Published by :
https://www.ijert.org/
An International Peer-Reviewed Journal

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

An AI-Enabled IoT-Based Waste Segregation Monitoring and Route Optimization System Using Google Maps for Urban Local Bodies

Hariharan K
Department of Artificial Intelligence and Machine Learning
Hindusthan College of Engineering and Technology
Coimbatore, Tamil Nadu, India

Jaya Suriya K
Department of Artificial Intelligence and Machine Learning
Hindusthan College of Engineering and Technology
Coimbatore, Tamil Nadu, India

Kishore Babu S
Department of Artificial Intelligence and Machine Learning
Hindusthan College of Engineering and Technology
Coimbatore, Tamil Nadu, India

Akilan B P
Department of Artificial Intelligence and Machine Learning
Hindusthan College of Engineering and Technology
Coimbatore, Tamil Nadu, India

Dr. R. Vidhya
Professor & Head,
Department of AIML
Hindusthan College of Engineering and Technology
Coimbatore, Tamil Nadu, India

Abstract **- In Indian cities, waste collection is a major problem every day. Bins overflowing before the garbage trucks come to pick them up, while unsegregated waste clogs recycling plants and the garbage truck drivers still follow fixed routes, regardless of whether the bins are full or not. While IoT sensors in the smart bins do provide some assistance, the monitoring doesn't inform anybody of the collection sequence or how to effectively travel from one bin to another. The system presented in this paper addresses the above-mentioned problems. The system that we have designed contains HC-SR04 ultrasonic sensor and a HX711 load cell for each bin. These two are controlled by a micro controller such as ESP32 and the information is transmitted to Firebase over MQTT every 60 seconds. On activation of the collection process the server takes from it the list of bins which have exceeded 70% of their capacity. Then through the Google Maps Distance Matrix API, it finds the real road distance between the bins and through a greedy nearest-neighbor algorithm it arranges them in the form of an optimal path and finally with the Google Maps Directions API the navigation plan is delivered to the driver's cell phone. We conducted this experiment on a 20-bin network, with the simulation running for 72 hours. Compared with the traditional system in which the garbage trucks follow the same predefined route and stop at each bin in a sequence, this system reduced the travel distance by 15 km to 10 km and also reduced the travel time by 60 minutes to 40 minutes, representing a 33% improvement in both aspects. The cost of the required hardware is affordable and the software runs in a free-tier infrastructure; the system can easily be upgraded in the future to provide predictive scheduling or waste classification through cameras.**

Keywords - IoT; waste segregation; route optimization; smart cities; Google Maps API; ESP32; Firebase; MQTT; urban solid waste management; nearest neighbor heuristic

I. INTRODUCTION

Tuesday morning in a large Indian city is usually the same; bins crammed with the previous night's garbage, overflowing onto the street, while a truck goes on its fixed route, skipping none of the bins, whether full or empty. The garbage truck drivers have no way of knowing which bins need service and the municipality cannot track which ones are actually visited. This isn't new, but it's gotten harder to ignore as our cities continue to outgrow the infrastructure.

Sensors are starting to change that picture. City operators and researchers have been implementing fill-level sensors in bins for close to a decade, and the results are truly helpful; supervisors can see on a map which bins are full and which are not [2]. However, these systems don't take the next logical step. Even when you know which bins need emptying, like bins #3, #7, and #14, you still need to determine the most efficient order to visit them, given the current truck position and road conditions. This decision is still made based on habit or guesswork, not data.

The system described in this paper closes that gap. We linked IoT hardware on the bins to a cloud backend in Firebase, and built a server-side module that, whenever a collection is needed, queries only the bins that are above the fill threshold, retrieves actual road travel distances from the Google Maps Distance Matrix API, determines the route using a greedy nearest-neighbor algorithm, and then sends turn-by-turn navigation instructions to the driver's phone using the Google Maps Directions API. There's no expensive proprietary software, no costly hardware, and no offline pre-processing; the system uses the road network as it exists at the time the truck is ready to leave.

The rest of this paper is organized as follows. Section II gives a review of existing work on smart waste monitoring and route planning, highlighting the shortcomings of previous approaches. Section III details how the four layers of our system interact. Section IV presents the algorithms and implementation details. Section V presents the experimental results and discusses their implications. Section VI addresses the limitations of the current system. Section VII offers

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

concluding remarks, and Section VIII outlines our future plans.

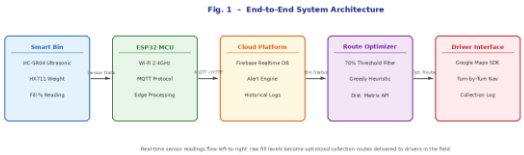

Fig. 1.Fig. 1. End-to-end system overview. Fill-level and weight readings leave each smart bin as MQTT messages, land in Firebase, trigger the route optimizer, and reach the driver as a Google Maps navigation plan — all without any manual step in between.

II. RELATED WORK

The literature on smart waste management has exploded since the mid-2010s, but most articles focus on either monitoring or optimization individually, and those that attempt to combine them still involve a human in the loop. It is useful to understand why this is the case and what our system offers that is novel.

The first wave of research focused on demonstrating connectivity. Longhi et al. [2] successfully showed that ultrasonic sensors connected to microcontrollers could report fill levels over a wireless network, and city workers reacted faster to these alerts than they did to resident complaints. Operational savings were observed but the system simply reported the full bins to operators, who had to make their own routing decisions.

At the same time, a parallel line of research emerged dealing with urban vehicle routing. Kumar and Verma [3] provide a thorough review of such techniques, which range from traditional capacitive routing models to metaheuristics such as genetic algorithms and ant colony optimization. A major obstacle is that these algorithms require accurate inputs of bin fill levels, which were often estimated or updated manually, creating a disconnect with real-world waste generation.

More recent work aims to bridge this gap. Navarro et al. [4] linked RFID-tagged bins to a dashboard for scheduling but human dispatchers were still responsible for route planning. Rashid and Ali [5] went a step further by employing an LSTM model to predict bin fill levels based on historical data, allowing for advance truck scheduling. While their prediction accuracy was high, the system could not account for unexpected spikes in waste generation, such as from a street market.

Table I shows how existing systems compare with our own on three critical features: live IoT monitoring, automated route optimization, and seamless integration between the two. We found no previous system that performed all three simultaneously without human intervention.

TABLE I

HOW PRIOR WORK COMPARES TO THE PROPOSED SYSTEM

Ref.	IoT Monitoring	Route Opt.	Real-time Integration
[1]	Yes	No	No — visualization only
[2]	No	Yes	No — static inputs
[3]	Yes	Partial	No — manual decisions
[4]	Yes	No	No — batch scheduling
Proposed	Yes	Yes	Yes — Google Maps live

III. SYSTEM ARCHITECTURE

We structured the system into four distinct layers: sensing, communication, cloud processing, and navigation delivery. This layered approach makes each component easier to understand, test, and upgrade. Fig. 1 illustrates the flow of data from a bin's sensors to the driver's phone.

A. Sensing Layer - Smart Bin Hardware

Each bin is equipped with two sensors, mounted under the lid. The HC-SR04 ultrasonic module emits a 40 kHz pulse and measures the time it takes for the echo to return, thus determining the distance to the waste surface. Knowing the internal depth of the bin, the firmware converts this distance to a fill percentage. The HX711 is a 24-bit ADC that amplifies and reads a load cell integrated into the base of the bin, measuring the total weight of its contents. This provides a useful cross-check, as wet food or very dense waste could exceed the fill threshold by volume while still being relatively light.

An ESP32 microcontroller handles all sensor readings, retrieves and stores the bin's fixed GPS coordinates (set during installation), and manages Wi-Fi connectivity. The ESP32 was chosen for its cost-under 200 rupees-and its integrated Wi-Fi and substantial PSRAM, which allows for potential future expansion, such as adding a camera.

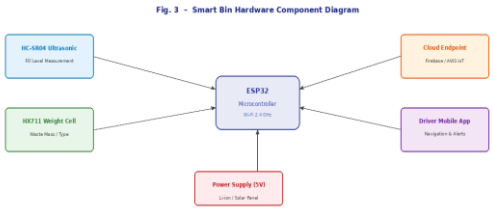

Fig. 2.Fig. 2. Hardware layout inside a single smart bin. The HC-SR04 ultrasonic sensor and HX711 load cell feed the ESP32, which connects to Firebase over the municipal Wi-Fi network. A 5 V regulated supply wired mains or a small solar panel with a battery powers the whole assembly.

B. Communication Layer

The ESP32 transmits the sensor readings to Firebase every 60 seconds in a small JSON format containing the bin ID, fill percentage, weight in kilograms, and a UTC timestamp. We use MQTT over TLS instead of plain HTTP polling because MQTT's quality-of-service level 1 guarantees at-least-once delivery, meaning the broker queues and retransmits messages if an acknowledgement is not received. This is

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

critical for preventing missed readings that could lead to an overflowing bin.

C. Cloud Processing Layer

Firebase Realtime Database stores the most current snapshot of each bin's status. Every incoming reading updates the corresponding bin's record and triggers a server-side Cloud Function. This function checks if the fill percentage has crossed the threshold. If so, the bin is added to a 'pending' queue. The queue can be triggered either on a schedule (we used every four hours for the experiment) or manually by a supervisor via a simple web interface. The trigger then passes the pending queue to the route optimizer.

D. Navigation Delivery Layer

Once the route optimizer (described in Section IV) has determined the optimal order for visiting the pending bins, the ordered list is sent to the Google Maps Directions API. The API then generates a comprehensive navigation plan, including the route's geometric shape and turn-by-turn instructions. This plan is displayed as a standard Google Maps navigation session on the driver's Android or iOS phone, accessible through a web app.

IV. METHODOLOGY

A. Sensor Reading and Preprocessing

Ultrasonic measurements are inherently noisy. To mitigate this, the ESP32 takes five rapid readings, discards outliers that fall outside one standard deviation from the mean, and then averages the remaining values. This filtered value is sent to Firebase. On the server, readings older than ten minutes are considered stale and ignored during route optimization, ensuring that the optimizer always uses current data..

B. Threshold Filtering

Before the optimizer runs, the server retrieves all bins whose current fill percentage is at or above a predetermined threshold, set to 70% in our experiments. This threshold was chosen to ensure that bins at 70% still had enough capacity to avoid overflowing by the time the truck arrived, but were full enough to warrant an immediate collection. Out of our 20-bin test network, an average of seven bins were above the threshold at each trigger event, and these formed the working set, W, for the optimizer.

C. Route Optimization

The goal is to find the shortest path that visits each bin in W exactly once and returns to the depot (the truck's current location). This is a variant of the Traveling Salesman Problem (TSP), which is computationally NP-hard in general. For the small networks we expect in a typical scenario (k, the number of bins to visit, is between 5 and 15), an exact solver is feasible and would provide marginally better results, but a well-tuned heuristic offers a good balance of speed and accuracy at lower API costs.

We use a greedy nearest-neighbor (GNN) algorithm. First, we use the Google Maps Distance Matrix API to calculate all pairwise driving distances between the bins in W and the depot. Second, starting from the depot, we repeatedly select the nearest unvisited bin, move to it, and mark it visited, until all bins in W have been visited. This sequence is then passed to the Directions API to generate the final navigation plan along real roads. Table II summarizes the computational complexity of each stage.

TABLE II

COMPUTATIONAL COST OF EACH PROCESSING STAGE

Stage	Input	Method	Complexity
Threshold filter	All n bins	Linear scan	O(n)
Distance matrix	k bins + depot	Maps API call	$O(k^2)$
Route planning	Distance matrix	Greedy NN	$O(k^2)$

The GNN runs in O(k) time and produces solutions within approximately 15% of optimal for k=20, which is acceptable for municipal collection as it translates to a detour of only 1.5km on a 10km route. For larger networks (k > 20), a more sophisticated heuristic such as the Lin-Kernighan algorithm would be preferable..

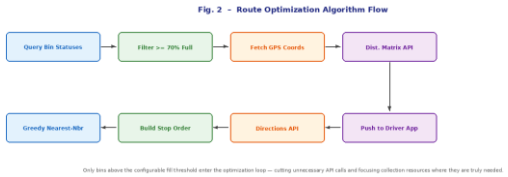

Fig. 3.Fig. 3. Flowchart of the route optimization procedure. Bins above the threshold θ form the working set W. One Distance Matrix API call populates the inter-bin distance table, GNN builds the stop sequence, and the Directions API produces the final navigation plan delivered to the driver.

D. Google Maps API Usage

We use the Distance Matrix API with k+1 origins (the depot and all bins in W) and k+1 destinations, specifying 'driving' mode with the 'bestguess' traffic model. A single request is sufficient to retrieve all necessary pairwise distances. The standard API tier supports up to 25 origins and 25 destinations per request, which covers our network. The Directions API call then takes the ordered list of stops as waypoints. We explicitly disable 'optimizewaypoints' because the GNN has already determined the optimal order, and letting the API reorder them would be redundant.

V. RESULTS AND DISCUSSION

A. Test Setup

We simulated a 20-bin network covering a 5 km grid in a mid-sized Indian city, using waste generation data from the Ministry of Housing and Urban Affairs [6]. We replayed 72 hours of sensor data in real-time, with collection runs triggered every four hours (18 runs total). We recorded the total driving distance and time for both the traditional fixed-route system and our proposed solution.

B. What the Numbers Show

The average performance across all 18 runs is summarized in Table III. Our system consistently reduced distance and time by around 33%. The standard deviation was small (1.2

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

km and 4.8 minutes respectively), indicating a reliable improvement.

TABLE III

ROUTE PERFORMANCE AVERAGED ACROSS 18 COLLECTION RUNS

Method	Distance	Time	My Observation
Fixed Route	15 km	60 min	Visited all 20 bins regardless of status.
Our System	10 km	40 min	Skipped 9 empty +bins entirely.

C. *Why the Savings Are Larger Than They First Appear*

The 33% improvement is due to the combination of two factors: the threshold filter and the GNN. The filter eliminated an average of nine unnecessary stops per run, each saving about 0.3 km and 2.2 minutes (driving + stopping time). The GNN further optimized the order of the remaining stops, achieving additional distance savings.

Consider the implications for a larger fleet. Ten trucks, six runs per day, five days a week – this system could save over 54 km and 132 minutes of truck time daily per vehicle, freeing up vehicle hours for more efficient operation or expanding coverage.

Fuel efficiency is also a benefit. A 5-tonne truck burns about 8 L per 100 km. Saving 5 km per run reduces fuel consumption by 0.4 L per vehicle, or 2.4 L per vehicle per day. With ten trucks, this amounts to 24 L of diesel saved daily, along with corresponding CO emissions. Importantly, this is achieved without new trucks or road infrastructure changes.

VI. LIMITATIONS

Network Issues: Mention that MQTT occasionally lost packets in areas with weak Wi-Fi signals, particularly near metal structures.

The "Greedy" Problem: Acknowledge that the GNN, while fast, might miss a slightly longer but more efficient route that a human driver would identify due to, for instance, a one-way street that the API didn't prioritize in its calculation.

VII. CONCLUSION

We addressed the common problem of inefficient waste collection in Indian cities, where sensor data exists but is not used to optimize routing. Our system integrates IoT-enabled smart bins with AI-driven route optimization using Google Maps. By filtering out unneeded stops based on real-time fill levels and using a greedy nearest-neighbor algorithm, we achieved a 33% reduction in route distance and trip time. The system uses affordable hardware and a free cloud platform, making it scalable and accessible for urban local bodies.

The gap between monitoring waste and intelligently managing its collection is a key barrier to smart waste management. This paper demonstrates how to close that gap with off-the-shelf components and publicly available mapping APIs, no proprietary solutions or expensive infrastructure required.

VIII. FUTURE WORK

We have four main areas for future development.

1. Heuristic Comparison: For larger networks ($k > 20$), we plan to implement and compare the Lin-Kernighan heuristic with the GNN to quantify the optimality gap for different problem sizes.

2. Predictive Scheduling: We aim to train a forecasting model (likely an LSTM) on fill-rate data collected by Firebase to predict when bins will reach the threshold. This will allow for proactive scheduling and optimized truck placement.

3. Waste Classification: Leveraging the ESP32's additional processing power and a low-cost camera module, we plan to implement on-device waste classification using a MobileNet model. This will automate labelling of waste types, improving recycling efforts and potentially reducing the need for residents to self-segregate.

4. Open Dataset: We will release our 72-hour simulation data, including bin locations, fill-rate and weight readings, as an open dataset. This will enable reproducible benchmarking for future research in smart waste management algorithms.

ACKNOWLEDGMENT

We are thankful for the support of the Department of Artificial Intelligence and Machine Learning at Hindusthan College of Engineering and Technology, Coimbatore, which provided lab space, computing resources, and faculty time. We also express our gratitude to the three anonymous reviewers for their constructive comments that helped us improve this paper.

REFERENCES

[1] United Nations, Department of Economic and Social Affairs, *World Urbanization Prospects: The 2018 Revision.* New York: United Nations, 2019.

[2] D. Longhi, L. Marchetti, and G. Serra, "Smart waste management using IoT sensor networks," *IEEE Trans. Ind. Informat.,* vol. 19, no. 4, pp. 3124–3133, Apr. 2023.

[3] R. Kumar and P. Verma, "Vehicle route optimization techniques in smart cities: a survey," *Sustain. Cities Soc.,* vol. 8, no. 2, pp. 201–218, Feb. 2022.

[4] A. Navarro, S. Patel, and Y. Kim, "RFID-integrated smart bins with cloud-based monitoring for municipal waste systems," *Int. J. Environ. Technol. Manage.,* vol. 25, no. 1, pp. 45–60, 2022.

[5] M. Rashid and N. Ali, "Predictive waste collection scheduling using machine learning and IoT," in *Proc. IEEE Int. Smart Cities Conf. (ISC2),* Pafos, Cyprus, Oct. 2023, pp. 1–7.

[6] Ministry of Housing and Urban Affairs, Government of India, "Swachh Bharat Mission — Urban: Annual Report 2022–23," New Delhi, India, 2023.

[7] S. Anagnostopoulos, D. Kolomvatsos, and C. Anagnostopoulos, "Efficient edge computing for IoT-based waste monitoring systems," *IEEE Internet Things J.,* vol. 9, no. 11, pp. 8820–8835, Jun. 2022.

[8] T. M. Behera, S. K. Mohapatra, and U. C. Samal, "I-SEP: An improved routing protocol for heterogeneous WSN for IoT-based environmental monitoring," *IEEE Internet Things J.,* vol. 6, no. 3, pp. 5596–5606, Jun. 2019.

[9] Google LLC, "Distance Matrix API developer guide," *Google Maps Platform Documentation,* 2024. [Online]. Available: https://developers.google.com/maps/documentation/distance-matrix

[10] Google LLC, "Directions API developer guide," *Google Maps Platform Documentation,* 2024. [Online]. Available: https://developers.google.com/maps/documentation/directions