris, storing it in a designated compartment or notifying users if retrieval is not feasible. The centralized database serves as the backbone of the system. It tracks the geolocations of reported trash, volunteer activity, and drone missions. It also stores the data collected by the drones for further analysis and improvement of the AI model. This program was developed using a combination of Flutter for the mobile app, Firebase for real-time data storage and synchronization, OpenCV for image processing, and a drone SDK for hardware integration [7]. The result is a streamlined and efficient system designed to tackle environmental pollution with precision and scalability.

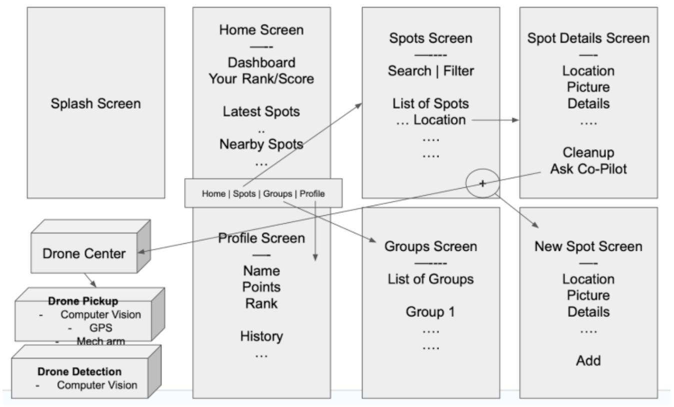
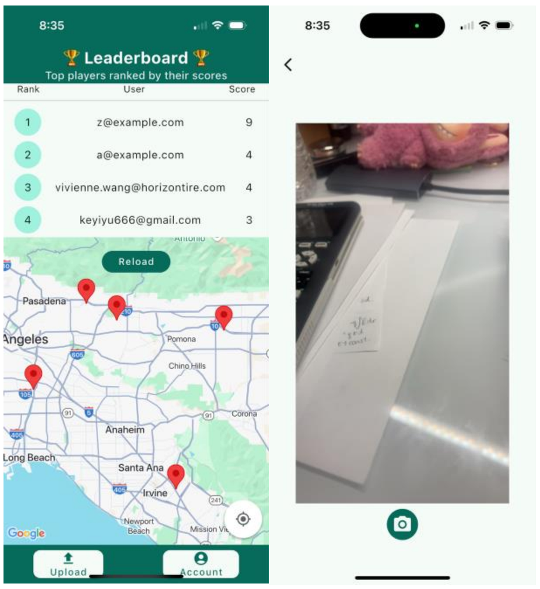


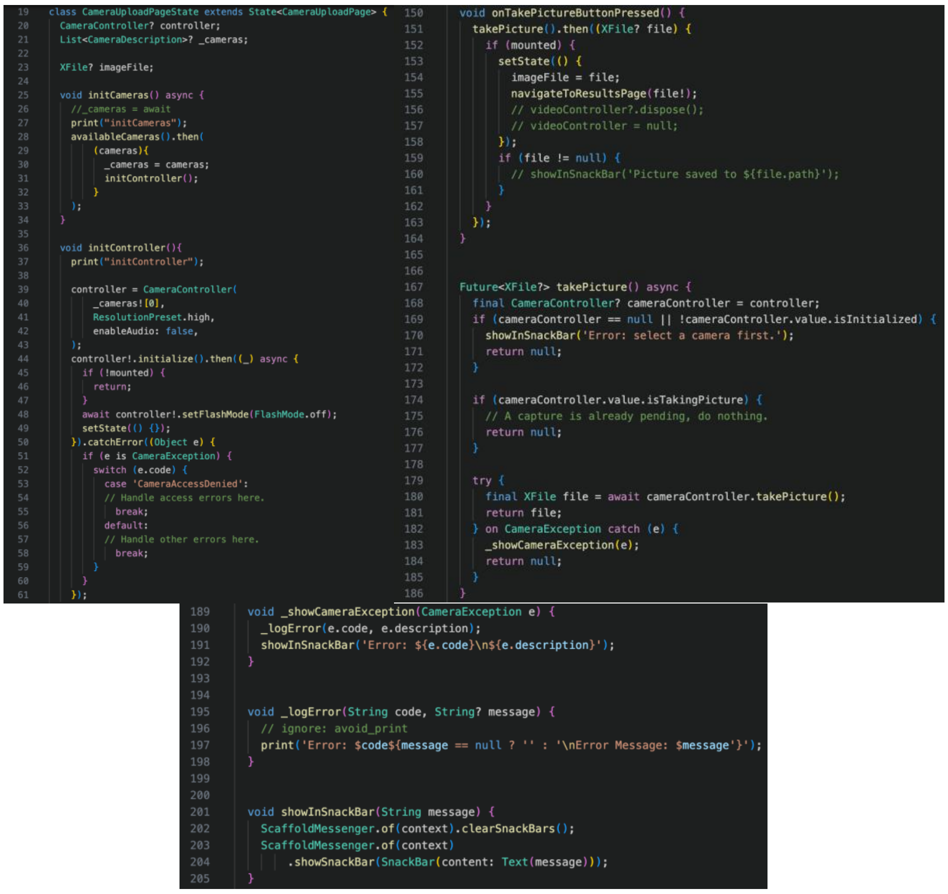
Figure 1. Overview of the solution

The mobile application component is designed to foster active participation in environmental cleanup efforts by enabling users to capture images of trash sites and upload them along with geolocation data. Through an intuitive interface built with Flutter, the application leverages device camera functionality to allow users to take clear pictures and automatically tag them with precise location information. Upon successfully uploading an image that evidences a completed cleanup task, the system rewards a point on the leaderboard, incorporating a gamification element intended to motivate ongoing engagement. The functionality is supported by a suite of robust backend services. Firebase is employed for concurrent data synchronization and secure storage of uploaded images and associated location data, ensuring that the system remains responsive and reliable. Firebase Authentication manages user profiles and maintains data integrity, while geolocation services accurately mark and track reported trash locations [8]. The mobile application's capability of collecting user generated data and maintaining user enthusiasm is a critical link between manual reporting and the overarching automated cleanup system. By incentivizing users with leaderboard points upon successful task completion, the design encourages widespread community involvement and fosters a competitive spirit. This approach not only enhances user engagement but also contributes to the overall effectiveness of the environmental cleanup strategy by ensuring that each report is both meaningful and verifiable.



Rank	User	Score
1	z@example.com	9
2	a@example.com	4
3	vivienne.wang@horizontire.com	4
4	keyiyu666@gmail.com	3

Figure 2. Screenshot of map



```dart
class CameraUploadPageState extends State<CameraUploadPage> {
  CameraController? controller;
  List<CameraDescription>? _cameras;

  XFile? imageFile;

  void initCameras() async {
    //_cameras = await
    print("initCameras");
    availableCameras().then(
        (cameras){
          _cameras = cameras;
          initController();
        }
    );
  }

  void initController(){
    print("initController");

    controller = CameraController(
        _cameras![0],
        ResolutionPreset.high,
        enableAudio: false,
    );
    controller!.initialize().then((_) async {
      if (!mounted) {
        return;
      }
      await controller!.setFlashMode(FlashMode.off);
      setState(() {});
    }).catchError((Object e) {
      if (e is CameraException) {
        switch (e.code) {
          case 'CameraAccessDenied':
          // Handle access errors here.
            break;
          default:
          // Handle other errors here.
            break;
        }
      }
    });
```

```dart
void onTakePictureButtonPressed() {
  takePicture().then((XFile? file) {
    if (mounted) {
      setState(() {
        imageFile = file;
        navigateToResultsPage(file!);
        // videoController?.dispose();
        // videoController = null;
      });
      if (file != null) {
        // showInSnackBar('Picture saved to ${file.path}');
      }
    }
  });
}


Future<XFile?> takePicture() async {
  final CameraController? cameraController = controller;
  if (cameraController == null || !cameraController.value.isInitialized) {
    showInSnackBar('Error: select a camera first.');
    return null;
  }

  if (cameraController.value.isTakingPicture) {
    // A capture is already pending, do nothing.
    return null;
  }

  try {
    final XFile file = await cameraController.takePicture();
    return file;
  } on CameraException catch (e) {
    _showCameraException(e);
    return null;
  }
}
```

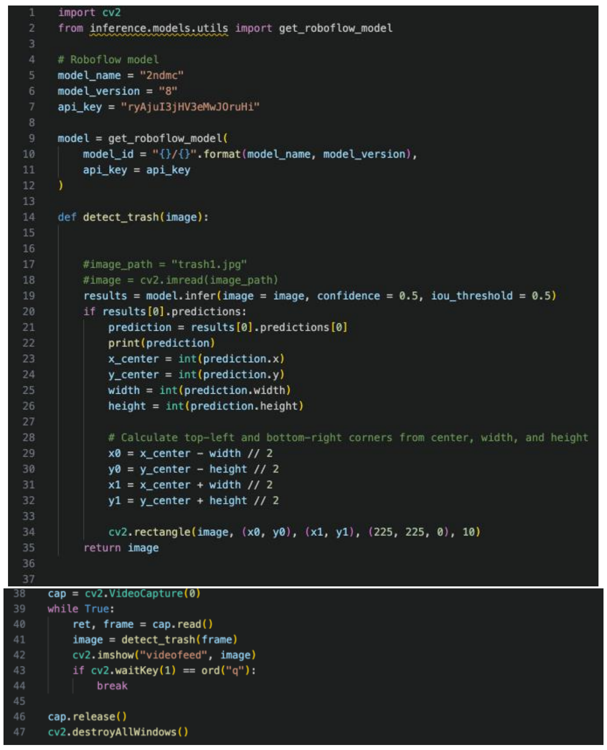
```dart
void _showCameraException(CameraException e) {
  _logError(e.code, e.description);
  showInSnackBar('Error: ${e.code}\n${e.description}');
}


void _logError(String code, String? message) {
  // ignore: avoid_print
  print('Error: $code${message == null ? '' : '\nError Message: $message'}');
}


void showInSnackBar(String message) {
  ScaffoldMessenger.of(context).clearSnackBars();
  ScaffoldMessenger.of(context)
      .showSnackBar(SnackBar(content: Text(message)));
}
```

Figure 3. Screenshot of code 1

The app allows detected trash sites to be uploaded with geolocation data. Volunteers can upload photos of trash for verification, and earn points and rewards. The Flutter frame work used creates a seamless interface where users can capture, and upload images, triggering functions that analyze the image and upload results to the Firebase Realtime Database.

The code defines a Flutter state class CameraUploadPageState, which manages camera functionalities, including initialization, image capture, and error handling. By utilizing the camera package to access available cameras and control the camera hardware, it initializes the camera list asynchronously and assigns the first available camera to a CameraController instance. The controller is then used to configure the camera with high resolution and disable audio before rendering the preview. The execution begins with the initState method, which calls initCamers to retrieve available cameras asynchronously. Once the camera list is obtained, initController initializes the CameraController to configure it with necessary settings, such as disabling the flash. The dispose method ensures proper cleanup by disposing of the camera controller when the widget is removed. Image capture is handled by onTakePictureButtonPressed, which calls takePicture to capture an image and store it in imageFile. The method verifies whether the camera is ready before capturing an image and handles any exceptions that may occur during the process. This implementation does not directly communicate with a backend server, but it processes and manages captured images locally. The error handling methods _showCameraExceptions and _logError ensures that camera-related issues are logged and displayed using a SnackBar for user feedback. This structured approach enables efficient camera handling within a Flutter application while ensuring user friendly error management.

The drone hardware is the operational component responsible for physically detecting and removing trash. Equipped with an AI-powered camera system, the drone autonomously identifies and classifies debris in environments that are difficult or unsafe for humans to access, such as high mountains or large bodies of water. The drone relies on a Drone SDK (Software Development Kit) for integration with its hardware components, such as claw motors and imagery sensors. OpenCV—an open-source computer vision library—is utilized for image processing and real-time trash detection. Roboflow was used to train the AI model by providing annotated datasets for object recognition. The AI model relies on Convolutional Neural Networks (CNNs) to process visual data and detect trash [9]. CNNs are particularly effective for image recognition tasks as they analyze spatial hierarchies in images. The drone also employs Machine Learning (ML) for continuous improvement by learning from real-world mission data stored in the centralized database. When assigned a mission, the drone navigates to the specified location using geolocation data. Its camera scans the environment, and the AI model identifies objects classified as litter. Once detected, the drone uses its mechanical arm to collect the trash or flags it for manual retrieval if it cannot be removed autonomously.



```python
import cv2
from inference.models.utils import get_roboflow_model

# Roboflow model
model_name = "2ndmc"
model_version = "8"
api_key = "ryAjuI3jHV3eMwJOruHi"

model = get_roboflow_model(
    model_id = "{}/{}".format(model_name, model_version),
    api_key = api_key
)

def detect_trash(image):


    #image_path = "trash1.jpg"
    #image = cv2.imread(image_path)
    results = model.infer(image = image, confidence = 0.5, iou_threshold = 0.5)
    if results[0].predictions:
        prediction = results[0].predictions[0]
        print(prediction)
        x_center = int(prediction.x)
        y_center = int(prediction.y)
        width = int(prediction.width)
        height = int(prediction.height)

        # Calculate top-left and bottom-right corners from center, width, and height
        x0 = x_center - width // 2
        y0 = y_center - height // 2
        x1 = x_center + width // 2
        y1 = y_center + height // 2

        cv2.rectangle(image, (x0, y0), (x1, y1), (225, 225, 0), 10)
    return image


cap = cv2.VideoCapture(0)
while True:
    ret, frame = cap.read()
    image = detect_trash(frame)
    cv2.imshow("videofeed", image)
    if cv2.waitKey(1) == ord("q"):
        break

cap.release()
cv2.destroyAllWindows()
```

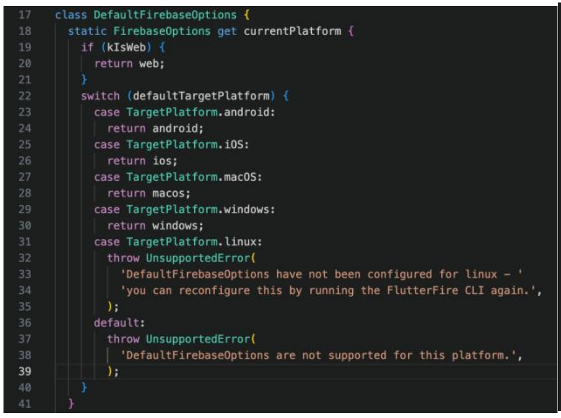
Figure 4. Screenshot of code 2

The provided code implements a concurrent trash detection system employing Roboflow–providing computer vision models—and OpenCV—a Computer Vision library that processes images and videos. The program operates by capturing live video feed from a camera mounted on the drone and detecting trash objects within each frame using a pre-trained trash detection model from Roboflow. The model is initialized using a specific model name, version, and API key, facilitating access to the server for inference tasks. The primary function, detect_trash, processes each video frame to infer the presence of trash, draw bounding boxes around detected objects, and display the processed output.

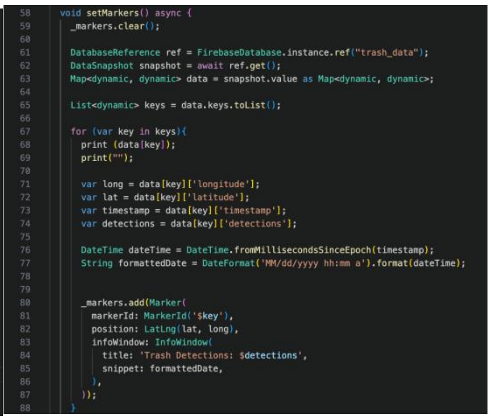
The execution begins with importing necessary libraries and initializing the Roboflow model through the get_roboflow_model function. This function communicates with the Roboflow backend, fetching the specified model and version using the provided API key. The detect_trash function accepts an image frame, where it calls the model's infer method to perform object detection with configurable parameters, such as confidence and IoU thresholds. If predictions are available, the coordinates of the detected objects are extracted, including the center position, width, and height. These values are used to calculate the top-left and bottom-right corners of the bounding box, which is then drawn on the frame using OpenCV's rectangle method. The modified frame is subsequently returned.

The program continuously captures video frames using OpenCV's VideoCapture class, passing each frame to the detect_trash function for processing. The live feed is displayed through OpenCV's imshow function, which is used to display an image in a window. The server's role is to provide inference capabilities, delivering predictions for each frame sent from the client application. This loop persists until a termination signal is received, at which point resources are released and the application exits gracefully. The implementation is a seamless integration of a cloud-based object detection model with local image processing to achieve efficient and interactive trash detection.

Firebase facilitates communication between the app, TrashTagger, and the physical drone by acting as an intermediary database that stores and synchronizes commands in real time. The app sends task related data, such as pickup requests, to Firebase Realtime Database. This data includes coordinates, timestamps, and task status updates. On the drone's side, it is assigned to provide detection information and complete incoming commands. Within the drone, a Python script running on a Raspberry Pi continuously listens for new data entries in Firebase. The script fetches task updates using Firebase's event listeners, which trigger actions when new tasks are assigned. Once the drone receives a pickup request, it processes the location data, navigates to the specified coordinates, and deploys its claw mechanism for retrieval. After the task is completed, the drone updates Firebase with its status—marking the trash as picked up and providing confirmation details such as completion time and final drop-off location. This updated information is then reflected in the app, allowing users to track task progress and ensuring seamless communication between volunteers, the app, and the drone. Additionally, a dedicated component is responsible for storing and mapping geolocation data for trash sites. The class responsible for default Firebase options configures the connection based on the running platform (web, Android, iOS, macOS, and Windows), while a specific method retrieves data from Firebase under the "trash_data" reference and populates the map with markers generated from latitude, longitude, timestamps, and detection counts. This component leverages Firebase's concurrent sychronization capabilities to maintain a central repository of mapping information that supports both automated drone operation and volunteer efforts.



```dart
class DefaultFirebaseOptions {
  static FirebaseOptions get currentPlatform {
    if (kIsWeb) {
      return web;
    }
    switch (defaultTargetPlatform) {
      case TargetPlatform.android:
        return android;
      case TargetPlatform.iOS:
        return ios;
      case TargetPlatform.macOS:
        return macos;
      case TargetPlatform.windows:
        return windows;
      case TargetPlatform.linux:
        throw UnsupportedError(
          'DefaultFirebaseOptions have not been configured for linux - '
          'you can reconfigure this by running the FlutterFire CLI again.',
        );
      default:
        throw UnsupportedError(
          'DefaultFirebaseOptions are not supported for this platform.',
        );
    }
  }
```



```dart
void setMarkers() async {
  _markers.clear();

  DatabaseReference ref = FirebaseDatabase.instance.ref("trash_data");
  DataSnapshot snapshot = await ref.get();
  Map<dynamic, dynamic> data = snapshot.value as Map<dynamic, dynamic>;

  List<dynamic> keys = data.keys.toList();

  for (var key in keys){
    print (data[key]);
    print("");

    var long = data[key]['longitude'];
    var lat = data[key]['latitude'];
    var timestamp = data[key]['timestamp'];
    var detections = data[key]['detections'];

    DateTime dateTime = DateTime.fromMillisecondsSinceEpoch(timestamp);
    String formattedDate = DateFormat('MM/dd/yyyy hh:mm a').format(dateTime);


    _markers.add(Marker(
      markerId: MarkerId('$key'),
      position: LatLng(lat, long),
      infoWindow: InfoWindow(
        title: 'Trash Detections: $detections',
        snippet: formattedDate,
      ),
    ));
  }
```

Figure 5. Screenshot of code 3

The code snippet consists of two primary sections that operate at distinct stages within the application. The first section defines a class that returns platform-specific Firebase configuration options. It checks whether the application is running on the web or a specific mobile/desktop platform (such as Android, iOS, macOS, or Windows) and provides the corresponding Firebase options; unsupported platforms (e.g., Linux) trigger an error message. The second section comprises a method responsible for updating map markers based on geolocation data stored in the Firebase Realtime Database [10]. This method begins by clearing any existing markers before establishing a reference to the "trash_data" node. It then retrieves the data snapshot, extracts relevant details (longitude, latitude, timestamp, and detection counts), converts timestamps into a readable format, and creates markers that include an information window detailing trash detection metrics. These markers are subsequently added to the map to provide users with an up-to-date visual representation of reported trash locations. The code is executed when the application requires a map refresh, ensuring synchronization between the backend server's data and the user interface, with Firebase acting as the critical backend component that stores and disseminates geolocation information.

3. EXPERIMENT

4.1. Experiment 1

A potential blind spot in the program is the AI's accuracy in detecting trash in cluttered or low-light environments. Ensuring accuracy is critical to prevent missed debris and optimize cleanup efficiency.
To test the AI's accuracy in detecting trash in cluttered or low-light environments, we will set up controlled experiments in various simulated conditions. The drone will operate in environments with varying levels of lighting and background complexity, including urban areas, forests, and beaches. Trash objects of different sizes, colors, and materials will be placed in these environments. Control data will be sourced from a pre-verified dataset of trash images, ensuring consistency in evaluation. By comparing the AI's detection results with the known locations of trash, we can quantify its accuracy and identify areas for improvement in challenging conditions.

4.2. Experiment 2

A potential blind spot is the drone's mechanical arm or gravity claw's ability to securely pick up various trash shapes and sizes. This functionality is crucial for effective cleanup missions.

To test the mechanical arm or gravity claw's reliability, we will simulate cleanup scenarios with diverse types of trash, including lightweight items like plastic bottles and heavier objects like metal cans. These objects will vary in size, shape, and material to mimic real-world conditions. The experiment will involve the drone attempting to pick up and store each object while being evaluated for grip strength, precision, and stability during flight. Control data will be sourced from standardized weight and dimension specifications for common trash items. This setup ensures consistent testing and identifies limitations in the arm's design or operational efficiency.

4. RELATED WORK

A scholarly approach to environmental cleanup involves deploying drones equipped with AI for waste detection and collection [11]. For instance, a study by Wang et al. (2024) introduced a framework utilizing drones and computer vision to detect, classify, and map marine debris without human-labeled data. This method achieved competitive performance in detection (0.69 mean IoU) and classification (0.74 F1 score) across seven debris object classes. However, its limitations include reliance on aerial imagery, which may miss submerged or obscured debris, and potential challenges in deploying drones in adverse weather conditions. Additionally, the framework focuses primarily on detection and mapping, without addressing the physical collection of waste. In contrast, our project enhances this approach by integrating a mechanical system capable of both detecting and physically collecting trash, including in challenging terrains like high mountains or large bodies of water. Furthermore, our volunteer-based app allows users to mark locations of trash, facilitating coordinated cleanup missions and community engagement, thereby providing a more comprehensive and scalable solution to environmental pollution.

Another study by Obaid et al. (2018) introduced a drone agent designed to support environmental cleanliness by detecting waste items and persuading nearby individuals to dispose of them properly [12]. The drone employed visual and audio cues to guide users to the nearest trash bin. While this approach leverages human participation, its effectiveness is limited by user compliance and the drone's ability to engage individuals in various settings. Our project addresses these limitations by automating the detection and collection process, reducing reliance on human intervention, and ensuring consistent cleanup efforts regardless of public participation.

Additionally, a review by Singh et al. (2023) explored the application of artificial intelligence in waste management within smart cities, highlighting the use of AI in waste-to-energy processes, smart bins, and waste-sorting robots [13]. While these technologies improve waste processing efficiency, they often operate within controlled environments and may not effectively address litter in open or remote areas. Our project extends the reach of AI-powered waste management to diverse and challenging terrains, utilizing drones to detect and collect litter in both urban and natural landscapes, thereby bridging the gap left by stationary or infrastructure-dependent solutions.

6. CONCLUSIONS

One notable limitation of our project is the reliance on consistent weather conditions and stable connectivity for drone operation. Adverse weather, such as high winds or heavy rain, can impede the drone’s ability to navigate and collect trash effectively [14]. To address this, future iterations could include robust drone designs capable of withstanding harsh conditions, as well as enhanced communication systems for better connectivity in remote areas.

Another limitation is the current AI model's dependence on a predefined dataset for trash detection, which may not account for all types of waste in various environments. Expanding the dataset to include diverse waste samples and employing real-time machine learning techniques could enhance detection accuracy.

Finally, scalability is a challenge, particularly in managing multiple drones simultaneously and coordinating volunteer efforts [15]. Implementing cloud-based task management and optimizing resource allocation algorithms would help improve the system’s efficiency and scalability, ensuring widespread adoption and long-term impact.

This project demonstrates the potential of integrating advanced drone technology, artificial intelligence, and community engagement to tackle environmental pollution. By addressing current limitations and iterating on its design, the system can become a scalable and transformative solution, significantly contributing to a cleaner, healthier planet for future generations.

REFERENCES

[1] Xie, Chao, Yan Sun, and Hong Luo. "Secured data storage scheme based on block chain for agricultural products tracking." 2017 3rd International Conference on Big Data Computing and Communications (BIGCOM). IEEE, 2017.

[2] Clayton, Susan D. "Environment, identity, and response to polluted landscapes." Sustainability 13.16 (2021): 9422.

[3] Plourde, Charles G. "A model of waste accumulation and disposal." The Canadian Journal of Economics/Revue canadienne d'Economique 5.1 (1972): 119-125.

[4] Wang, Chunhui, Jian Zhao, and Baoshan Xing. "Environmental source, fate, and toxicity of microplastics." Journal of hazardous materials 407 (2021): 124357.

[5] Ollander, Simon, et al. "A comparison of wearable and stationary sensors for stress detection." 2016 IEEE International Conference on systems, man, and Cybernetics (SMC). IEEE, 2016.

[6] Milton, Richard, and Anthony Steed. "Correcting GPS readings from a tracked mobile sensor." International Symposium on Location-and Context-Awareness. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005.

[7] Saito, Takuya, and Kenichi Mase. "DronePilot. NET Development: AR. Drone SDK Supporting Native and Managed Code." 2013 International Conference on Advanced Computer Science Applications and Technologies. IEEE, 2013.

[8] Moroney, Laurence, and Laurence Moroney. "Using authentication in firebase." The Definitive Guide to Firebase: Build Android Apps on Google's Mobile Platform (2017): 25-50.

[9] Li, Zewen, et al. "A survey of convolutional neural networks: analysis, applications, and prospects." IEEE transactions on neural networks and learning systems 33.12 (2021): 6999-7019.

[10] Moroney, Laurence, and Laurence Moroney. "The firebase realtime database." The Definitive Guide to Firebase: Build Android Apps on Google's Mobile Platform (2017): 51-71.

[11] Wang, Raymond, et al. "Designing A Sustainable Marine Debris Clean-up Framework without Human Labels." Proceedings of the 7th ACM SIGCAS/SIGCHI Conference on Computing and Sustainable Societies. 2024.

[12] Obaid, Mohammad, et al. "A drone agent to support a clean environment." Proceedings of the 3rd International Conference on Human-Agent Interaction. 2015.

[13] Fang, Bingbing, et al. "Artificial intelligence for waste management in smart cities: a review." Environmental Chemistry Letters 21.4 (2023): 1959-1989.

[14] Wayman, Chloe, and Helge Niemann. "The fate of plastic in the ocean environment–a minireview." Environmental Science: Processes & Impacts 23.2 (2021): 198-212.

[15] Schmid, Chiara, Luca Cozzarini, and Elena Zambello. "A critical review on marine litter in the Adriatic Sea: Focus on plastic pollution." Environmental Pollution 273 (2021): 116430.